Advanced Signal Processing 1

TUG

The FTW/FSAN xDSL simulation tool

Gernot Matzenauer

TU Graz
2002

Outline

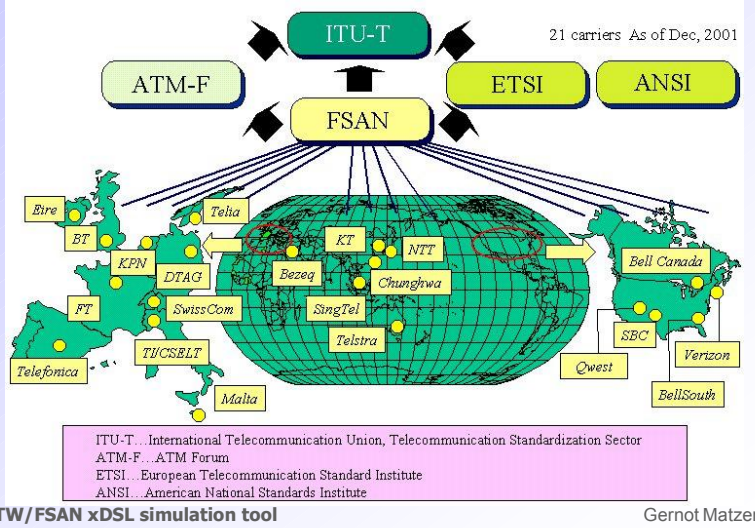
TUG

- FTW/FSAN
- The Simulation Tool
- GUI for VDSL
- Simulation Outline
- Examples

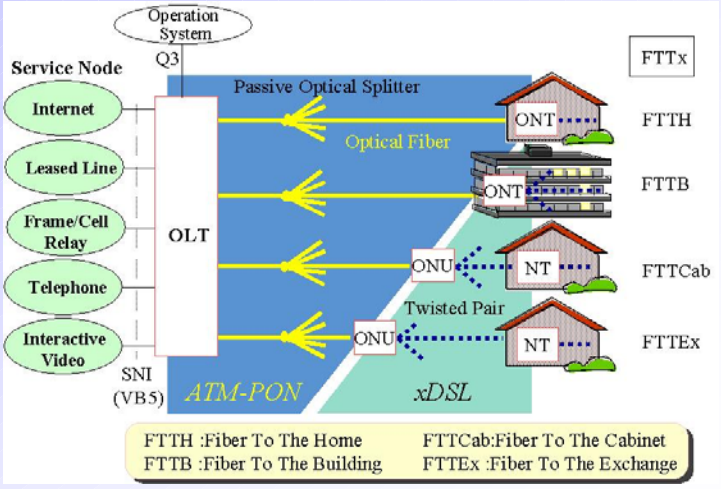
FTW

- **founded in 1999**
- **implements cooperation between science and industry**
- **www.ftw.at**

FSAN Structure



FTTx



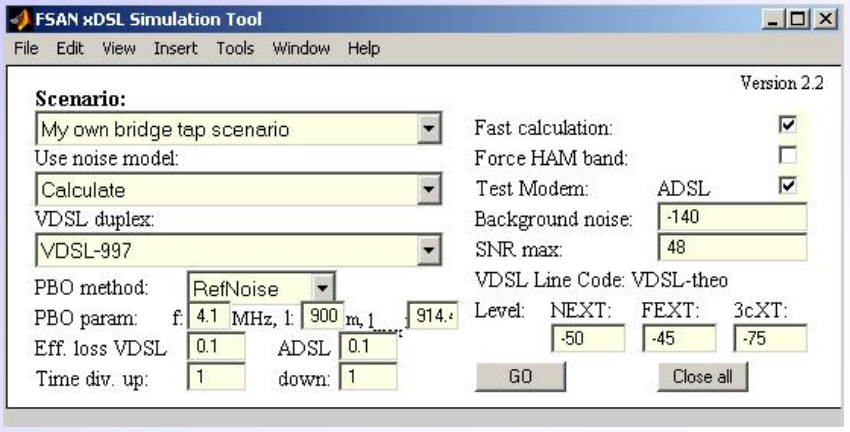
The simulation is outlined as

- An initiation phase where the experiment is described using a structure containing five parts
- The evaluation of the experiment
- Presentation of the result

Input Parameters

- *ex.param* general simulation parameters
- *ex.tfplist* list of modem's and disturber's time and frequency plans.
- *ex.lclist* list of line code definitions
- *ex.clist* list of cable definitions
- *ex.tt* the selected traffic and topology definition

