Vienna LTE Simulators
LTE-A Link Level Simulator Documentation, v1.3

Institute of Telecommunications
Vienna University of Technology, Austria
Gusshausstrasse 25/389, A-1040 Vienna, Austria
Email: {sschwarz, jcolom, msimko}@nt.tuwien.ac.at
Web: http://www.nt.tuwien.ac.at/ltesimulator

Abstract

This document contains documentation on how to use the Long Term Evolution Advanced (LTE-A) Link Level simulator [1] from the Vienna LTE simulator suite, as well as some insight on its structure and the assumptions that were made while developing it. This simulator is an extension of the previously published Vienna LTE Link Level Simulator. This document gives an introduction on how to use the simulator. The concept and the structure of the simulator is described in more detail in [2].

I. Foreword

The LTE-A link level simulator is published under a non-commercial academic use license. Please make sure that you understand the terms and conditions of the license before you use any of the available software packages. Would you require a license different to a non-commercial academic one please contact Stefan Schwarz, Michal Simko or Josep Colom Ikuno.

The detailed license agreement for the LTE-A Link Level simulator can be found in Section XVIII. Please read the license agreement carefully, as parts of the code are under the GNU Lesser General Public License [3], and the MIT License [4].

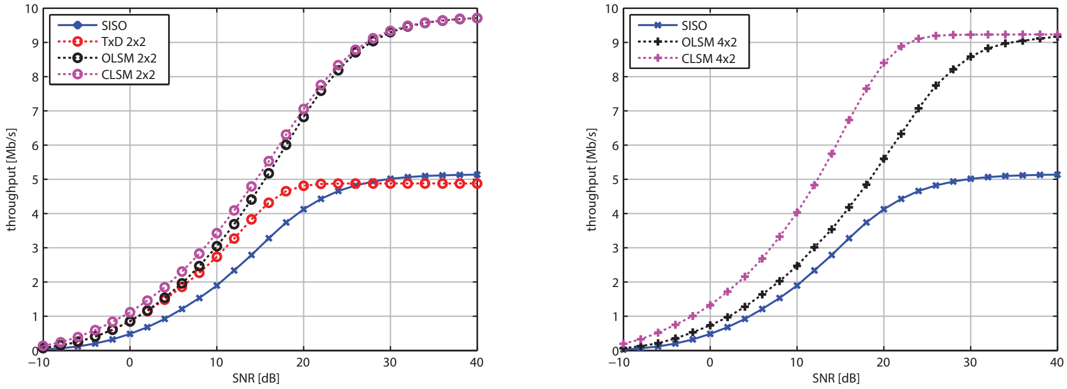
Contents

II. Running the simulator for the first time

The LTE-A Link Level simulator is provided with several test simulation scenarios that can be used to verify that the simulator runs in the expected way.

A. LTE

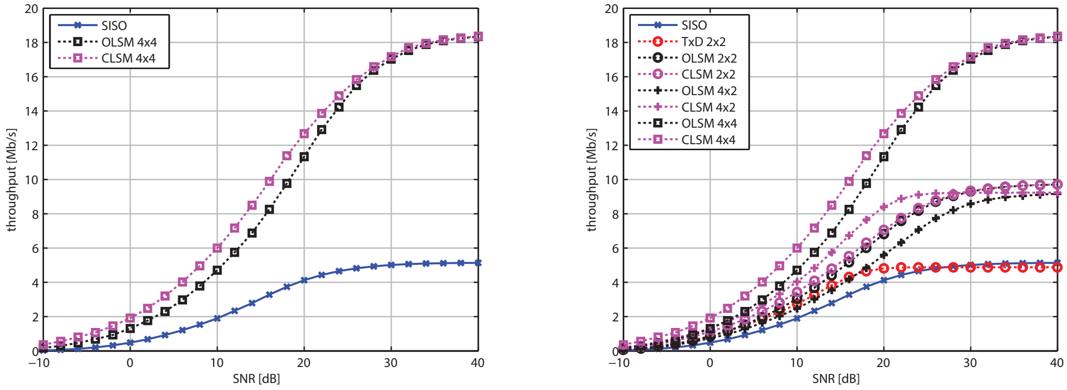
In the `./examples` directory you can execute the `LTE_sim_batch_quick_test` MATLAB script. This script provides a means to test the performance of a single user Long Term Evolution (LTE) transmission on an uncorrelated VehA channel for several transmission modes [5] and antenna configurations (number of transmit antennas N_t times number of receive antennas N_r). All simulated transmission modes (Single-Input Single-Output (SISO), Transmit Diversity (TxD), Open Loop Spatial Multiplexing (OLSM) and Closed Loop Spatial Multiplexing (CLSM)) utilize UE feedback to adapt important transmission parameters (code rate, modulation alphabet, Multiple-Input Multiple-Output (MIMO) preprocessing) to the channel quality (see [6] for details). The script produces plots of the cell throughput obtained with the different transmission modes. To verify their correctness, these plots can be compared to the figures provided in the subdirectory `./LTE quick test results` of the directory `./examples` or to the curves shown in Figure 2. The result of the simulation is a group of result files that is stored in the `./results` folder. Details to the simulation setup can be found in [7].



(a) 2×2 antenna configuration.

(b) 4×2 antenna configuration.

Fig. 1: Plots resulting from running the `LTE_sim_batch_quick_test.m`, MATLAB script. 1.4 MHz, PedA channel, zero-delay feedback, single user. 2×2 (left) and 4×2 (right) antenna configuration.



(a) 4×4 antenna configuration.

(b) Complete set of antenna configurations.

Fig. 2: Plots resulting from running the `LTE_sim_batch_quick_test.m` MATLAB script. 1.4 MHz, PedA channel, zero-delay feedback, single user. 4×4 (left) antenna configuration and aggregate plot containing all of the simulated antenna configurations (right).

The most important settings used in the `LTE_sim_batch_quick_test` MATLAB script can be found in Table I.

Parameter	Value
Number of UEs	1
Bandwidth	1.4 MHz
HARQ Retransmissions	0
Uplink delay	none
Channel type	Ped-A uncorrelated
Filtering	Block Fading
Receiver type	Zero Forcing
Simulation length	1000 subframes
Transmit modes	SISO, TxD 2×2, OLSM 2×2, 4×2, 4×4, and CLSM 2×2, 4×2, 4×4

TABLE I: Basic settings used for the `LTE_sim_batch_quick_test.m` MATLAB script

	SUMIMO settings	MUMIMO settings
Parameter	Value	Value
Number of UEs	1	5
Bandwidth	1.4 MHz	1.4 MHz
HARQ Retransmissions	0	0
Uplink delay	none	none
Channel type	TU uncorrelated	flat Rayleigh uncorrelated
Filtering	Block Fading	Block Fading
Receiver type	Zero Forcing	Zero Forcing
Simulation length	1000 subframes	1000 subframes
Transmit modes	CLSM 8×1, 8×4 and 8×8	Non-standard defined precoding (ZF beamforming) 4×1
Precoder feedback	Standard defined codebook	Perfect channel knowledge and RCVQ

TABLE II: Basic settings used for the `LTEA_sim_batch_quick_test_SUMIMO.m` and `LTEA_sim_batch_quick_test_MUMIMO.m` MATLAB scripts

B. LTE-A

To test the LTE-A capabilities of the simulator, the two MATLAB scripts `LTEA_sim_batch_quick_test_SUMIMO` and `LTEA_sim_batch_quick_test_MUMIMO` are provided. The SUMIMO test script compares the single user throughput performance of different LTE-A antenna configurations (eight transmit antennas), when utilizing the standard defined precoder codebook [8]. Again the transmission parameters are adapted in accordance with the channel quality experienced and fed back by the UE (assuming instantaneous feedback), see [9] for details. In Figure 3, the SUMIMO throughput for three different antenna configurations is shown. Running the script file `LTEA_sim_batch_quick_test_SUMIMO` should produce similar results. The simulation settings utilized in these simulations are given in the second column of table Table II.

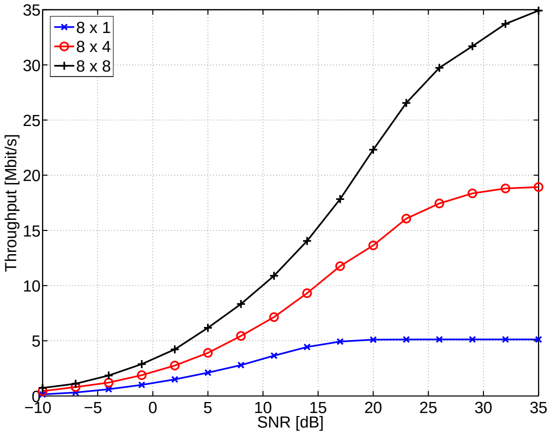


Fig. 3: Plots resulting from running the `LTEA_sim_batch_quick_test_SUMIMO.m` MATLAB script

In addition to standard defined precoding, LTE-A also supports non-standard defined precoding. As an example, Zero Forcing (ZF) beamforming for single receive antenna UEs is implemented in the LTE-A simulator. This precoding method spatially orthogonalizes the transmission to different UEs, by appropriately forming antenna beams. The precoders are either computed from perfect channel knowledge, or utilizing Channel Vector Quantization (CVQ), from quantized (imperfect) channel knowledge. For details on the utilized feedback and precoding methods, see [10]. Figure 4 shows the obtained cell throughput versus SNR for the two considered feedback methods, assuming a frequency flat Rayleigh fading channel model. Note that no precoder interpolation is currently implemented. Therefore the precoding method will not perform well in frequency selective channels, because only one channel value per Resource Block (RB) is fed back. Running the script `LTEA_sim_batch_quick_test_MUMIMO` in the `./examples` directory allows to reproduce these results. The most important simulator settings for these simulations are summarized in the last column of table Table II.

