

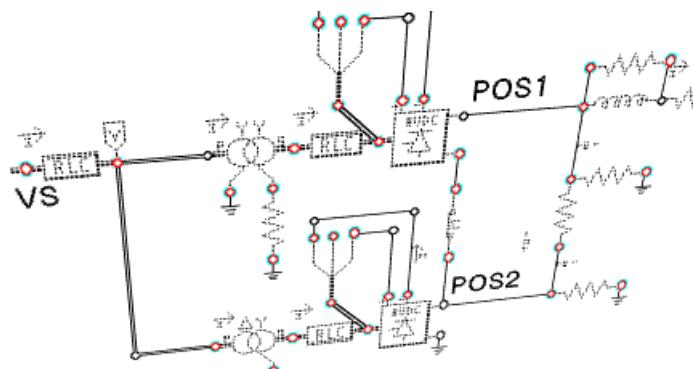
---

# **ATPDRAW**

## **version 7.5**

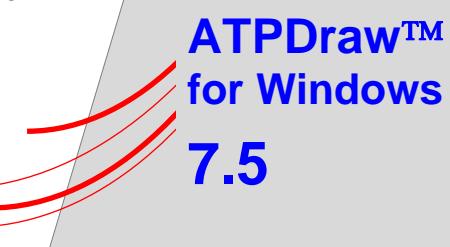
### **for Windows**

## **Users' Manual**



**Hans Kristian Høidalen,  
László Prikler, Francisco Peñaloza**

The manual is made available for distribution via the secure ATP FTP servers and Web sites, as well as via the regional EMTP-ATP Users Groups. An ATP license is required to utilize the ATPDraw program and this manual. Conversion of this manual to other formats and distribution on any kind of media requires explicit permission from the authors.



**Release No. 1.0  
November 2023**

---

## PREFACE

This Users' Manual documents all main features of ATPDraw version 7.3. The manual is an extensive update of the previous User Manual prepared by László Prikler at SYSTRAN Engineering Services Ltd. in Budapest for version 3.5 (SINTEF TR F5680) dated 2002 and the version 5.6 from 2009. Version 7.0 is very much updated compared to version 5.6. This v7.2 covers post-processing. The Reference Manual gives a summary of menu items and menu options. The Advanced Manual covers the features Grouping, Models, electrical machine, line/cable-, and transformer modeling, and optimization. Finally, the Application Manual is extended with several examples, where machine controls, relay protection, PV interface, all based on MODELS scripting, are covered in-depth. New ATPDraw users are advised to start with the Installation and Introductory manuals. Version 7.4 added internal parameter calculation for line and cable parameters that together with Vector Fitting enabled improved JMarti model fitting and the phase domain universal line model. Handling of variables was also significantly updated. Version 7.5 further extends the LCC module and introduces an embedded solver with limited capabilities still under development.

ATPDraw is developed by NTNU from 1999. Program development was earlier financed by Bonneville Power Administration, USA, version 5 in co-operation with EEUG and Schneider Electric, France.

For Norwegian University of Technology Trondheim, Norway, November 26<sup>th</sup>, 2023

Hans Kr. Høidalen  
Professor NTNU-Norway

---

## SUMMARY

ATPDraw is a graphical, mouse-driven preprocessor to the ATP version of the Electromagnetic Transients Program (EMTP) on the MS-Windows platform. The program is written in Embarcadero Delphi XE8 and runs under Windows 9x/NT/2000/XP/Vista/10. In ATPDraw the user can construct an electrical circuit using the mouse and selecting components from menus, then ATPDraw generates the ATP input file in the appropriate format based on "what you see is what you get". The simulation program ATP and plotting programs can be integrated with ATPDraw.

ATPDraw supports multiple circuit modeling that makes possible to work on more circuits simultaneously and copy information between the circuits. All kinds of standard circuit editing facilities (copy/paste, grouping, rotate, export/import, undo/redo) are available. In addition, ATPDraw supports the Windows clipboard and metafile export. The circuit is stored on disk in a single project file, which includes all the simulation objects and options needed to run the case. The project file is in zip-compressed, binary format that makes the file sharing with others very simple. The project file format is extensive changed in v7 with support of XML-formats as well.

Most of the standard components of ATP as well as TACS are supported, and in addition the user can create new objects based on MODELS or \$Include (Data Base Module). Line/Cable modeling (KCLee, PI-equivalent, Semlyen, JMarti, Noda and ULM) is also included in ATPDraw where the user specifies the geometry and material data and has the option to view the cross section graphically and verify the model in the frequency domain. Special components support the user in machine and transformer modeling based on the powerful Universal Machine and BCTRAN components in ATP-EMTP. In addition, the advanced Hybrid Transformer model XFMR and Windsyn support is included.

ATPDraw supports hierarchical modeling by replacing selected group of objects with a single icon in an unlimited number of layers. Components have an individual icon in either bitmap or vector graphic style and an optional graphic background. Version 7 of ATPDraw removes limits on the number and size of components and circuits.

---

## TABLE OF CONTENTS

	<i>Page</i>
<b>1. Introduction . . . . .</b>	<b>9</b>
1.1 What is ATPDraw?.....	11
1.2 What is ATP? .....	12
1.3 Operating principles and capabilities of ATP .....	12
1.3.1 Integrated simulation modules in ATP .....	13
1.3.2 Program capabilities .....	14
1.3.3 Main characteristics of plotting programs for ATP .....	14
1.3.4 Typical EMTP applications .....	15
1.3.5 Hardware requirements for ATP.....	16
1.4 Main characteristics of the embedded solver .....	16
1.5 Contents of this manual.....	17
1.6 Manual conventions.....	18
<b>2. Installation Manual . . . . .</b>	<b>19</b>
2.1 ATP licensing policy .....	21
2.2 How to download ATPDraw? .....	21
2.3 Program installation.....	22
2.4 Files and sub-folders in the ATPDraw system folder .....	22
2.4.1 Organizing the files.....	24
2.4.2 Configuring ATPDraw .....	24
2.4.3 ATPDraw command line options .....	24
2.4.4 Drag and drop project files .....	24
2.5 Interfacing ATPDraw with other programs of the ATP-EMTP package.....	25
2.6 Installing ATPDraw on servers with limited writing privileges .....	27
2.7 How to get help?.....	28
2.7.1 Help via the ATPDraw web-page forum .....	28
2.7.2 Help via the ATP-EMTP-L mailing list .....	28
2.7.3 Help from the author of ATPDraw .....	29
2.8 Available components in ATPDraw.....	29
<b>3. Introductory Manual . . . . .</b>	<b>31</b>
3.1 Operating windows.....	33
3.2 Operating the mouse.....	36
3.3 Edit operations.....	37
3.4 Overview of working with ATPDraw .....	37
3.5 Your first circuit ( <i>Exa_1.acp</i> ).....	40
3.5.1 Building the circuit .....	41
3.5.2 Storing the project file on disk.....	51
3.5.3 Creating the ATP input file.....	51
3.5.4 Running the simulation.....	53
3.5.5 Simulating with the internal solver .....	53
3.6 Node names .....	53
3.6.1 Multi-phase circuits .....	54
3.7 Data values .....	57
3.7.1 Parameter variations .....	57
3.8 Object organization, sequence and priorities.....	58
3.9 Post-processing.....	59

---

3.9.1	Probes .....	59
3.9.2	Power System Toolbox .....	59
3.9.3	COMTRADE.....	60
3.9.4	Embedded Plotting .....	60
<b>4.</b>	<b>Reference Manual . .</b>	<b>63</b>
4.1	Main window .....	65
4.2	Main menu .....	66
4.2.1	File.....	66
4.2.2	Edit .....	68
4.2.3	View .....	73
4.2.4	Zoom In .....	76
4.2.5	ATP .....	78
4.2.6	Library .....	95
4.2.7	Tools.....	102
4.2.8	Window .....	114
4.2.9	Help .....	115
4.3	Shortcut menu .....	117
4.4	Component selection menu.....	117
4.5	Component dialog box.....	118
4.6	Connection dialog box .....	122
4.7	Text dialog box .....	123
4.8	Shape dialog box .....	123
4.9	Picture dialog box .....	123
4.10	Attachment file dialog box .....	124
4.11	Plot dialog box.....	124
4.12	Node dialog box.....	125
4.13	Probe dialog box .....	127
4.14	Selection dialog .....	128
4.15	Circuit objects in ATPDraw .....	129
4.15.1	Probes & 3-phase .....	131
4.15.2	Branch Linear.....	133
4.15.3	Branch Nonlinear .....	135
4.15.4	Lines/Cables.....	136
4.15.5	Switches .....	140
4.15.6	Sources .....	141
4.15.7	Machines .....	143
4.15.8	Transformers .....	144
4.15.9	MODELS .....	145
4.15.10	TACS .....	151
4.15.11	User Specified.....	155
4.15.12	Steady-state .....	156
4.15.13	Power System Tools.....	157
4.15.14	All standard Comp.....	161
4.15.15	Add objects .....	162
4.15.16	Plugins.....	162
<b>5.</b>	<b>Advanced Manual . .</b>	<b>163</b>
5.1	Compress- multilevel modeling .....	165
5.1.1	Compressing nonlinear objects .....	170
5.2	Non-standard component dialog boxes.....	172

---

5.2.1	Saturable 3-phase transformer .....	173
5.2.2	Universal machines.....	175
5.2.3	Statistic/systematic switch .....	179
5.2.4	Harmonic source .....	180
5.2.5	Windsyn components.....	181
5.3	Using the integrated LCC object for line/cable modeling .....	183
5.3.1	Model and Data page settings for Overhead Lines .....	187
5.3.2	Model and Data page settings for Single Core Cable systems .....	191
5.3.3	Model and Data page settings for Enclosing Pipe type cables .....	193
5.3.4	Matrix Import.....	194
5.3.5	Node page settings .....	195
5.3.6	LCC Section.....	195
5.4	Verification of the Line/Cable model performance.....	198
5.4.1	Internal Line/Cable Verify .....	198
5.4.2	Tuning internal calculation of JMarti and ULM models .....	202
5.4.3	External Line Check .....	204
5.5	Using MODELS simulation language .....	206
5.5.1	The default approach .....	207
5.5.2	The MODELS editor .....	208
5.5.3	The manual approach.....	214
5.5.4	Recording internal MODELS variables.....	217
5.6	BCTRAN support in ATPDraw .....	218
5.7	Hybrid Transformer, XFMR .....	221
5.7.1	Overview.....	221
5.7.2	XFMR dialog box .....	223
5.8	Creating new circuit objects in ATPDraw.....	226
5.8.1	Creating a 6-phase rectifier bridge.....	226
5.8.2	Creating a user specified, nonlinear transformer model .....	232
5.9	Systematic parameter variations.....	235
5.9.1	Optimization .....	238
5.9.2	MonteCarlo simulations ( <i>Exa_21.acp</i> ) .....	242
6.	<b>Application Manual . . .</b>	<b>247</b>
6.1	Switching studies using JMarti LCC objects.....	249
6.1.1	JMarti model of a 750 kV line .....	249
6.1.2	Line to ground fault and fault tripping transients ( <i>Exa_7a.acp</i> ).....	251
6.2	Lightning overvoltage study in a 400 kV substation ( <i>Exa_9.acp</i> ) .....	254
6.3	Modeling Rectifiers, zigzag transformers and analysis of Harmonics ( <i>Exa_14.acp</i> ) .....	259
6.4	The Controlled Electric Rotating Machines .....	266
6.4.1	Synchronous machine control ( <i>Exa_22a.acp</i> ) .....	267
6.4.2	Universal machine control ( <i>Exa_22b.acp</i> ) .....	273
6.4.3	Data used in the ATPDraw cases described Chapt. 6.4.1-6.4.2 ( <i>Exa_22</i> ).....	276
6.4.4	The Controlled Induction Machines ( <i>Exa_23.acp</i> ).....	279
6.4.5	Windsyn machine control ( <i>Exa_17.acp</i> ) .....	287
6.5	Simulating transformer inrush current transients .....	292
6.5.1	Energization of a 400/132/18 kV auto-transformer ( <i>Exa_10.acp</i> ) .....	292
6.5.2	Energization of a 132/15 kV generator step-up transformer ( <i>Exa_11.acp</i> ) .....	299
6.5.3	Using the Hybrid Transformer component ( <i>Exa_16.acp</i> ) .....	303
6.6	Switching overvoltage studies with statistical approach ( <i>Exa_12.acp</i> ) .....	305
6.6.1	Setting program options for the statistical simulation .....	305
6.6.2	Results of the statistical study.....	306

---

6.7	Power system protection ( <i>Exa_24.acp</i> ) .....	309
6.8	Solar power interface via PWM controlled inverter ( <i>Exa_25.acp</i> ) .....	313
6.9	Post-processing .....	316
6.9.1	COMTRADE generation.....	317
6.9.2	Embedded plotting .....	320
7.	<b>Appendix . . . . .</b>	<b>323</b>
7.1	PFC simulations in ATPDraw.....	325
7.2	Line Check .....	328
7.2.1	Single phase systems .....	329
7.2.2	3-phase systems .....	332
7.3	Hybrid Transformer, XFMR .....	333
7.3.1	Leakage inductance .....	334
7.3.2	Winding resistance .....	335
7.3.3	Capacitance .....	336
7.3.4	Core .....	338
7.4	Windsyn manufacturers data input and controls .....	341
7.4.1	Induction machine modeling .....	341
7.4.2	The ATPDraw fitting approach.....	343
7.4.3	ATPDraw input dialogs.....	344
7.4.4	Synchronous machine modeling .....	346
7.4.5	Machine controls .....	347
7.5	Power system toolbox calculators .....	350
7.5.1	Filtering and down-sampling.....	351
7.5.2	Phasor calculations .....	352
7.5.3	Power and impedance calculations.....	354
7.5.4	FFT/DFT algorithm test .....	354
7.6	LCC Internal Calculation .....	355
7.6.1	Overhead line modelling .....	356
7.6.2	Underground cable modelling .....	357
7.7	Internal Solver.....	360
7.7.1	System description .....	361
7.7.2	Interpolation .....	361
7.7.3	Object oriented structure .....	362
7.8	XML data exchange .....	362
7.8.1	ATPDraw coordinate system.....	363
7.8.2	The XML format definition; DTD-file.....	364
7.8.3	XML skeleton.....	367
7.9	ATPDraw data structure and object model .....	368
7.10	Examples project distributed with ATPDraw v7.5 .....	370
7.11	References.....	371
7.12	Index .....	373



# **1. Introduction . . .**



**ATPDraw™  
for Windows**

**7.5**



## 1.1 What is ATPDraw?

ATPDraw™ for Windows is a graphical, mouse-driven preprocessor to the ATP version of the Electromagnetic Transients Program (EMTP). In ATPDraw the user can construct the digital model of the circuit to be simulated using the mouse and selecting predefined components from an extensive palette, interactively. Then ATPDraw generates the input file for the ATP simulation in the appropriate format based on "what you see is what you get". Circuit node naming is administrated by ATPDraw, thus the user needs to give a name only to nodes having special interest.

ATPDraw has a standard Windows layout and offers a large Windows help file system. All kinds of standard circuit editing facilities (copy/paste, grouping, rotate/flip, export/import, undo/redo) are available. Other facilities in ATPDraw are: built-in editor for MODELS and ATP-file editing, text viewer for displaying the output LIS-file of ATP, automatic LIS-file checking with special trigger strings to detect simulation errors, support of Windows clipboard and metafile export. ATPDraw supports multiple circuit modeling that makes possible to work on more circuits simultaneously and copy information between the circuits.

Most of the standard components of ATP (both single and 3-phase), as well as TACS are supported, and in addition the user can create new objects based on MODELS or \$INCLUDE (Data Base Module). Line/Cable modeling (KCLee, PI-equivalent, Semlyen, JMarti and Noda) is also included in ATPDraw where the user specifies the geometry and material data and has the option to view the cross section graphically and verify the model in the frequency domain. Objects for Harmonic Frequency Scan (HFS) have also been added. Special objects help the user in machine and transformer modeling including the powerful UNIVERSAL MACHINE and BCTRAN features of ATP. An advanced Hybrid Transformer model based on Test Report, Design or Typical values with topologically correct core is also supported. ATPDraw also integrated with Windsyn for Universal Machine modeling based on manufacturers data.

ATPDraw supports hierarchical modeling to replace a selection of objects with a single icon in unlimited numbers of layers. Data parameters can in most cases be assigned to global variables. Both ATP's native method \$PARAMETER and an embedded Internal Parser option is supported, allowing the user to specify a text string as input in a components' data field, then assign numerical values or expressions to these text strings later. The circuit is stored on disk in a single project file, which includes all the simulation objects and options needed to run the case. The project file is in zip-compressed format that makes the file sharing with others very simple.

ATPDraw is most valuable to new users of ATP-EMTP and is an excellent tool for educational purposes. However, the possibility of multi-layer modeling makes ATPDraw a powerful front-end processor for professionals in analysis of electric power system transients, as well.

Version 7.5 is written in Embarcadero Delphi XE10.4. The compiled help file system supported from Windows VISTA is used.

ATPDraw™ is a trademark and copyrighted by © 2005-2023 Norwegian University of Science and Technology, Norway. Program developer is Dr. Hans Kristian Høidalen in Trondheim, Norway.

The ATPDraw program is royalty free and can be downloaded free of charge from several Internet sites.

## 1.2 What is ATP?

The Alternative Transients Program (ATP) is one of the most widely used universal program system for digital simulation of transient phenomena of electromagnetic as well as electromechanical nature in electric power systems. With this digital program, complex networks and control systems of arbitrary structure can be simulated. ATP has extensive modeling capabilities and additional important features besides the computation of transients.

The Electromagnetic Transients Program (EMTP) was developed in the public domain at the Bonneville Power Administration (BPA) of Portland, Oregon prior to the commercial initiative in 1984 by the EMTP Development Coordination Group and the Electric Power Research Institute (EPRI) of Palo Alto, California. The birth of ATP dates to early in 1984, when Drs. Meyer and Liu did not approve of proposed commercialization of BPA's EMTP and Dr. Meyer, using his own personal time, started a new program from a copy of BPA's public-domain EMTP. Since then the ATP program has been continuously developed through international contributions by Drs. W. Scott Meyer and Tsu-huei Liu, the co-Chairmen of the Canadian/American EMTP User Group. Several experts around the world have been contributing to EMTP starting in 1975 and later to ATP in close cooperation with program developers in Portland, USA.

Whereas BPA work on EMTP remains in the public domain by U.S. law, ATP is *not* in the public domain and licensing is required before access to proprietary materials is granted. Licensing is, however, available free of all charge to anyone in the world who has not participated voluntarily in the sale or attempted sale of any electromagnetic transients program, (hereafter called "EMTP commerce").

## 1.3 Operating principles and capabilities of ATP<sup>1</sup>

The ATP program predicts variables of interest within electric power networks as functions of time, typically initiated by some disturbances. Basically, trapezoidal rule of integration is used to solve the differential equations of system components in the time domain. Non-zero initial conditions can be determined either automatically by a steady-state phasor solution or they can be entered by the user for simpler components.

ATP has many models including rotating machines, transformers, surge arresters, transmission lines and cables. Interfacing capability to the program modules TACS (Transient Analysis of Control Systems) and MODELS (a simulation language) enables modeling of control systems and components with nonlinear characteristics such as arcs and corona. Dynamic systems without any electrical network can also be simulated using TACS and MODELS control system modeling. Symmetrical or unsymmetrical disturbances are allowed, such as faults, lightning surges and several kinds of switching operations including commutation of valves. Frequency-domain harmonic analysis using harmonic current injection method (HARMONIC FREQUENCY SCAN) and calculation of the frequency response of phasor networks using FREQUENCY SCAN feature is also supported. The model-library of ATP at present consists of the following components:

- Uncoupled and coupled linear, lumped R,L,C elements.
- Transmission lines and cables with distributed and frequency-dependent parameters.
- Nonlinear resistances and inductances, hysteretic inductor, time-varying resistance, TACS/MODELS controlled resistance.

---

<sup>1</sup> Source: [WWW.EMTP.ORG](http://WWW.EMTP.ORG)

- Components with nonlinearities: transformers including saturation and hysteresis, surge arresters (gapless and with gap), arcs.
- Ordinary switches, time-dependent and voltage-dependent switches, statistical switching (Monte-Carlo studies).
- Valves (diodes, thyristors, triacs), TACS/MODELS controlled switches.
- Analytical sources: step, ramp, sinusoidal, exponential surge functions, TACS/MODELS defined sources.
- Rotating machines: 3-phase synchronous machine, universal machine model.
- User-defined electrical components that include MODELS interaction

### 1.3.1 Integrated simulation modules in ATP

**MODELS** in ATP is a general-purpose description language supported by an extensive set of simulation tools for the representation and study of time-variant systems.

- The description of each model is enabled using free-format, keyword-driven syntax of local context and that is largely self-documenting.
- MODELS in ATP allows the description of arbitrary user-defined control and circuit components, providing a simple interface for connecting other programs/models to ATP.
- As a general-purpose programmable tool, MODELS can be used for processing simulation results either in the frequency domain or in the time domain.

**TACS** is a simulation module for time-domain analysis of control systems. It was originally developed for the simulation of HVDC converter controls. For TACS, a block diagram representation of control systems is used. TACS can be used for the simulation of

- HVDC converter controls
- Excitation systems of synchronous machines
- power electronics and drives
- electric arcs (circuit breaker and fault arcs).

Interface between electrical network and TACS is established by exchange of signals such as node voltage, switch current, switch status, time-varying resistance, voltage- and current sources.

**Supporting routines** are integrated utilities inside the program that support the users in conversion between manufacturers' data format and the one required by the program, or to calculate electrical parameters of lines and cables from geometrical and material data. Supporting modules in ATP are:

- Calculation of electrical parameters of overhead lines and cables using program modules LINE CONSTANTS, CABLE CONSTANTS and CABLE PARAMETERS.
- Generation of frequency-dependent line model input data (Semlyen, JMarti, Noda line models).
- Calculation of model data for transformers (XFORMER, BCTRAN).
- Saturation and hysteresis curve conversion.
- Data Base Modularization (for \$INCLUDE usage).

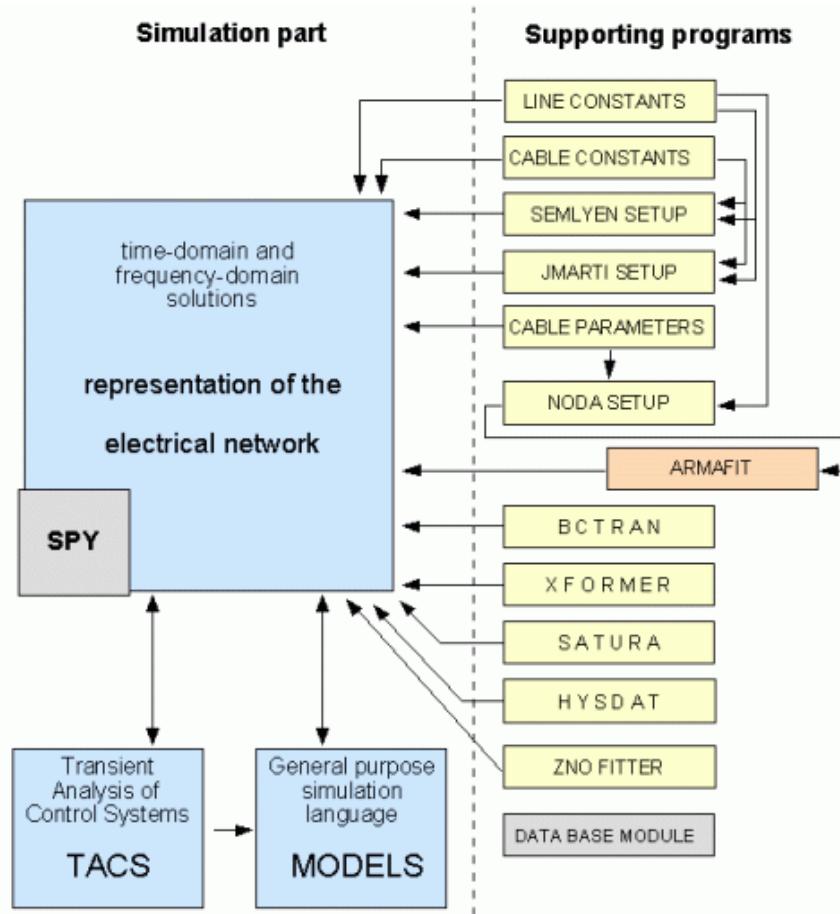


Fig. 1.1 - Supporting routines in ATP.

Source: www.emtp.org

### 1.3.2 Program capabilities

ATP-EMTP tables are dimensioned dynamically at the start of execution to satisfy the needs of users and their hardware (e.g., RAM). No absolute limits have ever been observed, and the standard version has limits that average more than 20 times default table sizes. Today, the largest simulations are being performed using Intel-based PC's. The following table shows maximum limits for standard program distribution.

Busses	6000	Sources	900
Branches	10000	Nonlinear elements	2250
Switches	1200	Synchronous machines	90

### 1.3.3 Main characteristics of plotting programs for ATP

These post-processors are interfaced with ATP via disk files and their main function is to display the results of a time- or frequency domain simulation. ATP simulation data are stored in a file having extension .pl4, and it can be processed either off-line, or on-line. The latter (i.e. to display results while the simulation proceeds) is available only if the operating system provides concurrent PL4-file access for ATP and the postprocessor program.

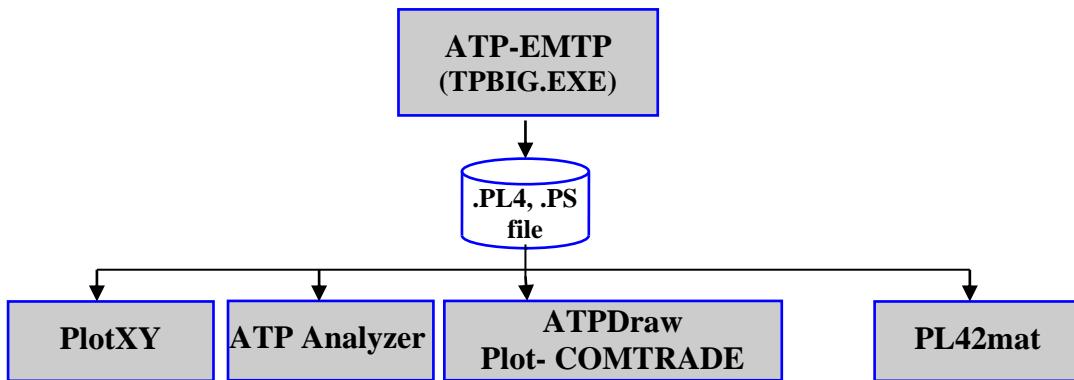


Fig. 1.2 – Plotting programs for ATP.

**ATP Analyzer** is a Windows based program intended for observing and analyzing analog signals and discrete channel data associated power generation, transmission and distribution systems. The program is capable of reading and displaying analog signals produced by ATP as type PL4 output file data, industry standard COMTRADE file and analog and digital data produced from protective relays and fault recorder equipment, analog signals from table ASCII text data, and audio wave files. A total of 254 signals can be managed.

Signals can be displayed in time domain in multiple overlay charts. One or more signals can be displayed as a function of another on an X versus Y chart. Up to three signals can be displayed simultaneously in the frequency domain as harmonics or as a broad frequency spectrum. Charts may be printed and copied to the Windows clipboard. The program can process the data for harmonic content and store processed data in a Windows Access Data base.

Developer: Bonneville Power Administration, USA.

Licensing: Distributed at no cost to the licensed ATP users.

Distribution: EEUG annual CD distribution, EEUG, JAUG secure Web sites.

**PlotXY** is a WIN32 plotting program originally designed for ATP-EMTP. The program is mainly designed to make, as easy and fast as possible, line plots in Microsoft Windows environments. It is also able to perform some post-processing on the plotted curves: algebraic operations, computation of the Fourier series coefficients. The program has an easy-to-use graphical user interface, and the 32-bit code provides very fast operation. Up to 3 PL4 or ADF files can be simultaneously held in memory for easy comparison of different data and up to 8 curves per plots versus time, or X-Y plots are allowed. The program has a clever automatic axis scaling capability and able to make plots with two independent vertical axes and provides easy tools for factors, offsets and zoom support, and a graphical cursor to see values in numerical format. Screen plots can be exported as Windows Metafile via win32 clipboard.

Developer: Dr. Massimo Ceraolo, ceraolo@dsea.unipi.it, University of Pisa, Italy.

Licensing: “acknowledgeware”. Distributed at no cost to the licensed ATP users. If user keeps it beyond the 30-day trial period, he/she must send an acknowledgement letter to the developer.

Distribution: EEUG annual CD distribution, EEUG, JAUG and MTU secure FTP sites.

Web-site: <http://ceraolo-plotxy.ing.unipi.it/default.htm>

Main characteristics of other postprocessors for ATP are summarized in [6].

### 1.3.4 Typical EMTP applications

ATP-EMTP is used world-wide for switching and lightning surge analysis, insulation coordination and shaft torsional oscillation studies, protective relay modeling, harmonic and power quality studies, HVDC and FACTS modeling. Typical EMTP studies are:

- Lightning overvoltage studies
- Switching transients and faults
- Statistical and systematic overvoltage studies
- Very fast transients in GIS and groundings
- Machine modeling
- Transient stability, motor startup
- Shaft torsional oscillations
- Transformer and shunt reactor/capacitor switching
- Ferro resonance
- Power electronic applications
- Circuit breaker duty (electric arc), current chopping
- FACTS devices: STATCOM, SVC, UPFC, TCSC modeling
- Harmonic analysis, network resonances
- Protection device testing

### 1.3.5 Hardware requirements for ATP

ATP is available for most Intel based PC platforms under DOS, Windows 3.1/9x/NT, OS/2, Linux and for other computers, too (e.g., Digital Unix and VMS, Apple Mac's, etc.). Most users, including program developers, use Intel Pentium-based PCs with MS-Windows 9x/NT. A standard Pentium PC configuration with min. 128 MB RAM, hard disk (20 MB free space) and VGA graphics is sufficient to execute ATP under MS-Windows. Most popular program versions are at present:

- MS-Windows XP/7-11: 32-bit *GNU-Mingw32*
- *Linux*: GNU version of ATP

## 1.4 Characteristics and capability of the embedded solver

The aim of the embedded solver is to support:

- Robust modeling of switches and nonlinearities to avoid numerical oscillations.
- State-of-the-art cable models.
- User-defined components through DLL features.
- More flexible scripting feature with support of Python and MatLab.
- The simplicity and speed of ATP.
- Backward compatibility with ATP for verification. Write PL4-file.
- The solver is invoked by holding down the Ctrl-key while selecting Run ATP.

The solver has the following characteristics:

- Controls are NOT implemented in Version 7.5. This is a main drawback that reduce the practical application areas significantly.
- Electrical machines are NOT supported in Version 7.5. Neither are the BCTRAN and XFMR components.
- The solver in v7.5 supports around 70 components. All linear branches, ideal and saturable transformers, some nonlinear branches, LCC, some switches and all sources. In addition comes a few report-only controls; WriteMinMax, Extract1/3, steady-state probe phasors.
- Uses interpolation of all states and history sources in its core. Identifies exact switching instances and gives a (potential) robust handling of power electronics. It uses interpolation also for (pseudo)nonlinear resistors and inductors, including type 96.
- Implements steady-state initialization and Frequency Scan for all components.

- Supports the universal line model (ULM) for cable and overhead line modeling.
- The solver itself is based on partial LU-factorization and sparse matrix storage. No limits on the system size. Scales well with the number of nodes.
- Multiple-run is supported with execution of the solver in parallel threads. The solution is thread-safe as no external data or files are involved in the process.
- Progress report is added also for single simulation.
- The solver plots result in a debug window that gives all plots, debug info, and simulation speed plot. This will be optionally turned off in future releases but makes sense now.
- The solver creates a PL4-file and a LIS-file (not complete at this stage).
- The embedded solver is for now invoked by holding down the Ctrl-key when run ATP is selected.

The following components are supported in ATPDraw v7.5:

- Probes and 3-phase
  - PROBE\_I, PROBE\_V, SPLITTER, TRANSP1-4
- Linear branches
  - RESISTOR, IND\_RP, CAP\_RS, RLC, RLC\_PHASOR, PQU, PQU\_PHASOR, RINF, (all with multi-phase), RLC3, RLCD3, RLCY3
- Transformers
  - SATTRAFO, SATTRAFO1, TRAFO\_S, TRAFO\_I, TRAFO\_I3
  - Not supported: BCTRAN and XFMR.
- Non-linear branches
  - NLINRES, NLININD, NLIND96 (all with multi-phase and managed with interpolation)
  - Not supported compensation-based (true) nonlinear. Could be replaced with the pseudo-nonlinear components.
- Line/Cables
  - LINEPI\_1 - LINEPI\_5, LINEPI3S, LINEPI6S, PI\_CAB3S
  - LINERL\_1 - LINERL\_6, LINERL3S, LINERL012, LINERL3F, LINERL6N, LINERL6S
  - LINEZT\_1, LINEZT\_2, LINEZT\_3, LINEZT\_6, LINEZT\_9, LINEZU\_2, LINEZU\_3
  - LCC with geometry, material data or external ZY-matrices. Internal calculation required. Bergeron, PI and ULM (JMarti not yet) supported.
- Switches
  - SWMEAS, TSWITCH, SWIT\_3X3, SWITCHVC
  - DIODE, DIONEN
- Sources
  - DC1PH, DC1PHUG
  - ACSOURCE, AC1PH, AC1PHUG
  - PULSE10, RAMP, SAW10, SLOPE\_RA
  - HEIDLER, HEIDLERF, STANDLER, STANDLERF, SURGE, TWOEXP
- Machines
  - Not supported.
- Controls (TACS and MODELS) are not supported.
  - A few "report-only" components that is based on MODELS are supported; steady-state probes (PROBE\_V, PROBE\_I), WriteMaxMin, EXTRACT1, EXTRACT3.

## 1.5 Contents of this manual

This User's Manual of ATPDraw for Windows 7.5 contains five parts:

### INSTALLATION MANUAL

- How to obtain the ATP license
- How to download ATPDraw
- How to install ATPDraw
- Hardware requirements

- How to configure your system
- How to use ATPDraw as operating shell for other ATP simulations
- How to communicate with other users and program developers

## **INTRODUCTORY MANUAL**

- How to create a circuit in ATPDraw
- Operating windows
- Your first circuit
- Three-phase circuits and connections

## **REFERENCE MANUAL**

- Reference of main menu items and program options
- Reference of the Component, Text, Connection, Node and Group dialog boxes
- Reference of ATPDraw circuit objects

## **ADVANCED MANUAL**

- How to create multi-level group components in ATPDraw
- How to use the integrated LCC object for line/cable modeling
- How to verify Line/Cable models
- How to use MODELS in ATPDraw?
- How to use the integrated BCTRAN object for transformer modeling
- How to use the Hybrid Transformer component
- How to create new circuit objects based on DBM and \$INCLUDE
- How to use parameter variations systematically, optimization++

## **APPLICATION MANUAL**

- Switching studies using JMarti LCC objects in a 750 kV system
- Lightning overvoltage in a 400 kV substation
- Analysis of harmonics in industrial AC/DC systems
- Controlling electrical machines (synchronous, induction, universal, windsyn)
- Simulation of inrush current transients
- Line energization studies with statistical approach
- Power system protection (IEEE 9-BUS, distance relays)
- Solar power interface with PWM controlled inverter
- Post-processing (Comtrade, Embedded Plotting)

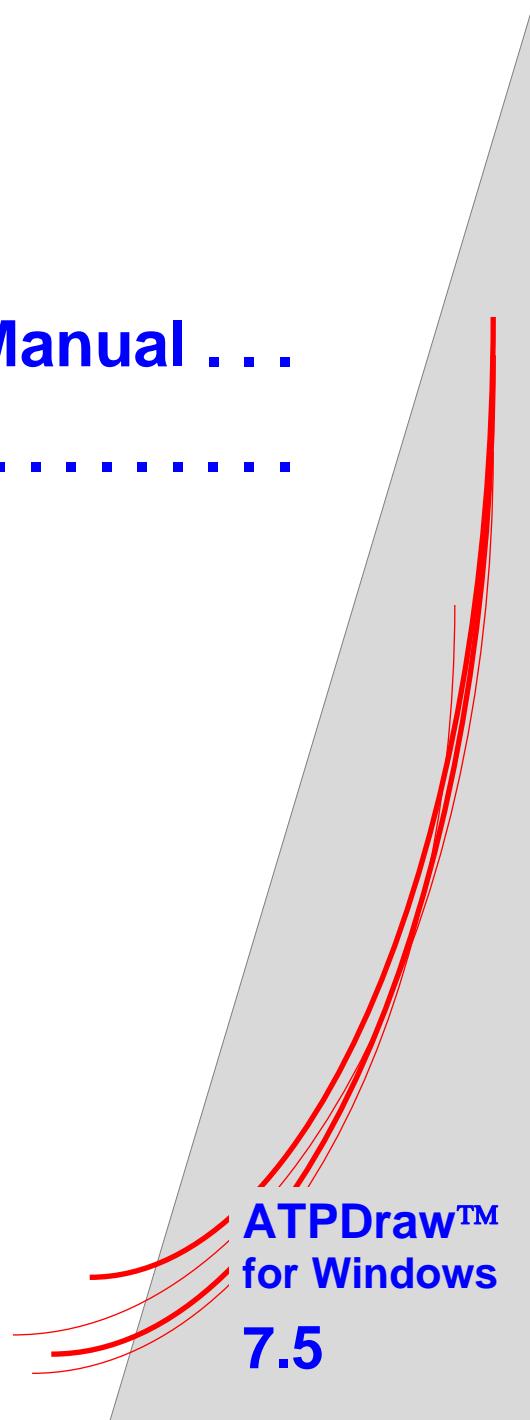
### **1.6 Manual conventions**

The following typographical conventions are used in this manual:

- Italic*: Menus in ATPDraw
- E.g.: Select *Edit / Rotate L* : Select *Rotate L* command in the main menu *Edit*.
- Courier 9 – 10**: Data files.
- E.g.: Listing of ATP input files, MODELS code, etc.  
Description of menu options in component dialog boxes.
- Courier 11 – 12**: Data code and file names.
- E.g.: Give the file the name HVDC\_6.LIB and store it in the \USP directory.  
The \USP directory is a directory under the main directory of ATPDraw.
- Courier 12** : Commands on the DOS prompt.
- E.g.: C:\TMP>**setup**: Type the command **setup** at C:\TMP>.

## **2. Installation Manual . . .**

.....





## 2.1 ATP licensing policy

ATPDraw and the present documentation includes ATP proprietary information, thus *ATP licensing is mandatory* prior to get permission to download the program from the Internet. ATP license is free of all charge for all who have not engaged in EMTP commerce, and it can be obtained from the Canadian/American EMTP User Group, or an authorized regional users group. In general, organizational licensing is preferred over licensing of individuals. Undergraduate students are not licensed personally. If ATP usage is to be organizational rather than personal (i.e., if ATP materials are to be used by, in, for, or on behalf of, a company, university, etc.), the licensee must certify that the organization has not participated in EMTP commerce -- nor has any employee, contractor, or other agent who would be granted access to ATP materials. Once one is licensed, he/she is authorized to download ATP materials from the secure Internet sites or obtain them from a similarly licensed user, or order these materials from the regional user groups.

At present the Canadian/American, European and the Japanese user groups accepts ATP license applications via the Internet. Interested parties are requested to visit the on-line licensing page at [www.atp-emtp.org](http://www.atp-emtp.org), fill-in and submit the appropriate web-form. Potential users of other continents must follow the licensing procedure of their regional EMTP user group. Geographical location of ATP-EMTP user groups and contact information details are shown below:

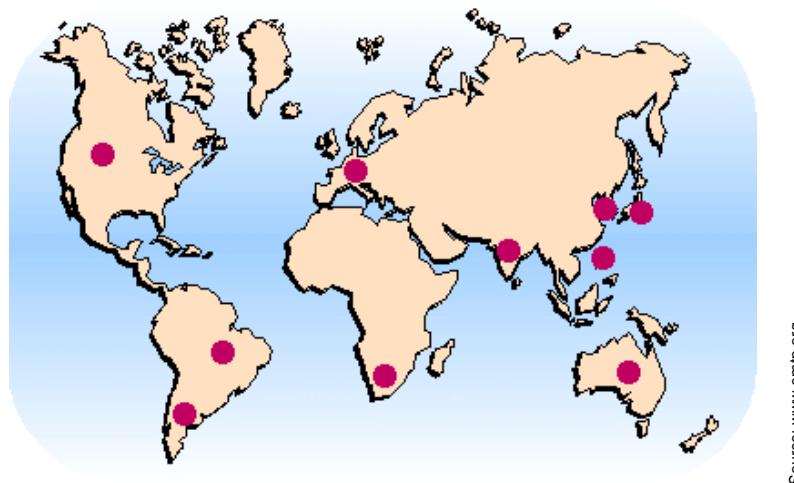


Fig. 2.1 - Location of ATP-EMTP user groups.

Chapter 2.7.2 of the Installation Manual gives further information about the ATP related Internet resources.

## 2.2 How to download ATPDraw?

ATP licensing is mandatory prior to receiving any materials. Following the license agreement approval by an authorized user group, you are eligible to use the ATP program and all ATP related tools, like ATPDraw and this manual. There are different sources of obtaining ATPDraw and additional ATP related tools and program manuals:

- Register at <https://www.atp-emtp.org> and apply for license. When granted, the ATP solver and supporting programs, including ATPDraw can be downloaded.
- Register and download the ATPDraw program itself without the solver or plotter from <https://www.atpdraw.net/>

## 2.3 Program installation

The /atpdraw subfolder under the above secure servers contains a zip-compressed archive `atpdraw7x.zip`, a short installation guide and the latest patch file (if any). Following a successful download of the distribution kit, perform the next operations:

- 1) Copy the `atpdraw7x_install.zip` file into a temporary directory and unzip it.
- 2) Run the program **Setup.exe**. The installation process will be assisted by a standard *Inno Setup* Wizard.
- 3) Specify a destination directory for ATPDraw when prompted. Note that in some cases, dependent on ATP versions, the *Result Directory* (/ATP folder default) should be without blanks in the path name.
- 4) The installation process will be completed after creating a new shortcut for ATPDraw under *Start / Programs / ATPDraw*. When you start ATPDraw.exe first time the sub-folders /ATP, /Projects, /GRP, /BCT, /HLP, /LCC, /MOD will be confirmed and optionally created. The user must have writing privileges to the *Result Directory* (/ATP default).
- 5) Download the latest patch file called `patchxv7.zip` (if exists on the server), then unzip it and simply overwrite the existing files in the ATPDraw system folder with the newer ones received in the patch file.

The program installation will create a directory structure as shown next. ATPDraw can be uninstalled in the standard manner using Windows' uninstaller (*Start menu / Settings / Control Panel / Add/Remove programs*).

```

04.11.2019 22:48      10 260 480 ATPDraw.exe
28.10.2019 20:34      269 494 ATPDraw.scl
03.07.2019 20:37      1 296 918 ATPDraw.chm
05.11.2019 13:24      258 ATPDraw.ini
05.11.2019 14:42    <DIR>          ATP
05.11.2019 14:42    <DIR>          Projects
05.11.2019 14:42    <DIR>          GRP
05.11.2019 14:42    <DIR>          BCT
05.11.2019 14:42    <DIR>          HLP
05.11.2019 14:42    <DIR>          LCC
05.11.2019 14:42    <DIR>          MOD
05.11.2019 14:42    <DIR>          UPS
05.11.2019 14:42    <DIR>          WEB
23.10.2018 14:43      499 068 Readme.pdf
05.11.2019 14:42      10 465 unins000.dat
05.11.2019 14:41      730 850 unins000.exe
7 File(s)           13 061 544 bytes

```

The files `unins000.dat` and `unins000.exe` are created by *Inno Setup* for uninstall purposes.

## 2.4 Files and sub-folders in the ATPDraw system folder

To use ATPDraw, three files are required: `ATPDraw.exe`, `ATPDraw.scl` (standard component library), and `ATPDraw.chm` (help file). Besides, there are a few required sub-circuits in the /GRP folder. Otherwise, the example files under /Projects are recommended starting points. The user can also create his own library components (user specified or models) and include files.

**Project file:** When the user saves a circuit, the work is stored in the project file (\*.acp = atpdraw circuit project). This file contains the circuit with all data and graphical representation. The project file is compressed by a public domain Pkzip 2.0 routine and can in fact be opened with any version of WinZip. A few project files are also installed under the /GRP folder.

**Support file:** All components inherit their properties from a support file. This file describes the type of component, the nodes (phases, position, identity) and data (default value, limits, parameter flag, number of digits, identity), the default icon (bitmap or vector) and a help text. The support files for standard components are zipped together in the file ATPDraw.scl (standard component library) and this file is required together with the project file to open and run a project. The support files can be edited inside ATPDraw in the *Library* menu. The default icon can also be modified by using the built-in icon editors. New user specified objects are created by specifying new support files.

**ATP file:** This file is produced by ATPDraw and used as input to ATP simulation. The .atp files with all \$Include files are written to the Result Directory with default location is specified as the \ATP sub-directory in the *ATP Connection Wizard*. The Result Directory can be changed via the -button in the toolbar or via *ATP/Sup-process/Make ATP file*. The ATP-file can be edited with any text-processors, including ATPDraw's own *Text Editor* (*Atp/Edit ATP file (F4)*). It is advised, however only for experts to modify this file manually.

**Include files:** User Specified Objects and Line&Cables components are described in a punch library file (.pch, .lib). This text file has a pre-defined format (as specified for the Data Base Module of ATP) and contains a header describing the positions of the parameters, further the ATP cards and finally a trailer containing the specification of the parameters. The library file is included in the ATP input file with \$Include. The include files are stored in memory and written to the Result Directory (same as ATP file) each time the ATP file is created. Some nonlinear components or saturable transformers might also have an include file for the nonlinear characteristic.

**Data files:** The user can export data for special components to a library for later use. Today this is a somewhat obsolete approach as it is easier and safer to simply create a library project (with backup) and copy/paste components from there. The data for a component in the circuit is stored internally in memory. The following file types are used:

- A line or cable is described by an .alc file (atpdraw line/cable). This binary file contains the line-, cable constants or cable parameter data. It should preferably be stored in the \LCC directory.
- A BCTRAN (Transformer) component is described in a .bct file. This binary file contains the input data required for the supporting routine BCTRAN of ATP-EMTP. It should preferably be stored in the \BCT directory.
- A Hybrid Transformer model is described by a .xfm file. This file contains the winding resistance, leakage inductance, capacitance, and core data. It should preferably be stored in the \BCT directory.
- A model is described in a model file (.mod). This text file starts with MODEL <name> and ends with ENDMODEL. The <name> must be equal to the model file name. It is recommended to store the models file in the \MOD sub-directory.

### 2.4.1 Organizing the files

When ATPDraw opens a project, no file is written to disk. All data are stored in memory. When the project is closed no disk files are deleted. Thus, as times goes by the number of files on disk grows. It is the user's responsibility to tidy up the directories. Remember: All required files are stored in the project and only the files you export/modify yourself outside a project need to be kept. Two house-keeping options are available under *Tools/Options/View/ATP*:

- Delete temp-files after simulation: Deletes all temporary BCTRAN/LCC files (.dat, .lis, .pch) and all temporary ATP files \*.bin when the simulation is finished. The files required to run ATP outside of ATPDraw (atp- and lib- files) are left on disk. In case of protected elements, the lib-files are immediately deleted and the atp-file is modified. During debugging a LCC or BCTRAN model, this button should be left unchecked.
- Delete result files on exit: Deletes the all temporary and result files (.atp, .lib, .lis, .pl4, .dat, .pch, .bin, .gnu) from ResultDir (the ATP folder as default) when the circuit is closed. All data is stored in the project files of ATPDraw anyway.

### 2.4.2 Configuring ATPDraw

The ATPDraw.ini file contains customizable program options. One such file for each user of the computer is stored in %APPDATA%\atpdraw\. The environmental variable APPDATA is system dependent but typical equal to 'c:\users\you\AppData\Roaming'. Note that Windows File Manager often hides the folder. Generally, default settings meet most of the user's requirements. When required, the .ini file can either be modified via *Tools / Options* menu, the *ATP Connection Wizard* in Fig. 2.2, or by using a text editor. A default ATPDraw.ini file is distributed with ATPDraw. This file is only used if there are no atpdraw.ini in the %APPDATA%/atpdraw location and can be used to configure the default interface.

### 2.4.3 ATPDraw command line options

Command lines are rarely used under Windows operating systems, nevertheless ATPDraw provides an option to load one or more project files at program start. In the example below, the project file my1st.acp and my2nd.acp will be loaded automatically and displayed in separate circuit windows.

```
C:\ATPDRAW>atpdraw c:\atpdraw\cir\my1st.acp c:\cir\my2nd.acp
```

In MS-Windows environment you can use this property to create a shortcut on the desktop for the ATPDraw project files. For instance, click with the right mouse button on an empty space of the desktop and select *New / Shortcut*, then browse and select ATPDraw.exe. Click right on the just created icon and select *Properties*. Specify the 'Target:' properties of the new shortcut as the full path of the program including the project file name (e.g. **c:\atpdraw\atpdraw.exe mycir.acp**), and the 'Start in:' parameter as the project file directory (e.g. **c:\atpdraw\project**).

### 2.4.4 Drag and drop project files

ATPDraw accepts project files dragged from the Windows File Manager. Dropping the project file (.acp) on the header, main menu or background causes the file to be opened in a new circuit window. Dropping the file in an existing circuit window causes the file to be imported into that circuit. Other file types dragged into the circuit will be added as zipped attachments.

## 2.5 Interfacing ATPDraw with other programs of the ATP-EMTP package

To configure ATPDraw and connect it with the desired solver (TPBIG.EXE) and plotter, use the *ATP/Setup ATP connection (F10)*, also called the *ATP Connection Wizard* shown in Fig. 2.2. In six steps the solver, environment variables, LIS-fil control, solver execution, result directory and plotting program is selected. Fig. 2.2 shows the recommended settings. Initially the “Execute solver in hidden mode” can be unchecked (step 4) and “Printout to screen” or “Capture screen output” checked (step 3) in order to identify possible configuration errors. If “Execute solver in hidden mode” is checked there is no DOS window popping up stealing focus, and the computer can be used for work while the simulation runs (also set NODISK=1 in graphix.aux to prevent JMarti line model diagnostics). #Simulations per core is used in Multiple Runs with the *Internal Parser* (systematic parameter variations or optimization). In this case, ATP is executed in parallel threads and folders. A low number will reduce the chances of file conflicts but also slow down the execution process. The results from the first run goes into the Result folder, and subsequent runs into \1\, \2\ etc sub-folders. A log-file (same name as ATP-file) containing information about the parameters is written to the Result folder.

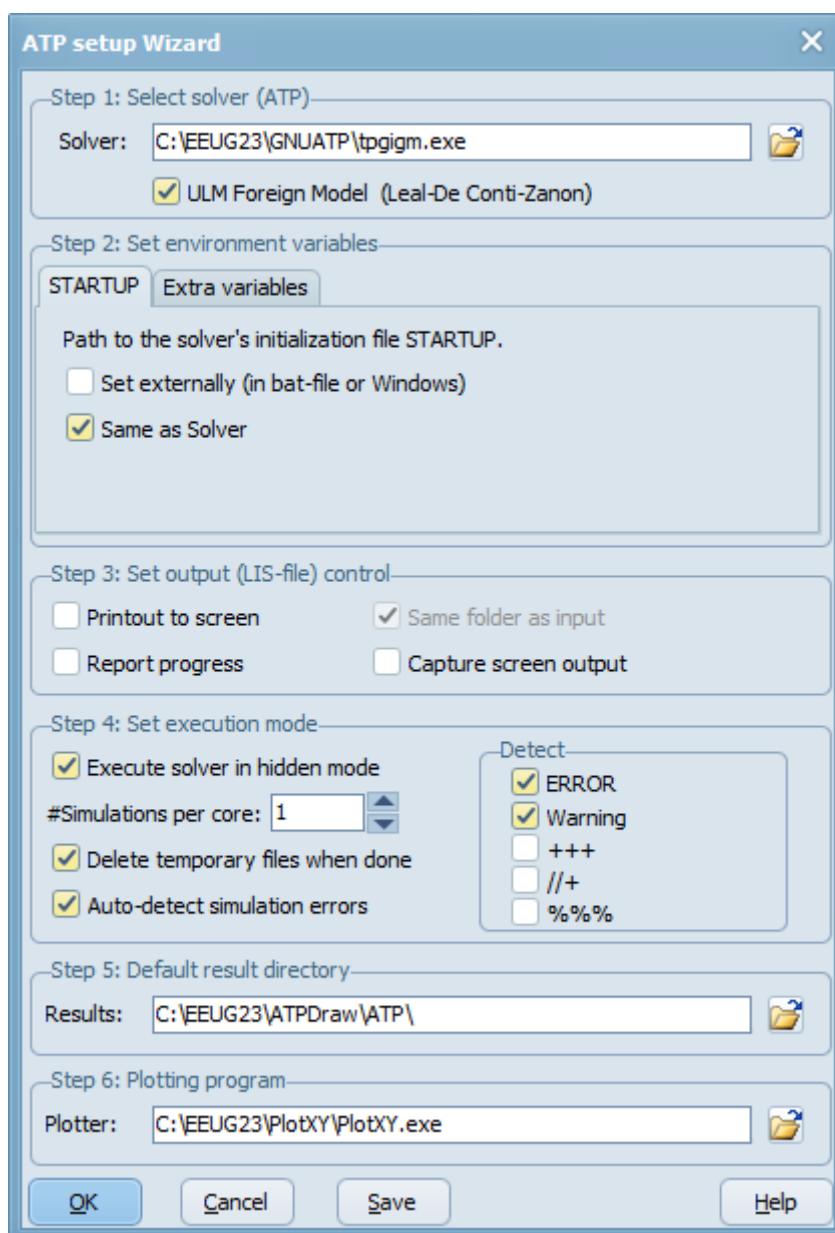


Fig. 2.2 ATP Connection Wizard

The ATP-EMTP simulation package consists of various separate programs which are communicating with each other via disk files: i.e. the output of pre-processors are used as input for the main program **TPBIG.EXE**, while the product of the simulation can be used as input for plotting programs. The main program itself is often used as pre-processor (e.g. for LINE CONSTANTS, CABLE CONSTANTS, BCTRAN or DATA BASE MODULE runs), and the punch-file results in that cases can be re-used as input in a subsequent run via \$Include, all handled directly by ATPDraw.

The *Edit Commands...* feature of ATPDraw supports to extend the command set under the **ATP** menu by integrating optional user commands, such as *Run ATP (file)* / *Run PlotXY* / *Run TPPPlot* / *Run PCPlot* / *Run ATP\_Analyzer* / *Run ACC* / *Run PL42mat*, etc. This option makes possible to use the ATPDraw program as a graphical operating environment and execute the other ATP programs in a user-friendly way as shown in Fig. 2.3.

The XML output from ATPDraw (alternative to native project file binary format .acp) can be used to exchange or modify project content more easily.

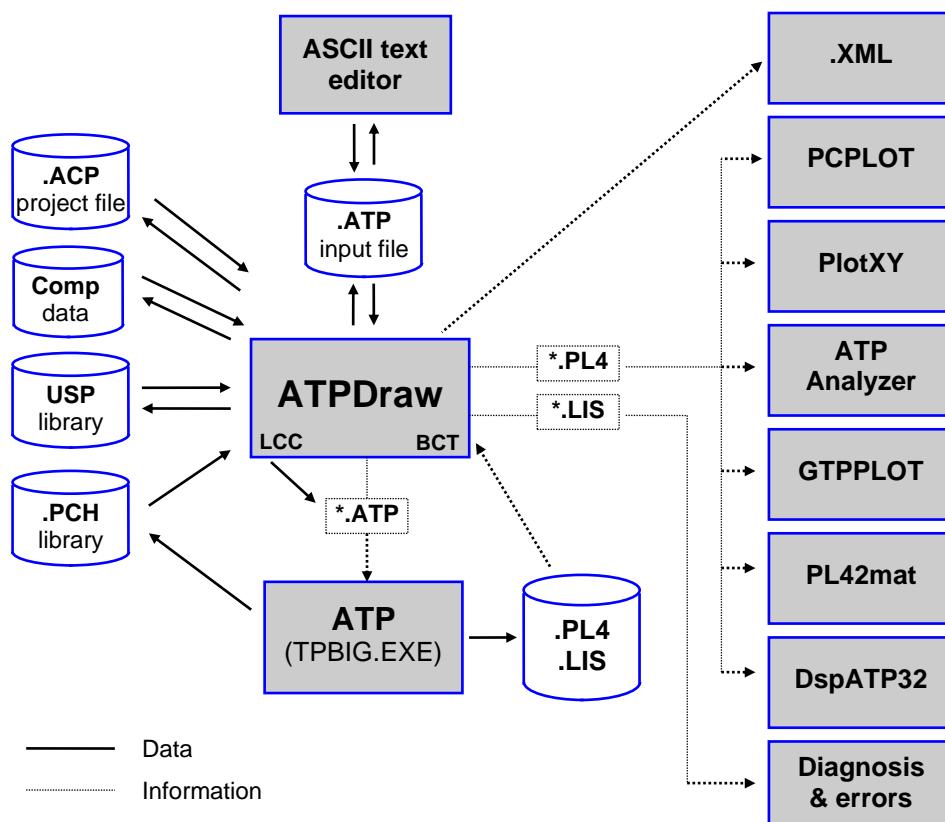


Fig. 2.3 - Interaction between ATPDraw and the other ATP programs

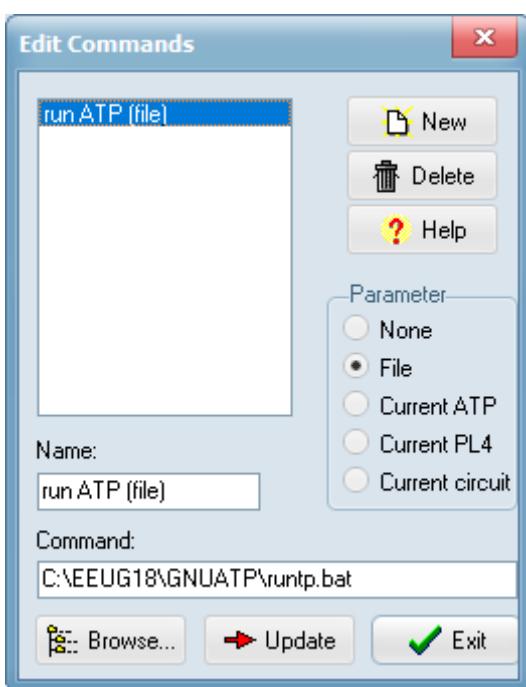


Fig. 2.4A - The Edit Commands dialog box.

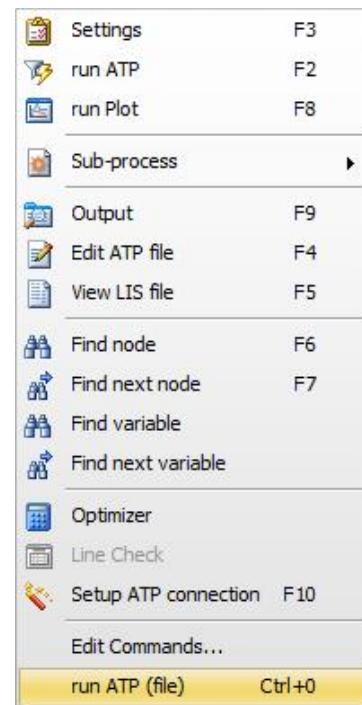


Fig. 2.4B - User specified commands.

In the *Edit Commands* dialog box of Fig. 2.4A the user can specify the name of a .bat or an .exe file and the name of a file, which then will be sent as parameter (e.g. ATP.bat <current .atp file> or PlotXY.exe <current .pl4 file>) when ATPDraw executes these external programs. The *Name* field specifies the name of the command, while the *Command* and *Parameter* fields specify the name of the file to be executed and the optional parameter. Selecting *Current ATP* radio button, the full name of the ATPDraw project in the current circuit window with extension .atp will be sent as parameter. When selecting the *File* button, the ATPDraw performs a file open dialog box before executing the command, where the user can select a file, which is then will be passed as parameter. The commands are inserted in the ATP menu dynamically, when the user activates the *Update* button as shown above.

## 2.6 Installing ATPDraw on servers with limited writing privileges

On servers, the users typically do not have writing privileges to the folders where ATPDraw is installed. This is particularly crucial for the Result Directory (default /ATP). The user can run the *ATP Connection Wizard* as shown in Fig. 2.2 and configure the setup manually, but it is also possible to edit the default ATPDraw.ini file to avoid any user interaction in the setup.

```
[Preferences]
ATPCommand=C:\EMTP\solver\tpgig.exe
PlotCommand=C:\EMTP\plotter\PlotXY.exe
runATPhidden=On
ATPDIR=C:\EMTP\solver\
ATPDirSameAsSolver=On
[Directories]
ATP=M:\ATPDraw\results\
Projects=M:\ATPDraw\projects\
[ATPDraw]
SaveOnExit=On
```

The ATPCommand and PlotCommand should preferable be the executables of the solver and plotter, respectively. ATPDIR points to the location of ATP's STARTUP file and is normally the same location as the solver itself. ATPDraw will create environment variables %ATPDIR% and %GNUMDIR% accordingly. A very important point is the ATP directory setting. This is also called the *Result Directory* and is where all the outputs from both ATPDraw and ATP go. This must be a location where the user has writing privileges. It is also advisable to avoid any blanks in the name of this directory. ATPDraw will prompt the user and create the ATP directory if it does not exist. Projects is optional and the default directory for *File/Open* and *Save* commands. It could point to a location with project files of interest. If not specified it will be set to \Projects. The user typically does not have writing privileges here and must choose *Save As* and a different folder. The SaveOnExit setting is also optional. If On, the ATPDraw.ini under %APPDATA%/atpdraw is created/updated when the user exits ATPDraw. This file is individual for each user and will contain all settings to be used the next time ATPDraw is run by the same user.

## 2.7 How to get help?

ATPDraw offers a standard Windows help file system. This file provides help on all windows and menus in ATPDraw and assists in building up a circuit. Several links between help pages and a relatively large index register for searching text or phrases are also available. A *Help* button is attached to all circuit objects, which shows a brief overview of the meaning of each parameter. Modification and extension of these help files with users' own remarks are also possible using the built in *Help Editor* in the *Tools* menu.

### 2.7.1 Help via the ATPDraw web-page forum

The ATPDraw Web page is maintained at address <https://www.atpdraw.net>

Users can register at the web-page (must pass the EMTP Quiz) and get access to the discussion forums (and cases); Beginner, Discussion, Bug report, Suggestions, Development. The discussion forums are thread-based, and upload of projects is allowed.

### 2.7.2 Help via the ATP-EMTP-L mailing list

The list server is an E-mail remailer program, which rebroadcasts incoming messages to all subscribers to the list. The European EMTP-ATP Users Group Association in cooperation with the German Research Network (DFN) and the University of Applied Sciences of Osnabrück, Germany operates a free electronic mailing list using address [atp-emtp-l@listserv.dfn.de](mailto:atp-emtp-l@listserv.dfn.de). This LISTSERV mailing list is for ATP-related announcements, questions, answers, etc. The ATP-EMTP-L list is *moderated* and only licensed ATP users are entitled to subscribe by means of the authorized persons of the regional ATP-EMTP user groups, who checks first the license status of the applicant, then send a subscription request to the list operator. To learn more about the subscription procedure and the operation rules of this very active mailing list, please visit the [www.eeug.org](http://www.eeug.org) web site.

After your name has been added to the list, you can post messages. To do this, you simply send e-mail to [atp-emtp-l@listserv.dfn.de](mailto:atp-emtp-l@listserv.dfn.de). Your message then will be submitted to moderators, who decide whether or not to accept it. The task of moderators is maintenance of the quality of communication and discussion. The language of communication is English. Messages written in any other language are not accepted. The author of each submission must be clearly identified. This includes name, organizational affiliation, and location. Attachments, especially encoded files, are not allowed. They can be forwarded later to interested persons by private e-mail. Any subscriber who sends a message to this mailing list gives up his right to confidentiality. This is

regardless of the message's possible declaration in auto-attached legal disclaimers, which are removed by moderators. Subscribers of the ATP-EMTP-L mailing list must fulfill the ATP license requirements. Specifically, they are forbidden to disclose to non-licensed persons ATP information that is received from this mail service.

### 2.7.3 Help from the author of ATPDraw

The author of the program is also available for serious questions from ATPDraw users, preferably via the ATPDraw web-page.

Address: Prof. Hans Kr. Høidalen  
 Norwegian University of Science and Technology, <https://www.ntnu.no>  
 Dept. Electric Power Engineering  
 7491 Trondheim - NORWAY  
 E-mail: [Hans.Hoidalen@ntnu.no](mailto:Hans.Hoidalen@ntnu.no)

## 2.8 Available components in ATPDraw

At the time of writing of this manual ATPDraw's standard component library contains 317 component support files. These 317 files support more than 170 of ATP's components, i.e. many components have several versions in ATPDraw for historical reasons.

### Standard components

#### *Linear branches:*

- Resistor, Inductor, Capacitor, RLC, PQU (multi-phase)
- Kizilcay F-dep.
- RLC 3-phase, symmetric and non-symmetric
- Inductor and capacitor with initial condition

#### *Non-linear branches:*

- Nonlinear R and L components (multi-phase)
- Current dependent resistor, type 99 (multi-phase)
- Type-93, 96 and 98 nonlinear inductors including initial flux linkage conditions
- Time dependent resistor, type 97 (multi-phase)
- MOV type 92 exponential resistor (multi-phase)
- TACS controlled resistor, inductor and capacitor (multi-phase)

#### *Line models:*

- Lumped, PI-equivalents (type 1, 2...) and RL coupled components (type 51, 52...)
- RL and PI symmetric, sequence input. 3 and 6-phase
- Distributed lines of constant parameters, Transposed (Clarke), untransposed (KCLee)
- Frequency dependent line models (Semlyen, JMarti, Noda, ULM (no steady-state))

#### *Switches:*

- Time controlled (multi-phase)
- Voltage controlled
- Diode, thyristor, triac (type 11 switches) (multi-phase)
- Simple TACS controlled switch of type 13 (multi-phase)
- Measuring switches (multi-phase)
- Statistic and systematic switches, independent and master-slave
- Nonlinear diode

#### *Sources:*

- Sawtooth and pulse train, type 10
- DC, type 11
- Ramp, type 12

Two-slope ramp, type 13  
 AC source. 1 and 3 phase, type 14  
 Double-exponential surge source, type 15, with fitting option  
 Heidler-type source, type 15, with fitting option  
 Standler-type source, type 15, with fitting option  
 CIGRÉ-type source, type 15  
 TACS source, type 60  
 AC source with TACS modulation (multiplication)  
 Empirical type 1 source with interpolation options  
 Ungrounded DC source, type 11+18  
 Ungrounded AC source, type 14+18  
 Trapped charge (disconnected at time zero)

#### *Machines:*

Synchronous machine type 59 (park) and 58 (phase) with TACS controls  
 Universal machines. Universal machines (type 1, 3, 4, 6, and 8)  
 Windsyn (embedded universal machine type 1 and 3 with manufacturer data)

#### *Transformers:*

Single-phase and 3-phase ideal transformer. Type 18 source  
 Single-phase saturable transformer  
 3-phase, 2- or 3 winding saturable transformer (Auto, Delta, Wye, and ZigZag)  
 BCTRAN. 1-3 phases, 2-3 windings. Auto-transformers, Y-, and D- connections  
 Hybrid Transformer (XFMR) with topological core; triplex, 3 or 5-legged, shell form. 3-phases. 2-4 windings. Auto, Y, D and ZigZag coupled windings.

## MODELS

Input/output and Data variables of MODELS code are recognized automatically  
 Corresponding support file for the model is automatically created  
 Type 94 (Thevenin, Norton, Iterative) objects are supported  
 WriteMaxMin cost function, WriteMonteCarlo

## TACS

Sources: Circuit variable, MODELS variable, Constant, DC, AC, PULSE, RAMP, ramped step and PWM 3-phase source  
 Transfer functions: General Laplace transfer function with or without limits, Integral, Derivative, first order Low and High Pass transfer functions  
 TACS devices (50-66).  
 Initial condition for TACS objects (type-77)  
 Fortran statements: Parameterized, General, Math, Trigonom. or Logical functions/operators

## User specified objects

Library: Users can create new objects using Data Base Modularization and \$Include  
 Additional: Insert additional CARD

## Power system tools

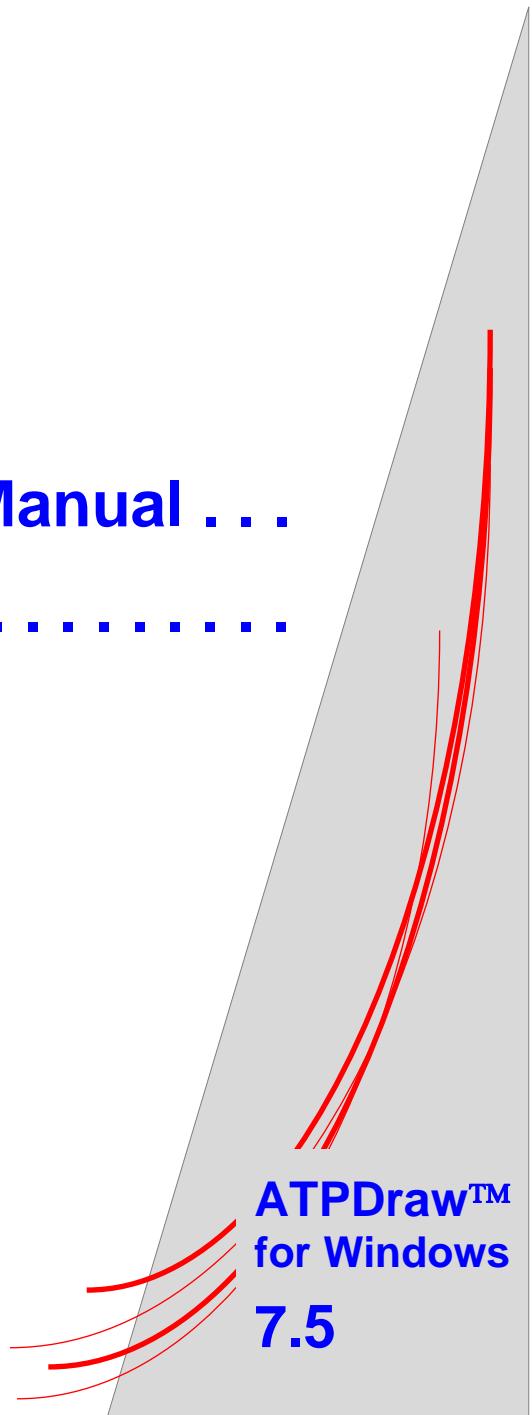
Various higher order or MODELS components for power system 3-phase studies; LINE3, BUS3, LOADPQ, Phasor, Transforms, RX and power calculators, filters, protective relays etc.

