



User Manual

The Vienna 5G Link Level Simulator v1.1

Institute of Telecommunications,
TU Wien

Authors

Stefan Pratschner, Bashar Tahir, Ronald Nissel, Ljiljana Marijanovic,
Mariam Mussbah, Kiril Kirev, Stefan Schwarz and Markus Rupp

Vienna, June 7, 2018



Institute of Telecommunications, TU Wien
Gusshausstrasse 25/389
A-1040 Vienna
Austria

web: <http://www.nt.tuwien.ac.at/>



The Vienna 5G Link Level Simulator is part of the Vienna Cellular Communications Simulators (VCCS) software suite. The simulator is currently available under a non-commercial, academic use license. For download and license information of the simulator, please refer to our **license agreement**.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 2 | Quick Start | 3 |
| 3 | Simulation Methodology | 4 |
| 4 | Simulation Examples and Scenarios | 7 |
| 4.1 | Simulation Scripts | 7 |
| 4.1.1 | Channel Coding with Short Block Length | 7 |
| 4.1.2 | Comparison of Coding Schemes | 8 |
| 4.1.3 | Comparison of Waveforms | 8 |
| 4.1.4 | LTE Comparison | 8 |
| 4.2 | Simulation Scenarios | 8 |
| 4.2.1 | Generic Scenario | 9 |
| 4.2.2 | LTE-Advanced | 9 |
| 4.2.3 | Multi-Link Simulation | 9 |
| 4.2.4 | Flexible Numerology | 12 |
| 4.2.5 | Non-Orthogonal Multiple Access | 13 |
| 5 | Comparison to LTE-A | 16 |
| 6 | Simulator Structure | 18 |
| 6.1 | Topology Configuration | 18 |
| 6.2 | Links Generation | 19 |
| 6.3 | The Link Object | 19 |
| 6.4 | Some Rules | 19 |
| 6.5 | General Simulation Parameters | 20 |
| 7 | Channel Models | 23 |
| 8 | Definition of SNR | 25 |
| 8.1 | Scheduling | 26 |
| 9 | Channel Coding | 28 |
| 9.1 | Block Length Calculation and Segmentation | 28 |
| 9.2 | Convolutional Code | 29 |
| 9.3 | Turbo Code | 29 |
| 9.4 | Low Density Parity Check (LDPC) Code | 29 |
| 9.5 | Polar Code | 30 |
| 9.6 | Interleaving and Rate Matching | 30 |

| | | |
|-----------|--|-----------|
| 9.7 | Object Usage | 31 |
| 10 | Modulation Waveforms | 33 |
| 10.1 | Orthogonal Frequency-Division Multiplexing | 33 |
| 10.2 | WOLA | 34 |
| 10.3 | Universal Filtered Multicarrier | 35 |
| 10.3.1 | Transmitter | 36 |
| 10.3.2 | Receiver | 36 |
| 10.4 | Filtered-OFDM | 36 |
| 10.5 | FBMC | 37 |
| 11 | Feedback | 39 |
| 11.1 | Feedback Calculation | 39 |
| 11.1.1 | PMI and RI | 39 |
| 11.1.2 | Channel Quality Indicator (CQI) | 39 |
| 11.2 | Object Usage | 40 |
| 11.2.1 | Scenario File | 40 |
| 11.2.2 | Object Generation | 40 |
| 12 | Non-Orthogonal Multiple Access (NOMA) | 43 |
| 13 | Releases and Changelog | 45 |

1 Introduction

Our research group has a long and successful history of developing and sharing open-source cellular communications simulators. The implementation of our work-horse of the past eight years, that is, the Vienna LTE Simulators, started back in 2009. Although, the evolution of this project was not straight forward from the beginning, in total three reliable simulators evolved. The system level simulator and the up- and downlink link level simulators attained quite some attention from academia as well as from industry. Over the years, many simulator versions, including new features according to the LTE standard and several bug fixes, were released. Many of these bugs were reported by an online community through our simulator forum. Today the Vienna LTE Simulators count more than 50 000 downloads in total. This historical development shows the need for a standard compliant reliable simulation tool for performance evaluation and comparison. We therefore extend our simulator suite and evolve to the next generation of mobile communication by introducing new 5G simulators.

In this user manual we will describe the general idea and scope, as well as implementation details and usage of the Vienna 5G Link Level Simulator. As an introduction, we explain the concept and functionality of the simulator in this document.

The Vienna 5G Link Level Simulator is the newest member of the family of Vienna Cellular Communications Simulators (VCCS). Although, as of this writing, there exists no definite 5G specification, the standardization process within 3rd Generation Partnership Project (3GPP) is ongoing and is already taking shape. Work on the first set of 5G standards is expected to start by the second half of 2017 within Long Term Evolution (LTE) Rel. 15. While it is not yet decided which physical layer methods will be standardized for 5G, there are several hot candidates for physical layer waveforms and channel coding schemes. Performance evaluation and comparison of these candidate physical layer methods is currently subject to scientific work and topic of many publications over the past years. Through our simulators, we intend to offer a unifying platform for performance evaluation as well as co-existence investigation of candidate 5G physical layer schemes. Since there exists no concrete specification yet, we provide great flexibility by supporting a broad range of simulation parameters. Thus, many different combinations of physical layer settings are comparable by our 5G link level simulator.

In general the purpose of link level simulations of communication systems is to evaluate the average performance of the physical layer transceiver architecture. Correspondingly, the focus of our 5G link level simulator is on point-to-point simulations. Nevertheless, there exists the abstract concept of

cells within our simulator. However, this is realized without implementing an underlying cellular geometry. There is neither a physical cell size nor a distance to the user being considered; the path-loss to a user is rather specified as an input parameter, leading to an average signal to noise ratio. The concept of a cell can be thought of a group of nodes, that is, one base station and several users are grouped within a cell.

To obtain an average system performance as simulation result, we perform Monte Carlo simulations and average over many random channel realizations. As a result, system performance in terms of throughput, Bit Error Ratio (BER) or Frame Error Ratio (FER), is calculated and plotted together with confidence intervals that indicate their statistical reliability.

In this current initial version, our 5G link level simulator supports both, up- and downlink simulations. The simulator includes parameter settings for Long Term Evolution-Advanced (LTE-A) compliant simulations and additionally further user defined settings for the simulation of future 5G cellular communications systems. While currently up- and downlink are implemented for Frequency Division Duplex (FDD) mode only, our simulator structure allows for future implementation of Device-to-Device (D2D) communications as well as a Time Division Duplex (TDD) frame structure. Further, for simulation of 5G communications systems, we offer high flexibility in choosing desired physical layer methods. Not only simulation parameters that were free to choose within the full LTE-A specification, such as channel model, bandwidth or receiver type, but also very basic parameters, such as sampling rate and frame duration, are adjustable in our simulator. To enable investigation of 5G physical layer candidate methods, we support features such as new PHY waveforms like filtered or windowed Orthogonal Frequency Division Multiplexing (OFDM), Filter Bank Multicarrier (FBMC) or Universal Filtered Multicarrier (UFMC), and different channel codes like Turbo coding, LDPC coding or Polar coding. All of these schemes support any combination of channel coding rate and Quadrature Amplitude Modulation (QAM) alphabet size, which results in many different Modulation and Coding Scheme (MCS). In addition, the employed physical layer schemes can be different for users of different cells such that their co-existence can be simulated.

2 Quick Start

The Vienna 5G Link Level Simulator may be used in two different ways. In the simplest case, a pre defined or user defined simulation scenario is simulated. Simulation scenarios are further described in Section 4.2. Alternatively, the implemented classes and functions may be re-used within another framework as explained in Section 4.1.

This quick start guide describes how to run a pre defined simulation scenario.

1. Open the script `main.m` in the simulator root directory.
2. Select a simulation scenario:

```
1 % select scenario
2 simulationScenario = 'genericScenario';
```

3. In case you want to specifically set simulation parameters or define your own simulation scenario, edit or create scenario files in the `/Scenarios` directory.
4. Depending on whether you want to simulate in parallel mode, change the loop over the sweep parameter to `for` or `parfor`, accordingly:

```
1 % loop over sweep parameter
2 for iSweep = 1:length(simParams.simulation.sweepValue)
    % this may be 'for' or 'parfor'
```

5. Run the `main.m` script. Plots will be shown as the simulation is finished. All results are stored in the `downlinkResults` and `uplinkResults` objects.

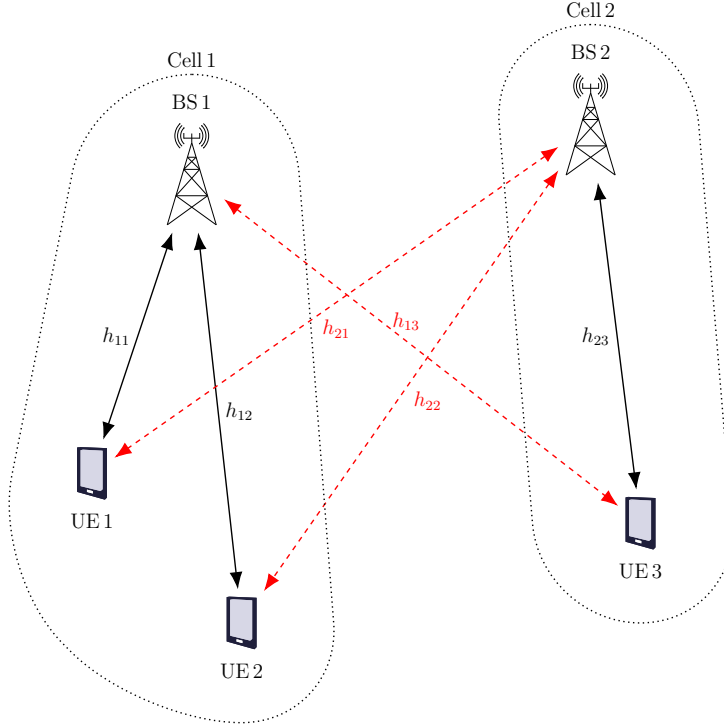


Figure 1: An exemplary network topology.

3 Simulation Methodology

With our 5G link level simulator we want to enable performance evaluation of future physical layer access schemes. For this we aim to maintain high flexibility of simulation scenarios and parameter settings. We not only support various waveforms and channel codes in general, but also allow these parameters to be different from cell to cell. This facilitates investigation of co-existence and interference of 4G and possible 5G physical layer schemes. Again, in our link level simulator there exists no underlying geometry. A cell should be thought of a collection of nodes (one base station and several users) rather than a physical area.

To further explain simulations with different waveforms in different cells, consider the exemplary cellular network topology as shown in Fig. 1. How such a topology is set up, is described in more detail in Section 6.1. Here we assume two cells, each with one base station. Users one and two are attached to base station one and therefore belong to cell one while user three is attached to base station two and belongs to cell two. The wireless channels $h_{i,j}$ are indicated with double arrows where the first subscript i indicates the base station and the second subscript j indicates the user. Desired or primary

channels are shown in solid black while interference or secondary channels are shown in dashed red.