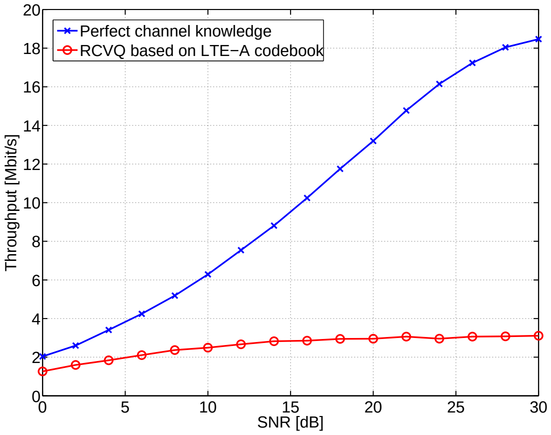


Fig. 4: Plots resulting from running the `LTEA_sim_batch_quick_test_MUMIMO.m` MATLAB script

III. MEX FILES

Computation-intensive parts of the LTE-A link level simulator are implemented in C and used in the simulator by means of MEX files. The source code for the given files can be found in the `./C-source` folder under the simulator root folder. Please note that some of the functions there are licensed under other license terms. Please check Section XVIII for more details.

The MEX files distributed with the simulator release are the Windows 64-bit version. Should you require them for any other platform you recompile them by using the `LTE_aux_mex_files` script. You can find more information on how to use/write/compile MEX files here.

IV. RUNNING YOUR OWN SIMULATIONS

While the `LTE_sim_batch_quick_test`, `LTEA_sim_batch_quick_test_SUMIMO` and `LTE_sim_batch_quick_test_MUMIMO` MATLAB scripts are adequate to get familiar with the basic structure of the simulator, you may need more flexibility than the one offered with the quick test scripts.

For this purpose, you can check the `LTE_sim_batch.m` script. It provides the means to set basic simulation parameters (SNR range, number of subframes simulated, choice of simulation configuration) and calls all necessary scripts in the appropriate order.

Below you can find a list of exemplary parameters that you may want to configure in the batch file:

- `cqi_i`: set of Modulation and Coding Schemes (MCSs) that are used for the simulation. In [11], 15 different Channel Quality Indicators (CQIs) are specified. If you want to simulate with all possible CQIs, just set the `cqi_i` loop to run over `[1:15]`. Note: depending on the settings you choose in `LTE_load_parameters`, the appropriate CQI value for the current channel conditions may be automatically determined and set during simulation.
- `N_subframes`: the length of the simulation, or how many subframes (Transmission Time Intervals (TTIs)) are simulated for each value of `cqi_i` and `SNR_vec`.
- `SNR_vec`: a vector containing the average Signal to Noise Ratios (SNRs) that will be used for each simulation run. All users will experience the same average SNR. See Section IX for a more detailed definition of the SNR. Alternatively, for simulations with multiple UEs, you can set `SNR_vec` as a matrix, where each row then corresponds to the SNR values of a single user. Note: the SNR matrix is then simulated column-wise, meaning that in each simulation run, a column defines the current average SNRs of the different users.
- `LTE_load_parameters`: load the parameter file that configures the simulator. A more detailed description of the available configuration parameters can be found in Section V. Six basic preconfigured options are given that can be directly used by appropriately setting the value of the variable `Simulation_type`, thus skipping the configuration of the rest of the parameters.
  1) `SUSISO`: Single-user SISO simulation
  2) `MUSISO`: SISO simulation with multiple users (preconfigured to two users)
  3) `SUMIMO`: Single-user MIMO simulation utilizing CLSM

4) MUMIMO: Multi-user (four users) MIMO simulation. In this simulation, the columns of the standard defined precoding matrices are shared by different users in a round robin fashion (no sophisticated scheduler is currently implemented). The number of transmit and receive antennas equals 4.
5) LTE-A MUMIMO: Non-standard defined precoding (ZF beamforming) based MUMIMO with 5 single antenna users and four transmit antennas.
6) LTE-A SUMIMO: SUMIMO based on the LTE-A specified single user codebook (enables up to eight layers for a single user).
7) IA: Interference Alignment simulation based on the closed-form IA expression for a 2×2 3 user system. Note that some parameters are confined for IA, e.g. a feedback delay is not allowed and only frequency flat simulations are possible (due to speed problems). Details about this setup can be found in "Interference Alignment in UMTS Long Term Evolution," J. Reitterer et.al.
8) DAS: This setting allows to simulate a distributed antenna system. Note that it overrules several parameters, because it uses a pathloss and shadow fading model. Details about this setup can be found in "Multiuser MIMO in Distributed Antenna Systems with Limited Feedback," S. Schwarz et.al.

- LTE_sim_main: main routine of the simulator.
- Generate the output filename and save the results in a .mat file.

V. Simulation parameters

Below you can find a list of the parameters that can be configured in LTE_load_parameters and LTE_load_advanced_parameters (or in files called therein)

A. General parameters

- LTE_params.nUE: number of UEs to simulate.
- LTE_params.nBS: number of eNodeBs (cells) that will be simulated.
- LTE_params.uplink_delay: the delay the uplink experiences. It applies to ACKnowledgments (ACKs), CQI, Precoding Matrix Indicator (PMI) and Rank Indicator (RI) reports. An integer number ≥ 0. Useful if you want to experiment with scheduling algorithms or feedback strategies. It is also possible to define independent uplink delays for different feedback values. See Section V-F for a more detailed description of this feature.
- LTE_params.show_plots: whether plots are shown during the simulation or not.
- LTE_params.trace_subcarrier_SNR: if set to true, a trace of the subcarrier SNRs is generated and stored in the simulation_results object.
- LTE_params.N_seed_reset: resets the random number generator seeds to a new value after LTE_params.N_seed_reset subframes. This is used for the case where a time-correlated channel is generated (modified Rosa Zheng model [12], [13]). When low speed channels are evaluated, a too-long simulation would be needed in order to obtain statistically meaningful results. Thus, to avoid such situations, the channel can be "reset" to a new seed every N subframes.
- LTE_params.carrier_freq: carrier center frequency [Hz]
- LTE_params.Bandwidth: system bandwidth. Allowed values are 1.4 MHz, 3 MHz, 5 MHz, 10 MHz, 15 MHz, and 20 MHz. This bandwidths are equivalent to 6, 15, 25, 50, 75, and 100 RBs respectively. Carrier aggregation (to enable bandwidths > 20 MHz) is currently not supported by the LTE-A simulator.
- LTE_params.HARQ_processes: number of parallel Hybrid-ARQ (HARQ) processes. The maximum value, according to [14] is 8.
- LTE_params.max_HARQ_retransmissions: maximum number of HARQ retransmissions, not including the original transmission. Valid values are 0, 1, 2 or 3. Higher numbers will give you an error, as the rate matching is not defined for retransmission numbers (rv_idx) higher than 3.
- LTE_params.SubcarrierSpacing: in Hz, 15 kHz, a 7.5 kHz subcarrier spacing is also possible (just for MBSFN-based multicast/broadcast transmissions). Tests were so far performed using a 15 kHz spacing, so the 7.5 kHz spacing is not thoroughly tested.
- LTE_params.CyclicPrefix: cyclic prefix length [15]. Either normal or extended for MBSFN-based multicast/broadcast transmissions.
- LTE_params.simulation_type: the simulator is capable of using the MATLAB Parallel Toolbox in order to speed up simulations by using parfor loops. If you happen to have the Distributed Computing Toolbox, you will also be able to make use of it by using this option. Set this variable to parallel or normal to parallelize the SNR loop in LTE_sim_main or just perform a single-core simulation. Keep in mind that some modifications you do to the code may not work in the parallel version or may directly cause it not to run.
- LTE_params.simulate_with_all_zero_sequences: true if you want that the transmitted data is an all-zero sequence (useful for interleaver testing and debugging).