## Steady-state components

Harmonic sources for Harmonic Frequency Scan studies  
 Single and 3-phase frequency dependent loads in CIGRÉ format  
 Single phase RLC element with frequency dependent parameters  
 Load flow components

### **3. Introductory Manual . . .**

.....





This part of the user's manual gives the basic information on how to get started with ATPDraw. The Introductory Manual starts with the explanation of how to operate windows and mouse in ATPDraw. The manual shows how to build a circuit step by step, starting from scratch. Then special considerations concerning three phase circuits are outlined.

### 3.1 Operating windows

ATPDraw has a standard Windows user interface. This chapter explains some of the basic functionalities of the *Main menu* and the *Component selection menu* of the *Main window*.

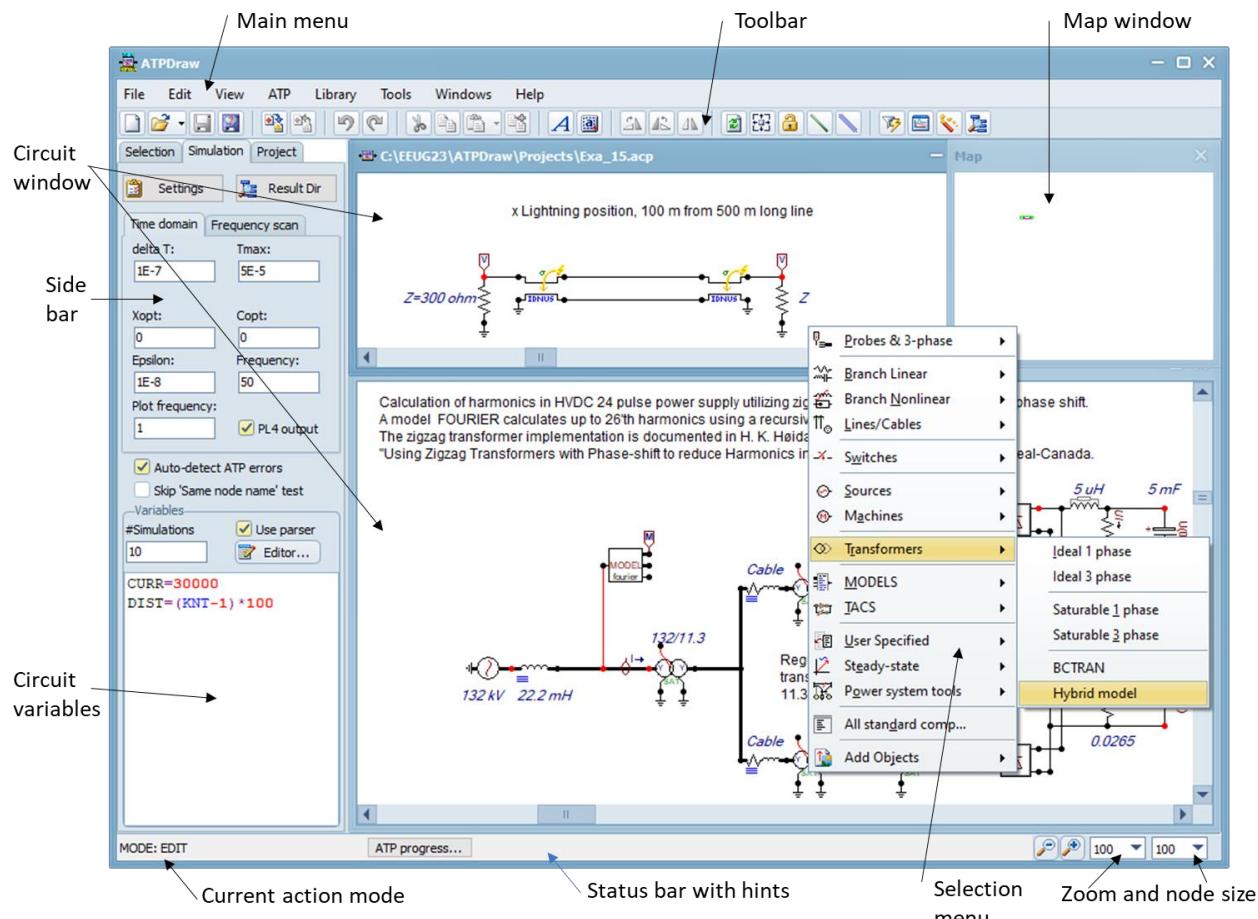


Fig. 3.1 - The Main window and the floating Component selection menu.

The *Component selection menu* is hidden and appears immediately when you click the right mouse in the open area of the *Circuit window*. Components can also be selected from the Sidebar/Selection which gives a full tree view of all components.

Fig. 3.1 shows the main window of ATPDraw containing two open circuit windows. ATPDraw supports multiple documents and offers the user to work on several circuits simultaneously along with the facility to copy information between the circuits. The size of the circuit window is much larger than the actual screen, as is indicated by the scroll bars of each circuit window. The *Main window* consists of the following parts:

#### *Header + Frame:*

As a standard Windows element, it contains the system menu on the left side, a header text and minimize, maximize, exit buttons on the right side. The main window is resizable.

**System menu:** Contains possible window actions: Close, Resize, Restore, Move, Minimize,

Maximize or Resize and Next. The last one exists only if multiple circuit windows are open.

**Header text:** The header text is the program name in case of the main window and the current circuit file name in case of the circuit window(s). To move a window, click in the header text field, hold down and drag.

**Minimize button:** A click on this button will iconize the main window.

**Maximize button:** A click on this button will maximize the window. The maximize button will then be replaced with a resize button. One more click on this button will bring the window back to its previous size.

**Corners:** Click on the corner, hold down and drag to resize the window.

#### *Main menu:*

The main menu provides access to all the functions offered by ATPDraw. The menu items are explained in detail in the Reference part of this Manual:

**File:** Load and save circuit files, start a new one, import/export circuit files, create postscript and metafile(bitmap files, print the current circuit and exit.

**Edit:** Circuit editing: copy/paste/delete/duplicate/flip/rotate, select, move label, copy graphics to clipboard and undo/redo etc.

**View:** Tool bar, status bar and comment line on/off, zoom, refresh and view options.

**ATP:** Run ATP, make and edit ATP-file, view the LIS-file, make node names, ATP-file settings (miscellaneous, file format, file sorting etc.), assign data to variables. Find Node and Line Check. Output Manager lists all output requests.

**Library:** Edit standard support files (default values, min/max limits, icon and help file), create new files for MODELS and User Specified Objects.

Synchronize the present circuit's icons or standard data from `atpdraw.scl`.

**Tools:** Icon editor, help file editor, text editor, setting of various program options.

**Window:** Arrange the circuit windows and show/hide the Map window.

**Help:** About box and Windows help file system.

#### *Circuit window:*

The circuit is built up in this window. The circuit window is the container of circuit objects. From the *File menu* you can load circuit objects from disk or simply create an empty window to start building a new circuit. Circuit objects include standard ATP components, user specified elements, MODELS and TACS components, connections and relations. To move around in the circuit, you can use the window scrollbars, or drag the view rectangle of the *Map window* to another position.

#### *Circuit objects:*

A circuit typically consists of the objects Components and Connections. These two classes take part in the node naming process and eventually in the ATP-file sent to the solver. In addition, comes objects used for information only. These are; Texts, Shapes, Pictures and Files (with drag&drop support). A special Component is the Group which contains a list of sub-objects.

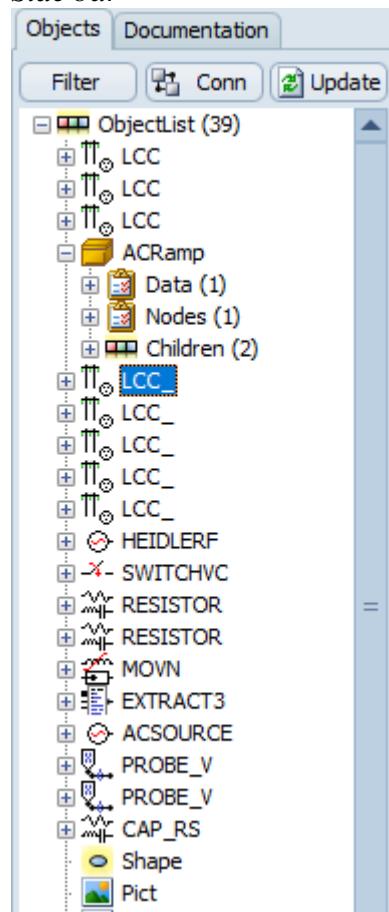
#### *MAP window:*

This window gives a bird's eye view of the entire circuit. The default size of a circuit window is 10000x10000 pixels (screen points); much larger than your screen would normally support. Consequently, the *Circuit window* displays only a small portion of the circuit. The actual circuit window is represented by a rectangle in the *Map window*.

Press and hold down the left mouse button in the map rectangle to move around in the map. When you release the mouse button, the circuit window displays the part of the circuit defined by the new rectangle size and position. The map window is a stay-on-top window, meaning that it will

always be displayed on the top of other windows. You can show or hide the map selecting the *Map Window* option in the *Window* menu, or pressing *Ctrl+M* character,

### Side bar



This bar to the left has three pages. The default Simulation page contains frequent simulation settings and variables besides some useful tools. The Selection page contains a tree structure for insertion of all components. The Project page contains some project properties and a tree structure with all objects in the active circuit. The Object Tree, Fig. 3.2 in the Sidebar contains options to inspect (with filter), navigate, arrange, find and open objects.

Components are marked with a symbol indicating branch, switch, source, transformer, machine, tacs, models. The Group component is marked with a box symbol with a list of Children (group content).

Connections, Texts, Shapes, Pictures, Files, and Plots are other circuit objects.

Left click on the object to mark and center it in the circuit window, right click to open its dialog, click hold and drag to rearrange it. Objects first in the list are prioritized in mouse clicks and ATP-file generation. *Conn* sends all Connections to the back and mimics <v7.0 behavior. Consider also *Edit/Arrange*.

The object tree is not automatically updated with circuit changes so click on *Update* to see the present situation.

Fig. 3.2 Object tree inspector

### Status bar - Action mode field:

The current action mode of the active circuit window is displayed in the status bar at the bottom of the main window, when the *Status Bar* option is activated in the *View* menu. ATPDraw can be in various action modes. The normal mode of operation is *MODE : EDIT*, in which new objects are selected and data are given to objects. ATPDraw's possible action modes are:

<b>EDIT</b>	The normal mode.
<b>EDIT TEXT</b>	Indicates that text editing is preferred. Hold down the Alt key to enter this mode of operation or select <i>Edit Text</i> from the <i>Edit</i> menu. Click left in an empty space to add a new text. Click the left mouse button on an existing text (circuit text, label, node name) to edit it directly on screen. Click left, hold down and drag to move it to a new position. If the text is overlapped by a component icon, this mode of operation is required to access the text.
<b>DRAW</b>	Mode when adding Shapes to the circuit ( <i>LINE</i> , <i>RECTANGLE</i> , <i>ELLIPSE</i> , <i>ARROW</i> ). To cancel drawing relation, click the right mouse button or press the <i>Esc</i> key.
<b>COMPRESS</b>	Mode when objects are selected and <i>Edit/Compress</i> is clicked. In this mode only the selected objects are shown with the Compress dialog on top.

***Status bar - Modified and Hints field:***

The middle field of the status bar is used to display the *Modified* state of the active circuit. As soon as you alter the circuit (moving a label, deleting a connection, inserting a new component, etc.), the text *Modified* appears, indicating that the circuit should be saved before exit. The field will be empty when you save the circuit or undo all modifications. The rightmost field of the status bar displays the menu option hints.

***Status bar – atpdraw.net field:***

Shows if the user has logged in to atpdraw.net from *Web/Log in*. In order to log in the user must register first at atpdraw.net this requires passing the EMTP Quiz. Logged in users have access to the database at atpdraw.net and can download examples and contribute to the forum with upload.

***Status bar - Zoom and node size:***

In these menus you can type in zoom and node size in [%] or select predefined values in the popup box.

### **3.2 Operating the mouse**

This chapter contains a summary of the various actions taken dependent on mouse operations. The left mouse button is generally used for selecting objects or connecting nodes; the right mouse button is used for specification of object or node properties.

**Left simple click:**

On object: Selects the object.

If the *Shift* key is pressed, the object is added to the current selection group.

On connection: Draw a new connection with the same properties.

On component node: Begins to draw a connection.

Move the mouse to the end node, left click to place, right to cancel.

On text, labels and node names: Edit the text directly on screen. Press *Alt* to favor the text selection compared to other objects.

In open area of the circuit window: Unselects object(s).

**Right simple click:**

In open area of the circuit window:

Shows the *Component selection menu* (after canceling any other drawing process).

On object node:

Shows the *Node input* window.

On unselected object:

Shows the *Object input* window.

On selected object(s):

Shows the circuit window *Shortcut menu*.

If *Shift* is pressed, rotates object(s).

**Left click and hold:**

On object: Moves the object or selected group of objects.

On connection: Select connection.

On node: Resizes connection (it is often necessary to select connection first).

In open area of the circuit window: Draws a rectangle for group selection.

Objects inside the rectangle are becoming member of the group when the mouse button is released.

On text, labels and node names: Move the text. Press *Alt* to favor the text selection compared to other objects.

#### Left double click:

On Component node:

Shows the *Node input* window

On selected or unselected single object:

Shows the *Object input* window.

On selected group of objects:

Shows the *Selection input dialog*

In open area of the circuit window:

Starts the group selection facility. Click left to create an enclosing polygon, click right to close. Objects inside the polygon become a group.

### 3.3 Edit operations

ATPDraw offers the most common edit operations like copy, paste, duplicate, rotate and delete. The edit options operate on a single object or on a group of objects. Objects must be selected before any edit operations can be performed. Selected objects can also be exported to a disk file and any circuit files can be imported into another circuit.

Tool	Shortcut key	Equivalent in menus
UNDO	Ctrl+Z	<i>Edit / Undo</i>
REDO	Ctrl+Y	<i>Edit / Redo</i>
Cut/Copy	Ctrl+X/Ctrl+C	<i>Edit / Cut/Copy</i>
Delete	DEL	<i>Edit / Delete</i>
Paste	Ctrl+V	<i>Edit / Paste</i>
Paste keep names	Ctrl+K	<i>Edit   Paste keep names</i>
Duplicate	Ctrl+D	<i>Edit / Duplicate</i>
Select/All	Ctrl+A	<i>Edit / Select All</i>
Select/Inside	Ctrl+I	<i>Edit / Select/ Inside</i> (or left double click in open space)
Select/Properties	Ctrl+P	<i>Edit / Select/ by Properties</i>
New/Select text	Ctrl+T	<i>Edit / Edit text</i>
Rotate clockwise	Ctrl+R	<i>Edit / Rotate R</i> (or right click)
Rotate left	Ctrl+L	<i>Edit / Rotate L</i>
Rubber Band	Ctrl+B	<i>Edit / Rubber Bands</i>
Draw LINE3	Ctrl+F3	<i>Edit   Draw LINE3</i>
Edit Group/Circuit	Ctrl+G/Ctrl+H	<i>Edit / Edit Group/Circuit</i> (one layer down or up)
Zoom In/Out	NUM + / -	<i>View / Zoom In / Out</i>
Refresh	Ctrl+Q	<i>View / Refresh</i> (redraw the circuit)

### 3.4 Overview of working with ATPDraw

After selecting a component in the *Component selection menu* (right-click open space in circuit window or Sidebar/Selection) the new circuit object appears in the middle of the circuit window enclosed by a lime-colored rectangle. Click on it with the left mouse button to move (right button to open the context menu), finally click in the open space to unselect and place the object.

To select and move an object, simply press and hold down the left mouse button on the object while moving the mouse. Release the button and click in an empty area to unselect and confirm its new position. The object is then moved to the nearest grid point (known as grid snapping). If two or more components overlap because of a move operation, you are given a warning message and

can choose to proceed or cancel the operation.

Selecting a group of objects for moving can be done in three ways: Holding down the *Shift* key while left clicking on an object. Pressing and holding down the left mouse button in an empty area enables the user to drag a rectangular outline around the objects he wants to select. And finally, double-clicking the left mouse button in an empty area enables the definition of a polygon-shaped region by repeatedly clicking the left mouse button in the circuit window. To close the region, click the right mouse button. Components with center point within the indicated region or rectangle are added to the selected objects group. Connections require both end points within the region to be selected. Select *Edit/Rubber Bands* to stretch connections with one end inside and one end outside. To move the selected group of objects, press and hold down the left mouse button inside the group while moving the mouse. Unselect and confirm the new position by clicking in an empty area. Any overlapping components will produce a warning. Selected objects or a group can be rotated by selecting *Edit/Rotate L/R* (*Ctrl+R* or *Ctrl+L*). Other object manipulation functions, such as undo/redo and clipboard options can be found in the *Edit* menu. Additionally, the most frequently used object manipulation functions can be accessed in the context menu with the right mouse button on a selected object or group of objects.

Components and component nodes can be opened for editing by a right-click (or left double-click) on an unselected component or node. Either the *Node data*, *Component* or *Probe* dialog box will appear, allowing the user to change component or node attributes and characteristics. The *Component* dialog box shown in Fig. 3.3 has the same layout for most circuit objects. In this window the user must specify the required component data. The number of DATA and NODES menu fields are the only difference between input windows for standard objects. The nonlinear branch components have a *Characteristic* page too, in addition to the normal *Attributes* page, where the nonlinear characteristics and some include file options can be specified. Some of the advanced components like LCC, BCTRAN, Hybrid Transformer have special dialog boxes for input.

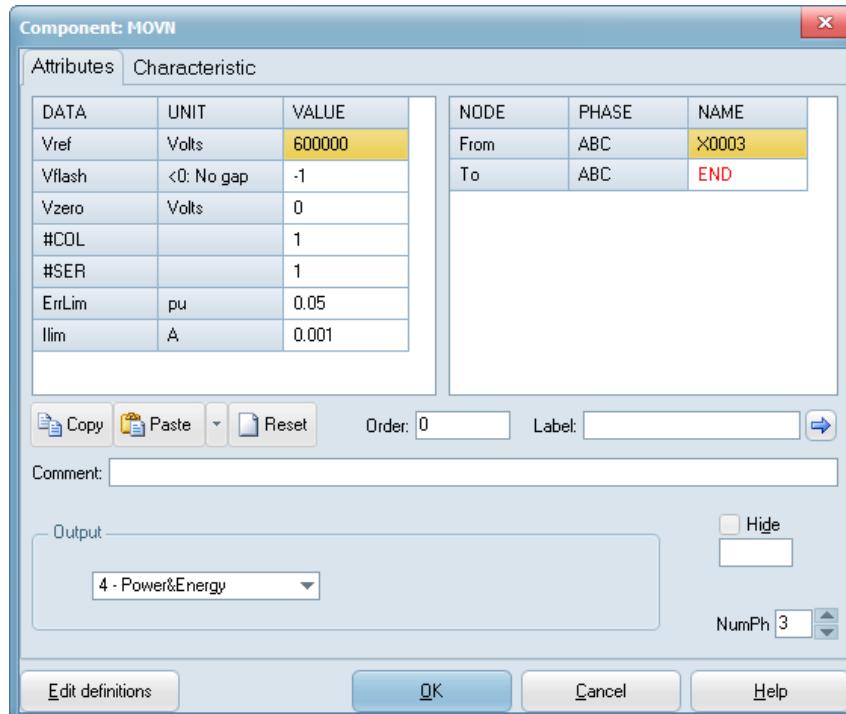


Fig. 3.3 – Component dialog box, attributes page.

The Component dialog box shown in Fig. 3.3 consists of a Data part and a Node part. In the Data part the user can specify values using '.' as the decimal symbol and 'e' or 'E' as exponent symbol.

Mathematical expressions are also supported and an input ‘`120E3*sqrt(2/3)`’ will be converted to a value when *OK* is clicked. If the value is illegal or outside the allowed range the user will be directed and forced to change the value. The range can be changed inside *Edit definitions*. A variable name can also be specified and given a global value in the *Sidebar* or under *ATP/Settings/Variables*. Unlimited number of characters are allowed from v7.4. Specifying a variable is only allowed if the *Internal Parser* is used or the *Param* property of the data in *Edit definitions* is set to unity. Warning messages will appear in case of illegal specifications and the user can modify the data. It is from v7.4 legal to combine variables and expressions like ‘`MyVar*1000`’. Data values in lime color are inherited from the parent (group) component and cannot be changed inside the child.

The *Copy/Paste* buttons allows copying the entire data set via the Windows clipboard. *Paste Use row number* simply paste row by row, while *Paste Use data name* will require the data names to be equal. *Reset* will apply the default values.

Node names (6 or 5 characters) can be specified in the right grid. Node names drawn in a red color are given a name by the user while black names are given by ATPDraw. If the user wants to change a node name the red names/nodes are preferred, otherwise name conflict warnings could appear. Node data are also given in the Node dialog box by clicking on the nodes. Multi-phase nodes can only take a 5 character name, and the phase sequence extension A..Z is added automatically. Node names in lime color are inherited from the parent (group) component and cannot be changed inside the child.

*Order* is optionally used for sorting (*ATP/Settings/Format*; sorting by order (low-high)), *Label* is a text string on screen with user-selectable rotation, and *Comment* is a line of text written to the ATP file in front of the component’s cards. *Hide* can be checked to make the Component grey and exclude it from the ATP-file. A variable can also be specified and if its value is positive the Component becomes hidden. A Component is also hidden if its parent (group) is hidden.

The *Output* panel varies somewhat between components but is usually used for branch output requests (select current, voltage, power or energy to be plotted). Electrical machines and Models/Tacs have a substantially extended panel.

In the lower left corner, there is the *Edit definitions* button. This gives access to all the local properties inherited from the template file, including the icon, local help, names of nodes and data, node positions, default values, param flags, range, and units.

Clicking on *Help* will display the help text for the component; first comes the global help obtained from the support files (ATPDraw.scl for standard components), next comes local help specific to this component, and finally comes global help from the /HLP directory.

Default component attributes are stored in template files. Access to create and customize template files is provided by the *Library* menu.

Components are connected if their nodes overlap or attached to the same Connection. To draw a Connection, click on a node with the left mouse button. A line is drawn between that node and the mouse cursor. Click the left mouse button again to place the Connection (clicking the right button cancels the operation). The gridsnap feature helps overlapping the nodes. If the Connection is drawn between nodes of different number of phases, the user must choose the actual phase to connect in the *Edit Connection* dialog. The default color coding (phase A=red, phase B=lime, phase C=blue) will visualize the connected phase. Connected nodes are given the same name by

the *run ATP* option in the ATP menu. Nodes can be attached along a Connection only if the connection is horizontal or vertical, but always at the Connection's end-points. A warning for node naming appears during the ATP-file creation if a Connection exists between nodes of different names, or if the same name has been given to unconnected nodes. Connections can be selected, moved and rotated as any other objects and are stretched when moving connected components if *Edit/Rubber bands* is checked. To resize a Connection, click on its end-point with the left mouse button, hold down and drag. If several Connections share the same node, the desired Connection to resize must be selected first. Selected Connection nodes are marked with squares at both ends of the selection rectangle. To avoid selecting Connections over Component nodes, consider *Edit/Arrange/Send Connections back*.

### 3.5 Your first circuit (*Exa\_1.acp*)

This chapter describes how to use ATPDraw step by step. As an example, composing the circuit file of a single-phase rectifier bridge (see Fig. 3.4) is presented. Reading this tutorial carefully, you will be proficient in the use of the most important ATPDraw functions, such as:

- How to select and assemble components?
- How to perform edit operations and give data to components?
- How to give node names, draw connections and specify grounding?
- How to create the ATP input file and perform the simulation?

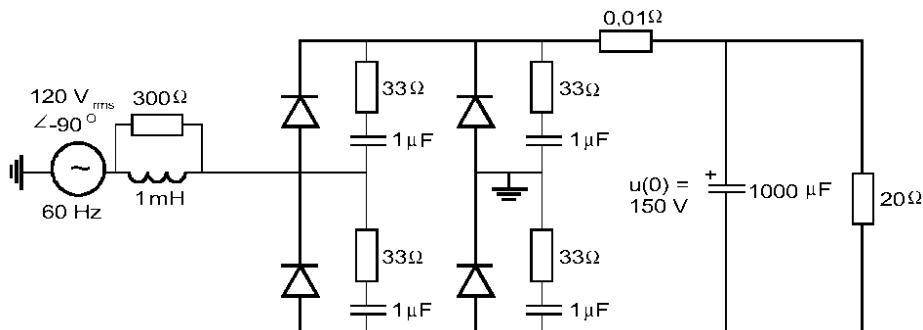


Fig. 3.4 – Single-phase rectifier bridge.

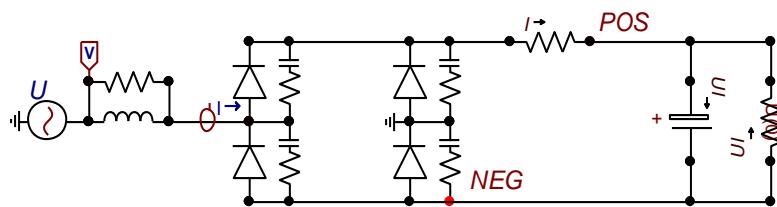


Fig. 3.5 – Your first circuit (*Exa\_1.acp*).

The circuit is a single-phase rectifier bridge, supplied by a  $120 \text{ V}_{\text{rms}}$ ,  $60 \text{ Hz}$  source. The source inductance is  $1 \text{ mH}$  in parallel with a damping resistor of  $300 \Omega$ . The snubber circuits across the rectifying diodes have a resistance of  $33 \Omega$  and a capacitance of  $1 \mu\text{F}$ . The smoothing capacitor is  $1000 \mu\text{F}$  and the load resistor is  $20 \Omega$ . The example has been taken from [2], exercise 1. The units given in Fig. 3.4 are based on settings of Xopt and Copt equal to zero as will be explained later. The circuit in Fig. 3.5 has been chosen since its construction involves the most commonly used edit operations.

### 3.5.1 Building the circuit

Most parts of the building process will be demonstrated in this chapter, along with the explanation of correcting possible drawing errors. The normal mode of operation is *MODE : EDIT*. You must always be in this mode to be able to select and specify data to objects. To return to EDIT from other modes, press *Esc*.

#### 3.5.1.1 Starting to create a new circuit

Selecting the *New* command in the *File menu* or pressing the new (empty) page symbol in the *Component Toolbar*, a new circuit window will be created.

#### 3.5.1.2 Source

First, an AC source is selected from the *Component selection menu*, which appears with a right mouse click on open area of the circuit window. Fig. 3.6 shows how to select a general AC (type 14) source under *Sources / AC source (1&3)*.

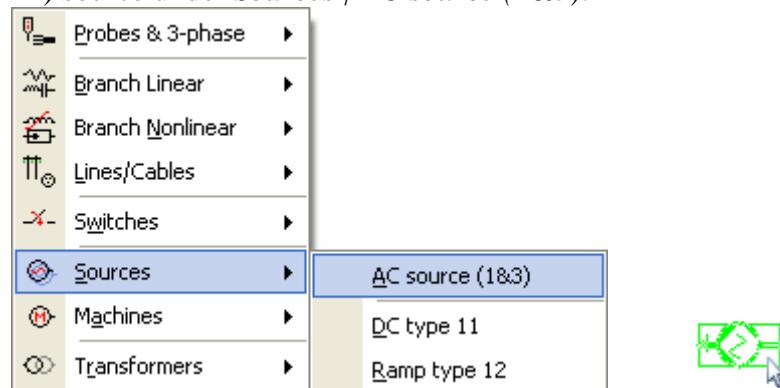


Fig. 3.6 - Selecting an AC source.

After you have clicked in the *AC source (1&3)* field, the selected source appears in the circuit window in lime color, enclosed by a rectangle. Click on it with the **left mouse button**, hold down and drag it to a desired position. Then click with the left mouse button in open space to place it. The AC object is redrawn in red color as an indication that no data have been given to the object. To give data to the AC source component, click on with the **right mouse button** (or left double click). You can give data to objects at any time during the building process. If you right click on the AC source icon, a window as shown in Fig. 3.7 appears. Click the radio button *Amplitude-RMS L-G* to specify the rms value 120 volts directly. ATPDraw will then multiply with  $\sqrt{2}$  internally (the *RMS L-L* option will also divide by  $\sqrt{3}$ ). To use a Variable (see p. 73) for the AmplitudeA value the *Peak L-G* (standard, no scaling) option is required. A negative value for StartA parameter means that the source is active during steady-state initialization.

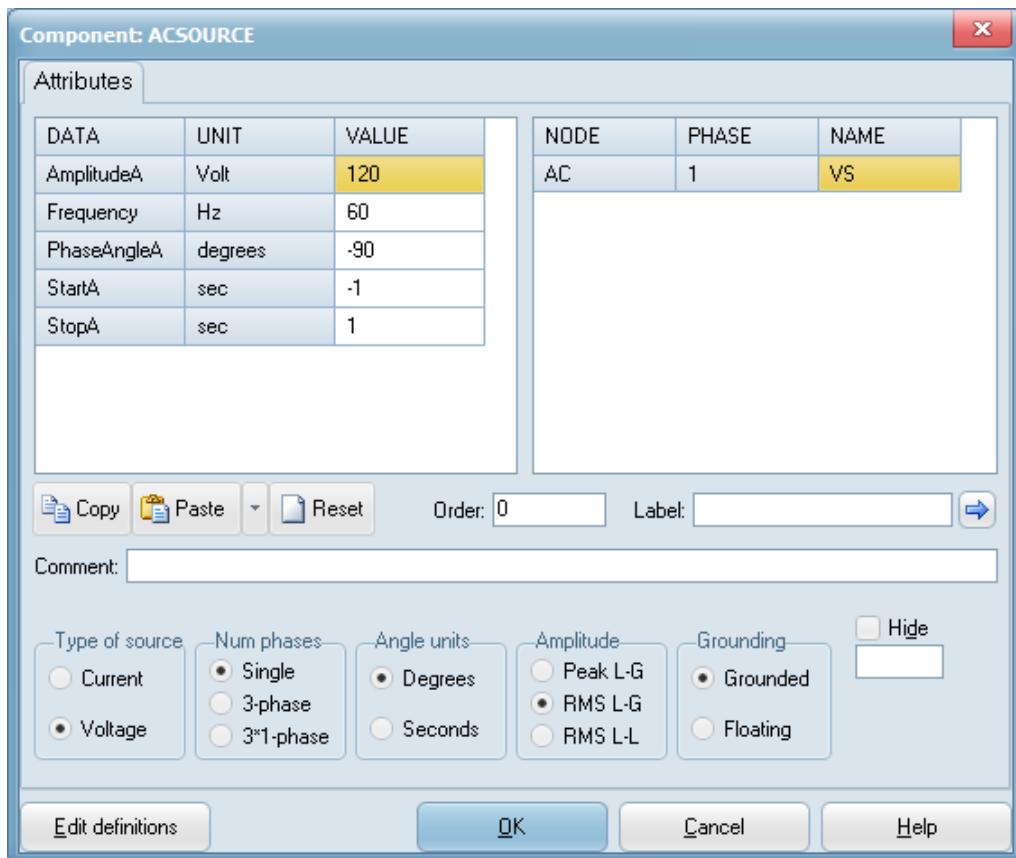


Fig. 3.7 - Component dialog box of the single-phase sinusoidal source.

Data values shown in Fig. 3.7 refer to the circuit parameters of Fig. 3.4. The name of the numerical fields is identical with that of used by the ATP Rule Book [3] for an AC source. This AC source has 5 input data and one node; AC (ACNEG and Internal nodes disappear for grounded voltage sources). Click on the *HELP* button to learn about the meaning of parameters.

The node names can also be specified in this window. Click *OK* to close the window and update the object values. Click on *Cancel* to just quit the window.

After you have given data to the AC source and closed the window (note how the object layout changes when you exit the window), proceed to the other objects. Next select the source inductance as shown in Fig. 3.8:

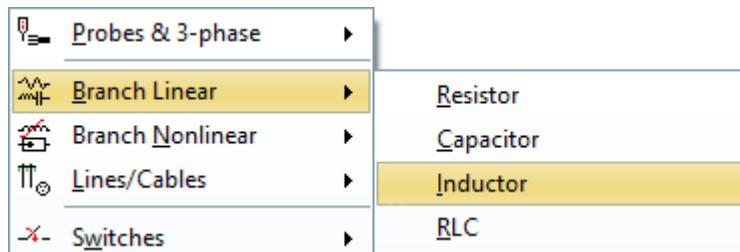


Fig. 3.8 - Selecting an inductor.

After you have clicked in the *Inductor* field, the selected inductor appears in the circuit window enclosed by a rectangle (an optional, parallel damping resistance is included). Click on it with the left mouse button, hold down and drag it to a position shown in Fig. 3.9:

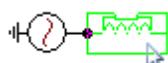
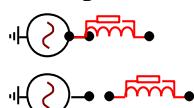


Fig. 3.9

Click on the white space with the left mouse button to place the inductor (the enclosing rectangle disappears). A grid snap facility helps you to place the inductor in the correct position. The component position is rounded to the nearest 10<sup>th</sup> pixel. (The included parallel resistor is shown in a gray style.)

The inductor in Fig. 3.9 should be placed so that the node of the inductor touches the source. Objects having overlapping node dots will automatically be connected.

The next figure shows two situations where the inductor has been misplaced and are disconnected. To correct the lower example, a connection could be drawn between the objects as will be explained later. In this example you are supposed to place the inductor so that its left node overlaps the AC source node. To move the inductor, follow the instructions below.

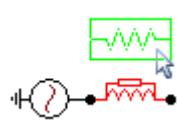


Click on the object with the left mouse button, hold down and drag it to the proper position, then click on white space. The grid snap feature will help you.

Fig. 3.10 – Not connected!

When you have placed the inductor, you can add the damping resistance (possibly directly included). After you have clicked in the *Resistor* field of the component selection menu, a resistor icon appears enclosed by a rectangle. Click on it with the left mouse button, hold down and drag it to a position shown in Fig. 3.11. Click in open space to place/unselect it.

This resistor is supposed to be parallel with the inductor and connections will be drawn later. The



resistor in Fig. 3.11 would also be recognized as in parallel with the inductor, if it had been placed in a position completely overlapping the inductor. This tricky way is not recommended however, since the readability of the drawing is strongly reduced (also warnings will be issued by the circuit compiler).

Fig. 3.11

We want to measure the source current flowing into the diode bridge. To be able to do so, you can add a measuring switch. A special multi-phase current probe is available for such measurements in ATPDraw. When using this object, you are requested to specify the number of phases and in which phases the current should be measured. Select the probe as shown in Fig. 3.12.

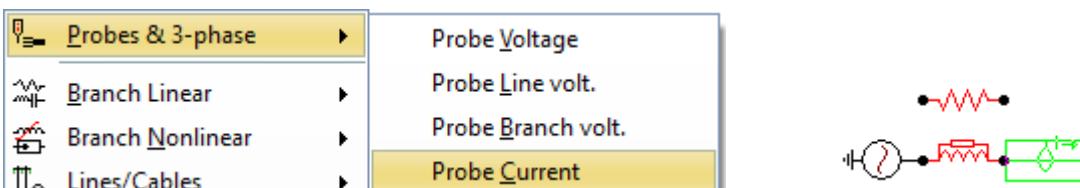


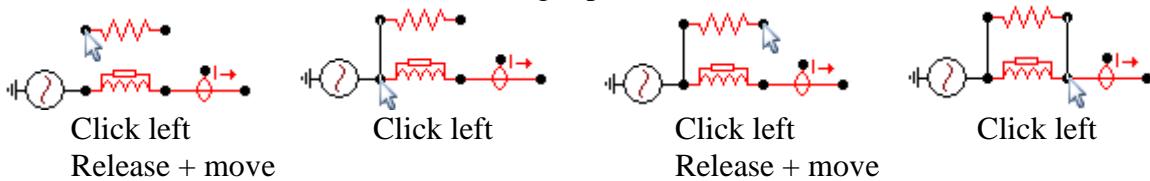
Fig. 3.12 - Selecting a current measuring probe.

After you have clicked in the *Probe Curr.* field, the selected probe appears in the circuit window enclosed by a rectangle. Click on it with the left mouse button, hold down and drag it to a position shown in the figure, then place it.

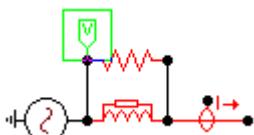
At this stage of the building process, it is time to draw some connections in the circuit diagram. To draw a connection, you just click the left mouse button on a node, release the button and move the mouse. The cursor style now changes to a pointing hand and a line is drawn between the starting position and the current mouse position. Click with the left mouse button again to place the connection or click with the right button to cancel the starting point.

Two Connections are required to connect the source inductance and the damping resistor in

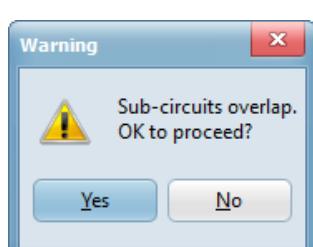
parallel as shown below. The Connection dialog (color, phase number) automatically appears for connections drawn between multi- and single-phase nodes, but not in this case.



The last object we want to introduce in the source part of the circuit is a voltage measuring probe, which results in an output request for the node voltage in the ATP input file. The voltage sensor can be selected via the *Probe & 3-phase / Probe Volt* in the component selection menu (see Fig. 3.12). The probe is drawn in the circuit window in marked and moveable mode. Use the left mouse button to drag and place the probe as shown on the figure to the left.



When you place an object by clicking on open area of the circuit window, you will sometimes receive a warning message as shown in Fig. 3.13. This message appears if a center of one of the permanent objects is inside the enclosing polygon of a marked object (or more general; a group of objects). This is to prevent unintentional object overlap if the left mouse button were pressed while moving the object.



If you click on *No*, the object is not placed but continues to be selected and you can move it further. Normally it is OK to click on *Yes*. If you change your mind later, the *Edit / UNDO* option provides an easy way to return to an earlier version of the circuit. If objects with the same icon completely overlap the visual clarity is violated (what you see is not what you get). A warning is thus issued during the compilation (MakeFile/run ATP).

Fig. 3.13 - Prevent object overlap.

Now, give data to the components placed so far. Click with the **right mouse button** on the resistor and inductor icon, respectively. The inductor has a built-in damping resistor option but turn this off by choosing Kp=0.

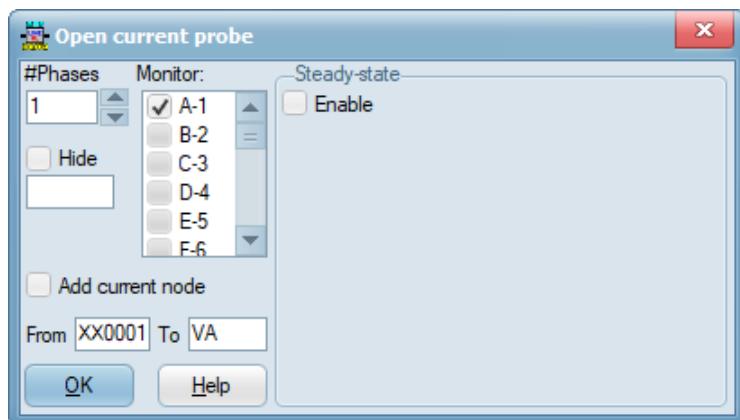


Fig. 3.14 - Open probe dialog box.

The probe objects have different input window than other objects. To open the voltage or current probe input window, click on its icon with the right mouse button. In this window, you can select the number of phases of the probe and which phases to monitor. In this single-phase example, default values (no. of phases=1, monitored phase=A) of both voltage and current probes should be selected, as shown in Fig. 3.14

### 3.5.1.3 Diode bridge

In this process, you will learn how to use some editing options like rotate, group, duplicate and paste. Since the diode bridge consists of four equal branches, you do not need to build all of them from scratch. First, you select a diode from the selection menu as shown in Fig. 3.15. After you

have clicked on *Diode (type 11)* the diode appears in the circuit window enclosed by a rectangle.

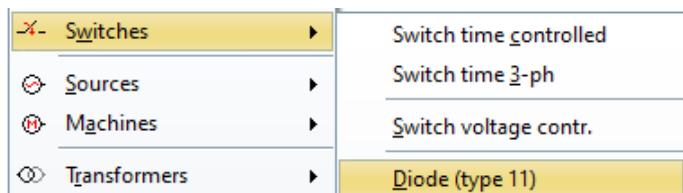


Fig. 3.15 - Selecting a diode.

The diode must be rotated so click the right mouse button or select *Edit* in the main menu and click on *Rotate L*. The diode is now rotated 90 deg. counter clock-wise. Click on the diode with the left mouse button, hold down and drag to the position shown in Fig. 3.16.



Click with the left mouse button on empty area to place the diode. Remember the grid snap facility and the overlap warning.

Fig. 3.16

Next, you select the snubber circuit across the diode. In this example the snubber circuit is a resistor and a capacitor in series. Select an RLC object from the component selection menu (Fig. 3.8).

Click on the selected RLC branch with the right mouse button to rotate, then click with the left button, hold down and drag the RLC branch to be in parallel with the diode. Click on the left mouse button to place.

The idea is further to copy the diode and the RLC branch, but before doing so, it is wise to give data to them (since the data are kept when copied). A simple click on the RLC or diode icon with the right mouse button activates the component dialog box to give data to objects.

Again, an explanation of the input parameters is given in a help file. Click the *HELP* button to see this help text. The numerical values of the diode are all zero, meaning that the diode is ideal and is open during the steady state. The RLC branch in Fig. 3.16 has been given a resistance of  $33 \Omega$  and a capacitance of  $1 \mu\text{F}$ . The icon then changes to a resistor in series with a capacitor.

You have now given data to the diode and the RLC branch and instead of repeating the drawing and data entering process four times, you can use the copy facility. First, you have to select a group of components. This can be done by selecting *Edit / Select/ Inside* field in the main menu or with a double click with the left mouse button on an empty space of the *Circuit window*. Then cursor style changes to a pointing hand and the action mode is *EDIT : GROUP*. The process is then to click with the left mouse button to create a corner in a fence and to click the right button to enclose the fence (polygon). All components having their center inside the fence are included in the group.

Alternative way of group selection is to draw a rectangle around the objects by a left mouse click and hold at the upper-left corner of the desired rectangle and moving thereafter to the lower-right corner. Objects inside the rectangle become a group when the mouse button is released.

You can follow the procedure shown in Fig. 3.17.



Fig. 3.17 - Drawing a polygon: First double click on white space, click the left mouse button at each corner of the polygon, then click the right button to enclose the polygon.

The group created in Fig. 3.17 can be copied/rotated etc. like a single object. Now we want to duplicate this group. Click on the main menu *Edit* field and choose *Duplicate* or press the *Ctrl+D* shortcut key. The selected group is copied to the clipboard and pasted in the same operation. The old group is redrawn in normal mode and the copy is drawn in the top of the original.

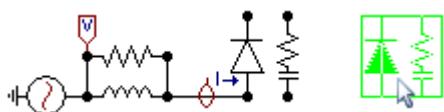


Fig. 3.18 - Move a group.

The enclosing polygon is now a rectangle. The pasted group is moveable, so you can click on it with the **left mouse button**, hold down and drag to a desired position. Click the left mouse button on open space to put the group in the position shown in Fig. 3.18.

If you misplaced the group, you can reselect it or use the *Undo* facility found in the *Edit* main menu field.

You can now paste a second copy of the diode/RLC group into the circuit. Since the duplicate facility has already copied the group to the clipboard, you can just select the *Paste* option from the *Edit* menu by using the mouse or pressing *Ctrl+V*, or selecting the *Paste* icon from the *Toolbar*. The pasted group is drawn on top of the original one enclosed by a rectangle. Click on this group with the left mouse button, hold down and drag it to a position shown in Fig. 3.19.

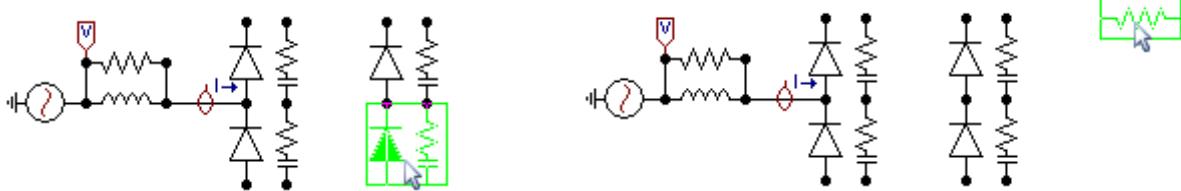


Fig. 3.19

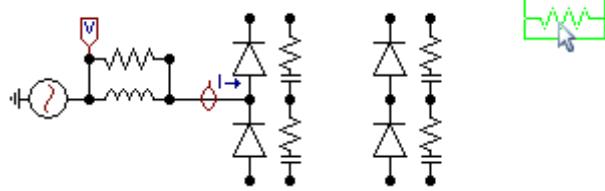


Fig. 3.20

As part of the connection between the rectifier bridge and the load a small resistor is included in Fig. 3.4. The resistor is included to demonstrate the option of using a small resistor for current measurement purposes.

Select a resistor in the component selection menu, then click on the resistor with the left mouse button, hold down and drag it to a desired position as shown in Fig. 3.20. You must place the resistor precisely, because the next step is to connect the top nodes of the diode bridge with the resistor.

Before doing so first, give data to this resistor opening the component dialog box by a right-click on the resistor. Specify data value  $RES= 0.01 \Omega$  and set *Output* to *I-Current* to get the branch current in the subsequent ATP run. Having closed the component dialog box a small  $\rightarrow$  symbol appears on the top-left side of the resistor indicating the current output request.

Now you can start to connect the diode bridge and the resistor together. The procedure is to first click with the left mouse button on a starting node, as shown in Fig. 3.21. The cursor style now changes to a pointing hand. Then release the mouse button and move the mouse (a rubber band is

drawn from the starting point to the current cursor position). To place a connection, click on the left mouse button again. Click on the right button or press *Esc* to cancel the connection make operation.

The connection draw in Fig. 3.21 picks up intermediate nodes so all the five nodes will be connected together. In this way, ATPDraw suits the requirement: “What you see is what you get” and the amount of required connections are significantly reduced.

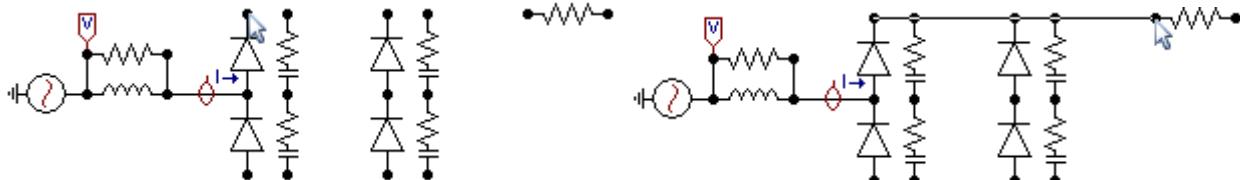


Fig. 3.21 - Click left button. Release + move, then click left button to place the connection.

If you made a mistake in the connection drawing process, you can correct the error easily, because connections are editable (copy/move/rotate) as any other objects. If you would like to correct/modify a misplaced connection, click on it and hold with the left mouse button. After this selection, the connection is enclosed by a rectangle and two squares replace node dots at the end of the line. To move the connection, click on an internal point of it using the left mouse button, then hold down and move, and release the mouse at the correct position. To reposition a connection, click on the node squares with the left button and stretch the connection as illustrated in Fig. 3.22:

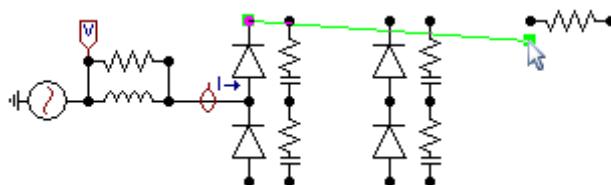


Fig. 3.22 - Edit connection. Click any point of the line then click node squares and stretch.

### 3.5.1.4 Load

The last part of this example circuit is the load consisting of a smoothing capacitor with initial condition and a load resistor. First, you can select the capacitor as shown in Fig. 3.23:

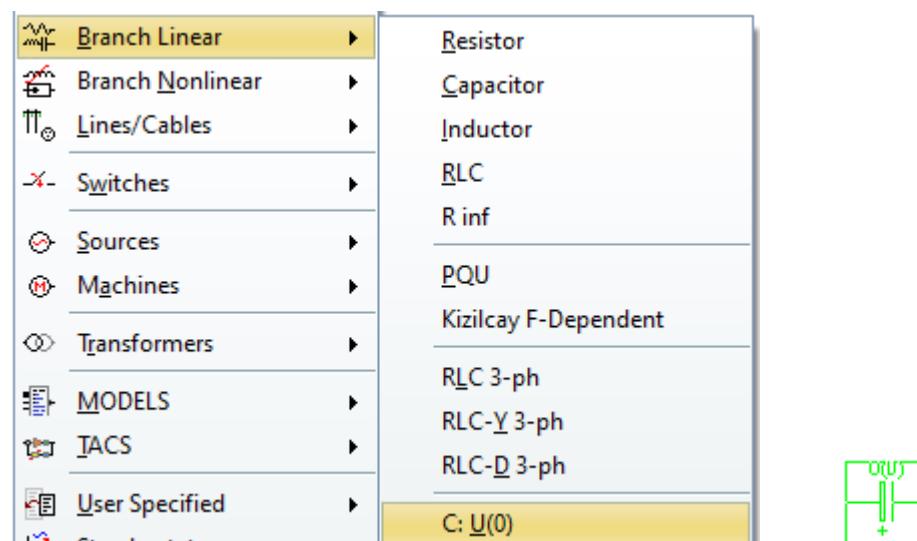


Fig. 3.23 - Select capacitor with initial condition.

After this selection, the capacitor appears in the middle of the circuit window in moveable mode enclosed by a rectangle. Click on the capacitor with the left mouse button, hold down and drag to a desired position, then click the right mouse button (or press *Ctrl+R*) to orient the capacitor as shown in Fig. 3.24. Finally, click on open space to place the capacitor.

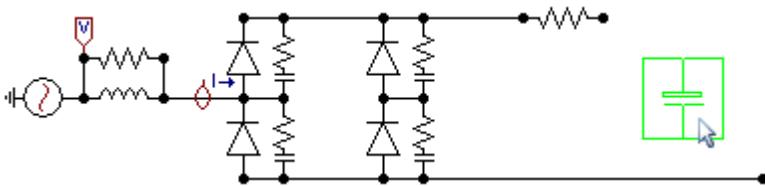


Fig. 3.24 - Placing a capacitor with initial conditions.

Next select the load resistor in the component selection menu *Branch linear + Resistor*. The resistor is drawn in moveable mode in the circuit window. Click on it with the right mouse button to rotate, then click with the left mouse button, hold down and drag it to a desired position and place as shown in Fig. 3.25.

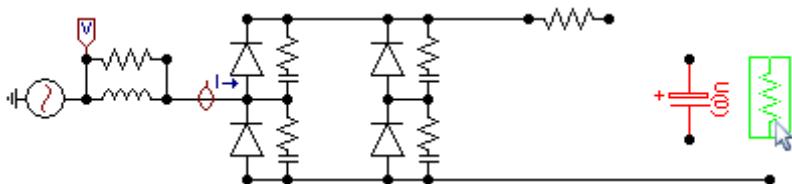


Fig. 3.25 - Place load resistor.

The time has come to connect the load to the rest of the diode bridge. The process has been explained before. Click on the component nodes you wish to connect with the left mouse button, sequentially, click left again to complete, right to cancel. Creating a circuit having only perpendicular connections (recommended for complex circuits, to improve the circuit readability) is a relatively simple task, as shown in Fig. 3.26.

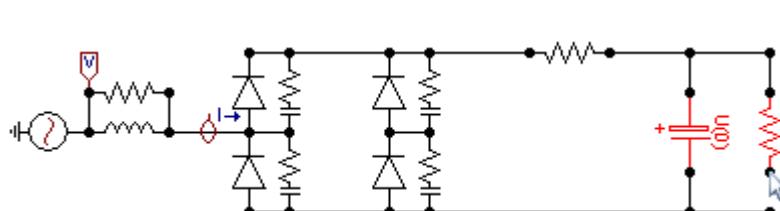


Fig. 3.26 - Your first circuit is almost ready!

After you have finished connecting the source side and the load side of the circuit, you can specify the load data. Click with the right mouse button on the capacitor and specify the parameters shown in Fig. 3.27.

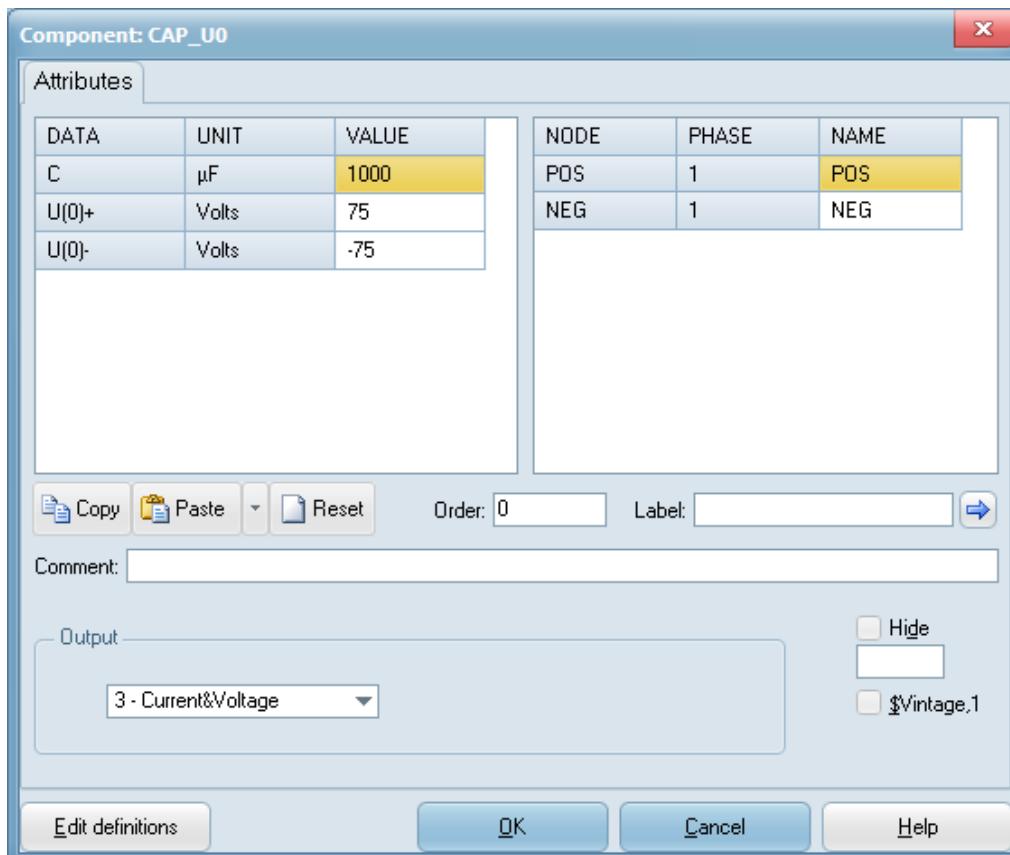


Fig. 3.27 - Capacitor data with initial condition.

The capacitance is 1000  $\mu\text{F}$  (if Copt=0 in *ATP / Setting / Simulation*). The positive node has an initial voltage of 75 V and the negative -75 V. Both branch current and voltage will be calculated, so the *Current&Voltage* is selected in the *Output* combo box. Following the branch output request, the appearance of the object's icon will change if the *Show branch output* is checked under *View / Option*. If this option is enabled, a small  $\text{V}_i$  symbol appears on the top-left side of the capacitor, indicating the branch voltage and the current output requests (see Fig. 3.28).

Next click with the right mouse button on the load resistor to get the input window and specify the load resistance of  $20 \Omega$ . Branch current and voltages will be calculated so the small  $\text{V}_i$  symbol appears again on the top-left side of the resistor after leaving the dialog box. Once all the entries in the component dialog box are completed, select the *OK* button to close the window and update the object values or click *Help* to obtain an on-line help.

### 3.5.1.5 Node names and grounding

The final step of building this circuit is to give data to nodes (node names and grounding). All nodes will automatically receive names from ATPDraw, so the user should normally **give name to nodes of special interest** only. It is advised in general to perform the node naming as the last step in building up a circuit. This is to avoid undesirable multiple node names (which is corrected by ATPDraw automatically but results in irritating warning messages).

To give data to a node, you simply have to click on this node once with the right mouse button. Fig. 3.28-Fig. 3.31 show how to give data to four different nodes.

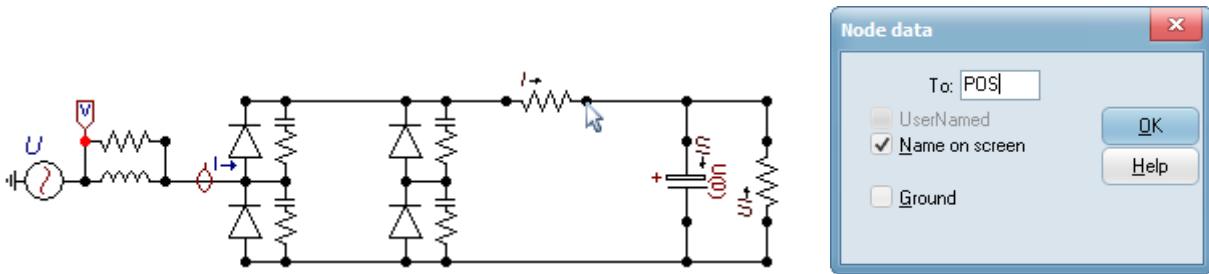


Fig. 3.28 - Click on a node with the right mouse button and specify a name in the dialog box.

When you exit the window in Fig. 3.28 by clicking *OK*, the circuit is updated as shown in Fig. 3.29 and the node dot turns red. All node names are forced left adjusted, and as a general rule in the ATP simulation, capital letters should be used. ATPDraw does accept lower case characters in the node data window, however this “feature” should be avoided, in particular if the node is connected with electrical sources.

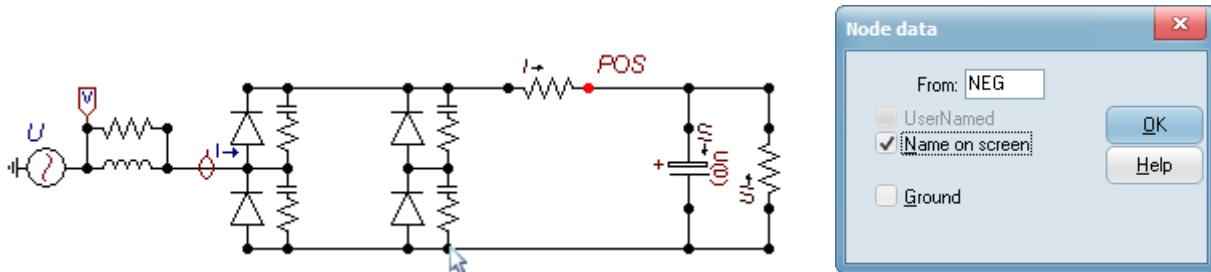


Fig. 3.29 - Click on a node with the right mouse button and specify a name in the node data window. The name ‘NEG’ will be assigned to all nodes visually connected.

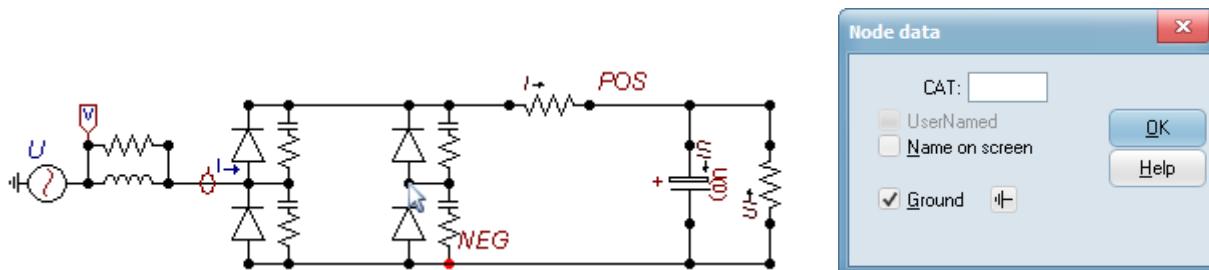


Fig. 3.30 - Click on a node with the right mouse button and check the *Ground* box indicating that the node is connected with the ground reference plane of the circuit. The button right to the *Ground* check box can be clicked to choose the ground symbol orientation.

The ground symbol is drawn at the selected node when you exit the window as Fig. 3.31 shows. The nodes not given a name by the user will automatically be given a name by ATPDraw, starting with XX for single phase and X for 3-phase nodes followed by a four-digit number. Nodes with a name specified by the user are drawn in a red color and the disabled check box *User Named* in their node dialog box is checked. Fig. 3.31 shows the final step in the drawing process.

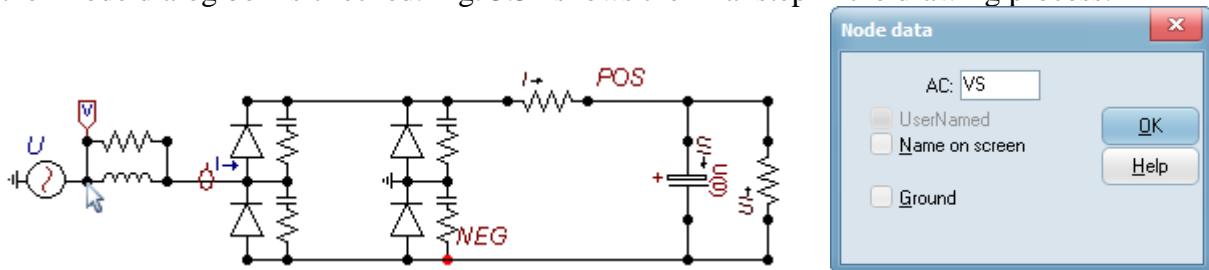


Fig. 3.31 - Click on the voltage source with the right mouse button and specify the node name.

### 3.5.2 Storing the project file on disk

You can store the project in a disk file whenever you like during the building process. This is done in the main menu with *File / Save* (or *Ctrl+S*). If the current project is new, a *Save As* dialog box appears where you can specify the project file name and location on the disk. The default extension is *.acp* in both cases and it is automatically added to the file name you enter.

The user can also choose two other project file formats (*Save as type*); *Version 5/6 compatible* and *XML text format*. Both these formats are less complete.

When the circuit is saved, the name of the disk file appears in the header field of the circuit window. Then if you hit *Ctrl+S* or press the *Save circuit* icon in the Toolbar, the circuit file is updated immediately on the disk and the *Modified* flag in the status bar disappears. The *File + Save As* option or the *Save As* Toolbar icon allows you to save the circuit currently in use under a name other than that already allocated to this project. There are no project file name restrictions.

### 3.5.3 Creating the ATP input file

The ATP-file describes the circuit according to the ATP RuleBook. You can create this file by selecting *Sub-process|Make ATP File* command in the *ATP* main menu. The ATP-file is regenerated whenever you execute the *run ATP* command (or press *F2*). In the latter case the process is automated. By default, the ATP file inherits its name from the project file.

However, before you create the ATP input file or run the simulation, you *must not* forget to specify the miscellaneous parameters (i.e. parameters, that are printed to the Misc. Data card(s) of the ATP input file). The default values of these parameters are given in the *ATPDraw.ini* file. Changing these default values can either be done in the *ATP / Settings / Simulation* sub-menu (or in the *Sidebar*) for the current project, or under the *Tools / Options / View/ATP / Edit settings / Simulation* for all new ATPDraw projects created henceforth.

Fig. 3.32 shows an example of the 1<sup>st</sup> miscellaneous data card settings of an ATP simulation (specifying time step, time scale of the simulation etc.). This window appears if you select the *Simulation* tab of the *ATP / Settings* dialog. The most important settings can also be made in the *Sidebar*. The simulation type (time domain or frequency scan) can also be set on this page.

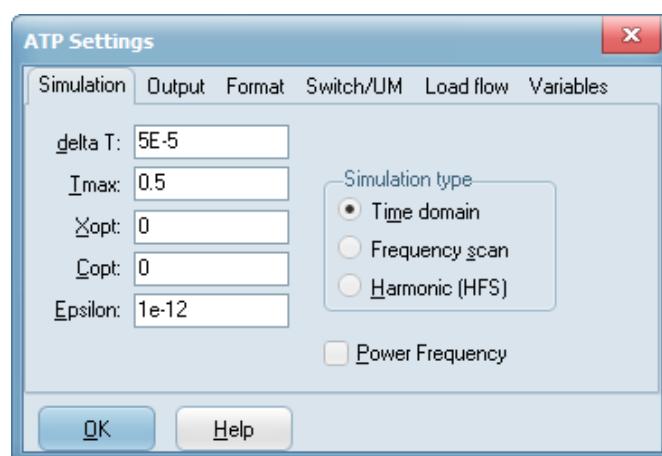


Fig. 3.32 - Simulation settings.

- Time step *delta T* in sec.
- End time of simulation *Tmax* in seconds.
- *Zopt=0*: Inductances in mH.
- *Copt=0*: Capacitances in  $\mu\text{F}$ .
- *Epsilon*: Accuracy value. A zero value means the default from STARTUP is used (1E-8?). It could be important to set *Epsilon* to 1E-12 or less to prevent incorrect singularity warnings.

Press *Help* to get more information or *OK* to close the dialog box.

The simulation settings are stored in the project file, so you should save the file after changing these settings.

Values on the first integer miscellaneous data card of ATP can be changed under the *ATP / Settings / Output* page. The next *ATP / Settings/ Switch/UM* tab is the home of control flags

required by statistical switching or universal machine simulations.

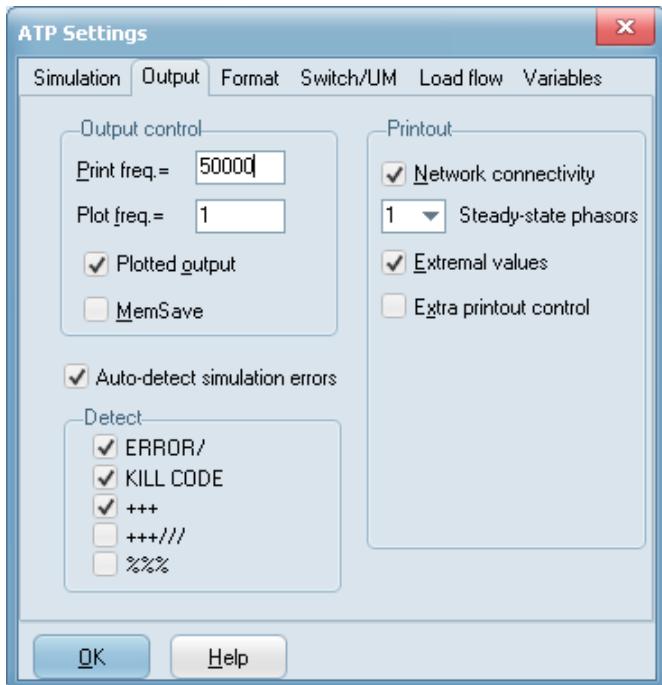


Fig. 3.33 - The ATP-file format menu.