While all nodes belonging to a cell must have the same waveform, channel code, total number of subcarriers¹ and number of symbols per frame (frame duration), these settings might be different for nodes of another cell. To enable discrete simulation of several nodes and many wireless channels, the sampling rate is a common parameter for all cells. Further, the frame duration, that is the number of samples per frame, has to be equal for all cells, independent of the employed waveform and modulation such that interference and desired signals can be superimposed.

For our example, assume that OFDM is used in both cells, but cell one employs a subcarrier spacing of 15 kHz while cell two employs a subcarrier spacing of 30 kHz. In order to obtain the same total bandwidth, the number of subcarriers in the first cell must be double the one in the second cell. Let's assume 72 subcarriers for cell one and 36 subcarrier for cell 2. Similarly, the number of symbols per frame in the second cell must be double the number of symbols in the first cell. As we choose the first cell to be LTE-A compliant, there are 15 total symbols (including the guard symbol for Cyclic Prefix (CP)) per frame in cell one and 30 symbols per frame in cell two. While there is a 15th OFDM symbol in an LTE-A frame spent on CP in cell one, we follow this idea and use two symbols of a frame for CP in cell two to obtain the same number of samples per frame. Still, the sampling rate has to be the same for all nodes and is a common parameter. The parameters described above are entered in a scenario file the following way

```

1 scStr.modulation.numOfSubcarriers = [72, 36];           % per BS
2 scStr.modulation.subcarrierSpacing = [15e3, 30e3];    % per BS
3 scStr.modulation.nSymbolsTotal    = [15, 30];         % per BS
4 scStr.modulation.nGuardSymbols     = [1, 2];          % per BS
5 scStr.modulation.samplingRate      = 15e3*72*2;
```

where the array index corresponds to the base station index, except for the sampling rate, which has to be the same for all nodes.

As the number of subcarriers and users are chosen individually for each cell, also the schedule has to be adapted accordingly. The schedule is fixed over time, that is, it stays constant for all frames, and for all symbols within a frame. For user assignment, blocks of subcarriers are designated to users of a cell. This schedule is then considered for up- and downlink. The number of users and total subcarriers has to correspond to the topology and modulation settings. The schedule has to be entered in a scenario file the following form

```

1 scStr.schedule.fixedSchedule{1} = [ 'UE1:36,UE2:36' ]; % BS1
```

¹Of course, the scheduled bandwidth may be different from user to user.

```
2 | scStr.schedule.fixedSchedule{2} = [ 'none:18,UE3:18' ]; % BS2
```

There is a schedule for each base station or cell. In our example there are two users attached to base station one, namely user one and user two. The total number of subcarriers is 72 and we chose to share them equally on the two users. Since the subcarrier spacing is doubled in cell two, the total number of subcarriers is 36 here. The first half of this bandwidth is left unassigned by the keyword *none* while the second half is assigned to user three. In this way, user three is scheduled on the exact same frequency resources as user two.

Depending on the desired simulation scenario, the interference channel from user three to base station one is critical in this setup. If a high attenuation of interference channels is set according to Section 6.1, cell one and cell two will not influence each other. If a low attenuation is selected, significant interference from user three to users one and two will occur. In this setting, users three and one will interfere although they are not overlapping in frequency since they are not orthogonal due to the different subcarrier spacings. If the co-existence of users with different subcarrier spacings is subject of investigation, the interference link attenuation may be set to the same value as the channel's path loss. In this case, interference channels and desired channels are generated following the same statistics with the same average channel power and are therefore equivalent. Considering the received signals at base station one, it is not distinguishable if user three is within cell one or cell two. By this method, users within one cell that employ different modulation schemes can be simulated.

4 Simulation Examples and Scenarios

We include several pre-defined simulation scenarios within the download package of our simulator. The easiest way to define simulation settings is via the scenario files located in the **Scenarios** folder. We provide some ready-to-simulate scenario files with the simulator, that are described in Section 4.2.

However, there might be simulations and comparisons which cannot be obtained by the 5G Link Level Simulator in its original form directly. Since the simulator is implemented in a modular way, using object oriented programming, parts of it might be reused within a different simulation script. We provide simulation examples in the **Examples** folder, that re-use parts of the simulator by exploiting objects. These examples are described in Section 4.1.

4.1 Simulation Scripts

This section describes simulation scripts that are provided in the **Examples** folder and do not exploit the whole implemented Vienna 5G Link Level Simulator structure. To run this example simulations, execute the desired script directly in MATLAB.

4.1.1 Channel Coding with Short Block Length

The example script `shortBlockChannelCoding.m` performs a comparison between the channel coding schemes of convolutional, turbo, LDPC, and polar codes for the case of short block length combined with a low code rate. Such scenario has its importance in various applications, such as the control channels in cellular systems, and also in the Fifth Generation (5G) Massive Machine Type Communication (MMTC) and Ultra-Reliable and Low-Latency Communication (URLLC) use cases. As mentioned in the introduction of the section, this example uses the coding object in a separate manner from the simulator. With this way, we set the target length and code rate irrespectively of the number of scheduled resources or the target CQI code rate. The simulation is then setup according to Table 1. Once the simulation is finished, the script calculates the confidence intervals and plots the results. It is clearly visible in the results that the polar code is the clear winner in such scenario. At the FER of 10^{-2} , we observe a lead for the polar code of about 1 dB against the LDPC code and of 1.5 dB against the turbo and convolutional codes. This result, however, does not alone rule out the choice of the coding scheme, as there are other considerations with respect to the decoding latency, hardware implementation, etc.

| parameter | value | | | |
|----------------------|----------------------------|----------------|-------------|-------------|
| channel code | convolutional | turbo | LDPC | polar |
| decoder | MAX-Log-MAP | Linear-Log-MAP | PWL-Min-Sum | CRC-List-SC |
| iterations/list size | - | 16 | 32 | 32 |
| block length | 64 bits (48 info + 16 CRC) | | | |
| code rate | 1/6 | | | |
| modulation | 4 QAM | | | |
| channel | AWGN | | | |

Table 1: The simulation parameters of the channel coding for short block lengths example. Four different channel codes are compared for the same short block length.

4.1.2 Comparison of Coding Schemes

The script `Comparison.CodingSchemes_AWGN_BPSK.m` results in a comparison in terms of BER over Signal to Noise Ratio (SNR) for supported channel coding schemes. Compared channel coding schemes are turbo coding, LDPC coding, convolutional coding and polar coding. For this example a block length of 1712 bits and an Additive White Gaussian Noise (AWGN) channel are employed.

4.1.3 Comparison of Waveforms

The example script `Comparison.Waveforms.m` provides a comparison of different waveforms that are supported by our simulator. Specifically, it shows results for OFDM, FBMC, Weighted OverLap and Add (WOLA), filtered OFDM (f-OFDM) and UFMC. The waveforms are compared in terms of their pulse shape in time and frequency domain. Further, simulation results for the BER over a frequency selective channel are generated.

4.1.4 LTE Comparison

The script `LTE.comparison.m` simulates all CQI values and saves the results. This results can then be compared to results obtained with the Vienna LTE-A simulator. Together with the `plotLTEComp.m` script, this enables to reproduce the comparison shown in Section 5.

4.2 Simulation Scenarios

Pre-defined simulation scenarios, defined via a corresponding scenario file, are described in this section. Choose the scenario to simulate via

```

1 simulationScenario = 'genericScenario'; % select a simulation
   scenario

```

in the `main.m` simulation script.

4.2.1 Generic Scenario

This scenario file does actually not fit any specific simulation setup. It serves as a reference, as all possible parameter settings are listed and describes. We chose to make this scenario file as self-explaining as possible rather than describing each single parameter in this documentation. Further, this scenario is as a good starting point to define your own simulation setup.

4.2.2 LTE-Advanced

This simulation scenario sets up a single cell downlink transmission with LTE-A standard compliant settings. This means a subcarrier spacing of 15 kHz and 14 OFDM symbols within a frame of 1 ms duration. Also the LTE downlink pilot pattern is employed together with pilot aided Least Squares (LS) channel estimation and linear interpolation. A 2×2 Multiple-Input Multiple-Output (MIMO) transmission with a fixed identity precoder is employed.

4.2.3 Multi-Link Simulation

In this scenario we want to investigate the impact of interference between two users. We consider only uplink transmissions for this example. We assume two users that employ different numerologies (subcarrier spacings) and are therefore non-orthogonal to each other. The two users are scheduled next to each other in frequency as shown in Fig. 2, such that they experience interference due to the high Out Of Band (OOB) emissions of OFDM. As already described in Section 3, there may only be one numerology, waveform and channel coding method per cell. Therefore, the topology setup is as follows. There are two base stations and two users in total, one user per cell. UE1 is assigned to BS1 and UE2 is assigned to BS2. However, to virtually place these two users within one cell, we exploit the interference links and set the inter-cell attenuation to the same value, as the channel path loss. By this, the interference channels and desired channels become indistinguishable. This enables to investigate the impact of inter-user interference. These topology settings are obtained by the following parameters:

```

1 scStr.topology.nodes = ['BS1',BS2,UE1,UE2'];
   % 2 cells with one user each

```

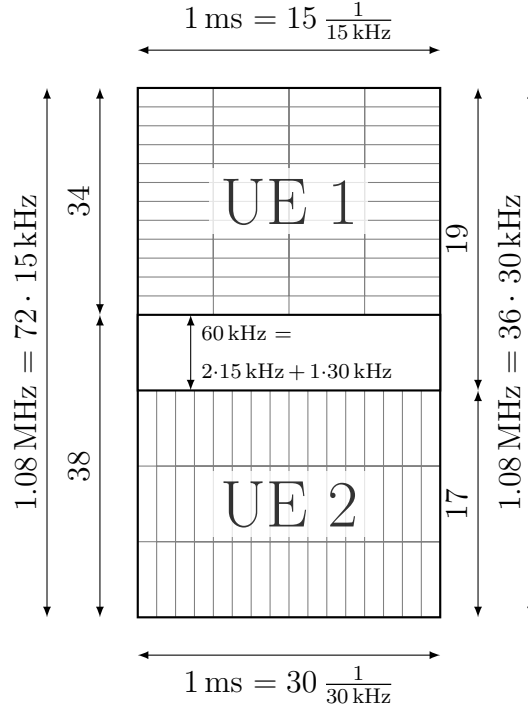


Figure 2: User resource allocation within a cell.

```

2 | scStr.topology.primaryLinks          = ['UE1:BS1,UE2:BS2'];
   | % downlink links only
3 | scStr.topology.interferenceGeneration = 'Automatic';
   | % automatic generation of interference links
4 | scStr.topology.attenuation           = 107;
   | % channel attenuation fits channel path loss

```

The interfering links from UE2 to BS1 and from UE1 to BS2 are generated automatically due to these settings. The transmit power of UE2 is swept over to obtain inter-user interference of different strengths.²

```

1 | scStr.simulation.sweepParam          = {'simulation.txPowerUser'};
   | % sweep over a user's transmit power
2 | scStr.simulation.sweepValue          = 10:5:60;
   | % transmit power of UE2
3 | scStr.simulation.applySweepingTo     = [0,1];
   | % apply sweep to second user only
4 | scStr.simulation.pathloss            = 107;
   | % channel path loss for an SNR of 40dB

```

²In this scenario we consider UE1 to BS1 as the desired link and UE2 to BS1 as the interfering link. Of course, this is an arbitrary interpretation and only serves the description of the scenario.