GUI for VDSL



Graphical Output

- PSD mask
- Time and topology structure
- Transmission curves (log or linear)
  blue is the recieved signal
  red is the total noise
  black is the alien noise

Examples

- First Example
- Mixed Service Example
- GUI Example

The FTW/FSAN xDSL simulation tool:

The ftw. Forschungszentrum Telekommunikation Wien (Telecommunications Research Center Vienna) was founded in 1999 and implements cooperation between science and industry following a new and challenging approach. Our partners are three departments from Vienna University of Technology, a number of well-known telecommunications companies (network operators and telecom manufacturers), some smaller enterprises and the Austrian Research Center Seibersdorf.

FSAN Full Service Access Network

Background
At the Boston meeting (October 7 - 8, 1998), FSAN discussed producing and providing a simulation tool for evaluating different network topologies, duplexing strategies, spectrum plans, noise models, etc. It was agreed that Telia would take the responsibility of the implementation of the simulation tool in close cooperation with the other members of the FSAN group. This package is the result of this effort.

Objectives
The evaluation of VDSL capacity is a complicated and lengthy task. This is especially true for scenarios where the SNR is different at both ends of the lines and in topologies where power boost and power back-off are to be applied. When so many parameters can be varied, it is important to have a common understanding of the simulation environment. FSAN has discussed creating and using a common simulation tool that would include noise models, the FSAN noise combination method, power back-off methods, power boost, etc. The performance of VDSL as well as its impact on existing services (in particular ADSL, which is FEXT limited) should be evaluated.
Many of the above-mentioned features of the simulation tool have already been developed. FSAN has done ground-breaking work in defining noise models describing both FTTCab and FTTEx scenarios with various disturbers, a noise combination method, and network topologies describing potential real-world cases. These are all straightforward to use in simulations.

The objective of the simulation tool is to concentrate the FSAN results into one simulation package to facilitate evaluation of VDSL proposals and scenarios.

Within the FSAN VDSL group there has been a lot of ground-breaking work done, for example, by defining noise models, researching an appropriate noise combination method, and defining network topologies together with various disturbers describing potential real world scenarios. These methods can of course be used in this simulator. That is, one of the objective of this simulation tool was to concentrate the FSAN results into one simulation package to facilitate evaluation of VDSL proposals and scenarios. Later the scope of the simulator has been expanded to include evaluation of any xDSL method.

Extension in the future will partly depend on user feedback, so please send in comments and contribute to the simulator and its examples.

Please note that this is probably the last version of the simulator to be using FSAN in its name. In the future FTW (Forschungszentrum Telekommunikation Wien, Austria) will serve as support for the tool and will release bug fixes and function as coordinator for future simulator developments. A web site for the simulator can be found at: http://www.xdsl.ftw.at/xdslsimu. Already in this version of the simulator FTWs broader interest in all xDSL technolgies is shown with a much better support for other xDSL technologies besides VDSL, for example, ADSL and SDSL.

The simulation is outlined as

- An initiation phase where the experiment is described using a structure containing four parts
    - param - general simulation parameters
    - tfplist - list of modem and disturbers time and frequency plans.
    - ttlist - a list of all available traffic and topology definitions
    - tt - the selected traffic and topology definition
- The evaluation of the experiment
- Presentation of the result

General Simulation Parameters ex.param :

The general simulation parameters are found in the structure ex.param. A default setup is done in setupParam.m by making a call like
ex.param = setupParam;
In the ex.param structure we have the following variables
ex.param.frequency.fastrecalc
Switch variable which determines if calculations should use an optimized frequency axis which is slightly (1-2%) less accurate but can be up to 20 times faster. Set to 1 if fast calculations should be used, otherwise a slow but more accurate calculation is done.
ex.param.frequency.min
Lower boundary of frequency axis (Hz) used for the evaluation.
ex.param.frequency.max
Upper boundary of frequency axis (Hz) used for the evaluation.
ex.param.frequency.granularity
Granularity (Hz) in the frequency axis if fastrecalc is not used.
ex.param.frequency.f