To create an ATP-file without starting the simulation you must select the *Sub-process|Make ATP File* in the *ATP* menu. This selection will start the compilation, which examines your circuit and gives node names to circuit nodes. Then a standard Windows' *Save As* file window appears, where you can specify the name and path of the ATP-file. The same name as the project with extension .acp file is suggested default. As the ATP file is sent to the ATP solver, the file name should not contain space characters. You can edit this file or just display it by selecting the *ATP / Edit ATP-file* menu. The ATP-file (*Exa\_1.atp*) you have just created will be as follows:

```
BEGIN NEW DATA CASE
C -----
C Generated by ATPDRAW August, Monday 26, 2019
C A Bonneville Power Administration program
C by H. K. Høidalen at SEFAS/NTNU - NORWAY 1994-2019
C -----
C   dT >< Tmax >< Xopt >< Copt >< Epsiln>
      5.E-5      .05
      500          1       1       1       1       1       0       0       1       0
C   1           2       3       4       5       6       7       8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n1 >< n2 ><ref1><ref2>< R >< L >< C >
C < n1 >< n2 ><ref1><ref2>< R >< A >< B ><Leng><><>0
  VA    XX0001            33.          1.          0
  VA    XX0001            33.          1.          0
  NEG   VA               33.          1.          0
  NEG   VA               33.          1.          0
  XX0001POS          .01
  POS    NEG              1.E3
  NEG    POS              20.
  VS     XX0002            1.
  VS     XX0002            300.
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CLOP >< type >
11VA    XX0001            0
11      XX0001            0
```

Under the *Output* page the user can select content of the output LIS and PL4 files. *Print freq.* tells the time step interval reported in the LIS file (this should be a large number to prevent very large unmanageable text files) *Plot freq.* tells the time step interval into the PL4 plotting file, and the checkbox *Plotted output* if a PL4 file will be produced at all.

*Printout* gives more details about what diagnostic data that will be written to the LIS-file.

Auto-detect simulation errors (also set in the ATP Connection Wizard) will enforce ATPDraw to read the LIS-file after the simulation and look for error and warning, and the show the message in a text file window

```

11NEG   VA          0
11NEG
  XX0002VA          0
/SOURCE
C < n 1><>< Ampl. >< Freq. ><Phase/T0>< A1    >< T1    >< TSTART >< TSTOP >
14VS      169.705627     60.       -90.           -1.        1.
/INITIAL
2POS          75.
2NEG          -75.
3POS   NEG      150.
/OUTPUT
  VS
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK INITIAL
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK

```

### 3.5.4 Running the simulation

Starting the ATP simulation is supported in ATPDraw in a user-friendly way. The user just has to press *F2* function key to create an ATP input file with the current project file as input and run the simulation. *ATP|run Plot (F8)* starts the default plotting program and sends the *pl4* file as parameter. The default commands that is executed when the user selects *run ATP* or *run Plot* under the *ATP* menu as it has been described in section 2.5 of the Installation Manual.

### 3.5.5 Simulating with the internal solver

Start the Internal Solver by holding down the *Ctrl*-key while selecting *ATP|run ATP*. This will create a *PL4*-file with the results and also a *LIS*-file with limited debug information. It will also show a plot window with all plots, the *LIS*-file and a speed plot of the time in microseconds used in the time-step loop.

## 3.6 Node names

The user should give names to nodes used in outputs in order to identify this in plotting programs. Otherwise, ATPDraw will take care of the node naming. The user can choose *ATP/Subprocess/Make node names* to force ATPDraw to create the node names. Selecting *ATP/Run ATP* will do the same thing but also execute ATP. After node names are created, the user can right click on the nodes to see the name and phase. Single phase nodes will have no phase extensions unless they are connected to a Splitter or Connections where the phase is specified. Three-phase nodes will have phase extension ABC unless Transposition objects are involved, or the DEF object is connected to the circuit. The node property Circuit is used to identify which nodes of the component that should be transposed together.

In the case of Groups there are a few restrictions from the otherwise *what-you-see-is-what-you-get*. The basic principle is that the Group component enforces the node names on its connected content. The content can never enforce node names back on the Group, this applies also to the nodes' phase index. The following restrictions thus apply:

- Not legal with internal connections between two nodes from parent
- Not legal with transposition between two nodes from parent
- Not safe with a different phase index in child and parent nodes. The Group nodes will indeed inherit the content's phase index during Compress and the user can click on the

Group node and override the phase index directly. However, the user must pay attention if a node phase index in the Group content changes later due to other edit operations.

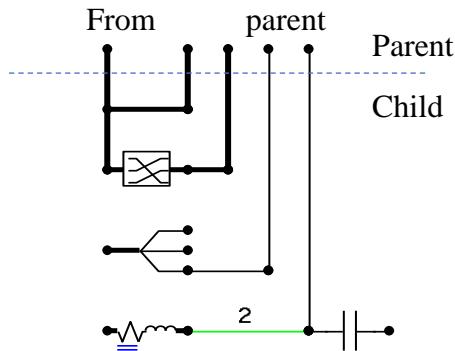


Fig. 3.34 - Illegal or unsafe child node connections in version 7.

### 3.6.1 Multi-phase circuits

From ATPDraw version 5 a node can have up to 26 phases (A..Z node name extension). This applies also to MODELS nodes. A more generalized *Connection* is introduced with a special handling between single phase and n-phase nodes. Transpositions will only take place through 3-phase connections. In this case the phase sequence will be further inherited throughout the circuit. Special ABC or DEF reference components found under from *Probes&3-phase* in the *Selection menu* can be placed on the reference node. The actual phase sequence of the node is written at the top right of the Node dialog box or in the PHASE field in the Component dialog box as shown in Fig. 3.3 (after *ATP/run ATP* or *ATP/Sub-process/Make node names*). A special component SPLITTER is available for connections between 3-phase and single-phase nodes. Some special restrictions apply to the Splitter objects (found under *Probes & 3-phase* in the component selection menu):

- Connecting splitter objects together on the 3-phase side or with connections on the 1-phase side is permitted, but transposition/disconnection is not allowed.
- If the name *NODEA* is given to what you know is phase A on the single-phase side, ATPDraw does not accept this and adds its own A at the end, creating the node name *NODEAA*. The general rule is that ATPDraw takes care of the phase sequence! The best solution is to specify a node name on the 3-phase side only.

Color, label, and phase properties are given to the *Connection* as well as the possibility to force node dots on. Fig. 3.35 shows the Connection dialog that appears after a right click on the connection and automatically when the user draws a connection between a single phase and a multi-phase node. The *Phase index* field is only enabled for single phase connections. 0-@ is used for connections between two single phase nodes (no phase extension to node names).

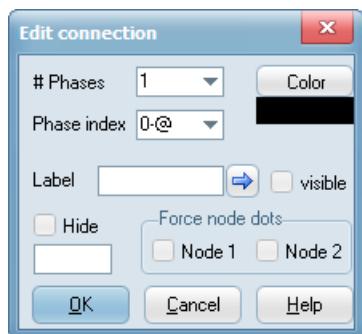


Fig. 3.35 - The Connection dialog box.

Fig. 3.36 illustrates the various options for (3-phase in this case) multiphase circuits in ATPDraw. The flag DEF set at the source node to the left. The color of each connection is user selectable as shown in Fig. 3.35 , but as default the color and phase sequence are inherited when the user clicks on one connection to draw a new one. There is an important difference compared to version 5&6; a Connection with phase index ‘1-A’ connected to a multi-phase node will carry the first phase and not necessarily phase A. This difference is only relevant in case of Transpositions.

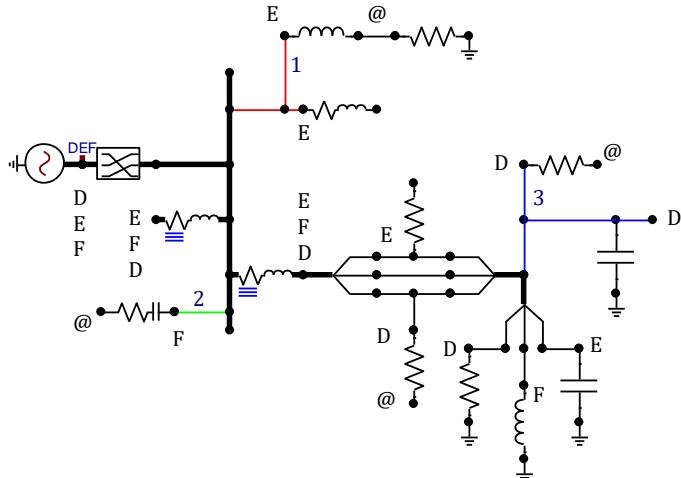


Fig. 3.36 - Illustration of various phase options in ATPDraw.

A typical example of connecting a single-phase node to a 3-phase node is the case of a single-phase ground fault as shown in Fig. 3.37 . Place the switch, then draw the connection between the three-phase node and the single-phase node. Select 1-A to ground the first phase of the node (depends of transpositions involved).

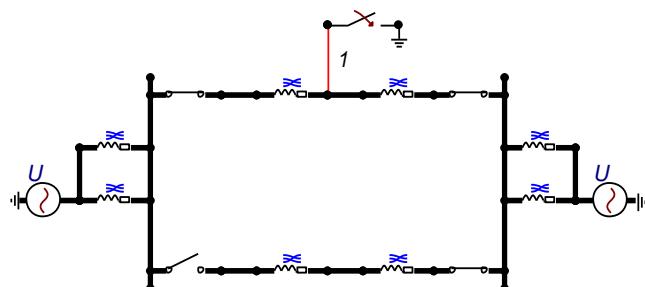


Fig. 3.37 - Single-phase ground fault.

Multi-phase nodes are first of all important for *MODELS* and *GROUPS*. An  $n$ -phase connection could also be useful just to clear up the circuit drawing. As an initial example, a 6-phase connection is shown in Fig. 3.38 for communication between a 6-pulse thyristor bridge and its control circuit. This will make the drawing much easier to read.

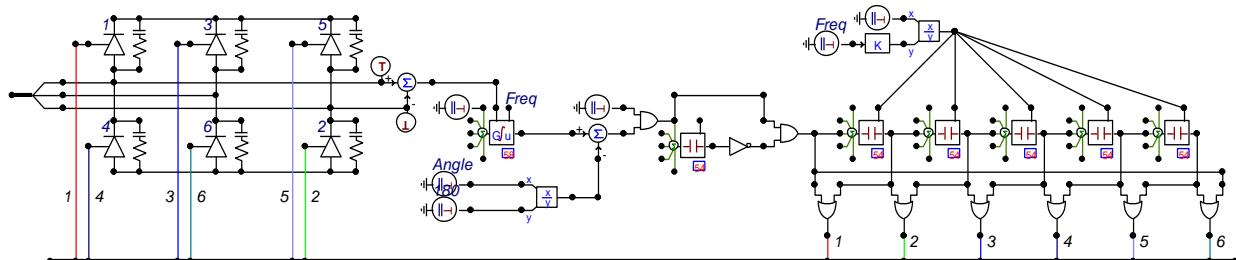


Fig. 3.38 - Communicating a 6-phase signal between a thyristor bridge and its control circuit.

All n-phase nodes have only 5 characters available in the *Node dialog box*. ATPDraw adds the extension *A*, *B* and *C* (etc.) at the end of the node name. By default, the phase sequence is *ABC*; the first data card uses *A*, the second *B* and the last *C*. The only way to change the phase sequence is to use the available transposition objects (*Transp1 - Transp4*) selectable under *Probes & 3-phase* in the component selection menu. Only 3-phase nodes can be transposed.

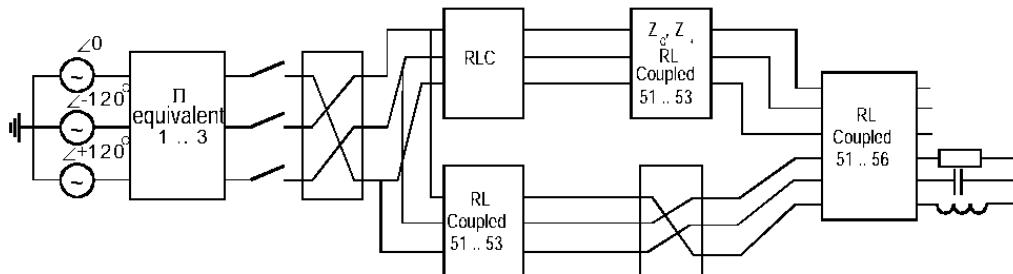


Fig. 3.39/a - Illustrative three-phase circuit.

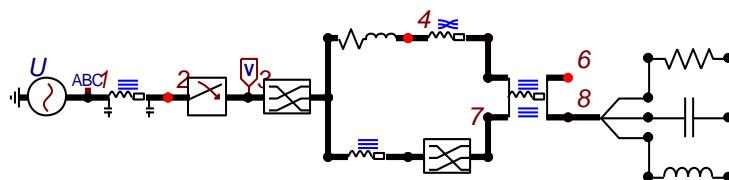


Fig. 3.39/b - Equivalent ATPDraw circuit (Exa\_2.acp).

The circuit shown in Fig. 3.39 was built up in the same way as your first circuit. You can note that connections between the three phase nodes appear to be thick. The circuit contains 3 special objects, the already mentioned transposition object (in this case from *ABC* to *BCA*), a Splitter object, which splits three phase nodes into three single-phase nodes and an *ABC reference* object. Fig. 3.40 shows the Node data dialog for a single phase and a three-phase node. If the extra option *Short circuit* is checked, the node becomes single-phase (all phases short circuited)

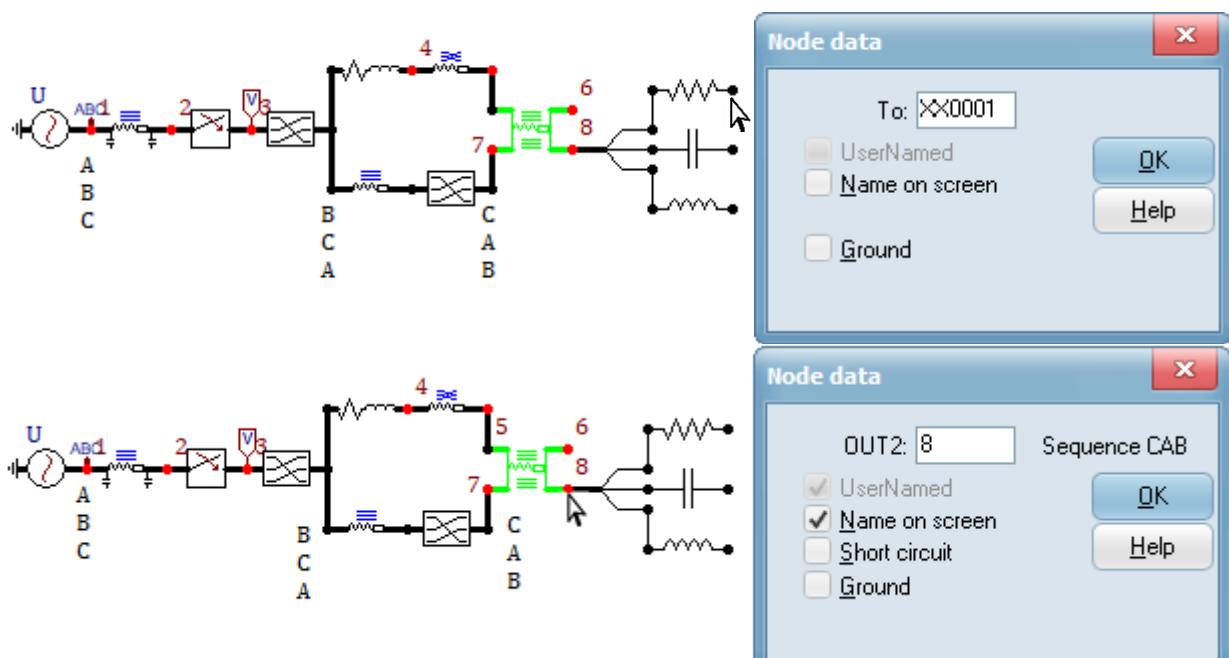


Fig. 3.40 – Default node names and phase sequence. Top: single phase node. Bottom: 3-phase.

### 3.7 Data values

In the data part of dialog boxes, the user can specify values using '.' as the decimal symbol and 'e' or 'E' as exponent symbol. Mathematical expressions are also supported and an input '120E3\*sqrt(2/3)' will be converted to a value when *OK* is clicked. If the value is illegal or outside the allowed range the user will be directed and forced to change the value. The range can be changed inside *Edit definitions*.

A variable name can also be specified and given a global value/expression in *Sidebar/Simulation* or under *ATP/Settings/Variables*. If the user specifies a variable, a Confirmation dialog pops up and ask if the variable should be accepted and added to the list of variables, as shown in Fig. 3.41. Click *Yes* or *All* if the variable specification was intentional. The purpose of assigning a variable is primarily if a value is used in several components or if its value should be modified in multiple-run parameter variations.

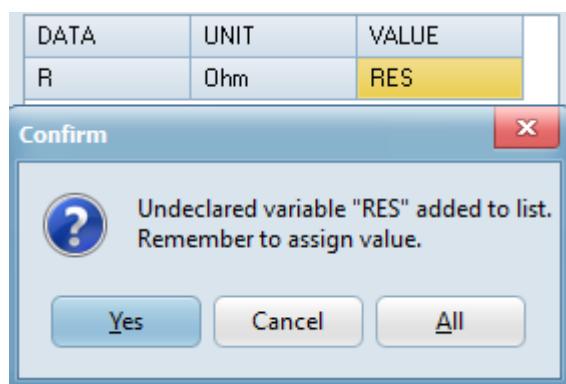


Fig. 3.41 – Add Variable to list confirmation.

From ATPDraw v7.4 there is no limit on the number of characters for Variables. If the ATP's native mechanism is used the variable name must be truncated to 5 characters and appended a number for uniqueness. With several, long and almost equal variable names are used it could be difficult to verify or debug the case. With *Use Parser* checked, the data value can even contain expressions 'RES\*1000' where RES must be defined among the Variables. Specifying a variable is only allowed if the *Internal Parser* is used or the *Param* property of the data in *Edit definitions* is set to unity. Warning messages will appear in case of illegal specifications and the user can modify the data.

Variables are not allowed in the BCTRAN and XFMR components, but in the LCC component.

Data values in lime color are inherited from the parent (group) component and cannot be changed inside the child.

#### 3.7.1 Parameter variations

Sometimes the user would like to study the effect of parameter variations. This can be done by assigning a Variable to data and then manipulate this under *Sidebar/Simulation* or *ATP/Settings/Variables*. Intermediate variables can be defined to support data variables. The default variable KNT is the simulation number counter and in the example in Fig. 3.42, this goes from 1 to #Sim making the resistance go from 100 to 1000 ohms in steps of 100 ohms. If *Internal Parser* is checked, an ATP-file for each simulation is created and ran in series utilizing multiple CPU-cores in parallel threads. This will also allow manipulation of data used in internal calculations. Otherwise, the native \$PARAMETER and PCVP mechanisms of ATP are used. This restricts the manipulation to data with the *Param* property set to unity. The length of

variables can any number of characters and with the *Internal Parser*, nesting of variables is allowed (previous values used), otherwise the variables must be declared purely sequential, and a period ‘.’ character must be used in all expression numbers. Multiple result files (PL4) are created and extended with ‘001’, ‘002’ numbers and this requires manual inspection in plotting programs. Extremal values can be extracted by the *WriteMaxMin* or the *Extract1/3* components, see Chapt. 5.9.

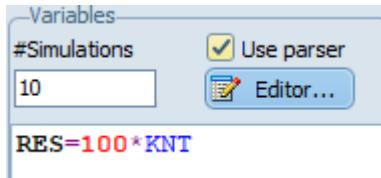


Fig. 3.42 – Variable variations

### 3.8 Object organization, sequence and priorities

ATPDraw is a preprocessor that creates the input file to ATP (ATP-file). ATPDraw organizes all objects (including components) in an object list. Components are written to the ATP-file in the sequence they appear in this list (unless sorting options are enforced), and this is the same as the sequence they were added to the project. If a component is deleted and *Undo* selected, it will appear at the end of the list. This is different from earlier versions where vacant positions were filled later and the sequence could be messy over time. The user can see and organize the sequence of all objects in the *Object Tree Inspector* in *Sidebar/Project*. Three mouse operations are available in Object Tree Inspector:

- Left click on object to highlight it and center the circuit around it.
- Right click to open the object for input.
- Left click and hold to drag it to a new position in the list.

The sequence of components in the ATP-file does typically not matter unless the user needs to dig into the file for debugging or manual editing. However, there are a few cases when the sequence is critical. A few examples are:

- A MODEL using output from another MODEL must come later in the ATP-file. A sorting feature for MODELS is supported and found under *Edit/Arrange/Sort all Models*.
- A switch current or status signal into MODELS/TACS. If several switches share the node used, the first switch is chosen.
- A GIFU switch will only invoke diodes following in the ATP-file.
- In UM, a control source (type 60) must come after the type 14 sources used for automatic initialization. Note that type 14 sources are hidden inside the UMSYN component.

In earlier ATPDraw versions, the Component Order and *Sort by Order* mechanism was the only way to change the component sequence. This option is still available, but not recommended due to lack of transparency. Besides the Sidebar’s Object Tree Inspector organization, the user can also select a Component and then *Edit/Arrange* (or right click and select *Arrange* in the context menu). Selecting ‘Send to back’ will move the object to the end of the list. Note that the Object Tree Inspector is not automatically updated, so the user should click on *Update*.

When clicking on objects in the circuit window, the first object hit in the list will be chosen. It could be inconvenient if you click on a Component node the Connection dialog appears. The option *Edit/Arrange/Send Connections Back* will, by sending all Connections back and thus reduce their priority, prevent this and make the appearance equal to earlier ATPDraw versions. I direct choice for this is also found above the Object list in the sidebar.

### 3.9 Post-processing

ATPDraw post-process data in several ways. Current and voltage probes can display simulation results (extract phasors) directly on screen. The Power System Toolbox introduced a method where MODELS is used to write data to the LIS-file and read this back into components. This approach is further utilized in the COMTRADE objects introduced in ATPDraw 7.0-7.1. ATPDraw v7.1 introduced a new object type TATPDrawPlot for direct embedding of PL4 simulation results.

#### 3.9.1 Probes

Node voltage and current probes (probe\_v, probe\_i) have options to display steady-state values on screen. There is one choice called *Steady-state/Enable* that turns the capture on/off and one option *On screen/Enable* that turns on/off presentation on screen. The type of format of the screen output is to a large extent user controlled and it is written as a movable green label on screen. The user can choose to extract phasors at a time  $T>0$  and in this case a MODEL WRITEPROBEV or WRITEPROBEI are added to the ATP-file. This will extract the phasor from a window looking  $1/FREQ$  back in time. All phases of the probe are captured in a grid where the user can copy out data to windows clipboard (without scaling). The values on the screen and in the grid are simulated peak values divided by the user specified Scale factor. The Scale should typically be  $\sqrt{2}$  to get RMS values or  $\sqrt{2}/3$  to get line-quantities in rms. For 3-phase probes, output on screen can be sequence parameters.

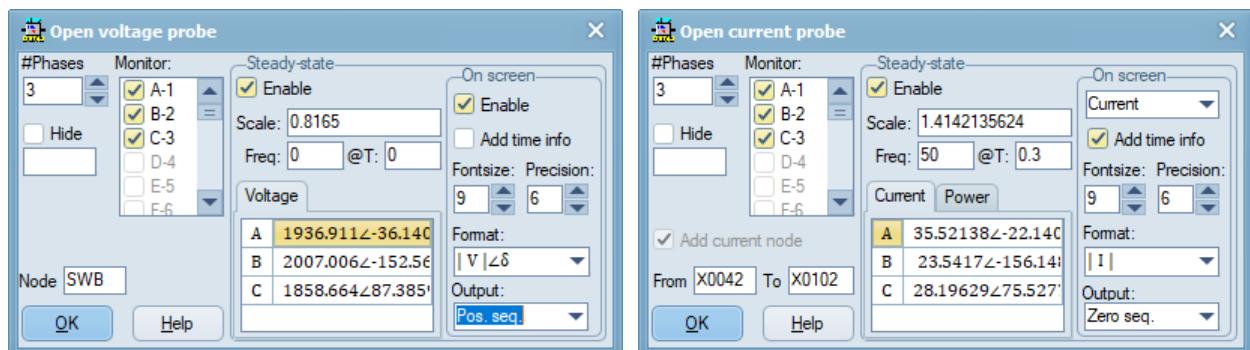


Fig. 3.43 – Voltage and current probes that write phasor values to screen.

A component under *MODELS>Show maxmin* will extract the extremal value of a simulation within a Tstart-TStop interval and display this on screen. The flexible transmission line model LINE3 can also output time zero steady state current and/or power flow out on the line.

#### 3.9.2 Power System Toolbox

Several components in the Power System Toolbox extract phasors and trajectories. This applies to all relay models. Relays have a View button that display trajectories (current or impedance) until tripping. This will display the standard plotting window in ATPDraw as shown in Fig. 3.44.

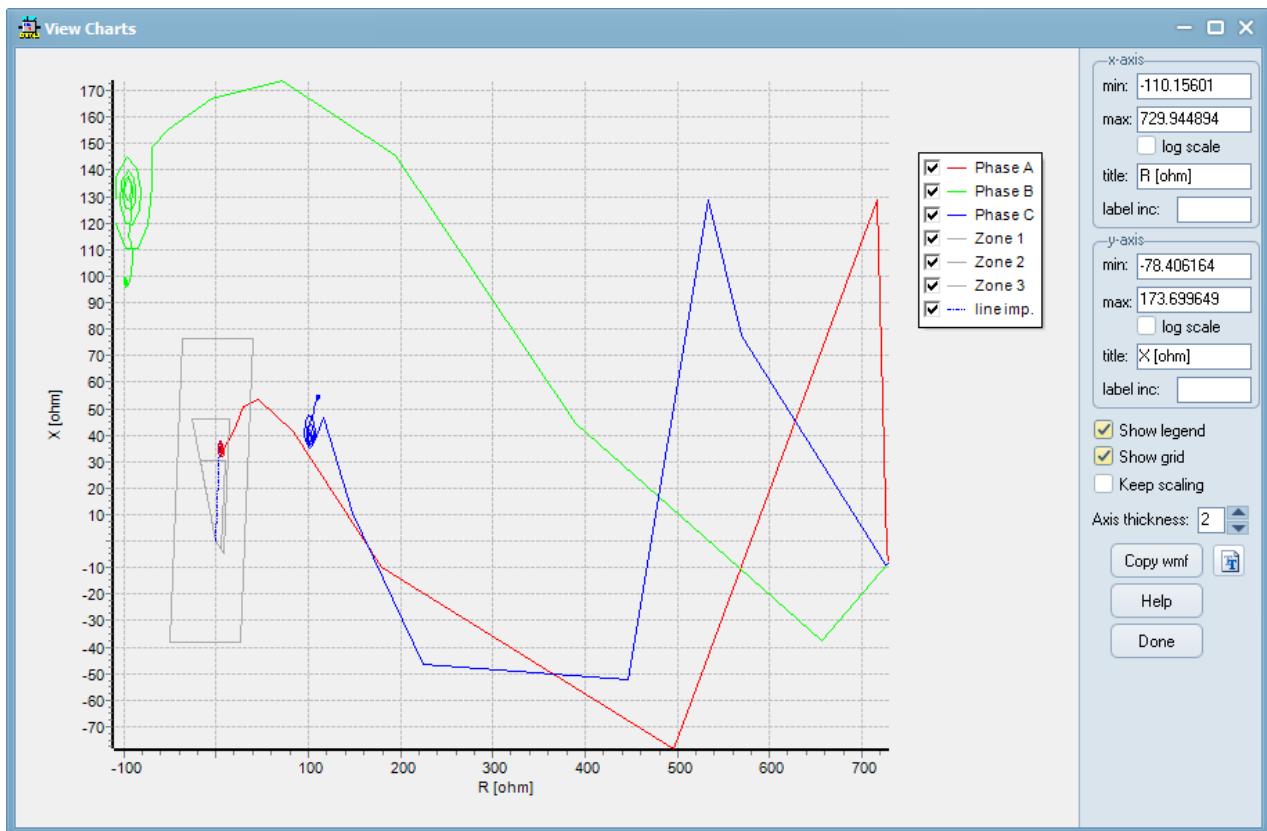


Fig. 3.44 – Plotting window used for distance relay.

### 3.9.3 COMTRADE

ATPDraw can also export simulation data as COMTRADE IEEE C7.11-1999 binary or ascii (alternatively MatLab v4) format. Three different COMTRADE objects are available. The first called just COMTRADE assumes 3-phase inputs and can be connected directly to three-phase nodes and thus knows the input's name and if the input is current, voltage or digital. The other two COMTRADE1 and COMTRADE2 both needs a packing object in MODELS that merge input and internal calculations to analog and/or digital channels. The COMTRADE objects offers a very flexible, user-selectable sample frequency (up or down sample), a starting time, and allocation of standard channel scaling and naming. The COMTRADE input dialog also has a View button that displays (similar to Fig. 3.44) the raw simulation results as read in from the LIS-file. The COMTRADE objects will automatically export .dat and .cfg files to the Result Directory after each run. The feature to export as MatLab v4 .mat-file works the same way.

### 3.9.4 Embedded Plotting

ATPDraw has from v7.1 an embedded plotting object that reads the PL4-file directly (NEWPL4=0/2 format) and displays user selectable traces. The user can choose to save data in the project file or not, but the names of selected traces are always stored. There can be many plotting objects inside a project, and they can be placed inside groups. The user can move the Plot objects around and scale them. One unique feature with the embedded plotting object is that the user can easily select to plot a trace as function of the run number in multi-run simulations as shown in Fig. 3.45 where the inrush current is plotted as function of the switching instant. The selection of Plot objects is somewhat different compared to other objects, since left and right mouse click&hold inside the Plot is used for zooming and panning, respectively. A handle on the right side can be used to drag the component without selecting it first via the Context menu shown in Fig. 3.45.

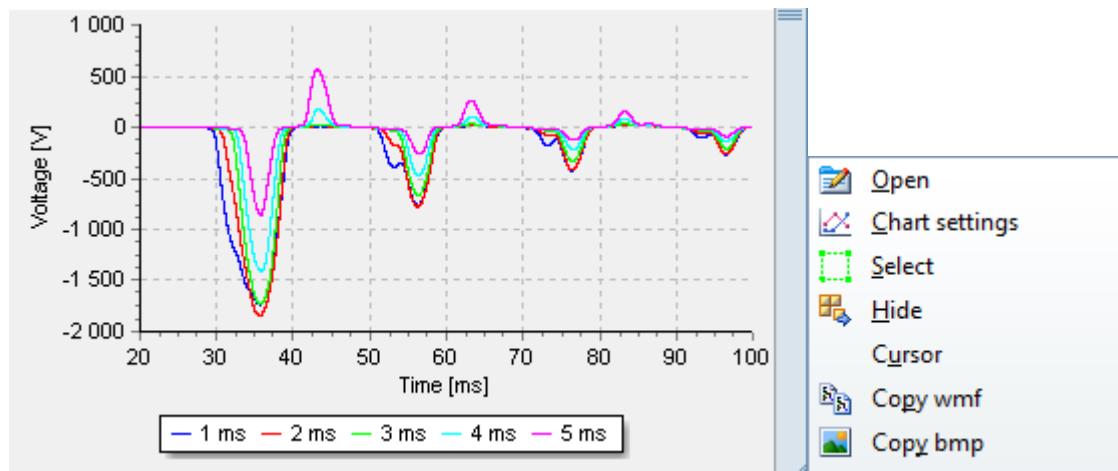


Fig. 3.45 – Embedded Plotting object. Context menu after right click.

The traces to plot are selected inside the dialog box of the Plot object (Open), Fig. 3.46. The number of plots must be selected first, then the actual plot from the combo box that appears when clicking in the *Series name* column. The sequence is the same as in the PL4 file and follows the Output Manager (F9) with naming convention like PlotXY. The colors option follows the standard Windows colors but selecting Custom will enable all possible colors for selection. The *run* column is used to select which multiple-run case to display. The curve is plotted as  $c(t)=PL4(t\text{-Skew})\cdot\text{Scale}+\text{Offset}$ . Clicking the Right column button will align the curve on the Plot's right axis. *Save plots in project* have three options. *Plot definition* will only save grid above, *+chart settings* will also save all Plot object settings (axis, zooming etc.) and *+data values* saves the actual data, so the curves displays immediately when loading the project. The data are stored with single precision (same as PL4) and compressed but beware of possible large project file sizes. *Draw reduced samples* reduces the accuracy somewhat but speeds up the drawing. *GDI+* draws more smooth curves. On the *Settings* page axis and panel can be adjusted as shown in Fig. 3.47. The *Advanced settings* brings up the extremely rich native chart setting dialog. This allows fine tuning of fonts, positions, and appearances, as shown in Fig. 3.48. The settings made in this dialog are also stored if *+chart settings* or *+data values* are selected in Fig. 3.46.

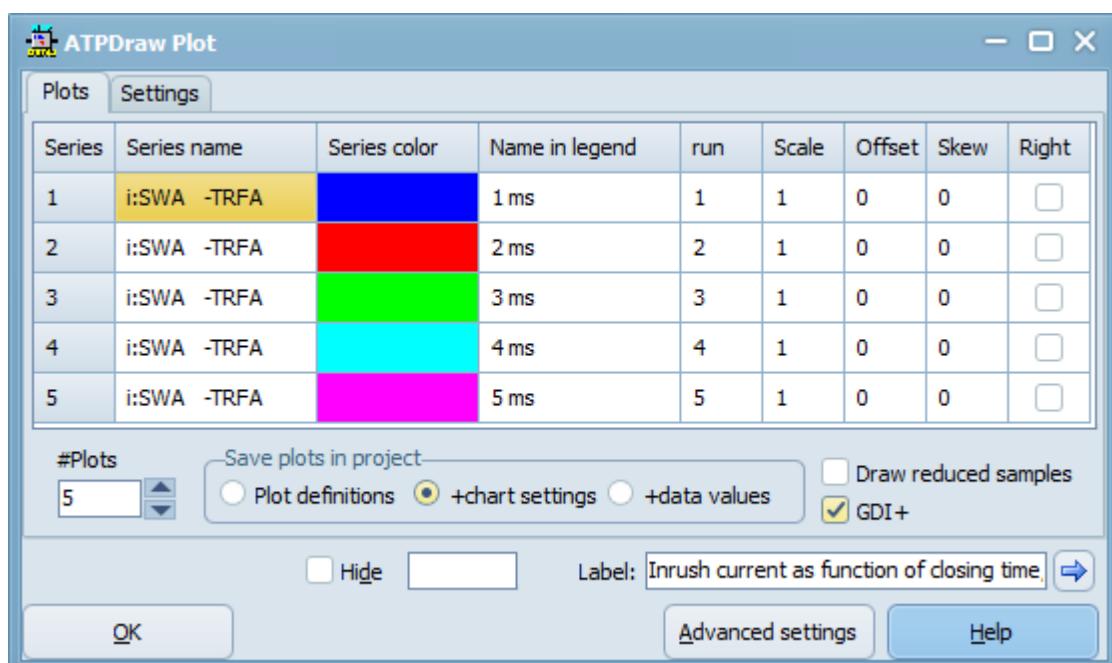


Fig. 3.46 – Embedded Plot dialog, main Plot page

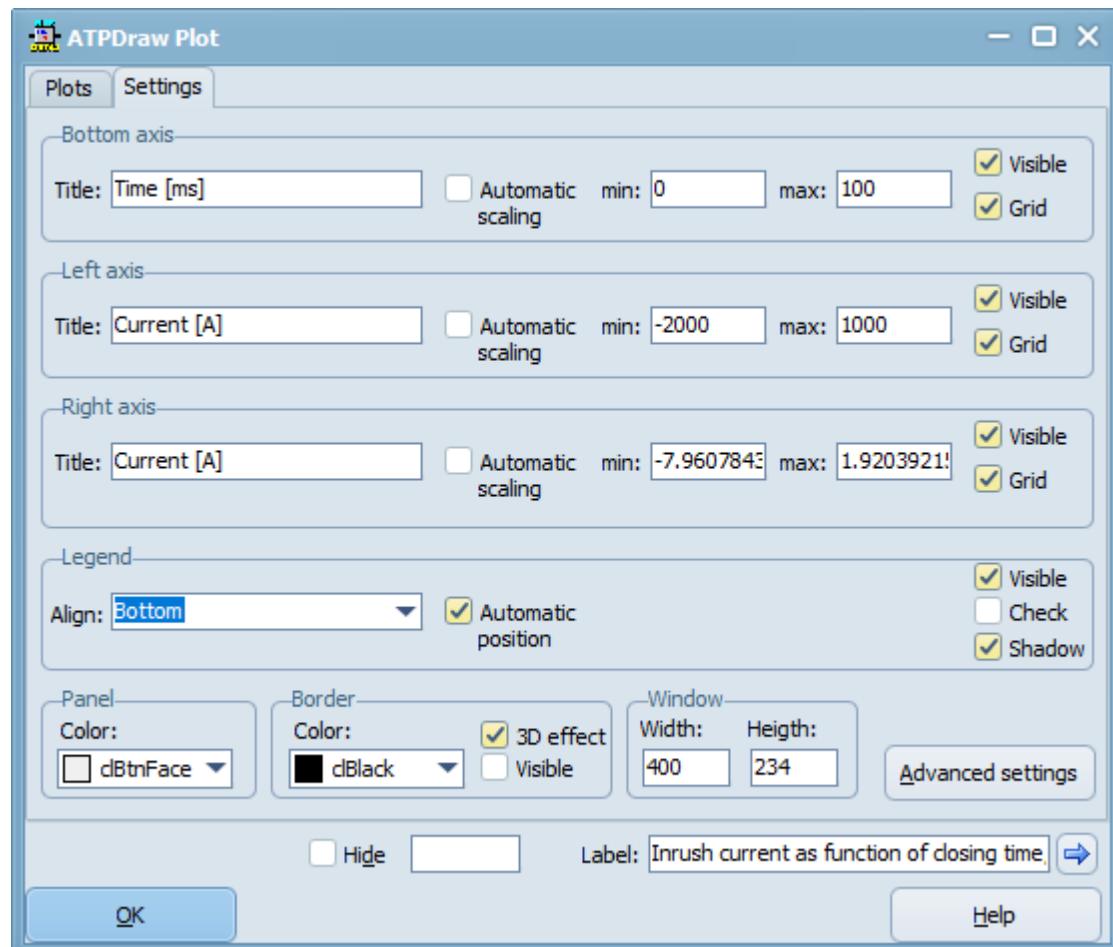


Fig. 3.47 – Embedded Plot dialog, Settings page.

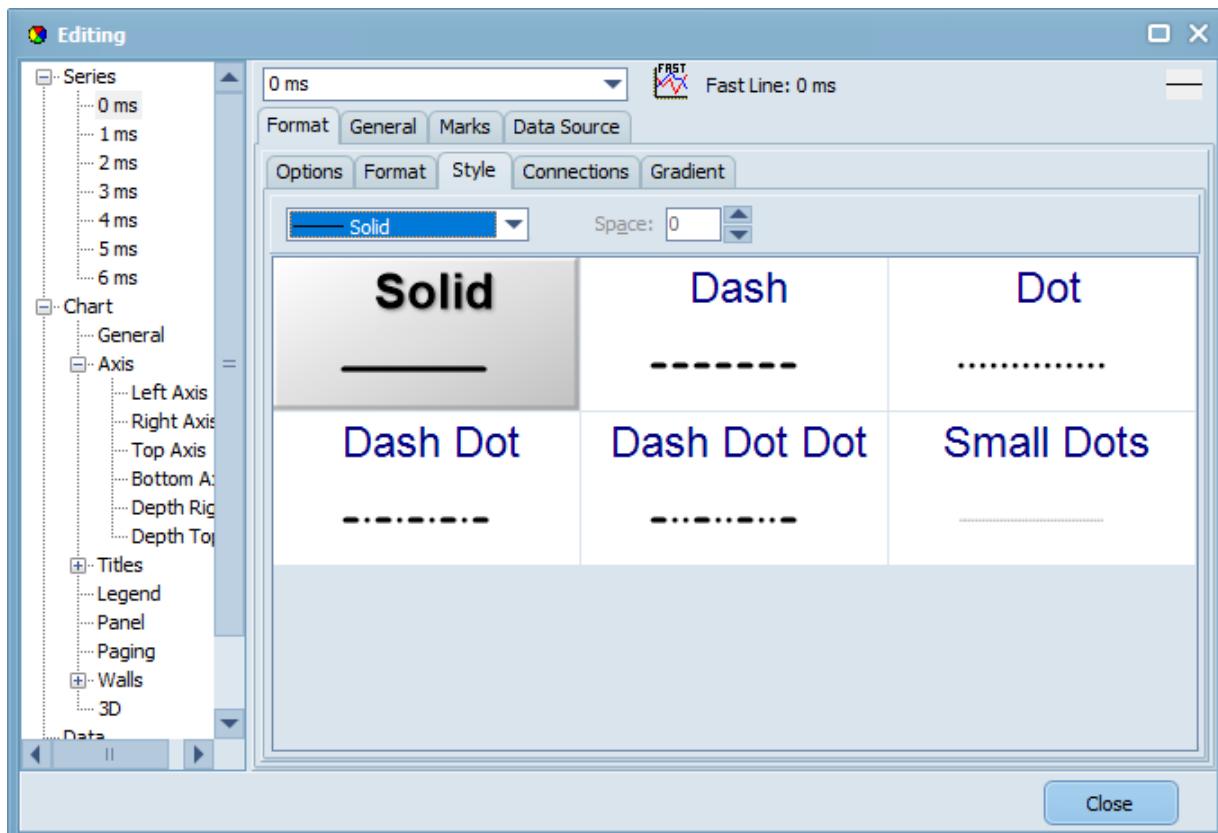
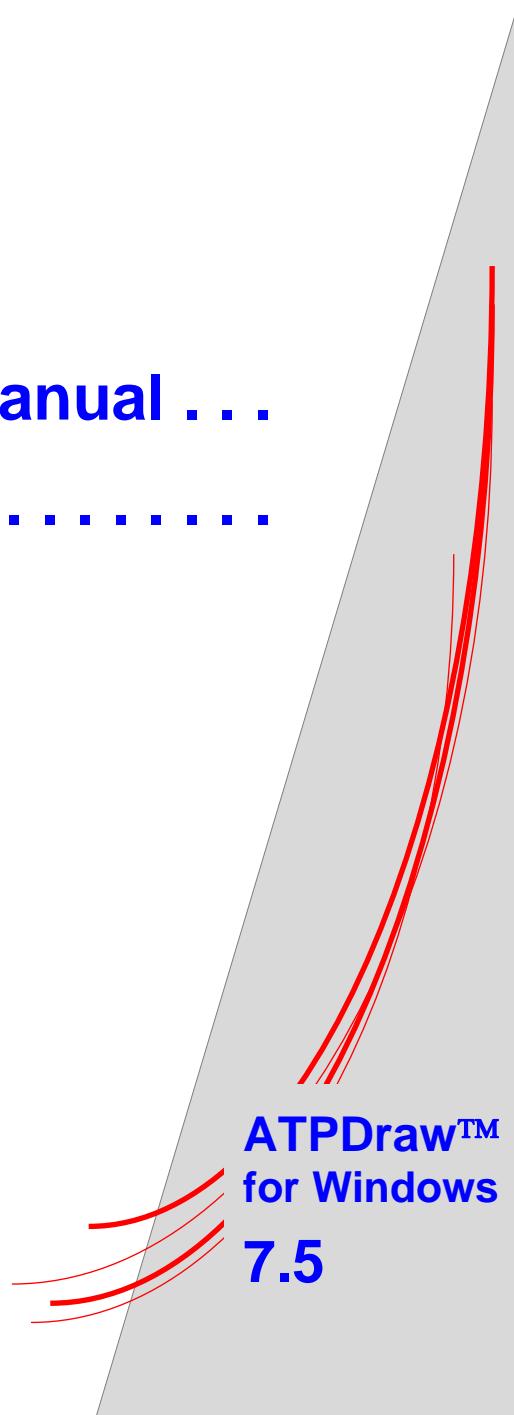


Fig. 3.48 – Embedded Plot dialog, native advanced settings dialog.

## 4. Reference Manual . . .





This part of the manual outlines all menu items and program options, and gives an overview of the supported ATP components, TACS, and MODELS features.

ATPDraw has a standard Windows user interface. The *Main window* of the program is shown in Fig. 4.1. The *Main menu*, the *Circuit window* and the *Component selection menu* are the most important items of that window. Elements of the *Main menu* and supported ATP components in the *Component selection menu* will be referenced in this part of the manual.

## 4.1 Main window

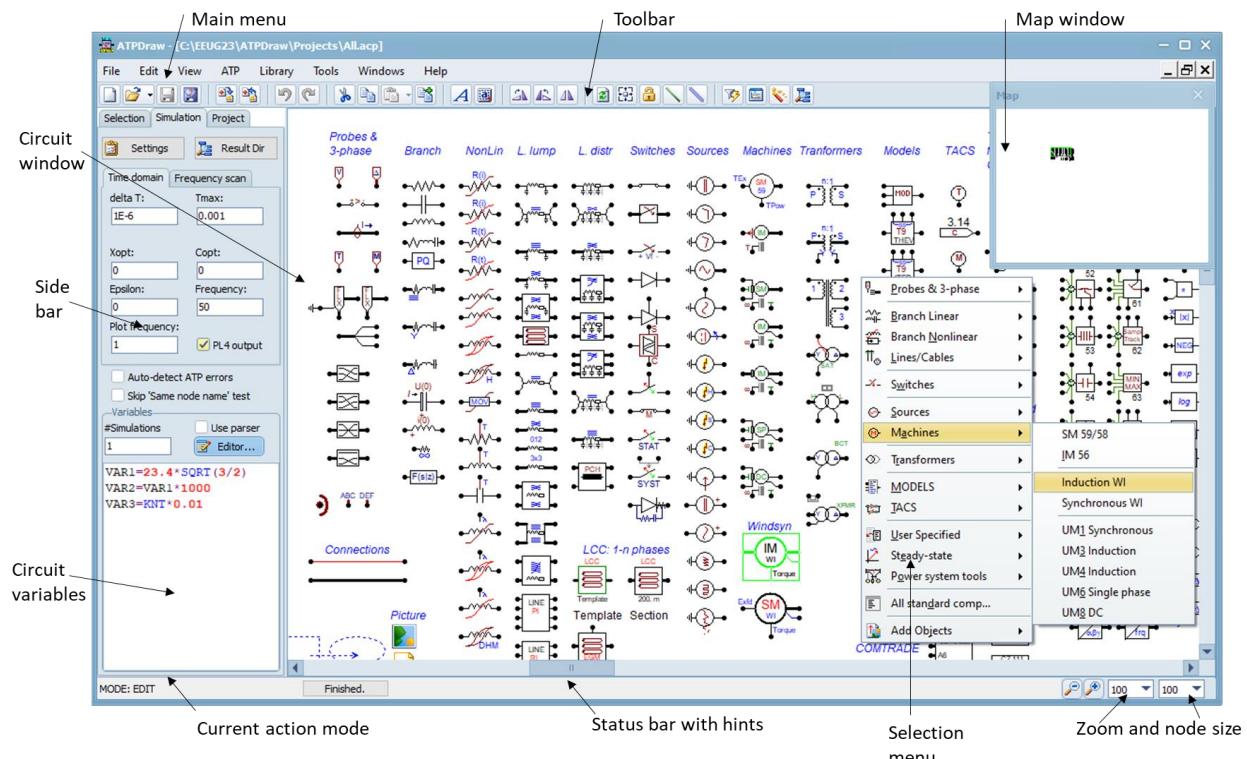
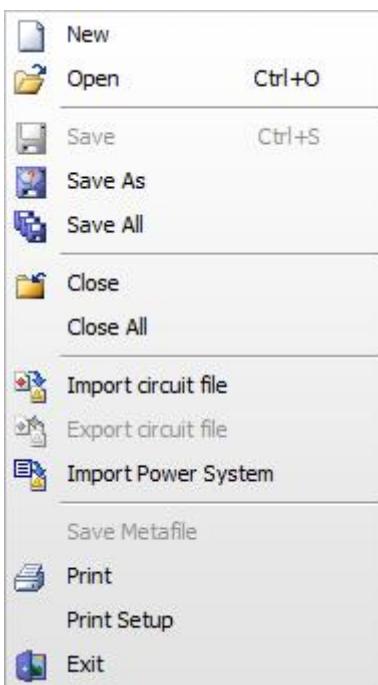


Fig. 4.1 - Components of ATPDraw's main window.

If you are unfamiliar with the use of ATPDraw, read the Introductory Manual to learn how to create a circuit or the Advanced Manual to learn how to create a new object in ATPDraw. The Introductory Manual starts with the explanation of operating windows and the mouse in ATPDraw, and shows how to build up a circuit and how to create an ATP-file to be used as input for a subsequent transient simulation.

## 4.2 Main menu

### 4.2.1 File



This field contains actions for input/output of ATPDraw projects. Selecting the *File* item in the main menu will result in a popup menu shown in Fig. 4.2.

Fig. 4.2 - File menu.

#### 4.2.1.1 New

Selecting this menu item will open a new empty *Circuit window*. ATPDraw supports to work on several circuits simultaneously and copy information between the circuits. The number of simultaneous open windows is limited only by the available MS-Windows resources. The circuit window is much larger than the actual screen, as it is indicated by the scroll bars of each circuit windows.

#### 4.2.1.2 Open

This menu performs a Windows standard Open dialog box. In this window the user can select a project file and load it into ATPDraw. Short key: *Ctrl+O*. The default directory is the previously used directory and the first time the dialog is used the *Project Folder* set under *Tools/Options/Files&Folders* (initially read from the ATPDraw.ini file) is suggested.

ATPDraw can read both circuit (.cir) files created by an earlier version of the program and project files (.acp and .adp). When opening a project file all data are stored in memory and no files are written to disk. The circuit files and project files are binary data files.

The Open/Save dialog box is used for several different selections in the main menu. An alternative MS-Windows 3.1 style is also supported. There is a check box in the *Tools / Options / General* tab to switch between the two supported alternatives.

#### 4.2.1.3 Save

Activating this menu item will save the project in the active circuit window into a disk file. If the name Noname.acp is shown in the circuit window a *Save As* dialog box will be performed, where the user can specify a new name for the current project file name. Short key: *Ctrl+S*.

#### 4.2.1.4 Save As

The project in the active circuit window is saved to disk under a new name. The name of the file can be specified in the *Save As* dialog, which is similar to the *Open Project*. This command allows the user to save the project under a name other than that is already used. ATPDraw can read circuit files (.cir) created by earlier program versions, but the *Save As* command supports only the newest file format. The default extension of the project files on disk is (.acp).

#### 4.2.1.5 Save All

Saves all modified projects to disk under their own project file names. If one or more open projects still have not got a name (Noname.acp), it will be requested in a *Save As* dialog boxes successively.

#### 4.2.1.6 Close

Close the active circuit window. If any changes to the circuit have not been saved yet, the user will be warned as shown in Fig. 4.3 to confirm before the circuit is closed. If the project has been modified, the user is given a chance to save it first.

#### 4.2.1.7 Close All



Close all circuit windows. If a project has been modified since the last save operation, a confirmation dialog will be prompted giving a chance for the user to save it first.

of unsaved project data.

Fig. 4.3 - Confirmation prevents the loss

#### 4.2.1.8 Import circuit file

This command inserts a circuit from disk file into the *active* circuit window contrary to the *Open* command, which loads the circuit into a *new* circuit window. Selecting this menu will result in an *Import Project* dialog box where the user can select the file to load. The imported circuit appears in the circuit window as a group in marked moveable mode. Existing node names will be kept or rejected upon the selection of the user.

#### 4.2.1.9 Export circuit file

Save the selected objects of the active circuit to a disk file. Same as *Save As*, but only the selected objects (marked by a rectangular or polygon area) of the circuit are written to the disk file.

#### 4.2.1.10 Import power system

Opens up the Import Power System Dialog where a text file describing the power system can be imported.

#### 4.2.1.11 Save Metafile

Write the selected objects of the active circuit to a disk file in Windows metafile (.wmf) format. If no objects are selected, the entire circuit window content is written to disk. This way even graphics of large circuits can be exported to other applications without loss of resolution seen on

the screen when the *Zoom* option is used to fit the circuit to the screen size. Metafiles created by this command can be imported as picture into other applications (like MS-Word or WordPerfect) having filter available for this format.

#### 4.2.1.12 Print

Print the graphics on the currently selected printer.

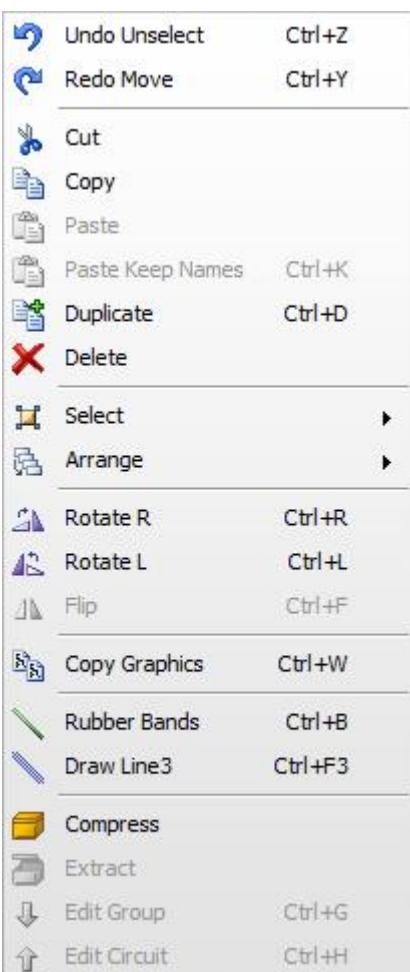
#### 4.2.1.13 Printer Setup

Select and setup the printer.

#### 4.2.1.14 Exit

This command closes all open circuit windows of ATPDraw. User will be asked to save any modified circuits before the application is terminated.

### 4.2.2 Edit



This menu contains the various edit facilities of circuit objects in ATPDraw. The *Edit* popup menu is shown in Fig. 4.4.

An object or group of objects must be selected before any edit operation can be performed on them. If the user clicks on an object with the left mouse button in the circuit window the icon of the object will be enclosed by a lime colored frame indicating that it is selected.

Fig. 4.4 - Edit menu.

#### 4.2.2.1 Undo/Redo

The *Undo* command cancels the last edit operation. The *Redo* cancels the last undo command. Short key for Undo/Redo: *Ctrl+Z* and *Ctrl+Y*. The number of undo/redo operations depends on the Undo/redo buffers: setting on the *Preferences* tab of the *Tools / Options* menu. Default value is 10. Almost all object manipulation functions (object create, edit, delete, move, rotate, etc.) can

be undone (or redone). Changes made to the circuit data in the component dialog box are also supported by the Undo/redo functions (this included also the extensive data in LCC, BCTRAN, XFMR). These functions also update the circuit's *Modified* state in the status bar to indicate that the circuit has been modified. During an undo operation, the modified state is reset its previous value. After *Save/Save As* the Undo/Redo buffer is cleared.

#### **4.2.2.2 Cut**

Copies the selected objects to the Windows clipboard and deletes them from the circuit window. The objects can later be pasted into the same or other circuit windows, or even other instances of ATPDraw. Short key: *Ctrl+X*.

#### **4.2.2.3 Copy**

The selected objects are copied to the clipboard. Short key: *Ctrl+C*. A single marked object or a group of objects can be copied to the clipboard. This command unselects the selected objects.

#### **4.2.2.4 Paste**

The contents of the clipboard are pasted into the current circuit when this menu item is selected. Short key: *Ctrl+V*. The pasted object or objects appear in the current window in marked moveable mode. The node names are deleted when pasting components.

#### **4.2.2.5 Paste Keep names**

Paste the content of the clipboard into the circuit, but keeps all node names. This can be useful in special situations when copying elements between different circuits. But should never be used when copying components from and to the same circuit, otherwise annoying duplicate node names warnings will appear.

#### **4.2.2.6 Duplicate**

Copies the selected object or a group of objects to the clipboard and then duplicates them in the current circuit window. Duplicated objects appear in the current window in marked moveable mode. Short key: *Ctrl+D*.

#### **4.2.2.7 Delete**

Selected objects are removed the from the circuit window. Short key: *Del*.

#### **4.2.2.8 Copy Graphics**

The selected objects are copied to the clipboard in Windows Metafile format. This way graphics of selected objects can be exported to other Windows applications. Short key: *Ctrl +W*.

#### **4.2.2.9 Select**

This menu has five sub-menus:

**None**: To cancels the object selection. Short key: *Ctrl +N*.

**All**: Select all objects in the current circuit window. Short key: *Ctrl +A*.

**Inside**: Enables object selection by a polygon shaped region. Short key: *Ctrl +I* (or double-click with the left button in an empty region of the circuit window).

**by Properties**: Enables selection by objects' support file name or order number (see below). Short key: *Ctrl +P*.

**Overlapped**: Select component that overlap other components. First *ATP/run ATP* must be choosed to identify overlapping component.

A selected object or group of objects can be subject of the most editing operations: *Move* (click left button, hold down and drag), *Rotate/Copy/Duplicate/Delete* or *Export* (in the *File* menu). To unselect a group, select *None*, or just click with the left mouse button in an empty space of the circuit window.

In *Inside* mode, the mouse cursor icon changes its style to a pointing hand and moves to the middle of the circuit window. The current action mode also changes to *MODE:GROUP* in the status bar. To draw a polygon around a group of objects move the cursor to the starting location and click the left mouse button. Then release the button and a rubber band line will be drawn between the starting point and the current mouse cursor location. And so forth: left click to create corners, right to complete the polygon. All objects with midpoint inside or connections with both endpoints inside the polygon will be included in the selection.

In the *by Properties* selection mode the group of components can be selected by their type and/or *Order* number. The type here is the name of the support file and the Order number is the identifier specified in the component dialog box.



The available component Names and Order numbers are listed in two combo boxes as shown in Fig. 4.5. When you click on *OK* the components with the selected order number and/or support file name become selected. Then all kinds of edit operation can be performed on the group (copy/paste, copy graphics, rotate, edit, grouping etc.).

Fig. 4.5 - Selecting objects by name or group no.

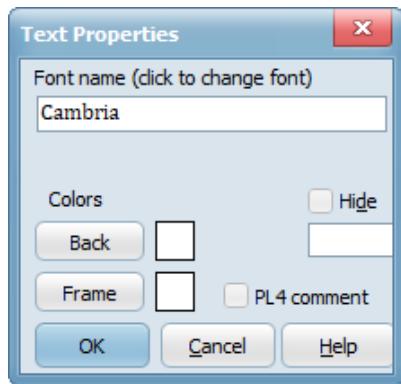
#### 4.2.2.10 Arrange

This menu has six sub-menus. All are related to the order of the objects. A component in front is prioritized when clicking and comes first in the ATP file:

- Bring forward:** Sends the selected object one step forward.
- Send backward:** Sends the selected object one step backward.
- Bring to front:** Sends the selected object to the front.
- Send to back:** Sends the selected object to the back.
- Send connections back:** Send all connections back. Connections not prioritized when clicking on nodes.
- Sort all Models:** Sorts the Models so that models used by other comes first.

Using the Arrange carefully, sorting by Order can be avoided. The Sidebar/Project contains a tree view of the circuit structure and allowed sorting as well by dragging objects.

#### 4.2.2.11 Edit Text



This menu is used to insert a new circuit text. In addition, the selection of texts, component labels or node names is favored in this mode. An alternative to this last property is to press the *Alt* key. This is beneficial when texts, labels or node names are drawn overlapped by components. If you click on existing texts, labels or node names you can edit the text directly on screen or move them (click and hold). Short key: *Ctrl+T*.

Fig. 4.6 – The circuit text dialog box. It appears after a right click (or left double) on a circuit text.

Selecting the *Edit Text* menu item, the mouse cursor style will change to a pointing hand and forced to stay within the circuit window. The action mode indicator in the status bar will also change to *MODE: EDIT TEXT*. You can leave this mode by pressing the *ESC* key.

#### 4.2.2.12 Rotate R/L

This command rotates the selected object(s) 90 degrees clockwise (R) counter-clockwise (L). The operation Rotate R can also be performed by clicking the right mouse button inside the selected group. Short key: *Ctrl + R/L*.

#### 4.2.2.13 Flip

Mirrors the icon left to right. For vector icons the texts are not flipped. This option is useful for instance for transformers since the primary and secondary node will be swapped. Short cut *Ctrl+F*.

#### 4.2.2.14 Copy Graphics

Copy the selected graphical content to the Windows clipboard in MetaFile format.

#### 4.2.2.15 Rubber Bands

If this option is checked, connections with one endpoint inside a selected region and one outside are treated as a rubber band between the selected group and the rest of the circuit. Short key: *Ctrl + B*. This command does not work for short cut single component selections: e.g. left click on several components while the *Shift* key is pressed, because this way no connections are selected.

#### 4.2.2.16 Draw Line 3

If this option is selected, LINE3 components are drawn instead of Connections. LINE3 components are 3-phase with sequential data input in either lumped or distributed parameter models. Various CB, CT and fault options are supported.

#### 4.2.2.17 Compress

This command *Compress* will replace a group of selected objects with a single icon having user selectable external data and nodes. ATPDraw supports real grouping or single icon replacement of sub-groups in an unlimited numbers of levels. The *Compress dialog* box (see Fig. 4.7/a) appears where the user designs the new group object. The user can later modify a compressed group by selecting it and click *Compress* once more.

In the Compress dialog the user can specify the external data and nodes of the Group object (Parent) and how group content (Children) will inherit this. A nonlinear characteristic can also be selected as external data. Only the members of the group are shown in the Compress process and moved to the middle of the circuit window with the Compress dialog as a stay-on-top window.

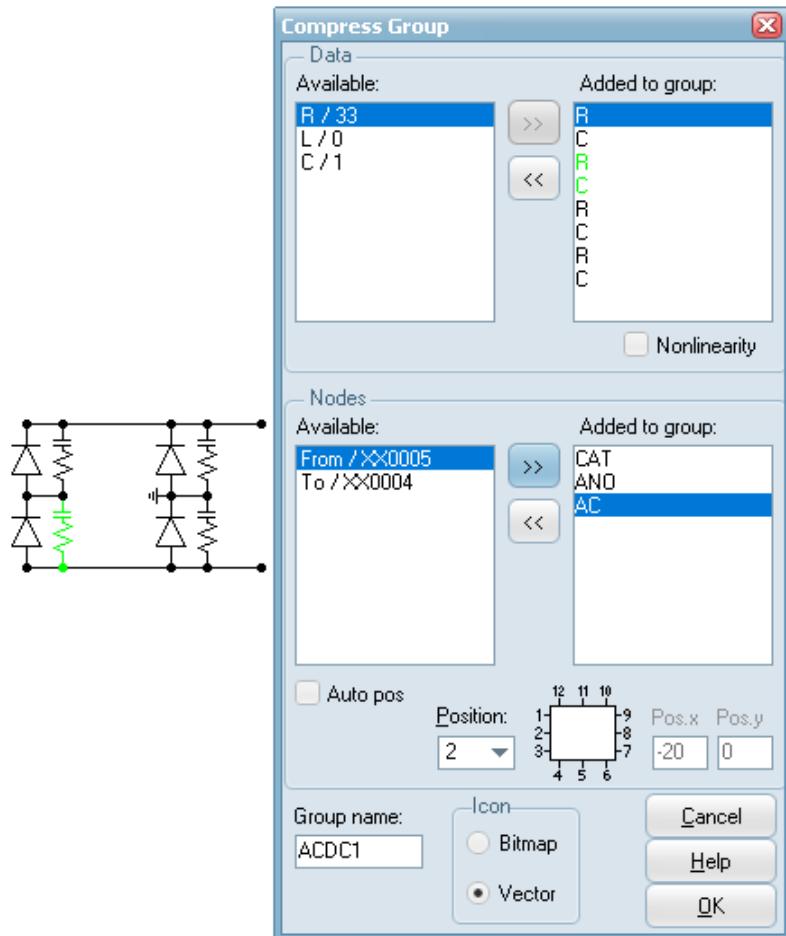


Fig. 4.7/a - The Compress dialog.

The user must first select a component in the circuit window. It is then drawn in a lime color and its data and nodes appear under *Available*:. Here the user can select a parameter and click on the >> button to transfer it to the *Added to group*: list. If the >> button is disabled it means that the data/node already is in *Added to group*: list and shown there with a lime color. Selected node in the *Available* list will be drawn in a lime color in the circuit window. All data and nodes listed in the *Added to group*: will be an external attribute of the new group object. The selected external nodes are drawn enclosed by a red circle. The positions of the external nodes are selected in the *Position* combo box. Positions 1-12 will be on the traditional border as shown in the graphic below, while position 0 will enable the user to specify positions in the *Pos.x* and *Pos.y* fields (in increments of 10 pixels, gridsnap). You can change the *Added to group*: names by double clicking on them. Data with the same name are treated as a single data in the component dialog box (Fig. 4.7/b). Selected data and nodes can also be removed from the *Added to group*: by clicking on the << button. The *Keep icon* check box can be used when Recompressing a group in cases where the user wants to keep its icon.

When you later open the component dialog box of the group-object, the selected data values and node parameters will appear as input possibilities. The values will automatically be transferred to the group members as shown in Fig. 4.7/b. Note that the 8 selected data are represented by two

external data in Fig. 4.7/b since the names are duplicated.

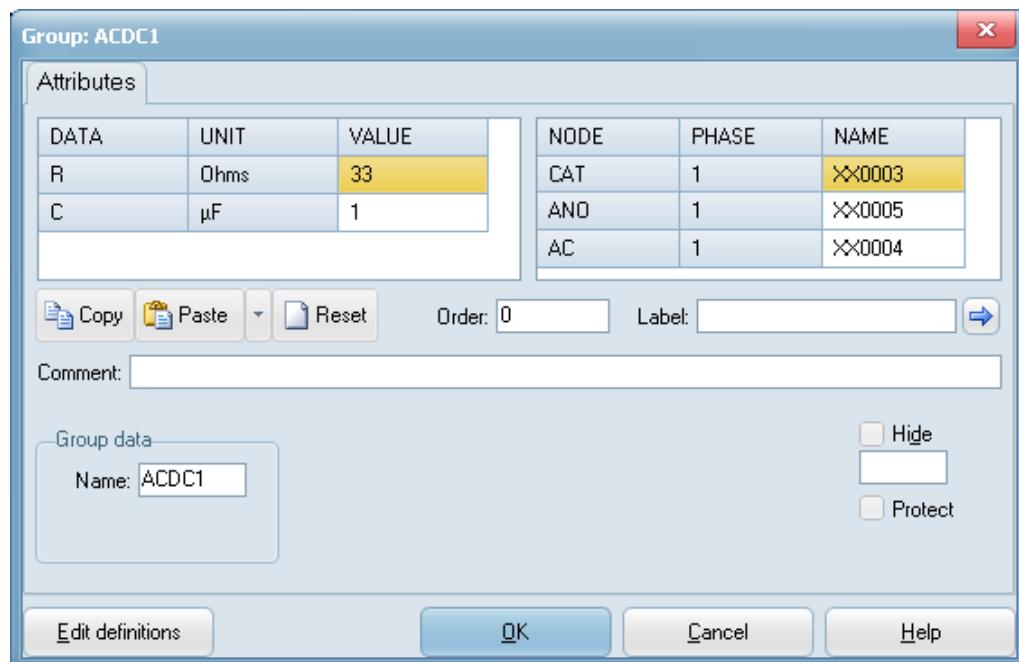


Fig. 4.7/b - Component dialog box for a sub-group object.

#### 4.2.2.18 Extract

This is the reverse operation of *Compress*. The group is extracted on the current circuit layer. To perform the operation, a compressed group (and only one!) must be selected first.

#### 4.2.2.19 Edit Group

This command shows the group content. Short key: *Ctrl+G*. The group is shown in a separate window. To perform the operation a compressed group (and only one!) must be selected first. It is possible to edit the group in a normal way, except deletion of the reference components. I.e. components having been referenced in one of the *Added to group:* lists cannot be deleted. If the user tries a "*Marked objects are referenced by compressed group...*" warning message appears.

#### 4.2.2.20 Edit Circuit

Displays the circuit to which the current group belongs. Short key: *Ctrl + H*. Actually, the grouping structure can be taken as a multi-layer circuit, where the *Edit Group* brings the user one step down in details, while *Edit Circuit* brings one step back. The group object (single icon replacement of objects) acts as the connection between the layers and transfers data between them.

### 4.2.3 View

This menu provides options for displaying and controlling the visibility of user interface and circuit window objects. The menu items are shown in Fig. 4.8.

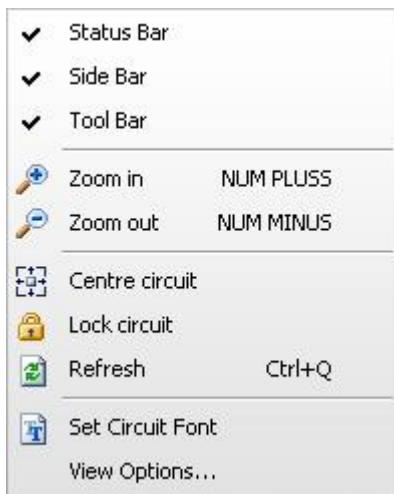


Fig. 4.8 - View menu.

#### 4.2.3.1 Status Bar

Status bar on/off at the bottom of the main window. The status bar displays status information about the active circuit window. The mode field on the left hand side shows which mode of operation is active at present. Possible modes are:

<i>EDIT</i>	The normal mode.
<i>EDIT TEXT</i>	Indicates that text editing is preferred. Hold down the Alt key to enter this mode of operation or select <i>Edit Text</i> from the <i>Edit</i> menu. Click left in an empty space to add a new text. Click the left mouse button on an existing text (circuit text, label, node name) to edit it directly on screen. Click left, hold down and drag to move it to a new position. If the text is overlapped by a component icon, this mode of operation is required to access the text.
<i>DRAW</i>	Mode when adding Shapes to the circuit ( <i>LINE</i> , <i>RECTANGLE</i> , <i>ELLIPSE</i> , <i>ARROW</i> ). To cancel drawing relation, click the right mouse button or press the <i>Esc</i> key.
<i>COMPRESS</i>	Mode when objects are selected and <i>Edit/Compress</i> is clicked. In this mode only the selected objects are shown with the Compress dialog on top.