- LTE_params.introduce_frequency_offset: whether you want a frequency offset to be introduced. Additional carrier offset options are also present in the UE parameters configuration Section V-C.
- LTE_params.random_noise_seeding: whether the seed for the random number generator that generates the noise is set (allows for repeatability of the noise realizations).
- LTE_params.noise_seed: Only used if the upper variable is set to true. Integer number that sets the random number seed of the noise random number generator.
- LTE_params.usePBCH: whether space for the physical broadcast channel is reserved inside the resource grid (no data is transmitted on these resource elements).
- LTE_params.usePDCCH: whether space for the physical downlink control channel is reserved inside the resource grid.
- LTE_params.trafficmodel.usetraffic_model: whether users generate traffic according to prespecified traffic models (RAN R1-070674) or a full buffer situation is assumed. Currently only a round robin scheduler (RoundRobin_Traffic) is supported in combination with the traffic models.

B. Channel matrix source

- LTE_params.channel_matrix_source: Controls the generation of the channel matrix trace. generated to generate it every time. trace to load it from a trace.
- LTE_params.store_channel_trace: Set to true or false. If mode is generated, the channel trace will be saved at the end of the simulation.
- LTE_params.channel_matrix_tracefile: filename of the trace file where the generated channel matrix trace is stored. Only applicable if trace mode is used (if the mode is set to trace, the channel matrix is already read from a trace, so it is meaningless to save it again in another trace).

C. UE, eNodeB and UE feedback specific parameters

- LTE_params.UE_config.LLR_clipping: specifies the LLR clipping level.
- LTE_params.UE_config.turbo_iterations: Number of iterations of the turbo decoder. Set by default to 8.
- LTE_params.UE_config.N_soft: Defines the total number of soft channel bits available for HARQ processing (TS 36.306 4.2.1.3 [16]).
- LTE_params.UE_config.channel_estimation_method: Currently the following channel estimators are available: PERFECT, LS, or MMSE for LTE. LTE-A (transmission modes 8 and 9) only supports perfect channel knowledge at the moment.
- LTE_params.UE_config.channel_interpolation_method: linear, cubic, spline, sinc_freq, sinc_time, or T-F. For fast fading, linear, cubic, and v4 are available.
- LTE_params.UE_config.autocorrelation_matrix_type: type of autocorrelation matrix. Either ideal or estimated.
- LTE_params.UE_config.user_speed: channel speed (m/s).
- LTE_params.UE_config.realization_num: number of channel realizations. Used for averaging to obtain the channel autocorrelation matrix.
- LTE_params.UE_config.realization_num_total: first xy number of channel realizations are used just for the estimation of the autocorrelation matrix.
- LTE_params.UE_config.CDD: Cyclic Delay diversity
  - 0: zero delay CDD (3GPP TS 36.211-820 Section 6.3.4.2.1 [15], page 37)
  - 1: small delay CDD (3GPP TS 36.211-820 Section 6.3.4.2.1 [15], page 37). Although in the newest standard version this is not defined anymore
  - 2: large delay CDD (3GPP TS 36.211-820 Section 6.3.4.2.2 [15], page 38)
- LTE_params.UE_config.mode: the transmission modes are defined in TS 36.213-a01 Section 7.1 [17]. Following transmission modes are implemented (for details see Section VIII):
  - 1: single antenna.
  - 2: TxD .
  - 3: OLSM.
  - 4: CLSM.
  - 5: Multiuser MIMO (only a pseudo implementation is available, no meaningful scheduling).
  - 8: LTE-A non-standard defined precoding (ZF beamforming implemented as an example)
  - 9: LTE-A SUMIMO (up to eight layer transmission).
- LTE_params.UE_config.nRX: number of receive antennas at the UE.
- LTE_params.UE_config.carrier_freq_offset: carrier frequency offset normalized to subcarrier spacing (not yet implemented).

- LTE_params.UE_config.perfect_freq_sync: whether the UE is perfectly synchronized in frequency.
- LTE_params.UE_config.rfo_correct_method: receiver frequency offset correction method. Either none, subframe, or FIR. Not yet implemented.
- LTE_params.UE_config.receiver: either SSD for a Soft Sphere Decoder, ZF for a ZF receiver and MMSE for a Minimum Mean Square Error (MMSE) receiver.
- LTE_params.UE_config.PMI_fb_granularity: granularity of PMI feedback in multiples of resource blocks
- LTE_params.UE_config.CQI_fb_granularity: granularity of CQI feedback in multiples of resource blocks
- LTE_params.UE_config.PMI_fb: whether PMI feedback is activated or not (true/false)
- LTE_params.UE_config.RIandPMI_fb: whether RI feedback is activated in addition to PMI feedback (true/false)
- LTE_params.UE_config.CQI_fb: whether CQI feedback is activated
- LTE_params.UE_config.predict: whether channel prediction is activated for the feedback calculation; the channel prediction algorithm is simple linear extrapolation on every subcarrier;
- LTE_params.UE_config.SINR_averaging.averager: defines the Effective Signal to Interference and Noise Ratio Mapping (ESM) averager used. Possible values are EESM and MIESM.
- LTE_params.UE_config.SINR_averaging.EESMbetas: defines the calibration factors required for Exponential Effective Signal to Interference and Noise Ratio Mapping (EESM). Those values are obtained from extensive training simulations and should not be changed.
- LTE_params.UE_config.SINR_averaging.MIESMbetas: defines the calibration factors required for Mutual Information Effective Signal to Interference and Noise Ratio Mapping (MIESM). Those values are obtained from extensive training simulations and should not be changed.
- LTE_params.UE_config.SINR_averaging.MCSs: defines the used MCS set. Modification of these values might lead to unexpected behavior, as the calibration factors (EESMbetas and MIESMbetas) are not trained for other MCSs.
- LTE_params.BS_config.nTx: number of antennas at the eNodeB.
- LTE_params.feedback.ignore_channel_estimation: whether the channel estimation mean square error is taken into account during the feedback calculation or not. If channel estimation is activated, the option should be set to true, otherwise the performance of the system might be overestimated.
- LTE_params.feedback.channel_averaging: whether channel averaging is used during feedback calculation or not. If set to true just a single average channel value per resource block is used to compute the feedback for complexity reduction. Especially in $\geq 4 \times 4$ systems this degrades the performance of the feedback method.
- LTE_params.feedback.codebook: channel vector quantization codebook used for ZF beamforming (transmission mode 8). PERFECT provides perfect channel knowledge to the eNodeB (subsampled to one value per RB), LTE utilizes the LTE precoder codebook to quantize the channel.
- LTE_params.feedback.codebook_size: size of the codebook (in number of code vectors) if a DFT or RANDOM codebook is employed.
- LTE_params.feedback.CQI_quantization: whether quantized or unquantized channel quality (receive SNR) feedback is provided.

D. Channel model parameters

Parameters that configure how the channel is generated and the signal filtered.

- LTE_params.ChanMod_config.filtering: BlockFading (channel is constant during one subframe) or FastFading (Fast Fading is currently not supported for LTE-A TX modes 8 and 9).
- LTE_params.ChanMod_config.interpolation_method: the channel interpolation method for the channels which are generated in the simulator. Either shift_to_nearest_neighbor for nearest neighbor interpolation or sinc_interpolation for sinc interpolation, which is more precise. Necessary if the channel sampling rate is not equal to the sampling rate of the transmit signal.
- LTE_params.ChanMod_config.sin_num: specifies the number of sin realizations used for the modified rosa-zheng model [12], [13].
- LTE_params.ChanMod_config.type: specifies the type of channel used. The available ones are:
  - AWGN: Additive White Gaussian Noise channel.
  - flat Rayleigh: temporally uncorrelated frequency flat Rayleigh fading channel.
  - flat Rayleigh corr: temporally correlated frequency flat Rayleigh fading channel.
  - Tap-delay based models: PedA, PedB, PedBcorr, VehA, VehB, TU, RA, and HT [18], [19].
  - Externally-generated channel coefficients: winner_II. Uses the publicly-available Winner II implementation to generate the channel coefficients [20]. The following parameters can be configured when using the Winner II channel model.
    - LTE_params.ChanMod_config.winner_settings.Scenario: 1=A1, 2=A2, 3=B1, 4=B2, 5=B3, 6=B4, 7=B5a, 8=B5c, 9=B5f, 10=C1, 11=C2, 12=C3, 13=C4, 14=D1 and 15=D2a.

    - ∗ LTE_params.ChanMod_config.winner_settings.PropagCondition: LOS or NLOS.
    - ∗ LTE_params.ChanMod_config.winner_settings.SampleDensity: number of time samples per half wavelength
    - ∗ LTE_params.ChanMod_config.winner_settings.UniformTimeSampling: use same time sampling grid for all links (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.FixedPdpUsed: nonrandom path delays and powers (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.FixedAnglesUsed: nonrandom AoD/AoAs (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.PolarisedArrays: usage of dual polarised arrays (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.TimeEvolution: usage of time evolution (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.PathLossModelUsed: usage of path loss model (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.ShadowingModelUsed: usage of shadow fading model (yes or no).
    - ∗ LTE_params.ChanMod_config.winner_settings.PathLossModel: path loss model function name (pathloss).
    - ∗ LTE_params.ChanMod_config.winner_settings.PathLossOption: Available options are CR_light, CR_heavy, RR_light, RR_heavy. CR=Corridor-Room, RR=Room-Room NLOS.
    - ∗ LTE_params.ChanMod_config.winner_settings.RandomSeed: sets a random seed. Can be left empty.
    - ∗ LTE_params.ChanMod_config.winner_settings.UseManualPropCondition: whether to use manual propagation condition (LOS/NLOS) settings or not (yes or no). If not, the propagation condition is drawn from probabilities.