Frequency axis used for evaluation (derived from the settings of the preceeding parameters)
ex.param.backgroundNoise
Background noise level to be used (dBm/Hz).
ex.param.XTlevel.NEXT
Reference near end crosstalk level at 1 MHz (dB)
ex.param.XTlevel.FEXT
Reference far end crosstalk level at 1 MHz (dB)
ex.param.XTlevel.thirdCXT
Reference crosstalk level for third circuit crosstalk at 1 MHz (dB)
ex.param.Zterm
Reference (loop terminating) impedance for attenuation calculation (Ohms)
ex.param.modemlist
List of modems to be evaluated in the simulation run. For each modem type (xDSL service) to be evaluated the modem's name must be contained in the list as a string. For example to evaluate VDSL as well as ADSL Modems of the given scenario a setting like ex.param.modemlist=['VDSL'; 'ADSL']; is needed. If ex.param.modemlist contains a modem name (service name) which does not exist in the selected scenario an error occurs.
ex.param.HAMBandName
Name of HAM band plan to be used as a string. The HAM bands are defined as PSD mask templates and must be contained in the experiment's list of time/frequency plans ex.tfplist.
ex.param.FSANNoiseModel
Predefined FSAN noise model to be used for the evaluation. The variable must contain the name of the considered noise model as a string. If set to 'Calculate' the real noise caused by the traffic on the scenario is evaluated in the simulation run. Predefined noise models are defined as PSD mask templates and must be contained in the experiment's list of time/frequency plans ex.tfplist.
ex.param.xDSLlist
A structured list with elements containing fields name and used . This list maps the generic VDSL (or other xDSL) name into a specific name. For example the setting ex.param.xDSLlist(1).name = 'VDSL' and ex.param.xDSLlist(1).used='VDSL-TDD-sym1:1' leads to the use of the time/frequency plan (tfplan) of 'VDSL-TDD-sym1:1' when the generic name 'VDSL' appears in the simulation parameters.

List of modem's and disturber's time and frequency plans ex.tfplist :
A list of all time and frequency plans considered in the experiment's scenario must be set up. This is simply done by using the definitions in the src\xdsldefs directory, for example:

```
% initiate ex.tfplist by fetching a HAM band definition
[ex.tfplist, ex.param.HAMBandName] = itu_tfplanHAM([]);
ex.tfplist  = fsan_tfplansMISC(ex.tfplist);     % Get plans for alien noise
ex.tfplist  = etsi_tfplansVDSL(ex.tfplist);     % Get some VDSL plans
```

Each entry in this list is a time and frequency plan (tfplan) containing the following fields:
ex.tfplist.name

Contains the modem names as strings. The first 4 letters must indicate the type of the xDSL modem (i.e.VDSL, ADSL, ...) and should be followed by characters which give a more specific description of the tfplan (e.g. 'VDSL-TDD-sym1:1', 'ADSL-overISDN', 'SDSL-asym', ...)
ex.tfplist.PSD.downstream
PSD mask definition strings for downstream, that is, any function/vector is possible. This also includes out of band PSD.
ex.tfplist.PSD.upstream
PSD mask definition strings for upstream, that is, any function/vector is possible. This also includes out of band PSD.
ex.tfplist.PSD.active.downstream
Vector containing minimum and maximum active frequencies for downstream. By active we mean the part that should be used for capacity calculations thus excluding any out of band energy from the PSD definition.
ex.tfplist.PSD.active.upstream
Vector containing minimum and maximum active frequencies for upstream. By active we mean the part that should be used for capacity calculations thus excluding any out of band energy from the PSD definition.
ex.tfplist.PSD.PBO.method
Connected to the tfplan is a power back off (PBO) method. The name of the PBO method is contained in this variable as a string. Currently there are four methods defined: reference length 'RefLen', reference FEXT 'RefFEXT', reference Noise 'RefNoise' and reference frequency 'RefFreq'. If no PBO is wanted a 'None' is selected.
ex.tfplist.PSD.PBO.param.len
PBO parameter length (m); needed for PBO methods 'RefLen' and 'RefFEXT'.
ex.tfplist.PSD.PBO.param.freq
PBO parameter frequency (Hz) ; needed for PBO method 'RefFreq'.
ex.tfplist.PSD.PBO.param.maxlen
For PBO method 'RefFreq' a maximum length can also be given.
ex.tfplist.PSD.HAM.active
Defines if HAM band should be active or not (1 is active).
ex.tfplist.timeDivision.up and ex.tfplist.timeDivision.down
Relative time contingent allocated for upstream respectively downstream in time division duplex methods. For example, in case of symmetric TDD these variables should be set to 0.5 each. For frequency division methods both should be set to 1).
ex.tfplist.timeDivision.sync
Flag indicating if the time division is synchronous or not. (1 is synchronous)
ex.tfplist.lcname
The name (as a string) of the line code definition to be used for the plan. This will the be found in ex.lclist (described below).