The field to the right of the mode field displays the modified status of the active circuit. As soon as you alter the circuit (moving a label, deleting a connection, inserting a new component, etc.), the text *Modified* will show up to indicate that the circuit needs saving. The field will be empty when you save the circuit or undo all modifications. Note that the number of available undo buffers is limited (default value is 10 but can be increased on the *Preferences* tab of the *Tools / Options* menu). In the default case, if more than 10 modifications are done, the field will indicate a modified status until you save the circuit.

The next rightmost field of the status bar displays the menu option hints and Drag-over information. To the very right of the status bar comes items for controlling the zoom and the node sizes.

#### 4.2.3.2 Side bar

This bar to the left has three pages. The default Simulation page contains frequent simulation settings and variables besides some useful tools. The Selection page contains a tree structure for selection of all components. When selecting a component in the Selection page, a new component will be created in the middle of the circuit window. The Simulation page contains useful and frequent ATP settings and a list of Variables used. The user can right-click in the Variables grid to

sort or copy/paste/delete variables. When *Show values* is checked an extra column appears together with a panel for parsing the variables. Clicking on or the Up-Down arrows will execute the script with simulation number KNT (this has some relevance for object with Hide linked to Variables). The *Project page* contains some project properties in the *Documentation* part and a tree structure with all objects in the active circuit in the *Objects* part. Click on *Update* to refresh the object tree after circuit window edit operations and *Filter* to show only objects of interest. The *Conn* button will bring all connections in the back (similar behavior as in pre v7.0) Left-click on an object will center and highlight it in the circuit window, right-click will open it, and left click and hold will enable to drag and drop it in the list for ordering (not legal to move object in or out of groups). Objects first in the list will be prioritized in mouse clicks and comes first in the ATP-file. Components are marked with a symbol indicating branch, switch, source, transformer, machine, tacs, models. The Group component is marked with a box symbol with a list of Children (group content). Connections, Texts, Shapes, Pictures and Files are other types of circuit objects.

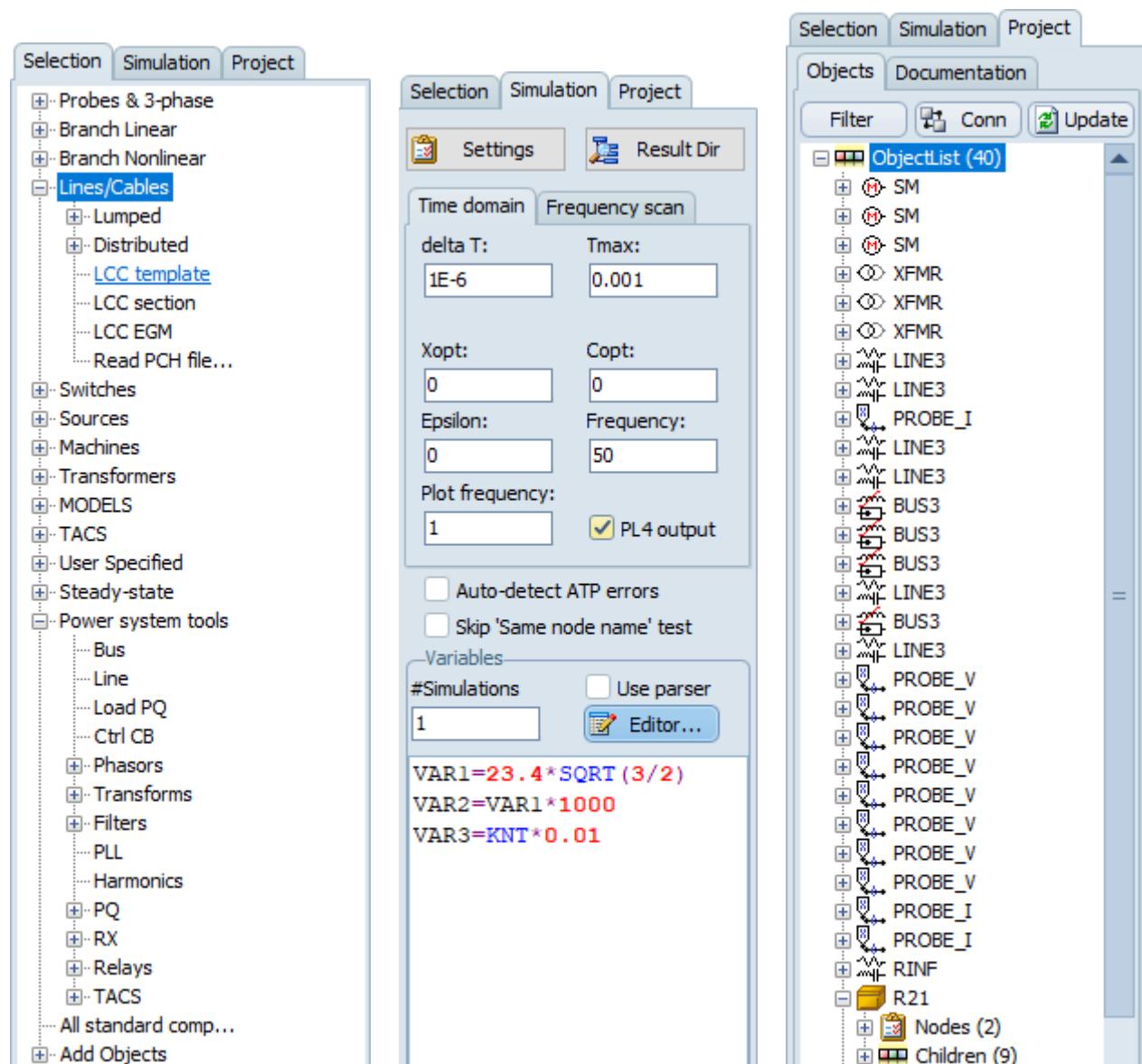


Fig. 4.9 – Side bar pages.

### 4.2.3.3 Toolbar

The standard toolbar is:



Fig. 4.10 – Toolbar.

From the left the tools are:

Item Menu	Shortcut	Description
New File New Open	--	Open an empty circuit file.
File Open	CTRL+O	Loads a circuit file into a new window. Contains also a dropdown with the five recent opened projects.
Save File Save	CTRL+S	Saves the active circuit window to the current project file.
Save As File Save As	--	Saves the active circuit window to a new project file.
Import File Import	--	Inserts a stored circuit into the current circuit.
Export File Export	--	Export the selected circuit to an external project file.
Undo Edit Undo	CTRL+Z	Undo the previous operation.
Redo Edit Redo	CTRL+Y	Redo the previous undo operation.
Cut Edit Cut	CTRL+X	Copy the current selected circuit to the clipboard and then delete it.
Copy Edit Copy	CTRL+C	Copy the current selected circuit to the clipboard.
Paste Edit Paste	CTRL+V	Paste the ATPDraw-content from the clipboard into the circuit.
Edit Duplicate	CTRL+D	Copy+Paste.
Edit Edit text	CTRL+T	Go into Edit text mode for adding and selecting text. Required to add new text to the circuit window.
Edit Select>All	CTRL+A	Select the entire circuit.
Edit Rotate-R	CTRL+R	Rotate 90 deg. clock-wise.
Edit Rotate-L	CTRL+L	Rotate 90 deg. counter clock-wise.
Edit Flip	CTRL+F	Flip left-to-right. The nodes changes position. Vector text is not flipped.
View Refresh	CTRL+Q	Redraw circuit.
View Centre circuit	--	Centre the circuit in the circuit window
View Lock circuit	--	Turn on “child safety”, prevent edit operations except for input.
Edit Rubber band	Ctrl+B	When selecting components, connections (and LINE3) stretches automatically.
Edit Draw LINE3	--	When check, LINE3 components are drawn instead of Connections.
ATP run ATP	F2	Make node names + write the ATP file+ run ATP by executing the ATP command (ATP Connection Wizard F10).
ATP run Plot	F8	Plot Executed the Plot Command (ATP Connection Wizard F10) and send the current PL4 file as parameter.
ATP Setup ATP..	F10	Open the ATP Connection Wizard to select Solver, Execution mode and Plotter.
Result Dir	--	Select the Result Directory/Folder (all output goes here, ATP, LIB, LIS, PL4). Default set in ATP Connection Wizard (F10) and also with ATP/Sub-process/Make ATP file.

### 4.2.4 Zoom In

Enlarges the objects in the active circuit window by increasing the current zoom factor by 20 percent. Short key: + (*plus sign on the numeric keypad or “=+” alphanumeric key*).

#### 4.2.4.1 Zoom Out

Reduces the icon size in the active circuit window by 20 percent. Short key: - (*minus sign on the numeric keypad or the “-/” alphanumeric key*).

#### 4.2.4.2 Centre circuit

Centers the circuit in the circuit window.

#### 4.2.4.3 Lock circuit

Turns on “child safety”, prevents edit operations except for input.

#### 4.2.4.4 Refresh

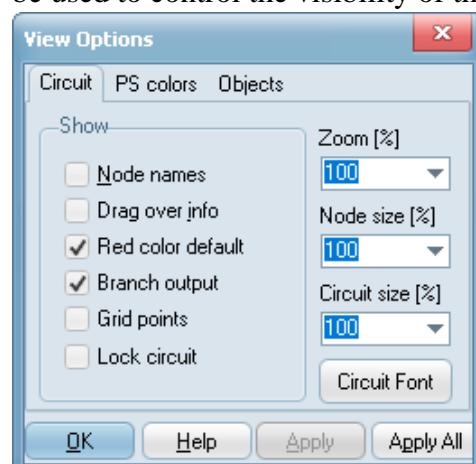
This command redraws all objects in the active circuit window. Short key: *Ctrl+Q*. This command can also be activated by clicking the Toolbar icon: 

#### 4.2.4.5 Set Circuit Font

Enables you to select a font type and size for the node names and labels on the screen (and also for the metafile export). The default font is MS Sans Serif, regular, 8 pt size. This also becomes the default font for circuit text, but this can be adjusted individually. To get the angle  $\angle$  symbol used for phasors on screen, the Cambria font can be used instead.

#### 4.2.4.6 Options

Selecting this menu item will bring up the *View Options* dialog box. The *View Options* dialog can be used to control the visibility of the objects in the active circuit window.



The options dialog consists of three pages. *Circuit* controls circuit appearance and size. *PS colors* gives the colors used in the power systems components (LINE3, BUS3) and Connections. *Objects* turns on/off classes of circuit objects, but this is rarely useful. By default, all objects are visible.

Fig. 4.11 - View Options dialog box.

#### *Circuit*

The meaning of options assumed checked ( are listed below:

*Node names* Node names are visible on the screen (overrides the *Display* attribute of the Node data window). This option is useful after a *Make Names* selection in the *ATP* menu.

*Drag over info* List information about the component (name, number of data and nodes) under the mouse cursor. No clicking is required. Can slow down the application in case of large circuits.

*Red color default* Components and node dots are drawn with a red color until the component or node is opened for the first time.

*Show branch output* Small U/I symbols indicate the selected branch output requests. Branch output requests can be specified in most of the component dialog boxes.

*Lock circuit* Components cannot be selected and moved only opened for input.

#### *PS Colors*

Specify 6 voltage ranges and associate color.

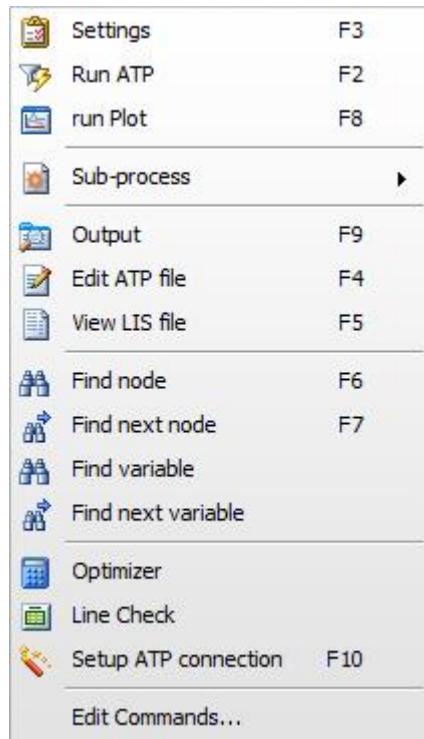
#### *Objects*

Turn on/off classes of objects. The meaning of options assumed checked ( are listed below:

<i>Components</i>	All standard and user specified components are displayed.
<i>Connections</i>	All connections (short circuits between nodes) are displayed.
<i>Texts</i>	All Text objects are displayed
<i>Pictures</i>	All Picture objects are displayed
<i>Shapes</i>	All Shape objects (line, rectangle, ellipse and arrow) are displayed
<i>Probes</i>	All probes are displayed
<i>Labels</i>	Component labels are displayed on the screen.

To accept the current view options and return from the dialog, select the *OK* button. To set and view new options without returning, select the *Apply* button. If you want the current settings be applied to all current and future circuit windows, select the *Apply All* button before you exit the dialog box (this saves the selections to the ATPDraw.ini file).

#### 4.2.5 ATP



The *ATP* menu provides options to create, display and modify the ATP input files and to set circuit specific ATP options (e.g.  $\Delta T$ ,  $T_{max}$ ) before running the case by the *run ATP* command or the *F2* function key. From this menu all output requests can be managed and the ATP and LIS files edited and inspected. The *Find node* and *Find next node* navigation tool is also available here. The Optimization module works with a cost function and perform multiple ATP runs. The *Line Check* feature calculate sequential parameters of transmission lines and sub-circuits. Other components of the ATP-EMTP package (e.g. pre- and post-processors, supporting programs and utilities) can also be launched from this menu. Besides the default commands, the user can add additional commands (e.g. *Run PlotXY / Run Analyzer / Run PCPlot / Run TPPPlot, etc.*) to the existing program items, which are listed immediately below the *Edit commands...* as shown in Fig. 4.12.

Fig. 4.12 - The ATP menu.

##### 4.2.5.1 Settings

In the *ATP Settings...* dialog box several options for the active circuit window can be specified. These settings are used when ATPDraw generates the ATP input file. Options are sorted in six tabs, such as the *Simulation* and *Output* for the miscellaneous data card settings, *Format* for specification of data-card sorting options and miscellaneous request, *Switch/UM* for statistical and Universal Machine studies, and *Variables* for specification of global \$Parameter and Pocket Calculator options.

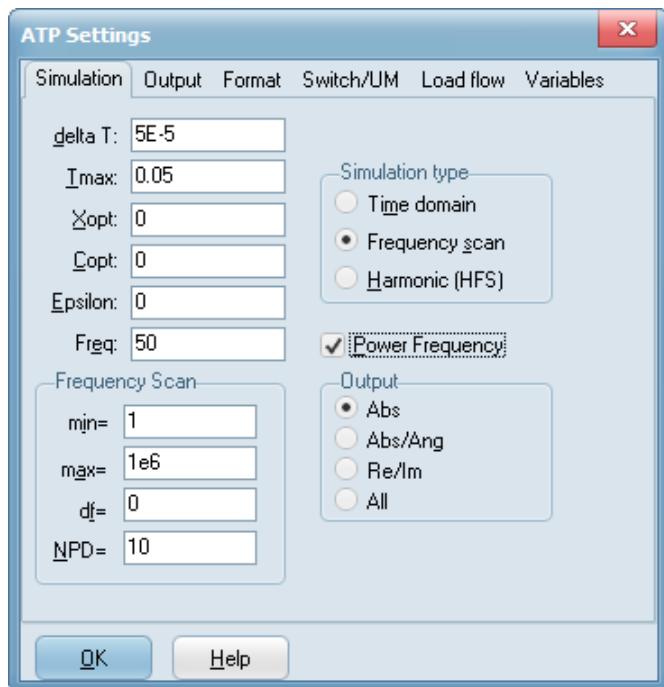


Fig. 4.13 - Simulation settings.

### Simulation settings

**Simulation type:** Select between the simulation methods supported by ATP:

- Time domain
- Frequency scan
- Harmonic Frequency Scan (HFS)

#### Time domain

**delta T:** Time step of simulation in seconds.

**Tmax:** End time of the simulation in seconds.

**Xopt:** Inductances in [mH] if zero; otherwise, inductances in [Ohm] with **Xopt** as frequency

**Copt:** Capacitances in [mF] if zero; otherwise, capacitances in [Ohm] with **Copt** as frequency.

**Epsilon:** Sensitivity in singularity check. Set to 1E-12 or less. Zero gives default value from STARTUP-file.

**Freq:** System frequency in Hz.

**Power Frequency:** when checked the SYSTEM FREQUENCY request card is written in the ATP-file. The ideal transformer component uses this frequency.

#### Frequency scan

If Frequency scan is selected the FREQUENCY SCAN option of ATP is enabled.

**min:** Starting frequency for the frequency scan

**max:** Ending frequency for the frequency scan

**df:** Frequency increment. Leave 0 for logarithmic frequency scale

**NPD:** Number of frequency points per decade in logarithmic scan

#### Harmonic Frequency Scan (HFS)

Selecting HFS will run the ATP data case so many times as specified in the Harmonic source component dialog box (see chapter 4.15.12). The frequency of the harmonic source will for each ATP run be incremented. The power frequency specification is mandatory for HFS simulations.

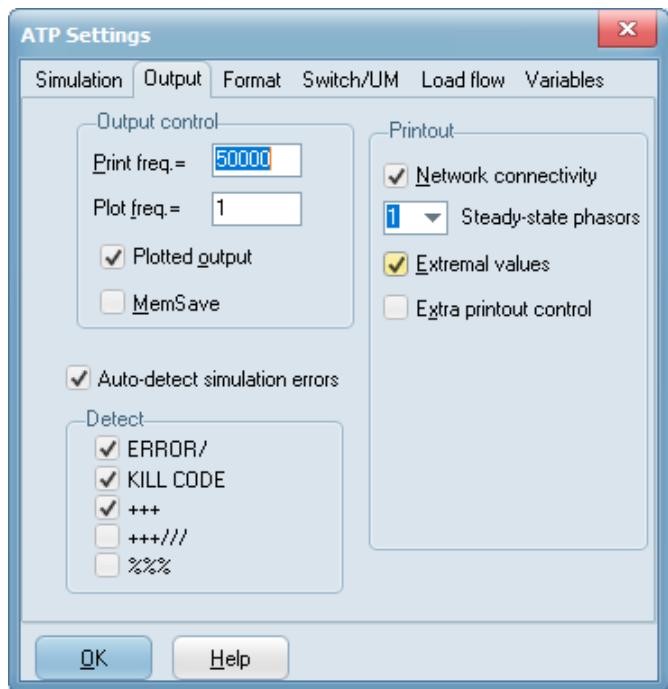
If Frequency scan or HFS is selected the user must specify which component of the solution to print out:

**Magnitude only:** Default request

**Magnitude & Angle:** Results are printed in POLAR

**Magnitude & Angle & Real/Imag:** Both POLAR and RECTANGULAR

**Real/Imag:** RECTANGULAR output request. Other combinations are illegal and are prevented by button logic.



## Output settings

### statistical

**Print freq.**: Frequency of LUNIT6 output within the time-step loop. For example, a value of 500 means that every 500<sup>th</sup> simulation time step will be printed to the LIS-file. This option controls ATP's 1<sup>st</sup> misc. data parameter IOUT

**Plot freq.**: Saving frequency of the simulation data to the .pl4 output file. A value of 5 means for example, that every fifth time step will be written to the PL4-file. This option controls ATP's 1<sup>st</sup> misc. data parameter IPLOT

Fig. 4.14 - Output request tab.

**Plotted output**: If checked ATPDraw sets the 1<sup>st</sup> misc. data parameter ICAT=1 in the ATP input file which results in a .pl4 output file.

**MemSave**: Controls the dumping of EMTP memory to disk at the end of simulation if START AGAIN request is specified. If checked indicates memory saving.

**Auto-detect simulation errors**: If this option is selected, ATPDraw will analyze the output LIS-file of ATP following the completion of the simulation. If the specified Detect string is found, the corresponding section of the file is displayed in a text editor window. This feature helps the user to recognize the simulation errors/warnings generated by ATP during the time step loop or input data interpretation. The string or strings, which makes this function work, are user selectable and activating at least "Error" and "Kill code" are highly recommended.

### Printout

**Network connectivity**: If >0 connectivity table (description of the topology of the circuit) is written to the LUNIT6 output file. This option controls ATP's 1<sup>st</sup> misc. data parameter IDOUBLE. If zero, no such table is written.

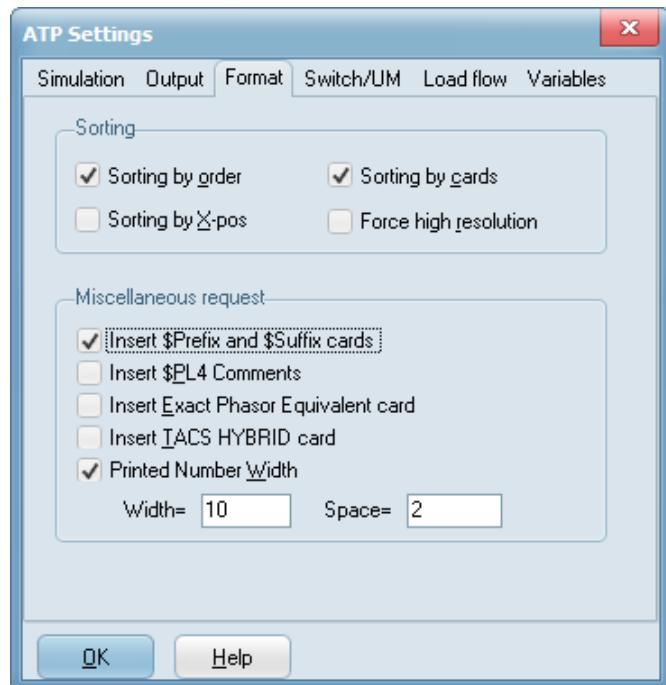
**Steady-state phasors**: If checked, complete steady state solution (branch flows, switch flows and source injection) is written to the LUNIT6 output file. This option sets ATP's 1<sup>st</sup> misc. data parameter KSSOUT=1. If unchecked, no such output is produced by ATP.

**Extremal values**: If checked, extrema of each output variables will be printed at the end of the LIS-file. This option controls ATP's 1<sup>st</sup> misc. data parameter MAXOUT. If unchecked, no such output is produced by ATP.

**Extra printout control**: Additional control for the frequency of LUNIT6 output within the time-step loop. If checked, the 1<sup>st</sup> misc. data parameter IPUN is set to -1 and a 2<sup>nd</sup> misc. data card will appear in the ATP input file. Parameters KCHG and MULT control the breakpoints and the new Print freq. value. If unchecked, IPUN is set to 0 and LUNIT6 printout frequency will be constant throughout the simulation.

## Format settings

The *Format* settings page contains four buttons for setting of ATP input file data format, a button for controlling the auto path generation and several other buttons for miscellaneous request cards. The *Additional* button supports the user to insert any request card or text strings in the ATP-file on precise location.



### Sorting

**Sorting by cards:** The sequence of ATP input data follows the default sequence of / data sorting cards (i.e. BRANCH cards are written first, followed by SWITCH cards and the SOURCE cards).

**Sorting by order:** The Order number that can be specified in the component dialog box for each object determines the sequence of cards. The lowest Order number comes first.

**Sorting by x-pos:** The leftmost object in the circuit window is written first.

Any combination of the three different sorting mechanisms can be specified.

**Force high resolution:** Use \$Vintage, 1 (if possible), for high precision data input.

Fig. 4.15 - ATP-file format settings.

### Miscellaneous request

**Insert \$Prefix and \$Suffix cards :** If this option is checked, ATPDraw will assume that all \$Include files (User Specified , LCC and external nonlinear characteristics) are located in the Result Directory and have the extension '.lib'. Two cards \$Prefix and \$Suffix will be inserted into the ATP file and the \$Include commands are specified without path and extension. This should be a preferred choice as this path and extension generally are used and that increased readability of the ATP file is obtained this way.

**Insert \$PL4 Comments:** If checked, ATPDraw writes the circuit comments in a \$BEGIN PL4 COMMENTS...\$END PL4 COMMENTS block. This may result in an error for some (older?) ATP versions.

**Insert Exact Phasor Equivalent card:** If checked, ATPDraw writes an EXACT PHASOR EQUIVALENT request in the ATP-file. This is recommended for Frequency Scan simulations including constant and distributed parameter overhead lines.

**Insert TACS HYBRID card:** Checking this button forces TACS HYBRID .. BLANK TACS to be written to the ATP-file. Useful when TACS objects are only present inside a *User Specified Object*.

**Printed Number width:** Enables the PRINTED NUMBER WIDTH request card, which controls the printout of the LUNIT6 device (output LIS-file). *Width:* is the total column width of printed output including blanks separating the columns. *Space:* is the number of blanks between columns of printed output.

## Switch/UM settings

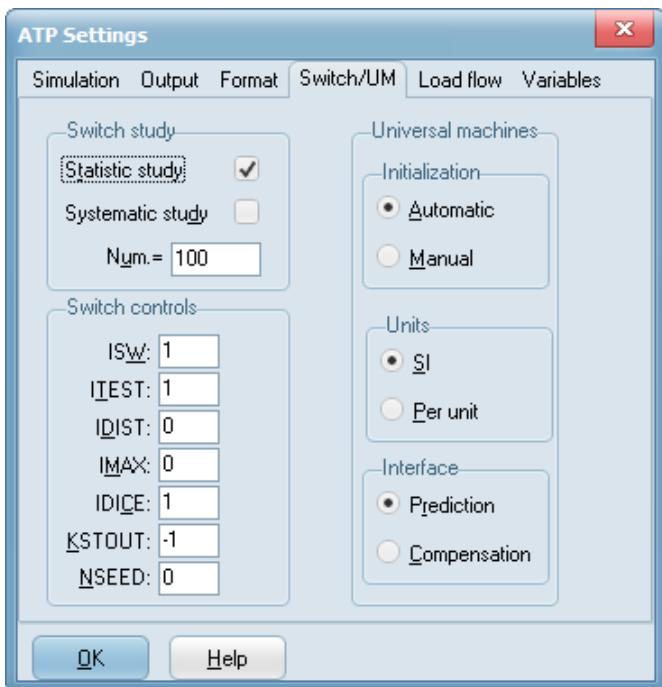


Fig. 4.16 - Switch/UM settings.

### Switch controls

**ISW:** If 1, printout of all switch closing/opening time appear in the output LIS-file. No such printout if the parameter is set to 0.

**ITEST:** Extra random delay using DEGMIN, DEGMAX and STATFR in STARTUP.

Possible values are:

- 0: Extra random delay for all switches.
- 1: No random delay.
- 2: Extra random time delay added to all closing switches.
- 3: Extra random time delay added to all opening switches.

**IDIST:** Select probability distribution function of subsequent switching operations. Zero means Gaussian distribution and 1 means uniform distribution.

**IMAX:** If 1, printout of extrema is written to the ATP output LIS-file for every energization. If 0 (zero), no such printout.

**IDICE:** Controls use of the random generator. A value of 0 implies computer-dependent random generator and a value of 1 means standard random generator.

**KSTOUT:** If 0, extra printed (LUNIT6) output for each energization. Output of the time-step loop and variable extrema (if *Extremal* values is selected on the *Output* tab) will be printed. If -1, no such output.

**NSEED:** Repeatable Monte-Carlo simulations. Possible values are:

- 0: Every simulation on the same data case will be different.
- 1: Same result each time the data case is run on the same computer.

### Universal machines

Here the user specifies the global data for the Universal electrical machine models in ATP. The selections here apply to all universal machines in the circuit.

**Initialization:** *Manual*: Terminal quantities of all machines must be specified.

*Automatic*: Initial conditions will be calculated by ATP. See section 9D1.5 for more details in the ATP Rule Book.

**Units:** Input variables are specified in *SI* units or *Per unit* (p.u.) quantities.

### Switch study

**Statistic study:** Study with statistic switches

**Systematic study:** Study with systematic switches

**Num:** Number of simulations. This value influences ATP's 1<sup>st</sup> misc. data parameter NENERG. ATPDraw sets the correct sign of NENERG: i.e. >0 for statistic or <0 for systematic switch studies.

**Interface:**

**Compensation:** The machine does appear to be a nonlinear element to the external network. Certain rules regarding connecting machines together must be followed. Inclusion of stub lines is often required. Preferred method.

**Prediction:** The machine does not appear to be a nonlinear element to the external network. This option is not available for single phase machines.

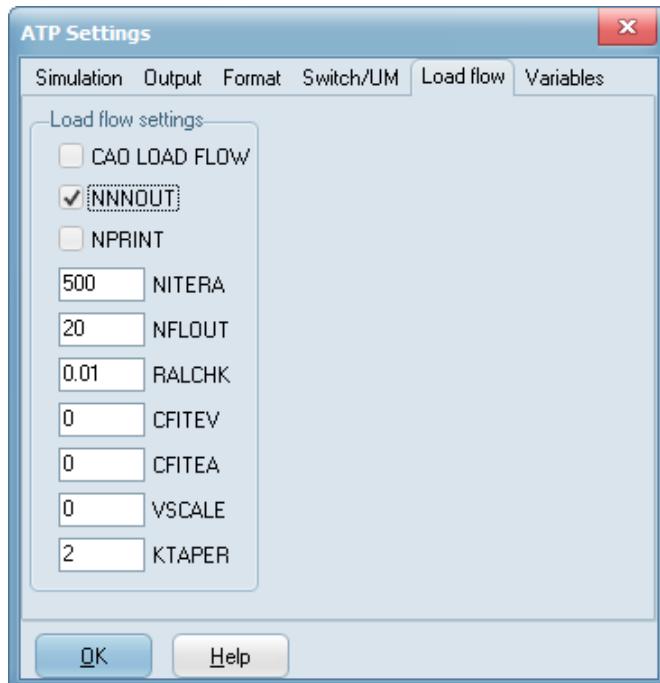
**Load flow**

Fig. 4.17 – Global load flow settings.

Sets the global variables of load flow according to RuleBook chapt. X.

**CAO LOAD FLOW** Does not work as intended.

**NNNOUT** Additional interactive output during load flow iteration.

**NPRINT** Tabular printout for nodes with power constraints.

**NITERA** Maximum number of iterations. Default 500.

**NFLOUT** Buffer size convergence monitoring, printout per line. Default 20.

**RALCHK** Relative convergence tolerance. Default 1/100.

**CFITEV** Acceleration factor ref.  $dQ/dU$ . Default 2/10.

**CFITEA** Acceleration factor ref.  $dP/dTh$ . Default 2.5.

**VSCALE** Voltage scaling factor. Use 1.4142 to get rms values output. Zero=Unity.

**KTAPER** =0: Constant acceleration factors. =2 used also in DC25/DC26 examples.

**Variablesvariables**

The *Variables* page supported originally only the \$PARAMETER feature of ATP-EMTP, but since v6 an *Internal Parser* option is added. Using variables can also be called parametrization or scripting and instead of a data value the user can specify a variable name and assign a value to it externally. The advantage is when data values are used in many different components, and when there is a need for frequent updates of some key parameters. In addition, Variables allow systematic variations of values with *Multiple Runs* as the default variable KNT is available as the simulation number. A key component compatible with *Multiple Runs* is the model WriteMaxMin which is designed to extract extremal values of a simulation as function of variables or the simulation number.

With the classical \$PARAMETER option, parametrization is managed inside ATP. This means that variables used in internal ATPDraw calculations cannot be assigned to variables. The data attribute *Params* is set to unity when parametrization is allowed. With the *Internal Parser* option, the variables get their values assigned before ATP execution in parallel threads and folders. The restrictions on parametrization of data is thus removed. Even data in LCC objects can be parametrized. The user is allowed to specify a text string (6 characters with *Internal Parser*, 5 otherwise) instead of a numerical value in the component dialog boxes as shown in Fig. 4.18 . If the *Internal Parser* is used or *Data.Params* is set to unity (under *Edit definitions*), the variable is

recognized and the user is asked if it should be added to the list of variables, otherwise the user is asked if the *Internal Parser* should be turned on. There is also a sanity check on variable names as they must consist of A-Z, 0..9 and ‘\_’ characters and cannot start with a number.

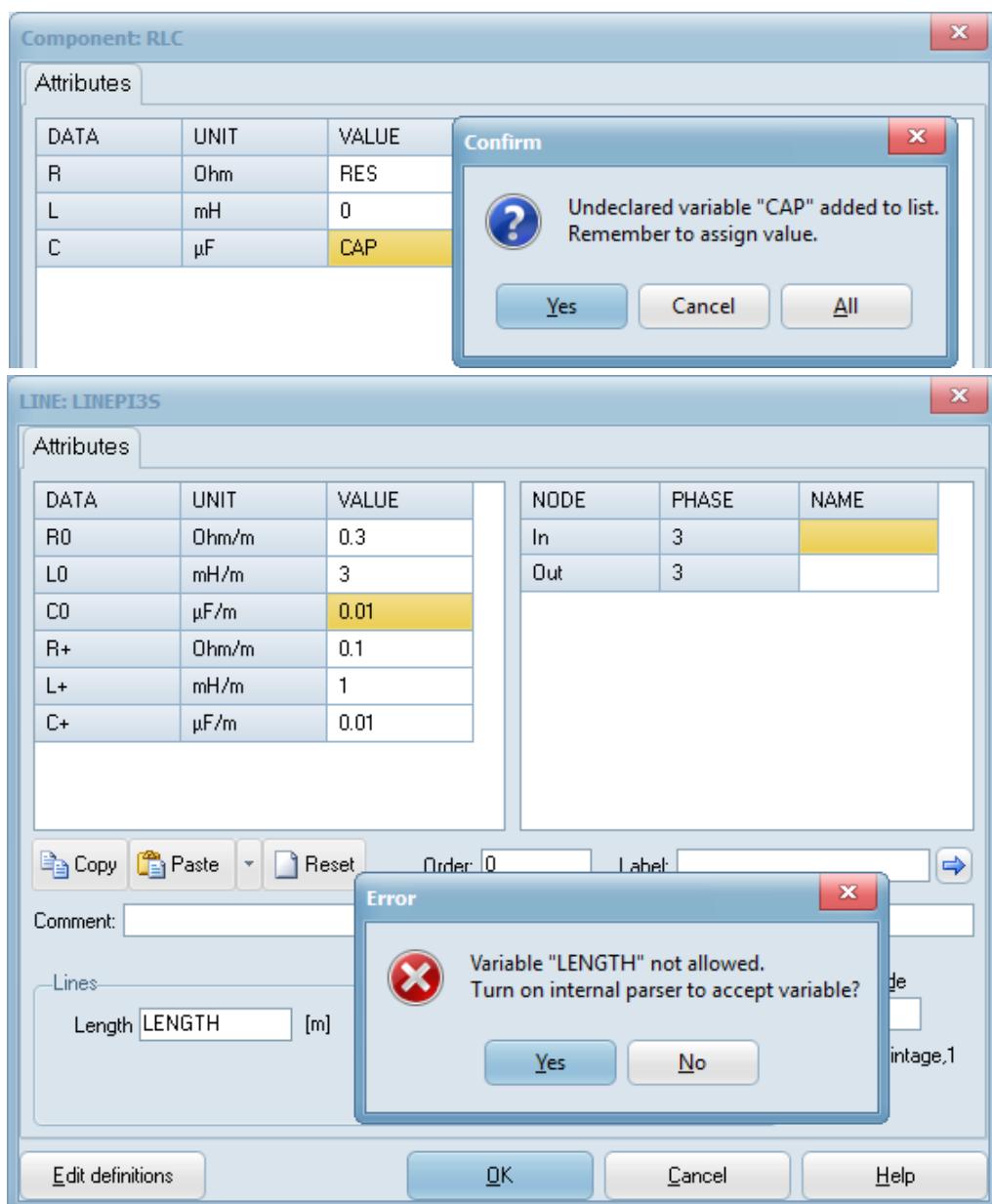


Fig. 4.18 - Using text string instead of variables in the RLC and LINEPI3S component dialog box.

An expression or a numerical value can be assigned to these text strings under *Variables* or in the *Sidebar*. The variables specified by the user appear in the NAME column and the user has to assign data values in the EXPRESSION column as shown by Fig. 4.18 . The user can also add intermediate variables and use these in subsequent expressions. If the *Internal Parser* is used, restrictions on the expressions are relaxed. The length of Variables can from v7.4 be of any length. Without, a period ‘.’ must be added to all numerical values in the expression, e.g. ‘U0\*1000.\*sqrt(2./3.)’. The following functions are defined in the *Internal Parser*:

One parameter functions (also available without the Internal Parser):

- ABS, SQR, SQRT, LOG, LOG10, SIGN, DEG, RAD, INVRS(x)=1/x, RECIP(x)=1/x, FLOOR, CEIL, TRUNC, RND, RANDOM

- SIN, COS, TAN, COTAN, ASIN, ACOS, ATAN, SINH, COSH, TANH, ASINH, ACOSH, ATANH

Two parameter functions (partly supported with the Internal Parser):

- POW(base, exp), INTPOW(base,exp), MIN, MAX, MOD, LOGN(base, value) , ATAN2(y,x)  
{atan(y/x) in 4 quadrants}, HYPOT(x,y)=sqrt(x\*x+y\*y)

Statistical two parameter functions; return random sample from the distribution:

- UNIFORM(a,b);  $f(x)=1/(b-a)$  for  $x$  in  $[a,b]$ ; returns a floating point value between  $a$  and  $b$
- UNIFORMI(a,b);  $f(x)=1/(b-a)$  for  $x$  in  $[a,b]$ ; returns an integer value between  $a$  and  $b$
- NORMAL(mu,sigma);  $f(x)=1/\sigma\sqrt{2\pi}\exp(-0.5\frac{(x-\mu)^2}{\sigma^2})$
- LOGNORMAL(M,sigma);  $f(x)=1/x/\sigma\sqrt{2\pi}\exp(-0.5\frac{\ln(x/M)-\mu}{\sigma^2})$ ;  $M=\exp(\mu)$
- WEIBULL(scale, shape);  $f(x)=shape/scale*(x/scale)^{shape-1}\exp(-(x/scale)^{shape})$

Statistical bounded (from  $a$  to  $b$ ) two parameter functions:

- NORMALB(a,b, mu,sigma);  $f(x)=1/\sigma\sqrt{2\pi}\exp(-0.5\frac{(x-\mu)^2}{\sigma^2})$
- LOGNORMALB(a,b, M,sigma);  $f(x)=1/x/\sigma\sqrt{2\pi}\exp(-0.5\frac{\ln(x/M)-\mu}{\sigma^2})$ ;  $M=\exp(\mu)$
- WEIBULLB(a,b, scale, shape);  $f(x)=shape/scale*(x/scale)^{shape-1}\exp(-(x/scale)^{shape})$

Double log-normal distribution:

- LOGNORMAL2(a,b,Ib,M1,sigma1,M2,sigma2) where  $a, b$  is the range,  $Ib$  is the boundary between regions (shielding failure and backflash).  $M1, \sigma_1$  are for the low region,  $M2$  and  $\sigma_2$  are for the high region. Related to next CIGRE functions;  $M1=61.0, \sigma_1=1.33, M2=33.3, \sigma_2=0.605$ . Single parameter set found by setting  $Ib=b$  and  $M1=M2=31.1$  and  $\sigma_1=\sigma_2=0.484$ .

Special distribution functions for lightning amplitudes:

- LACIGRE(a,b);  $cfd=0.5*(1+\text{erf}(\ln(x/61)/\sqrt{2}/1.33))$  for  $x<20$  and  $0.5*(1+\text{erf}(\ln(x/33.3)/\sqrt{2}/0.605))$  and  $x$  in  $[a,b]$ ;  $a$  and  $b$  in kA, answer returned in A
- LACIGRE1(a,b);  $cfd=0.5*(1+\text{erf}(\ln(x/31.1)/\sqrt{2}/0.484))$  and  $x$  in  $[a,b]$ ;  $a$  and  $b$  in kA, answer returned in A
- LAIEEE(a,b);  $cfd=1/(1+(31/x)^{2.6})$  and  $x$  in  $[a,b]$ ;  $a$  and  $b$  in kA, answer returned in A

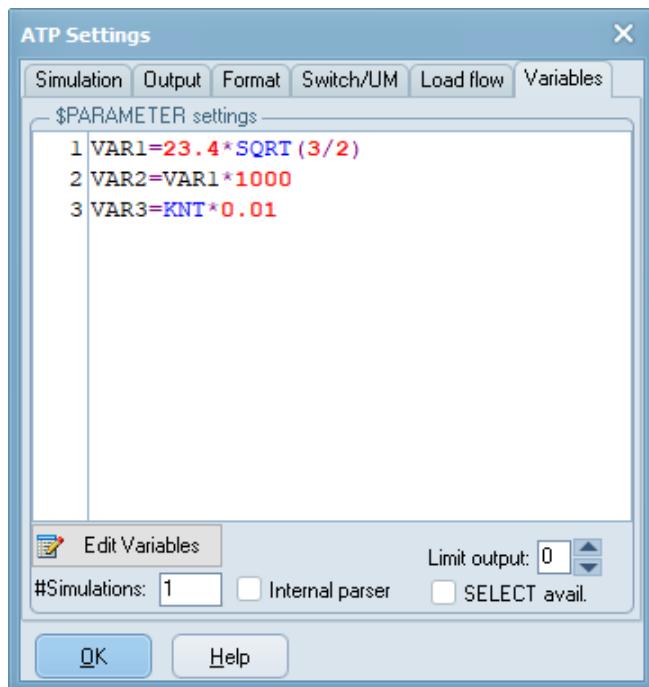
Three parameter function:

- IF(bool, true, false);  $y=\text{IF}(x>3, 2, 4)$  returns  $y=2$  if  $x>3$  otherwise  $y=4$ .

Resident variables (also available without the Internal Parser):

- PI, KNT {simulation number; 1, 2...}

Users do not have to think about the number of characters in the final ATP-file since ATPDraw automatically obtain the maximum resolution, in \$PARAMETER by adding underscore characters. A variable RES used both for high and low precision resistances will thus be declared twice with 3 and 13 underscore characters added. This process is hidden, but the result is seen in the final ATP-file after the \$Parameter declaration. Also, Models can utilize Variables and the default number of digits is set to 10 in this case. There is a limit in ATP on the number of internal variables in \$PARAMETER.



### \$PARAMETER option

The variables RES and CAP are circuit variables while R0 is a pure intermediate variable. The ATP file becomes:

```
PCVP 100
$PARAMETER
R0I =0.25 $$  

RESI =R0I*KNT $$  

CAPI =1.2E-3 $$  

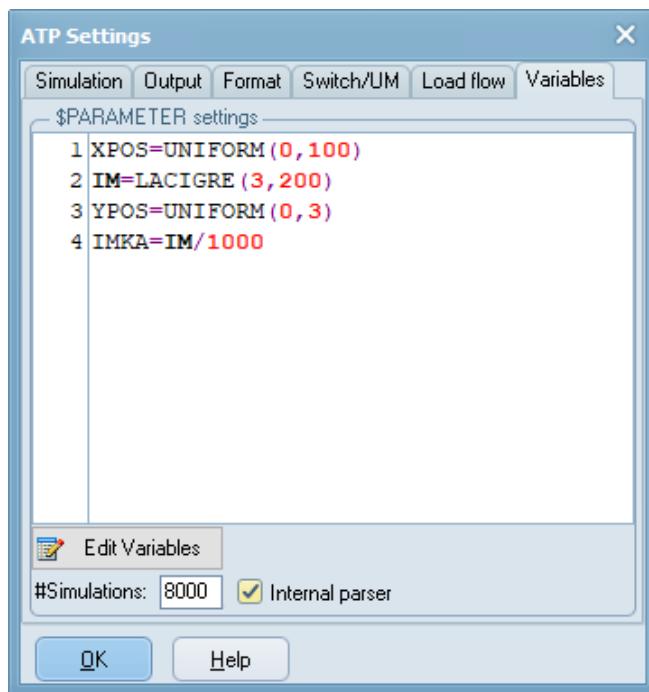
RES____=RESI  

CAP____=CAPI  

BLANK $PARAMETER
```

KNT is the simulation number (1..10 in this case).

**IMPORTANT!** Always use a period '.' after a number in the value field. ATPDraw v7.4> tries to add this automatically.



### Internal Parser option

No period '.' required in integer numbers

More statistical functions available

Six character variable names.

ATP executed several times, but in parallel in case of multiple core computers.

Fig. 4.19 - Assigning expressions and values to variables

Under *Limit output* the value of IOPCVP is set: 0: No LIS-file suppression, 1: Writes only extrema and parameters, 2: Writes only extrema, 3: Writes just the KNT information. IOPCVP must be zero for the Optimization or the WriteMaxMin module to work.

ATPDraw supports some special syntax for loop control (variables as function of the simulation number KNT). Without the Internal Parser there are severe restrictions in the number of managed elements. The special syntax is:

***MyVar=@[a b c ... n]***

First run (KNT=1): MyVar=a

Second run (KNT=2): MyVar=b

...

Last item and beyond (KNT >=n): MyVar=n

The characters '@[' are used to identify this format. Space or comma can be used to separate the numbers (integer or floating point). For \$PARAMETER, the ATP syntax SELECT was supposed to manage this, but as this manual is written there are problems with how ATPDraw implements intermediate variables and it is recommended to not check SELECT.

***MyVar=@FILE FileName Col***

'@FILE' is the keyword, *FileName* is the name of a text file assumed stored in the *ResultDirectory* unless a full path is given (same as final ATP file) (enclose the file name within " " if it contains space), and *Col* is an optional parameter identifying which column in the text file to use. The text file can have integer or floating-point values in free format space or comma separated. If *Col* is not specified, the first column of the file is loaded. The length of the file does not need to match the chosen *Number of Simulations*.

First run (KNT=1): MyVar=First value of column *Col*

Second run (KNT=2): MyVar=Second value of column *Col*

etc.

Both the '@[' and '@FILE' syntax requires a lot of intermediate variables and ATP puts a limit on this.

***MyVar=@FILEA FileName Col***

Same behavior as the @FILE syntax, but instead of a file on disk an attached file is used.

***MyVar=@LIN Lo Hi***

'@LIN' is the keyword. Creates a linear space. MyVar=a\*(KNT-1)+b

***MyVar=@LOG Lo Hi***

'@LOG' is the keyword. Create a logarithmic space. MyVar=10\*\*(a\*(KNT-1)+b)

***MyVar=@POW Lo Hi P***

'@POW' is the keyword. MyVar = a\*(KNT-1)\*\*P+b

***MyVar=@EXP Lo Hi P***

'@EXP' is the keyword. MyVar = a\*P\*\*(KNT-1)+b

If P ='e' this is replaced by exp(1)

*a* and *b* are calculated based on *Lo* and *Hi*: First run (KNT=1) MyVar=*Lo*, Last run (KNT=Number of Simulations) MyVar=*Hi*. The last four options could easily be managed directly by the user.

The user should normally not change the name of the variables listed by ATPDraw in the NAME column, but if you do you will be asked to take an Action regarding the old Variable still defined in the circuit, as shown in Fig. 4.20. The action can be to reset the parameter to zero or the default or a specific value, reintroduce the old variable and give an expression or select a new variable name.

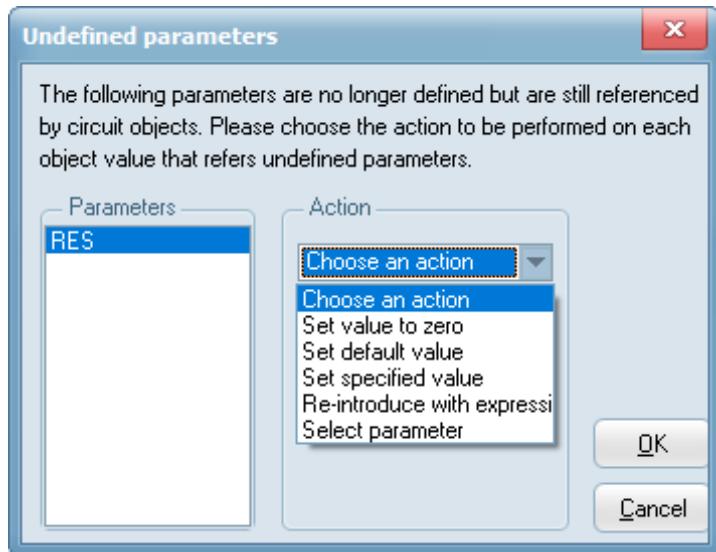


Fig. 4.20 - Actions to take when non-defined parameters are found.

#### 4.2.5.2 Run ATP

Executing the *run ATP* command at the top of the ATP menu will create the ATP input file (the project file name (with extension .atp) and the /ATP system folder are default, but changeable via *Sub-process/Make ATP file*). Then ATP is executed based on the default *ATP command* (specified in the *ATP* field of the *Preferences* page under *Tools / Options*). The current ATP-file is sent as parameter to the ATP-EMTP. Note that users do not need to select *Make Names* and *Make ATP File* before running the simulation. These commands are internally executed before the ATP run. If the user needs to do manual changes of the ATP-file and run the modified case, use *ATP/Sub-process/run ATP file*. After executing ATP, ATPDraw examines the LIS-file and displays any error or warning messages if exist.

#### 4.2.5.3 Run Plot

Execute the *Plot program* (defined under *Tools/Options/Preferences*) with the current ATP file name and the extension .pl4.

#### 4.2.5.4 Sub-process

This sub-menu contains the individual three parts of the run ATP command.

- Run ATP file: Executes ATP and sends the current ATP file as parameters. This choice must be used if the user has manually modified the ATP file under *ATP/Edit ATP file*.
- Make ATP file: Creates the ATP file from the circuit without executing ATP (but calls Make node names first). This choice must be used to change the current ATP file name and the Result Directory.
- Make node names: Gives node names to all nodes in the circuit. Overlapping and/or connected nodes get the same name. Whenever a "same name on different nodes" or "duplicate names on same node" are found, ATPDraw produces a warning and the user is asked to confirm this operation. While ATPDraw establishes the node names a **Generating node names** message is displayed in the middle of the current circuit window. Following *Make Names*, the node name and phase sequence attributes in the *Component* dialog and in the *Node data* window will be updated. *Make ATP file* and *run ATP* perform this sub-process initially.

**IMPORTANT!** All nodes will automatically receive names from ATPDraw, so the user should normally only give names to nodes of special interest, e.g. involved in output requests and displayed in the Output Manager.

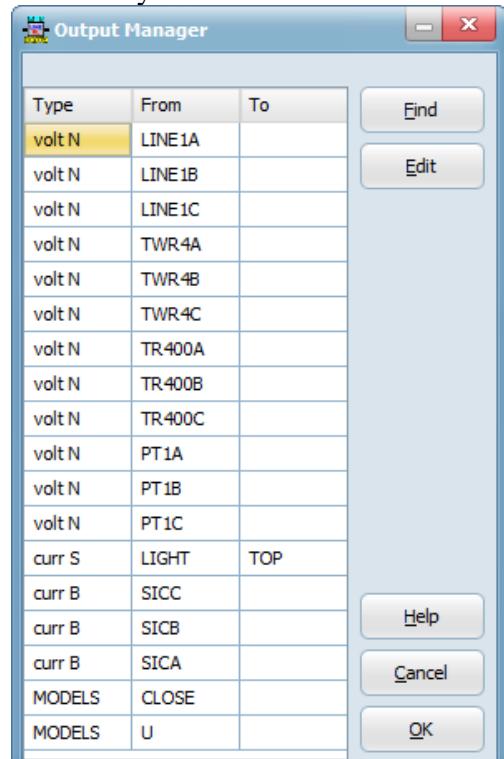
#### 4.2.5.5 Output manager

The Output Manager list “all” requested outputs in the data case in the order that they appear in the pl4 file. The sorting option of the components is considered. The Output Manager even goes into User Specified, Additional data cards and Windsyn components to find outputs requested there. There is a limit of 32 output requests per component (voltage&current counts as one). The sequence of the output is:

- Branch voltages and power
- Switch voltages and power
- Node voltages
- Switch currents and energy
- Branch currents and energy
- SM
- TACS
- MODELS
- UM

When launching the Output Manager it compiles the circuit to generate the node names and presents a list of the outputs as shown in Fig. 4.21. The Windows Manager is a stay-on-top window that lets the user go back to edit the circuit. Two additional features are available; *Find* and *Edit*. Both are linked to the current selected row in the grid. The *Find* button finds the involved component and displays it in the middle of the screen in a lime color. If necessary, it goes down into groups to display internal components. The *Edit* button brings up the involved component’s input dialog where the user can edit the settings. However, the user must leave the Output Manager and reopen it to refresh its content.

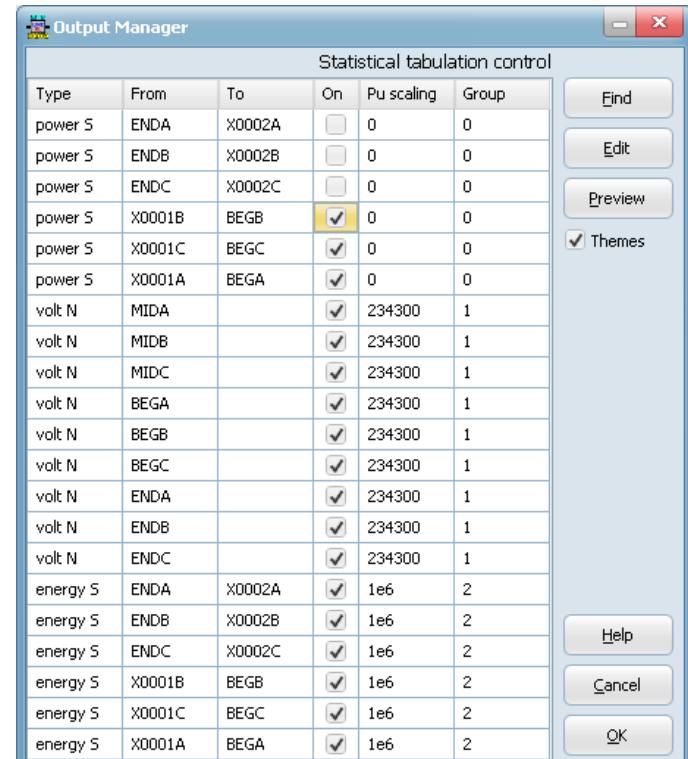
When ATPDraw goes into User Specified components it lists the node names found in the expected columns. This could however be an argument in the \$Include call, and this is not handled by ATPDraw.



The screenshot shows the Output Manager window with the title "Output Manager". It contains a table with three columns: "Type", "From", and "To". The rows include various output types such as "volt N", "curr S", and "energy S", along with their corresponding connection points like "LINE1A", "TOP", and "BEGB". On the right side of the window, there are four buttons: "Find", "Edit", "Help", and "OK".

Type	From	To
volt N	LINE1A	
volt N	LINE1B	
volt N	LINE1C	
volt N	TWR4A	
volt N	TWR4B	
volt N	TWR4C	
volt N	TR400A	
volt N	TR400B	
volt N	TR400C	
volt N	PT1A	
volt N	PT1B	
volt N	PT1C	
curr S	LIGHT	TOP
curr B	SICC	
curr B	SICB	
curr B	SICA	
MODELS	CLOSE	
MODELS	U	

Fig. 4.21- Output Manager from Exa\_9.acp



The screenshot shows the Output Manager window with the title "Output Manager". It contains a table with six columns: "Type", "From", "To", "On", "Pu scaling", and "Group". The rows include various output types and their connections, similar to Fig. 4.21. On the right side, there are five buttons: "Find", "Edit", "Preview", "Themes", "Help", "Cancel", and "OK".

Type	From	To	On	Pu scaling	Group
power S	ENDA	X0002A	<input type="checkbox"/>	0	0
power S	ENDB	X0002B	<input type="checkbox"/>	0	0
power S	ENDC	X0002C	<input type="checkbox"/>	0	0
power S	X0001B	BEGB	<input checked="" type="checkbox"/>	0	0
power S	X0001C	BEGC	<input checked="" type="checkbox"/>	0	0
power S	X0001A	BEGA	<input checked="" type="checkbox"/>	0	0
volt N	MIDA		<input checked="" type="checkbox"/>	234300	1
volt N	MIDB		<input checked="" type="checkbox"/>	234300	1
volt N	MIDC		<input checked="" type="checkbox"/>	234300	1
volt N	BEGA		<input checked="" type="checkbox"/>	234300	1
volt N	BEGB		<input checked="" type="checkbox"/>	234300	1
volt N	BEGC		<input checked="" type="checkbox"/>	234300	1
volt N	ENDA		<input checked="" type="checkbox"/>	234300	1
volt N	ENDB		<input checked="" type="checkbox"/>	234300	1
volt N	ENDC		<input checked="" type="checkbox"/>	234300	1
energy S	ENDA	X0002A	<input checked="" type="checkbox"/>	1e6	2
energy S	ENDB	X0002B	<input checked="" type="checkbox"/>	1e6	2
energy S	ENDC	X0002C	<input checked="" type="checkbox"/>	1e6	2
energy S	X0001B	BEGB	<input checked="" type="checkbox"/>	1e6	2
energy S	X0001C	BEGC	<input checked="" type="checkbox"/>	1e6	2
energy S	X0001A	BEGA	<input checked="" type="checkbox"/>	1e6	2

Fig. 4.22 - Output Manager from an extension of Exa\_12.acp as shown in Fig. 4.23.

In the case of a statistical study (chosen under *ATP/Settings/Switch*) the Output Manager lists three additional columns as shown in Fig. 4.22. In the fourth columns in Fig. 4.22 the user can turn available output requests on and off for statistical tabulation. Only node voltages are on as default. In the sixth column the user can assign a group number to the statistical output request and in the fifth column assign a scaling factor to this group. There is also a *Preview* button available in this mode that lets the user examine how the final statistical tabulation will look like. This text will appear under */STATISTICS* in the final ATP file.

```
/STATISTICS
 234300.MIDA  MIDB  MIDC  BEGA  BEGB  BEGC  ENDA  ENDB  ENDC
-4          1.E6ENDA    ENDB      ENDC      BEGA      BEGB      CONT.
-4          1.E6BEGC
```

There is one challenge related to SATURABLE TRANSFORMERS and the request of magnetizing branch outputs. This would require a very complicated identification of the transformer that is not handled in ATPDraw. The magnetization output is presented in the Output Manager (using an alias node name) but it is not possible to add this to a statistical tabulation.

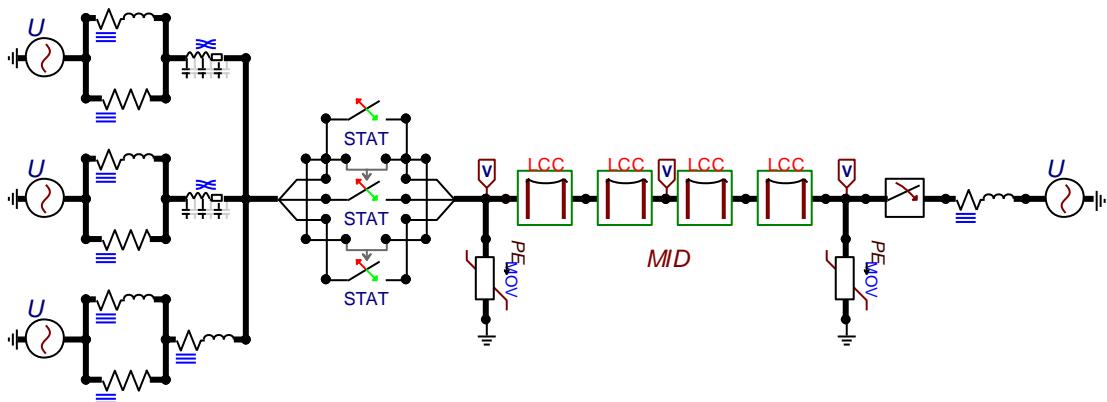


Fig. 4.23 - Exa\_12.acp requesting additional output (both side node voltages and arrester powers and energies).

#### 4.2.5.6 Edit ATP-file

This selection calls a text editor, which enables the user to view or edit the ATP-file. When the *Edit File* option is selected (or the *F4* function key is pressed) a file having the same name as the active circuit file with extension *.atp* is searched for, and will be opened in the built in Text Editor as shown in Fig. 4.24. The editor will be show as a non-modal window and the user has to close it manually.

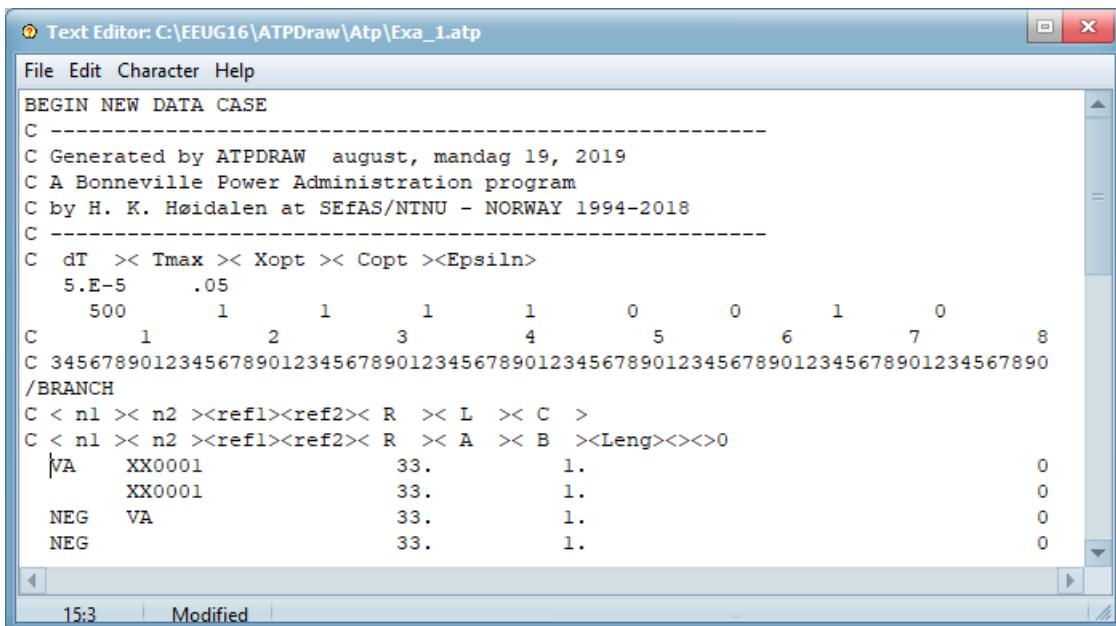


Fig. 4.24 - The main window of the built-in text editor.

The status bar at the bottom of the window displays the current line and column position of the text cursor, and the buffer modified status. Basic text editing facilities (Open/Save, Print, Copy/Paste, Find & Replace) are supported. The default text font can be changed by selecting the *Font* option in the *Character menu*. A detailed description of all the available options can be found in the menu options help topic. The text buffer of this editor is limited to maximum 2 GB in size. The user can specify his own favorite text editor (wordpad.exe, write.exe, notepad.exe) on the *Preferences* page of the *Tools / Options* dialog box. The right-click context menu offers 50 different request card templates via the *Insert* field.

*Text Editor* option in the *Tools* menu provides an alternative way of invoking this editor. In that case the text buffer will initially be empty.

#### 4.2.5.7 View LIS-file

This selection calls the built-in text editor, which enables the user to contemplate the LUNIT6 output of ATP (often called as LIS-file). This file has extension .lis and can be found in the Result Directory (default the /ATP system folder) following a successful simulation. In certain cases when the simulation is halted by an operating system interrupt or a fatal error in the ATP input file (illegal file name, I/O-xx bad character in input field, etc.) the LIS-file does not exist and cannot be displayed either.

#### 4.2.5.8 Find node and Find next node

The *Find node* helps the user to find a node with a specific name in the circuit. You type in the node name in the simple dialog. For multi-phase node you only type in the root name without phase extensions 'A'..'Z'. *Find next node* is used to proceed to the next node with the same name. *Find node* goes into groups as well, and (multiple) *Edit/Edit circuit* (Ctrl+H) may be necessary to navigate back into the main circuit.

#### 4.2.5.9 Find variable and Find next variable

The *Find variable* helps the user to find a variable with a specific name in the circuit. You type in the variable name in the simple dialog and the first instance of a component containing the variable is marked in lime color and the circuit is centered around it. *Find next variable* is used to

proceed to the next node with the same name. *Find variable* goes into groups as well, and (multiple) *Edit/Edit circuit* (Ctrl+H) may be necessary to navigate back into the main circuit.

#### 4.2.5.10 Optimization

To use the optimization module there must be variables declared in the circuit and a cost function object must have been added to the circuit (*MODELS/WriteMaxMin*). The optimization module will change chosen circuit variables to optimize the cost function based on either a Gradient Method, a Genetic Algorithm, or a Simplex Annealing method. This is further documented in the Advanced Manual, chapter 5.11.

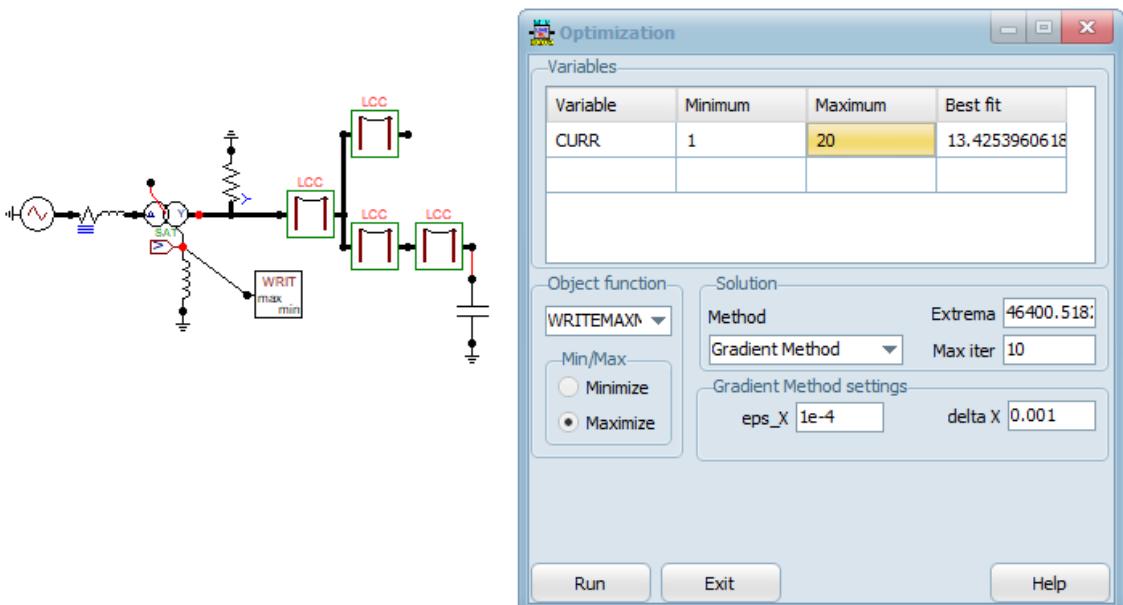


Fig. 4.25 – Finding the neutral grounding coil value giving resonance, Exa\_18.acp.

#### 4.2.5.11 Line Check

First, the user selects the line he wants to test and then clicks on *ATP/LineCheck* as shown in Fig. 4.26 . Then the input/output selection dialog box shown in Fig. 4.27 appears.

The *LineCheck* feature in ATPDraw supports up to 3 circuits. ATPDraw suggests the default quantities. The leftmost nodes in the circuit are suggested as the input nodes, while the rightmost nodes become the output. The circuit number follows the node order of the objects. For all standard ATPDraw components the upper nodes have the lowest circuit number. The user also must specify the power frequency of the line/cable test. Finally, the user can check the *Exact phasor equivalent* button which will result in a slightly better results for long line sections.

When the user clicks on OK in Fig. 4.27 an ATP-file (/LCC/LineCheck.dat) is created and ATP executed. For a 3-phase configuration 4 sequential data cases are created (Z+, Y+, Z0, Y0) while for a 9-phase configuration 24 cases are created (Z11+, Y11+, Z110, Y110, Z12..., Z22..., Z13..., Z23..., Z33...), since symmetry is assumed. Finally, the entire LIS-file is scanned. The calculated values are then presented in the result window shown in Fig. 4.26 . The user can switch between polar and complex coordinates and create a text-file of the result. The mutual data are presented on a separate page. The unit of the admittances is given in Farads or Siemens (micro or nano) and the user can scale all values by a factor or by the length.

The series impedances are obtained by applying 1 A currents on the terminals while the output ends are grounded (the other circuits are left open and unenergized). For mutual coupling, 1 A is applied at both circuits. On the other hand the shunt admittances are obtained by applying a voltage source of 1 V at one terminal leaving the output end open. For mutual coupling, 1V is applied at one circuit while a voltage of 1E-20 is applied at the other.

Special attention must be paid to long lines and cables. This applies in particular to PI-equivalents. Usage of 'Exact phasor equivalent' is recommended but is no guarantee of success. No attempt is made in ATPDraw to obtain a better approximation since the line/cable system to be tested in general is unknown. The mutual coupling in the positive sequence system is in symmetrical cases very small and vulnerable to the approximations made. Appendix 7.2 documents the calculation procedure.

It might happen that the result for multi-circuit test gives strange result as a consequence of ATP failing to execute the required number of stacked cases. In this case it might help to reduce the LIMCRD parameter in ATP's STARTUP file. A value below 100.000 should be used.