While a subcarrier spacing of 15 kHz is used in the first cell, 60 kHz are used in the second cell. The total number of subcarriers of 72 and 36 are chosen such that the total utilized bandwidths are equal in both cells, i.e., $15 \text{ kHz} \cdot 72 = 30 \text{ kHz} \cdot 36$. As the simulator performs numerical computations on a frame basis, frame durations of both cells must match. Therefore, the number of symbols is chosen such that both frames have the same duration, i.e., $1 \text{ ms} = 15 \frac{1}{15 \text{ kHz}} = 30 \frac{1}{30 \text{ kHz}}$. Also the guard duration (CP length) is chosen accordingly. In an LTE like system, one out of 15 symbols is split up upon the remaining 14 symbols for CP duration of $\frac{1}{15 \text{ kHz} \cdot 14} = 4.76 \mu\text{s}$. For matching frame durations, 2 symbol durations are shared among the remaining 28 symbols for a CP duration of $\frac{2}{30 \text{ kHz} \cdot 28} = 2.38 \mu\text{s}$. Obviously, the factor 2 also appears between the two CP durations of cell one and cell two. The numerology is obtained by the following settings:

```

1  scStr.modulation.subcarrierSpacing    = [15e3, 30e3];
    % 15kHz subcarrier spacing in cell 1 and 60kHz in cell 2
2  scStr.modulation.numOfSubcarriers    = [72, 36];
    % total bandwidth is equal for both cells
3  scStr.modulation.nSymbolsTotal       = [15, 30];
    % number of symbols such that frame durations match
4  scStr.modulation.nGuardSymbols       = [1, 2];
    % guard durations for matching frame duration
5  scStr.modulation.samplingRate        = 30*72*2;
    % a suitable sampling rate

```

For a fair comparison, both users have the same number of total allocated resource elements. A guard band in frequency domain is employed between the users by means of scheduling. This guard band is split among the users such that they both lose the same number of resource elements. This means, UE1 loses 2 subcarriers to the guard band while UE2 loses one subcarrier compared to half the total bandwidth. The schedule is obtained by the following:

```

1  scStr.schedule.fixedSchedule{1} = [ 'UE1:34,none:38' ];    %
    UE1 gets the upper half of BS1's schedule
2  scStr.schedule.fixedSchedule{2} = [ 'none:19,UE2:17' ];    %
    UE2 gets the lower half of BS2's schedule

```

Simulation parameters are summarized in Table 2. The effects of interference between users is clearly visible in the simulation results. While the sweep of UE2's transmit power is carried out up to very high values of 60 dBm, this power has to be seen relative to the transmit power of UE1. Since UE1 has a transmit power of 30 dBm, a transmit power of 60 dBm of UE2 means that UE2 is stronger than UE1 by 30 dB. This could correspond to a near UE2 and a far UE1 in an uplink transmission.

| parameter | value | | |
|--------------------|--|--------------|--------------|
| waveform | OFDM | f-OFDM | FBMC |
| filter type/length | - | 7.14 μ s | PHYDYAS-OQAM |
| CP length | 4.76 μ s | 4.76 μ s | - |
| subcarrier spacing | User 1: 15 kHz, User 2: 30 kHz | | |
| guard band | $2 \times 15 \text{ kHz} + 1 \times 30 \text{ kHz} = 60 \text{ kHz}$ | | |
| bandwidth per user | $34 \times 15 \text{ kHz} = 17 \times 30 \text{ kHz} = 0.51 \text{ MHz}$ | | |
| modulation/coding | 64 QAM/LDPC, $r = 0.65$ (CQI 12) | | |
| channel model | block fading Pedestrian A | | |

Table 2: Multi-link scenario simulation parameters overview.

The pre-defined waveform for this scenario is OFDM. You may want to run the simulation with this parameters and then change the waveform to f-OFDM or FBMC. Comparing the simulation results, you will observe that the weaker UE2 profits from a waveform with quickly decreasing sidelobes.

4.2.4 Flexible Numerology

Flexible numerology is proposed by 3GPP for New Radio (NR) physical layer design. Numerology refers to the parametrization of the multicarrier scheme. It means that we are flexible to choose different subcarrier spacing and thus symbol and CP duration in order to fulfill different service and user demands. The goal of this scenario file is to show how different numerology behaves according to the different channel conditions, i.e. Doppler shift and delay spread of the channel. We assume a Single-Input Single-Output (SISO) OFDM downlink transmission with parameters as summarized in Table 3. In terms of numerology, we specify subcarrier spacing and number of subcarriers for the desired bandwidth and according to the subcarrier spacing we choose appropriate symbol as well as CP duration [1].

```

1  scStr.modulation.numerOfSubcarriers = [48];
   % 384, 96, 48
2  scStr.modulation.subcarrierSpacing = [120e3];
   % 15e3, 60e3, 120e3
3  scStr.modulation.nSymbolsTotal      = [120];
   % 14, 56, 112
4  scStr.modulation.nGuardSymbols      = [8];
   % 1, 4, 8

```

When the total number of subcarriers is changed, the downlink schedule needs to be adapted accordingly.


```

1 scStr.schedule.fixedScheduleDL{1} = ['UE1:48'];
  % ['UE1:384'], ['UE1:96'] or ['UE1:48']

```

In order to change the Doppler shift we sweep over different user velocities with the carrier frequency of 5.9 GHz that is typical for vehicular communications.

```

1 scStr.simulation.sweepParam = {'simulation.userVelocity'};
  % sweep over user velocity
2 scStr.simulation.sweepValue = [5, linspace(50,300,6)./3.6];
  % user velocity values

```

At the same time we employ the Tap Delay Line (TDL)-A channel model with desired Root Mean Square (RMS) delay spread [2].

```

1 scStr.channel.powerDelayProfile = 'TDL-A_45ns';
  % 'TDL-A_45ns' or 'TDL-A_250ns'

```

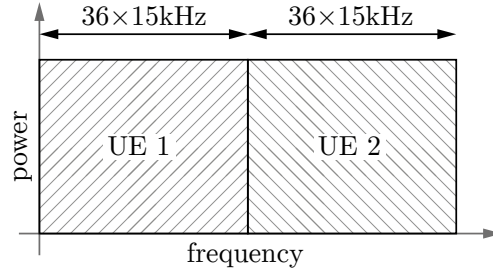
The final results are determined by interplay between Inter-Carrier Interference (ICI) and Inter-Symbol Interference (ISI). Namely, larger subcarrier spacings are more robust to ICI and hence they outperform the smaller subcarrier spacings. Of course, this holds true only with small RMS delay spread channels, since the CP durations of larger subcarrier spacings are sufficient compared to the maximum channel delay spread. If we have high delay spread channels, then due to the insufficient CP duration for large subcarrier spacings we have ISI present, deteriorating our performance. With 120 kHz subcarrier spacing and above, ISI is superior over ICI for the entire range of considered user velocities, making these curves flat.

| parameter | value | | |
|-----------------------------|-----------------|--------------|--------------|
| subcarrier spacing | 15 kHz | 60 kHz | 120 kHz |
| number of symbols per frame | 14 | 56 | 112 |
| CP duration | 4.76 μ s | 1.18 μ s | 0.59 μ s |
| bandwidth | 5.76 MHz | | |
| carrier frequency | 5.9 GHz | | |
| modulation alphabet | 64 QAM | | |
| channel model | TDL-A | | |
| channel RMS delay spread | 45 ns or 250 ns | | |

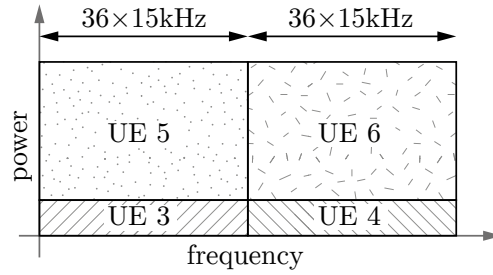
Table 3: Parameters for the flexible numerology simulation example. Three different numerologies are employed.

4.2.5 Non-Orthogonal Multiple Access

The purpose of this scenario is to show the operation of Non-Orthogonal Multiple Access (NOMA) in the simulator, and demonstrate the gain offered



User schedule for the OMA case.



User schedule for the NOMA case.

Figure 3: User assignment for the NOMA simulation scenario.

the 3GPP Multi-User Superposition Transmission (MUST). For this, we set up two cells, the first one operates with Orthogonal Multiple Access (OMA), and the other one with MUST. In each cell, the Base Station (BS) splits the bandwidth equally between two User Equipments (UEs) that have good channel conditions (strong users); however, since the second BS supports MUST, it can superimpose those two strong UEs with two other cell-edge UEs (weak users). In this ideal case, a cell overloading of 200% is achieved. The topology of this scenario is setup according to

```

1 scStr.topology.nodes = ['BS1', 'BS2', 'UE1', 'UE2', 'UE3', 'UE4',
2   'UE5', 'UE6'];
3 scStr.topology.primaryLinks = ['BS1:UE1', '...
4   'BS1:UE2', '...
5   'BS2:UE3', '...
6   'BS2:UE4', '...
7   'BS2:UE5', '...
8   'BS2:UE6'
];

```

Next we setup the strong and the weak users. This is done by adjusting their corresponding path loss

```

1 scStr.simulation.pathloss = [80, 90, 80, 90, 110, 115];

```

Therefore, UE1 link gets a path loss of 80 dB, UE2 gets 90 dB, etc. Next, we tell the simulator that the pairs UE3-UE5 and UE4-UE6 are operating in MUST mode. This is achieved by setting the downlink schedule as follows

```

1 % BS1 does Orthogonal Multiple Access
2 scStr.schedule.fixedScheduleDL{1} = ['UE1:36,UE2:36'];
3
4 % BS2 does MUST operation
5 scStr.schedule.fixedScheduleDL{2} = ['UE3:36,UE4:36,UE5:UE3
   ,UE6:UE4'];

```

The user assignment is illustrated in Fig. 3. Notice how the weak users get most of the transmit power in the NOMA case. Table 4 summarizes the simulation parameters for this scenario.