- LTE_params.ChanMod_config.corr_coefRX: correlation between the receiver antennas. Only compatible with block fading filtering.
- LTE_params.ChanMod_config.corr_coefTX: correlation between the transmitter antennas. Only compatible with block fading filtering.
- LTE_params.ChanMod_config.time_correlation: sets whether the channel realizations are time-correlated or not. correlated or independent. This parameter is also related with LTE_params.N_seed_reset.

E. *Scheduler parameters*

Configuration of the scheduler parameters.

- LTE_params.scheduler.type: type of scheduler. Already numerous schedulers are supported (see [21] and [22]) the most important being:
  - round robin: round robin scheduler without support of UE feedback.
  - fixed: round robin scheduler but with support of UE feedback (MCS and MIMO preprocessing are adapted).
  - best cqi: scheduling decisions are taken according to the reported channel quality, such as to maximize the system throughput.
  - proportional fair: proportional fair scheduler.

  Other supported schedulers are (note: these are not thoroughly tested for all configurations):
  - max throughput: max. throughput scheduler based on a linear program
  - max min: max. min. scheduler
  - var fair: throughput suboptimal scheduler with variable, adjustable fairness
    (it utilizes the parameter LTE_params.scheduler.fairness)
  - variable fair: throughput optimal scheduler with variable, adjustable fairness
  - alpha fair: scheduler with fairness indirectly set by the parameter LTE_params.scheduler.alpha

  All those schedulers assume full buffer simulations. The only scheduler that supports the traffic models is the RoundRobin_Traffic scheduler.
- LTE_params.scheduler.assignment: either static, semi static or dynamic. Whether the scheduler will statically assign or dynamically assign CQIs and other params. The best cqi scheduler is capable of assigning MCSs dynamically. The semi static scheduler adapts automatically the precoder and layer number according to the PMI and RI feedback (use this one in conjunction with closed loop spatial multiplexing and activated PMI and RI feedback). The semi static scheduler assigns resource blocks in a round robin fashion.
- LTE_params.scheduler.fixed_scheduler_assignment: this option is used only for the fixed scheduler. In that case, the parameter is a vector of length LTE_params.nUE containing the number of RBs allocated to each user.

The total number or assigned RBs must be equal or less than the number of RB available in the specified bandwidth: eg. for 2 UEs and 1.4 MHz `[4 2]` would assign four RBs to the first user and two to the second.

- `LTE_params.scheduler.cqi`: the CQI the scheduler is going to use when transmitting data. When set to `'set'`, when in the `static` case, the CQI value to use will be read from the `cqi_i` variable, which is to be set from the script file that launches the simulation (eg. `LTE_sim_batch`). This is only used with the `round robin` and `fixed` scheduler in `static` mode.
- `LTE_params.scheduler.PMI`: sets the Precoding Matrix Indicator, if no feedback is employed.

F. Variable uplink delay

Additionally to the `LTE_params.uplink_delay` variable, it is possible to specify individual delay values for specific variables. The delay of any property of the `outputs.ueOutput` class may independently defined. If you were to add additional properties to the object, it would be possible to assign a specific delay with this option without further code modifications.

Although the feature is quite flexible, the most common application is to define a common uplink delay for most of the parameters and (for example) a different one for the CQI, RI and PMI feedback.

Below is an example that sets the global uplink delay to one, while setting different delays for the ACKs and CQI/PMI/RI. The extra delays are specified with the optional `LTE_params.extra_uplink_delays` parameter, containing a cell array of property names and delays in TTIs.

```
LTE_params.uplink_delay = 1;
LTE_params.extra_uplink_delays = {...
  'ACK',3,...
  'RI',8,...
  'PMI',8,...
  'CQI',8 };
```

In this example, all feedback will be delayed one TTI, while the ACKs will experience a delay of three TTIs and the CQI, PMI and RI feedback will be delayed eight TTIs.

IMPORTANT: while the configuration may allow for flexibility, setting different delays for the CQI, RI and/or PMI can lead to unexpected results. e.g., if the CQI dimensions do not match those resulting from a given rank value or if the retrieved PMI (after delay) would not be in the range of the retrieved RI due to them referring to actually different TTIs, the simulator (very probably the scheduler) will in the best case crash, and in the worst case not crash and maybe yield erratic results, leaving you wondering why the results look strange.

The simulator also allows the possibility to specify a configurable update rate for Channel State Information (CSI) feedback. The feedback to which this now applies comprises the following `outputs.ueOutput` variables: `RI`, `PMI`, `PMI_WB`, `CQI`, and `CQI_bar`. However, the variable choice can be changed with the `CSI_feedback_variables` of the `outputs.ueOutput` class. In the case you would decide to add additional feedback to the simulator and would require this feature, the considered CSI variable set can be easily modifiable.

This feature is configured by the optional `LTE_params.CSI_feedback_interval` parameter. If not defined, it is equivalent to setting it to one. For `LTE_params.CSI_feedback_interval=1`, the values output by the uplink channel correspond to the delayed feedback. for higher values (N), N values are sent and the feedback is updated again after N TTIs. i.e., in the normal case, the received feedback would correspond to the following TTIs:

$$[1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ \ldots],$$

if `LTE_params.CSI_feedback_interval=3` is specified, the feedback received by the scheduler will correspond to the following TTIs:

$$[1,\ 1,\ 1,\ 4,\ 4,\ 4,\ 7,\ 7,\ 7,\ \ldots].$$

It is important to not that this overwriting of the actual feedback with an old one is performed after valid feedback is received, i.e., after a number of TTIs equal to the maximum uplink delay has passed.

VI. Reference simulation results included

Besides the one mentioned on Section II, the simulator is provided with some reference simulation results, which can be compared with performance curves from 3GPP RAN documents such as [23] to cross-check the results of the simulator.

The example simulation results distributed with the simulator are described below.

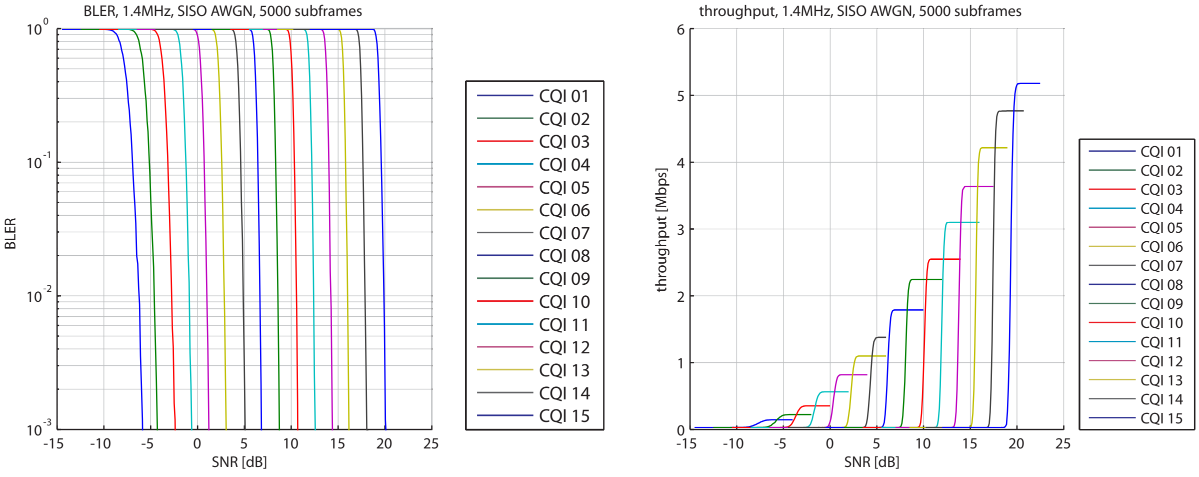
- `plot_reference_BLER_curves_r655` produces the plots in Figure 5, showing Block Error Ratio (BLER) and throughput for an Additive White Gaussian Noise (AWGN) simulation using the MCS defined in [11] (CQIs 1-15).
- `plot_R1071967_throughput_curves_r553` shows the BLER and throughput curves obtained from performing AWGN simulations with the MCSs specified in R1-071967, page 16 [23], as seen in Figure 6.

The MCS, including the CQI value used in the simulator, the modulation and Effective Code Rate (ECR) used in each of the simulations are shown in Table III.