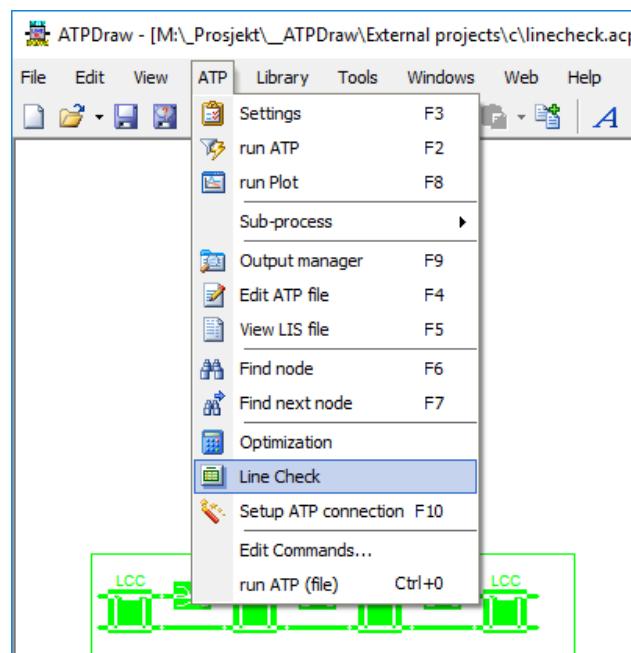


Fig. 4.26 – Selecting a line.

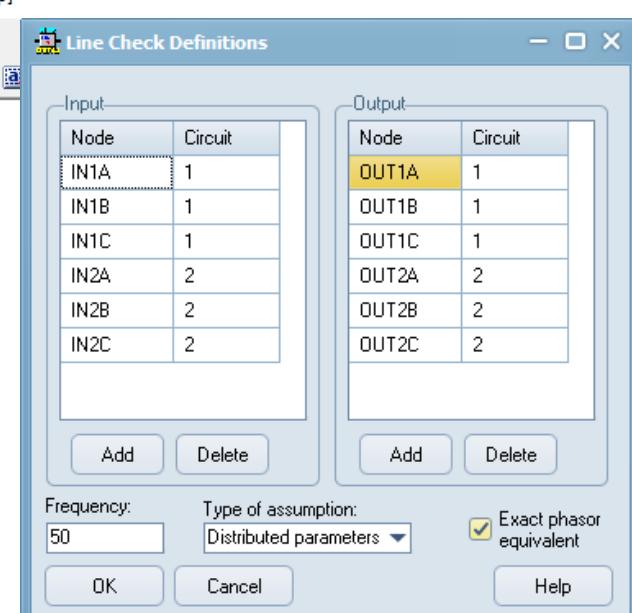


Fig. 4.27 – Selecting the inputs and outputs.

ATPDraw v7.4 introduced a new option in Line Check; *Type of assumption*. These can be Distributed parameters:  $Z_{sc}=3U_s/(I_s+2\cdot I_r)$ ,  $Y_{oc}=3I_s/(U_s+2\cdot U_r)$  as described in Chapt. 7.2  
 PI-equivalent:  $Z_{sc}=2U_s/(I_s+I_r)$ ,  $Y_{oc}=2I_s/(U_s+U_r)$   
 Sending end only:  $Z_{sc}=U_s/I_s$ ,  $Y_{oc}=I_s/U_s$  as is used in the Verify part in the LCC dialog.

The subscripts s and r stand for the sending and receiving end, respectively.

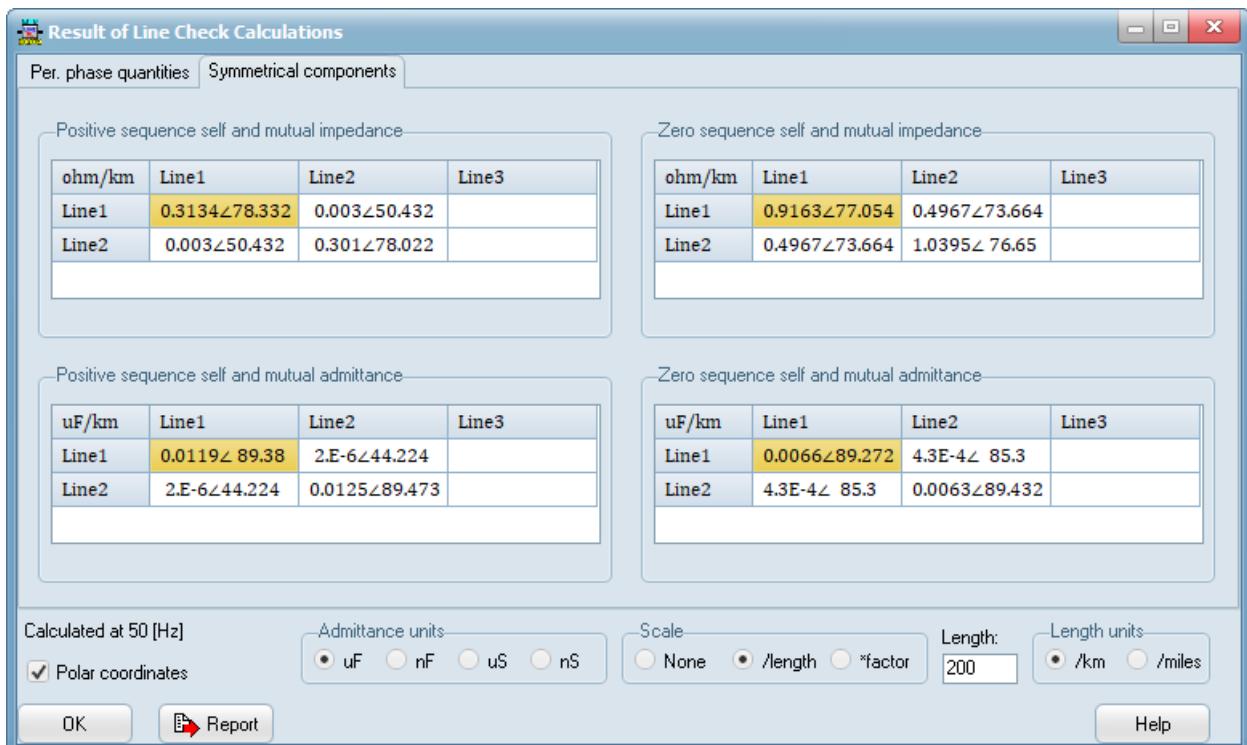
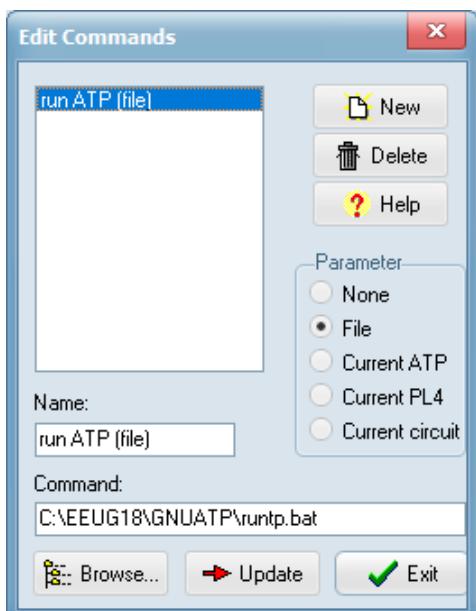


Fig. 4.28 - Presentation of the results.

#### 4.2.5.12 Edit Commands...

This feature enables to specify executable files (\*.exe or \*.bat) to run from the ATP menu. New commands will appear as menu items below the *Edit Commands...* After clicking on the *New* button of the dialog box as shown in Click Update to confirm and store the command.

Fig. 4.29, the user is requested to specify:



- Name* of the command displayed under the ATP menu
- Command* is the full path of the executable file (\*.exe or \*.bat),
- Parameter* is the file to send as parameter when calling the executable file. Click Browse to select.
  - None: No file sent as parameter
  - File: A file open dialog box is displayed where the user can select a file
  - Current ATP: send the current ATP-file
  - Current PL4: send the current PL4-file
  - Current Circuit:

Click *Update* to confirm and store the command.

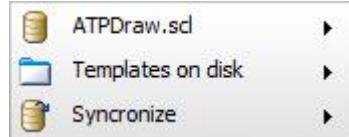
Fig. 4.29 - Specifying your own executable commands.

When you completed editing the batch job settings, click on the *Update* button and the new commands will be inserted into the ATP menu. This feature can be used for many different purposes in ATP simulation: e.g. running different ATP versions (Salford, Watcom, GNU-

MingW32) within ATPDraw; running external post-processors like TPPLOT, PCPlot or PlotXY; or launching any other data assembler.

As any other program options, the previous settings can be saved to the `ATPDraw.ini` file by using the *Tools / Save Options* command or by selecting the “Save options on exit” program options on the *General* page of the *Tools / Options* menu.

#### 4.2.6 Library



This menu contains options for creating and customizing component support files. Support files contain definitions of data and node values, icon and help text. Circuit components in ATPDraw can be either:

1. Standard,
2. User specified, or
3. Model

Each component has a unique support file (template), which includes all information about the input data and nodes of the object, the default values of the input variables, the graphical representation of the object and the associated help file. Standard components have their support files stored in `ATPDraw.scl` (standard component library). When a component is added to the circuit this component inherit the properties from its support file and the support file is not used anymore. Except for the help text of standard components. In order to define and use User Specified components a support file `.sup` is required. Models can optionally be managed without a support file since a default template can be automatically created based on the Models text header.

All components' support files can be edited in the *Library* menu. The user can create new MODELS and User Specified components as described in the Advanced Manual.

##### 4.2.6.1 ATPDraw.scl

Under this menu the user can edit standard component templates stored in `ATPDraw.scl`. This can be to fix bugs in the templates. It is not a good idea to put effort into changing a lot here, as the `ATPDraw.scl` file will be overwritten if you update the installation. The Developer has more options under this menu. Selecting the *Edit Standard* field will first perform a select file dialog box of Fig. 4.30, where the support file to be edited can be selected, then a dialog box shown in Fig. 4.31 appears. The Num. data and Num. node specifiers cannot be changed.

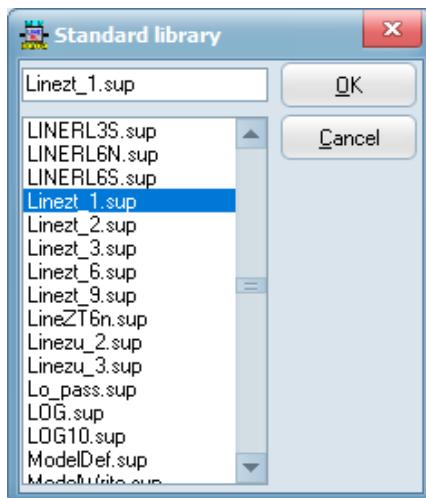


Fig. 4.30 - Specify the support file of the standard component to be edited.

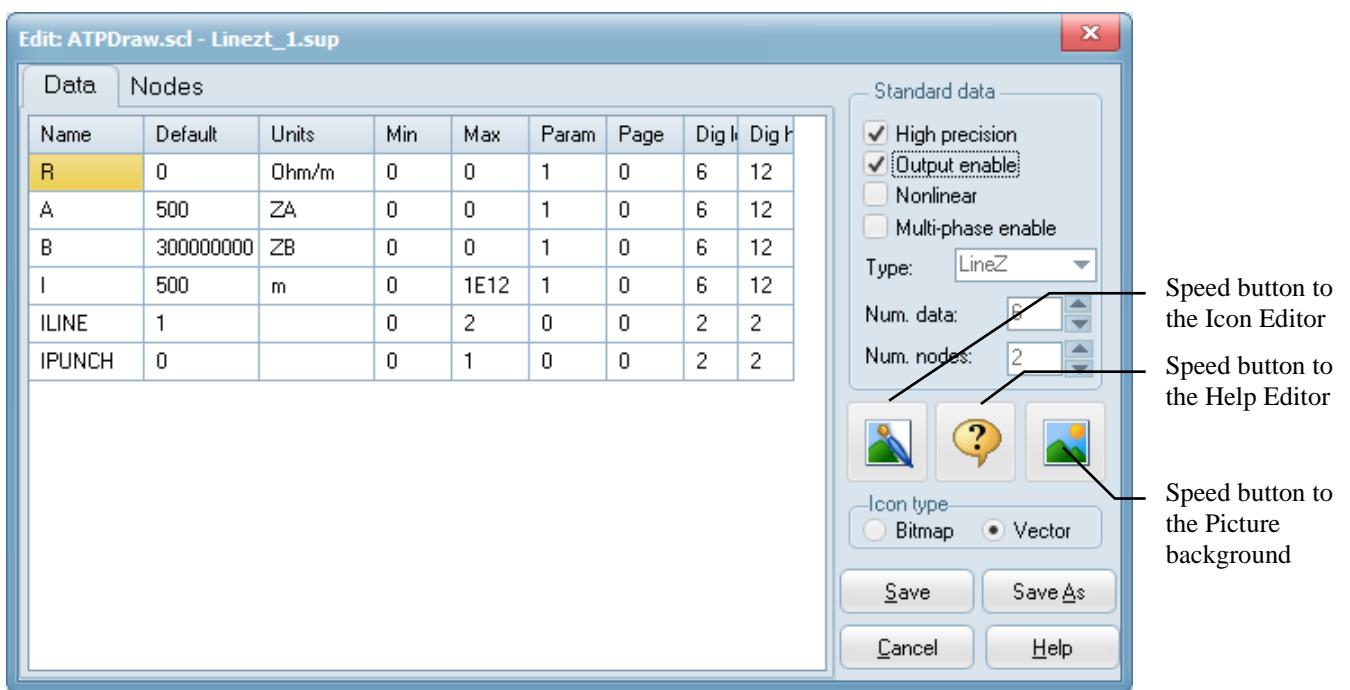


Fig. 4.31 – Template editor.

#### 4.2.6.2 Templates on disc|New User specified sup-file

The user can create and edit templates for User Specified components and Models (\*.sup) on disc, as well as Models script files (\*.mod) on disc. This indirect way of working with Models is not recommended.

User specified objects are either customized standard objects or objects created for the use of \$INCLUDE and Data Base Modularization feature of ATP-EMTP. The *Library|Templates on disc|New User Specified* menu enables the user to create a new support file for a user specified object or customize data and node values, the icon and the help text of an existing one.

Support files of USP objects are normally located in the /USP folder. The *Edit Definitions* dialog box opens with empty *Data* and *Nodes* tabs in this menu. Number of nodes and data must be in line with the ARG and NUM declarations in the header section of the Data Base Module (DBM) file. The number of data can be in the range of 0 to 2 giga, and the number of nodes in the range of 0 to 2 giga. Control parameters for the object data can be entered on the *Nodes* and *Data* pages

of Fig. 4.32.

On the *Data* page of the *Edit Object* dialog box, control variables of the support file (one row for each object data) can be specified.

Name	The name of the parameter. Used to identify the parameter in the <i>Component</i> dialog box. This name often reflects the name used in the ATP Rule Book.
Default	Initial value of the parameter.
Units	Maximum 12 character text string with the unit that appear in the Component dialog box. The units COPT and XOPT are defined keywords responding to the users choice of COPT/XOPT under the ATP Settings/Simulation.
Min/Max	Minimum/Maximum value allowed. Set equal to cancel range checking.
Param	If set equal to 1, a variable text string can be assigned to the data value. These values are assigned under ATP Settings/Variables.
Digits	Maximum number of digits allowed in the ATP-file. When high precision is checked, \$Vintage, 1 is enabled and Digits is split in two values for high and low precision.

An error message will appear in the *Component* dialog if a parameter value is out of range. To cancel range checking, set Min=Max (e.g. set both equal to zero).

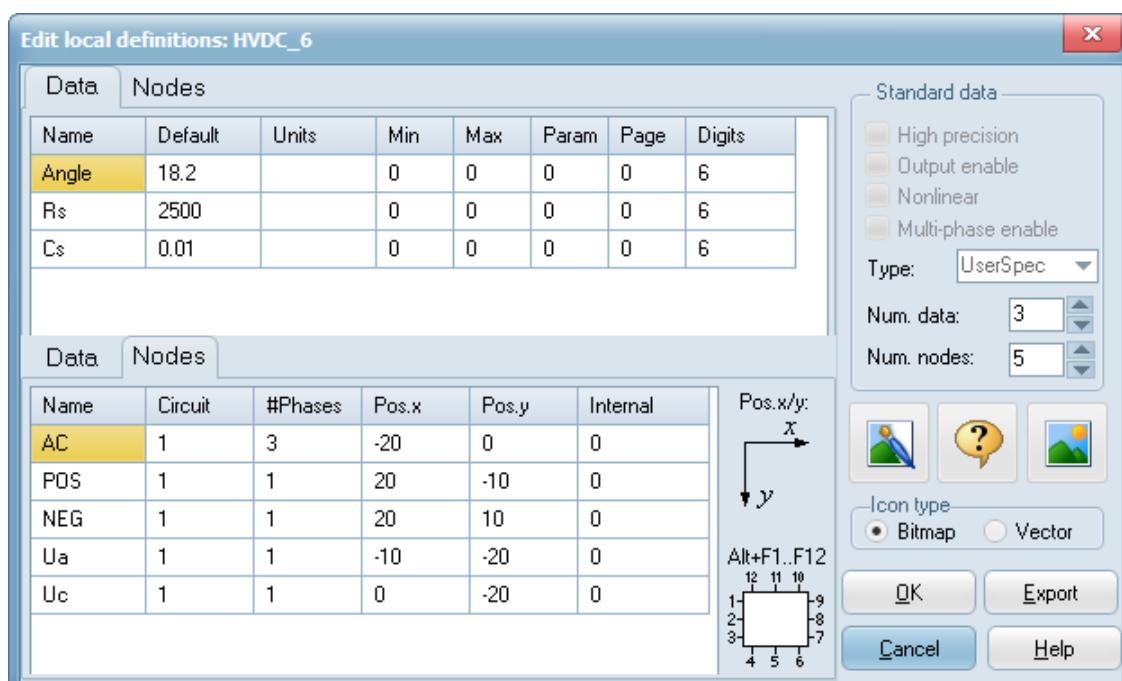


Fig. 4.32 - Control page of a new user specified object.

On the *Node* page of the *Edit definitions* dialog box, the node attributes of the support file (one row for each component node) can be specified.

<b>Name:</b>	The name of the node. Used to identify the node in the <i>Open Node</i> and <i>Component</i> dialog boxes.
<b>Circuit:</b>	3-phase circuit number of the object. The number is used to handle transposition of 3-phase nodes correctly for objects having more than 3 phases. Kind=1 for all nodes of single phase objects. 3-phase nodes with the same Kind get the same phase sequence.
1:	1st to 3rd phase
2:	4th to 6th phase
3:	7th to 9th phase

4: 10th to 12th phase  
 The *Circuit* parameter has a different meaning for MODELS or TACS component nodes. It is used to specify the type of input/output.

MODELS node values:

- 0: Output node.
- 1: Current input node.
- 2: Voltage input node.
- 3: Switch status input node.
- 4: Machine variable input node.
- 5: TACS variable (tacs)
- 6: Imaginary part of steady-state node voltage (imssv)
- 7: Imaginary part of steady-state switch current (imssi)
- 8: Output from another model. Note that the model, which produces this output, must be USED before the current model. This can be done by specifying a lower Order number for the model and then select the Sorting by Order number option under ATP|Settings/Misc.
- 9: Global ATP variable input.
- 10: Variable from connected PL4-file.

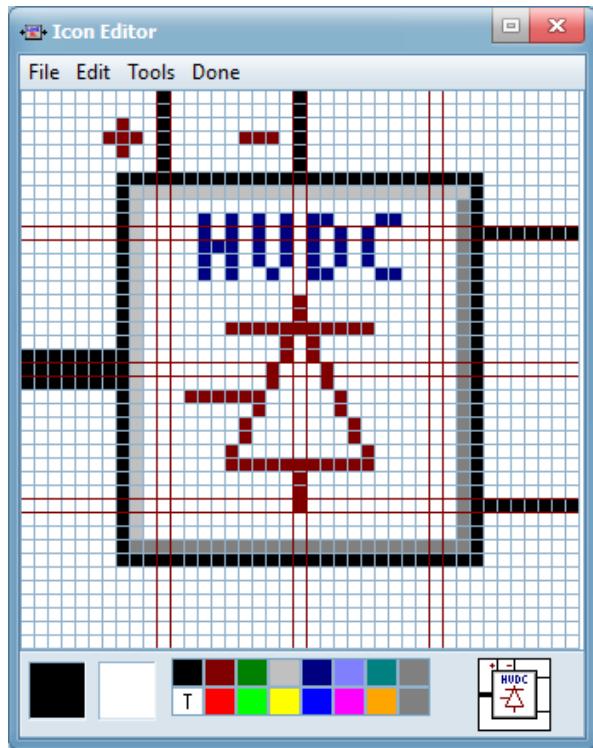
TACS node values:

- 0: Output node.
- 1: Positive sum input node.
- 2: Negative sum input node.
- 3: Disconnected input node.

**#Phases:** Number of phases (1..26) for the component node. If #Phases is set to >1 the length of the node name is limited to 5. The last character of nodes (in the proper phase sequence according to *Kind*) will be appended by ATPDraw.

**Pos:** Specifies the relative node position in steps of 10 pixels (grid). The standard border positions shown in the picture to the left of Fig. 4.38 have short cut keys Alt+F1..Alt+F12. The position (x, y) can in general be in the range -120,-110,...-10,0,10,...,110,120. The x-axis is oriented to the right while the y-axis is oriented downwards. The node positions should correspond with icon drawing.

Each circuit object has an icon, which represents the object on the screen. This icon can be of bitmap type or vector graphic type as selected under *Icon type*. The conversion from Bitmap to Vector style is not possible so you should not unintentionally change the icon style. Vector graphic enables better zooming and graphic export, font handling and editing, but for simplicity reasons the Bitmap option is shown here. The leftmost of the three speed buttons on the right-hand side of the Fig. 4.31 invokes the built-in pixel editor where icons can be edited. Each icon has equal width and height of 41x41 pixels on the screen.



Clicking with the left mouse button will draw the current color selected from a 16 colors palette at the bottom. Clicking the right button will draw with the background color. Dark red colored lines indicate the possible node positions on the icon border. Menu field items of the *Icon Editor* are described in the section 4.2.7.1 of this manual. The user can draw individual pixels and in addition lines, rectangles, circles, and fills. Text must be manually put together by pixels. The Vector graphic editor has far better text capabilities.

Fig. 4.33 - Icon Editor.

Each component has a pre-defined help file, which can be edited by a built in *Help Editor* accessible via the speed button on the middle speed button in the *Edit definitions* dialog in Fig. 4.31 . Using the help editor, users can write optional help file for the objects or add their notes to the existing help text. Available functions and menu field items of the *Help Editor* are described in the 4.2.7.3 section of this manual.

With the rightmost speed button in Fig. 4.38 the user can add a background bitmap/metafile image of any size to the icon. This should only be used in special cases since it could heavily occupy memory and increase the project file dramatically. No down-sampling of the imported image is performed.

When the user has completed all modifications of the component data and of the icon and help, the new support file can be saved to disk using *Save* (existing support file will be overwritten) or *Save As* (new file will be created in the \USP folder) buttons.

#### 4.2.6.3 Templates on disc|New Model sup-file

Usage of MODELS [4] in ATPDraw is described in the Advanced Manual. When the user changes the Model header (input, output or data section) in a circuit in ATPDraw the component and its icon is automatically updated. So for the usual case of a dynamic Model there is no point in pre-defining support and model files. These files can anyhow be exported from a finished Model. If you want a static Model, however, you can specify a support file under this menu item. To use this feature, you first must write a model file using the built in *Model Editor* as shown in section 4.2.6.4. This file must have a legal MODELS structure (e.g. starting with MODEL name and ending with ENDMODEL), have an extension .mod and stored in the \MOD system folder. ATPDraw is capable of reading such a .mod file, examining its input/output and data variables and suggesting a support file on the correct format (see in section 4.15.9 and 5.5.1). If the user wants a different icon or other node positions on the icon border, he is free to modify the default sup-file, or create a new one by selecting the *Objects | Model / New sup-file* menu. This menu item will perform the *Edit Definitions* dialog as shown in Fig. 4.34.

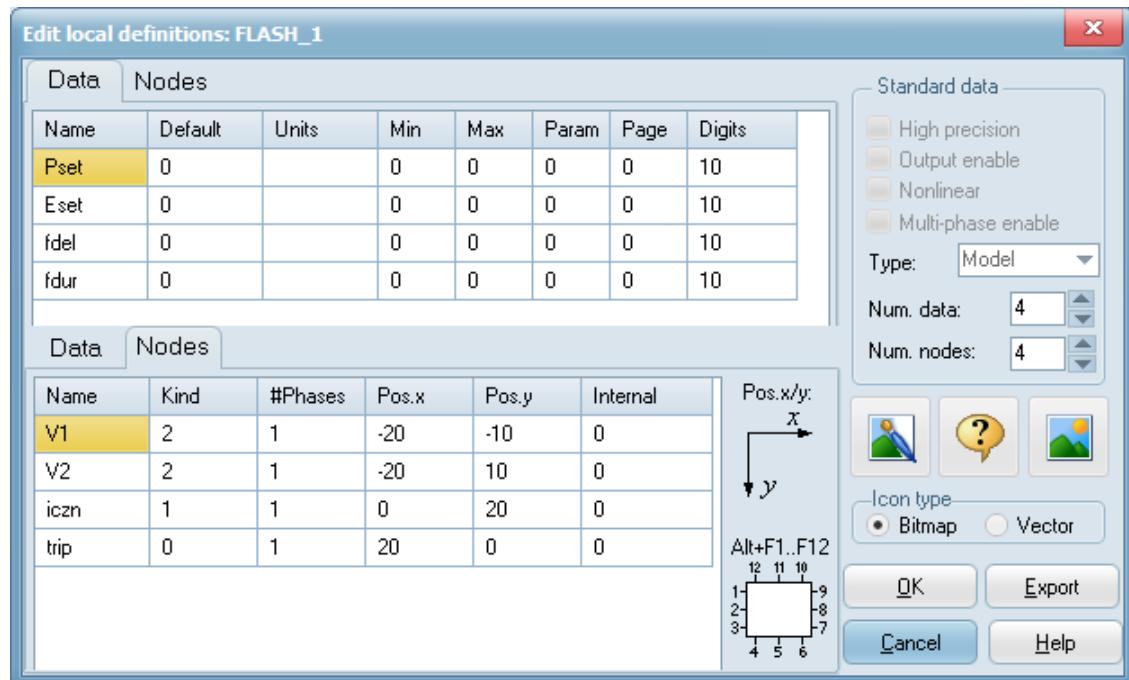


Fig. 4.34 - Control page for a New Model sup-file.

**Name:** Identifies the node in the *Node* and *Component* dialog boxes. 12 characters maximum. Must be equal to the name used in the Model header.

**Kind:** Specifies the input/output type of the node.

**#Phases:** Number of phases can be from 1 to 26 and must be defined as V1[1..n].

**Pos:** Specifies the relative node position in steps of 10 pixels (grid). The standard border positions shown in the picture to the left of Fig. 4.38 have short cut keys Alt+F1..Alt+F12. The position (x, y) can in general be in the range -120,-110,...-10,0,10,...,110,120. The x-axis is oriented to the right while the y-axis is oriented downwards. on the icon border. The node positions should correspond with icon drawing.

Supported *Kind* values for MODELS objects are shown next:

- 0: Output node.
- 1: Current input node.
- 2: Voltage input node.
- 3: Switch status input node.
- 4: Machine variable input node.
- 5: TACS variable (tacs)
- 6: Imaginary part of steady-state node voltage (imssv)
- 7: Imaginary part of steady-state switch current (imssi)
- 8: Output from other model. Note that the model which produces this output must be USED before the current model. This can be done by specifying a lower *Order number* for the model and then select the *Sorting by Order number* option under ATP|Settings/Format.
- 9: ATP global variable. MNT is for instance the simulation number and the Pocket Calculator KNT equivalent.
- 10: PL4 Variable

The number of *Nodes* is the sum of inputs and outputs to the Model. The number of *Data* must be equal to the number of DATA declarations of the actual Model. The *Kind* parameter can be changed later in the Model node input window (right click on the node dot). All model nodes are assumed a single-phase one. The maximum number of nodes is 32 and the maximum number of data that can be passed into a Model is 64.

The *Save* or *Save As* buttons can be used to save the new support file to disk. Default location of Model support files is the \MOD folder.

#### **4.2.6.4 Templates on disc|New Model mod-file**

In addition to a support file and icon definition, each Model component needs a text file which contains the actual Model description. This file may be created outside ATPDraw or using the built in *Model Editor*. Selecting the *Library | New object/ Model mod-file* menu, the well-known internal text editor of ATPDraw pops-up.

ATPDraw supports only a simplified usage of MODELS. It is the task of the user to write the model-file and ATPDraw takes care of the INPUT/OUTPUT section of MODELS along with the USE of each model. The following restrictions apply:

Only INPUT, OUTPUT and DATA supported in the USE statement.

Not possible to specify expressions, HISTORY of DELAY CELLS under USE

Not possible to call other models under USE.

#### **4.2.6.5 Templates on disc>Edit User Specified sup-file**

An existing user specified object can be edited in the same way as any standard components as described in chapt. 4.2.6.1.

#### **4.2.6.6 Templates on disc>Edit Model sup-file**

A model object can be edited like any other circuit object. If the user clicks on the *Library | Edit object / Model sup-file*, the *Edit Definitions* dialog box appears with the model object controls. Here the user is allowed to customize data and node values, icon and help text of the object.

#### **4.2.6.7 Templates on disc>Edit Model mod-file**

Selecting the *Objects | Model / Edit mod-file* menu, the well-known internal text editor of ATPDraw pops-up. Each model object has a .mod file which contains the description of the model. This file can be edited inside ATPDraw using the built in *Model Editor*.

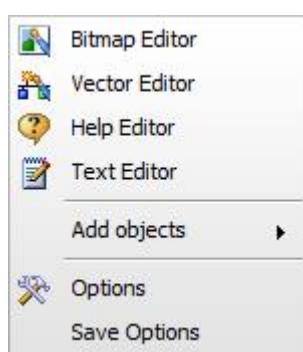
#### **4.2.6.8 Synchronize| Reload Icons**

Reads and displays standard component icons from their respective support files. This function is useful when the user has redesigned one or more support file icons and wants the changes to be reflected in the circuit window. User Specified and Models components icons are not updated.

#### **4.2.6.9 Synchronize| Reload Standard Data**

Updates data properties (name, units, digits, range, param) for all standard components from ATPDraw.scl. The properties are stored in project files, but sometimes there are updates in the ATPDraw.scl file that should be manually synchronized (that was the case with the LINEZT\_1... components that got updated ZA, ZB units with special interpretation).

#### 4.2.7 Tools



Items under the *Tools* menu enable you to edit component icons or help text, view or edit text files, add special circuit objects, customize several program options and save them to the *ATPDraw.ini* file.

Fig. 4.35 - Tools menu.

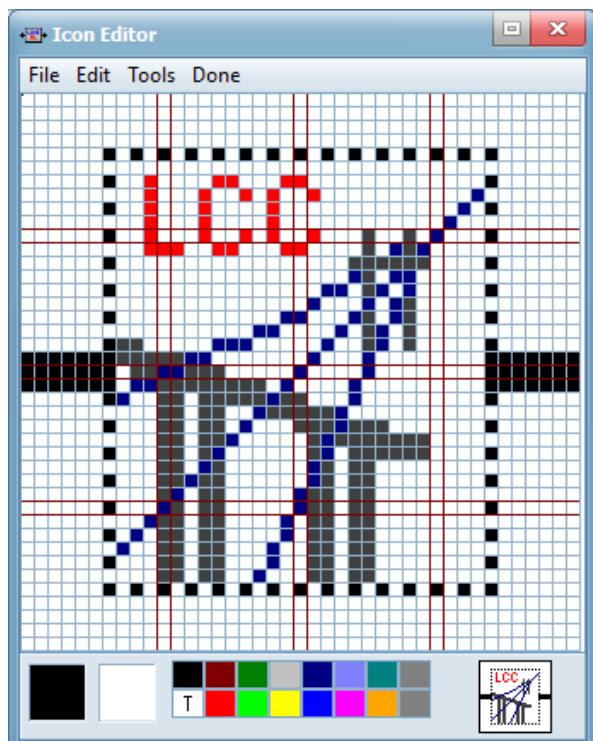


Fig. 4.36 - Icon Editor menus.

##### 4.2.7.1 Bitmap Editor

Brings up an icon editor shown in Fig. 4.36 where the user can edit the icon of the component. It can be invoked either from the *Template editor* or by selecting the *Icon Editor* option in the *Tools* menu

Depending on how the editor was invoked, the file menu provides different options. When called from the *Library* menu (*Edit Standard*, *User Specified* or *Edit Model sup-file*), the user can import icons from other support files or cancel the edit operation and close the editor window. In this case, the *Done* option in the main menu is seen to accept and store the modified icon in the *.sup* file.

When the icon editor is called from the *Tools* menu, additional options like the *Open* and *Save* appears in the *File* menu.

At the bottom of the editor window there is a color palette with two boxes indicating the current foreground and background color selections, and the real-size image of the icon at right. In the color palette, the color marked with a capital letter *T* is the transparent color.

To select a color from the palette, click either the left or the right mouse button in one of the color boxes. The selected color will be assigned to the mouse button you clicked until you use the same mouse button to select another color. The leftmost box displays the color currently assigned to the left mouse button. The one to the right displays the color assigned to the right mouse button.

The foreground color is normally used to draw with, and the background color to erase any mistakes made during the drawing. It is therefore convenient to assign the transparent color (indicated by *T*) to the right mouse button, and desired drawing color to the left button. Mistakes can then easily be corrected by alternating left/right mouse button clicks.

The vertical and horizontal lines of dark red color indicate the icon node positions. These are in the same position as indicated on the *Nodes* pages of the *Template editor*.

The icon editor has a *File* menu, an *Edit* menu and a *Tools* menu. In addition, a *Done* option appears to the right of the *Tools* menu if the editor has been called from the *Component* dialog box. Selecting *Done*, changes made to the icon will be accepted. Available menu options are described below:

<i>File</i> options	
Open	Loads the icon of a support file into the icon buffer.
Save	Stores the contents of the icon buffer to disk.
Import	Reads the icon of a support file and inserts it into the icon buffer.
Merge	Request an external support file and adds its icon to the current icon.
Exit/Cancel	Closes the icon editor window. If the option Exit is selected and the icon buffer have been modified, you are given a chance to save the icon before closing. If the Done option is visible in the main menu, the name of this menu item is Cancel, and the icon editor window is closed without any warning with respect to loss of modified data.
<i>Edit</i> options	
Undo	Cancels the last edit operation.
Redo	Cancels the undo command.
Cut	Copies a bitmap version of the icon to the Clipboard and clears the icon buffer. This bitmap can be pasted into other applications (e.g. pbrush.exe).
Copy	Places a bitmap version of the icon in the Clipboard.
Paste	Inserts the bitmap in the Clipboard into the icon buffer. If colors are different from those used in the original bitmap, it is because the icon editor calculates which color in its own color palette provides the nearest match to any bitmap color.
Delete	Clears the icon buffer.
<i>Tools</i> options	
Pen	Selects the pen drawing tool, enabling you to draw single icon pixels, or lines or shapes by pressing and holding down the left or right mouse button while you move the mouse.
Fill	Selects the flood fill tool. Fills any shape with the current color.
Line	Selects the line drawing tool, enabling you to draw a rubber band line by pressing and holding down the left or the right mouse button while you move the mouse.
Circle	Selects the circle drawing tool, enabling you to draw a dynamically sized circle by pressing and holding down the left or the right mouse button while you move the mouse.
Rectangle	Selects the box drawing tool, enabling you to draw a rubber band box by pressing and holding down the left or the right mouse button while you move the mouse.

#### 4.2.7.2 Vector graphic editor

In ATPDraw all icons of standard components are in vector graphic style. This enables better zooming and dynamic icon capabilities. A component can have either a bitmap or a vector icon, but not both. The building block of the vector graphic format is the Element. An Element has a Visible flag and can belong to a Layer, it is thus possible to easily turn on/off element as a response to user settings. Further an element can either be a Shape or a Text. A shape can be of various standard Windows types (lines, rectangle, ellipses, poly-lines, polygons, arcs, pies, and Bezier curves), while a Text is simpler. A Shape can consist of maximum 255 points which is very beneficial for poly-lines, polygons and Bezier curves. The vector graphic editor has been developed from scratch utilizing an internal graphic format for fast drawings. The editor is shown in Fig. 4.37 .

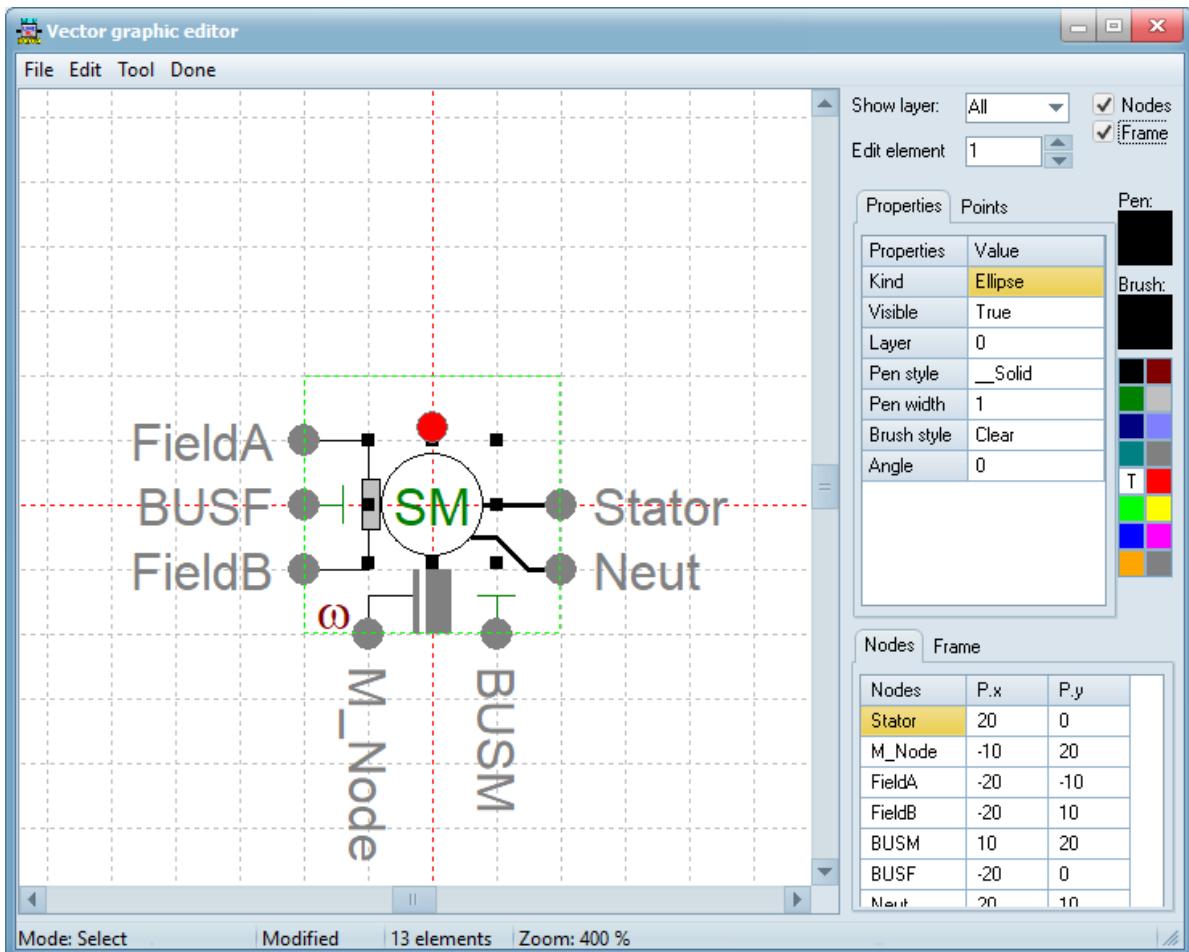


Fig. 4.37 – Vector graphic editor (400 % zoom).

An element can be selected by clicking in the icon window or by specifying the *Edit element* spin edit field to the top right. The selected element is shown with its properties below. In the *Properties* grid the pen and brush colors and styles can be selected with colors from the palette to the right. In addition, rotation angle of rectangles and ellipses and rounding of rectangles can be set. In the *Points* grid the co-ordinates for the points are shown and can be edited.

Fig. 4.37 also shows the items *Nodes* and the *Frame*. These are turned on/off via the checkboxes in the very top right corner. The *Frame* is the selection area of the icon; mouse clicks inside this area in the circuit will select or open the component. A too large *Frame* will result in overlapping conflicts with other icons. The Frame is not changeable with the mouse; the user has to specify the coordinates in the *Frame* string grid. The External point drawn as red is used for branch output of some of the components. The Nodes are drawn as gray dots with their node names oriented relative to the Frame. The Node positions and name can be specified in the Nodes string grid. The nodes can also be moved with the mouse selecting *Tool/Move nodes*. The nodes have to be on the grid so the nodes are only moved in steps. The grid is also drawn in Fig. 4.37 with the red lines indicating the center. The grid can be turned on/off via *Edit/Grid*.

When the editing process is completed the user clicks on *Done*.

#### 4.2.7.2.1 Properties

Fig. 4.38 – Fig. 4.40 shows the properties grids. Most of the properties have combo boxes and pop-up dialogs attached as shown in Fig. 4.39 for selection of possible values.

The figure shows three separate property grid windows side-by-side. The left window is for a shape, the center for another shape, and the right for text.

Properties	Value
Kind	Ellipse
Visible	True
Layer	0
Pen style	<u>Solid</u>
Pen width	1
Brush style	Clear
Angle	0

Properties	x	y
P1	-21	-10
P2	-11	-10
P3	-11	10
P4	-21	10

Properties	Value
Text	SM
Visible	True
Layer	0
Style	0
Size	6
Font	Arial
Rotate	False
Angle	0
P.x	0
P.y	0

Fig. 4.38 – Properties grid. Left and center: Shapes. Right: Texts.

The figure shows three property grid windows illustrating alternative settings for shape properties.

Properties	Value
Kind	Ellipse
Visible	Line
Layer	Rectangle
Pen color	Ellipse
Pen style	Polyline
Pen width	Polygon
Brush color	Arc
Brush style	Bezier
Brush color	Pie
Brush style	Clear

Properties	Value
Kind	Ellipse
Visible	True
Layer	False
Pen color	True
Pen style	<u>Solid</u>
Pen width	1
Brush color	246
Brush style	Clear

Pen style	<u>Solid</u>
Pen width	Solid
Brush color	<u>—Dash</u>
Brush style	....Dot
Brush style	-. Dashdot
Brush style	. DashDotDot
Brush style	Clear
Brush style	Solid
Brush style	<u>Clear</u>
Brush style	Horizontal
Brush style	Vertical
Brush style	BDiagonal
Brush style	FDiagonal
Brush style	Cross
Brush style	Diag cross

Fig. 4.39 – Shape properties alternatives.

The figure shows a property grid window for text properties.

Properties	Value
Angle	0
P.x	0
P.y	90
P.y	180
P.y	270

Font	Arial
Rotate	Arial
Angle	System
P.x	MS Sans Serif
P.y	Arial Narrow
P.y	Comic Sans MS
P.y	Courier
P.y	Times New Roma
P.y	Symbol

Fig. 4.40 – Text properties.

The *Shape* points are given in the co-ordinate system where x increases from left to right, and y increases from top to bottom. 32 bit integer are used to store the data, so this is no practical size restriction. The Text point P is specified in the center of the text. The Node co-ordinates have to be rounded off to the nearest 10.

The colors can be chosen from the 16 standard colors in the color palette to the right. Left click for pen (outline) color, right click for brush (fill) color. Full 24 bit colors are found by clicking on the Pen/Brush squares which will bring up the Windows Color selector.

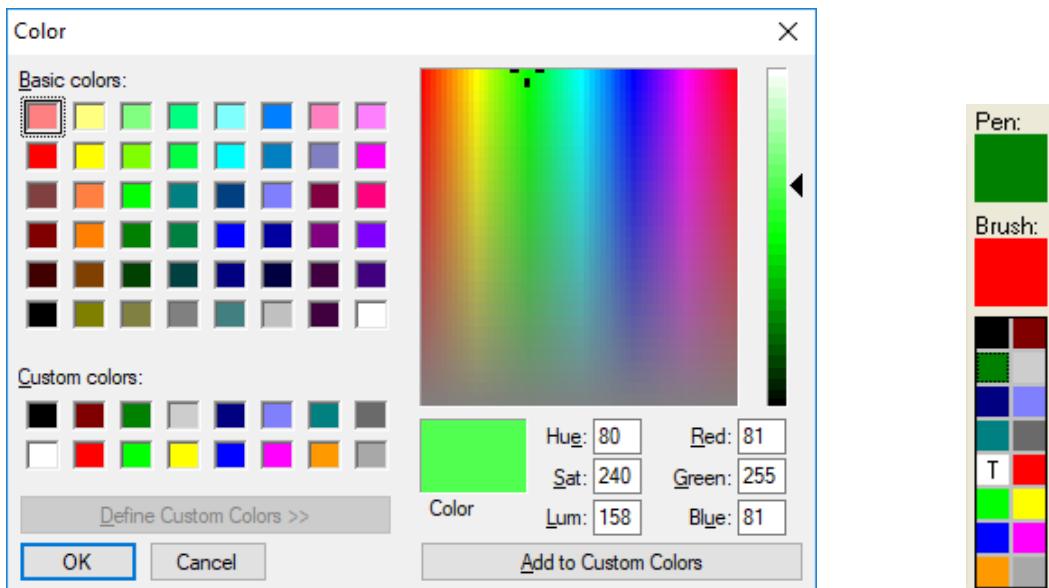


Fig. 4.41 – The Full 24-bit RGB color palette.

Standard ATPDraw colors.

#### 4.2.7.2.2 Editing: Selecting, moving, resizing and clipboard

An element is selected by clicking on it in the icon window. If the brush color is clear the user has to click on the visible border (does not apply to arcs and pies). Extensive code is added to support clicking on Bezier curves. If an element is already selected it is given priority in the selection process. Click in open space to unselect the element. Several elements can be selected by holding down the shift key or by clicking in open space and draw an enclosing rectangle. A single element or a group of elements can be moved clicking and holding down the left mouse key. Elements can be resized by clicking on one of the eight black marking squares (the mouse cursor changes style in this case). A group of elements can also be resized.

It is possible to move all elements via the *Tool/Move all* menu, and this is the same as *Edit>Select all* + normal move.

The position of elements can be fine-tuned by holding down the shift key and use the arrow keys to move the selected group one pixel. The point position can also be typed directly into the points grid shown in Fig. 4.38.

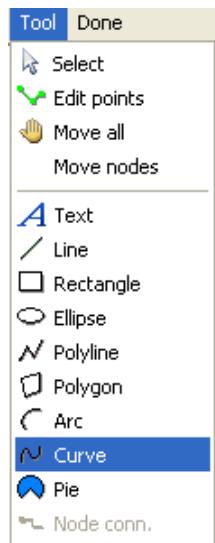
The order of elements can be changed via the *Edit/Arrange* menu where the four choices, send up/down, send to back/front are available. Elements or groups of elements can also be rotated 90 deg. and flipped left to right or top to bottom via the *Edit/Flip&Rotate* menu.

It is possible to copy selected elements to the windows clipboard. This can then be pasted into other icons (or duplicated). To place the graphical content in metafile format on the clipboard select *Edit/Copy Graphics*.

#### 4.2.7.2.3 Drawing new elements

A new element is drawn by selecting the proper tool under *Tools*. The following tools are available:

After selecting the tool click with the left mouse button to place points and with the right mouse button to place the final point. Line, rectangles, ellipses, arcs, and pies take a fixed number of points so the left/right clicking does not really matter in this case. For polylines, polygons, and



(Bezier) curves the number of points can range up to 255 maximum. When drawing Bezier curves only the curve points follow the mouse clicks (point 1-4-7-10 etc.) while the intermediate control points (2-3, 5-6 etc.) are calculated internally.

Fig. 4.42 – Available modes and tools

The shape points can be edited later by entering the *Tool/Edit points* mode. The shape points are then drawn as green squares which can be moved directly. It is also possible to add or delete points by clicking the right mouse button and choose from the pop-up menu. Bezier curves are handled in a special way as shown in Fig. 4.43. The curve points are drawn in a lime color while the control points are drawn in red with a line to their curve points. The curve points lies on the curve while the control points sets the curve derivative. (In the drawing tool in Windows office (Word and Power points) the left and right control points are forced to lie on the same tangent and this will force a smooth curve). When points are added to or deleted from Bezier curves this directly affects the curve point while the control points are automatically added/removed. The Bezier curve can be closed by selecting Brush style solid.

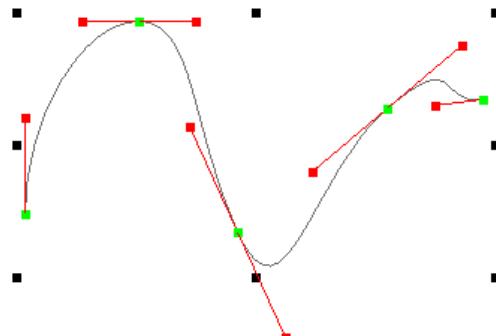


Fig. 4.43 – Bezier curve drawn in *Edit points* mode. Green squares: curve points, red squares: control points.

#### 4.2.7.2.4 Layers and visible

Each element can belong to a specific layer as specified in the properties grid in Fig. 4.38-Fig. 4.39. The layers can be shown individually by changing the Show layer item in Fig. 4.37. Elements with Layer=0 are always drawn. The practical usage of this for user specified icons is limited to separation of elements in the drawing process. For standard elements though, the Layer property is used to turn on/off elements dynamically. This is hard coded in the source code of ATPDraw an affects RLC elements, transformers, time controlled and statistical switches, TACs devices, sources (current/voltage), LCC transmission lines (overhead line, single core cables, enclosing pipe + length), and universal machines. The Layer information is used to control the

Visible property. Elements with `Visible=false` are not drawn in the circuit window, but they are drawn in the icon editor.

#### 4.2.7.2.5 Example of complex icons

In the new vector graphics editor, quite complex icons can be created. There is no limit of the size of the icon or the number of elements.

One of the benefits with vector graphic icons is that it is possible to create larger and much more complex icons. Fig. 4.44 shows an example of a created windmill and transformer icons.

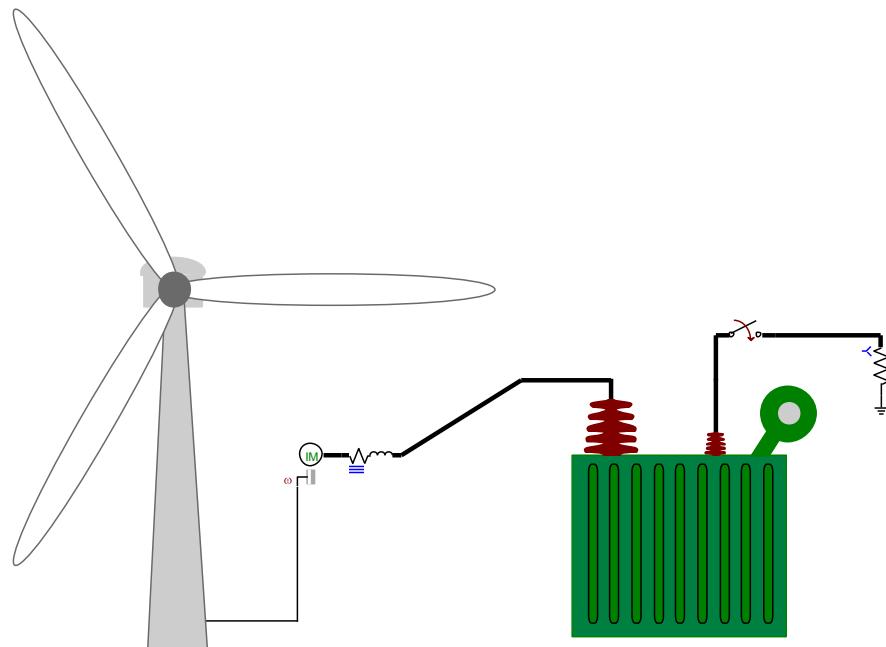


Fig. 4.44 – Wind turbine and transformer icon with connecting universal machine and load in standard size.

#### 4.2.7.3 Help Editor/Viewer

Displays the *Help Editor* where the current help text assigned to components can be modified. The *Help Editor* and the *Viewer* supports a simple rich text format (rtf), but so far this is not much utilized in help files. However, it is possible with different font styles and colors, bulleted lists etc, but not pictures. To edit help file of standard objects, the user must select the *Help Editor* speed button in any *Edit Template* dialogs. In this case a *Done* option appears in the main menu and the *File* menu provides printing options and a *Cancel* choice. By selecting *Done* you accept any changes made to the help text.

When the editor is called from the *Tools* menu, the *File* menu contains an *Open* and a *Save* option, as well. In that case the text buffer is initially empty, so the user must select the *File / Open* first to load the help text of a support file. The default font can be changed by selecting the *Font* option in the *Character* menu. This menu will bring up the Windows standard font dialog box where you can specify a new font name and character style, size or color. Note that ATPDraw does not remember the current font setting when you terminate the program, so if you don't want to use the default font, you must specify a new one each time you start ATPDraw. The *Word Wrap* option toggles wrapping of text at the right margin so that it fits in the window.

When the built-in editor is used as a viewer of component help text, editing operations are not allowed and the *File* menu provides printing options only. Additionally, the *Find & Replace* option is missing in the *Edit* menu.

The status bar at the bottom of the window displays the current line and character position of the text buffer caret, and the buffer modified status. This status bar is not visible when viewing component help. A more detailed description of menu options is given in the next sub-section.

#### 4.2.7.4 Text Editor

To invoke the editor, you may select the *Text Editor* option in the *Tools* menu or the *Edit ATP-file* or *Edit LIS-file* in the *ATP* menu. In the latter case, the file having the same name as the active circuit file with extension .atp or .lis are automatically loaded. When the program is called from the *Tools* menu, the text buffer will initially be empty.

The status bar at the bottom of the window displays the current line and character position of the text buffer caret, and the buffer modified status. Any other text processor (e.g. notepad.exe or wordpad.exe) can be used, if *Text editor:* setting of the *Preferences* page in the *Tools / Options* menu overrides the default one.

A detailed description of the menu options is given below:

##### *File* options

New	Opens an empty text buffer. ( <i>Built-in text editor only!</i> )
Open	Loads the help text of a support file or the contents of a text file into the text buffer.
Save	Stores the contents of the text buffer to disk.
Save As	Stores the contents of the text buffer to a disk file.
Print	Sends the contents of the text buffer to the default printer.
Print Setup	Enables you to define default printer characteristics.
Exit/Cancel	Closes the editor or viewer window. If the option displays Exit and the text buffer has been modified, you are given a chance to save the text before closing. If a Done option is available from the main menu, this option displays Cancel, and the window will close without any warning with respect to loss of modified data.

##### *Edit* options

Undo	Cancels the last edit operation.
Cut	Copies selected text to the Clipboard and deletes the text from the buffer.
Copy	Puts a copy of the selected text in the Clipboard.
Paste	Inserts the text in the Clipboard into the text buffer at the current caret position.
Delete	Deletes any selected text from the text buffer.
Select All	Selects all the text in the buffer.
Find	Searches the text buffer for the first occurrence of a specified text string and jumps to and selects any matching text found. This option displays the Windows standard Find dialog box.
Find Next	Searches for the next occurrence of the text string previously specified in the Find dialog.
Find&Replace	Searches the text buffer for one or all occurrences of a specified text string and replaces any instance found with a specified replacement string. This option displays the Windows standard Replace dialog box.

##### *Character* options

Word Wrap	Toggles wrapping of text at the right margin so that it fits in the window.
-----------	---

**Font** From the Windows standard Font dialog box you can change the font and text attributes of the text buffer.

#### 4.2.7.5 Add objects

From this menu Texts, Shapes, Pictures, Files and Plot object can be added to the circuit. Shapes are further split into Lines, Arrows, Rectangles and Ellipses and requires two left-clicks for the upper-left and bottom-right corners.

#### 4.2.7.6 Options

In the *Tools / Options* menu several user customizable program options for a particular ATPDraw session can be set and saved to the ATPDraw.ini file read by all succeeding sessions. During the program startup, each option is given a default value. Then, the program searches for an ATPDraw.ini file in the current directory, the directory of the ATPDraw.exe program, the Windows installation directory and each of the directories specified in the PATH environment variable. When an initialization file is found, the search process stops and the file is loaded. Any option values in this file override the default settings. The ATPdraw.ini file is stored under %APPDATA%\atpdraw (typically c:\documents and settings\user\program data\atpdraw) and is unique for each user of the computer. The file is ATPDraw version independent.

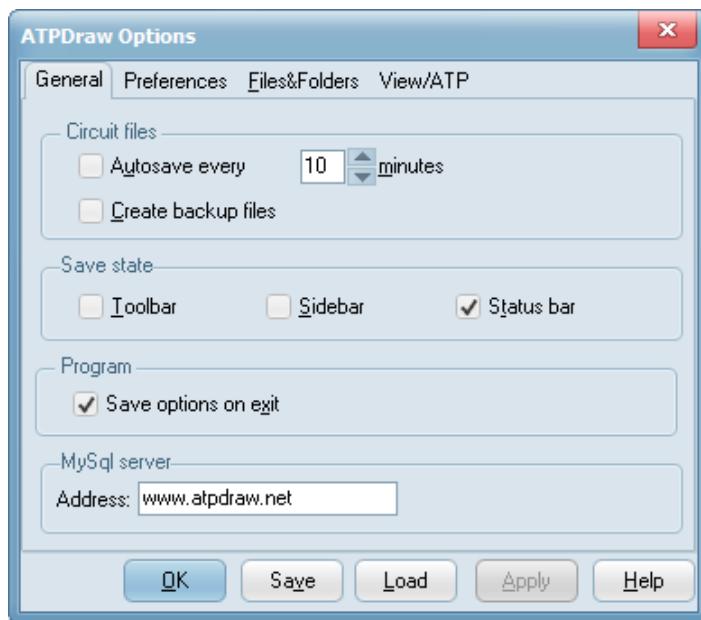


Fig. 4.45 - Customizing program options.

The *ATPDraw Options* dialog enables you to specify the contents of the ATPDraw.ini file without having to load and edit the file in a text editor. As shown on Fig. 4.45 this dialog box has four sub-pages: *General*, *Preferences*, *Directories* and *View/ATP*.

#### General

The *General* tab specifies the project file and ATPDraw main window options. The following list describes the available options:

Option	Description
Autosave every ? minutes	Saves all modified circuits to a separate disk file every specified interval of minutes. The file name is the same as the project file but with extension '.\$ad'. Modified state of the circuit window does not change as a consequence of

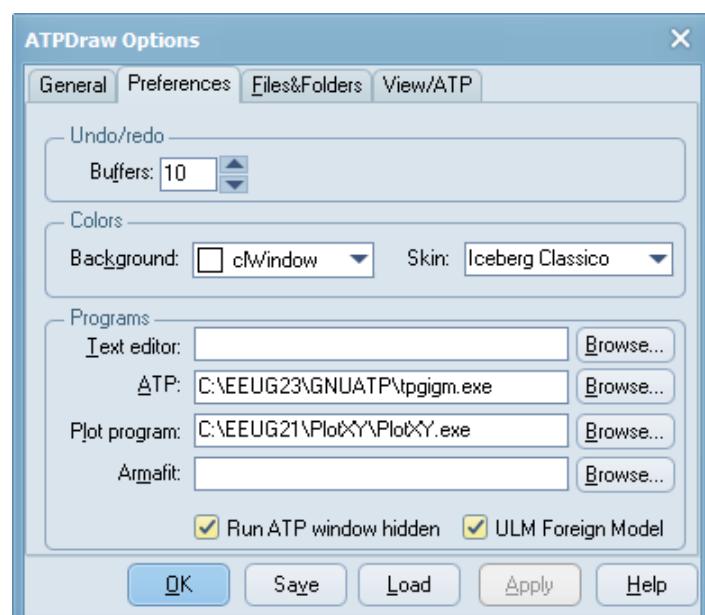
Create backup files	Changes the extension of the original project file to '.~ad' each time the circuit is saved. This option does not apply to autosave operations.
Save toolbar state	Records the current view state (visible or hidden) of the main window toolbar, so it can be redisplayed in the same state next time when ATPDraw is started.
Save sidebar state	Records the current view state (visible or hidden) of the main window sidebar, so it can be redisplayed in the same state next time when ATPDraw is started.
Save status bar state	Records the current view state (visible or hidden) of the main window's status bar, so it can be redisplayed in the same state next time when ATPDraw is started.
Save options on exit	Causes program options to be automatically saved to the initialization file when the program is terminated.

Note that the ‘save state’ options will have no effect unless program options are saved to the initialization file (ATPDraw.ini) by the *Save* command at the bottom of the *ATPDraw Options* dialog, or by selecting the ‘Save options on exit’ check box, or by the *Tools / Save Options* menu.

At the bottom of the *ATPDraw Options* dialog box the five buttons provide the following functionality:

Option	Description
OK	Stores current settings into program option variables, updates the screen and closes the dialog box. Changes made will only affect the current session.
Save	Saves the current settings to the ATPDraw.ini file.
Load	Loads settings from the ATPDraw.ini file.
Apply	Same as OK but does not close the dialog box.
Help	Displays the help topic related to the options on the current page.

Note that, if no initialization file exists, ATPDraw will create a new file in its installation directory when the user selects the *Save* button or the *Save Options* in the *Tools* menu.



### Preferences

On the *Preferences* page the user can set the size of undo/redo buffers, background colors and skins, specify the default text editor and command files to execute ATP-EMTP (TPBIG\*.EXE) and *Armafit* programs.

Setting up the ATP and Plot programs is now recommended from the ATP Connection Wizard.

Fig. 4.46 - Customizable program options on the Preferences page.

Option	Description
Undo/redo buffers:	Specifies the number of undo and redo buffers to allocate for each circuit window. Changing this option does not affect the currently open circuit windows; only new windows will make use of the specified value. Almost all object manipulation functions (object create, delete, move, rotate, etc) can be undone (or redone). These functions also update the circuit's modified state to indicate that the circuit needs saving. During an undo operation, the modified state is reset to its previous value, so if you undo the very first edit operation, the 'Modified' text in the status bar will disappear. Any operation undone can be redone. Since only a limited number of buffers are allocated, you are never guaranteed to undo all modifications. For example, if the number of undo/redo buffers is set to 10 (default) and eleven successive modifications to the circuit are made, the first modification can no longer be undone, and the modified state will not change until you save the circuit.
Background color:	Selects the background color of circuit windows. The color list provides available system colors, but you may customize your own from the Windows standard Color dialog displayed by the Custom button. The current color selection is shown in the box to the right of the Custom button.
Skin:	Iceberg Classico is the standard skin. This suffers from some bugs in Windows context menu, and in this case the Standard Windows skin can be chosen. The compiler offers many Skins, but they are not distributed in v7 due to increased ATPDraw.exe size.
Text editor program:	Holds the name and path of the text editor program to use for editing ATP-files (e.g. notepad.exe or wordpad.exe). If no program is specified (the field is empty), the built-in text editor will be used. Note that the program specified here must accept a filename on the command-line; otherwise the ATP-file will not be automatically loaded by the editor.
ATP:	Holds the ATP program command, which is executed by the run ATP command (or F2 key) at the top of the ATP menu. A batch file is suggested as default (runATP_S.bat for the Salford, runATP_W.bat for the Watcom and runATP_G.bat for the MingW32/GNU versions). Watcom/GNU versions can also be executed directly as %WATDIR% TPBIGW.EXE DISK \$\$ * -r or %GNUDIR%TPBIGG.EXE DISK \$\$ s -r where \$\$ replaces the %1 sign normally used in a batch file.
ARMAFIT:	Holds the name of the Armafit program used for NODA line/cable models. A batch file runAF.bat is suggested.
Plot:	Holds the preferred plotting command. Executed under ATP run Plot (F8).
Run hidden:	Run ATP as a hidden process without the DOS window.
ULM F M:	ULM Foreign model is available.

## Files&Folders

The following table describes the available options on the *Directories* page:

Option	Description
Project folder	The directory where ATPDraw stores the project files (.acp).
ATP folder	Specifies the directory in which .atp files are created. This is also the default Result Directory.
Web folder	Default directory for Web Download.
Model folder	Directory containing support (.sup) and model (.mod) files for MODELS components.

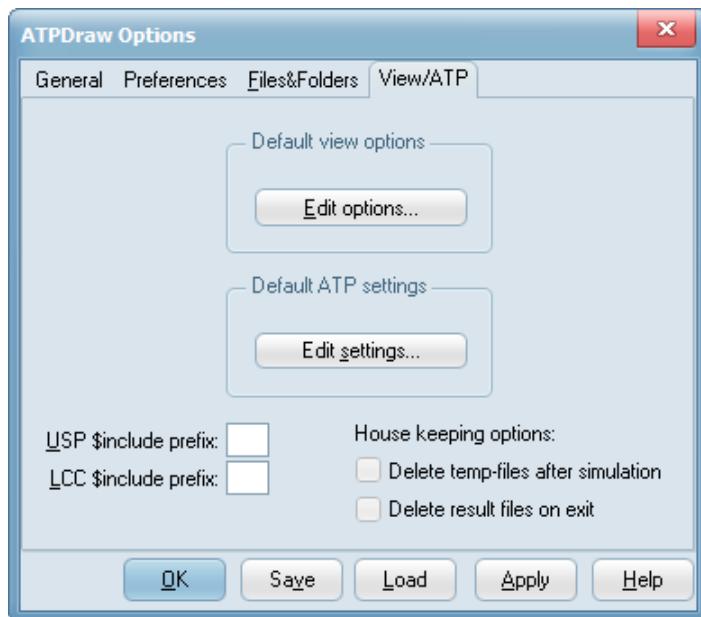
Help folder	The user can write help text files for instance resistor.txt (same name as the support file and extension txt) and store it in this folder. It will then automatically be added after the standard help text.
User spec. folder	Directory containing support (.sup), library (.lib) files for user specified components.
Line/Cable folder	Default folder for the line and cable models. This folder will contain .alc files (ATPDraw line/cable data), intermediate .atp and .pch files, and .lib files (include). If the .alc files are stored in that directory, the resultant .lib files used in \$Include statements in the final ATP input file are also stored in this directory. The \$Prefix/\$Suffix option should in this case be turned off. The Noda format in ATP does not allow to specify the full path for \$include files. Therefore, Noda lines (.alc files) must be stored in the same directory as the final ATP-file.
Transformer folder	The default folder for BCTRAN multi-phase, multi-winding linear transformer models. This folder will contain .bct files (ATPDraw Bctran data), intermediate .atp, .pch and .lis files. In addition the Hybrid transformer (XFMR) files could be stored here (.xfm).
Plugins folder	This is a user definable folder that appears in the bottom of the Selection menu. The user can add project files (acp) and sub-folders to this folder structure.

### **View/ATP**

Two groups of options can be specified in the *View/ATP* page. These are the *Default view options* and the *Default ATP settings*.

The *Edit options* button opens the *View Options* dialog, which enables you to specify view options to apply as default to all new circuit windows. Available options are described in section 4.2.4.6. Note that all circuit windows maintain their own set of view options, and only the new circuit windows you open will use the options specified here. To change the view options of an existing circuit window, select the *Options* item in the *View* menu (section 4.2.4.6).

The *Edit settings* button calls the *ATP settings* dialog described in section 4.2.5.1 of this manual. ATP settings specified here will be applied as default to all new project files. Note that all circuits have their own settings; stored together with the objects in the project files. The settings specified here will only be used by the new circuits you create. To customize ATP settings of an existing project select the *Settings...* item in the *ATP* menu or press *F3* function key.



The prefix tags are text strings added in front of the \$include file name. This is because User Specified (USP) and Line&Cable (LCC) components both have their \$include files dumped to the Result Directory (same as the ATP-file). In the case of duplicate file names in these categories, file conflicts will occur. The prefix option can then be used to avoid the conflict. If two UPS component have the same name for instance, the \$include file is anyhow forced to be equal.

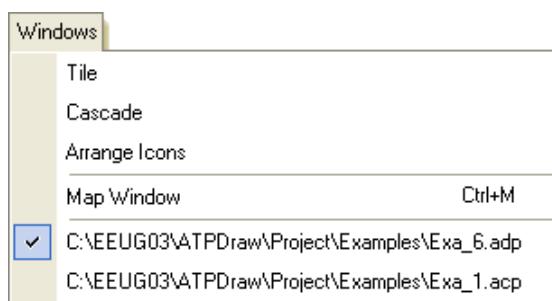
Fig. 4.47 - Setting default view and ATP options.

The House-keeping options delete temporary files after the simulation or exit. In the case of debugging a Line&Cable model the *Delete temp-files after simulation* option should not be checked.

#### 4.2.7.7 Save Options

Saves program options into the ATPDraw.ini. This file is normally located in the program installation directory and can be used to store default options and settings.

#### 4.2.8 Window



The *Window menu* contains options for activating or rearranging circuit windows and showing or hiding the *Map window*.

Fig. 4.48 - Supported options on the Window menu.

##### **Tile**

The *Tile* command arranges the circuit windows horizontally in equal size on the screen. To activate a circuit, click the title bar of the window. The active circuit window is marked by a ✓ symbol in front of the circuit file name.