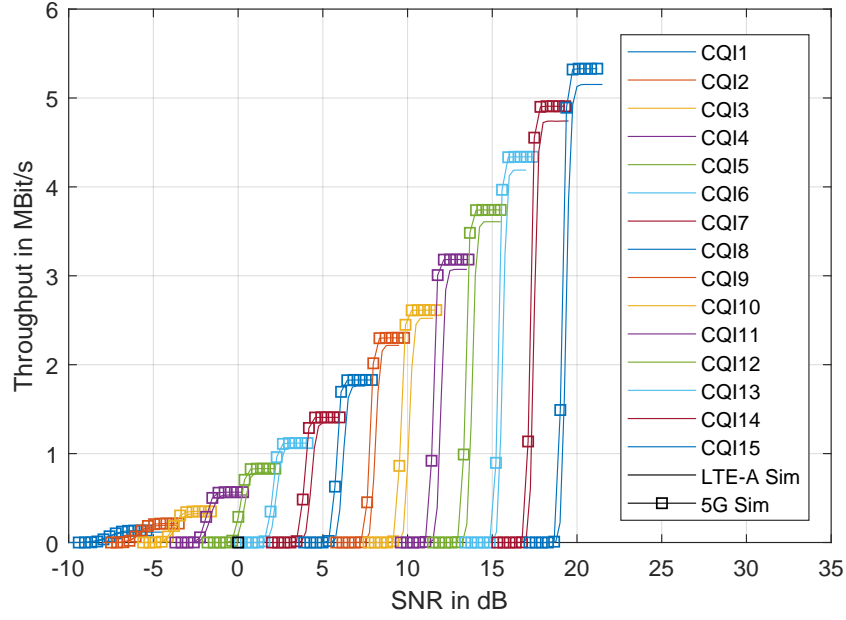
The resulting curves show that MUST allows the BS to support more users, and when combined with a sufficiently high transmit power, it offers a higher downlink spectral efficiency.

| parameter | value | |
|----------------------|--------------------------|---|
| cells | OMA | NOMA |
| number of users | 2 | 4 (2 strong, 2 cell-edge) |
| path-loss | 80, 90 dB | strong: 80, 90 dB cell-edge: 110, 115 dB |
| NOMA receiver | - | ML |
| MUST power-ratio | - | fixed (second ratio) |
| bandwidth | 1.4 MHz (72 subcarriers) | |
| waveform/coding | OFDM, LDPC | |
| MIMO mode | 2×2 CLSM | |
| modulation/code rate | adaptive (CQI based) | |
| feedback delay | no delay (ideal) | |
| channel model | Pedestrian A | |

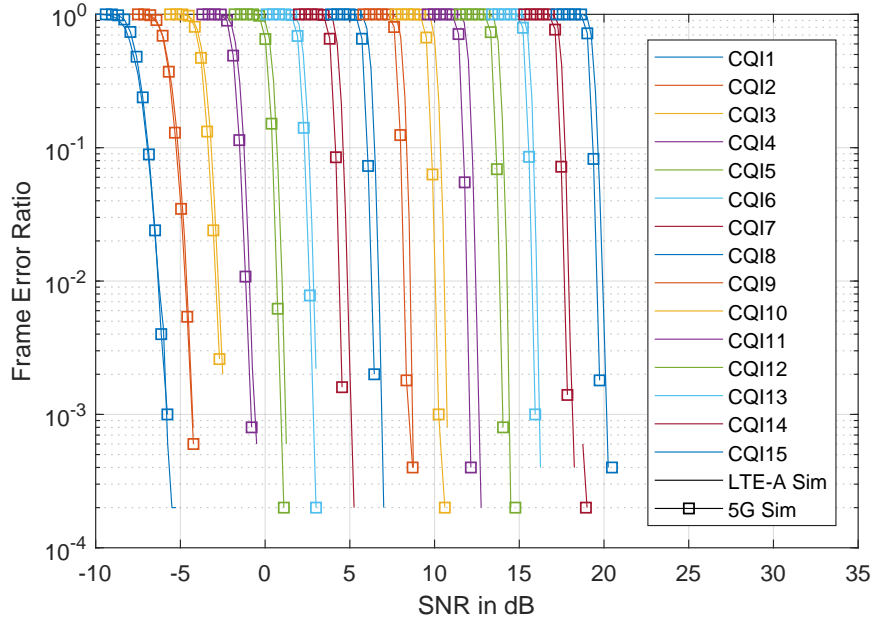
Table 4: The simulation parameters for the NOMA example scenario.

5 Comparison to LTE-A

We offer an extremely flexible simulator, that is able to simulate almost any multi-carrier system. Therefore, physical layer methods for future wireless communications systems can be investigated and compared. On the other hand, the Vienna 5G Link Level Simulator is also able to perform LTE-A compliant simulations. To justify this claim, reference simulations for 15 CQI values were carried out on a SISO AWGN channel. Simulation results obtained with the Vienna LTE-A Downlink Link Level Simulator and the Vienna 5G Link Level Simulator are shown in terms of throughput and FER in Fig. 4. These results are reproducible via a simulation example, see Section 4.1.4.



(a) Throughput comparison.



(b) Frame Error Ratio comparison.

Figure 4: LTE-A simulator to 5G simulator comparison on SISO AWGN channel for 15 CQI.

6 Simulator Structure

The simulator is built upon a flexible structure that can run complicated tasks, yet allows for easy addition of new features later on. The basis of the structure is the Link object, which carries all the information and functionalities of the connection between nodes (base stations and users) in the cellular map.

6.1 Topology Configuration

The generation of the links between the nodes is based on the supplied ‘topology’, which basically tells how the nodes are connected together. For example, in order to configure the simulator for the topology shown in Figure 1, open the file `Scenarios.genericScenario.m`, and modify `topology.nodes` and `topology.primaryLinks` as follows

```

1 scStr.topology.nodes = ['BS1',BS2,UE1,UE2,UE3'];
2 scStr.topology.primaryLinks = [ 'BS1:UE1',' ...
3                               'BS1:UE2',' ...
4                               'BS2:UE3',' ...
5                               'UE1:BS1',' ...
6                               'UE2:BS1',' ...
7                               'UE3:BS2'];
8 scStr.topology.interferenceGeneration = 'Automatic';
9 scStr.topology.attenuation           = 30; % in dB

```

The first property `topology.nodes` contains all the nodes in the network. The nodes names must be entered in an ascending order, i.e. UE1,UE2,UE3 and the node number must be larger than zero, i.e. BS0 or UE0 are not allowed. Once the participating nodes are entered, the next step is to define the connected links. The property `topology.primaryLinks` takes care of that, where each line indicates a connection. For each line, the first entry is the transmitting end, the second entry is the receiving end. For example, ‘BS1:UE1’ indicates a downlink from BS1 to UE1, ‘UE2:BS1’ indicates an uplink from UE2 to BS1, etc. The two entries are separated by a colon, and each line is terminated by a comma.

Interference from other cells is defined in the same manner using additional links. These interference links can be setup automatically by setting `topology.interferenceGeneration` to ‘Automatic’, in which all possible interference links between the nodes in the different cells are automatically generated. The strength of the interference can be controlled by introducing attenuation to the interference links. This is set in dB through `topology.attenuation`, where low values indicate high amount of inter-cell interference. Alternatively, custom configuration of the interference links is also possible. To see how the attention property works, check out the example scenario in Section 4.2.3.

6.2 Links Generation

Once the desired topology is entered, the links are generated using the function `Topology.getTopology`. The output of this function are three collections: BS, UE, Links. The BS collection contains the base stations, UE contains the users, and Links contains the generated Link objects. All the generated nodes (base stations and users) have unique IDs. These IDs are used to access the associated links between these nodes.

The Links collection has a very specific structure, namely, it's a 2D cell where the rows indicate the transmitters ID, and the columns indicate the receivers ID. Assume you would like to access the downlink and uplink between BS1 and UE2, this can be done as follows

```
1 downlink = Links{BS{1}.ID, UE{2}.ID};
2 uplink   = Links{UE{2}.ID, BS{1}.ID};
```

As states above, the row index is the transmitter ID, and the column index is the receiver ID. Such unique ID association is important to avoid conflict between the uplink and downlink, otherwise they would share the same index.

6.3 The Link Object

As mentioned above, the building block of the structure is the Link object. On the one hand it contains all the properties that define a connection between two nodes, like the connected nodes IDs, scheduled resources, path loss, receiver SNR, velocity, etc. It also contains all the generated signals throughout the transmission, like the data bit stream, modulation symbols, soft bits, etc. On the other hand, it offers signal processing functionalities like channel coding, modulation, MIMO processing, the transmission channel, equalization, feedback, etc.

Such structure simplifies the simulation of multiple base stations and users, since each link (connection) can have a different set of parameters, which are grouped together and accessed under the same package. This in turn, greatly reduces the processing overhead, since all the information is available locally to the object.

6.4 Some Rules

Almost every parameter can be applied either to each node or link individually, or it can be applied globally. This can greatly simplify the way the parameters are entered in the scenario file. Let us consider the following example. Assume you have the following topology

```
1 scStr.topology.nodes = ['BS1,BS2,UE1,UE2'];
```

Now you would like to have both base stations transmitting with 30 dBm power. This can be either entered individually

```
1 scStr.simulation.txPowerBaseStation = [30,30];
```

or, since both of them have the same value, it can be entered globally. To do so, you only have to enter the parameter once, i.e.,

```
1 scStr.simulation.txPowerBaseStation = [30];
```

and the simulator will take care of applying the parameter to all the nodes or links. The `txPowerBaseStation` is a per BS parameter. Similar thing can be applied to a per link parameter such as the `pathloss`. For example, assume you have the following links

```
1 scStr.topology.primaryLinks = ['BS1:UE1','...
2                               'BS1:UE2','...
3                               'BS2:UE3','...
4                               'BS2:UE4','...
5                               'BS2:UE5','...
6                               'BS2:UE6'
7                               ];
```

Then setting the `pathloss` according to

```
1 scStr.simulation.pathloss = [80,90,70,88,110,115];
```

indicates that 'BS1:UE1' has a pathloss of 80 dB, 'BS1:UE2' has a pathloss of 90 dB, 'BS2:UE3' of 70 dB, etc. Is the parameter per node, per BS, per UE, or per link. In the next subsection we mention some of them and how they are used, and also in the scenario files that are supplied with the simulator, we give a description for each parameter and its type.

6.5 General Simulation Parameters

This subsection is dedicated to the category of the general simulation parameters. They allow you to control the flow of your simulation, and in certain cases allow you to explore new simulation ideas. For the other set of parameters, such as the channel, MIMO, modulation, coding, etc, you get to know them throughout the other sections of the manual.

Let start by listing all of them below

```
1 %% General Simulation Parameters
2 scStr.simulation.simulateDownlink = true;
3 scStr.simulation.simulateUplink   = false;
```

```

4  scStr.simulation.simulatedD2D      = false;
5
6  scStr.simulation.plotResultsFor     = [1];
7  scStr.simulation.plotOverSNR       = true;
8  scStr.simulation.saveData          = false;
9
10 scStr.simulation.sweepParam         = {'simulation.pathloss'};
11 scStr.simulation.sweepValue         = linspace(130,80,6);
12 scStr.simulation.applySweepingTo    = [1];
13
14 scStr.simulation.nFrames            = 100;

```

The first three parameters `simulateDownlink`, `simulateUplink`, `simulateD2D` control the direction in which the simulation is carried out. These can be useful when the loaded scenario has a lot of connections in the downlink, uplink, and D2D, and then at certain point you are interested in simulating only a single direction.

The next set of parameters control the results processing. The parameter `plotResultsFor` manages which nodes will get their results plotted. When you set this parameter to 1, then all the nodes will get their results plotted, while if you set it to zero, then no plots will be shown. It is also possible to show the plots of only a certain number of nodes. For example, assume your topology is defined as follows

```

1  scStr.topology.nodes = ['BS1',BS2,UE1,UE2'];

```

and now if you want to show the results for only BS2 and UE2, then you need to set their corresponding positions in the `plotResultsFor` to 1, while set the others to 0, i.e.,

```

1  scStr.simulation.plotResultsFor     = [0,1,0,1];

```

The parameter `plotOverSNR` plots the final results in terms of the SNR when it is set to 1. This only works when the sweep parameter is the pathloss. The last parameter in this set is `saveData`, and it controls whether the whole generated signals (input bits, symbols, decoded bits, etc) are saved in the results, or only the instantaneous (per frame) FER and BER are saved. It is recommended to keep it turned off, especially if you are going for a lot of realizations and simulation points.

