Simulation Code Structure of the paper “On Carrier Sensing Accuracy and Range Scaling Laws in Nakagami Fading Channels”

The codes (Matlab functions) are classified into the following categories. You can use “help function\_name” to find the description of a function.

**Data plotting functions:**

They are used for plotting figures from the simulation data.

* **plotAvgSensingAccuracy.m:** calculate the average sensing accuracy from DB.
* **plotBerVsSnrt802\_11a\_g.m:** plot for 802.11 a & g's '6M', '9M', '12M', '24M', '36M', '48M', '54M' the average packet success ratio as x\_i (interferer's location) varies when x\_r (transmission distance) is fixed comparing the BER versus SIRT reception model simulations, without considering carrier sensing.
* **plotBerVsSnrt802\_11b\_g.m:** plot for 802.11 b & g's 1M, 2M, 5.5M, 11M the average packet success ratio as x\_i (interferer's location) varies when x\_r (transmission distance) is fixed comparing the BER versus SIRT reception model simulations, without considering carrier sensing.
* **plotBerVsSnrt802\_11n.m:** plot for 802.11 ns '6.5M', '13M', '19.5M', '26M', '39M', '52M', '58.5M', '65M' the average packet success ratio as x\_i (interferer's location) varies when x\_r (transmission distance) is fixed comparing the BER versus SIRT reception model simulations, without considering carrier sensing.
* **plotCsRange.m:** plot the carrier sense range for a given CS threshold.
* **plotFadingVsTheoryWoCs802\_11b\_g.m**: plotting average packet success ratio as d\_i (interference distance) varies when d\_r (transmission distance) is fixed comparing a fading channel simulation and the theoretical result, without considering carrier sensing.
* **plotFadingVsTheoryWoCs802\_11a\_g.m:**
* **plotFadingVsTheoryWoCs802\_11n.m:**
* **plotPlaneRanges.m:** plot the planes of interference ranges and cs ranges.
* **plotStaticVsFadingCS.m:** plot static versus fading channels with carrier sensing as x\_i varies using the BER reception model.

**Simulation functions:**

They are the important simulation scripts.

* **constants.m:**It is the header file included by other source files and it consists of simulation constants.
* **genCurveData.m:** generating the curve of link performance for various interference distance when the locations of transmitter and receiver are fixed.
* **linkBerPerf.m:** running simulation to get the link performance according to the BER reception model for a given line topology.
* **linkSnrtPerf.m:** running simulation to get the link performance according to the SNRT reception model for a given line topology.
* **linkTheoryPerf.m:** calculating the value of theoretical performance formulae based only on the SNRT model for both static (no fading) and fading channels.
* **mainSimFadingCs802\_11.m:** *the important function*, which simulates static and fading channels with carrier sensing for all scenarios and store the data into the database. We use the database of *sqlite3* (<http://www.sqlite.org/>) and the JDBC driver *sqlitejdbc* (sqlite-jdbc-3.8.11.2.jar) for Matlab. Here is a short introduction on installation and how to use it: <https://bitbucket.org/xerial/sqlite-jdbc>

**Wireless library:**

They are the wireless reception codes (some of them are adopted from ns-3).

* **bpskBer:** calculate the bit error rate of BPSK at the given SINR**.**
* **calcSnrThreshes.m:** calculate the SNR threshold for each bit rate in a set of bit rates and the given packet length.
* **calcCsThresh:** calculate the aggressive and conservative CSTs for a given bit rate.
* **calculatePe:** calculate the \*coded\* bit error rate for the given BER and bValue.
* **db2origin.m:**converting dB (or dBm) to the ratio value (or milli-watts).
* **dqpskFunction.m:** used for calculating the success decoding rate of a packet for 2Mbps 802.11b.
* **dsssDbpskSuccessRate.m:**calculating the success decoding rate of a packet for 1Mbps in 802.11b.
* **dsssDqpskCck5\_5SuccessRate.m:**calculating the success decoding rate of a packet for 11Mbps in 802.11b.
* **dsssDqpskCck11SuccessRate.m:** calculating the success decoding rate of a packet for 11Mbps in 802.11b.
* **dsssDqpskSuccessRate.m:** calculating the success decoding rate of a packet for 2Mbps in 802.11b.
* **fec16QamSuccessRate.m:** calculate the success rate of a packet transmitted in the FEC 16QAM modulation scheme at the given SINR.
* **fec64QamSuccessRate.m:** calculate the success rate of a packet transmitted in the FEC 64QAM modulation scheme at the given SINR.
* **fecBpskSuccessRate.m:** calculate the success rate of a packet transmitted in the FEC BPSK modulation scheme at the given SINR.
* **fecQpskSuccessRate.m:** calculate the success rate of a packet transmitted in the FEC QPSK modulation scheme at the given SINR.
* **friis.m:** caculating the large-scale average receive power in dBm by the Friis (free space) model.
* **friisRange.m:** calculating the tx range for the given transmission power and reception threshold power under Friis path-loss model.
* **generalPathLoss.m:** calculating the large-scale average receive power in dBm by a general path loss model.
* **isFarField.m:** Judging whether d is in the far field, which a validity condition of the Friis model.
* **logNormPower.m:**calculating the receive power in dBm by the log normal model (for shadowing).
* **nakagamiFading.m**: generating a random reception power by the Nakagami-m multipath fading model
* **nakagamiFadingPSR.m:** calculate the theoretical packet success ratio in Nakagami fading, assuming the SNRT model.
* **origin2dB.m:** converting SNR ratio value (or milli-watts) to dB (or dBm).
* **packetSuccessRate.m:** calculate the success rate of a packet transmitted for any modulation scheme at the given SINR.
* **qpskBer:** calculate the bit error rate of QPSK at the given SINR.
* **rayleighFading.m:**generating a rx power by the Rayleigh multipath fading model.
* **sixteenQamBer.m:** calculate the bit error rate of 16QAM at the given SINR.
* **sixtyFourQamBer.m:** calculate the bit error rate of 64QAM at the given SINR.
* **twoRayGround.m:** caculating the "average" receive power by the two ray ground model.
* **twoRayGroundRange.m:** calculating the tx range for the given tx and rx power under Two ray ground path.

**Common library:**

They are codes for common usage.

* **draw\_arrow.m:** draw an arrow.
* **errorExit.m:** displaying an error message and exit the program.
* **twoDimDist.m:** caculating the distance between n\_0 and n1 in a 2D plane.
* **plotCircle.m:** plot a circle.
* **plotCurve.m:** plot a curve on the figure fig.

**Other obsolete codes:**

They are codes for some old simulations, scattered in the code directory.

nakagamiFadingPsrFormula.m

plotAvgHiddenProb.m

plotFadingVsTheoryWoCs802\_11a\_g.m

plotFadingVsTheoryWoCs802\_11b\_g.m

plotFadingVsTheoryWoCs802\_11n.m

plotNakagamiTheoryWoCs802\_11b\_11M.m

plotSnrPdf.m

rayleighFadingPdf.m