##### **Cascade**

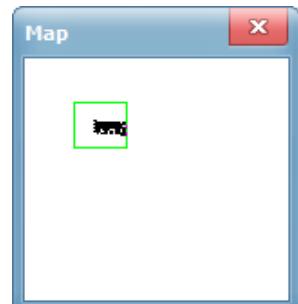
The *Cascade* command rearranges the circuit windows so that they overlap such a way that the title bar remains visible. To activate a circuit, click the title bar of the window.

## Arrange Icons

The *Arrange Icons* command arranges the icons of minimized circuit windows so that they are evenly spaced and don't overlap.

### 4.2.8.1 Map Window

The *Map Window* command (Shortcut: *Ctrl+M*) displays or hides the map window. The map window is a stay-on-top style window, meaning that it will always be displayed on top of all other windows. You can show or hide the map by pressing the *Ctrl+M* character of the keyboard to enable it when you need it or hide it when it conceals vital circuit window information.



The map window displays the entire contents of the active circuit. The circuit window itself is represented by a map rectangle and the circuit objects are drawn as black dots.

Fig. 4.49 - Map window.

When you press and hold down the left mouse button in the map rectangle, you move the display of the circuit world continuously. If any circuit objects are currently selected when you reposition the map rectangle, they remain in the same position in the circuit window. This functionality can be used to quickly move a collection of objects a relatively large distance.

### 4.2.9 Help

The *Help menu* contains options for displaying the help of ATPDraw, and the copyright and version information. The help file *ATPDraw.chm* is distributed with ATPDraw and it follows the compressed HTML standard compatible with Windows Vista.

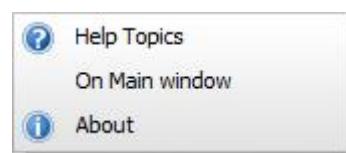


Fig. 4.50 - Help menu.

ATPDraw's HTML help is displayed in a standard Windows dialog, which provides indexed and searchable help on all ATPDraw dialogs and options.

#### 4.2.9.1 Help Topics

The *Help Topics* command invokes the MS-Windows standard help dialog box. Several links and a relatively large index register support the users in searching. Selecting the *Contents* tab you get a lists of available help functions as shown on Fig. 4.51.

This page allows you to move through the list and select an entry on which you need help. To display an entry, select one from the list by a simple mouse click and press *Display*, or double click on the entry with the mouse.

*Index* and *Find* tabs can be used to get help by the name of a topic. E.g. if you ask for help on topics "Circuit Window" type this phrase into the input field of the *Index* page and press the *Display* button. The ATPDraw help file consists of 136 topics.

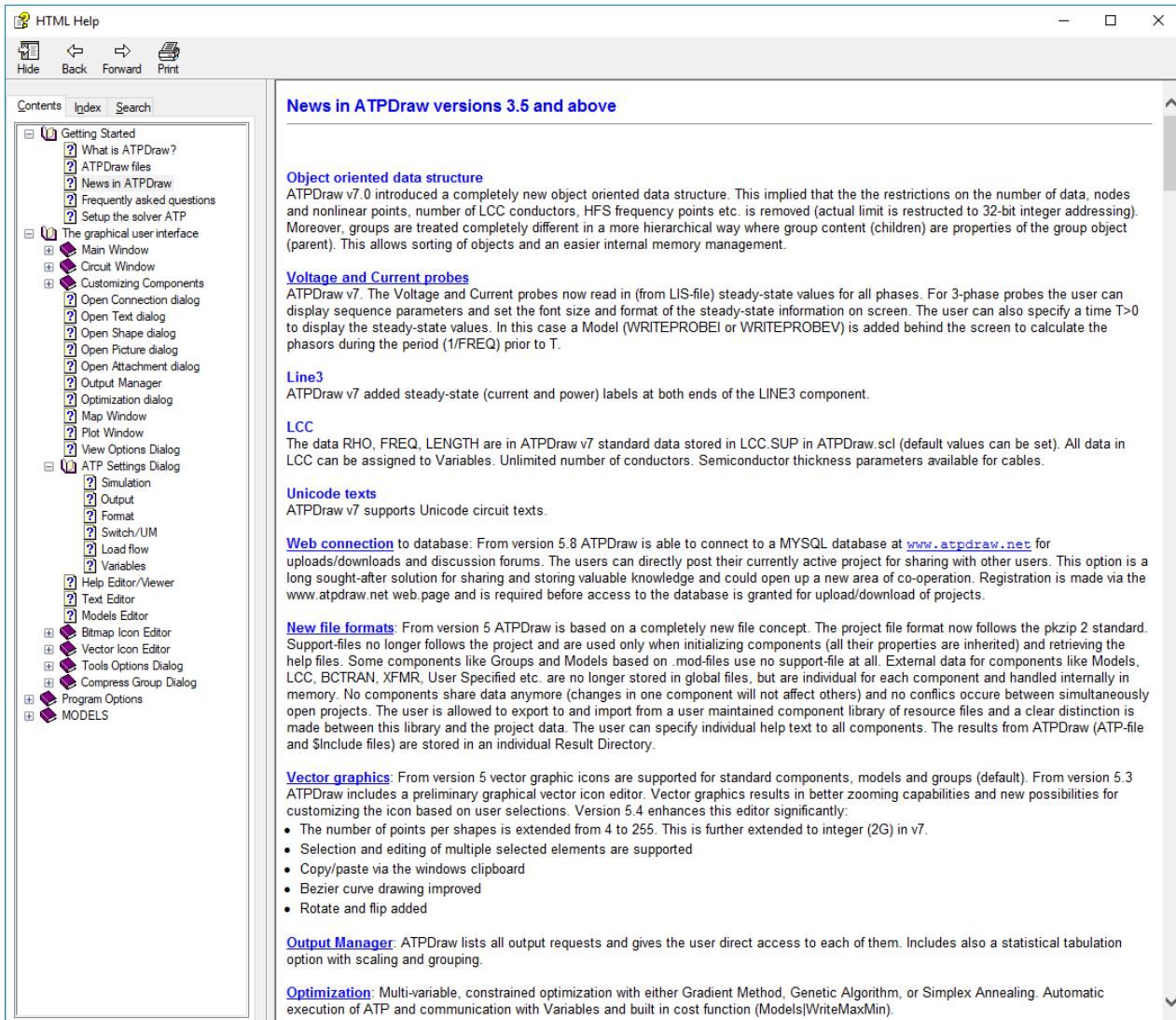


Fig. 4.51 - HTML help of ATPDraw.

#### 4.2.9.2 On Main Window

The menu item *On Main Window* displays help about the ATPDraw main window.

#### 4.2.9.3 About ATPDraw

Selecting this menu item shows the ATPDraw copyright information and the program version used.

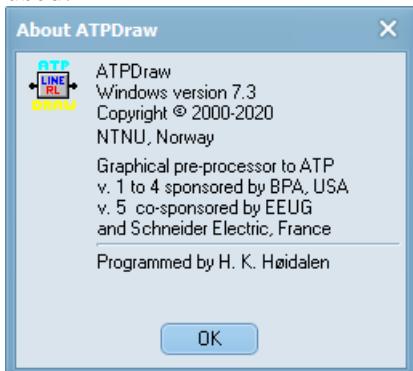
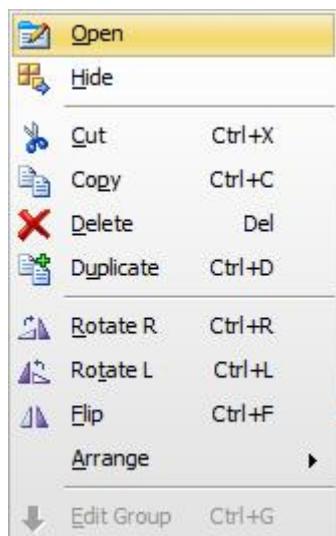


Fig. 4.52 - About window of ATPDraw.

### 4.3 Shortcut menu

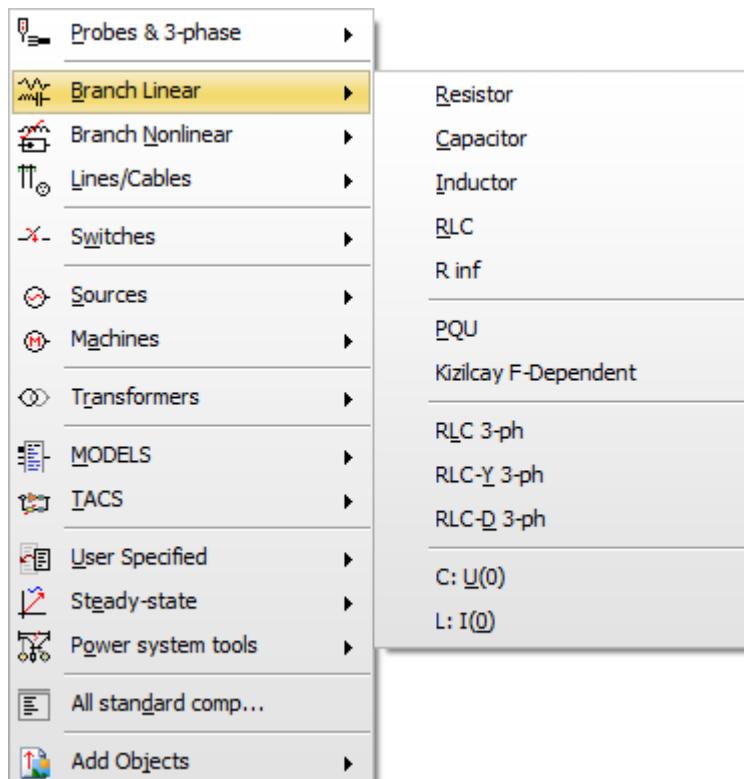
The *Shortcut menu* provides access to the most frequently used object manipulation functions. To show the shortcut menu, click the right mouse button on an selected object or a selected group of objects in the circuit window. Most of the items on this menu are identical with that of the *Edit menu* (section 4.2.2). The *Open* menu item at the top of the menu is an addition to these normal edit functions. If this command is performed on a single object, the *Component* dialog box appears. If you select this command for a group of selected objects, the *Selection* dialog appears.



- |                           |  |
|---------------------------|--|
| <b>Open:</b>              | Enables the component customization by showing the Component dialog or the Group dialog when several objects are selected. |
| <b>Hide:</b>              | Hides the selected object(s)   |
| <b>Cut, Copy, Delete,</b> | Provides access to the standard clipboard functions  |
| <b>Duplicate:</b>         |  |
| <b>Rotate, Flip:</b>      | Rotates and flips the selected object(s)   |
| <b>Arrange:</b>           | Move object forward or backward.<br>Objects in front are selected first and comes first in the ATP-file.                   |

Fig. 4.53 - Available options in the Shortcut menu.

### 4.4 Component selection menu



The *Component selection menu* provides options for inserting new components into the circuit window. This menu is normally hidden. To open it you must click on the right mouse button in an empty area of the circuit window. The component selection menu collects all the available circuit objects of ATPDraw in a structured way as shown in Fig. 4.54. After selecting a component in one of the floating menus, the selected object is drawn in the circuit window.

A complete list of all components comes in chapter 4.15.

Fig. 4.54 - Component selection menu.

The upper section of the menu provide access to the probe, splitter and transposition and reference objects, the next seven to many standard ATP components: linear and nonlinear elements, lines

and cables, switches, sources, electrical machines and transformers. The next section is dedicated for the control systems MODELS and TACS components. User specified objects and Frequency dependent components for Harmonic Frequency Scan (HFS) studies are accessible in the next group followed by the Power System Toolbox with its phasor calculators and protective relays etc.. Towards the end comes a list of all the standard supported components (for instance older component replaced by new versions). The final menu item called Plugins points to a user defined folder structure for import of project files (sub-circuits).

#### 4.5 Component dialog box

After selecting a component in the *Component selection menu* the new circuit object appears in the middle of the circuit window enclosed by a rectangle. Click on it with the left mouse button to move, or the right button to rotate, finally click in the open space to unselect and place the object. The *Component* dialog box appears when you click the right mouse button on a circuit object (or double click with the left mouse). Assuming you have clicked on the icon of an RLC element, a dialog box shown in Fig. 4.55 appears. These dialog boxes have the same layout for all circuit objects except probes, which can be edited from the *Probe* dialog box.

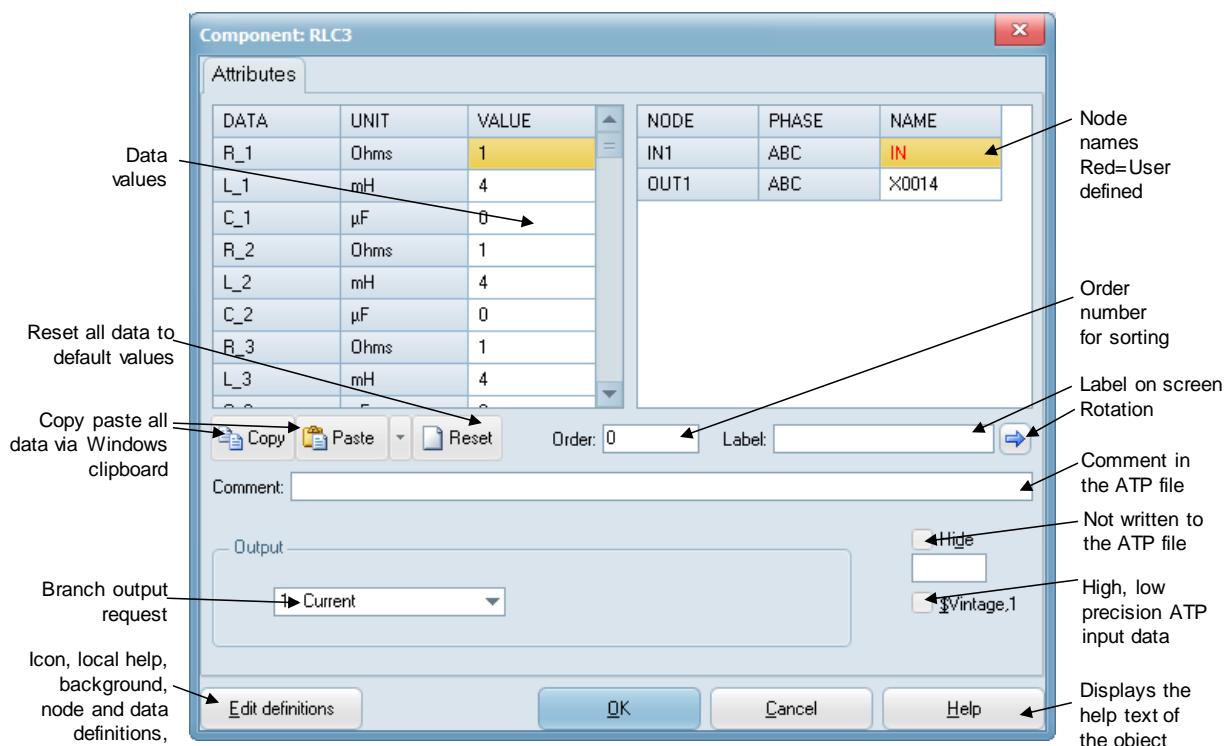


Fig. 4.55 - The Component dialog box.

Component data can be entered in the *Value* field of the *Attributes* page. The *Node*, *Phase* and *Name* fields are initially empty and you can enter node names in the *Name* field (without phase extensions 'A'..'Z'). You have to run *ATP/Sub-process/Make node names* or (*ATP/run ATP*) to obtain the ATPDraw specified node names.

Numerical values in the data input fields can be specified as real or integer, with an optional exponential integer, identified by 'E' or 'e'. A period '.' is used as decimal point. Many data parameters have a legal range specified. To check this legal range, place the input caret in a data field and press the *Ctrl+F1* keys. If you specify an illegal value, an error message is issued when

you move to another data field, or select the *OK* button. The legal range can be set under *Edit definitions*. Instead of a value you can also assign a 6 (or less) character text string as input data for most of the standard components. This requires the *Param* property of the data to be set to unity (see *Edit definitions*) unless the *Internal Parser* is used. Numerical values can later be assigned to these variables under *ATP/Settings/Variables* using the *Internal Parser* or *\$PARAMETER* feature of ATP-EMTP (see in 4.2.5.1).

Just below the node input column, there is an *Order* input field. It can be used later as optional sorting criteria (the lowest order number will be written first in the ATP-file) on the *ATP / Settings / Format* page. The Sidebar's Project-page allows direct sorting in the object inspector tree.

The content of the *Label* input text field is written on the screen. The visibility of the component label is controlled by the *Labels* option in the *View / Options* dialog box. The label is movable and directly editable on the screen. The font of the Label is controlled via *View/Set circuit font*. The component dialog box has a *Comment* input text field. If you specify a text in this field, it will be written to the ATP-file as a comment (i.e. as a comment line before the data of the object).

Many standard components such as branches, non-linear, switches and transformers contain an *Output* section for setting the branch output request in a combo box. Possible values are Current, Voltage, Current&Voltage, Power&Energy or none.

Like the *Order*, *Label* and *Comment* fields, the *Hide* button is common to all components. Besides checking *Hide*, the user can also specify a Variable and if its value is positive the component becomes hidden. Hidden components are not included in the ATP-file and are displayed as light gray icons in the circuit window. All components where the high precision format is available has a *\$Vintage, 1* check button in the component dialog box. It is thus possible to control the precision format for each individual component. Selecting *Force high resolution* under the *ATP/Settings/Format* page will overrule the individual setting and force *\$Vintage, 1* for all components if possible. The components User specified, Models and Groups has also a *Protect* button for password protection of their content.

The *OK* button will close the dialog box and the object data, and all properties are updated in the data structure. Then the red drawing color of the object icon will be turned off, indicating that the object now has user specified data. When you click on the *Cancel* button, the window will be closed without updating. The *Help* button calls the *Help Viewer* to show the help text of the object. Further help about the *Component* dialog is also available through the Windows standard HTML help system of ATPDraw if you press the *F1* key.

The non-linear components (non-linear branches, saturable transformers, TSWITCH, and TACS Device 56) have a *Characteristic* page too, as shown in Fig. 4.56.

On the *Characteristic* tab of the dialog box, you define the input characteristic for non-linear components. Data pairs can be specified in a standard string grid. To add new points after the cursor position, click on *Add*. Delete the marked point by clicking on *Delete*. You can manipulate the order of points by the *Sort* button (the characteristic for non-linear components is automatically sorted after increasing x-values, starting with the lowest number) or the  $\uparrow$  and  $\downarrow$  arrows. The user can edit the data points directly any time.

It is possible to export the characteristic to an external file or to the Windows clipboard as text. The whole characteristic is copied (no marking is supported or required). You can also paste a characteristic from the clipboard. It is thus possible to bring an old .atp file up in a text editor,

mark the characteristic (the flag 9999 is optional) and copy it to the clipboard, then paste it into the characteristic page. The number of points will automatically be adjusted. Therefore, you do not have to click on *Add* or *Delete* buttons before pasting. ATPDraw uses fixed format 16 character columns to separate the numbers. Note! Pasting in from a text file with 'C' in the first column is not possible; Delete leading 'C' characters first.

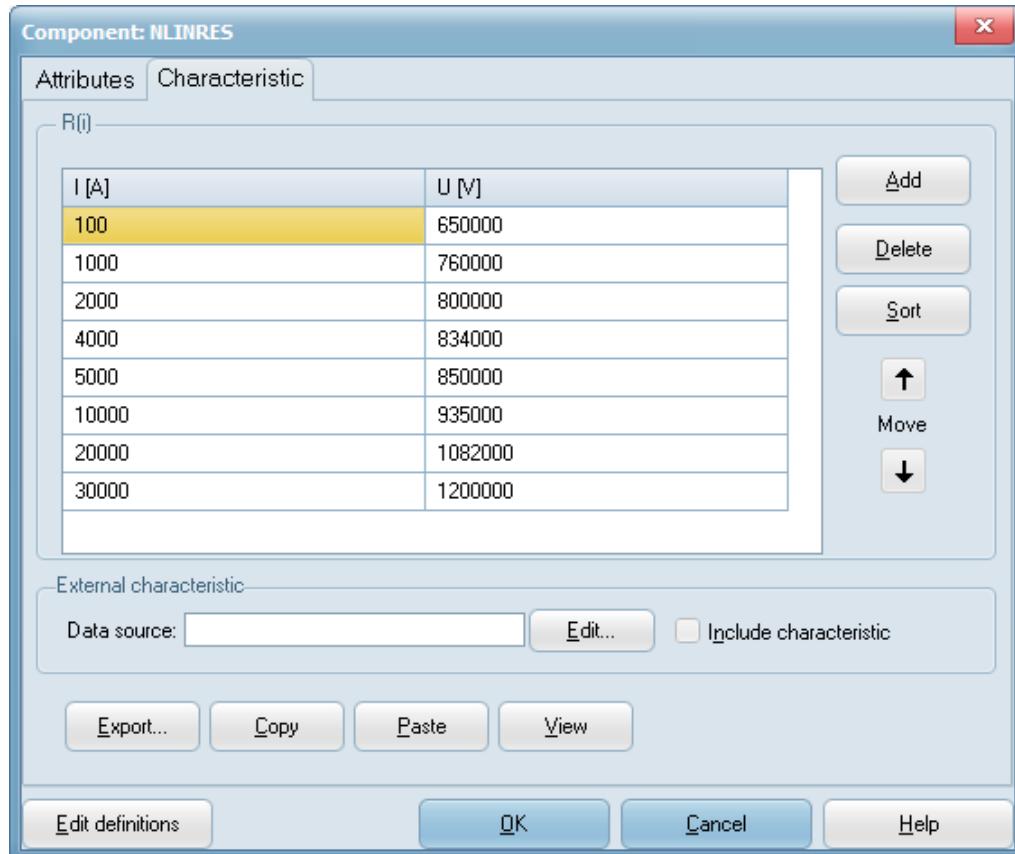


Fig. 4.56 - The Characteristic page of non-linear components.

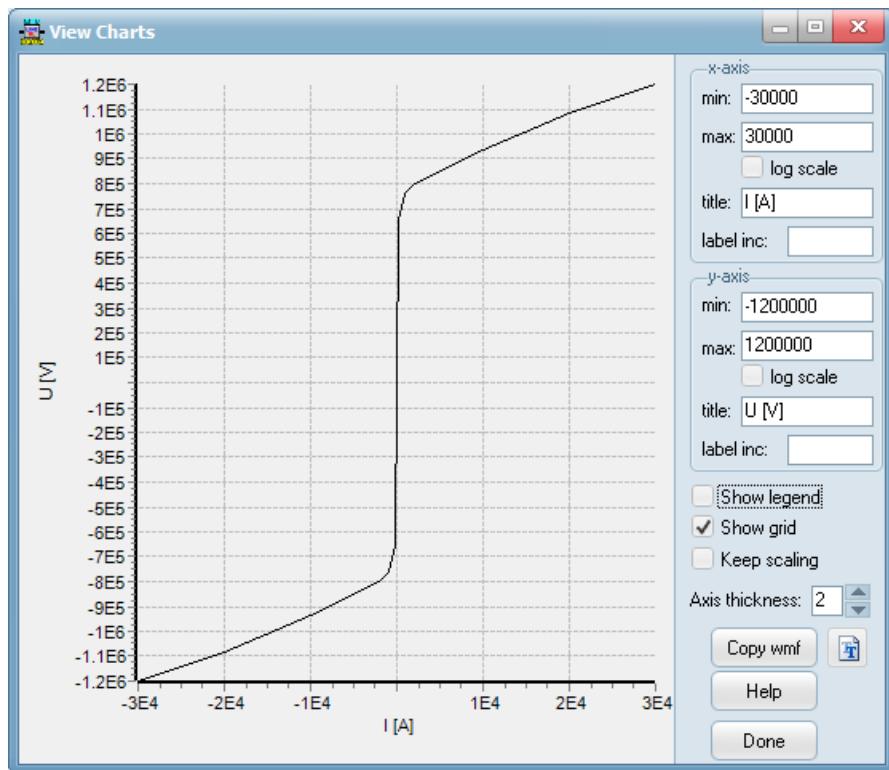


Fig. 4.57 - The View nonlinearity window.

The *External characteristic* section at the bottom of the page contains a *Data source* field where you can specify the name of a standard text file containing nonlinear characteristic. If the '*Include characteristic*' button is checked, this file will be referenced in the \$INCLUDE statement in the ATP-file rather than including each of the value pairs from the points table. ATPDraw reads the specified file into memory and inserts it directly in the final ATP file.

The nonlinear characteristic specified by the user can be displayed by clicking on the *View* button. In the *View Nonlinearity* window (Fig. 4.57) the min and max axis values are user selectable as well as the use of logarithmic scale (if min>0). It is possible to left click and drag a rectangle for zooming. Click right to restore. The Add (0,0) check box will add the origin point, and *1st quad* will display only the first quadrant. It is also possible to copy the graphic to the Windows clipboard in a metafile format with *Copy wmf*. Selecting *Done* will close the nonlinearity display.

The following components deviate somewhat from the above description and will be referenced in the Advanced part of this Manual:

- Saturable 3-phase transformer (SATTRAFO)
- Universal machine (UM\_1, UM\_3, UM\_4, UM\_6, UM\_8)
- Statistical switch (SW\_STAT)
- Systematic switch (SW\_SYST)
- Harmonic source (HFS\_SOUR)
- BCTRAN transformer (BCTRAN3)
- Line/Cable LCC objects (LCC)
- Windsyn embedded component (UMIND, UMSYN)
- Hybrid Transformer (XFMR)
- Models&Type 94

Depending on the type of component opened, the group box in lower-left corner of the *Attributes* page may display additional options:

- a) For Models you can enter the editor for inspecting or changing the Models text. In addition, you can specify a *Use As* string and defined the output of internal variables RECORD. The WriteMaxMin class of models have options to View extracted data.
- b) For the Fortran TACS components ATPDraw provides an extra OUT field here to specify the Fortran expression. Some TACS transfer functions and devices also have options to turn on/off inputs visually.
- c) For user specified components you specify the name of the library file in the *\$Include* field. If *Send parameters* option is selected, the *Internal phase seq.* controls how the node names are passed. i.e. unselect this option if your library file expects 5-character 3-phase node names. If the library file name does not include a path, the file is expected to exist in the /USP folder.

#### 4.6 Connection dialog box

The Connection dialog box appears if you draw a Connection between a single-phase node and a multi-phase node or double click on a Connection. This dialog allows you to select the number of phases in the Connection and the phase number of a single-phase Connection (*Phase index*). A pure single-phase connection between two single phase nodes should have the *Phase index* 0-@. You can also select the Color of the Connection and a text *Label* which can be displayed on screen (in rotated options). In addition, you can choose to *Hide* the connection and in this case the connection do not affect the node names. A special option is to force the *Node dots* on regardless of the Node dot size set in the main menu for visual clarity. 3-phase connections have a *kV* option to force its color to follow the *View/Options/PS color* settings.

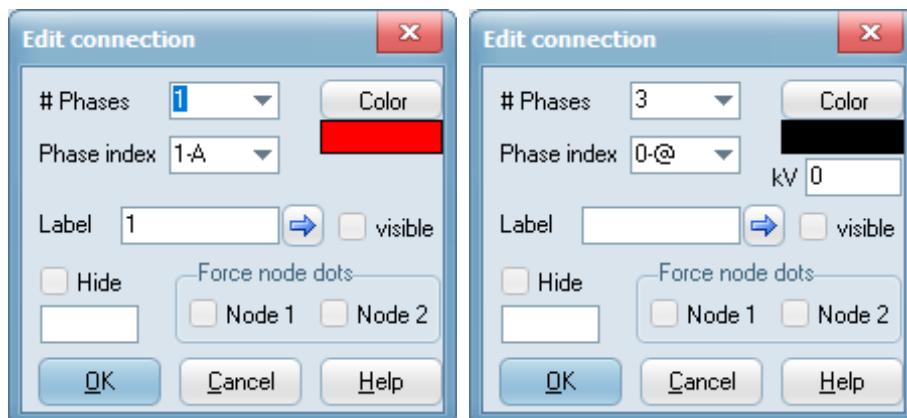
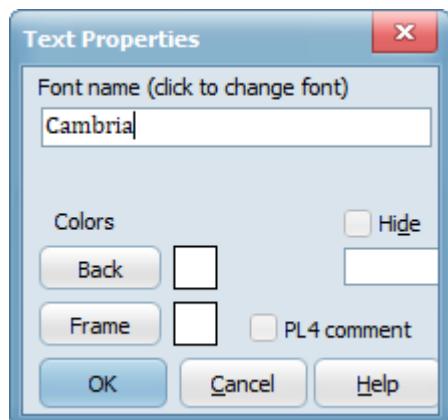


Fig. 4.58 – Connection dialog box

## 4.7 Text dialog box



The *Text dialog box* appears if you right click or double click on a *Circuit Text* (not a *Label* or *Node Name*). In this dialog you can specify the *Font*, *Size* and *Colors* of the font used in the *Circuit Text*. You can edit *Circuit Text*, *Label* and *Node Names* directly in the Circuit Window by a left, simple click on them. *Circuit Texts* can hold multiple lines and the entire text uses the same font. You can move the *Circuit Texts*, *Labels*, and *Node Names* by left click and hold. Press the *Alt* key to avoid selecting other circuit objects.

Fig. 4.59 – Circuit Text dialog box

## 4.8 Shape dialog box

Shapes can be lines, rectangles, ellipses, and arrows. This are used for documentation purposes and to illustrate/highlight parts in a circuit. The objects are inserted via *Add objects* under *Tools* or from the *Selection menu*. Click left mouse cursor to place start point and click left mouse button again to finish (click right mouse button to cancel). Right click on the object to change its pen (shape outline) and brush (shape fill) with color, style, and fill pattern. If gradient fill is selected a second brush color option appears. Arrows have selection of arrow styles instead of brush styles.

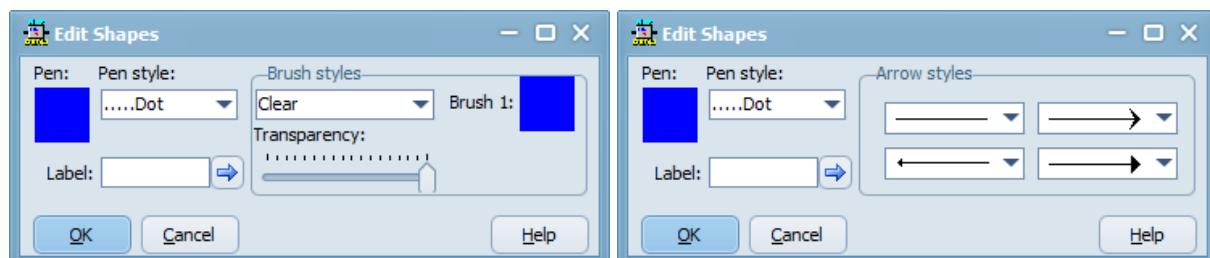


Fig. 4.60 – Circuit Shape dialog box

## 4.9 Picture dialog box

Pictures can be inserted for documentation purposes. The bitmap formats png, jpeg, bmp and windows meta file vector graphics wmf are supported. The user can paste in pictures from the clipboard or open image files. The *Width* in pixels (100% zoom) can be set and the aspect ratio is always maintained. *Transparent* means that the lower-left color is turned into transparent color. If *Rotate* is checked the image is rotatable as any other object, but this does not apply to wmf-files unless *Store as bmp* is selected. Pictures also have a rotatable caption.



Fig. 4.61 – Circuit Picture dialog box

#### 4.10 Attachment file dialog box

Attached files can be added via *Add objects* from *Tools* or the *Selection menu* but Drag&drop from the file explorer is also supported. The various file types get unique symbols in the circuit window. Text-files for ATP (atp, dat, pch, lib, lis) will get an ‘ATP’ symbol, while many Office files and PL4-files will have a specialized symbol. The Attachment File dialog (right click on the objects) have a *Name* from its source, information about the file size (uncompressed, the file is stored compressed in memory and project), an option to *Export* the file and a button to *View* it. If *View* is clicked the attachment is uncompressed and opened in the standard viewer for the file type. This applies to office files, text/ATP files and PL4-files. Text file viewer and PL4-viewer is set up under *Tools/Options/Preferences* and this follows what is used otherwise in the project.

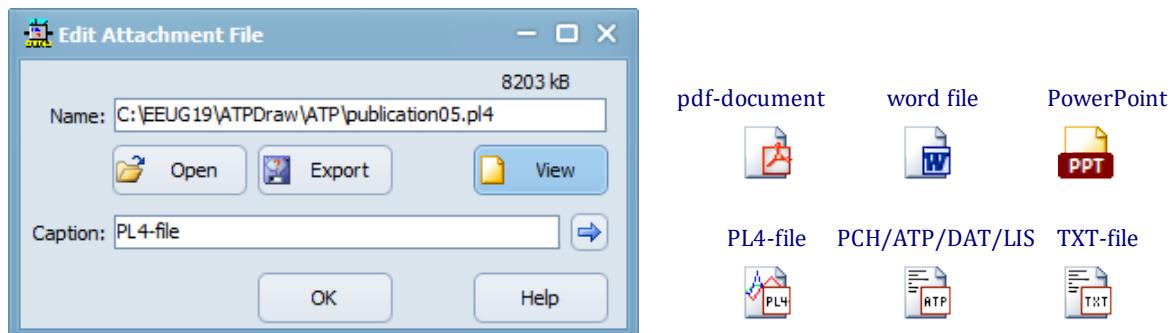


Fig. 4.62 – Circuit File dialog box (left). File type circuit appearance (right).

#### 4.11 Plot dialog box

Plot objects are also inserted from *Add objects* under *Tools* or the *Selection menu*. These objects can read PL4-files of type NEWPL4=0 or 2 and include the plotted results directly in the circuit (as well as inside groups). Direct embedded plotting has an advantage since it is directly coupled

to the simulation number in multiple-run cases. It also helps to document the result, ensure reproducibility, and enables quality plots for publication purposes. However, it could also be somewhat risky especially for large PL4-files (more than million samples). It should be used with care also because this is a new feature in ATPDraw v7.1. The user first specifies the number of plots in *#Plots*, then selects the *Series name* in the string grid above. A drop-down menu with all available series following the PlotXY syntax and the listing in the Output manager (F9) appears when clicking in the Series name column. Series containing ATPDraw generated node names (XXnnnn, XnnnnA/Z) involve a possible future naming confusion and should be avoided. The user should consequently name all nodes involved. The colors option follows the standard Windows colors but selecting Custom will enable all possible colors for selection. The column *Run* indicates which multi-run case to use. In the example below the same output is studied for the first 5 runs. If run or scale are zero, 1 will be used. The curve is plotted as  $c(t)=PL4(t-Skew)\cdot Scale+Offset$ . The *Right* column is to assign a series to the right plot axis.

*Save plots in project* have three options. *Plot definition* will only save grid above, *+chart settings* will also save all Plot object settings (axis, zooming etc.) and *+data values* saves the actual data, so the curves displays immediately when loading the project. The data are stored with single precision (same as PL4) and compressed but beware of possible large project file sizes. *Draw reduced samples* reduces the accuracy somewhat but speeds up the drawing. *GDI+* draws more smooth curves. On the *Settings* page axis and panel can be adjusted as shown in Fig. 3.47. The *Advanced settings* brings up the extremely rich native chart setting dialog. This allows fine tuning of fonts, positions, and appearances, as shown in Fig. 3.48. The settings made in this dialog are also stored if *+chart settings* or *+data values* are selected.

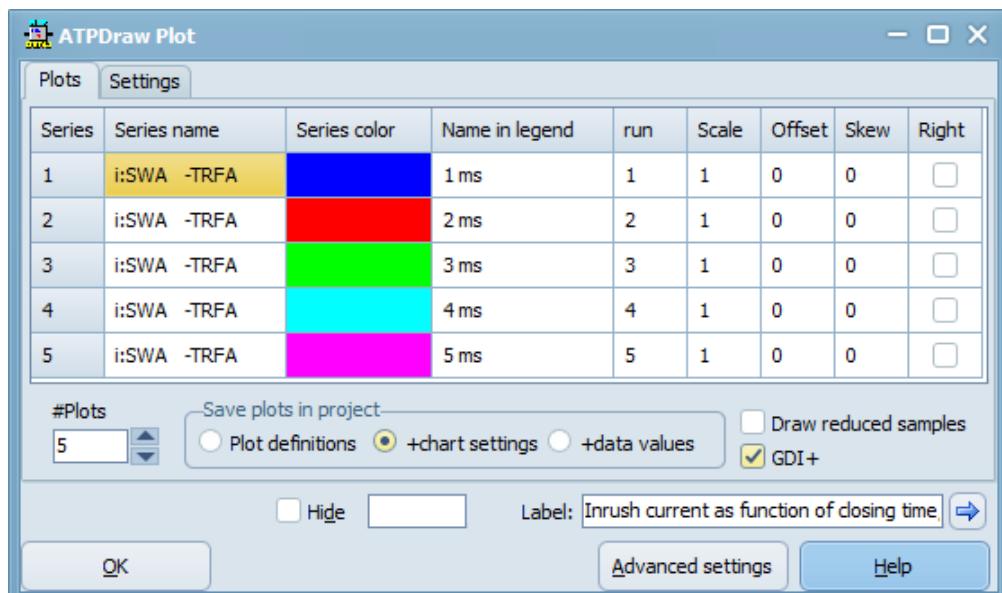


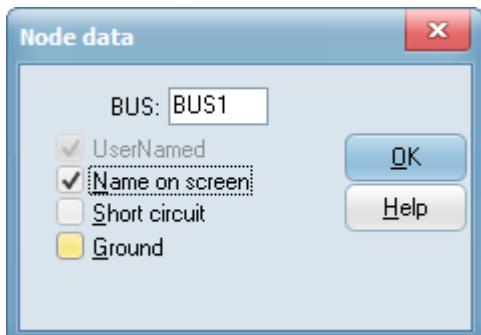
Fig. 4.63 – Circuit Plot dialog box

## 4.12 Node dialog box

In the *Node data* dialog box you specify data for a single component node. Input text in this dialog boxes should contain only ASCII characters, but characters like \* - + / \$ etc. should not be used. Avoid using space in the node name and lower-case letters, as well. The user does not need to give names to all nodes, in general. The name of the nodes without special interest are recommended to be left unspecified and allow ATPDraw to give a unique name to these nodes. The node dots given a name by the program are drawn in black, while those whose names were

specified by the user are drawn with red color.

There are four different kinds of nodes, each treated slightly different in this dialog box:



- 1) Standard and user specified nodes
- 2) MODELS object nodes
- 3) TACS object nodes
- 4) TACS controlled machine nodes

Fig. 4.64 - Node dialog box for standard components.

Parameters common to all nodes are:

**Name** A six or five (3-phase components) characters long node name. The name caption is read from the support file. If you try to type in a name on the reserved ATPDraw format (XX1234 for single phase or X1234 for three-phase nodes) you will be warned. Ignoring this warning can result in unintentional naming conflicts.

**UserNamed** This disabled checkbox shows whether this node name is specified by the user or ATPDraw. If the user wants to change a node name he should do this where the *UserNamed* box is checked. If not, duplicate node name warnings will appear during the compilation. Node with *UserNamed* set are also drawn with a red node dot.

**Name on screen** If checked, the node name is written on screen, regardless of the current setting of the Node names option in the *View | Options* dialog box.

The following list explains the type specific node parameters:

#### Standard and USP components:

**Ground** If checked, the node is grounded. A ground symbol appears for rotation of the graphical grounding symbol.

**Short circuit** Appears only for multi-phase nodes and if checked, the node becomes a single-phase node with all phases short-circuited.

#### MODELS node:

**Type** 0=Output.  
1=Input current (i)  
2=Input voltage (v)  
3=Input switch status (switch)  
4=Input machine variable (mach)  
5=TACS variable (tacs)  
6=Imaginary part of steady-state node voltage (imssv)  
7=Imaginary part of steady-state switch current (imssi)  
8=Output from other model. Note that the model that produces this output must be USED before the current model. This is done by specifying a lower Order number for the model and then select the *Sorting by Order number* option under *ATP | Settings / Format*.  
9=Global ATP variable.  
10=Variable from connected PL4-file.

#### TACS node:

**Type** 0=Output.  
1=Input signal positive sum up.  
2=Input signal negative sum up.  
3=Input signal disconnected. (necessary only if the node name is user specified)

TACS controlled synchronous machine type 58/59 node:

**Type**

- 0=No control.
- 1=D-axis armature current. Out.
- 2=Q-axis armature current. Out.
- 3=Zero-sequence armature current. Out.
- 4=Field winding current. Out.
- 5=D-axis damper current. Out.
- 6=Current in eddy-current winding. Out.
- 7=Q-axis damper current. Out.
- 8=Voltage applied to d-axis. Out.
- 9=Voltage applied to q-axis. Out.
- 10=Zero-sequence voltage. Out.
- 11=Voltage applied to field winding. Out.
- 12=Total mmf in the machines air-gap. Out.
- 13=Angle between q- and d-axis component of mmf. Out.
- 14=Electromagnetic torque of the machine. Out.
- 15=Not used.
- 16=d-axis flux linkage. Out.
- 17=q-axis flux linkage. Out.
- 18=Angle mass. Out.
- 19=Angular velocity mass. Out.
- 20=Shaft torque mass. Out.
- 21=Field voltage. In.
- 22=Mechanical power. In.

#### 4.13 Probe dialog box

Probes are components for output of node- or branch voltages, branch current or TACS values, and are handled differently than other components you open. In the *Probe dialog* you can specify the number of phases of a probe and which phases to produce output in the PL4-file. There are six different probes in ATPDraw:

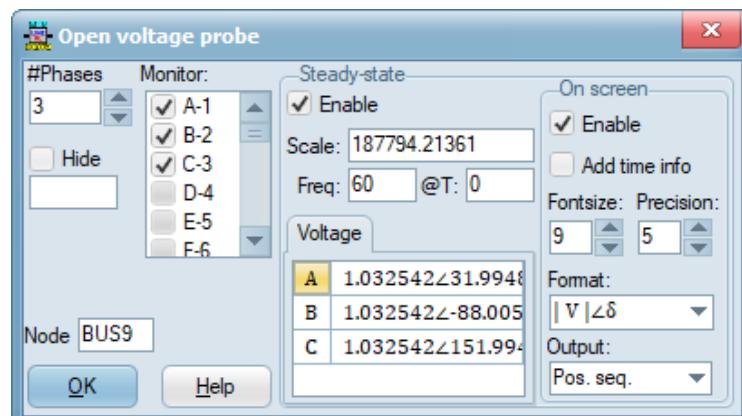


Fig. 4.65 – Probe dialog box for voltage probes.

The *Steady-state* option is only available for (Node) Voltage and (Switch) Current probes. ATPDraw reads the lis-file and identifies the steady-state ATP output. In *On screen* the user can specify how the steady-state information is shown on screen. If the @T value is larger than zero, ATPDraw will insert a hidden MODELS component that calculates that phasors for one period (1/Freq) prior to the specified time. For multi-phase nodes all phases are analyzed, for 3-phase nodes sequential output is possible. ATPDraw divide the steady-state value with the *Scale* factor (187794.21361=230000\*sqrt(2/3) gives pu of a 230 kV system) before displaying it on screen or only in the grid below.

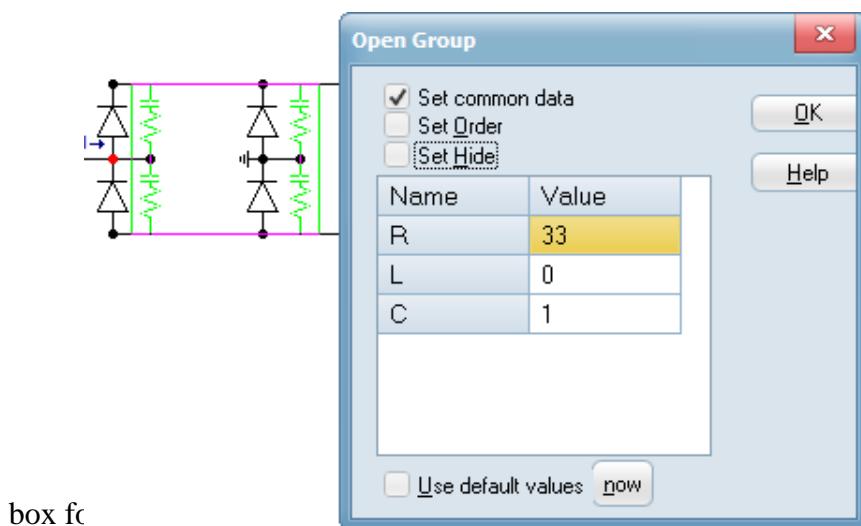
- Probe\_v: Node voltage output request.
- Probe\_b: Branch voltage output request.
- Probe\_d: Line voltage output request
- Probe\_i: Switch current output request.
- Probe\_t: TACS variable output request. Type33.
- Probe\_m: MODELS node output request.

For current probes the user can choose between current and power flow (active and/or reactive) for output while both are shown in the Current and Power grids. There is also an *Add current node* option (which is forced on for beyond T=0 outputs to assist the hidden MODELS component). This option confuses many users, but it means that two measuring switches are added in series and the middle node is displayed as a unique connection point. The rationale is to avoid confusion about which switch ATP uses for current measurement in the case of more than one switch/current probe are connected to the same node.

Note that the number of phases is critical for a current probe and this must match the circuit.

#### 4.14 Selection dialog

If you double-click in a selected group of objects, the *Selection dialog* will appear, allowing you to change attributes common to all components in that group, such as data values, *Order* number and *Hide* state. The common data parameters are listed in a dialog as of Fig. 4.63 where you can change the data for all the involved components, simultaneously. The data names from the definition properties are used to classify the data. If there are no common data, you can still select the type of component to open and set data for this type.



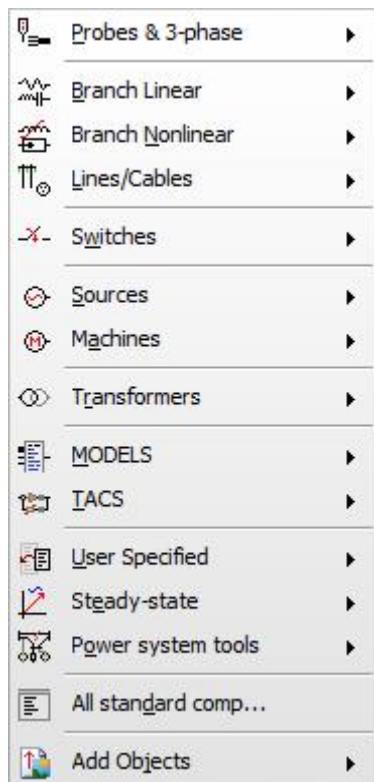
An alternative way to change the data parameter for several component simultaneously is to use *Variables* (see Fig. 4.18 in section 4.2.5.1).

Fig. 4.66 - Selection dialog

Every component has an *Order* number. By specifying a value in the *Order* field, all components in the selected group of objects are assigned the same number. The order number serves as an optional sorting criterion for the ATP-file (components with the lowest order number are written to the .atp file first if *Sort by Order* is checked). Consider direct ordering via sidebar, object inspector tree for more visual control.

The *Hide* state of multiple components can also be specified. Hidden components are not included in the ATP-file and are displayed as gray icons. You can also choose to reset to the default values inherited from the support files by clicking on the *now* button. Selecting the *Use default values* check box will cause default values to be loaded automatically next time the dialog box is opened.

## 4.15 Circuit objects in ATPDraw



The *Component selection menu* provides options for creating and inserting new components into the circuit window. This menu is normally hidden. To show and activate the menu, click the right mouse button in an empty circuit window space. Following a selection in one of the floating sub-menus, the selected object will be drawn where you clicked the mouse button in the active circuit window enclosed by a rectangle. You can move (left mouse click and drag), rotate (right mouse button) or place the object (click on open space).

The *Component selection menu* has several sub-menus; each of them include circuit object of similar characteristics as briefly described below:

Fig. 4.67 - Component selection menu.

### Probes & 3-phase

- Probes for node voltage-, branch voltage, current-, TACS, and Models output monitoring
- Various 3-phase transposition objects
- Splitter (coupling between 3-phase and single-phase circuits) and Collector.
- ABC/DEF Reference objects for specifying the master node for phase sequence

### Branches

- Branch linear: multi-phase and 3-phase non-coupled components. RLC.
- Branch nonlinear: multi-phase nonlinear R and L components, MOV. Type-93, 96 and 98 nonlinear inductors including initial conditions for the flux linkage.
- TACS controlled and time dependent resistor, inductor and capacitor.

### Lines/Cables

- Lumped, PI-equivalents (type 1, 2...) and RL coupled components (type 51, 52...)
- Distributed lines of constant, frequency independent parameters. Transposed (Clarke) up to 9-phases, untransposed 2 or 3-phase (KClee) line models.
- LCC, the user can select 1-9 phase models of lines/cables. In the input menu of these components, the user can specify a LINE CONSTANT or CABLE PARAMETER data case. The resulting include file contains the electrical model and the LIB-file is generated automatically if the ATP setup is correct. Bergeron (KClee/Clarke), nominal PI, JMarti, Semlyen and Noda models are supported. Templates, Sections and EGM.
- Read PCH-file. This is a module in ATPDraw to read the punch-files from Line Constants, Cable Constants or Cable Parameters and to create an ATPDraw object automatically (sup-file and lib-file). ATPDraw recognizes: PI-equivalents, KClee, Clarke, Semlyen, and JMarti line formats.

### Switches

- Time and voltage controlled. 3-phase time-controlled switch
- Diode, thyristor, triac
- Simple TACS controlled switch
- Measuring switch
- Statistic and systematic switches

### Sources

- AC and DC sources, 3-phase AC source. Ungrounded AC and DC sources.
- Ramp sources, sawtooth and pulse train
- Surge sources with front and half-value time fitting
- TACS controlled sources, AC modulated.

### Machines

- Type 59/58 synchronous machine, Type 56 induction.
- Universal machines (type 1, 3, 4, 6, and 8)
- Windsyn components

### Transformers

- Single phase and 3-phase ideal transformer
- Single phase saturable transformer
- 3-phase, two- or three-winding saturable transformer
- BCTRAN. Automatic generation of .pch file. 1-3 phases, 2-3 windings. Auto-transformers, Y-, and D- connections with all possible phase shifts. External nonlinear magnetizing inductance(s) supported.
- Hybrid Transformer (XFMR). Advanced topologically correct transformer with Test Report, Design data or Typical value input. 3-phase, 2-4 windings, Auto, Y, D and zigzag coupling.

### MODELS

- Under MODELS the user can either select a default model and write/update the script internally, or select an existing external model component by specifying a sup-file or a mod-file.
- Only input, output, data and variables declared in front of TIMESTEP, INTERPOLATION, DELAY, HISTORY, INIT and EXEC are recognized by ATPDraw when interpreting the model script and converting this to a component.
- Type 94: General, multi-phase type 94 component. Specify the type; THEV, ITER, NORT, NORT-TR and the number of phases. Specify a mod-file describing the Type-94 models component (templates available). The same rules as specified under MODELS apply.
- WriteMaxMin, Extract1, Extract3 are components that extract extremal values from the simulation and also support multiple runs. The Extract1 and Extract3 components also displays extremal values on screen and allows more control of start-stop and scaling.

### TACS

- Coupling to Circuit. Input to TACS from the circuit must be connected to this object.
- 4 types of TACS sources: DC, AC, Pulse, Ramp.
- Transfer functions: General Laplace transfer function. If the Limits are not specified or connected, no limits apply. First order dynamic icon with limits. Simple Integral, Derivative, first order Low and High Pass transfer functions.
- TACS devices. Type 50-66.
- Initial condition for TACS objects (Type-77)
- Fortran statements: General Fortran statement (single line expression). Simplified Math statements or Logical operators.
- Draw relations. Relations are drawn in blue and are used just to visualize connections between Fortran statements and other objects. Relations will not affect the ATP input file.

**User specified**

- Library: \$Include is used to include the lib-file into the ATP input file. The user must keep track of internal node names in the include file.
- Additional: Free format user specified text for insert in the ATP file. Selection of location.
- Single and 3-phase reference: These objects are not represented in the ATP input data file and serve only as visualization of connectivity.
- Files: Select a support file (sup). Import a lib-file (Data Base Module format) via the Edit menu. \$Include is used to include the user specified lib-file into the ATP input file and pass node names and data variables as parameters.

**Steady-state components**

- RLC Phasor component only present at steady state
- Harmonic source for Harmonic Frequency Scan studies
- Single and 3-phase frequency dependent loads in CIGRÉ format
- Single phase RLC element with frequency dependent parameters
- Load flow components PQ, UP, TQ

**Power system tools**

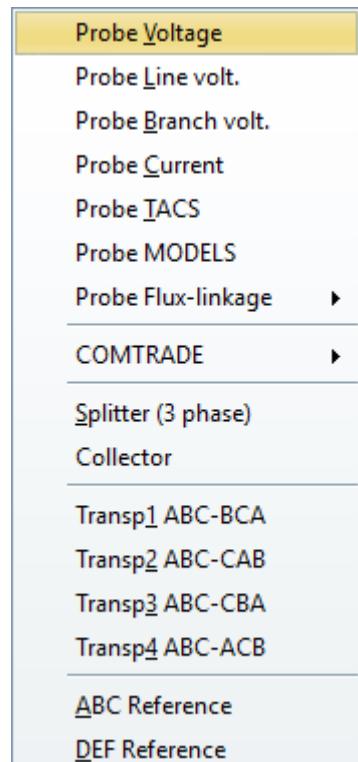
- 3-phase basic components LINE3, BUSS3, LOADPQ, CTRLCB
- Phasor, RX and power calculators
- Filters, RMS, Phase-locked-loops
- Protective relays components
- TACS block for RMS, 012, PQ0, AB conversion.

**All standard comp...**

- Complete list of standard components in alphabetical order sorted by support file names.

**Plugins**

- User defined folder structure containing project files (.acp) for import.

**4.15.1 Probes & 3-phase**

The menu *Probes & 3-phase* appears when the mouse moves over this item in the *Component selection menu* or when the user hits the *P* character.

Probes are components for monitoring the node or branch voltage, branch current or TACS values. In the *Open Probe dialog* you can specify the number of phases to connect to and select phases to be monitored.

Fig. 4.68 – Drawing objects on the Probe & 3-phase menu.

**Probe Volt, PROBE\_V**

-  Selecting this item draws the voltage probe to specify a node voltage-to-ground output request in the ATP-file.

**Probe line volt., PROBE\_D**

-  Selecting this item draws the line voltage probe to specify voltage difference output requests in the ATP-file. Useful for 3-phase circuits where the user can specify A-B, B-C, C-A or A-C, B-A, C-B voltages.

**Probe Branch volt., PROBE\_B**

-  Selecting this item draws the branch voltage probe to specify a branch voltage output requests in the ATP-file.

**Probe Curr, PROBE\_I**

-  Selecting this item inserts a current probe (measuring switch) into the circuit to specify current output request in column 80 in the ATP-file. The number of monitored phases are user selectable. *Add current node:* Two switches in series. Middle node available.

**Probe Tacs, PROBE\_T**

-  Selecting this item draws the Tacs probe to specify signal output and inserts TACS Type-33 object into the ATP-file.

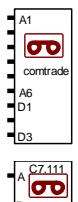
**Probe Model, PROBE\_M**

-  Selecting this item draws the Model probe which can be added to Models output nodes. Inserts RECORDS cards into ATP-file.

**Probe flux-linkage, FLUX3, FLUX3N, FLUX1**

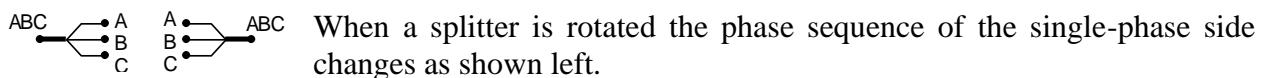
-  Selecting this item draws the flux-linkage probe (these components are groups located in the /GRP folder). Inside the group is a Model that integrates the input from a specified time instant, that can be set negative to initialize the integration from steady-state.

**COMTRADE, COMTRADE, COMTRADE1, COMTRADE2**

-  These objects create COMTRADE (.dat, .cfg) or MATLAB (.mat) automatically following the simulation. The objects offer flexible user selectable sampling rate and specification of channel name and scaling. The COMTRADE object can be connected to 3-phase nodes for direct loading, while the two others require packing objects/merging units. Maximum 26 analog and 26 digital channels. Example of packing objects (user specified MODELS) found in Exa\_26.acp.

**Splitter, SPLITTER**

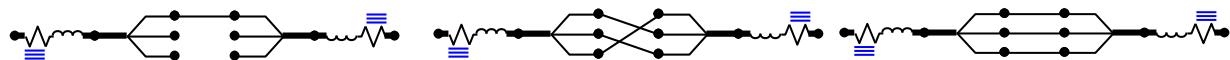
-  The *Splitter* object is a transformation between a 3-phase node and three 1-phase nodes. The object has 0 data and 4 nodes.



 If a name is given to the 3-phase node, the letters *A B C* are added automatically on the single-phase side of splitters.

Note! Do not give names to nodes at the single-phase side of splitters and do not connect splitters

together on the single-phase side (except all three phases). I.e. next two examples are illegal!



Disconnection is illegal this way! Transposition is illegal this way! This is legal, however.

### **Collector, COLLECTOR**

- The *Collector* object is a component with a single multi-phase node. It is useful in Compress, since only components can have external nodes, not connections.

### **Transp 1 ABC-BCA ...Transp 4 ABC-ACB**

*Transposition* objects can be used to change the phase sequence of a 3-phase node. The following transpositions are supported:

- Change the phase sequence from *ABC* to *BCA*.
- Change the phase sequence from *ABC* to *CAB*.
- Change the phase sequence from *ABC* to *CBA*.
- Change the phase sequence from *ABC* to *ACB*.

Handling of transpositions for objects with several 3-phase nodes is controlled by the circuit number *Kind* under *Library / Edit/New USP / Nodes* (see in 4.2.6.2). 3-phase nodes having the same *Circuit* property will receive the same phase sequence.

### **ABC reference , ABC**

- When attached to a 3-phase node in the circuit this node becomes the “master” node with phase sequence *ABC*. The other nodes will adapt this setting.

### **DEF reference, DEF**

- When attached to a 3-phase node in the circuit this node becomes the “master” node with phase sequence *DEF*. The other nodes will adapt this setting. A combination of *ABC* and *DEF* references is possible for e.g. in 6-phase circuits.

## **4.15.2 Branch Linear**

This sub-menu contains linear branch components. The name and the icon of linear branch objects, as well as a brief description of the components are given next in tabulated form. Data parameters and node names to all components can be specified in the *Component* dialog box (see Fig. 4.55 ), which appears if you click on the icon of the component with the right mouse button in the circuit window.

<u>Resistor</u>
<u>Capacitor</u>
<u>Inductor</u>
<u>RLC</u>
R inf
<hr/>
PQU
Kizilcay F-Dependent
<hr/>
RLC 3-ph
RLC-Y 3-ph
RLC-D 3-ph
<hr/>
C: U(0)
L: I(0)

The *Help* button on the Component dialog boxes calls the *Help Viewer* in which a short description of parameters and a reference to the corresponding ATP Rule Book chapter is given. As an example, Fig. 4.70 shows the help information associated with the ordinary RLC branch.

Fig. 4.69 – Supported linear branch elements.

Selection	Object name	Icon	ATP card	Description
<i>Resistor</i>	RESISTOR	-VV-	BRANCH type 0	Pure resistance in $\Omega$ . Multi-phase
<i>Capacitor</i>	CAP_RS	-  - -□  -	BRANCH type 0	Capacitor with damping resistor. C in $\mu\text{F}$ if Copt=0. Multi-phase
<i>Inductor</i>	IND_RP	-~~~ -~~~~-	BRANCH type 0	Inductor with damping resistor. Inductance in mH if Xopt=0. Multi-phase
<i>RLC</i>	RLC	-VVV -	BRANCH type 0	R, L and C in series. Dynamic icon. Multi-phase
<i>R inf</i>	RINF	-W	BRANCH type 0	General resistor to ground to fix floating-subnetwork problems Multi-phase
<i>PQU</i>	PQU	-[PQ]-	BRANCH type 0	Resistance and Inductance or Capacitance calculated internally based on PQ and U=voltage. Multi-phase
<i>Kizilcay F-dep</i>	KFD	-[F(s z)]-	BRANCH IV.I	High order admittance transfer in frequency (s) or time (z) domain
<i>RLC 3-ph</i>	RLC3	-W~~  -~~  -W~~	BRANCH type 0	3-phase R, L and C in series. Independent values in phases. Dynamic icon.
<i>RLC-Y 3-ph</i>	RLCY3	-W~~  -W~~	BRANCH type 0	3-phase R, L and C, Y coupling. Independent values in phases. Dynamic icon.
<i>RLC-D 3-ph</i>	RLCD3	-W~~  -  -W~~	BRANCH type 0	3-phase R, L and C, D coupling. Independent values in phases. Dynamic icon.
<i>C : U(0)</i>	CAP_U0	-U(0)	BRANCH + initial condition	Capacitor with initial condition.
<i>L : I(0)</i>	IND_I0	-i(0)~~-	BRANCH + initial condition	Inductor with initial condition.

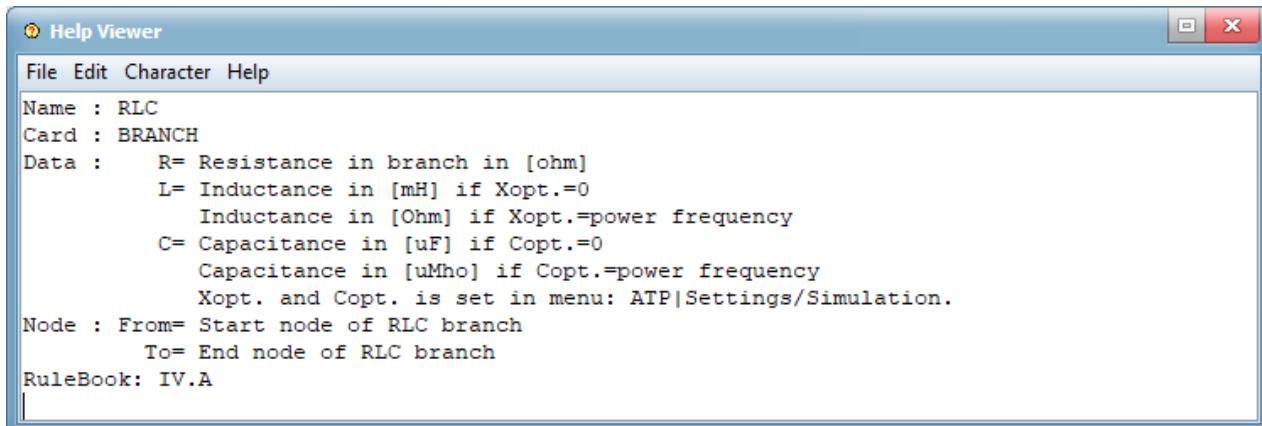


Fig. 4.70 – Help information associated with the series RLC object.

### 4.15.3 Branch Nonlinear

R(i) Type 99
R(i) Type 92
R(t) Type 97
R(t) Type 91
L(i) Type 98
L(i) Type 93
L(i) Type 96
L(i) Hevia 98->96
L(i) Zirka-Moroz
MOV Type 92
R(TACS) Type 91
L(TACS)
C(TACS)
L(i) Type 98, init
L(i) Type 96, init
L(i) Type 93, init

This menu contains the supported nonlinear resistors and inductors. All the objects except the TACS controlled resistor can also have a nonlinear characteristic. These attributes can be specified by selecting the *Characteristic* tab of the *Component* dialog as shown in Fig. 4.56. The nonlinear characteristic of objects can be entered as piecewise linear interpolation. The number of data points allowed to enter on the current/voltage, current/flux or time/resistance characteristics are specified in the *Help* file of objects.

U/I characteristics of nonlinear resistances are assumed symmetrical, thus (0, 0) point should not be entered. If the saturation curve of a nonlinear inductor is symmetrical start with point (0, 0) and skip the negative points. The hysteresis loop of Type-96 reactors is assumed symmetrical, so only the lower loop of the hysteresis must be entered. The last point should be where the upper and lower curves meet in the first quadrant. If you specify a metal oxide arrester with MOV Type-92 component, ATPDraw accepts the current/voltage characteristic and performs an exponential fitting in the log-log domain to produce the required ATP data format.

Fig. 4.71 – Nonlinear branch elements.

Selection	Object name	Icon	ATP card	Description
R(i) Type 99	NLINRES		BRANCH type 99	Current dependent resistance. Multi-phase.
R(i) Type 92	NLRES92		BRANCH type 92	Current dependent resistance. Multi-phase.
R(t) Type 97	NLINR_T		BRANCH type 97	Time dependent resistor. Multi-phase.
R(t) Type 91	NLRES91		BRANCH type 91	Time dependent resistor. Multi-phase.
L(i) Type 98	NLININD		BRANCH type 98	Current dependent inductor. Multi-phase.
L(i) Type 93	NLIND93		BRANCH type 93	True non-linear current dependent inductor. Multi-phase.
L(i) Type 96	NLIND96		BRANCH type 96	Pseudo-nonlinear hysteretic inductor. Multi-phase.
L(i) Hevia 98->96	HEVIA98		BRANCH type 98	Pseudo-nonlinear hysteretic inductor. Multi-phase.
L(i) Zirka-Moroz	DHM96		BRANCH Type96	Magnetic hysteresis model with list of predefined materials
MOV Type 92	MOVN		BRANCH type 92	Current dependent resistance on exponential form. Multi-phase.

<i>R(TACS) Type 91</i>	TACRES		BRANCH type 91	TACS / MODELS controlled time dependent resistor. Multi-phase.
<i>L(TACS)</i>	TACSIND		BRANCH	TACS / MODELS controlled time dependent inductor. Multi-phase.
<i>C(TACS)</i>	TACSCAP		BRANCH	TACS / MODELS controlled time dependent capacitor. Multi-phase.
<i>L(i) Type 98, init</i>	NLIN98_I		BRANCH type 98	Current-dependent inductor. With initial flux.
<i>L(i) Type 96, init</i>	NLIN96_I		BRANCH type 96	Pseudo-nonlinear hysteretic inductor with initial flux.
<i>L(i) Type 93, init</i>	NLIN93_I		BRANCH type 93	True non-linear inductor with initial flux.

All nonlinear inductances have a *Based on Irms/Urms* option and if checked the current/fluxlinkage characteristic is calculated internally (similarly to SATURA routine of ATP). The routines is the same as has been used in transformer for a long time.

#### 4.15.4 Lines/Cables

The *Lines/Cables* menu has several sub-menus for different types of line models. Available line models are: Lumped parameter models (RLC  $\pi$ , RL coupled), distributed parameter lines with constant (i.e. frequency independent) parameters, lines and cables with constant or frequency dependent parameters (Bergeron, PI, JMarti, Noda, Semlyen or ULM), calculated by means of the LINE CONSTANTS, CABLE CONSTANTS or CABLE PARAMETERS supporting routine of ATP-EMTP.

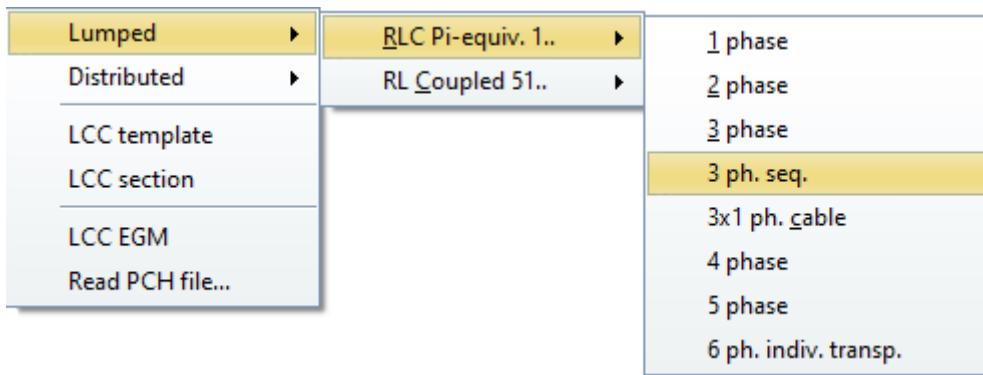


Fig. 4.72 – PI-equivalents with electrical data input.

##### 4.15.4.1 Lumped parameter line models

**RLC Pi-equiv. 1:** These line models are simple, lumped,  $\pi$ -equivalents of ATP Type 1, 2, 3 etc. branches of ATP.

**RL Coupled 51:** These line models are simple, lumped, mutually RL coupled components of Type-51, 52, 53 etc. branches of ATP.