List of linecode definitions ex.lclist:
New in version 2 of this simulator is the possibility to define line code dependent features. Default line code structure is defined by calling setupLClist.m . Many others are defined in directory src\xdsldefs, for example (lcDefADSLDMT.m, lcDefVDSLDMT.m, lcDefVDSLSCM.m, lcDefSDSL_sym.m, lcDefSDSL_asym.m, ...).
ex.lclist = setupLClist;

The elements in the linecode list consist of the following four fields:
ex.lclist.name
The name of the line code definition as a string.
ex.lclist.param
A set of parameters needed for performance calculations.
ex.lclist.calcRate
This variable must contain the name of a function (as a string) to call for calculating the resulting bit rate (or margin in case of SDSL simulations) of a specific modem (xDSL service). The function's input arguments are the time/frequency fplan of the specific modem , the experiment result (result struct created by evalExperiment.m), the linecode definitions of the specific modem, and the used frequency vector. The function specified by ex.lclist.calcRate is typically called by the function calcFSANResult after execution of evalExperiment.
ex.lclist.lcPrint The function (as a string) which could be called to print out the parameters used in an experiment. Its argument is lc (the linecode structure to print).

List of cables ex.clist:
This list must contain all cable types used in the experiment scenario. A default list of cables is set up by calling
ex.clist = etsi_cables([]);

ex.clist consists of the following fields:

ex.clist.name The name of the cable as a string. This name is used to identify a cable in the loop topology definition.
ex.clist.model The name of the cable model (as a string) which is used to compute the cable's secondary line parameters (as attenuation, characteristic impedance, A,B,C,D parameters, ...) from their primary electric parameters (defined in ex.clist.param). Different models are needed because primary cable parameters are given according to different measurement methods.
ex.clist.param This is again organised as a struct which contains multiple fields for the cables primary parameters, depending on the definition format.

Traffic and topology definition ex.tt
The scenario used for the simulation run is described in the structure ex.tt (topology and traffic). When using a GUI all available scenarios can be stored in a traffic/topology list (typically gui.ttlist). Default scenarios are defined in src\xdsldefs, for example (fsan_loops.m, etsi_loopsSDSL.m, ansi_loops.m). For example the command
gui.ttlist = fsan_loops([]);
would set up a list of scenarios (with respect to traffic and topology) which are defined within the function fsan_loops.
For running a simulation for a specific scenario from this list the command

ex.tt = getList(gui.ttlist,scenario);
will set the ex.tt struct properly by using the function getList, assuming that the variable scenario contains the name (as a string) of the scenario to be considered in the simulation run.
Each traffic and topology structure consist of three fields:

ex.tt.name The name of the scenario as a string.
ex.tt.topology A vector of cell arrays where each cell array defines
{distance (meters), cable name, node name, line name or comment}
ex.tt.traffic A vector of cell arrays where each cell array defines
{from node (referring to topology), to node (refering to topology), tfplan, number of modems}

To define a bridge-tap information about it is added to the definition of a node. The new fields are the length of the bridge-tap, the cable used, a name for the tap, and a name for the cable (typically used to describe the bridge tap length).

GUI:

With a menu selection one can select a scenario among a set of FSAN predefined scenarios. The topology and traffic setup for this scenario is shown as a sub-figure below the setup section. Secondly it is possible to select if a pre defined FSAN noise model should be used or if it should be calculated from the scenario itself. The third menu makes the selection of VDSL modem/duplex method. The PSD mask for this modem is shown in a sub-figure below the setup section.

Connected the the VDSL duplex is the power back off (PBO) method. Currently there is four methods defined: reference length (RefLen), reference FEXT (RefFEXT), reference Noise (RefNoise) and reference frequency (RefFreq). If no PBO is wanted a "None" is selected in the PBO method menu. The PBO parameter is either a length (for RefLen or RefFEXT) or a frequency (for RefFreq). For the RefFreq method a maximum length can be given. Also connected with a duplex method is an efficiency

loss figure where efficiency losses due to cyclic prefix/suffix, guard bands etc. can be stated.

For time division duplex methods a time division can be given for up respectively down stream (for frequency division methods this is given as 1 and 1).

There is three flags that can be set.

- Fast calculation - determines if calculations should use an optimized frequency axis which is slightly (1-2%) less accurate but can be up to 20 times faster.
- Forces HAM band - determines if HAM band suppression should be imposed on VDSL PSD masks.
- Test modem - determines if we should evaluate ADSL modems.