CQI	Modulation	ECR [23]	ECRx1024
101	4QAM	1/9	114
102	4QAM	1/6	171
103	4QAM	0.21	215
104	4QAM	1/4	256
105	4QAM	1/3	314
106	4QAM	0.42	430
107	4QAM	1/2	512
108	4QAM	0.58	594
109	4QAM	2/3	683
110	4QAM	0.73	748
111	16QAM	0.43	440
112	16QAM	0.46	471
113	16QAM	1/2	512
114	16QAM	0.54	553
115	16QAM	0.58	594
116	16QAM	0.61	625
117	16QAM	2/3	683
118	16QAM	0.73	748
119	16QAM	4/5	819
120	64QAM	0.58	594
121	64QAM	0.62	635
122	64QAM	2/3	683
123	64QAM	0.70	717
124	64QAM	0.74	758
125	64QAM	4/5	819
126	64QAM	0.85	870
127	64QAM	0.90	922

CQI	Modulation	ECR	ECRx1024 [11]
1	4QAM	0.0762	78
2	4QAM	0.1172	120
3	4QAM	0.1885	193
4	4QAM	0.3008	308
5	4QAM	0.4385	449
6	4QAM	0.5879	602
7	16QAM	0.3691	378
8	16QAM	0.4785	490
9	16QAM	0.6016	616
10	64QAM	0.4551	466
11	64QAM	0.5537	567
12	64QAM	0.6504	666
13	64QAM	0.7539	772
14	64QAM	0.8525	873
15	64QAM	0.9258	948

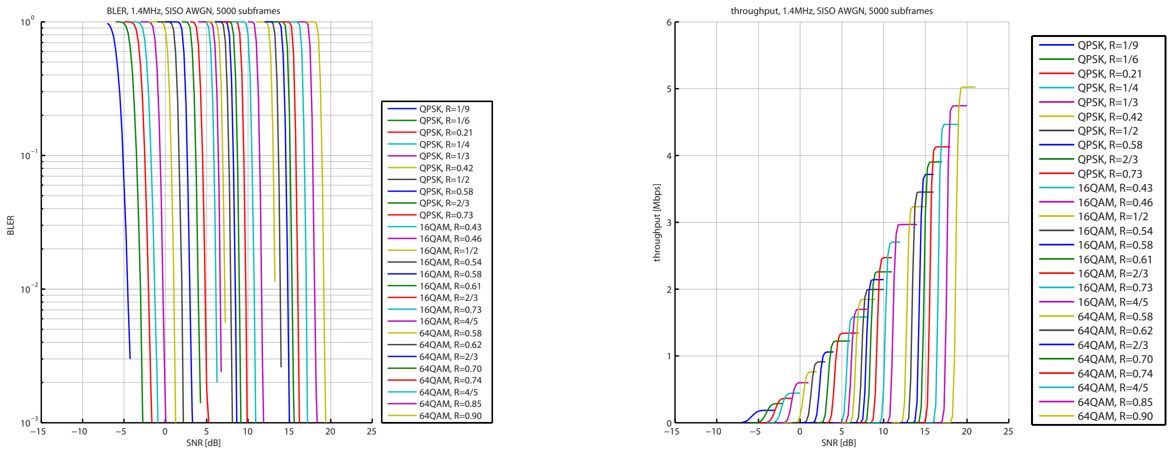
TABLE III: MCSs used in the `LTE_sim_batch_R1_07196` (left) and `LTE_sim_batch_quick_test` (right) MATLAB scripts



(a) BLER, AWGN no HARQ

(b) Throughput, AWGN, no HARQ

Fig. 5: Reference BLER and throughput plots for the 15 MCSs defined in [11]



(a) BLER, AWGN no HARQ

(b) Throughput, AWGN, no HARQ

Fig. 6: Reference BLER and throughput plots for the 27 MCSs defined in [23]

VII. Reproducing the results/plots presented on a paper

One of the main points of the simulator is to allow you to reproduce and review our results, as well as the algorithms that produce them. For each of the following publications a script is prepared in the folder *paper scripts* that will reproduce the results in the corresponding paper:

- Currently no papers presenting results obtained with the LTE-A simulator are published. The Vienna LTE Link Level Simulator contains several scripts for reproducing published results.

VIII. Extensions of the Vienna LTE-A Link Level Simulator compared to the Vienna LTE Link Level Simulator

The Vienna LTE-A Link Level Simulator is a direct extension of the Vienna LTE Link Level simulator. It contains all the features and transmit modes implemented in the LTE simulator. The LTE simulator supports standard compliant transmission from up to four transmit antennas in one of the following transmit modes:

- Single antenna port scheme (mode 1): transmission from a single antenna port only.
- Transmit diversity scheme (mode 2): transmission from two antenna ports utilizing the Alamouti transmit diversity scheme.
- Open loop transmit diversity scheme/large delay CDD scheme (mode 3): transmission from 2 or 4 antenna ports, utilizing large delay Cyclic Delay Diversity (CDD).
- Closed loop spatial multiplexing (mode 4): transmission from 2 or 4 antenna ports, utilizing UE feedback for choosing the appropriate precoding matrix. Corresponds to the single and dual layer transmission schemes defined in the LTE spec.

These four transmission modes all rely on the LTE reference signal structure, consisting of Cell Specific (CS) Reference Signals (RS).

In addition to these transmission modes, the LTE-A simulator supports the following modes:

- MUMIMO based on the LTE codebook (mode 5): this mode allows to share spatial transmission layers between users. It is also based on CS-RS. Just a dummy mode is implemented here, assuming:
  - A round robin scheduler (`fixed MUMIMO`) that equally distributes time, frequency and space resources to users.
  - Only “benchmark” receivers are implemented, assuming full channel knowledge. Here each user decodes all spatial layers and throws away the ones that are not intended for him.
- Transmission modes based on UE specific RS and CSI RS
  - Non-standard defined precoding (mode 8): ZF beamforming is implemented as an example, utilizing the `ZF_MUMIMO` scheduler.
  - Up to eight layer transmission (mode 9): this mode is the LTE-A counterpart of the LTE modes 4 and 5. It allows fast switching between SUMIMO and MUMIMO with up to eight transmit antennas. For MUMIMO the same dummy scheme as for mode 5 is implemented (employing the `fixed MUMIMO` scheduler), and the SUMIMO scheme is based on the UE feedback algorithms presented in [9].

More advanced features of LTE-A like e.g. carrier aggregation and basestation cooperation are not implemented in the first release, but the structure of the simulator allows to include such features as well (by implementing appropriate schedulers,...).

IX. SNR DEFINITION

In a MIMO-Orthogonal Frequency Division Multiplexing (OFDM) transmission system the SNR γ is the measurement for channel quality information and is a key factor of link error prediction. There are different measures or calculation procedures for the SNR in SISO and MIMO systems. For SISO systems the SNR can be viewed as receiving SNR, that is, received signal-to-noise ratio before the detector. Usually post-detection Signal to Interference and Noise Ratio (SINR) is used for MIMO link prediction. It is representing the SNR after combining in the receiver and measures the likelihood that a MCS will be decoded successfully.

The LTE Link Level simulator SNR γ is defined as follows:

- `BS_output.y_tx` contains the Tx-signal vector $\mathbf{x} = [\mathrm{x}_1, \ldots, \mathrm{x}_{N_\mathrm{T}}]^\mathrm{T} \in \mathbb{C}^{N_\mathrm{T} \times 1}$, where $\mathrm{x_k} \in \mathbb{C}$ with $\mathrm{k} \in [1, \ldots, N_\mathrm{T}]$ is the Tx-symbol sent from the k-th Tx-antenna ($N_\mathrm{T} \ldots$ number of Tx-antennas). We get
  - the total Tx-power, $\sigma_\mathrm{x}^2 = \mathrm{trace}(\mathbf{R}_\mathrm{x}) = \mathrm{trace}(\mathrm{E}\left\{\mathbf{x}\mathbf{x}^\mathrm{H}\right\}) = 1$
  - the Tx-power per Tx-antenna, $\sigma_{\mathrm{x_k}}^2 = \mathrm{E}\{|\mathrm{x}_k|^2\} = 1/N_\mathrm{T}$
- `ChanMod_output.H` defines the channel matrix $\mathbf{H}$, with $\|\mathbf{H}\|_\mathrm{F}^2 = N_\mathrm{T} N_\mathrm{R}$ ($N_\mathrm{R} \ldots$ number of Rx-antennas)
- `BS_output.cell_genie.v` defines the noise vector $\mathbf{v}$ with respect to the size of the Fast Fourier Transform (FFT) (`LTE_params.Nfft`) and the number of subcarriers (`LTE_params.N_tot`) before the detector, where $vec(\mathbf{v}) \sim \mathcal{CN}(\mathbf{0}, \sigma_\mathrm{v}^2 \mathbf{I})$
- `BS_output.cell_genie.n` defines the noise vector $\mathbf{n}$ after the FFT, where $vec(\mathbf{n}) \sim \mathcal{CN}(\mathbf{0}, \sigma_\mathrm{n}^2 \mathbf{I})$
- `ChanMod_output.y_rx` contains the Rx-signal vector $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}$
  - we get the receive SNR (before the detector),

$$\gamma_\mathrm{prefft} = \frac{\|\mathbf{H}\mathbf{x}\|_\mathrm{F}^2}{N_\mathrm{R}\sigma_\mathrm{v}^2} = \frac{N_\mathrm{R}}{N_\mathrm{R}\sigma_\mathrm{v}^2} = \frac{1}{\sigma_\mathrm{v}^2}$$

  - where the SNR after the FFT becomes,

$$\gamma_\mathrm{postfft} = \frac{\|\mathbf{H}\mathbf{x}\|_\mathrm{F}^2}{N_\mathrm{R}\sigma_\mathrm{n}^2} = \frac{N_\mathrm{R}}{N_\mathrm{R}\sigma_\mathrm{n}^2} = \frac{1}{\sigma_\mathrm{n}^2}$$

The difference between γ_prefft and γ_postfft is given by the relation $\sigma_\mathrm{v}^2/\sigma_\mathrm{n}^2 = N_\mathrm{fft}/N_\mathrm{tot}$ (using a FFT-size of 128 and 72 subcarrriers, we get $N_\mathrm{fft}/N_\mathrm{tot} = 128/72 = 1.778 \triangleq 2.5\,dB$).
We recommend to use the SNR after FFT (γ_postfft) when doing simulations over SNR, as this allows direct comparisons to theoretic results.