The following selections are available in the two pop-up menus:

Selection	Object name	Icon	ATP card	Description
<i>RLC Pi-equiv. 1.. + 1 phase</i>	LINEPI_1		BRANCH type 1	Single phase RLC $\pi$ -equivalent.
<i>RLC Pi-equiv. 1.. + 2 phase</i>	LINEPI_2		BRANCH type 1-2	2-phase RLC $\pi$ -equivalent Symmetric.
<i>RLC Pi-equiv. 1.. + 3 ph.</i>	LINEPI_3		BRANCH type 1-3	3-phase RLC $\pi$ -equivalent Symmetric matrix input.
<i>RLC Pi-equiv. 1.. + 3 ph. Seq.</i>	LINEPI3S		BRANCH type 1-3	3-phase RLC $\pi$ -equivalent Sequence params. Transposed.
<i>RLC Pi-equiv. 1.. + 3x1 ph. Cable</i>	PI_CAB3S		BRANCH type 1-3	3-phase RLC $\pi$ -equivalent No mutual coupling
<i>RLC Pi-equiv. 1.. + 4 ph.</i>	LINEPI_4		BRANCH Type 1-4	4-phase RLC $\pi$ -equivalent Symmetric matrix input.
<i>RLC Pi-equiv. 1.. + 5 ph.</i>	LINEPI_5		BRANCH Type 1-5	5-phase RLC $\pi$ -equivalent Symmetric matrix input.
<i>RLC Pi-equiv. 1.. + 6 ph. indiv. transp.</i>	LINEPI6S		BRANCH type 1-6	6-phase RLC $\pi$ -equivalent Individually transposed circuits
<i>RL Coupled 51.. + 1 phase</i>	LINERL_1		BRANCH type 51	Single phase RL coupled line model.
<i>RL Coupled 51.. + 2 phase</i>	LINERL_2		BRANCH type 51-52	2-phase RL coupled line model. Symmetric matrix input.
<i>RL Coupled 51.. + 3 phase</i>	LINERL_3		BRANCH type 51-53	3-phase RL coupled line model. Symmetric matrix input.
<i>RL Coupled 51.. + 3 ph. Seq.</i>	LINERL3S		BRANCH type 51-53	3-phase RL coupled line model with sequence impedance (0, +) input. Transposed.
<i>RL Coupled 51.. + 4 phase</i>	LINERL_4		BRANCH type 51-54	4-phase RL coupled line model. Symmetric matrix input.
<i>RL Coupled 51.. + 5 phase</i>	LINERL_5		BRANCH type 51-55	5-phase RL coupled line model. Symmetric matrix input.
<i>RL Coupled 51.. + 6 ph. indiv. transp.</i>	LINERL6S		BRANCH type 51-56	6-phase RL coupled line model with individually transposed circuits.
<i>RL Coupled 51.. + 6 ph. full transp.</i>	LINERL6N		BRANCH type 51-56	6-phase RL coupled line model with full transposition.
<i>RL Coupled 51.. + 6 phase L+Rs</i>	LINERL_6		BRANCH type 51-56	2x3 phase RL coupled line model. Non-symmetric. Off-diagonal R is set to zero.
<i>RL Sym. 51 + 3 ph. seq. 012</i>	LINERL012		BRANCH type 51-53	3-phase RL coupled line model with sequence impedance (0 +-) input. Unsymmetric.
<i>RL Sym. 51 + 3 ph. Full matrix</i>	LINERL3F		BRANCH type 51-53	3-phase RL coupled line model with full matrix input. Unsymmetric.

#### 4.15.4.2 Distributed parameter line models

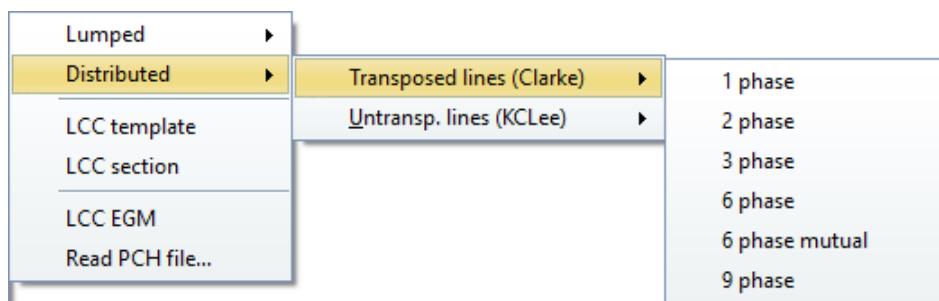


Fig. 4.73 – Distributed transmission line models.

Selecting *Distributed* opens a popup menu where two different types of line models can be selected: *Transposed lines* or *Untransposed lines*. Both types are distributed parameters, frequency independent lines of class Bergeron. Losses are concentrated at the terminals ( $R/4$ ) and of the mid-point ( $R/2$ ). The time step has to be less than half the travel time of the line.

**Transposed lines (Clarke):** These components can be characterized as symmetrical, distributed parameter and lumped resistance models (called as Clarke-type in the ATP Rule-Book). Six different types are supported:

Selection	Object name	Icon	ATP card	Description
<i>Transposed lines + 1 phase</i>	LINEZT_1		BRANCH type -1	Single phase, distributed parameter line, Clarke model.
<i>Transposed lines + 2 phase</i>	LINEZT_2		BRANCH type -1.. -2	2-phase, distributed parameter, transposed line, Clarke model.
<i>Transposed lines + 3 phase</i>	LINEZT_3		BRANCH type -1.. -3	3-phase, distributed parameter, transposed line, Clarke model.
<i>Transposed lines + 6 phase</i>	LINEZT6N		BRANCH type -1.. -6	6-phase, distributed parameter, transposed line, Clarke model.
<i>Transposed lines + 6 phase mutual</i>	LINEZT_6		BRANCH type -1.. -6	2x3 phase, distributed Clarke line. With mutual coupling between the circuits.
<i>Transposed lines + 9 phase</i>	LINEZT_9		BRANCH type -1.. -9	9-phase, distributed parameter, transposed line, Clarke model.

**Untransposed lines (KCLee):** Parameters of these nonsymmetrical lines are usually generated outside ATPDraw. These components can be characterized as untransposed, distributed parameter and lumped resistance models with real or complex modal transformation matrix (called as KCLee-type in the ATP Rule-Book). Double-phase and 3-phase types are supported:

Selection	Object name	Icon	ATP card	Description
<i>Untransposed lines (KCLee)+ 2 phase</i>	LINEZU_2		BRANCH	2-phase, distributed parameters, untransposed (KCLee) line model with complex transf. matrix.
<i>Untransposed lines (KCLee)+ 3 phase</i>	LINEZU_3		BRANCH	3-phase, distributed parameters, untransposed (KCLee) line model.

#### 4.15.4.3 LCC objects

In this part of ATPDraw, you specify the geometrical and material data for an overhead line or a cable and the corresponding electrical data are calculated automatically by the LINE CONSTANTS, CABLE CONSTANTS or CABLE PARAMETERS supporting routine of ATP-EMTP. The LCC module supports line/cable modeling with no limits on the number of phases or conductors.

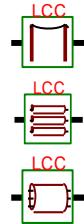
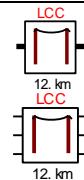
To use the *LCC* module of ATPDraw the user must first select a line/cable component. The number of phases is selected internally in the LCC dialog box. This will display an object (3-phases default) in the circuit window that can be connected to the circuit as any other component. Clicking on this component with the right mouse button will bring up a special input dialog box called *Line/Cable Data* dialog box with two sub-pages: *Model* and *Data*, where the user selects between the supported *System type*:

- Overhead Line: LINE CONSTANTS
- Single Core Cables: CABLE PARAMETERS or CABLE CONSTANTS
- Enclosing Pipe: CABLE PARAMETERS or CABLE CONSTANTS

and *Model type* of the line/cable:

- Bergeron: Constant parameter KCLee or Clark models
- PI: Nominal PI-equivalent (short lines)
- JMarti: Frequency dependent model with constant transformation matrix
- Noda: Frequency dependent model
- Semlyen: Frequency dependent simple fitted model.
- ULM: Frequency dependent phase domain model. Requires *Internal Calculation*. No steady-state initialization in ATP.

The *Line/Cable Data* dialog box completely differs from the *Component* dialog of other components, therefore it is described in chapter 5.3 of the Advanced Manual. An LCC template component can be a stand-alone object written to the ATP-file, or an actual template (checkbox inside) serving as a common data source for *LCC section* objects using it.

Selection	Object name	Icon	ATP card	Description
<i>LCC template</i>	LCC		\$Include or Embed as /BRANCH	Multi-phase LCC object. Overhead line Single core cables Enclosing pipe Matrix Import (Z(f), Y(f) or C) Bergeron/PI/Jmarti/Semlyen/Noda Internal calculation enables ULM.
<i>LCC section</i>	LCC_		\$Include or Embed as /BRANCH	Uses the data of an LCC template with local modification of standard data length, frequency and ground resistivity. Optional single-phase layout.
<i>LCC EGM</i>	LCC_EGM		\$Include or Embed as /BRANCH	Same as LCC_ but with an electro-geometrical model for lightning studies included. Top node to be connected to lightning source.

#### 4.15.4.4 Read PCH file...

ATPDraw can read the .pch output files obtained by external run of ATP-EMTP's Line Constants or Cable Constants supporting routines. Selecting the *Read PCH file...* menu item, the program performs an *Open Punch File* dialog in which the available .pch files are listed. If you select a file and click *Open*, ATPDraw attempts to read the file and if succeed in creates a .lib file and stores it in memory in the Data Base Module format of ATP. When the .lib file is successfully created the icon of the new LCC component appears in the middle of the circuit window. Supports files for 1-12 phases are included as standard.

#### 4.15.5 Switches

Switch time controlled
Switch time 3-ph
Switch voltage contr.
Diode (type 11)
Valve (type 11)
Triac (type 12)
TACS switch (type 13)
Measuring
Statistic switch
Systematic switch
Nonlinear diode

ATPDraw supports most of the switch type elements in ATP, such as ordinary time- or voltage-controlled switches, options for modeling diodes, valves and triacs, as well as measuring and statistical switches.

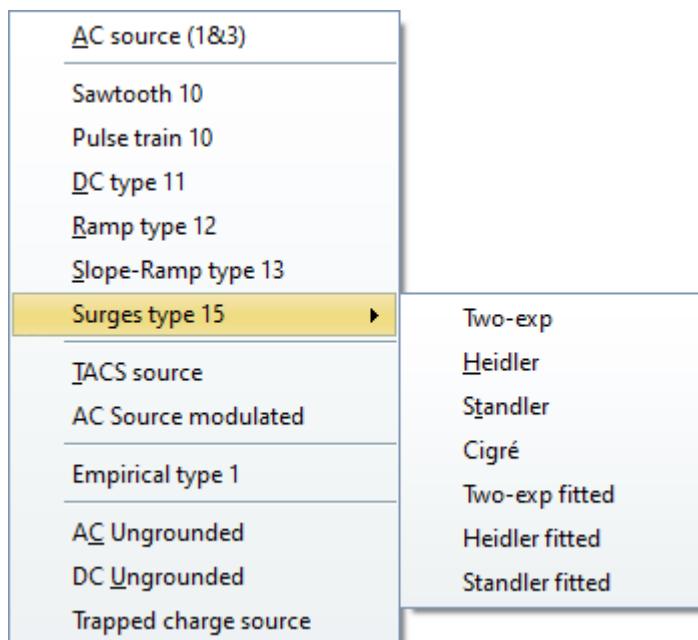
The *Switches* sub-menu contains the following switch objects:

Fig. 4.74 – Supported switch type ATP components.

Selection	Object name	Icon	ATP card	Description
Switch time controlled	TSWITCH		SWITCH type 0	Single or 3-phase time controlled switch. Multiple closing/openings. Dynamic icon; will open, will close...
Switch time 3-ph	SWIT_3XT		SWITCH type 0	Three-phase time controlled switch, Independent operation of phases.
Switch voltage contr.	SWITCHVVC		SWITCH type 0	Voltage controlled switch.
Diode (type 11)	DIODE		SWITCH type 11	Diode. Switch type 11. Uncontrolled.
Valve (type 11)	SW_VALVE		SWITCH type 11	Valve/Thyristor. Switch type 11. TACS/MODELS- controlled. GIFU.
Triac (type 12)	TRIAC		SWITCH type 12	Double TACS/MODELS controlled switch.
TACS switch (type 13)	SW_TACS		SWITCH type 13	Simple TACS/MODELS controlled switch. GIFU.
Measuring	SWMEAS		SWITCH type 0	Measuring switch. Current measurements.
Statistic switch	SW_STAT		SWITCH	Statistic switch. See ATP / Settings / Switch/UM.

Systematic switch	SW_SYST		SWITCH	Systematic switch. See ATP / Settings / Switch/UM.
Nonlinear diode	DIODEN		SWITCH BRANCH	Ideal or nonlinear resistance with forward resistance and snubbers.

#### 4.15.6 Sources



The popup menu under *Sources* contains the following items:

Fig. 4.75 – Electrical sources in ATPDraw.

Selection	Object name	Icon	ATP card	Description
AC source (1&3)	ACSOURCE		SOURCE type 14	AC source. Voltage or current. Single or 3-phase. Ungrounded or grounded. Phase voltage and rms scaling.
Sawtooth type 10	SAW10		SOURCE Type 10	FORTRAN source with frequency and amplitude
Pulse train type 10	PULSE10		SOURCE Type 10	FORTRAN source with frequency, duty cycle and amplitude
DC type 11	DC1PH		SOURCE type 11	DC step source. Voltage or current.
Ramp type 12	RAMP		SOURCE type 12	Ramp source. Voltage or current.
Slope-Ramp	SLOPE_RA		SOURCE	Two-slope ramp

<i>type 13</i>			type 13	source. Voltage or current.
<i>Two-exp type 15</i> F-type source with front and half value time in-line fitting or ATPDraw optimization.	SURGE TWOEXP		SOURCE type 15	Double exponential source Type-15. Voltage or current.
<i>Heidler type 15</i> F-type source with front and half value time in-line fitting or ATPDraw optimization.	HEIDLER HEIDLERF		SOURCE type 15	Heidler type source. Voltage or current.
<i>Standler</i> F-type source with front and half value time in-line fitting or ATPDraw optimization.	STANDLER STANDLERF		SOURCE type 15	Standler type source. Voltage or current.
<i>Cigre</i>	CIGRE		SOURCE type 15	Cigre type source. Voltage or current.
<i>TACS source</i>	TACSSOUR		SOURCE type 60	TACS/MODELS controlled source. Voltage or current.
<i>AC Source modul.</i>	ACSRCMOD		SOURCE type 14+17	Same as ACSOURCE, but with additional TACS input as multiplication factor.
<i>Empirical type 1</i>	SOUR_1		SOURCE type 1	Source with user defined time characteristic. Voltage or current.
<i>AC Ungrounded</i>	AC1PHUG		SOURCE type 14+18	Ungrounded AC source. Voltage only.
<i>DC Ungrounded</i>	DC1PHUG		SOURCE type 11+18	Ungrounded DC source. Voltage only.
<i>Trapped charge</i>	TRAPCHRG		SOURCE Type 14	Special quasi DC voltage source disconnected at t=0.

The sources TWOEXP, HEIDLERF and STANDLERF support ATP's in-line fitting of provided front time (T1) and half-value time (T2). Alternatively, ATPDraw can also calculate the parameters using optimization. A PERC parameter enables various definitions of the times:  
 0: T1 defined as time from zero to peak. T2 defined from zero to 50% on tail.  
 10:T1 defined as (T90-T10)/0.8. T2 defined from virtual zero to 50% on tail.

30:T1 defined as  $(T90-T30)/0.6$ . T2 defined from virtual zero to 50% on tail.  
 50:T1 defined as  $(T90-T10)$ . T2 defined from zero to 50% on tail.

It is not always possible to calculate parameters to fit any shape and the help file provides some in-line fitting restrictions. ATPDraw optimization seems more robust and can also find approximate solutions in some cases.

#### 4.15.7 Machines

SM 59/58
IM 56
Induction WI
Synchronous WI
UM1 Synchronous
UM3 Induction
UM4 Induction
UM6 Single phase
UM8 DC

Two categories of electrical machines are available in ATPDraw: *Synchronous Machines* and *Universal Machines*. ATPDraw does not support machines in parallel or back-to-back.

Fig. 4.76 – Supported electric machine alternatives.

The *Synchronous Machine* models in ATPDraw have the following features/limitations:

- With and without TACS control.
- Manufacturers data.
- No saturation.
- No eddy-current or damping coils.
- Single mass.

The *Universal Machine* models in ATPDraw have the following features/limitations:

- Manual and automatic initialization.
- SM, IM and DC type supported.
- Raw coil data (internal parameters). Manufacturers data in Windsyn.
- Saturation is supported in d, q, or both axes.
- Maximum five excitation coils, sum d and q axis.
- Network option for mechanical torque only.
- Single torque source.

The *Component* dialog of *Universal Machines* is significantly different than that of the other objects. A complete description of parameters in this dialog box is given in chapter 5.2.2 of the Advanced Manual. The WI (embedded WIndSyn by Gabor Furst) components support the following machine types; Synchronous machines with salient or round rotor with damping options. Induction machines with wound, single cage, double cage, or deep-bar rotors. The Windsyn component is documented it chapter 5.2.5 in the Advanced Manual.

The popup menu under *Machines* contains the following items:

Selection	Object name	Icon	ATP card	Description
SM 59/59 + No control	SM		MACHINE type 59 or 58	Synchronous machine. Max 5 TACS outputs. 3-phase armature.
IM 56	IM56A		MACHINE Type 56	Induction machine with multiple controls. 3-phase armature.

<i>Induction WI</i>	UMIND		UM-MACHINE Type 3, 4	Universal machine with manufacturers data input.
<i>Synchronous WI</i>	UMSYN		UM-MACHINE Type 1	Universal machine with manufacturers data input.
<i>UM1 Synchronous</i>	UM_1		UM-MACHINE type 1	Synchronous. Set initialization under ATP / Settings/Switch/UM.
<i>UM3 Induction</i>	UM_3		UM-MACHINE type 3	Induction. Set initialization under ATP / Settings/Switch/UM.
<i>UM4 Induction</i>	UM_4		UM-MACHINE type 4	Induction. Set initialization under ATP / Settings/Switch/UM.
<i>UM6 Single phase</i>	UM_6		UM-MACHINE type 6	Single phase. Set initialization under ATP / Settings/Switch/UM.
<i>UM8 DC</i>	UM_8		UM-MACHINE type 8	DC machine. Set initialization under ATP / Settings/Switch/UM.

#### 4.15.8 Transformers

Ideal 1 phase	ATPDraw supports the transformer components; Ideal transformer, saturable transformer, BCTRAN and the Hybrid Transformer. The BCTRAN model is documented in chapters 5.6 and the Hybrid Model in chapter 5.7 of the Advanced Manual.		
Ideal 3 phase			
Saturable 1 phase			
Saturable 3 phase			
BCTRAN			
Hybrid model			

Fig. 4.77 – Transformer models in ATPDraw.

The popup menu under *Transformers* contains the following items:

Selection	Object name	Icon	ATP card	Description
<i>Ideal 1 phase</i>	TRAFO_I		SOURCE type 18	Single-phase ideal transformer.
<i>Ideal 3 phase</i>	TRAFO_I3		SOURCE type 18	3-phase ideal transformer.
<i>Saturable 1 phase</i>	TRAFO_S		BRANCH TRANSFORMER	Single-phase saturable transformer.
<i>Saturable 3 phase</i>	SATTRAF0		BRANCH TRANSFORMER	General saturable transformer. 3-phase. 2 or 3 windings.
<i>BCTRAN</i>	BCTRAN		BRANCH Type 1...9	Direct support of BCTRAN transformer matrix modeling.
<i>Hybrid model</i>	XFMR		BRANCH	Winding resistance, leakage inductance, topologically correct core, capacitance. Test report, design data or typical.

The characteristic of the nonlinear magnetizing branch of the three saturable-type transformers can be given in the *Characteristic* tab of the component dialog box. The saturable transformers

have an input window like the one in Fig. 4.56. In this window the magnetizing branch can be entered in  $I_{RMS}/U_{RMS}$  or  $I_A/FLUX_{Vs}$  coordinates. The *RMS* flag on the *Attributes* page select between the two input formats. If the *Include characteristic* check box is selected on the *Attributes* page, a disk file referenced in the *\$Include* field will be used in the ATP input file. If the nonlinear characteristic is given in  $I_{RMS}/U_{RMS}$ , ATPDraw will calculate the flux/current values automatically and use them in the final ATP input file.

The BCTRAN transformer component provides direct support of BCTRAN transformer matrix modeling. The user is requested to specify input data (open circuit and short circuit factory test data) in BCTRAN supporting routine format, then ATPDraw performs an ATP run to generate a punch-file that is inserted into the final ATP-file describing the circuit. The user can specify where the factory test was performed and where to connect the excitation branch. The excitation branch can be linear or non-linear. In the latter case, the nonlinear inductors must be connected to the winding closest to the iron core as external elements.

The *BCTRAN* dialog and the *Component* dialog of the *Saturable 3-phase* SATTRAFO differ in many ways from the input data window of other objects. A more comprehensive description of the input parameters is given in chapters 5.6 and 5.2.1 of the Advanced Manual, respectively.

The Hybrid Transformer model is based on development made by Dr. Bruce Mork and his group at Michigan Technological University. It offers both advanced and simplified usage. The XFMR dialog box and the implementation is documented in chapter 5.7.2 of the Advanced Manual.

#### 4.15.9 MODELS

Besides the standard components, the user can create his/her own control modules using the MODELS simulation language in ATP [4]. ATPDraw supports only a simplified usage of MODELS. The user writes a model-file and ATPDraw takes care of the INPUT/OUTPUT section of MODELS along with the USE of each model. The following restriction applies:

- Only INPUT, OUTPUT and DATA supported in the USE statement. Not possible with expressions, call of other models or specification of HISTORY or DELAY CELLS under USE

Using this feature requires knowledge about the syntax and general structure of MODELS language. There are two options for creating a model object in ATPDraw:

- Create a script internally (or load a .mod file created externally) and rely on ATPDraw for automatic identification and layout/icon. The icon and node positions can later be fine tuned. This is the preferred option.
- Create a template manually under *Library / Template on disc / New Model sup-file* and a corresponding .mod file.

The Advanced part of this Manual Chapter 5.5 gives detailed information about both procedures and a general overview about the use of MODELS in ATPDraw. In this chapter only the automatic template generation is introduced.

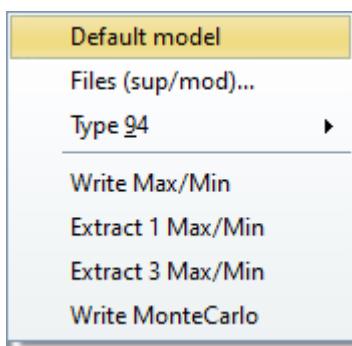
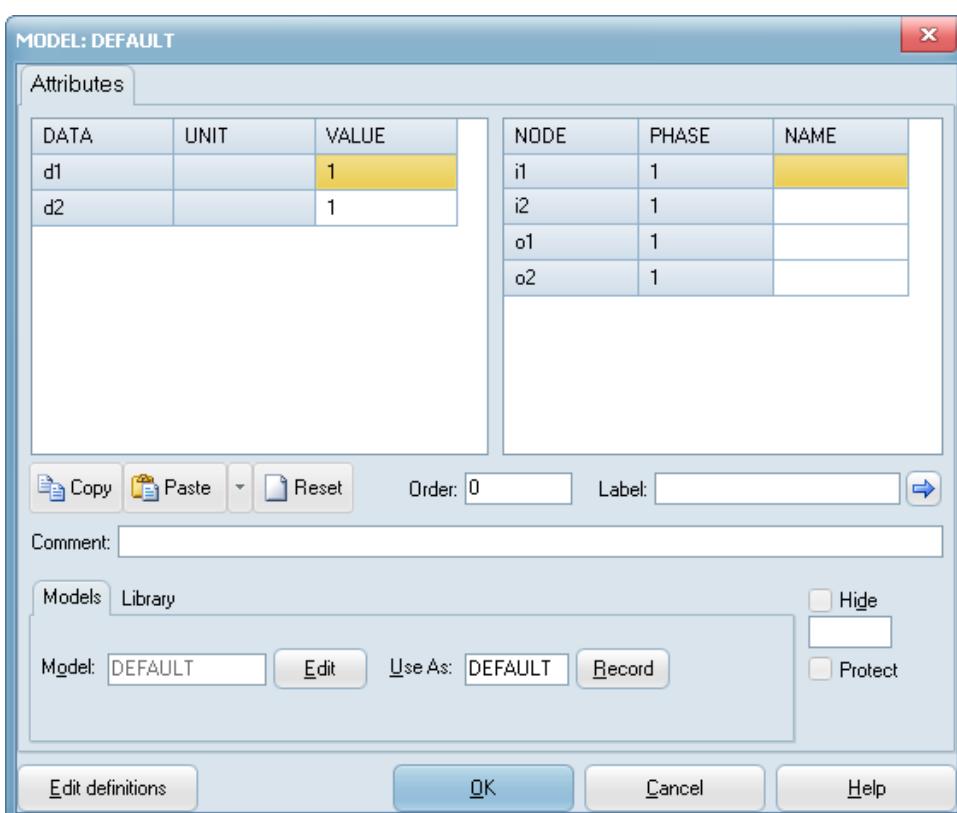


Fig. 4.78 – Options under the MODELS sub-menu.

### **Default model**

This will load a simple, default object and display it in the circuit window. Its input dialog box will look as shown in Fig. 4.79 (2 data, 4 nodes). Click on the *Edit* button to show the *Models Editor* and modify the script directly or to import a script from file or clipboard. Click on *Done* in the main menu of the Models Editor when finished. ATPDraw will then identify the object and create the required template, including icon, see Fig. 4.80. Inputs and outputs are placed to the left and right of the icon, respectively. You can always go back and modify the script, and if you change the number of input or outputs the icon will be recreated. The *Models Editor* support simple debugger, syntax highlighting, code folding, context menu (right click) insert of most language feature, and a simple debugging. ATPDraw will correctly identify array INPUT, OUTPUT and DATA when the first index is unity MYDATA[1..24], MYINPUT[1..3] (for INPUT/OUTPUT there is a upper limit of 26, A..Z phase extension) etc.



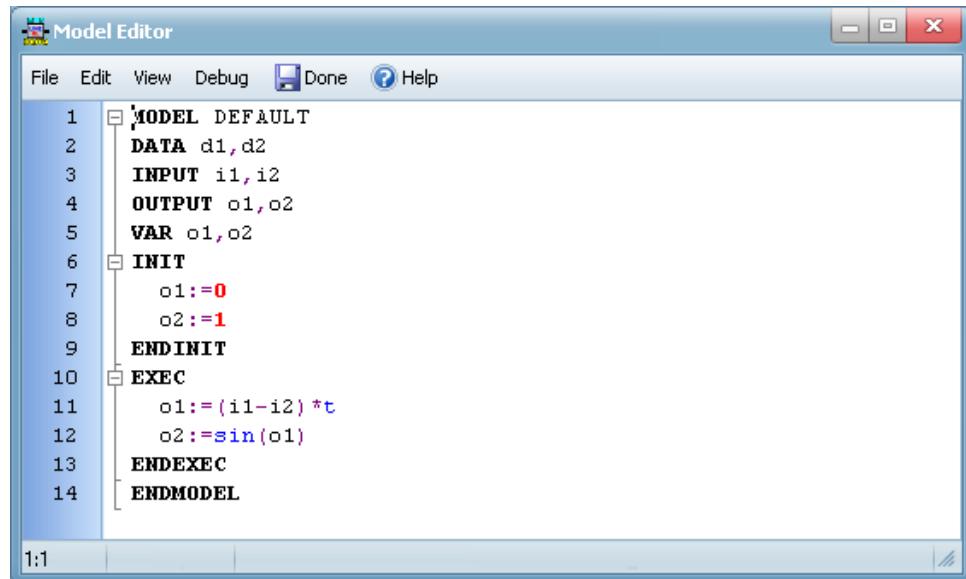


Fig. 4.79 – Model component dialog box. And Text Editor

#### *Files (sup/mod)...*

Selecting MODELS | *Files (sup/mod)...* in the component selection menu performs an *Open Model* dialog box where the user can choose a model file name or a support file name. These files are normally stored under the \MOD folder. If a .mod file was selected ATPDraw interprets the file as shown in Fig. 4.80 and a model component with the corresponding definition and icon appears. If a support file with the same name as the model file exist in the same folder, this file is used instead as basis for the model definitions. In this case the new model object appears immediately in the circuit window, i.e. the *Information* dialog shown in Fig. 4.80 does not show up.

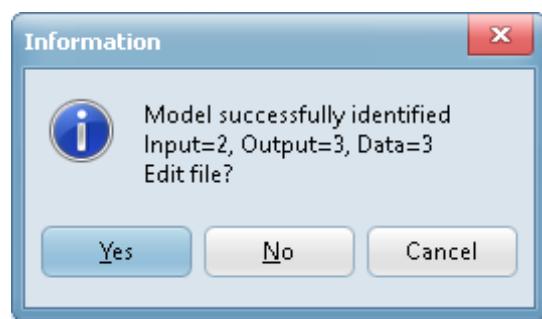


Fig. 4.80 – Interpretation of the model.

The *Component* dialog of model objects has a new input section *Models* below the *DATA* and *NODES* attributes as shown in Fig. 4.81. This new section has two fields: *Model* which is disabled (but automatically follows what is defined in the Model text found using the *Edit* button) and a *Use As* field for specification of the *model\_name* in the USE model AS *model\_name* statement of MODELS. The *Record* button is used for output of internal model variables. On the *Library* page the link to the original support file on disk is given and a *Reload* option is made available. Remember that the original support file on disk not necessarily match the present Model text if the user has changed this.

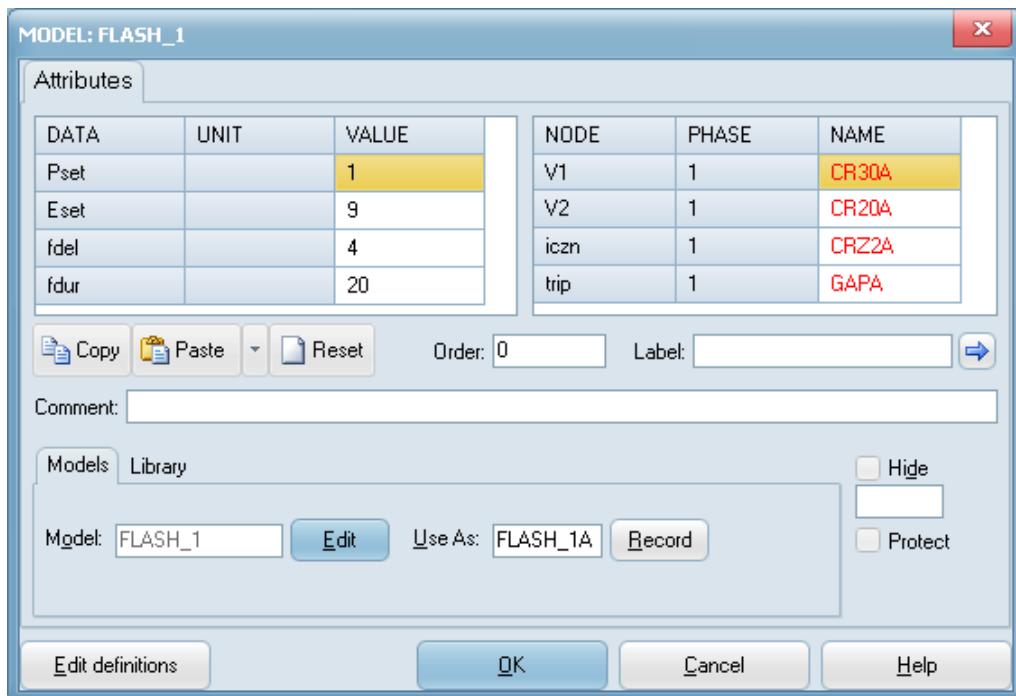


Fig. 4.81 – The component dialog box of model object FLASH\_1 .

The input/output to MODELS, the use of the model and interfacing it with the rest of the circuit are handled by ATPDraw, automatically. Model descriptions are written directly in the ATP file instead of using \$Include. Blank lines are removed when inserting the model file in the ATP-file. The general structure of the MODELS section in the .atp input file is shown below:

```

MODELS
/MODELS
INPUT
  IX0001 {v(CR30A)}
  IX0002 {v(CR20A)}
  IX0003 {v(CRZ2A)}
OUTPUT
  GAPA
MODEL FLASH_1
-----
Description of the model is pasted here
-----
ENDMODEL
USE FLASH_1 AS FLASH_1
INPUT
  V1:= IX0001
  V2:= IX0002
  iczn:= IX0003
DATA
  Pset:=      1.
  Eset:=      9.
  Fdel:=      4.
  Fdur:=     20.
OUTPUT
  GAPA:=trip
ENDUSE
ENDMODELS

```

### Type 94

Selecting MODELS | Type 94/THEV, ITER, NORT, NORT-TR will load a corresponding default model component. You can then open the component which will bring up the Type 94 component dialog box as shown in Fig. 4.82. As for simple models you can then click on the *Edit* button to inspect or modify the type 94 models text. When you click on *Done* in the Text Editor ATPDraw tries to identify the model and then displays a message box similar to Fig. 4.80. Be aware of that the name of the models must be six characters or less. The bottom section of the input dialog has to the right four radio buttons: THEV, ITER, NORT and NORT-TR for specification of the solution method for ATP when interfacing the Type-94 object with the rest of the electrical network. The Data, Node fields and the icon will update dependent on the choice of type. You can also specify the number of phases (#Ph: 1..26) in the component. Branch output and Record of internal variable are also available.

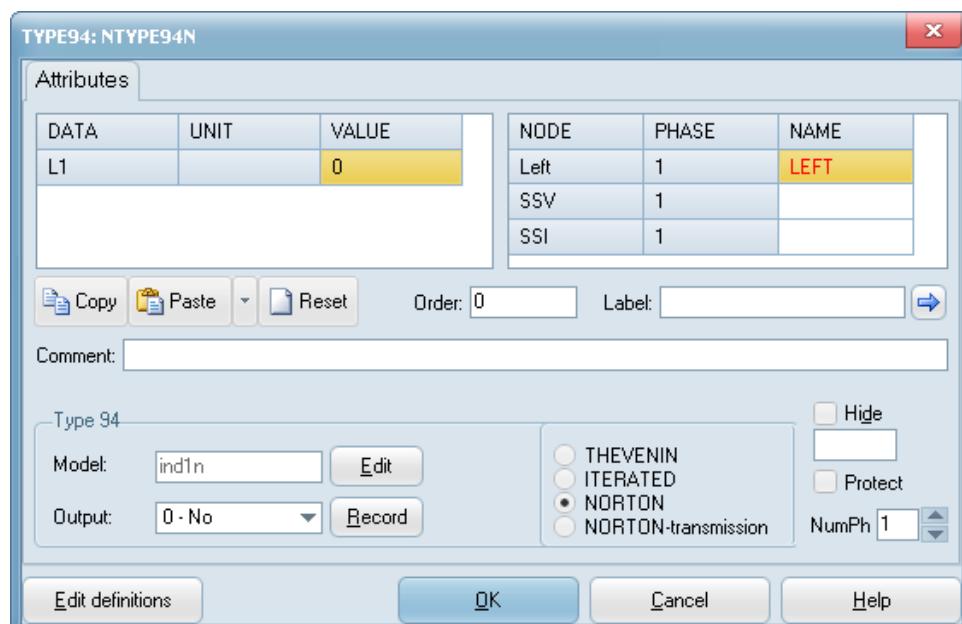


Fig. 4.82 – Component dialog box of Type-94 model objects.

Signal input and data values for a Type-94 object are loaded by ATP and the output of the object are also used automatically by ATP. Interfacing it with other components of the circuit is handled by ATPDraw. A Type-94 compatible .mod files must have a fixed structure and the use of such an object also requires special declarations in the ATP input file as shown next:

Structure of a Type-94 compatible script:

```

MODEL ind1n
comment -----
| Internal circuit: 1-ground : L1      |      |
|                                     1 o      |      |
|                                     - ground   |
| Built for use as a 1-phase non-transmission type-94 Norton component |
----- endcomment
comment -----
| First, declarations required for any type-94 iterated model
| (the values of these data and input are loaded automatically by ATP)
| (the values of these outputs are used automatically by ATP)
| (DO NOT MODIFY THE SEQUENCING OF THE DATA, INPUT, AND VAR IN THIS GROUP)
| (the names may be modified, except 'n')
| (when built for n=1, the array notation is not required)
----- endcomment
DATA  n          -- number of phases
      ng {dflt: n*(n+1)/2} -- number - conductances
INPUT v          -- voltage(t) at terminal 1
  
```

```

v0          -- voltage(t=0) at terminal 1
i0          -- current(t=0) into terminal 1
VAR   i          -- current(t) into terminal 1
        is         -- Norton source(t+timestep) at terminal 1
        g          -- conductance(t+timestep) at terminal 1
        flag       -- set to 1 whenever a conductance value is modified
OUTPUT i, is, g, flag
comment -----
| Next, declarations of user-defined data for this particular model |
| (values which must be defined when using this model as a type-94 component) |
----- endcomment

DATA L1      -- [H] reference value of inductance L
comment -----
| Next, declarations private to the operation of this model |
----- endcomment

VAR   st      -- used for converting Laplace s to time domain
        L       -- [H] variable value of inductance L
INIT
    st := 2/timestep -- trapezoidal rule conversion from Laplace
    L := L1           -- initialize variable inductance value
    g := 1/(st*L)     -- conductance converted from Laplace 1/sL
ENDINIT
EXEC          -- L is constant in this example
    IF t=0 THEN
        flag := 1      -- conductance values have been changed
        i := i0         -- t=0 current through L
        is := -i0 -g*v0 -- -istory term for next step
    ELSE
        flag := 0      -- reset flag
        i := g*v -is   -- applying trapezoidal rule, calculate from v(t)
        is := -i -g*v   --history term from trapezoidal rule, for next step
    ENDIF
ENDEXEC
ENDMODEL

```

The use of a Type-94 Norton model in the ATPDraw generated input file is shown next.

```

C Time varying inductor
94LEFT      IND1N NORT          1
>DATA L1      0.1
>END

```

### Write Max/Min

This is a special cost function or reporting component using Models. The component extracts a value from a simulation by reading from the LIS file. As default the minimum or maximum value of a single input signal is extracted, but the user can add more sophistication to this. Only the signal after a user selectable time *Tlimit* is identified. The component supports multiple run via the Sidebar or ATP/Settings/Variables and contains a View module for displaying the result. A data parameter *AsFuncOf* can be used to pass a loop variable from the Variables (if a number is specified here, the simulation number is used instead). The component is used extensively in circuit optimization and can extract results of systematic parameter variations, see Chapt. 5.9.

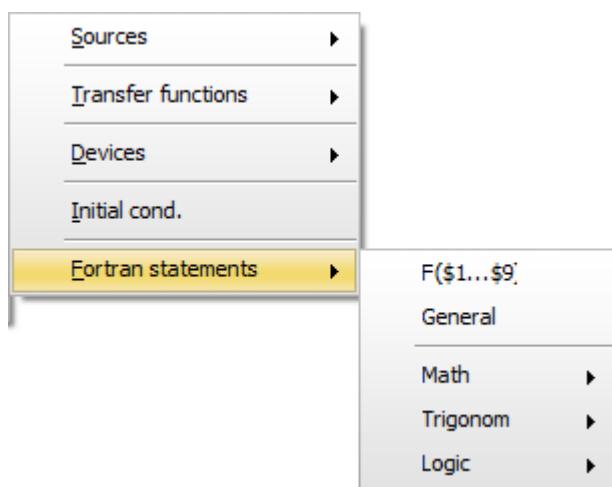
### Extract1, Extract3

These components extend the WriteMaxMin component and plots extremal values on screen also. The comes with setting of Tstart and Tstop as well as scaling.

### Write MonteCarlo

Uses the same approach as WRITEMAXMIN and extracts data from the LIS-file. Accumulates extremal values from Monte Carlo studies in statistical probability density function with user specified resolution.

#### 4.15.10 TACS



The TACS menu gives access to most type of TACS components of ATP. The TACS submenu on the component selection menu contains the following items:

Fig. 4.83 - Supported– TACS objects.

##### 4.15.10.1 TACS sources

The Sources of TACS menu contains the following items:

Selection	Object name	Icon	ATP card	Description
<i>Circuit variable</i>	EMTP_OUT	(T)	TACS type 90-93	Value from the electrical circuit into TACS. 90 - Node voltage 91 - Switch current 92 - internal-variable special EMTP comp. 93 - Switch status Manages also data from universal machines
<i>Models variable</i>	TMODVAR	(M)	TACS type 27	Models output can be connected to TACS via this component
<i>Constant</i>	TCONST	(120_c)	TACS type 98	Displays a TACS constant on screen
<i>DC – 11</i>	DC_01	(T)	TACS type 11	TACS step signal source.
<i>AC – 14</i>	AC_02	(T)	TACS type 14	TACS AC cosine signal source.
<i>Pulse – 23</i>	PULSE_03	(T)	TACS type 23	TACS pulse train signal.
<i>Ramp – 24</i>	RAMP_04	(T)	TACS type 24	TACS saw-tooth train signal.
<i>Ramp step</i>	RAMPSTEP	(T)	TACS type 24 +11 +98	TACS ramp to constant value. Connect with AC source modulated to get easy ramp up AC source.
<i>PMW 3-phase</i>	TPMW6	(PWM 3-p)	TACS type 23 +14 +98	Pulse width modulated TACS source, 3-phase.

The *Circuit variable* object (T) provides an interface for TACS HYBRID simulations. This object must be connected with an electrical node to pass node voltages, or the branch currents / switch status to TACS. The type of the variable sent to TACS is controlled by the *Type* settings in the EMTP\_OUT component dialog box. Users are warned that only single-phase electrical variables

can be interfaced with TACS input nodes, this way. In case of 3-phase modeling, a splitter object is also required, and the coupling to circuit object must be connected at the single-phase side of the splitter as shown in Fig. 4.84.

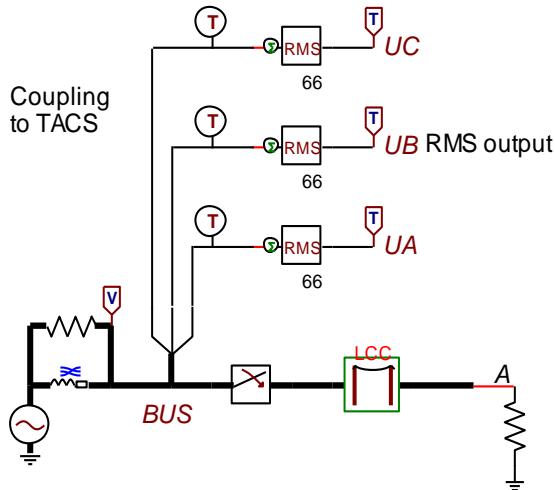


Fig. 4.84 - Coupling a 3-phase electrical node to TACS.

#### 4.15.10.2 TACS transfer functions

All the older TACS transfer functions of previous ATPDraw versions are supported in version 3, but some of them has been removed from the component selection menu and replaced by a more general component: the *General transfer function*. This object defines a transfer function in the  $s$  domain and it can be specified with or without limits. The *Order 1* component offers order 0/1 transfer function with a dynamic icon containing values and optional limits. Four more simple transfer functions are also supported: *Integral*, *Derivative*, first order *High* and *Low pass* filters.

Selection	Object name	Icon	ATP card	Description
<i>General</i>	TRANSF		TACS	General transfer function in $s$ domain. Order 0-7. Named dynamic limits.
<i>Order 1</i>	TRANSF1		TACS	Order 0/1 with optional limits. Dynamic icon with transfer function.
<i>Constant</i>	TRANSFK		TACS	General transfer function in $s$ domain. Order 0. Named dynamic limits.
<i>Integral</i>	INTEGRAL		TACS	Integral of the input multiplied by K.
<i>Derivative</i>	DERIV		TACS	Simple derivative transfer function.
<i>Low pass</i>	LO_PASS		TACS	First order low pass filter.
<i>High pass</i>	HI_PASS		TACS	First order high pass filter.

#### 4.15.10.3 TACS devices

The following TACS Devices are supported in ATPDraw:

Selection	Object name	Icon	ATP card	Description
<i>Freq sensor - 50</i>	DEVICE50		TACS type 88,98 or 99	Frequency sensor.

<i>Relay switch - 51</i>	DEVICE51		TACS type 88,98 or 99	Relay-operated switch.
<i>Level switch - 52</i>	DEVICE52		TACS type 88,98 or 99	Level-triggered switch.
<i>Trans delay - 53</i>	DEVICE53		TACS type 88,98 or 99	Transport delay.
<i>Pulse delay - 54</i>	DEVICE54		TACS type 88,98 or 99	Pulse delay.
<i>Digitizer - 55</i>	DEVICE55		TACS type 88,98 or 99	Digitizer.
<i>User def nonlin - 56</i>	DEVICE56		TACS type 88,98 or 99	Point-by-point non-linearity.
<i>Multi switch - 57</i>	DEVICE57		TACS	Multiple open/close switch.
<i>Cont integ - 58</i>	DEVICE58		TACS type 88,98 or 99	Controlled integrator.
<i>Simple deriv - 59</i>	DEVICE59		TACS type 88,98 or 99	Simple derivative.
<i>Input IF - 60</i>	DEVICE60		TACS type 88,98 or 99	Input-IF component.
<i>Signal select - 61</i>	DEVICE61		TACS type 88,98 or 99	Signal selector.
<i>Sample_track - 62</i>	DEVICE62		TACS type 88,98 or 99	Sample and track.
<i>Inst min/max - 63</i>	DEVICE63		TACS type 88,98 or 99	Instantaneous minimum/maximum.
<i>Min/max track - 64</i>	DEVICE64		TACS type 88,98 or 99	Minimum/maximum tracking.
<i>Acc count - 65</i>	DEVICE65		TACS type 88,98 or 99	Accumulator and counter.
<i>Rms meter - 66</i>	DEVICE66		TACS type 88,98 or 99	RMS value of the sum of input signals.

#### 4.15.10.4 Initial conditions

The initial condition of a TACS variable can be specified by selecting TACS object (type 77) under the *TACS / Initial cond.* menu. The name of this component is `INIT_T` and its icon is

#### 4.15.10.5 Fortran statements

The component dialog box of the *Fortran statements / General* object provides a *Type* field where the user is allowed to specify the type of the object (input, output, inside) and an *OUT* field for the single line Fortran-like expression. These statements are written into the /TACS subsection of the ATP input file starting at column 12.

The *Fortran statements / Math* and *Logic* sub-menus include additional simple objects for the basic mathematical and logical operations.

#### General

Selection	Object name	Icon	ATP card	Description

$F(\$1...$9)$	TFORTRAN		TACS type 98	Preferred FORTRAN expression. Use \$Ii for inputs and \$Dd for data to make object, modular.
<i>General</i>	FORTTRAN1		TACS type 88,98 or 99	User specified FORTRAN expression (old type, not modular; can't copy/paste it).

**Fortran statements / Math**

Selection	Object name	Icon	ATP card	Description
$x - y$	DIFF2		TACS 98	Subtraction of two input signals.
$x + y$	SUM2		TACS 98	Addition of two input signals.
$x * K$	MULTK		TACS 98	Multiplication by a factor of $K$ .
$x * y$	MULT2		TACS 98	Multiplication of $x$ by $y$ .
$x / y$	DIV2		TACS 98	Ratio between two input signals.
$ x $	ABS		TACS 98	Absolute value of the input signal.
$-x$	NEG		TACS 98	Change sign of the input signal.
$\sqrt{x}$	SQRT		TACS 98	Square root of the input signal.
$\exp(x)$	EXP		TACS 98	Exponent of input signal. $e^x$
$\log(x)$	LOG		TACS 98	Natural logarithm of input signal.
$\log_{10}(x)$	LOG10		TACS 98	Logarithm of input signal.
$\text{rad}(x)$	RAD		TACS 98	Converts the input signal from degrees to radians.
$\text{deg}(x)$	DEG		TACS 98	Converts the input signal from radians to degrees.
$\text{rnd}(x)$	RND		TACS 98	Random number generator <x>.

**Fortran statements / Trigonom**

Selection	Object name	Icon	ATP card	Description
$\sin$	SIN		TACS 98	Sinus
$\cos$	COS		TACS 98	Cosinus
$\tan$	TAN		TACS 98	Tangens (sin/cos)
$\cotan$	COTAN		TACS 98	Cotangens (cos/sin)
$\text{asin}$	ASIN		TACS 98	Inverse sinus
$\text{acos}$	ACOS		TACS 98	Inverse cosinus
$\text{atan}$	ATAN		TACS 98	Inverse tangens
$\sinh$	SINH		TACS 98	Sinus hyperbolic
$\cosh$	COSH		TACS 98	Cosinus hyperbolic
$\tanh$	TANH		TACS 98	Tangenss hyperbolic

**Fortran statements / Logic**

Selection	Object name	Icon	ATP card	Description
<i>NOT</i>	NOT		TACS type 98	Logical operator. OUT = NOT IN.
<i>AND</i>	AND		TACS type 98	Logical operator. OUT = IN_1 AND IN_2.
<i>OR</i>	OR		TACS type 98	Logical operator. OUT = IN_1 OR IN_2.
<i>NAND</i>	NAND		TACS type 98	Logical operator. OUT = IN_1 NAND IN_2.
<i>NOR</i>	NOR		TACS type 98	Logical operator. OUT = IN_1 NOR IN_2.
>	GT		TACS type 98	Logical operator. Output = 1 if x > y, 0 otherwise.
>=	GE		TACS type 98	Logical operator. Output = 1 if x >= y, 0 otherwise.
=?	EQ		TACS type 98	Logical operator. Output = 1 if x = y, 0 otherwise.

#### 4.15.11 User Specified



Selecting the *Library* item will draw the predefined user specified object LIB. This object has no input data and cannot be connected with other objects because it has no input or output nodes.

Fig. 4.85 - Supported user specified objects.

##### Library

Using this object will result in a \$Include statement in the ATP-file inserted in the BRANCH part. No parameters are used in this case. The *User specified* section at the bottom contains an *Edit* button that brings up the Text Editor where the user can edit or import an external text. The user can type in the name of the component in the \$Include field. The text will be dumped to a file with this name and extension .lib and location in Result Directory (same as ATP file) when the ATP file is created.

##### Additional

Like the *Library* component but in addition it allows the user to choose under which section in the ATP file to insert the text. The input dialog of this component contains a larger memo field where the user can write in free format text with a row and column indication below. The Additional section at the bottom contains an *Edit* button that brings up a more advanced Text Editor that allows the user to import a text from file or clipboard. This Text Editor also has a right-click context menu with an Insert option of 50 predefined request cards. There is no \$Include field in this component because the text will be inserted directly into the ATP file. Instead the user can select the section; REQUEST, TACS, MODELS, BRANCH, SWITCH, STATISTICAL, SOURCE, INITIAL, OUTPUT, LOAD FLOW, MACHINE type 59/56, UNIVERSAL MACHINE, FREQUENCY COMP. The *Order* number can be used for fine tuning of the location within each section (together with ATP/Settings/Format-Sorting by Order). The three character text in the icon will adapt to the selected section.

**Ref. 1-ph**

Selecting *Ref. 1-ph* will draw the object LIBREF\_1. This object has zero parameters and two nodes. Reference objects are not represented in the ATP input data file, but serve only as visualization of connectivity.

**Ref. 3-ph**

Selecting *Ref. 3-ph* will draw the object LIBREF\_3. This object has zero parameters and two nodes. Reference objects are not represented in the ATP input data file but serve only as visualization of connectivity.

**Files...**

Besides the standard components, the user is allowed to create *User Specified* components. The usage of this feature requires knowledge about ATP's DATA BASE MODULARIZATION technique. The procedure that is described in the Advanced part of this Manual consists of two steps:

1. Creating a new support file (.sup) using the *Library / New object|User Specified* menu.
2. Creating a Data Base Module file (.LIB), which describes the object.

Selecting *Files...* in the component selection menu executes the *Open Component* dialog and the existing support files in the \USP directory are listed. If you select a .sup file from the list and click on the *Open* button, the icon of the object will appear in the middle of the active circuit window. In the dialog box of this component type there is a *User Specified* section with an *Edit* button which will bring up the Text Editor where a .lib file can be imported. A checkbox *Send parameters* is used if the library file is on the Data Base Module format with external parameters. A second checkbox *Internal phase seq.* is used if the phase extension 'A', 'B'... is hard coded inside the Data Base Module and only the five-character root node name should be sent. Henceforth the user specified objects operate similarly than standard objects.

**4.15.12 Steady-state**

RLC PHASOR
PQU PHASOR
CIGRE Load 1 ph
CIGRE Load 3 ph
Linear RLC
HFS Source
Load flow PQ
Load flow UP
Load flow TQ

Harmonic frequency scan and load flow components.

The Harmonic Frequency Scan (HFS) is one of the options under *ATP / Settings / Simulation*. General load flow specification is given under *ATP/Settings/Load flow*.

Fig. 4.86 – Supported HFS components.

Selection	Object name	Icon	ATP card	Description
<i>RLC Phasor</i>	RLC_PHASOR		BRANCH	RLC component only present during steady-state ( $t < 0$ )
<i>PQU Phasor</i>	PQU_PHASOR		BRANCH	RLC component only present during steady-state ( $t < 0$ )

Cigre load 1 ph	CIGRE_1		BRANCH type 0	Single-phase CIGRE load
Cigre load 3 ph	CIGRE_3		BRANCH type 0	3-phase CIGRE load
Linear RLC	RLC_F		BRANCH type 0	Linear RLC for HFS studies
HFS Source	HFS_SOUR		SOURCE type 14	Harmonic frequency source
Load flow PQ	LF_PQ		SOURCE Load flow	Load flow component with active and reactive power restriction
Load flow UP	LF_UP		SOURCE Load flow	Load flow comp. with voltage and active power restriction
Load flow TQ	LF_TQ		SOURCE Load flow	Load flow component with angle and reactive power restriction

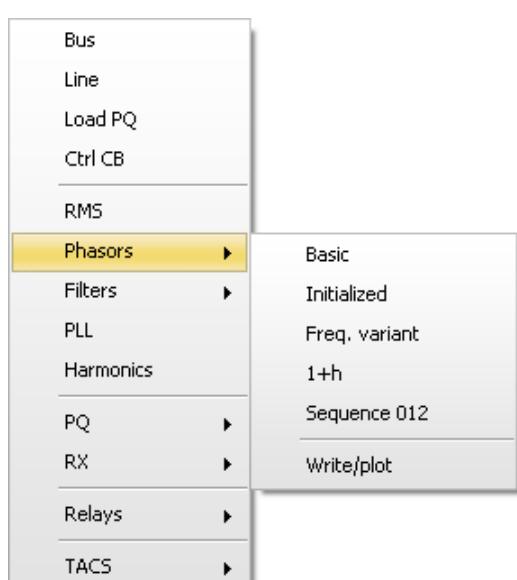
F/n	Amplitude	Angle
1	1	0
5	0.1	0
7	0.15	0
11	0.03	0
13	0.02	0

Selecting HFS under *ATP / Settings / Simulation* will run the ATP data case so many times as specified in the *Harmonic source* component dialog box. The frequency of the harmonic source will for each ATP run be incremented. In the example shown at left, 5 harmonic components are specified in the *F/n* column, and the ATP data case will run 5 times.

Fig. 4.87 - Specification of harmonic source frequencies.

In the first run the source frequency will be 1x50 Hz, the second run 5x50 Hz etc. up to the fifth run  $f = 11 \times 50 \text{ Hz} = 550 \text{ Hz}$ . The *Freq.* value specified by the user under *ATP / Settings / Simulation* is used here as base frequency. The source frequency can also be specified directly in Hz and in such case the first *F/n* must be greater or equal to the Power Frequency. Specifying the frequencies F/n like 50, 250, 350, 450, and 550 would be equivalent to what is shown in Fig. 4.87.

#### 4.15.13 Power System Tools



The Power System Toolbox consists of 3-phase components for power system studies (short circuit and fault analysis and relay protection). Key elements are the LINE3 component that are drawn and edited like Connections and the various phasor calculators based on MODELS.

Fig. 4.88 – Components in the Power System Toolbox

### Fundamental components

These are the building blocks in power system studies with easy fault application. Combine this with the standard voltage and current probes with steady-state output.

Selection	Object name	Icon	ATP card	Description
<i>Bus</i>	BUS3		BRANCH	Multi-node (Bays) connection point with optional load and arrester.
<i>Line</i>	LINE3		BRANCH	LINEPI3S or LINEZT_3 component with faults, CB and CT. Edited like Connections.
<i>Load PQ</i>	LOADPQ		MODELS RLC+SRC	Voltage dependent load model. Group with MODELS controlled source based on phasor calculation
<i>Ctrl CB</i>	CTRLCB		MODELS SWITCH	3-phase TACS switch, opens at MODELS controlled zero crossing

### Basic MODELS calculators

These calculators takes in 3-phase current or voltage chosen in the input node (left) dialog box.

Selection	Object name	Icon	ATP card	Description
<i>RMS</i>	ABC2RMS		MODELS	Outputs RMS value of all 3-phase inputs
<i>Phasors +Basic</i>	ABC2PHR		MODELS	Outputs phasors (re, im) for all 3-phase inputs. DFT recursive or FFT methods with down sampling.
<i>Phasors +Initialized</i>	ABC2PHRI		MODELS	Same as ABC2PHR but with steady-state initialization at t=0
<i>Phasors +Freq. variant</i>	ABC2PHRF		MODELS	Same as ABC2PHR but with frequency as input (from PLL)
<i>Phasors +1+h</i>	ABC2PHRH2		MODELS	Calculate phasors (re, im) in fundamental frequency and one extra harmonic.
<i>Phasors +Sequence 012</i>	ABC2SEQ		MODELS	Calculate phasors and convert it to sequence 012 output (re, im).
<i>Phasors +Write/Plot</i>	WRITEPHASOR		MODELS	Takes in phasors (re, im) to View this at specific instance in time
<i>Transforms+ RMS</i>	ABC2RMS		MODELS	Outputs RMS value of all 3-phase inputs
<i>Transforms +Park-D</i>	ABC2DQ0D		MODELS	Outputs the power invariant Park transform with angle as data

<i>Transforms +Park-I</i>	ABC2DQ0I		MODELS	Outputs the power invariant Park transform with angle as input
<i>Transforms +Clarke</i>	ABC2xyz		MODELS	Outputs the power invariant Clarke transform
<i>Transforms +Cartesian-Polar</i>	CAR2POL		MODELS	Converts quantities in cartesian coordinates to polar
<i>Filters +Low pass</i>	BLPFILT		MODELS	0-3 order Butterworth low pass filter with gain adjustment
<i>Filters +High pass</i>	BHPFILT		MODELS	0-3 order Butterworth high pass filter with gain adjustment
<i>PLL</i>	PLLDQ		MODELS	Calculation of frequency based of park transform of 3-phase inputs
<i>Harmonics</i>	HARMONICS		MODELS	Calculation of 1-26 harmonics based on DFT recursive alg. Old.

### Power and RX calculators

These calculators require input of 3-phase currents and voltages (top-left node is voltage, bottom-left node is current). Normally the two input nodes can simply be connected and further to a unique 3-phase switch node, as shown in Fig. 4.89 . Note that currents can only be obtained from switches and ATP will give you the current in the *first* switch connected to the node you specify. To make sure you get the correct current in the case of several switches connected to the same bus, you should connect to a node not shared by any other switch with a different current. The current probe with *Add current node* adds two measuring switches in series behind the scene and gives you access to the unique middle point.

Selection	Object name	Icon	ATP card	Description
<i>PQ +PQ 3*I</i>	UI2PQ		MODELS	Calculate active and reactive power per phase for 3-phase volt and curr.
<i>PQ +PQ 3-phase</i>	UI2PQ3		MODELS	Calculate active and reactive 3-phase power. DFT+FFT+DQ alg.
<i>PQ +Watt meter</i>	WMETER		MODELS	Calculate average active power by integral method. No downsampl.
<i>RX +(Ua-Ub)/(Ia-Ib)</i>	UI2RXL		MODELS	Calculation of positive sequence impedance (R, X) for phase-to-phase faults.
<i>RX +(Ua-Ub)/(Ia-Ib)</i>	UI2RXE		MODELS	Calculation of positive sequence impedance (R, X) for phase-to-ground faults.
<i>RX +Ua/Ia</i>	UI2RX		MODELS	Calculates the impedance seen in each phase

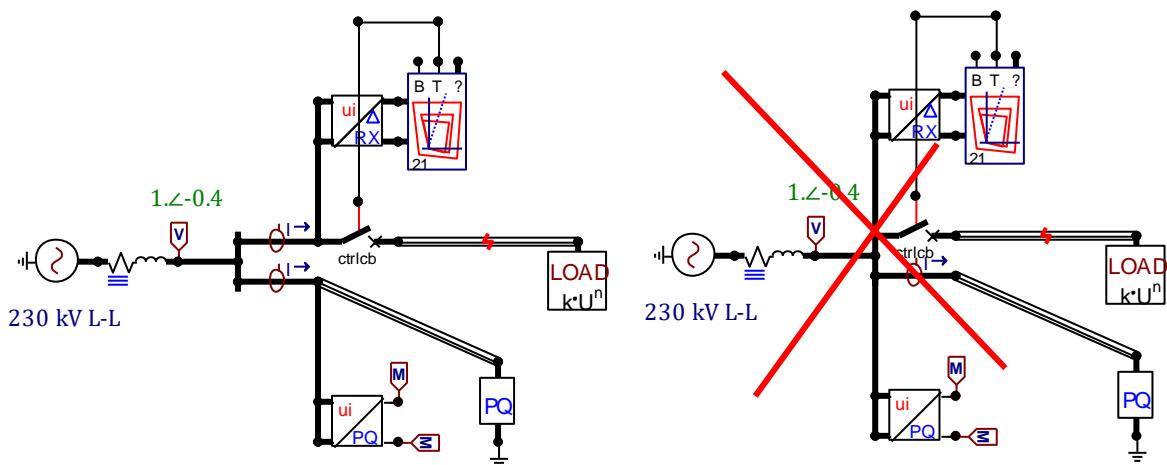


Fig. 4.89 Connection of PQ and RX calculators, correct (left) and incorrect way (right).

### Protective relays

These relays should be connected to RMS (over current, and under/over voltage), Phasor calculators (differential), and RX calculators (distance). The directional over-current relay has its own zero sequence phasor calculator, and the frequency relay requires frequency input. The user must click on the input node of the models to make sure the correct signals are recorded. The relays using signals from other models has default input from models and must be ordered after the model producing the signals. Sorting by order was previously the only option to accomplish this, but ATPDraw v7 offers an option *Edit/Arrange/Sort all Models* to help ordering correctly. All the relays has a blocking control signal input, trip signal and trip zone information outputs. In the dialog window the user sets the zone characteristics and there is also View option that plots the zones and recorded trajectories (with down sampling) until trip. The W1RELAY21P component also has a zone helper.

Selection	Object name	Icon	ATP card	Description
Relays +Overcurrent 50/51	W1RELAY51		MODELS	Based on rms input and a definite time zone gives a trip signal out. Blocking option available. Zone output not relevant.
Relays +Time over-current	W1RELAY51I		MODELS	Based on rms input and standardized inverse time zone gives a trip signal out.
Relays +Dir. Ground 67N	W1RELAY67N		MODELS	Current and voltage input with internal zero sequence phasor calculation. Trips if $ V_0  > V_{lim}$ and $I_0$ in zone relative to $V_0$ .
Relays +Distance quad 21	W1RELAY21P		MODELS	Input from RX calculator. Three delayed zones with 4 points. Zone helper available.
Relays +Distance circ 21	W1RELAY21C		MODELS	Input from RX calculator. Three delayed zones with mho characteristic.

<i>Relays +Diff transf. 87T</i>	W1RELAY87T		MODELS	Input from current phasors 1+h. Calc. differential current with turns ratio, phase shift scaling and harmonic blocking (inrush). Current into unit on both sides.
<i>Relays +Diff line 87L</i>	W1RELAY87L		MODELS	Input from current phasors 1+h. Calc. differential current with time delay compensation and harmonic blocking. Current into unit on both sides.
<i>Relays +Under-voltage 27</i>	W1RELAY27		MODELS	Input from rms voltage. Two undervoltage zones with time delays and 3 different standard characteristics.
<i>Relays +Over-voltage 59</i>	W1RELAY59		MODELS	Input from rms voltage. Two overvoltage zones with time delays and 3 different standard characteristics.
<i>Relays +Frequency 81</i>	W1RELAY81		MODELS	Input from frequency calculator. Two zones for both under and over frequency with fixed time delay.

**TACS**

Selection	Object name	Icon	ATP card	Description
<i>TACS +RMS</i>	TABC2RMS		TACS	3-phase RMS calculator. DEVICE66
<i>TACS +Freq</i>	TABC2FRQ		TACS	3-phase frequency calculator. DEVICE50
<i>TACS +DQ0</i>	TABC2DQ0		TACS	3-phase park transform calculator.
<i>TACS +alpha-beta</i>	TABC2ABG		TACS	3-phase alpha-beta transform calculator.

**4.15.14 All standard Comp...**

In ATPDraw the standard component support files are stored in a single file called `ATPDraw.scl`. The *Standard library* dialog is the container of supported circuit objects in alphabetical order. Any component can be selected from this list, then the object's icon appears in the circuit window the same way as after other selections in the component selection sub-menus.

Support files of the present and even all retired objects (which once were supported in earlier program versions but have been removed from the component selection menu) are included in the standard library. An old circuit file may of course contain such older components, which are also supported internally in ATPDraw and the program will produce correct output.

#### 4.15.15 Add objects

From this menu Texts, Shapes, Pictures, Files and Plot object can be added to the circuit. Shapes are further split into Lines, Arrows, Rectangles and Ellipses and requires two left-clicks for the upper-left and bottom-right corners. Press ESC or click right to cancel the drawing operation.

ATPDraw supports Drag&Drop of project files and attachments from the File Explorer. If a project file is dragged into an existing project window it is imported and if it is dragged into background of the main window it is opened. Any file can be dragged into an existing project and will then become an attachment. Microsoft office and pdf files will get their own symbol and can be opened directly with the default program. ATP, LIS and DAT will be marked as ATP-files and opened in the text editor. PL4-files will be opened in the installed plotter.

#### 4.15.16 Plugins

The Plugins Item points to a user defined disk structure with project files (.acp) and sub-folders. This thus gives an easy access to a user defined library of sub-circuits for import. This is similar to *File/Import* but enables the possibility of direct access. The Plugin directory is defined under *Tools/Options/Files&Folders*.

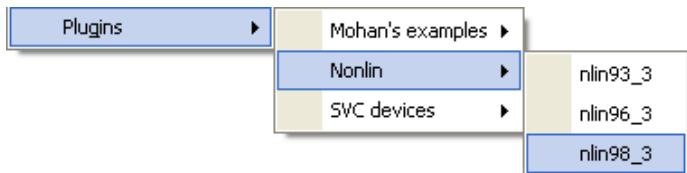


Fig. 4.90 – Example of Plugins menu.

## **5. Advanced Manual . . .**

---

**ATPDraw™  
for Windows  
7.5**



This chapter gives an overview of several more advanced features in ATPDraw: Grouping, special components, usage of the integrated LINE/CABLE CONSTANTS, BCTRAN and the UNIVERSAL MACHINE support, including the Hybrid Transformer model and Windsyn. This chapter also describes how to use MODELS in ATPDraw and how to create new user specified object by means of ATP's \$Include and DATA BASE MODULARIZATION features. You will not be shown how to create the example circuits, but these project files (Exa\_\*.acp) are part of the ATPDraw distribution. To load these example circuits into ATPDraw, use the *File / Open* command (or *Ctrl + O*) and select the file name in the *Open Project* dialog box.

### 5.1 Compress- multilevel modeling

The Compress feature in ATPDraw allows multilevel modeling by replacing a group of objects with a single icon in an almost unlimited numbers of layers. The grouping structure can be imagined as a multi-layer circuit, where the *Edit / Edit Group* brings you one step down in details, while the *Edit / Edit Circuit* menu brings you one step back. This feature increases the readability of the circuit and the feature is especially useful for TACS blocks or frequently reused circuit elements. A group can be copied within the circuit and from other circuits. The grouping feature is demonstrated by re-designing the circuit Exa\_4n.acp in the ATPDraw distribution. This circuit is an induction machine supplied by a pulse width modulated (PWM) voltage source. The induction machine is represented by a Universal Machine type 3 with a typical mechanical load.

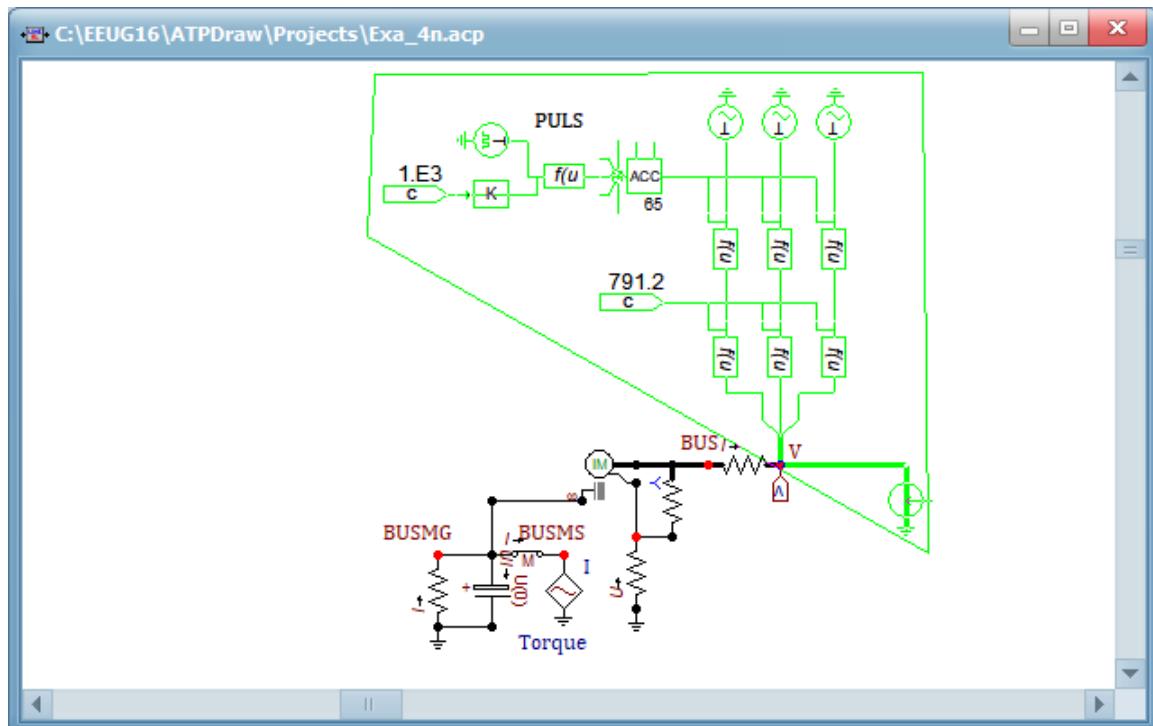


Fig. 5.1 - An induction machine supplied by a pulse width modulated voltage source.