Perhaps the most important set here is the sweep parameter set. The first entry `sweepParam` selects the parameter that will be swept over. Currently, the simulator supports the following sweep parameters: `attenuation` (per link), `pathloss` (per link), `txPowerBaseStation` (per BS), `txPowerUser` (per UE), `userVelocity` (per UE). The values of sweeping are set through the parameter `sweepValue`, this works for most of the values as long as they are

valid. The next parameter `applySweepingTo` is the most exciting here, as it allows you to setup an even more flexible scenarios. As the name suggest, it allows you select which nodes or links are swept over. The selection is done in a similar way as with `plotResultsFor`; you basically insert a 1 in the position of the link or node, and set the rest to zero. Then only the selected links/nodes will be swept over, while other nodes will use the default values that are entered globally. Again, entering only a single 1 means that all the links/nodes will be swept over.

To demonstrate how it works, let us consider the following setup (showing only the relevant parameters)

```

1 scStr.topology.nodes          = ['BS1,UE1,UE2'];
2 scStr.topology.primaryLinks   = ['BS1:UE1,' ...
3                               'BS1:UE2'];
4
5 scStr.simulation.sweepParam    = {'simulation.pathloss'};
6 scStr.simulation.sweepValue    = linspace(130,80,6);
7 scStr.simulation.applySweepingTo = [0,1];
8
9 scStr.simulation.pathloss      = [80];

```

The pathloss is a per link parameter, and therefore setting `applySweepingTo` to `[0,1]` means that link 'BS1:UE2' will not be swept over, but rather it will take the value that is entered in `simulation.pathloss`. In our example it is 80 and it will continue to have this value until the simulation end. Link 'BS1:UE2' on the other hand will be swept over and its path loss value is given by the current sweep value in the simulation loop. Let us now consider another example that uses `userVelocity`. Consider the following setup (again showing only the relevant parameters)

```

1 scStr.topology.nodes          = ['BS1,UE1,UE2,UE3'];
2 scStr.topology.primaryLinks   = ['BS1:UE1,' ...
3                               'BS1:UE2,' ...
4                               'BS1:UE3'];
5
6 scStr.simulation.sweepParam    = {'simulation.
7   userVelocity'};
8 scStr.simulation.sweepValue    = linspace(0,250,10);
9 scStr.simulation.applySweepingTo = [0,0,1];
10 scStr.simulation.userVelocity  = [5];

```

In this setup, only UE3 will have its velocity changing during the simulation. UE1 and UE2 will have their velocity fixed to 5 m/s. Check out the scenario in Section 4.2.3 to see how useful this can be.

7 Channel Models

As the aim of LL simulation is acquisition of the average link performance, many random channel realizations are necessary per scenario. There exists no network geometry and therefore no path-loss model. A link's path-loss is an input parameter, determining the user's average SNR. Therefore, the channel model only includes small scale fading effects while its average power is dictated by the given path-loss. We offer frequency selective and time selective fading channel models. The frequency selectivity is implemented as tapped delay line model. Currently we offer implementations for Pedestrian A, Pedestrian B, Vehicular A [3], TDL-A to TDL-C [2], Extended Pedestrian A and Extended Vehicular A [4]. For the TDL channels, the effective RMS delay spread can be chosen via the Power Delay Profile (PDP) parameter, for example:

```
1 scStr.channel.powerDelayProfile = 'TDL-A_45ns';
```

To model the channel's time selectivity, the fading taps change over time to fit a certain Doppler spectrum. A Jake's as well as a uniform Doppler spectrum are currently implemented. Jake's model also supports time-correlated fading across frames. In the case that all users are stationary, i.e they all have velocity set to zero, the Two-Wave with Diffuse Power (TWDP) fading model is employed. In the mixed case of moving and stationary users, Rayleigh fading will be used for the stationary users. TWDP is a generalization of the Rayleigh and Rician fading models [5]. In contrast to the Rayleigh fading model, where only diffuse components are considered, and the Rician fading model, where a single specular component is added, two specular components together with multiple diffuse components are considered in the TWDP fading model. The two key parameters for this model are K and Δ . They can be added to a scenario file the following way

```
1 scStr.channel.K = 40;           % TWDP model parameter K,
2 scStr.channel.delta = 0;        % TWDP model parameter delta
```

In order to use the TWDP model, the maximum Doppler shift must be set to 0, which corresponds to a time-invariant case. Similar to the Rician fading model, the parameter K represents the power ratio between the specular and diffuse components and is always positive. The parameter Δ is related to the ratio between peak and average specular power and thus describes the power relationship between the two specular components, and is limited to the interval from 0 to 1. By cleverly choosing of K and Δ , the TWDP fading model is able to characterize small scale fading for a wide range of propagation conditions, from no fading to hyper-Rayleigh fading. Table 5 shows typical

| Fading statistic | K | Δ |
|------------------|----------|-------------|
| No fading | ∞ | 0 |
| Rician | > 0 | 0 |
| Rayleigh | 0 | - |
| Hyper-Rayleigh | ∞ | ≈ 1 |

Table 5: Parameters of the TWDP fading model.

parameter combinations and their corresponding fading statistic. In contrast to classical models, the TWDP fading model allows for destructive interference between two dominant specular components. This enables for a possible worse than Rayleigh fading, depending on the fading model parameters. Spatial correlation of MIMO channels is implemented via a Kronecker correlation model with correlation matrices. A possible initialization of the channel object would look as follows

```

1 Channel.FastFading(...)
2     2.94e6,...           % sampling rate
3     'Flat', ...         % channel model
4     200, ...            % total number of samples
5     0, ...              % doppler frequency
6     'Jakes',...        % doppler model
7     200,...             % number of paths
8     false,...           % time correlated fading
9     'none',...          % spatial correlation
10    0,...               % spatial correlation coeff TX
11    0,...               % spatial correlation coeff RX
12    1,...               % number of transmit antennas
13    1,...               % number of receive antennas
14    true,...            % show checks
15    10,...              % TWDP model parameter K
16    0.5...             % TWDP model parameter Delta
17 );

```

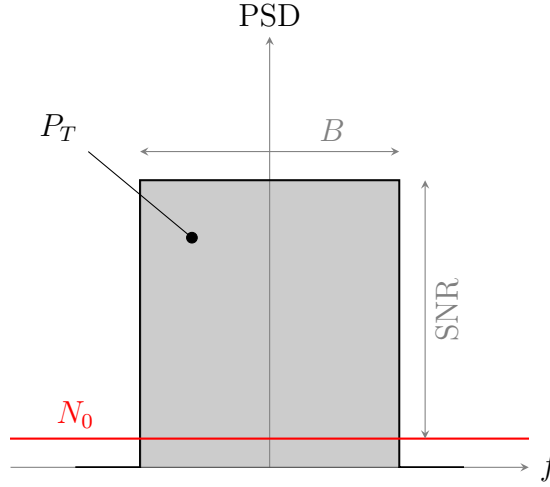


Figure 5: Calculation of SNR.

8 Definition of SNR

With our link level simulator we facilitate simulation of several waveforms, which are considered for 5G. In order to achieve a fair comparison between different waveforms, we fix the transmit power for each node (base station or user). At the receive side, thermal noise of constant Power Spectral Density (PSD) is added to the signal. The resulting SNR then depends on the employed bandwidth of a specific waveform.

The calculation of SNR is illustrated in Fig. 5. The total transmission power is fixed for each node and is denoted by P_T . This power is spend equally on the whole scheduled transmission bandwidth B . Therefore, the signal PSD is scaled such that P_T stays constant for any bandwidth B . This means, the gray shaded region in Fig. 5 corresponds to P_T and has a constant area.

The definition of the transmit power is illustrated in Fig. 6. In this figure, the mean signal power $\mathbb{E}\{|s(t)|^2\}$ is plotted. Here, K denotes the number of symbols per frame and T is the time spacing between two consecutive symbols. The total signal power is then given by

$$P_T = \frac{1}{KT} \int_{-\infty}^{\infty} \mathbb{E}\{|s(t)|^2\} dt . \quad (8.1)$$

The thermal noise PSD is denoted by N_0 and constant over frequency. It is given by $N_0 = k_B \vartheta$, with Boltzmann's constant k_B and temperature ϑ .

The SNR is defined as the ratio between signal PSD P_T/B and thermal

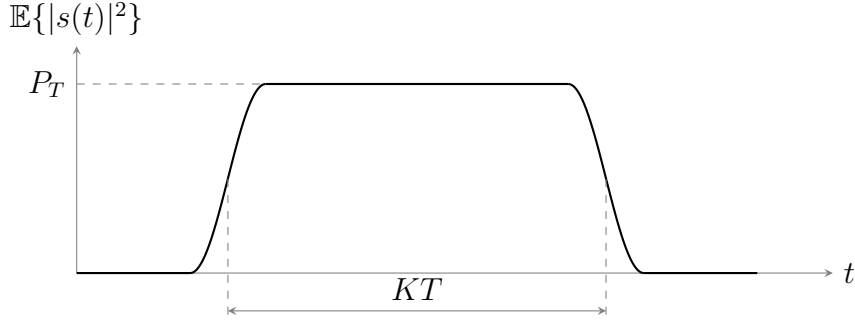


Figure 6: Definition of transmit power.

noise PSD N_0 , which yields

$$\text{SNR} = \frac{1}{N_0} \frac{P_T}{B} . \quad (8.2)$$

Of course, the received signal power depends on the wireless channel. However, we normalize the channel to have an average power of one. In this sense, the SNR calculated in (8.2) is an average SNR.

To enable sweeps over SNR, several simulation runs are carried out with different transmit powers. While the actual transmission power is fixed for each node, an artificial parameter, referred to as pathloss, is introduced. As there is no network geometry and no large scale fading in our link level simulator, this pathloss value is not a channel property, but only serves the purpose to adapt the transmission power. In this sense, the transmit power P_T in (8.2) is modified as

$$P_T = \text{PL} P_T^* , \quad (8.3)$$

where PL denotes the pathloss and P_T^* is the fixed transmission power.

8.1 Scheduling

For a downlink transmission, B is the total scheduled bandwidth for all users. In this case, the signal PSD increases only when a portion of the whole available bandwidth is unused, that is, when not the whole available bandwidth is scheduled. For an uplink transmission, a user transmits with power P_T on its scheduled bandwidth B . In this case, the signal PSD increases with decreasing scheduled bandwidth B , such that P_T remains constant.

From the previous paragraph it is clear, that the SNR depends on the scheduled bandwidth. A time variant schedule, that is, a resource allocation that changes from transmission frame to transmission frame, therefore means a different resulting SNR in each frame. Predicting the resulting SNR and

the number of simulated frames per SNR value is not straight forward, thus significantly complicating the simulation procedure. Further, obtained results need to be sorted according to the SNR for plotting and interpretation. We therefore do not allow for time variant scheduling in our link level simulator.