X. NOTE ON PARALLEL SIMULATIONS AND RANDOM NUMBER GENERATION

Note that when using parallel simulations the random number generator will return the same sequence of numbers in each of the parallel-running MATLAB labs. Since the `parfor` loop is over SNR values, this would mean that each SNR iteration is in principle identical to the other ones just with a different noise level (this may not be the case depending on the circumstances, though). If this is not adequate for your needs, you may need to modify the code regarding the `RandStream` initialization.

Below is a code that illustrates what happens when using `RandStreams` in parallel mode:

```
sim_length=15;
n_sims = 10;
output = zeros(n_sims,sim_length);
for sim_=1:n_sims
    a_RandStream = RandStream('mt19937ar','Seed',0);
    matlabpool open
    parfor t_=1:sim_length
        pause(1); %simulate doing something
        output(sim_,t_) = rand(a_RandStream);
    end
    matlabpool close
end
output
```

XI. Changelog

Changelog of the Vienna LTE-A Link Level simulator:

- v.1.3r1924,2013-03-08
  - Added LTE-A fast fading compatibility: The fast fading channel estimators `LS`, `MMSE`, `MMSE_full` , `Smoothness_match`, `Smoothness_deviate` and `Smoothness_approx` using the LTE-A reference symbol structure have been added as well as a `ZF` receiver for fast fading.
  - Added a new proportional fair scheduler (`prop fair gran`) that allows to vary the scheduling granularity. All other schedulers operate on a RB basis, while this scheduler can be set to assign resources in multiples of RBs only. To do this, open the scheduler object (`prop_fair_gran_scheduler.m` in `+schedulers` folder) and adjust the object-property `gran`. The fairness parameter α of the scheduler can also be adjusted using the variable `LTE_params.scheduler.alpha` in `LTE_load_parameter.m`.
  - Extended the feedback granularity options for CQI and PMI from either wideband or per RB, to any value in multiples of RBs. This is only supported for the standard compliant PMI, CQI feedback and not for specialized feedback, e.g., for non-codebook based MU-MIMO.
  - Fixed crash when using the `EPedA` channel model due to a missing correlation matrix generation for that channel.
  - Added feature to enable for a configurable update rate for CSI feedback. The feedback to which this now applies comprises the following `outputs.ueOutput` variables: `RI`, `PMI`, `PMI_WB`, `CQI`, and `CQI_bar`. However, the variable choice can be changed with the `CSI_feedback_variables` of the `outputs.ueOutput` class. See Section V-F for a more detailed description of this feature.
  - It is now possible to independently define the uplink delay of any variable contained in the `outputs.ueOutput` object. An example use case would be to define a different feedback value to the PMI and RI feedback, while the rest employ the common value defined in `LTE_params.uplink_delay`. See Section V-F for a more detailed description of this feature.
  - Reorganized channel matrix generation-related functions to its own folder.
  - Updated the AWGN BLER scripts. They are now merged into a single `LTE_sim_batch_BLER_curves.m` script in the `examples` folder.
  - Updated `LTE_sim_batch_quick_test` (see Section II-A). Simulation results for SISO, TxD (2×2), OLSM (2×2, 4×2, 4×4), and CLSM (2×2, 4×2, 4×4) configurations are now provided. As opposed to before, where this test simulation was based on a fix CQI, this example now features Adaptive Modulation and Coding (AMC) and MIMO feedback over the whole SNR range.
  - Small performance improvements.
  - Added MIMO support to the round robin scheduler.
  - Optimized the MIESM SINR averaging and the code to generate the CLSM precoders (i.e., noticeably faster simulations).
  - Merged the `LTE_sim_single` and `LTE_sim_par` implementations into a single implementation supporting both normal and parallel simulations and a simpler input-output variable flow.
  - Fixed rank indicator feedback calculation bug that limited the OLSM transmission with four transmit antennas to rank 3.
- v.1.2r1546, 2012-10-16 (marked before as `LTE-A_v1.0_r200.zip`)
  - Added support of distributed antennas. Note that DASs are only used in combination with a pathloss and shadow-fading model, because otherwise there would not be a difference between centralized and distributed antennas.
  - Improved support of interference alignment (IA) with a new structural element in the simulator, the base station coordinator (BS_coordinator). With this element it should be possible to incorporate also other coordination schemes (simply follow the example of IA).
  - Added least squares (LS) and linear minimum mean squared error (LMMSE) channel estimators for LTE-A.
  - Fixed normalization error in the sinc interpolator. A `sqrt` was missing in the constructor. Thanks to Han Zhou (Shanghai Advanced Research Institute (SARI), China), for pointing out this bug.
  - Added new AWGN BLER curves for both 1.4 MHz and 5 MHz. See the `examples/AWGN_BLER_curves` folder.
- v.1.1r100, 2011-09-27 (marked before as `LTE-A_v1.0_r100.zip`)
  - First release of the Vienna LTE-A Link Level simulator.

The Vienna LTE-A Link Level simulator is an extension of the Vienna LTE Link Level simulator. It is built around Version 1.7 of the LTE simulator. For convenience, the changelog of the Vienna LTE Link Level simulator until this version is also included:

- v.1.7r1089, 2011-10-04
  - MEX files are now compiled with Microsoft Visual C++ 2010, so the appropriate runtime files are needed. Refer to Section XIII for more information on how to get them.

  - Fixed typos in the `LTE_sim_batch_BLER_curves` file that reproduces the AWGN BLER curves.
  - BICM capacity curves are now loaded by default not from a file, but from a hardcoded source. Loading from a file is still supported, though.
  - Fixed error in the rate matching that caused an incorrect number of bits than otherwise expected by the layers for cases of bandwidth higher than 1.4 MHz and certain resource block allocations.
  - Changed the layout of the `load_parameters` file. It was getting so long it was no longer readable. Moved the individual configuration parameters for each simulation type to separate object files in the `simulation_config` package.
  - Fixed discrepancy initialization in the gold code generation MEX file. Thanks to Srikanth Isanaka for pointing out this bug and Mitsuo Sakamoto for generously providing an updated version of the gold code generation code.
- v.1.6r917, 2011-01-19
  - Fixed error in `LTE_sim_results_plots` when there is no frequency offset estimation.
  - Fixed signaling bug for TB sizes smaller than 40 bits. Thanks to Wang Dongming (Southeast University, China) for pointing out this bug.
  - Fixed bug in the new implementation of the rate matching. The LLR repetition was not taken into account, therefore affecting performance.
- v.1.5r811, 2010-11-05
  - Added experimental multi-base station simulation support. Still in very early development stages.
  - Reimplementation in MATLAB of most of the rate-matching procedures. The code is optimized enough so that MEX functions are now much less needed. This should improve code readability and maintainability.
  - Added “Companies (no matter profit-oriented or not) are not allowed for free usage and have to contact the licensor before usage.” to the license agreement (Section A).
  - Fixed bug in the fixed scheduler. The `PBCHsyms` variable was being used, thus causing the simulation to crash when the scheduler function was called.
  - The CQI used for each Transport Block (TB) transmission in now stored in the results trace (`used_CQI` variable in `ueSpecificTraces`, in turn stored in `simulation_results`).
  - Added trace to track the codeblock BLER. The ACK value is the decimal representation of the codeblock ACK vector for each codeword.
- v.1.4r715, 2010-07-12
  - Fixed bug in the channel coding process. According to [14]: ”If the code to be encoded is the 0-th code block and the number of filler bits is greater than zero, i.e. $F > 0$, then the encoder shall set $c_k = 0, k = 0, \dots, (F-1)$ at its input and shall set d_k^0 =`<NULL>`, $k = 0, \dots, (F-1)$ and d_k^1 =`<NULL>`, $k = 0, \dots, (F-1)$ at its output”. These bits were erroneously set to 0 instead of `<NULL>`. In `LTE_rx_DLSCH`, the first `BS_signaling.TB_segmentation.F` LLR values of $d^{(0)}$ and $d^{(1)}$ are replaced by `-LTE_params.UE_config.LLR_clipping`. Thanks to Wu Gaojin (University of Posts and Telecommunications, China) for pointing out this bug.
  - Fixed bug in the RX rate matching process that caused a crash when no filler (`<NULL>`) bits were present (ie. `UE_signaling.turbo_rate_matcher(stream_index).null_positions{i}` was empty). Thanks to Narciso García Cano (Universidad Politécnica de Madrid, Spain) for helping us out with this bug.
  - The Gold sequence generation was based on version 8.2.0 of the standard. Between the latest release and v8.2.0., pseudo random sequence generator was updated to include N_c [15]. Thanks to Mitsuo Sakamoto (Couei Corp, Japan) for helping us out with this bug and kindly providing an updated version of the Gold Sequence generation that is also $\sim$ 40x faster.
  - Added comment on issues that may arise from using random number generators and parallel simulations.
  - Added support of user equipment feedback (CQI, RI and PMI). The feedback values are evaluated as described in [6] (added a script to allow reproduction of the figures in this paper). CQI feedback is based on ESM. MIESM and EESM are supported for SINR averaging. The feedback calculation is just tested for the ZF receiver.
  - The receiver for 4x1 and 4x2 TxD mode was modified. The performance is unchanged, but execution speed has been greatly improved.
- v.1.3r620, 2010-02-18.
  - Fixed bug in the turbo decoding process. The second decoder was initialized to the interleaved systematic bits instead of zero. Due to this there is a small performance improvement in the order of 0.2 dB. Thanks to Klaus Hueske (Technische Universität Dortmund Information Processing Lab, Germany) for helping us out with this bug.
  - Fixed errors in the way the fixed scheduler object was called that made it impossible to use it. Thanks to Tommaso Balercia (Comneon GmbH, Germany) for pointing out this bug.
  - Changed structure of the simulations parameters' loading file in order to decrease the number of configuration files.
  - Added doubly dispersive channel estimation with scalable complexity [24] and related files needed to reproduce the figures from the paper.