Among the global parameters one can enter new values for

- Background noise (dBm/Hz)
- Maximum SNR available (dB)
- The NEXT, FEXT and third circuit cross talk (3cXT) levels at 1MHz.

With the button "GO" the experiment is evaluated and a result window will appear. With "Close all" all windows will be closed (including the setup window).

Graphical Output

For the plots in the result window:

blue is the recieved signal
red is the total noise
black is the alien noise

Two plot routines have been provided to display parts of the ex structure, the time-frequency plan plot and the topology and traffic information plot:
plotTFplan(tfplan,ftype,fax);
Plots the PSD mask for a certain TF (time/frequency) plan,whereby the function input parameters are as follows:
tfplan tfplan struct for tfplan to be plotted
ftype type of frequency axis as a string ('log' or 'linear')
fax struct for frequency range to be displayed (fax.min, fax. max boundaries of displayed f-axis axis in Hz
plotTTstructure (tt, inScale);
Plots the TT (time and topology) structure, whereby the function input parameters are as follows:
tt tt struct of scenario to be plotted

inScale flag for indicating if the graph should be plotted in scale or not (1 = in scale).
There is also a function to display transmission curves (signal and noise curves) from the evaluated results:

plotResult(ex, result, modemno, side, ftype, fax);
Plots the resulting transmission curves,
ex experiment input parameter structure
result basic simulation output computed by evalExperiment.m
modemno number of the modem for which the result will be plotted (to entry in ex.param.modemlist)
ftype type of frequency axis as a string ('log' or 'linear')
fax struct for frequency range to be displayed (fax.min, fax. max boundaries of displayed f-axis axis in Hz

EXAMPLES:

First Example

This is the first experiment just to show how a simple VDSL experiment can be performed using mostly predefined definitions and routines. To run this experiment change your working directory to src\examples\first and type startup <ENTER> in the MATLAB Command Window. The called startup routine now sets the MATLAB search path to all directories needed and run the the example experiment out of a single Matlab file: Main.m. After starting the example experiment as described above the experiment results will apear within a few seconds on your screen as a text based output in the MATLAB Command Window (indicating bitrates at NT and LT side as well as the downstream to upstream bitrate ratio) and some figures as graphical output. The first figure of the graphical output shows a topology graph (entitled with the name of the examined scenario) indicating additionally the traffic on the loop. The second figure shows the PSD mask used for the considered VDSL service on the loop. Following that, a PSD figure for each service evaluated during the simulation run is shown containing the conditions at NT and LT side. Each diagram in these figures contains graphs for the received signal PSD, the alien noise PSD, and the total noise PSD versus frequency.
In the following a brief description is given to support a better understanding what's going on during the execution of this simulation example. If you are not familiar with

the simulation tool it may be useful to open the Main.m file in the MATLAB editor (by just clicking on the link) during you go through the following explanation steps:
The executable part of the file begins with declaring the input parameter structure ex as a global variable followed by a check regarding numerical interpretation.

After that, the important part for setting up the experiment follows. During execution of this program lines all the input parameters to the simulation are completely defined, i.e. the simulation input parameter structure ex is set up:

The name of the scenario (traffic and topology definitions for the investigated loop) which is considered in the experiment is defined by the comand line
scenario='FSAN scenario FTTEx #1';
This name must correspond to an object in the list of scenarios which is set up a few lines below.

The command
gui.vdslDuplex = 'VDSL-FDD';
defines the name of the time/frequency plan to be used for the VDSL service in the scenario. Similar to above this name must correspond to an object in the list of time/frequency plans which is set up a few lines below.

The line
ex.param = setupParam;
allocates default values for the general simulation parameters contained in the structure ex.param .

In the following two lines the list of time/frequency plans is set up by inserting time/frequency plans of HAM band definitions and some miscellaneous time/frequency plans of services which are also needed for the simulation because the corresponding services are running on the considered loop and therefore they are acting there as sources of alien noise for the investigated services (Compare with traffic/topology graph of the simulation output and see also explanation of structure ex.tfplist and description of functions itu_tfplanHAM and fsan_tfplansMISC)

```
[ex.tfplist, ex.param.HAMBandName] = itu_tfplanHAM([]);
ex.tfplist  = fsan_tfplansMISC(ex.tfplist);
```

The next two lines set up a list of linecode definitions of the xDSL services to be investigated (indicated by the entries in the string matrix ex.param.modemlist, see below and also description of functions setupLClist and fsan_lcdefs).

```
ex.lclist   = setupLClist;
ex.lclist   = fsan_lcdefs(ex.lclist);
```

A list of cable definitions needed for evaluating the loop characteristics during the simulation run is set up by the command
ex.clist = etsi_cables([]);
(see also definition of structure ex.clist and description of function etsi_cables).