9 Channel Coding

The first block in the processing chain is the channel coding, where redundancy is added to provide error correction and detection capabilities for the wireless transmission. The simulator supports the four candidates (or were candidates) of 5G: convolutional, turbo, LDPC, and polar codes. The aim was to have a single structure that can handle the four coding schemes simultaneously, with challenges arising due to the different requirements of the different coding schemes. Table 6 summarizes the supported schemes, their construction, and the corresponding decoding algorithms.

Table 6: Supported channel coding schemes.

| scheme | construction/ encoding | decoding algorithms |
|---------------|---------------------------|--|
| turbo | LTE | Log-MAP Linear-Log-MAP MAX-Log-MAP |
| LDPC | 5G NR | Sum-Product PWL-Min-Sum Min-Sum |
| polar | currently custom | SC List-SC CRC-List-SC |
| convolutional | LTE | Log-MAP MAX-Log-MAP |

The procedure of the channel coding is identical across all the aforementioned schemes. We describe in the following subsections the main steps.

9.1 Block Length Calculation and Segmentation

The first step is to determine how many information bits are supported by the current transmission. This depends on how many resources are scheduled, the modulation order, and on the code rate. The modulation order and code rate are obtained using the current CQI. Once this is determined and depending on the chosen coding scheme, filler bits might be added to the block when its length does not match the size of the interleaver (in case of the turbo code) or the dimensions of the parity check matrix (for the LDPC code). If the block

length is too long, then code block segmentation is performed. For the turbo and convolutional codes, the segmentation follows the LTE standard, while for the LDPC code, it follows the current 5G specs. For the polar code, the segmentation is similar to that of the turbo code, but it has more granularity in the selection of the block length, since the polar code does not require a strict set of input lengths.

9.2 Convolutional Code

The implementation of the convolutional code is based on the LTE standard [6]. More specifically, it is a tail-biting convolutional code, meaning that the starting and ending states of the encoder is the same. In the simulator, we pass this state directly to the decoder (i.e., *gene-aided*). This should not have an impact on the performance of the code, however, it will reduce the decoding complexity, as the decoder does not have to spend time figuring out that state. The encoder is implemented using a shift register, and it is initialized with the last bits of the information block, guaranteeing that the initial and final states of the encoder are the same, i.e., *tail-bitten*.

The decoder is based on the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [7], which is the efficient implementation of the bit-wise Maximum A-Posteriori (MAP) decoder. The supported algorithms are the ‘Log-MAP’, that is the original MAP algorithm in the log domain, and the sub-optimal ‘MAX-Log-MAP’ which provides lower complexity.

9.3 Turbo Code

Similar the convolutional code, the turbo code is also based on the LTE standard [6]. The encoder consists of two recursive constituent convolutional encoders. The encoders are initialized with zeros, and the final states are tracked through trellis-termination.

The iterative (turbo) decoding is based on the BCJR algorithm. It supports the ‘Log-MAP’, ‘MAX-Log-MAP’, and the ‘Linear-Log-MAP’ algorithms. The latter uses a linear function to approximate the exponential correction term in the original MAP algorithm.

9.4 LDPC Code

The LDPC code is based on the current 5G NR specs [8]. The employed code is quasi-cyclic and therefore allows easy adaptation of the parity check matrix to different input lengths. The standard defines two base parity check matrices, or as it is called, a Base Graph (BG). Depending on the code rate

and the input length, either BG 1 or BG 2 is used. One BG is better suited for short lengths and low code rates, while the one is more suited for long lengths and high code rates. Thanks to the diagonal and double-diagonal structures of the parity check matrix, the encoding can be carried out with low complexity. The encoding is systematic, and at the output of the encoder, a certain amount of the systematic bits are punctured. These punctured bits never enter the circular buffer.

The decoder is based on layered Belief Propagation [9] or usually called Sum-Product algorithm. The layering is utilized through the Column Message Passing schedule [10]. The supported decoding algorithms are ‘Sum-Product’, the lower complexity ‘Min-Sum’, and the ‘PWL-Min-Sum’, where PWL stands for Piecewise Linear. Similar to the turbo decoder, it uses linear functions to approximate the correction terms.

9.5 Polar Code

The construction of the polar code (i.e., finding the frozen set) is currently based on method in [11]. Once the set is found, the next power-of-two polarization transform is selected. In case the codeword length does not match the generator matrix (polarization transform) size, then the extra positions from the bottom of the generator matrix are set to zero and then removed at the output. At the decoder side, the Log-Likelihood Ratio (LLR) of these positions is set to a high value, reflecting a $+\infty$ LLR. The generator matrices for sizes above 512 are precalculated for faster operation.

The decoder is based on Successive Cancellation (SC) and its extensions of List-SC and Cyclic Redundancy Check (CRC)-aided List-SC.

9.6 Interleaving and Rate Matching

For the turbo and convolutional codes, the rate matching procedure is identical to the standard. The interleaving is carried out directly on each of the three streams at the encoder output, i.e., subblock-interleaving. After interleaving, the streams are passed to the circular buffer, where puncturing and/or repetition is performed in order to meet the output length, and consequently the target code rate. For the turbo code, part of the systematic stream is skipped at the first transmission from the circular buffer.

For the polar code, the whole non-systematic codeword is interleaved. The main rate adaption is carried out at the input of the generator matrix, by adjusting how many bits are frozen. After interleaving, the codeword is passed to the circular buffer, where further slight puncturing or repetition is performed in case the codeword does not exactly match the target length.

As for the LDPC code, we follow the 5G NR chain, in which the systematic codeword is passed to the circular buffer directly without interleaving. The codeword is then punctured/repeated in order to meet the target length. After the codeword is rate matched, it is then interleaved using a rectangular interleaver. The interleaving pattern depends on the modulation order.

9.7 Object Usage

The object is initialized in the following manner:

```

1 ChannelCodingObj = Coding.ChannelCoding( ...
2     'Turbo', ...           % coding scheme
3     'Linear-Log-MAP', ...  % decoding algorithm
4     1/2, ...               % code rate
5     8 ...                 % iterations/list size
6 );

```

Once the object is initialized, the next step is to update the object parameters based on the required input or output length. This can be done in two ways, the first one is

```

1 inputLength = ChannelCodingObj.update('Output', N, R, Qm,
    SoftBufferRatio);

```

where N , R , Q_m , and `SoftBufferRatio` are the output length, code rate, modulation order, and soft-buffer rate-matching ratio, respectively. This is used when the output length is known and you would like to get how many information bits are needed at the input. Alternatively, it can be used in the following way

```

1 ChannelCodingObj.update('Input', K, R, Qm, SoftBufferRatio);

```

where K is the input length. In this case, the function `update()` does not return anything and the output length is set automatically based on the input length. Beside calculating the input length, the function `update()` prepares the object for codeblock segmentation, and in the case of LDPC and polar codes, the update function also performs LDPC lifting, and construct the appropriate polar code. This function has to be called every time the code rate or the output code length changes. The rest of the object usage is straightforward, the encoding and decoding is carried out through

```

1 codedBits = ChannelCodingObj.encode(inputBits);
2 ...
3 decodedBits = ChannelCodingObj.decode(channelLLRs);

```

The parameter **SoftBufferRatio** controls how much of the circular buffer is used (i.e., soft rate matching). Its value ranges from 0 to 1, with 1 indicating that the full circular buffer is used.

10 Modulation Waveforms

10.1 Orthogonal Frequency-Division Multiplexing

CP-OFDM (CP-OFDM) is the most prominent multicarrier scheme and is applied, for example, in Wireless LAN and LTE-A. CP-OFDM employs a rectangular transmit and receive pulse, which greatly reduce the computational complexity. Furthermore, the CP implies that the transmit pulse is slightly longer than the receive pulse, preserving orthogonality in frequency selective channels. Thus, frequency-selective broadband channels transform into multiple, virtually frequency flat, sub-channels (subcarriers) without interference. This allows the application of simple one-tap equalizers, corresponding to maximum likelihood symbol detection in case of Gaussian noise. Furthermore, the channel estimation process is simplified, adaptive modulation and coding techniques become applicable, and MIMO can be straightforwardly employed. Unfortunately, the rectangular pulse in OFDM leads to high out-of-band emissions. This is one of the biggest disadvantages of CP-OFDM. Additionally, the CP simplifies equalization in frequency-selective channels but also reduces the spectral efficiency. In order to reduce the OOB emissions, 3GPP is currently considering windowing and filtering, see the next subsections.

One of our most important implementation aspects is that we consider a fixed sampling rate f_s instead of a fixed Fast Fourier Transform (FFT) length N_{FFT} , as often done in literature. The main reason for a fixed sampling rate is to enable a fair comparison between different subcarriers spacings and to guarantee that a specific channel power delay profile fits approximately the sampling rate. Additionally, the sampling rate is often predetermined by real world hardware and cannot be changed easily. The relationship between FFT size, sampling rate and subcarrier spacing F is:

$$N_{\text{FFT}} = \frac{f_s}{F}. \quad (10.1)$$

Note that the FFT size must be larger or equal than the number of active subcarriers. In practice, the FFT size will always be larger than the number of active subcarriers. For example, in 10 MHz LTE-A, we have 600 active subcarriers and an FFT size of 1024. We also advise to use a (much) larger FFT size than the number of active subcarriers.

The OFDM object can be initialized by:

```

1 OFDMObject = Modulation.OFDM(...)
2 L,...      % Number of subcarriers
3 K,...      % Number OFDM symbols in time

```

```

4  F,...      % Subcarrier spacing (Hz)
5  fs,...     % Sampling rate (Samples/s)
6  fI,...     % Intermediate frequency of the 1st subcarrier (Hz)
7  false,...  % Transmit real valued signal, true/false
8  TCP, ...   % Length of the cyclic prefix (s)
9  TZG ...    % Length of the zero guard time (s), (frame)
10 );

```

whereas we always consider a block transmission of L subcarriers and K OFDM symbols. The intermediate frequency f_I corresponds to a circular shift of the FFT.

A given symbol vector $\mathbf{x} \in \mathbb{C}^{L \times K}$, for example chosen from a QAM signal constellation, can then be modulated by:

```

1  s = OFDMObject.Modulation(x);

```

where \mathbf{s} represents the transmitted signal in time. The demodulation, on the other hand, can be performed by applying the following method on the received time signal \mathbf{r}

```

1  y = OFDMObject.Demodulation(r);

```

In case of a back-to-back transmission, $\mathbf{r} = \mathbf{s}$, we will recover the transmitted data symbols, that is, $\mathbf{y} == \mathbf{x}$.

For a concrete implementation of our OFDM object (without unnecessary overhead), we refer to the example file in “Example/Comparison_5GWaveforms.m”.

10.2 WOLA

The windowed OFDM scheme is called OFDM with WOLA [12]. At the transmitter, the edges of the rectangular pulse is replaced by a smoother function (windowing) and neighboring WOLA symbols overlap in time. The receiver also applies windowing but the overlapping and add operation is performed within the same WOLA symbol, reducing the inter-band interference. Fig. 7 illustrates the WOLA concept. Compared to CP-OFDM, the time spacing is increased by $T_{w,tx} + T_{w,rx}$. However, the CP can usually be reduced because some small interference is often acceptable.