  - Added precoding matrix indicator feedback support as described in [25] and related files needed to reproduce the figures from the paper.
- v.1.2r553, 2009-12-25.
  - Minor bugfixes and improvements.
  - Fixed bug in the rate matching process that caused the `<NULL>` bits inserted during the sub-block interleaving process to be treated as zeros (see [14]). Due to this fix the performance of the channel coding is slightly improved. Reference and example simulations included with the simulator have been updated accordingly. Thanks to Ching Hsiang Wu (Institute for Information Industry Networks and Multimedia Institute, Taipei) for pointing out this bug.
  - Fixed bug that caused poor performance of the ZF receiver. This was due to a bad noise scaling being passed on to the demapper. After the fix the ZF and Soft Sphere Decoder (SSD) receivers have the same performance for the SISO case.
  - Since according to the MathWorks,the `seqgen` function will be removed from future versions of the Communications Toolbox software, calls to `seqgen.pn` have been substituted by calls to `commsrc.pn`. See Section XIII for issues this change may cause.
  - Fixed bug that caused performance decrease for fast fading simulations at high user velocities. For the decoding of the last seven OFDM symbols, the channel of the first seven OFDM symbols was used.
  - Added support of the Winner II + channel model [26]. See Section XIV on more information on how to enable this functionality.
- v.1.1r450, 2009-08-25.
  - Minor bugfixes and improvements.
  - Added Parallel and Distributed Toolbox support (parfor).
  - Added best CQI (only for SISO and fixed schedulers.
- v.1.0r400, 2009-05-15.
  - First publicly available version of the LTE Link Level Simulator.

XII. REFERENCING

A version of the LTE Link Level Simulator paper is available in our publication data-base **here**.

If you are using the simulator for your scientific work, please use the reference below:

```
@ARTICLE{JASP2011,
author = {C.~Mehlf\"uhrer and  J.~Colom Ikuno and  M.~Simko and  S.~Schwarz and  M.~Wrulich and  M.~Rupp},
title = {The Vienna {LTE} Simulators - Enabling Reproducibility in Wireless Communications Research},
journal = {{EURASIP} Journal on Advances in Signal Processing},
year = {2011},
volume = {Vol. 2011},
pages = {1-13}
}


C. Mehlf\"uhrer, J. Colom Ikuno, M. Simko, S. Schwarz, M. Wrulich, M. Rupp,
The Vienna LTE Simulators - Enabling Reproducibility in Wireless Communications Research,
EURASIP Journal on Advances in Signal Processing, Vol. 2011, pages 1 - 13, 2011.
```

XIII. KNOWN ISSUES

- The LTE simulators make use of the new Object-Oriented capabilities of Matlab (available since R2008a), the simulators will not run under older Matlab releases without extensive changes.
- Please note that MEX-files generated using Microsoft Visual C++ 2010 require that Microsoft Visual Studio 2008 run-time libraries be available on the computer they are run on. The runtime files can be downloaded **here** (x86) or **here** (x64).
- In order to be able to use the parallel version of the simulator (when setting `LTE_params.simulation_type` to `parallel`, you need the parallel toolbox (included by default with MATLAB r2009a and above or as an add-on with previous versions). It will not work if you don't have the toolbox, just crashing the moment the `matlabpool` function is called.
- In MATLAB versions prior to r2009a the code may not work, as the `commsrc.pn` function does not exist. You will need to replace every call to `commsrc.pn` with a call to `seqgen.pn` in order to run the simulator. No change in the arguments is needed. Such changes should be applied to the code in the `LTE_common_gen_Synchronization_Signal` and `LTE_common_gen_Reference_Signal` functions.
- In the `LTE_rx_turbo_decode` function, only the max-log-map decoder type has been tested. The `decoder_type` variable is used as input an configures the SISO decoder function, which is part of [27].
- It was pointed out that in [12], the phase ϕ is not different for each sinusoid. We are using a modified version [13].

XIV. Using the WINNER Phase II channel model reference implementation

Starting with v.1.2r553, it is possible to use channels generated with the publicly-available MATLAB implementation of the WINNER Phase II Channel Model [26]. Since the code is distributed under the GNU GPL, its files are not included in the simulator release. In order to use to be able to use it, you will have to download it yourself. For this, go to the WINNER Phase II Model website, download the `WIM2_3D_ant_ver064_220908.zip` file and unzip the `.mat` files in the `./Winner Channel Model` folder.

XV. Questions

For questions please check our **forum**, where you will be able to post your questions/comments/bug reports. It makes it easer for you to see what other people asked and also makes it easier for us to answer you (when we have time).

XVI. Mailing List

If you want to receive information about future updates you can subscribe to our LTE simulator mailing list **here**. Note that you can change the display language to english in the selection panel to the right.

XVII. The People (so far) behind the development of the simulator

- Dagmar Bosanska
- Josep Colom Ikuno
- Govinda Lilley
- Christian Mehlführer
- Michael Meidlinger
- Martin Müller
- Stefan Pendl
- Jörg Reitterer
- Markus Rupp
- Stefan Schwarz
- Michal Šimko
- Qi Wang
- Martin Wrulich

XVIII. License agreement

These terms (license for the LTE link level simulator) refer to the use of the *LTE-A link-level simulator* (the ”Original Work), developed by the Institute of Telecommunications, Vienna University of Technology (the licensor).

A. Academic Usage

Academic Usage in the context of this license describes the use of the Original Work in scientific projects without any reimbursement or financial claims that bear on results derived by the Original Work, but subject however to the restrictions provided for in Clauses B and H hereinbelow. The main goal in the sense of Academic Usage shall be to obtain cientifically significant results that can be used for publication. Academic Usage of the Original work will only be granted after this license agreement is:

1) printed on letterhead of university department or research institute,
2) signed by the department head, including date and stamp of the department,
3) then scanned and, including the above, and emailed to: Univ.Prof. Markus Rupp (mrupp@nt.tuwien.ac.at).

Companies (no matter profit-oriented or not) are not allowed for free usage and have to contact the licensor before usage.

B. Grant of copyright license

Licensor grants You a worldwide, royalty-free, non-exclusive, non-sublicensable license, restricted to non-commercial use, for the duration of the copyright, to install the Original ork and any Derivative Works thereof on one personal computer. The license allows You to:

1) Use the Original Work only for Academic Usage. Any usage of the Original Work, entirely or in part or modified, requires the proper citation, e.g. as reference in a publication.
2) Translate, adapt, alter, transform, modify, or arrange the Original Work, thereby creating derivative works (Derivative Works) based upon the Original Work. Distribution, either royalty-free or commercially, in parts or in modified form of the Original Work, i.e. also of Derivative Works, is prohibited and not covered by ”Academic Usage”.
3) Display results derived from the Original Work, or in modified form, publicly, without commercial usage.

C. Grant of source code license

The term ”Source Code” means the preferred form of the Original Work for making modifications to it and all available documentation describing how to modify the Original Work. Licensor agrees to provide a machine-readable copy of the Source Code of the Original Work along with each copy of the Original Work that Licensor distributes. Licensor reserves the right to satisfy this obligation by placing a machine-readable copy of the Source Code in an information repository reasonably calculated to permit inexpensive and convenient access by You for as long as Licensor continues to distribute the Original Work.