The command line
ex.tfplist = etsi_tfplansVDSL(ex.tfplist);
now adds the specific time/frequency plans of the investigated services (modems) to the list of time/frequency plans (see above and also description of etsi_tfplansVDSL).

Further a list of scenarios (traffic/topology definitions) is set up by
gui.ttlist = fsan_loops([]);
For further details see structure of traffic/topology definitions and description of function fsan_loops (within the function fsan_loops the considered scenario 'FSAN scenario FTTEx #1' must of course be specified).

The next command simply fetches the specific considered scenario (variable scenario, see above) from the list of scenrios.
ex.tt   = getList(gui.ttlist,scenario);

Finally the last few lines of the 'experiment setup section' of the program file define what services (modems) should be investigated during the simulation run (should be considered as disturbed modems) and what specific time/frequency plans should be used for the services (the services are indicated in the traffic definition structure only by generic names as for example 'ADSL' and 'VDSL').

The list of the modems to be investigated must be allocated to the parameter ex.param.modemlist and this is done by the command
ex.param.modemlist=['VDSL';'ADSL'];

and the lines
xDSL=getList(ex.param.xDSLlist,'VDSL');
xDSL.used=gui.vdslDuplex;
ex.param.xDSLlist=setList(ex.param.xDSLlist,xDSL.name,xDSL);
xDSL.name='ADSL';
xDSL.used='ADSL';
ex.param.xDSLlist=insertList(ex.param.xDSLlist,xDSL);
are needed to match the generic service names to specific time/frequency plan definitions (see ex.param.xDSLlist)

After the program section described above the experiment is completely defined and the simulation run starts by firstly showing the traffic topology graph and secondly showing the PSDmask for the VDSL modem. This is done by the command sequences:

```
%Show traffic and topology structure
figure(1);
plotTTstructure(ex.tt);

% Show PSD masks for VDSL modem
tfplan = getList(ex.tfplist,gui.vdslDuplex);
figure(2);
gui.fax.min=1e3; gui.fax.max=12e6;
```

```
plotTFplan(tfplan,'Lin',gui.fax);

drawnow; % Show it now
```

For explanation of the function used in this sequences see Description of all Program Files and Functions

After that the experiment is now evaluated by calling the program routine evalExperiment which returns all results within a structure result containing signals and noises for each investigated modem at NT as well as NT side:

```
result = evalExperiment;
```

The next step is now to show the results numerically in the MATLAB Command Window, whereby the function calcFSANresult returns the resulting bitrates (in MBit per second) for each of the investigated modems in upstream and downstream direction:

```
format compact
[bitrate_LT, bitrate_NT]=calcFSANresult(ex,result);
sprintf('LT Rates')
bitrate_LT
sprintf('NT Rates')
bitrate_NT
sprintf('Ratio')
bitrate_NT./bitrate_LT
```

Finally the simulation result is shown graphically by figures showing received singnals and noises at LT and NT side for each of the investigated service (modem). This is done by the sequence:

```
for current=1:length(result),
   figure;
   tmp_str=sprintf('FSAN Duplex Simulation Result, Modem %d (%s-%s)',...
      current, ex.tt.topology{result(current).Modem.LT_Node,3}, ...
      ex.tt.topology{result(current).Modem.NT_Node,3});
   set(gcf,'NumberTitle','off','Name',tmp_str);

   % Plot the LT side
   subplot(211)
   plotResult(ex,result,current,'LT');

   % Plot the NT side
   subplot(212)
   plotResult(ex,result,current,'NT');
end
```

Which figure belongs to which modem is indicated by the title of the figure window, also indicating the node names corresponding to the traffic/topology graph.