The WOLA object can be initialized by:

```

1  WOLAobject = Modulation.WOLA(...
2  L,...      % Number of subcarriers
3  K,...      % Number OFDM symbols in time
4  F,...      % Subcarrier spacing (Hz)
5  fs,...     % Sampling rate (Samples/s)
6  fI,...     % Intermediate frequency of the 1st subcarrier (Hz)

```

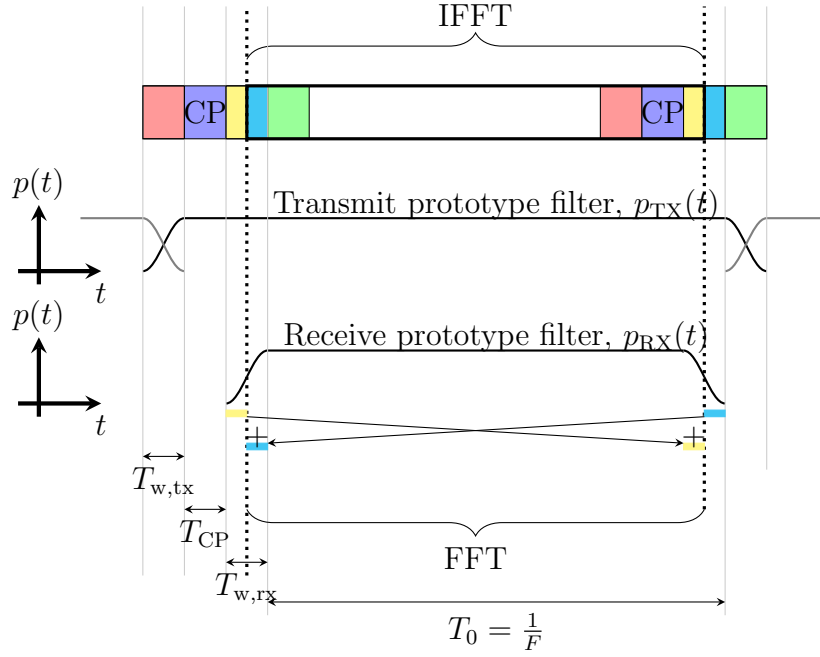


Figure 7: Illustration of WOLA

```

7  false,... % Transmit real valued signal, true/false
8  TCP, ... % Length of the cyclic prefix (s)
9  TZG ... % Length of the zero guard time (s), (frame)
10 TWTX, ... % Length of the window overlapping (s) at the TX
11 TWRX ... % Length of the window overlapping (s) at the RX
12 );

```

It works similarly as the OFDM object, see Section 10.1, but has the additional option of $T_{w,tx}$ and $T_{w,rx}$, representing the window length at the transmitter and at the receiver. The window function itself is based on a (root) raised cosine function.

10.3 Universal Filtered Multicarrier

UFMC is filtered OFDM technique proposed as one of the candidates for new 5G waveforms below 6 GHz. The main advantage of this scheme is a better fragmentation of spectrum and more suppressed side lobes compared to OFDM.

The implementation of UFMC in this code is based on Nokia's proposal of transceiver structure.[13]

10.3.1 Transmitter

The assigned bandwidth in UPMC is divided into multiple subbands according to different user requirements and services.

```

1 for m = 1:obj.Nr.ResourceBlocks
2   n = (m-1)* obj.Nr.SubcarriersPerRb+1:(m-1)*obj.Nr.
      SubcarriersPerRb+obj.Nr.SubcarriersPerRb;
3   b = reshape(DataSymbolsTemp(n,:),obj.Nr.SubcarriersPerRb,obj.
      Nr.MCSymbols);
4 end

```

On each of those subbands we apply Inverse Fast Fourier Transform (IFFT) with corresponding length. By doing this we obtain the transmit data in time domain. The choice of the transmit window is arbitrary. We choose Hanning window since it maximizes the side lobe attenuation. Before we do subband filtering, we shift each filter to the center frequency of the corresponding subband:

```

1 freqShift (l) = exp(2*pi*1i*(l-1)*(centralFreq-1)/obj.
      Implementation.FFTSize);

```

We apply this subband filter on the transmit data from the same subband. Unlike OFDM, UPMC uses Zero-Postfix (ZP) instead of CP in order to avoid inter-symbol interference in a case of high delay spread channels. The length of ZP is chosen to be one sample shorter than the filter length. In order to obtain the total transmit signal we summarize all subbands transmit signals together.

10.3.2 Receiver

At the receiver side we apply N-FFT, resulting in the same complexity level as CP-OFDM. In order to apply N-FFT we do some modifications of the received signal. First, we decompose our received signal into two parts, so called *body* and *tail*. Then we transform this received vector by copying the *tail* to the beginning of the signal:

```

1 receiveSignalTemp(1:guardLength,:) = ReceivedSignalResh(1:
      guardLength,:) + ReceivedSignalResh(obj.Implementation.
      FFTSize+1:end,:);

```

10.4 Filtered-OFDM

The second filter-based OFDM scheme considered within 3GPP is f-OFDM [14]. Here, the number of subcarriers for one subband is usually much higher than

in UFMC and often includes all subcarriers belonging to a specific use case. The idea of f-OFDM is quite simple: we modify a conventional CP-OFDM transmission by applying digital filtering at both, transmitter and receiver. If the total CP length is longer than the combined filter length, we restore orthogonality in an AWGN channel. However, some (small) interference is usually acceptable to keep the overhead low. The induced interference can be adjusted by the filter length and the CP length ($T_{CP,f}$). The filter itself is based on a sinc pulse (perfect rectangular filter) which is multiplied by a Hann window; other filters are also possible [14], but currently not implemented.

The FOFDM object can be initialized by

```

1 FOFDMobject = Modulation.FOFDM(...
2   L,...      % Number of subcarriers
3   K,...      % Number OFDM symbols in time
4   F,...      % Subcarrier spacing (Hz)
5   fs,...     % Sampling rate (Samples/s)
6   fI,...     % Intermediate frequency of the 1st subcarrier (Hz)
7   false,...  % Transmit real valued signal, true/false
8   TCP, ...   % Length of the cyclic prefix (s)
9   TZG ...    % Length of the zero guard time (s), (frame)
10  TFTX, ...  % Length of the transmit filter (s)
11  TFRX, ...  % Length of the receive filter (s)
12  TCPF ...   % Length of the additional cyclic prefix (s).
13 );

```

Again, this is similar to OFDM, see Section 10.1, but with the additional option of $T_{f,tx}$ and $T_{f,rx}$, representing the filter length at the transmitter and at the receiver. Furthermore, we have an additional CP with length $T_{CP,f}$ to combat the effects of filtering. To total CP overhead is then given $T_{CP,total} = T_{CP} + T_{CP,f}$.

10.5 FBMC

Although 3GPP decided that FBMC will not be employed in 5G [15], FBMC still has many advantages over OFDM, namely, much lower OOB emissions and a maximum symbol density (i.e., no CP overhead) [16]. Those advantages, however, come at the price of sacrificing the complex orthogonality condition with the less strict real orthogonality condition. In many cases, however, this has either no, or only a minor influence on the performance. In other cases, such as channel estimation or some MIMO methods, on the other hand, special treatment of the imaginary interference becomes necessary. Fortunately, there exists many efficient methods to deal with those challenges [16]. The signal generation in FBMC is similar to that of windowed OFDM, see Fig. 8, whereas we ignored receive filtering to keep the illustration simple.

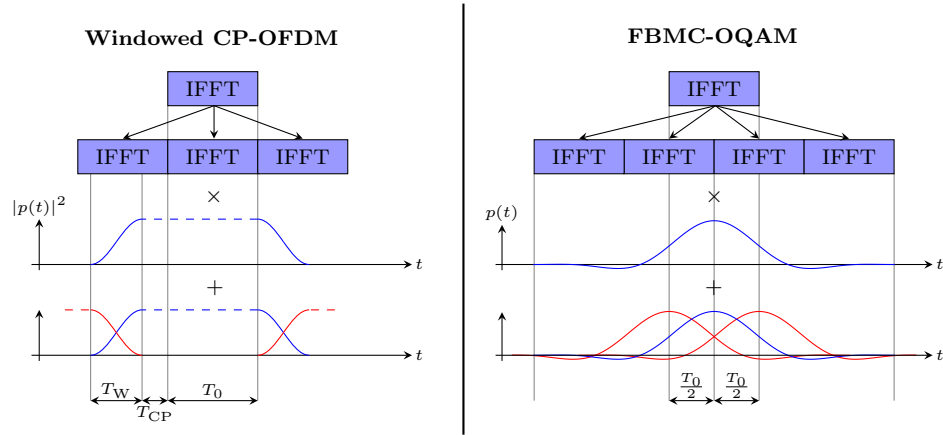


Figure 8: The signal generation in FBMC and windowed OFDM requires the same basic steps [16].

The initialization of the object is similar to OFDM:

```

1 FBMC = Modulation.FBMC(...)
2 L,...    % Number of subcarriers
3 K,...    % Number FBMC symbols in time
4 F,...    % Subcarrier spacing (Hz)
5 fs,...   % Sampling rate (Samples/s)
6 fI,...   % Intermediate frequency of the 1st subcarrier (Hz)
7 false,... % Transmit real valued signal, true/false
8 'PHYDYAS-OQAM',... % Prototype filter (Hermite, PHYDYAS, RRC)
9 0, ...   % Overlapping factor, e.g., 3,4,5,6,7,8
10 0, ...  % Initial phase shift, e.g., pi/2
11 true ... % Polyphase implementation, true/false
12 );

```

11 Feedback

In wireless communications, Channel State Information at the Transmitter (CSIT) is needed to adapt the transmission to the current channel condition, in order to achieve a better performance. The receiver has to feedback the Channel State Information (CSI) to the transmitter. Limited feedback is employed to reduce the overhead. The receiver has to feed back the CQI, the Rank Indicator (RI) and the Precoding Matrix Indicator (PMI). The CQI informs the transmitter about the MCS suitable for the current channel conditions. The RI informs the transmitter about the number of useful spatial streams, and the PMI represents the codebook index of the pre-coding matrix.

11.1 Feedback Calculation

The feedback calculation is based on [17, 18]. In this scheme the optimal PMI and RI are calculated jointly. Once a pre-coding matrix is determined, the optimal CQI is calculated.

11.1.1 PMI and RI

In the codebook-based PMI, the receiver performs an exhaustive search, to find the pre-coding matrix $\mathbf{W}_i \in \mathcal{W}$ which maximizes the sum mutual information $I_{k,n}$ over all resource elements (11.1). The mutual information calculation is based on the post-equalization **SINR** (11.2).

$$\mathbf{W}_i = \arg \max_{\mathbf{W}_j \in \mathcal{W}} \sum_{k=1}^K \sum_{n=1}^N I_{k,n}(\mathbf{W}_j) \quad (11.1)$$

$$I_{k,n} = \sum_{l=1}^L \log_2(1 + \mathbf{SINR}_{k,n,l}) \quad (11.2)$$

11.1.2 CQI

The CQI calculation is based on Effective Signal to Interference and Noise Ratio Mapping (ESM). The post-equalization Signal to Interference and Noise Ratio (SINR) of all scheduled resource elements is mapped to an equivalent SNR value of a SISO AWGN channel, the CQI is then chosen, so that it has the highest value with Block Error Ratio (BLER) < 0.1 for the equivalent AWGN channel.