D. Exclusions from license grant

Neither the names of Licensor, nor the names of any contributors to the Original Work, nor any of their trademarks or service marks, may be used without express prior permission of the Licensor, except as expressly provided otherwise in Clause B1 hereinabove. Except as expressly stated herein, nothing in this License grants any license to Licensor's trademarks, copyrights, patents, trade secrets or any other intellectual property. No license is granted to the trademarks of Licensor even if such marks are included in the Original Work. Nothing in this License shall be interpreted to prohibit Licensor from licensing under terms different from this License any Original Work that Licensor otherwise would have a right to license.

E. Warranty of provenance and disclaimer of warranty

Licensor warrants that the copyright in and to the Original Work is owned by the Licensor or is sublicensed to You under the terms of this License with the permission of the contributor(s) of those copyrights and patent rights. Except as expressly stated in the immediately preceding sentence, the Original Work is provided under this License on an ”AS IS” BASIS and WITHOUT WARRANTY, either express or implied, including, without limitation, the warranties of noniinfringement, merchantability or fitness for a particular purpose. THE ENTIRE RISK AS TO THE QUALITY OF THE ORIGINAL WORK IS WITH YOU. This DISCLAIMER OF WARRANTY constitutes an essential part of this License. No license to he Original Work is granted by this License except under this disclaimer.

F. Limitation of liability

Under no circumstances and under no legal theory, whether in tort (including negligence), contract, or otherwise, shall the Licensor be liable to anyone for any indirect, special, incidental, or consequential damages of any character arising as a result of this License or the use of the Original Work including, without limitation, damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses. This limitation of liability shall not apply to the extent applicable law prohibits such limitation.

G. Termination

If, at any time, You infringe upon the grants of this License, it shall terminate immediately and You may no longer exercise any of the rights granted to You by this License.

H. Open source and code under other license terms

The original work also contains work licensed under other licenses other than the license for the LTE link-level simulator. The terms and conditions described in this document are only applicable to the parts of the Original Work not under other licenses. A list of the parts of the Original Work not under the license for the LTE link-level simulator can be found in Appendix I.

I. Appendix I

The following parts of the original work are not under the terms of the license for the LTE link-level simulator, and are thus excluded from the terms and conditions stated by this license.

The usage and adaptation of these sections for use with the original work is done in compliance with the license terms they are released under. Any translation, adaptation, alteration, transformation, modification, or further use of the hereinbelow stated parts of the original work must be done under the terms of the applicable licenses for that specific part, which are also included in the package.

- CRC calculations C-code: MEXed version of the code generated by pycrc [28]. Under the MIT License.
- Turbo and convolutional encoder/decoder C-code, from the Coded Modulation Library (CML) by Iterative Solutions [27]. Licensed under the GNU lesser General Public License.

Use	c-source	MATLAB functions	License

CRC generation and checking	`crc16.c`
	`crc16.h`
	`crc24a.c`
	`crc24a.h`
	`crc24b.c`
	`crc24b.h`
		`LTE_common_crc16`
		`LTE_common_crc24a`
		`LTE_common_crc24b`
			MIT License [4]

Channel coding	`ConvEncode.c`
	`convolutional.h`
	`maxstar.h`
	`siso.h`
	`SisoDecode.c`
		`LTE_tx_convolutional_encoder`
		`LTE_rx_siso_decode`
			GNU Lesser GPL [3]

XIX. Acknowledgment

The authors would like to thank the whole LTE research group for continuous support and lively discussions. This work has been funded by A1 Telekom Austria AG, the Christian Doppler Laboratory for Wireless Technologies for Sustainable Mobility, as well as the Institute of Telecommunications Vienna University of Technology. The views expressed in this paper are those of the authors and do not necessarily reflect the views within mobilkom austria AG.

References

[1] [Online]. Available: http://www.nt.tuwien.ac.at/ltesimulator/

[2] C. Mehlführer, M. Wrulich, J. C. Ikuno, D. Bosanska, and M. Rupp, "Simulating the long term evolution physical layer," in *Proc. of the 17th European Signal Processing Conference (EUSIPCO 2009)*, Glasgow, Scotland, Aug. 2009.

[3] I. Free Software Foundation, "GNU lesser general public license, version 2.1." [Online]. Available: http://www.gnu.org/licenses/lgpl-2.1.html

[4] M. I. of Technology, "MIT license." [Online]. Available: http://www.opensource.org/licenses/mit-license.php

[5] Technical Specification Group Radio Access Network, "Evolved universal terrestrial radio access (E-UTRA); LTE physical layer – general description," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 36.201 Version 8.3.0, Mar. 2009.

[6] S. Schwarz, C. Mehlfhrer, and M. Rupp, "Calculation of the Spatial Preprocessing and Link Adaption Feedback for 3GPP UMTS/LTE," in *Proc. IEEE Wireless Advanced 2010*, London, UK, June 2010.

[7] C. Mehlfuhrer, J. Colom Ikuno, M. Simko, S. Schwarz, M. Wrulich, and M. Rupp, "The vienna lte simulators - enabling reproducibility in wireless communications research," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, no. 1, p. 29, 2011. [Online]. Available: http://asp.eurasipjournals.com/content/2011/1/29

[8] 3GPP, "Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation (Release 10)," December 2010, [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/36211.htm.

[9] S. Schwarz and M. Rupp, "Throughput maximizing feedback for MIMO OFDM based wireless communication systems," in *Signal Processing Advances in Wireless Communications SPAWC 2011*, San Francisco, CA, June 2011.

[10] G. Dietl, O. Labreche, and W. Utschick, "Channel vector quantization for multiuser MIMO systems aiming at maximum sum rate," in *IEEE Global Telecommunications Conference 2009*, Dec. 2009, pp. 1 –5.

[11] Technical Specification Group Radio Access Network, "Evolved universal terrestrial radio access (E-UTRA); physical layer procedures," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 36.213, Mar. 2009.

[12] Y. R. Zheng and C. Xiao, "Simulation models with correct statistical properties for rayleigh fading channels," *Communications, IEEE Transactions on*, June 2003.

[13] T. Zemen and C. Mecklenbräuker, "Time-Variant Channel Estimation Using Discrete Prolate Spheroidal Sequences," *IEEE Transactions on Signal Processing*, vol. 53, no. 9, pp. 3597–3607, Sept. 2005.

[14] Technical Specification Group Radio Access Network, "Evolved universal terrestrial radio access (E-UTRA); multiplexing and channel coding," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 36.212, Mar. 2009.

[15] ——, "Evolved universal terrestrial radio access (E-UTRA); physical channels and modulation," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 36.211 Version 8.7.0, May 2009.

[16] ——, "Evolved universal terrestrial radio access (E-UTRA); user equipment (UE) radio access capabilities," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 36.306, 2009.

[17] 3GPP, "Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures (Release 10)," December 2010, [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/36213.htm.

[18] ITU-R, "Guidelines for evaluation of radio transmission technologies for IMT-2000," ITU-R, Tech. Rep. M.1225, 1997.

[19] Technical Specification Group GSM/EDGE Radio Access Network, "Radio transmission and reception, annex c.3 propagation models," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 05.05 V.8.20.0 (Release 1999), 2009.

[20] L. Hentilä, P. Kyösti, M. Käske, M. Narandzic, and M. Alatossava. (2007) MATLAB implementation of the WINNER Phase II channel model ver1.1. [Online]. Available: http://www.ist-winner.org/phase_2_model.html

[21] S. Schwarz, C. Mehlführer, and M. Rupp, "Low complexity approximate maximum throughput scheduling for LTE," in *44th Annual Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, California, Nov. 2010.

[22] ——, "Throughput maximizing multiuser scheduling with adjustable fairness," in *International Conference on Communications ICC 2011*, Kyoto, Japan, June 2011.

[23] Alcatel-Lucent, "DL E-UTRA performance checkpoint," 3GPP TSG-RAN1, Tech. Rep. R1-071967, 2007.

[24] M. Šimko, C. Mehlführer, M. Wrulich, and M. Rupp, "Doubly Dispersive Channel Estimation with Scalable Complexity," in *Proc. WSA 2010*, Bremen, Germany, Feb. 2010, accepted.

[25] S. Schwarz, M. Wrulich, and M. Rupp, "Mutual information based calculation of the precoding matrix indicator for 3GPP UMTS/LTE," in *Proc. WSA 2010*, Bremen, Germany, Feb. 2010, accepted.

[26] L. Hentilä, P. Kyösti, M. Käske, M. Narandzic, and M. Alatossava, "MATLAB implementation of the WINNER Phase II Channel Model ver1.1," December 2007. [Online]. Available: https://www.ist-winner.org/phase_2_model.html

[27] I. Solutions, "Iterative Solutions Coded Modulation Library (ISCML)." [Online]. Available: http://www.iterativesolutions.com/

[28] T. Pircher, "pycrc CRC calculator and C source code generator." [Online]. Available: http://www.tty1.net/pycrc/

List of Figures

List of Tables