A Simple Example

Another simple example exists in directory src\examples\simple. To run it change your working directory to src\examples\first and type startup <ENTER> in the MATLAB Command Window. The called startup routine now sets the MATLAB search path to all directories needed and run the expample experiment calling the main routine ExMain.m. This routine is very similar to the one described in the section regarding the First Example (see above) except for the setup parameter section (program file section, where the experiment is defined). The routine ExMain.m here uses the underlying routine userDefinitionsExample1.m to setup many of the user changable parameters (just a dummy example). This is to show how user defined scenarios can be most easily specified for simulation runs.
After setting up some paramters (identical to the main routine of the First Example, see above) regarding allocation of default values for general simulation parameters, and setting up lists of time/frequency plans (containing HAM band definitions and alien time/fequency plans), linecode definitions, and cables the routine userDefinitionsExample1.m is called by the main routine ExMain.m . This underlying routine now sets up the user defined details of the experiment:

It firstly adds another (default) time/frequency plan definition to the list of time/frequency plans

```
[ex.tfplist, ex.param.HAMBandName] = etsi_tfplanHAM(ex.tfplist);
```

Then it specifies the name for the scenario, the specific name of the service, and it initializes a list of traffic/topology definitions:

```
gui.scenario = 'My own Scenario';
gui.vdslDuplex = 'VDSL-XXX';
gui.ttlist      = [];
```

In the next step the parameter ex.param.xDSLlistis set to specify that the generic name 'VDSL' will be matched to the specific time/frequency plan 'VDSL-XXX' :

```
xDSL=getList(ex.param.xDSLlist,'VDSL');
xDSL.used=gui.vdslDuplex;
ex.param.xDSLlist=setList(ex.param.xDSLlist,'VDSL',xDSL);
```

Now the (default) time/frequency plan added previousely to the list of time/frequency plans is fetched from the list by using the function getList and afterwards it is modified in a way to create a time/frequency plan definition according to the user demands. This is done by the following sequence of commands :

```
tmp_tfplan=getList(ex.tfplist,ex.param.HAMBandName); % initiate tmp_tfplan
tmp_tfplan.name=gui.vdslDuplex;
tmp_tfplan.PSD.downstream ='calcPSD([.3e6 -160 .3e6 -60 3.5e6 -60 3.5e6 -
160],"Linear",ex.param.frequency.f)';
tmp_tfplan.PSD.upstream   ='calcPSD([3.5e6 -160 3.5e6 -60 10e6 -60 10e6 -
160],"Linear",ex.param.frequency.f)';
tmp_tfplan.timeDivision.up=1;  % Time used in up resp. down link
tmp_tfplan.timeDivision.down=1;
tmp_tfplan.timeDivision.sync=1;
tmp_tfplan.PSD.PBO.method='None';
```

```
tmp_tfplan.PSD.PBO.param.freq=2e6;
tmp_tfplan.PSD.PBO.param.len=0;
tmp_tfplan.PSD.PBO.param.maxlen=500; )
tmp_tfplan.lcname='VDSL-theo';
tmp_tfplan.PSD.active.upstream=[0.3e6 10e6];
tmp_tfplan.PSD.active.downstream=[0.3e6 10e6];
tmp_tfplan.PSD.HAM.active=1;
ex.tfplist=insertList(ex.tfplist,tmp_tfplan);
```

Note that after execution of this sequence a new time/frequency plan definition (identified by the name 'VDSL-XXX') is added to the list of time/frequency plans (using function insertList).

In the next steps of userDefinitionsExample1.m it can be seen how to change some linecode specific parameters.

```
lc=getList(ex.lclist,tmp_tfplan.lcname);
lc.param.efficiencyLoss=0.15;
ex.lclist=setList(ex.lclist,lc.name,lc);
lcPrintTheo(lc);
```

The linecode is fetched from the list of linecodes using function getList, the new value for the parameter to be changed is allocated, and finally the modified linecode is returned to the list using the function setList. The last line of the sequence above prints out the linecode settings in the MATLAB Command Window.

Following that the traffic/topology structure of the experiment is defined by the command lines

```
tt.name=gui.scenario;
tt.topology=[
   {0  '' 'CO' ''};
   {500  'DTAG_40' 'N1' '500m'};
   {500  'DTAG_40' 'N2' '500m'};
   {500  'DTAG_40' 'N3' '500m'};
   {1500 'DTAG_40' 'C'  '1500m'};
   {500  'DTAG_40' 'N4' '500m'};
   ];
tt.traffic=[
   {1 2 'VDSL' 3};
   {1 2 'ADSL' 4};
   {1 3 'VDSL' 4};
   {1 4 'ISDN-2B1Q' 3};
   {1 4 'ADSL' 1};
   {1 5 'HDSL-1' 3};
   {5 6 'VDSL' 3};
   {5 6 'ADSL' 4};
   ];
gui.ttlist=insertList(gui.ttlist,tt);
ex.tt=tt;
```

In the first line the name 'My own Scenario' is allocated to the traffic topology structure. Afterwards the topology structure and the traffic structure is specified (see format of traffic and topology definitions), and then inserted into the list of traffic

topology definitions (initialized above). Finally the previously defined scenario is taken as the one to be evaluated during the simulation run (loaded into ex.tt).