11.2 Object Usage

11.2.1 Scenario File

To run a simulation with feedback, the following lines have to be included in the scenario file:

```

1 % Feedback Parameters
2 scStr.feedback.delay          = 1;
3 % Feedback Delay
4 scStr.feedback.averager.Type  = 'miesm';
5 % 'eesm', 'miesm'
6
7 % for the custom transmission mode the following parameters
  are used to configure the feedback
8 scStr.feedback.enable        = true;
9
10 % when the feedback is enabled the following parameters are
   used to configure the individual indicators:
11 scStr.feedback.pmi           = true;
12 scStr.feedback.ri            = true;
13 scStr.feedback.cqi           = true;

```

In line 2 the feedback delay is set; a value of 1 equals a delay of 1 frame duration. A delay larger than 0, only makes sense when subsequent channels are temporarily correlated. The type of averager is chosen in line 4. Line 8 to 13 are relevant for the custom transmission mode, for the Closed Loop Spatial Multiplexing (CLSM) and Open Loop Spatial Multiplexing (OLSM) the feedback parameters are set automatically and therefore these lines are irrelevant. To enable the feedback, line 8 is set to true. By setting line 11 to true, the PMI and the RI are activated (since the PMI and RI are calculated jointly). To activate the CQI feedback, line 13 has to be set to true. If the feedback is not enabled, line 11 to 13 are ignored.

11.2.2 Object Generation

A feedback object can be initialized as follows:

```

1 FeedbackObj = Feedback.Feedback(...
2             codebook,...
3             feedback,...
4             nTXAntennas,...
5             nRXAntennas,...
6             precodingMatrix,...
7             mcs);

```

where codebook is a cell array containing all pre-coding matrices with nTX-Antennas transmit antennas, and feedback is a structure as described in Sec-



Fig. 9 shows how a transmission works. After the channel is generated, the feedback is calculated. If the delay is zero - which means that the channel has to be known before transmitting - the newly generated channel is used for feedback calculation. If the delay is larger than zero, the channel estimated at the receiver in the previous transmission is used for feedback calculation. The buffer between the feedback calculation and the transmission takes into account the delay and the size of the buffer equals the delay.

[illegible]

```
modulationOrder]);
```

the `pilotMatrix` determines the Resource Element (RE) which are used for the feedback calculation ³, `simParams.phy.noisePower` is the noise power, and the last input argument is the modulation order of all MCS, defined in `SimulationParameters`. After the feedback is calculated, the feedback values can be accessed as follows:

```
1 PMI = obj.Feedback.Pmi.pmiArray(1);
2 RI  = obj.Feedback.Ri.riArray(1);
3 CQI = obj.Feedback.Cqi.cqiArray(1);
```

For the transmission, where no feedback is available due to the feedback delay, the receiver sets the value of the PMI, RI, and CQI to 1.

³Note that, in order to reduce the simulation time, the SINR is only calculated at the pilot positions.

12 Non-Orthogonal Multiple Access (NOMA)

NOMA is a key technology for next generation communications systems. It allows the users to access the available resources in a non-orthogonal manner, and in turn allowing the system to accommodate more users compared to the case of OMA. This would of course require employing advanced receivers in order to cope with the induced interference between those users. On top of that, many NOMA schemes fit naturally in the context of grant-free access, allowing the users to access the resources more often and therefore reducing the latency. These two aspects; massive connectivity and low latency operation, are main components of the future systems, and NOMA is able to tackle both of these issues in a natural manner.

In the current version of the simulator, we support the 3GPP MUST technique. It is a downlink power-domain NOMA version that works by superimposing a maximum of two users on the same resources in the power-domain. The gain provided by this scheme is maximized when the two imposed users have a large difference in their channel quality, i.e., a user with good channel conditions (NearUE) and a user with bad channel condition (FarUE). A typical example for a FarUE would be a cell-edge user. In the simulator, the notion of strong and weak users can be controlled through `scStr.simulation.pathloss` parameter. The superposition works by assigning the FarUE with most of the transmit power. Then, at the receiving side, Successive Interference Cancellation (SIC) can be used to first detect the high power user, subtract its signal from the total received signal, and then proceed to detect the low power user. Alternatively, one can view the superimposed signal as just a normal signal with symbols being drawn from a super composite constellation. This in turn allows us to perform the detection using a Maximum Likelihood (ML) detector running on the composite constellation. In the standard, FarUE is limited to 4 QAM, while NearUE can use up to 64 QAM. This further simplifies the detection process for the FarUE, since it can treat the received superimposed signal as legacy 4 QAM and proceed to detect its signal as if NearUE is just an extra noise. On the other hand, this also means that the BS does not need to transmit control information to the FarUE about the MUST operation, which is always a good thing. Furthermore, the standard defines three power ratios that control the allocation of the power between the two superimposed users. This can be controlled using the `MUSTIdx` parameter. The default value in the simulator is 2, which is the middle power ratio.

MUST is enabled by scheduling two users at the same resources. For example, assume you would like to have UE1 and UE2 to operate in MUST mode. They need to be scheduled in the following way

```
1 scStr.schedule.fixedScheduleDL{1} = ['UE1:72,UE2:UE1'];
```

This way, UE1 will be the NearUE and UE2 is the FarUE.

When it comes to producing results with MUST, it is recommended to use the transmit power of the BS as the sweeping parameter. The path loss parameter can then be used to set the quality of the user channels, as shown in the simulation scenario of Section 4.2.5.

13 Releases and Changelog

2. Vienna 5G LL Simulator **1.1**, released June 2018.

Newly introduced features:

- introduce various input parameter checks for improved usability
- introduce time correlated Rayleigh fading channel
- implement spatial correlation for MIMO channels according to TS36.101 Annex B
- implementation of NOMA (3GPP MUST)
- implement feedback and transmission modes CLSM and OLSM
- support for 256 QAM with feedback
- TWDP fading model for static channels
- non-linear power amplifier model
- add PAPR (signal power CCDF) as simulation result
- add channel estimation mean squared error as simulation result
- new channel coding algorithms (additional decoding algorithms)

Bug fixes and improvements:

- make LLR value calculation numerically robust
- bugfix in the transmitter signal generation
- bugfix in the superposition of signals in the channel
- bugfix in the automatic sampling rate calculation
- bugfix in the parameter check for FBMC support

1. Vienna 5G LL Simulator **1.0**, released June 2017.

Acknowledgements

The financial support by the Austrian Federal Ministry of Science, Research and Economy, the National Foundation for Research, Technology and Development, and by TU Wien is gratefully acknowledged. This work has been co-financed by A1 Telekom Austria AG, Kathrein-Werke KG and Nokia Solutions and Networks.



KATHREIN

NOKIA

References

- [1] 3rd Generation Partnership Project (3GPP). *Technical Specification Group Radio Access Network; NR; Physical channels and modulation*. TS 38.211. 3rd Generation Partnership Project (3GPP), Dec. 2017.
- [2] 3rd Generation Partnership Project (3GPP). *Technical Specification Group Radio Access Network; Study on channel model for frequencies from 0.5 to 100GHz*. TR 38.901. 3rd Generation Partnership Project (3GPP), Dec. 2017.
- [3] 3rd Generation Partnership Project (3GPP). *Technical Specification Group Radio Access Network; High Speed Downlink Packet Access: UE Radio Transmission and Reception*. TR 25.890. 3rd Generation Partnership Project (3GPP), May 2002.
- [4] Technical Specification Group Radio Access Network). *Evolved Universal Terrestrial Radio Access: Base Station radio transmission and reception*. TR 36.104. 3rd Generation Partnership Project (3GPP), Dec. 2017.
- [5] Gregory D Durgin, Theodore S Rappaport, and David A De Wolf. “New analytical models and probability density functions for fading in wireless communications”. In: *IEEE Transactions on Communications* 50.6 (2002), pp. 1005–1015.
- [6] 3rd Generation Partnership Project (3GPP). *Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding*. TS 36.212. 3rd Generation Partnership Project (3GPP), Dec. 2017.
- [7] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. “Optimal decoding of linear codes for minimizing symbol error rate (Corresp.)” In: *IEEE Transactions on Information Theory* 20.2 (Mar. 1974), pp. 284–287. ISSN: 0018-9448. DOI: 10.1109/TIT.1974.1055186.
- [8] 3rd Generation Partnership Project (3GPP). *Technical Specification Group Radio Access Network; NR; Multiplexing and channel coding*. TS 38.212. 3rd Generation Partnership Project (3GPP), Apr. 2018.
- [9] D. J. C. MacKay. “Good error-correcting codes based on very sparse matrices”. In: *IEEE Transactions on Information Theory* 45.2 (Mar. 1999), pp. 399–431. ISSN: 0018-9448. DOI: 10.1109/18.748992.
- [10] P. Radosavljevic, A. de Baynast, and J. R. Cavallaro. “Optimized Message Passing Schedules for LDPC Decoding”. In: *Conference Record of the Thirty-Ninth Asilomar Conference on Signals, Systems and Computers*. Oct. 2005, pp. 591–595. DOI: 10.1109/ACSSC.2005.1599818.

-
- [11] Bashar Tahir and Markus Rupp. “New construction and performance analysis of Polar codes over AWGN channels”. In: *24th International Conference on Telecommunications (ICT)*. May 2017, pp. 1–4. DOI: 10.1109/ICT.2017.7998250.
 - [12] Qualcomm Incorporated. “Waveform Candidates”. In: *3GPP TSG-RAN WG1 84b*. Busan, Korea, Apr. 2016.
 - [13] “R1-165014, Subband-wise filtered OFDM for New Radio below 6 GHz”. In: *3GPP TSG-RAN WG1 85* (May 2016).
 - [14] Xi Zhang, Ming Jia, Lei Chen, Jianglei Ma, and Jing Qiu. “Filtered-OFDM-Enabler for Flexible Waveform in The 5th Generation Cellular Networks”. In: *IEEE Global Communications Conference (GLOBECOM)*. 2015, pp. 1–6.
 - [15] Technical Specification Group Radio Access Network 3rd Generation Partnership Project (3GPP). *Study on New Radio Access Technology; Physical Layer Aspects*. TR. 3rd Generation Partnership Project (3GPP), Mar. 2017.
 - [16] Ronald Nissel, Stefan Schwarz, and Markus Rupp. “Filter bank multicarrier modulation schemes for future mobile communications”. In: *IEEE Journal on Selected Areas in Communications* 35.8 (2017), pp. 1768–1782.
 - [17] Stefan Schwarz, Christian Mehlführer, and Markus Rupp. “Calculation of the spatial preprocessing and link adaption feedback for 3GPP UMTS/LTE”. In: *6th conference on Wireless advanced (WiAD)*. IEEE. 2010, pp. 1–6.
 - [18] Stefan Schwarz, Martin Wrulich, and Markus Rupp. “Mutual information based calculation of the precoding matrix indicator for 3GPP UMTS/LTE”. In: *International ITG Workshop on Smart Antennas (WSA)*. IEEE. 2010, pp. 52–58.