The last command of userDefinitionsExample1.m sets up the list of modems to be investigated :
ex.param.modemlist=['VDSL'];
i.e. only VDSL modems are investigated in this example.

Afterwards program execution returns to the main routine ExMain.m and the experiment evaluation (simulation) starts and is executed identically to the First Example (see above).

Example GUI_VDSL

This is the main graphical user interface (GUI) example for VDSL experiments. It is started by calling uiMain.m in the examples/GUI_VDSL directory.
Input parameters
The basic set of input parameters can best be described by looking at the GUI setup window

With a menu selection one can select a scenario among a set of FSAN predefined scenarios. The topology and traffic setup for this scenario is shown as a sub-figure below the setup section. In the second menu it is possible to select if a pre defined FSAN noise model should be used or if it should be calculated from the scenario itself. The third menu makes the selection of VDSL modem/duplex method. The PSD mask for this modem is shown in a sub-figure below the setup section.

Connected to the VDSL duplex is the power back off (PBO) method. Currently there are four methods defined: reference length (RefLen), reference FEXT (RefFEXT), reference Noise (RefNoise) and reference frequency (RefFreq). If no PBO is wanted a "None" is selected in the PBO method menu. Two PBO parameters can be set: a length (for RefLen or RefFEXT) or a frequency (for RefFreq). For the RefFreq method a maximum length can be given. Also connected with a duplex method is an efficiency loss figure where efficiency losses due to cyclic prefix/suffix, guard bands etc. can be stated.

For time division duplex methods a time division can be given for up respectively down stream (for frequency division methods this is given as 1 and 1).

There are three flags that can be set.

Fast calculation - determines if calculations should use an optimized frequency axis which is slightly (1-2%) less accurate but can be up to 20 times faster.
Forces HAM band - determines if HAM band suppression should be imposed on VDSL PSD masks.
Test modem - determines if we should evaluate bit rates also for the ADSL modems in the structure.
Among the global parameters one can enter new values for
Background noise (dBm/Hz)

Maximum SNR available (dB)
The NEXT, FEXT and third circuit cross talk (3cXT) levels at 1MHz.

With the button "GO" the experiment is evaluated and a result window will appear. With "Close all" all windows will be closed (including the setup window).

User definitions
To be able to use new tt scenarios and new tf plans in the grapical user interface the user should make such additions in the file userDefinitions.m which is evaluated before the uiSetup is run.
The following examples show what can be made in the user definitions file:

userDefinitionsExample1.m - Shows many of the possible user changes (just a dummy example)
userDefinitionsBTap.m - Shows how to define a bridge tap scenario.
Output
With the graphical user interface a result window (example) is presented after the main evaluation routine is called. This window contains the rate information as well as figures showing the LT respectively NT side signals and noise curves.
The middle section of the result window contains bit rate information for each of the modems under test. For the downstream and upstream rates the numbers within parentheses are the xtalk margins to the given bitrate.

In the menu at the top left corner of the plot it is possible to select plots from any of the simulated modems. With the PlotIt button the plot is redrawn in a seperate window, thus allowing for further manupulation, e.g., zooming. A logarithmic frequency axis is possible by selecting 'Log' in the menu near f axis type.

For the plots in the result window:

blue is the received signal
red is the total noise
black is the alien noise

A PBO Scenario

In examples/PBO there is an example file ExPBO.m showing an example of a PBO experiment for VDSL using reference-frequency (a.k.a. Constant) power back off (PBO). This experiment tries to reproduce some of the results (curves) in [FSAN VDSL working group, "Power backoff methods for VDSL", ETSI TM6 Luleå 983/17A0, Sweden 22-26 June 1998.]
The whole experiment is run from ExPBO.m and is a good example on how to set up the ex structures to correspond to a certain experiment.

A Mixed Service scenario

In examples/MixedServices there is another PBO test: MixedServices.m. However, this exemplifies how we can run many PBO tests on some VDSL mixed service scenarios.

The whole experiment is run from the MixedServices.m file and is a relatively complex example of how to set up complex PSD definitions (dynamic frequency plan and dynamic PSD mask containment), as well as dynamic tt structures.