The process of creating a group is as follows:

- Select a group of components. Left click and hold to make a selection rectangle. Double left, left, left, and finally right click to create a polygon (as shown in Fig. 5.1).
- Select *Edit/ Compress* in the main menu.

After selecting a group the *Edit /Compress* command will replace it with a single icon. First the selected sub-circuit is redrawn alone in the middle of the circuit window and the *Compress* dialog

appear as shown in Fig. 5.2. The process is now to graphically select components in the circuit window behind and optionally add their data and nodes to the *Added to groups* list.

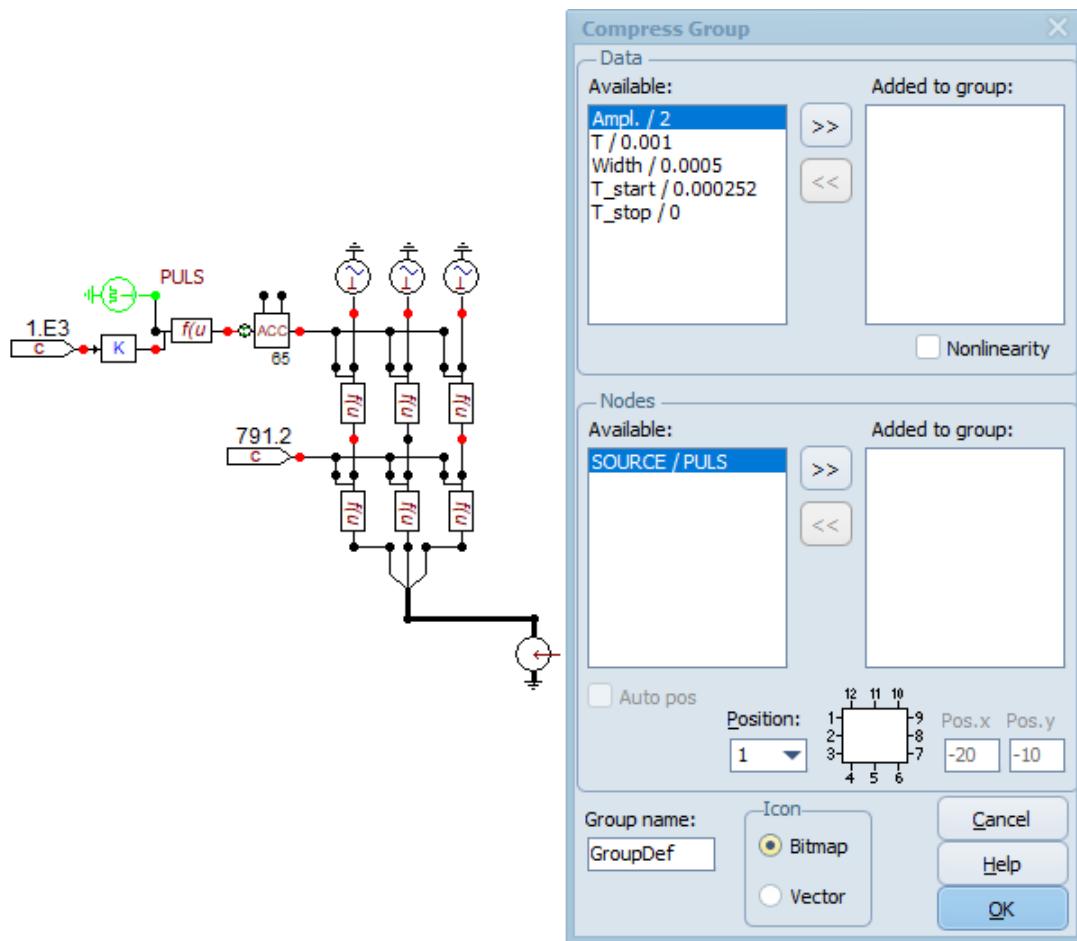


Fig. 5.2 - The *Compress* dialog window.

In the *Compress* dialog the user can specify the external data and nodes of a group of components. The selected data and nodes appear as input in the group object inherited by the content (children).

When you select a component in the circuit window, its data and nodes are listed under *Available*. The component is also drawn in a lime color in the circuit window. The already selected external data/node from this component is drawn with a lime color in the *Added to groups*.

You can then select a parameter under *Available* and click on the **>>** button to transfer it to the *Added to* list. If it is already added, the button is disabled. Selected nodes in the *Available* node list are also drawn in a lime color in the circuit window. Nodes in the *Added to* list are drawn enclosed by a red ring in the circuit window as shown for the 3-phase node of the Splitter chosen to the external in Fig. 5.2. The node position 2 is chosen for this node and this is the middle left standard position.

Vector icon is chosen for this group object. The *Group name* PWM is used in the icon and displayed as an indicator in the *Component dialog* as shown in Fig. 5.5. The *Auto pos* option is available for vector icons only. Later in this example we will change the icon to bitmap style. All data and nodes listed in the *Added to* groups will be the external attribute of the new group object. You can also for each selected node specify its position relative to the object's. The node

positions different from the default 1-12 must be specified by selecting *Position 0* and then give the relative coordinates of the node in the *Pos.x* and *Pos.y* fields. The x-axis is oriented to the right and the y-axis downwards. The *Auto pos* button is only available for Vector graphic icons. Selected data and nodes can also be removed from the *Added to* groups by clicking on the <> button. When later opening the component dialog box for the group object the selected data and node parameters will appear as input possibilities and the values will automatically be transferred to the sub-group.

It is also possible to change the data/node labels by double-clicking on the texts in *Added to* lists. **Important! Two or more data labels with the same name are treated as a single data in the component dialog box.**

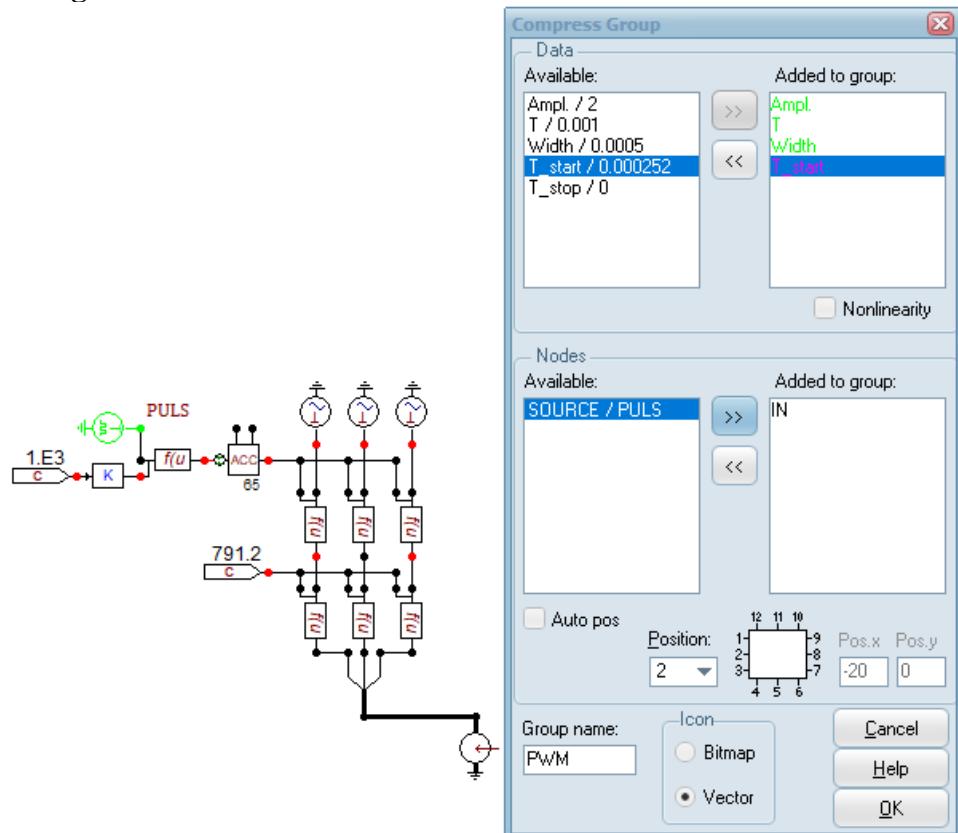
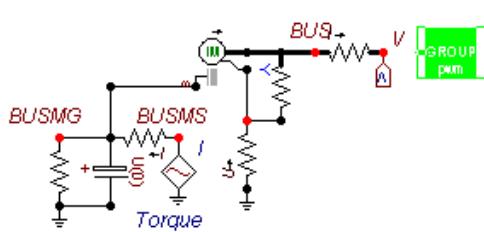


Fig. 5.3 - Name and position of the external nodes of the group.

The Compress process continues in Fig. 5.3 by selection of the external data all belonging to the PULSE\_03 object. Click on *OK* when you have finished. If you need to change the group attributes, you can later select the group and once again choose *Edit/Compress* to reopen the Compress dialog. In such case a *Keep icon* checkbox enables you to preserve the groups icon.



After selecting all the required data and nodes click on *OK*, then a object will automatically be created. The group content disappears, and the new group object is drawn in the circuit window as shown in

Fig. 5.4. The user is then allowed to connect this group object to the rest of the circuit.

Fig. 5.4 - On return from the *Compress* the circuit is redrawn.

Group objects operate like any other objects. You can drag and place the new group in the desired location. The component dialog of the group can be opened by a right or double mouse click and it appears as shown in Fig. 5.5. The data and node values are as specified under Fig. 5.2 and Fig. 5.3.

When changing the data parameter in this window the value will also be transferred to the member components. A change in the node name will be transferred in the same way. In this particular case the Fortran TACS objects are connected to the single-phase side of a splitter. The name of the 3-phase node V will be transferred as real names VC, VB and VA (from left to right) at the Fortran objects' output node. The user must follow this phase sequence in the PWM group object, too.

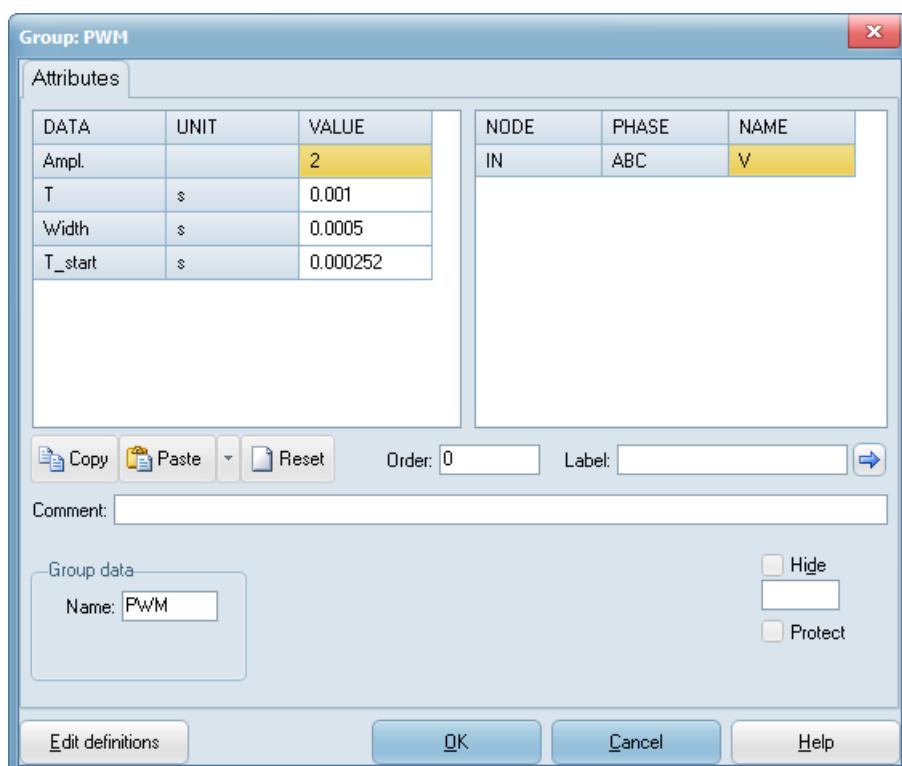


Fig. 5.5 - Opening the new group dialog box.

The *Compress* process for the mechanical load of the induction machine and the component dialog of the new group can be seen in Fig. 5.6 and Fig. 5.7, respectively.

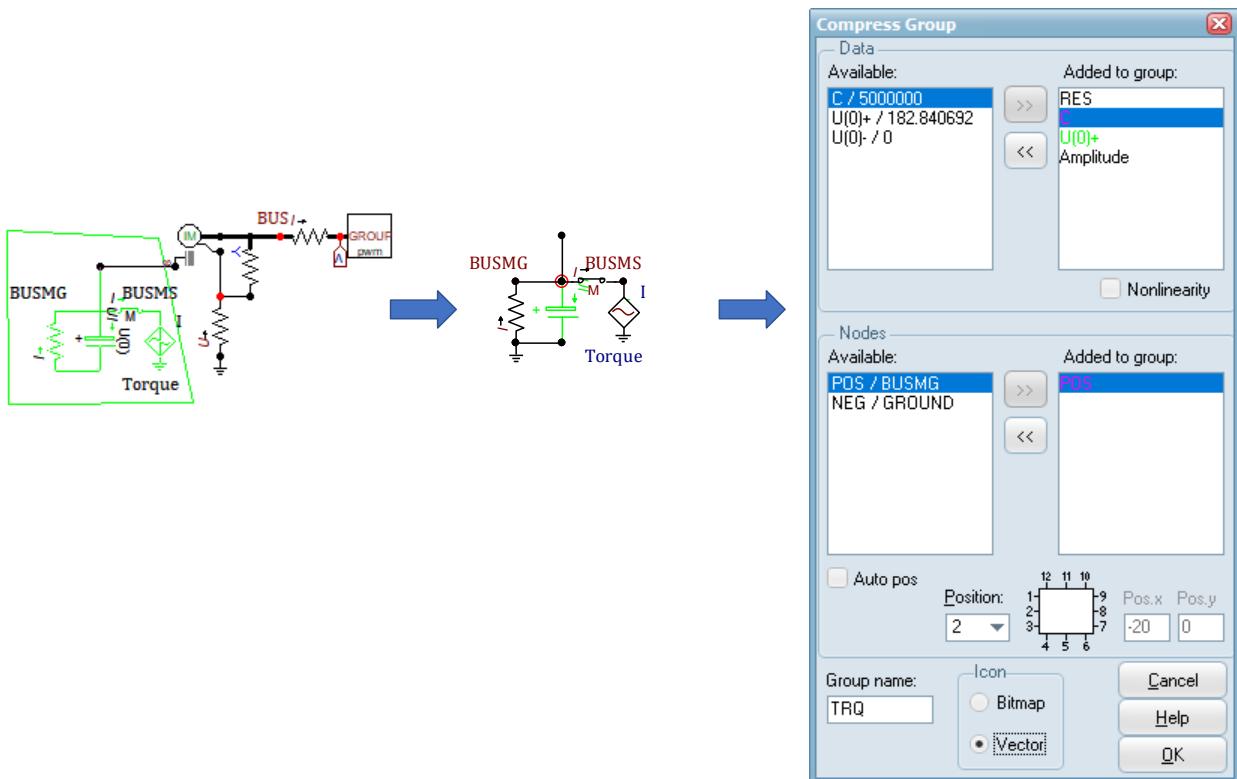


Fig. 5.6 - Selection of data values and external nodes for the mechanical load group.

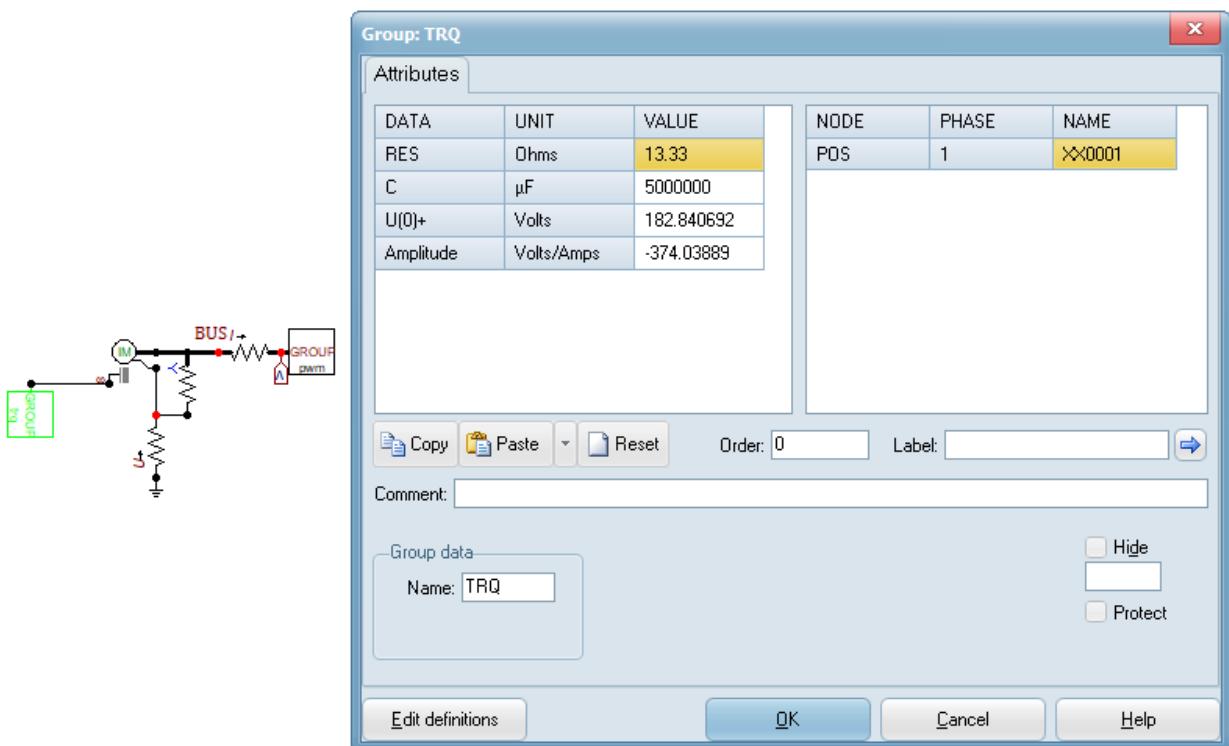


Fig. 5.7- Component dialog box of the mechanical load group-object.

To view/edit a group the user must first select it and then click *Edit* | *Edit Group* in the main menu (or *Ctrl+G*). The group is then extracted on the current circuit window. Actually, the grouping structure can be taken as a multi-layer circuit, where the *Edit Group* brings the user one step down in details, while the *Edit Circuit* brings him one step back. The group is editable in normal way, but the user can't delete components with reference nodes or data in the mother group. I.e.

components having been referenced in one of the *Added to group:* lists cannot be deleted. If the user attempts to do so, a "Marked objects are referenced by compressed group..." warning message reminds him that the operation is not allowed. Selecting the main menu *Edit | Edit Circuit* (or short key *Ctrl+H*) will close the group edit window. It is possible with several levels of groups in the circuit. The maximum number of group levels is 1000.

To customize the icon, click the *Edit definitions* speed button in the lower left corner of the *Component dialog* as shown in Fig. 5.5. The icon editor will appear where the user is free to modify the icon. Fig. 5.8 shows the *Exa\_4g.acp* circuit after grouping the PWM-source and the mechanical load and modifying their icons. Such process is convenient for documentation purposes, because increases the readability of the circuit.

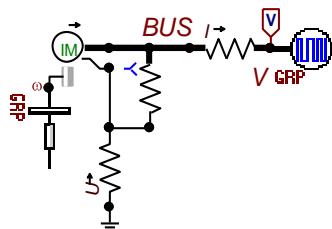


Fig. 5.8 - The icon of the PWM source and the load group has been customized.

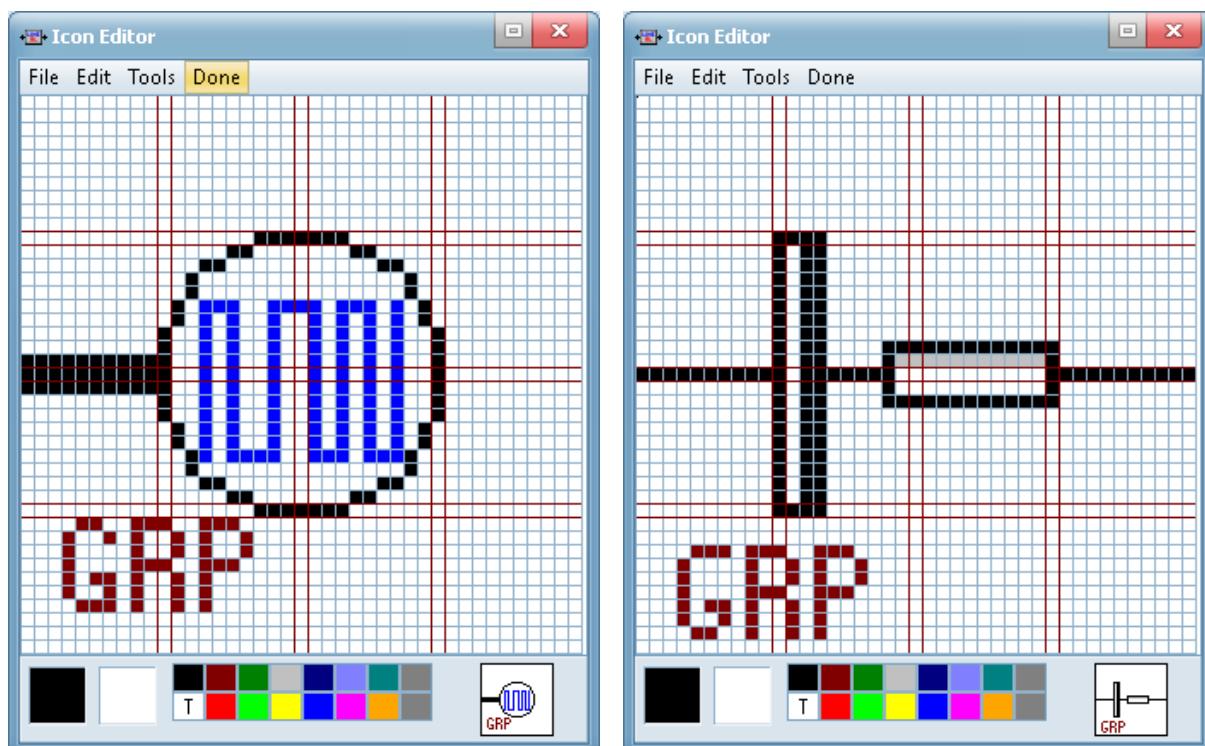


Fig. 5.9 - Customizing the icon of the PWM source and TRQ mechanical torque model. The icon is oriented so that node connections fit with border position 2 (left, middle).

### 5.1.1 Compressing nonlinear objects

A non-linearity can also be external data in a group object. Up to three objects can share the same external nonlinearity. As an example, this section shows how to create a 3-phase, Type-96 hysteretic inductor. You can draw a circuit as shown to the left of Fig. 5.10. To create a group, mark the 3 single-phase inductor and the splitter then select *Edit / Compress*. The data CURR, FLUX and RESID are set as external parameters for all the three inductors. The non-linearity

button under *Added to group* is checked and the *Add nonlinear* button is checked, too for all three inductors.

When you press *OK* the group object is created. The group dialog box shown in Fig. 5.11 contains only one entry for CURR, FLUX, RESID, and FL(0) which are used for all phases, although 3 copies of them are present in the data structure. Only one characteristic is entered in the group's dialog box and inherited by all subscribing inductors. If the 26 data points were insufficient to describe the characteristic as you wish, select the *Include characteristic* option and specify the characteristic in a disk file. The name of that file must be entered in the \$Include field.

The new 3-phase Type-96 group object can be stored as a project file in a special library location and later copied into any circuit using the *File / Import* command, or place in the *Plugins* library.

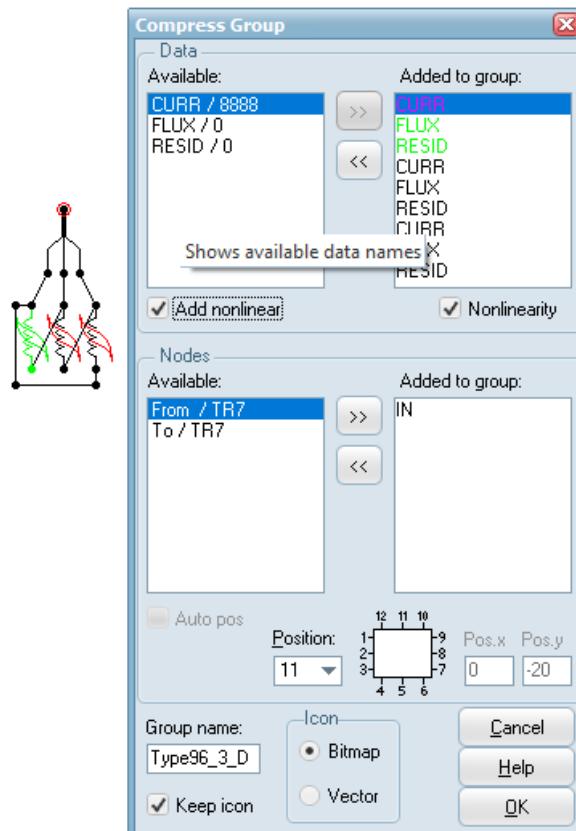


Fig. 5.10 - Creating a 3-phase hysteretic inductor.

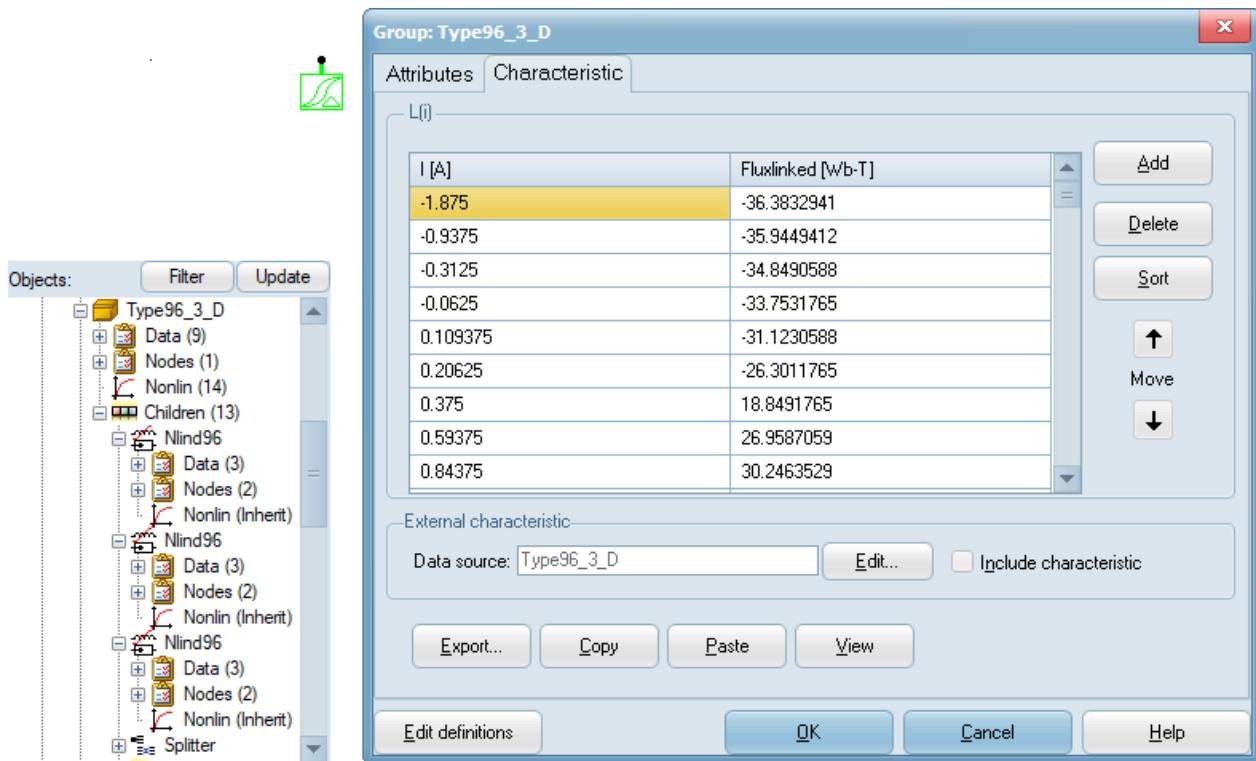


Fig. 5.11 - Nonlinear characteristic of the 3-phase Type-96 group. Sidebar object tree to the left. (notice that only one characteristic is specified, that is used for all phases).

You can customize the group icon as shown in Fig. 5.11 (vector icon illustrated in this case). The hysteresis loop originates from the original inductor icon. This is done by executing the next sequence of operations: click on *Edit definitions* and go into the vector icon editor (leftmost speed button). The default icon is shown as a box with the text 'GROUP' and 'nl96\_3d'. Modify the 'GROUP' text to 'GRP' and move it toward the upper left corner of the box. Modify the text 'nl96\_3d' to 'D' and choose font 'symbol' (you may also increase the font size and pick a different color) and move it towards the lower right corner of the box. Now choose *File/Append std* and choose the standard icon NLIND96. Adjust the left and right node connections. Click on *Done*.

## 5.2 Non-standard component dialog boxes

The component dialog box in which the user can change the object's attributes shows a considerable similarity nearly for all components: on the *Attributes* page the components data and nodes can be specified, on the optional *Characteristic* page you specify the input characteristic of non-linear components.

The following components deviate somewhat from the above description:

- Saturable 3-phase transformer (SATTRAFO)
- Universal machine (UM\_1, UM\_3, UM\_4, UM\_6, UM\_8)
- Statistical / Systematic switch (SW\_STAT, SW\_SYST)
- Harmonic source (HFS\_SOUR)
- Windsyn manufacturers data UM component.

In addition comes Models and User Specified component, explained later.

### 5.2.1 Saturable 3-phase transformer

The component dialog box of this transformer model is shown in Fig. 5.12. This dialog box also has an *Attributes* and a *Characteristic* page, but the former is largely differs from the standard layout. The function of the *Order*, *Label*, *Comment* and *Output* fields are the same as on any other component dialog boxes, the meaning of the other fields are given next. The pair  $I_o$ ,  $F_o$  defines the magnetizing branch inductance at steady state.  $R_m$  is the resistance of the magnetizing branch representing the hysteresis and eddy current losses of the iron core.  $I_o$ ,  $F_o$ ,  $R_m$  may be left blank if the magnetizing branch is neglected in the simulation. Checking the *3-leg core* turns the transformer into a TRANSFORMER THREE PHASE type with high homopolar reluctance that can be specified in the appearing  $R_0$ -field. With the button *3-leg core* unchecked, the model is a saturable transformer with low homopolar reluctance (e.g. a 3-phase transformer with at least one delta winding).

Checking the *RMS* button enables specification of the saturation characteristic in rms values for current and voltage on the *Characteristic* page. A conversion to flux-current values is performed internally in ATPDraw. If the button is unchecked, normal flux-current values should be entered. The tertiary winding can be turned on or off by checking the *3-wind.* button. The nominal voltage of the transformer windings is given in volts. The short circuit inductances may be specified in [mH] if  $X_{opt}$ . parameter is 0 (default) on the *ATP / Settings / Simulation* page. Otherwise, the impedance is given in [ $\Omega$ ] at frequency  $X_{opt}$ .

Windings coupled in wye, delta, auto with all possible phase shifts are supported. In addition zigzag configuration can be selected with arbitrary phase shift from  $<-60,0>+<0,60>$ . In this case the winding is split in two parts internally and the leakage inductance recalculated.

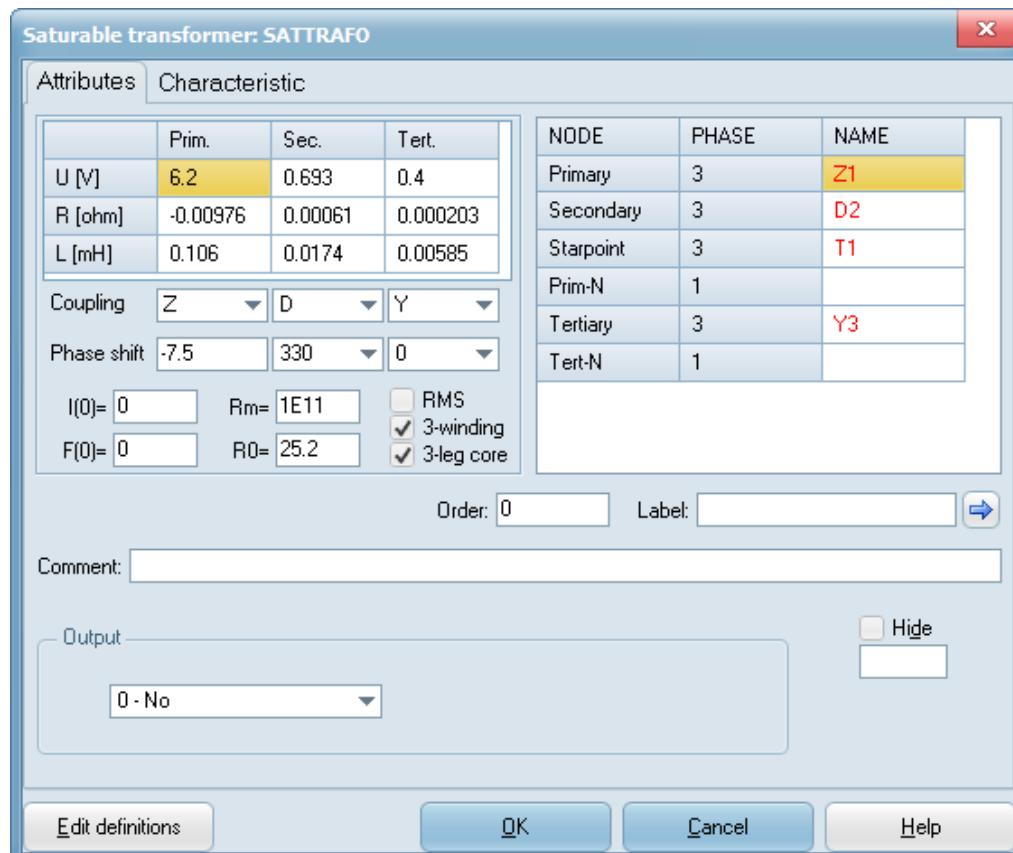


Fig. 5.12 - General saturable transformer dialog.

The *Saturable 3-phase* object is found under *Transformers* in the component selection menu and it can be edited and connected to the main circuit as any other components.

The *Help* button at the lower right corner of the dialog box displays the help file associated with the SATTRAFO object. This help text briefly describes the meaning of input data values:

```

Name : SatTrafo - General saturable transformer. 3 phase. 2 or 3 windings.
       Wye, Delta with all phase shifts. Auto, and Zigzag.
Card : BRANCH
Data :   Io= Current [A] through magnetizing branch (MB) at steady state.
        Fo= Flux [Wb-turn] in MB at steady state.
        The pair Io, Fo defines the inductance in MB at steady state.
        Rm= Resistance in magnetizing branch in [ohm]. 5-leg core or 3-leg shell.
        The magnetizing branch is always connected to the PRIMARY winding and Rm
        is referred to this voltage.
        R0= Reluctance of zero-sequence air-return path for flux. 3-leg core-type
        Vrp= Rated voltage in [V] primary winding (only the voltage ratios matter).
        Rp= Resistance in primary winding in [ohm].
        Lp= Inductance in primary winding in [mH] if Xopt.=0
        Inductance in primary winding in [ohm] if Xopt.=power freq.
        Vrs= Rated voltage in [V] secondary winding.
        Rs= Resistance in secondary winding in [ohm].
        Ls= Inductance in secondary winding in [mH] if Xopt.=0
        Inductance in secondary winding in [ohm] if Xopt.=power freq.
        Vrt= Rated voltage in [V] tertiary winding.
        Rt= Resistance in tertiary winding in [ohm].
        Lt= Inductance in tertiary winding in [mH] if Xopt.=0
        Inductance in tertiary winding in [ohm] if Xopt.=power freq.
        RMS= unchecked: Current/Flux characteristic must be entered.
              checked: Irms/Urms characteristic must be entered.
              ATPDRAW performs a SATURATION calculation.

3-leg core = checked: 3-leg core type transformer assumed. TRANSFORMER THREE PHASE
              unchecke: 5-leg or 3-leg shell type assumed. TRANSFORMER.

3-wind.= turn on tertiary winding.
Output specified the magnetization branch output (power&energy not supported).
Node :   P= Primary side. 3-phase node.
        S= Secondary side. 3-phase node.
        PN= Neutral point primary side.
        SN= Neutral point secondary side.
        T= Tertiary side. 3-phase node.
        TN= Neutral point tertiary side.

Sat= Internal node, connection of the magnetization circuit with saturation.
The coupling is specified for each winding, with four coupling options: Y, D, A, Z
All phase shifts are supported.
Special note on Auto-transformers:
The primary and secondary windings must be of coupling A(uto).
Special note on ZigZag-transformers:
For this type the user can specify a phase shift in the range <-60,0>&<0,60>.
Note that the values -60, 0 and +60 degrees are illegal (as one of the winding parts
degenerates).
The phase shift is given relative to a Y-coupled winding.
If the primary winding is Zigzag-coupled, all other windings will be shifted with it.
If the primary winding is D-coupled, 30 deg. must be added/subtracted to the phase
shifts.
For negative phase shifts the phase A winding starts on leg 1 (called z with voltage
Uz) and continues in the opposite direction on leg 3 (called y with voltage Uy).
For negative phase shifts the phase A starts on leg 1 and continues in the opposite
direction on leg 2.
The normal situation is to specify a phase shift of +/- 30 deg. in which case the two
parts of the winding have the same voltage level and leakage impedance.
In general the ratio between the second part of the winding Uy and the first part Uz
is n=Uy/Uz=sin(a)/sin(60-a) where a is absolute value of the phase shift.
This gives:
Uz=U/(cos(a)+n*cos(60-a)) and Uy=Uz*n
Lz=L/(1+n*n) and Ly=Lz*n*n, Rz=R/(1+n) and Ry=Rz*n
where Lz and Ly are the leakage inductance of each part of the winding (L is the
total leakage inductance) and Rz and Ry are the winding resistance of each winding
part (R is the total).
The parameters Uz, Uy, Zz, and Zy are automatically calculated by ATPDraw based on
the equivalent parameters U and Z and the phase shift, a.

```

Points: It's possible to enter an infinite number of points on the current/flux characteristic.

The required menu is performed immediately after the input menu.

The points should be entered as increasingly larger values.

The point (0,0) is not permitted (added internally in ATP).

RuleBook: IV.E.1-2 or 3.

### 5.2.2 Universal machines

Handling of electrical machines in version 3 of ATPDraw has been updated substantially to provide a user-friendly interface for most of the electrical machine modeling options in ATP. Supported Universal Machine (UM) types are:

- Synchronous machine (UM type 1)
- Induction machines (UM type 3 & 4)
- DC machine (UM type 8)
- Single-phase machine (UM type 6)

The component dialog box of the Universal Machine object is substantially different to the standard dialog box layout, as shown in Fig. 5.13. In the UM component dialog box the user enters the machine data in five pages: *General*, *Magnet*, *Stator*, *Rotor*, *Init*. Several UM models are allowed with global specification of initialization method and interface. These *Global* options can be specified under *ATP / Settings / Switch/UM*.

On the *General* page data like stator coupling and the number of *d* and *q* axis coils are specified. On the *Magnet* page the flux/inductance data with saturation are specified, while on the *Stator* and *Rotor* pages the coil data are given. *Init* page is for the initial condition settings.

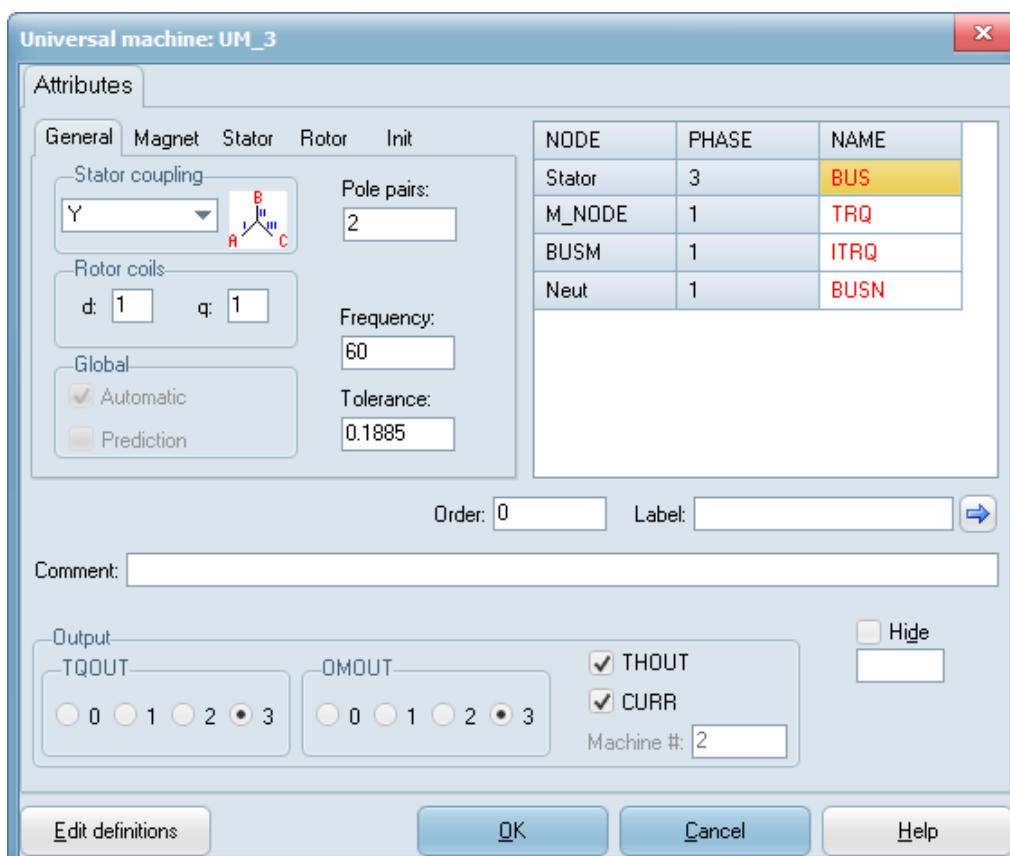


Fig. 5.13 - Universal machine input dialog, auto init. with BUSM node

The dialog boxes for all the universal machines are similar. The type 4 induction machine does not have the *Rotor coils* group, since this is locked to 3. None of the type 3 and 4 induction machines have the field node of course.

The single-phase machine (type 6) and the DC machine (type 8) do not have the *Stator coupling* group. For the type 6 machine the number of *d*-axis is locked to 1. Even if the number of rotor coils or excitation coils can be set to maximum 3, only the first *d*-axis coils will have external terminals for a type 1, 6, and 8 machine. The other coils will be short circuited. Rotor coils are short circuited in case of type 3 machine, while the type 4 machine has an external terminal for all its 3 coils.

Fig. 5.14 shows the various pages for universal machine data input (induction machine, type 3). The buttons under the *Saturation* on the *Magnet*. page turns on/off the various saturation parameters for the *d*- and *q*-axis. This is equivalent to the parameter JSATD and JSATQ in the ATP data format. Selecting *symm* is equal to having JSATD=5 and JSATQ=0 (total saturation option for uniform air gap).

On the *Stator* page, you specify the Park transformed quantities for resistance and inductance for the armature winding. The number of coils on the *Rotor* page and on the *Init* page for manual initialization adapts the specification of the number of rotor coils. First the *d*-axis coils are listed then comes the *q*-axis coils.

The function of the *Order*, *Label*, *Comment* fields are the same as on any other component dialog boxes. The *Help* button at the lower right corner of the dialog box displays the help file associated with the UM objects.

With *UM automatic initialization* checked, ATP will initialize the machine based on slip (induction machines) and stator voltage phasor (synchronous machine). For all machine types a node BUSM will appear and the user must add an AC single-phase, current source with ultra-low frequency to this node. ATP will in the initialization give an amplitude to this equivalent torque source (Current=Torque). Further the must be a connection (switch or low resistance) to the M\_NODE where the mechanical network is connected. For synchronous and DC machines, there will similarly appear a BUSF node where the user must add an ultra-low frequency AC voltage source. ATP will assign the field voltage to this source initially.

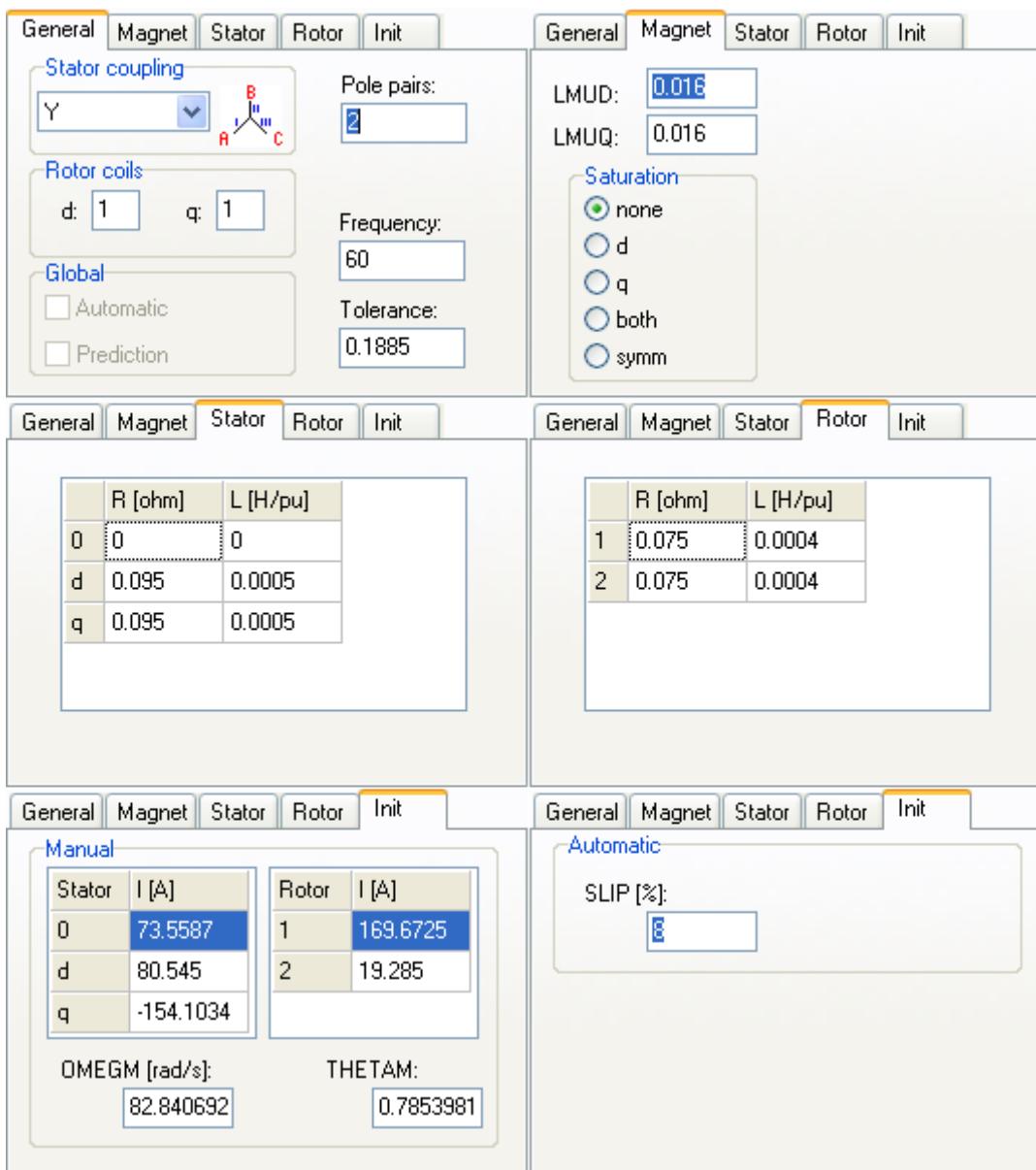


Fig. 5.14 - Data pages of the universal machines dialog box.

The *Help* text briefly describes the meaning of input data values and node names as the example shows next for UM type 1 (Synchronous machine):

**Data:**

*General page:*

Pole pairs - Number of pole pairs

Tolerance - Rotor-speed iteration-convergence margin.

Frequency - Override steady state frequency.

Stator coupling

Select between Y, Dlead (AC, BA, CB) and Dlag (AB, BC, CA)

Selecting Y turns neutral node Neut on.

Rotor coils

Specify the number of d- and q- axis rotor coils. Maximum total number is 3. Only terminals for 1st d-axis coil. The other coils are assumed short circuited.

Global

Visualization of mode of initialization and interface.

Set under the main menu ATP|Settings/Switch/UM for each circuit.

*Stator page:*

Specify resistance and inductance in Park transformed

quantities (d- q- and 0- system). All inductances in H or pu.

*Rotor page:*

The total number of coils are listed and given data on the Rotor

page. First the d-axis coils then the q-axis coils are listed.

Specify resistance and inductance for each coil. All the coils  
except the first is short circuited. All inductances in H or pu.

*Magnet. page:*

LMUD - d-axis magnetization inductance.

LMUQ - q-axis magnetization inductance.

Turn on/off the saturation.

Symm. is equal saturation in both axis, specified only in d.

LMSD - d-axis saturated inductance.

FLXSD - d-axis flux-linkage at the saturation knee point.

FLXRD - d-axis residual flux-linkage (at zero current).

LMSQ - d-axis saturated inductance.

FLXSQ - q-axis flux-linkage at the saturation knee point.

FLXRQ - q-axis residual flux-linkage (at zero current).

NB! All inductances in H or pu.

*Initial page:*

Initial conditions dependent on manual or automatic  
initialization is chosen under ATP|Settings/Switch/UM

*Automatic:*

AMPLUM - initial stator coil (phase) voltage [V].

ANGLUM - angle of phase A stator voltage [deg].

*Manual:*

Specify stator current in the d- q- and 0-system

Specify rotor current inn all coils

OMEGM - initial mechanical speed [mech rad/sec or unit]

THETAM - initial pos of the rotor [elec rad]

*Output:*

TQOUT=1: air gap torque

=2: 1 + d-axis common flux

=3: 2 + d-axis magnetization current

OMOUT=1: rotor shaft speed in [rad/sec]

=2: 1 + q-axis common flux

=3: 2 + q-axis magnetization current

THOUT=checked: rotor position in [mech rad]

CURR =checked: all physical coil currents

**Node:**

Stator - 3-phase armature output terminal.

M\_NODE - air-gap torque node.

FieldA - Pos. terminal of excitation rotor coil.  
(the other coils are grounded)

FieldB - Neg. terminal of excitation rotor coil.

BUSM - torque-source node for automatic initialization.

BUSF - field-source node for automatic initialization.

Neut - Neutral point of Y-coupled stator coils.

The final section of the *Help* file describes the equivalent electrical network of the mechanical network for torque representation:

Shaft mass (moment of inertia)  $\leftrightarrow$  Capacitance (1kg\*m<sup>2</sup>  $\leftrightarrow$  1 Farad)

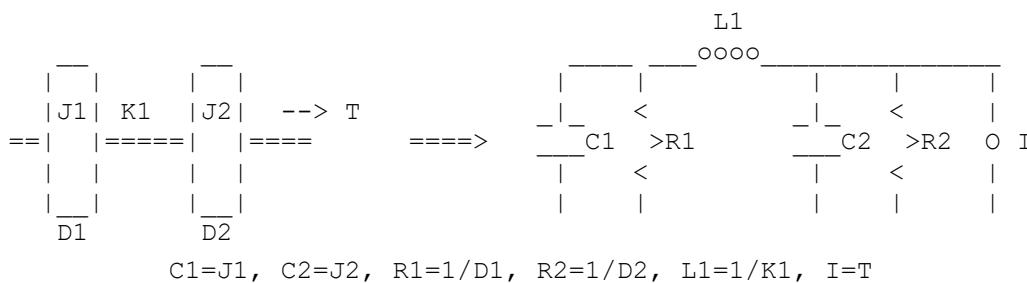
Shaft section (spring constant)  $\leftrightarrow$  Inverse inductance. (1 Nm/rad  $\leftrightarrow$  1/Henry)

Shaft friction (viscous damping)  $\leftrightarrow$  Conductance. (1 Nm/rad/s  $\leftrightarrow$  1/ohm)

Angular speed  $\leftrightarrow$  Voltage (1 rad/s  $\leftrightarrow$  1 Volt)

Torque  $\leftrightarrow$  Current (1 Nm  $\leftrightarrow$  1 Amp)

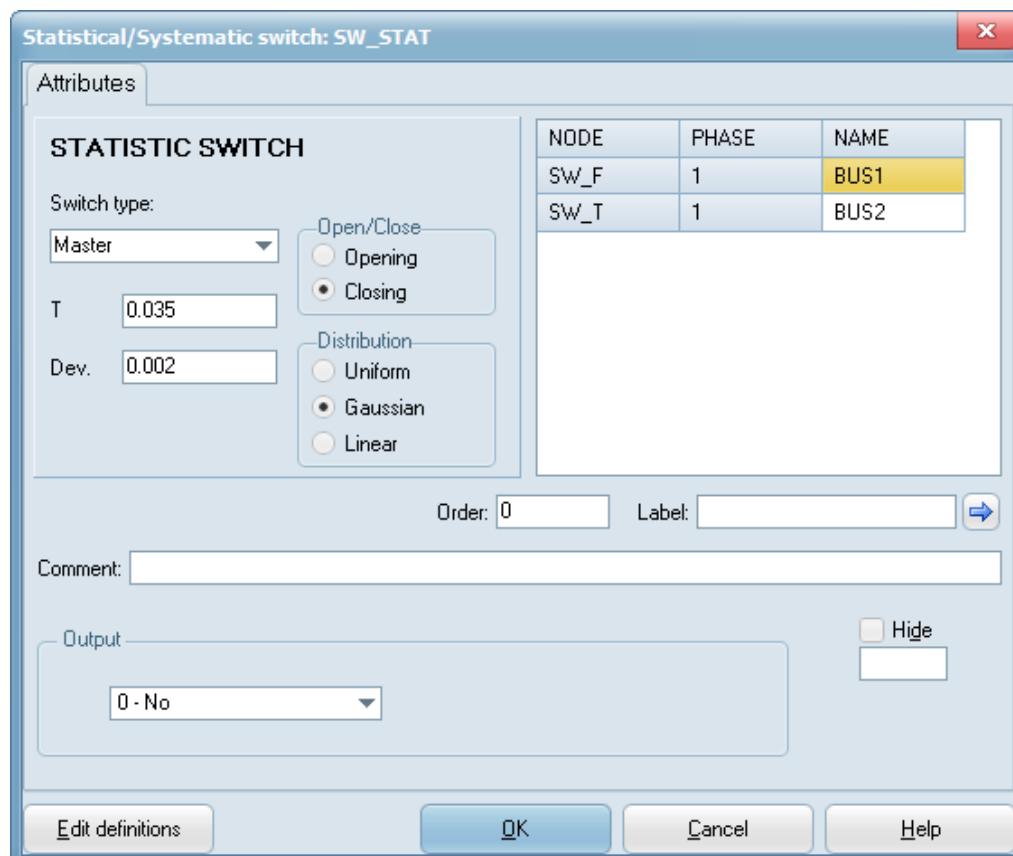
Angle  $\leftrightarrow$  Charge (1 rad  $\leftrightarrow$  1 Coulomb)



### 5.2.3 Statistic/systematic switch

Handling of statistic/systematic switches in version 3 of ATPDraw has been made more general by introducing the independent/master/slave concept. The component dialog boxes of the statistical switches slightly differ however from the standard switch dialog box layout as shown in Fig. 5.15.

The user can select the *Switch type* in a combo box out of the supported options: *Independent*, *Master* or *Slave*. This will also enable the possible input fields and change the number of nodes (note that slave switch has 4 nodes). The *Distribution* for the statistical switch takes into account the specification of the IDIST parameter on the miscellaneous switch card (ATP / Settings / Switch/UM). Selecting IDIST=1 will disable the *Distribution* group and force *Uniform* distribution. The *Open/Close* radio buttons select if the switch closes or opens with *Ie* as current margin for opening switches. The number of ATP simulations is set by the miscellaneous switch parameter *Num.* on the ATP / Settings / Switch/UM page. This value influences the 1<sup>st</sup> misc. data parameter NENERG of ATP. ATPDraw sets the correct sign of NENERG: i.e. > 0 for statistic or < 0 for systematic switch studies. The function of the *Order*, *Label*, *Comment* and *Output* fields are the same as for any other standard components.



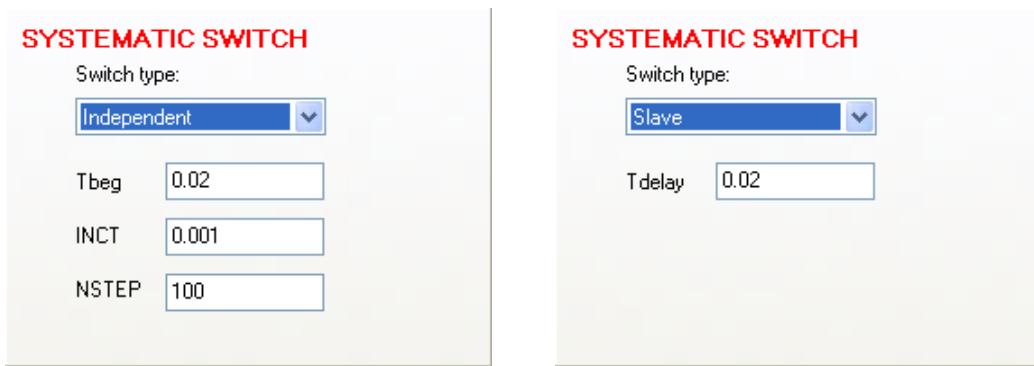


Fig. 5.15 - Dialog box of the statistic switch (top) and data windows of the systematic switch.

The *Help* button at the lower right corner of the dialog box displays the help file associated with the object. This text briefly describes the meaning of input data values and node names as shown below:

**SW\_STAT** - Statistic switch.

Distribution: Select uniform or gaussian distribution.

If IDIST=1 under ATP|Settings/Switch/UM only uniform is possible.

Open/Close: Select if the switch closes or opens.

Current margin available for opening switch.

T = Average switch opening or closing time in [sec.]

For Slave switches this is the average delay.

Dev.= Standard deviation in [sec.] .

For Slave switches this is the deviation of the delay.

Ie = Switch opens at a time T>Tmean and the current through the switch is less than Ie.

Switch type:

INDEPENDENT: Two nodes

MASTER : Two nodes. 'TARGET' punched. Only one is allowed.

SLAVE : Four nodes. Specify node names of MASTER switch.

The icon and nodes of the objects adapt the switch type setting.

Node: SW\_F= Start node of switch.

SW\_T= End node of switch.

REF\_F= Start node of the MASTER switch

REF\_T= End node of the MASTER switch

**SW\_SYST** - Systematic switch.

Tbeg = When ITEST=1 (ATP|Settings/Switch/UM)

Tmid = When ITEST=0 (ATP|Settings/Switch/UM)

Tdelay= For SLAVE switches. If ITEST=0 : T=Tmid.

INCT = Size of time increment in [sec.] .

NSTEP = Number of time increments.

Switch type:

INDEPENDENT: Two nodes

MASTER : Two nodes. 'TARGET' punched.

SLAVE : Four nodes. Specify node names of MASTER switch.

The icon and nodes of the objects adapt the switch type setting.

Node : SW\_F = Start node of switch.

SW\_T = End node of switch.

REF\_F = Start node of the MASTER switch

REF\_T = End node of the MASTER switch

## 5.2.4 Harmonic source

The component dialog box of the *Harmonic source* that is used in HFS studies deviates somewhat from the standard source dialog box layout shown in Fig. 4.87.

F/n	Amplitude	Angle
1	1	0
5	0.1	0
7	0.15	0
11	0.03	0
13	0.02	0

Selecting HFS under *ATP / Settings / Simulation* the ATP will run the case so many times as specified in the *Harmonic source* component dialog box. The frequency of the harmonic source will for each ATP run be incremented. The user selects the source type by the *Voltage* or *Current* radio button. In the example shown here, the data case will run 5 times because the *F/n* column has 5 harmonics entered.

Fig. 5.16 - Harmonic source dialog box.

The base frequency here is the *Freq.* value specified under *ATP / Settings / Simulation*. The amplitude and angle of the *F/n' th* harmonic source is given in columns *Ampl.* and *Angl.*

### 5.2.5 Windsyn components

Windsyn was a program by the late Gabor Furst in Vancouver-Canada. It took manufacturers machine data as input, made a fitting and produced an electrical universal machine model with startup. The source code of this program is now rewritten and directly embedded in ATPDraw and in addition internal exciter and governor controls are added. This facilitates the usage of electrical machines in ATP/ATPDraw considerably. Seven electrical different machine types are supported; Induction machines (wound, single cage, double cage, and deep-bar rotors), Synchronous machines (salient rotor; d-damping, dq-damping, round rotor; dq-damping).

Fig. 5.17 shows the Windsyn induction machine input dialog in ATPDraw. It follows the same design as most components. The input data consists of the standard Data grid to the left and a page control at the bottom. On the *Model* page the user can select the type of machine, add inertia in three different formats with a damping factor, add an optional governor and perform fitting of the manufacturers data to electrical quantities. The current machine number is presented to the user, but this number could change as the circuit develops. As default an induction machine with wound rotor is assumed. If the user changes the machine type under *Rotor* the Data grid is automatically updated. On the *Start-up* page the user can set the start-up options for the machine dependent on the machine type and initialization INITUM set under *ATP/Settings/Switch/UM* and add an optional extra load. Under *Output* the user can select various outputs calculated via TACS.



Fig. 5.17 - Windsyn dialog box in ATPDraw.

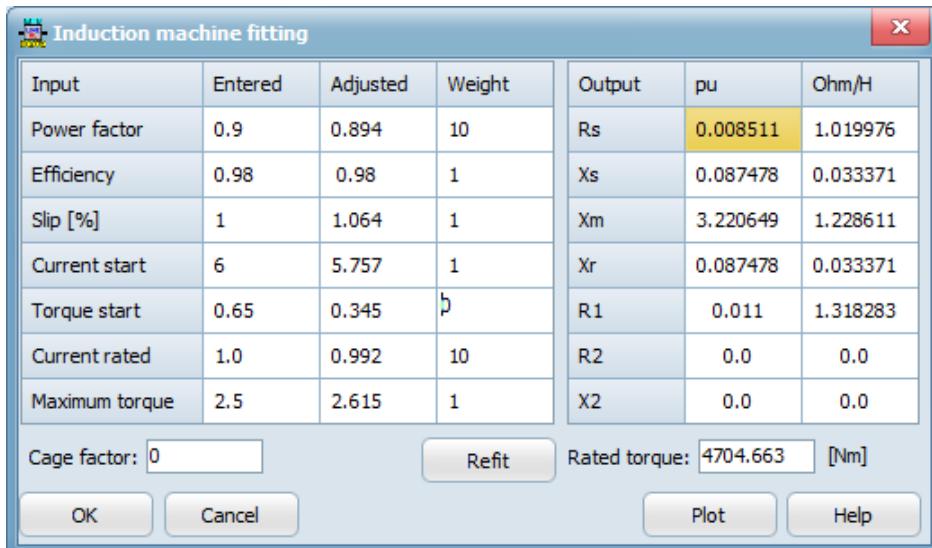


Fig. 5.18 - Windsyn induction machine fitting.

For induction machines parameters the electrical parameters can be fitted to the manufacturer data (power factor, efficiency, slip, starting and rated current, starting and maximum torque) with user tunable fitting factors as shown in Fig. 5.18. The torque-speed characteristic is shown via *Plot*.

Two different and simplified Exciter (Voltage or Reactive power) and Governor (Speed or Active power) controls via TACS are also embedded as shown in Fig. 5.19 .

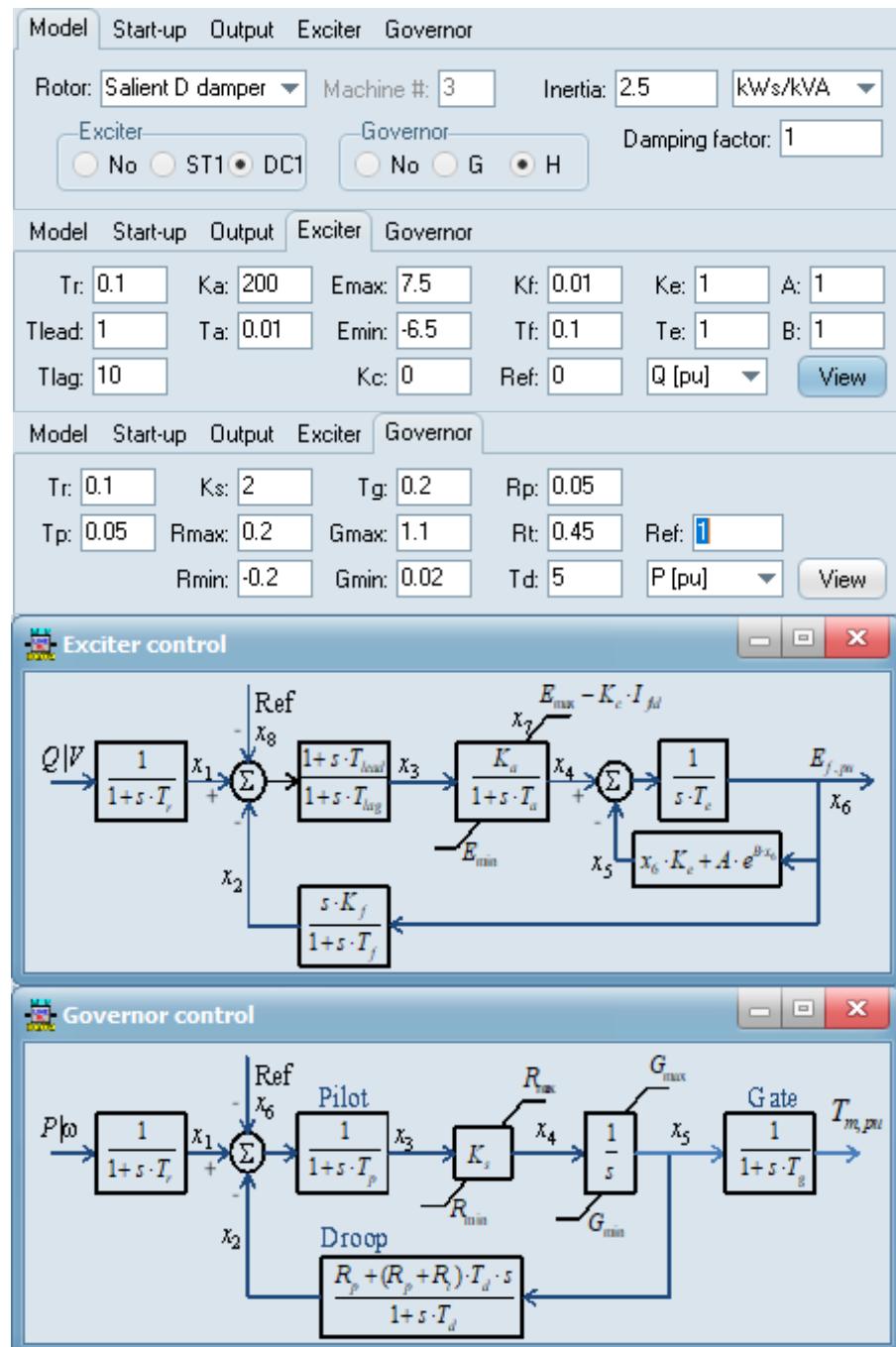


Fig. 5.19 – Windsyn Synchronous machine controls.

### 5.3 Using the integrated LCC object for line/cable modeling

The integrated LCC objects in ATPDraw are based on the LINE CONSTANTS, CABLE CONSTANTS or CABLE PARAMETERS supporting routines of ATP-EMTP. ATPDraw has in addition from v7.4 an option to calculate parameters internally as described in Chapt. 7.6. The user must first describe the geometry of the system and the material constants and ATPDraw then run ATP (or its internal routines) to process this data case and converts the output punch-file containing the electrical model of the line or cable into standard punch-file format. This pch-file will then be included in the final ATP-file via a \$INSERT call. The idea in ATPDraw is to hide as much as possible of the intermediate ATP execution and files and let the user work directly with geometrical and material data in the circuit. Only those cases producing an electrical model of the line or cable are supported in ATPDraw.

The user can either model each line/cable section individually or create a template that can be reused by many sections or variable length. Both these approaches start with choosing an LCC object under *Lines/Cables/LCC template*, as shown in Fig. 5.20. This will display a component in the circuit window that is connected to the circuit as any other component.

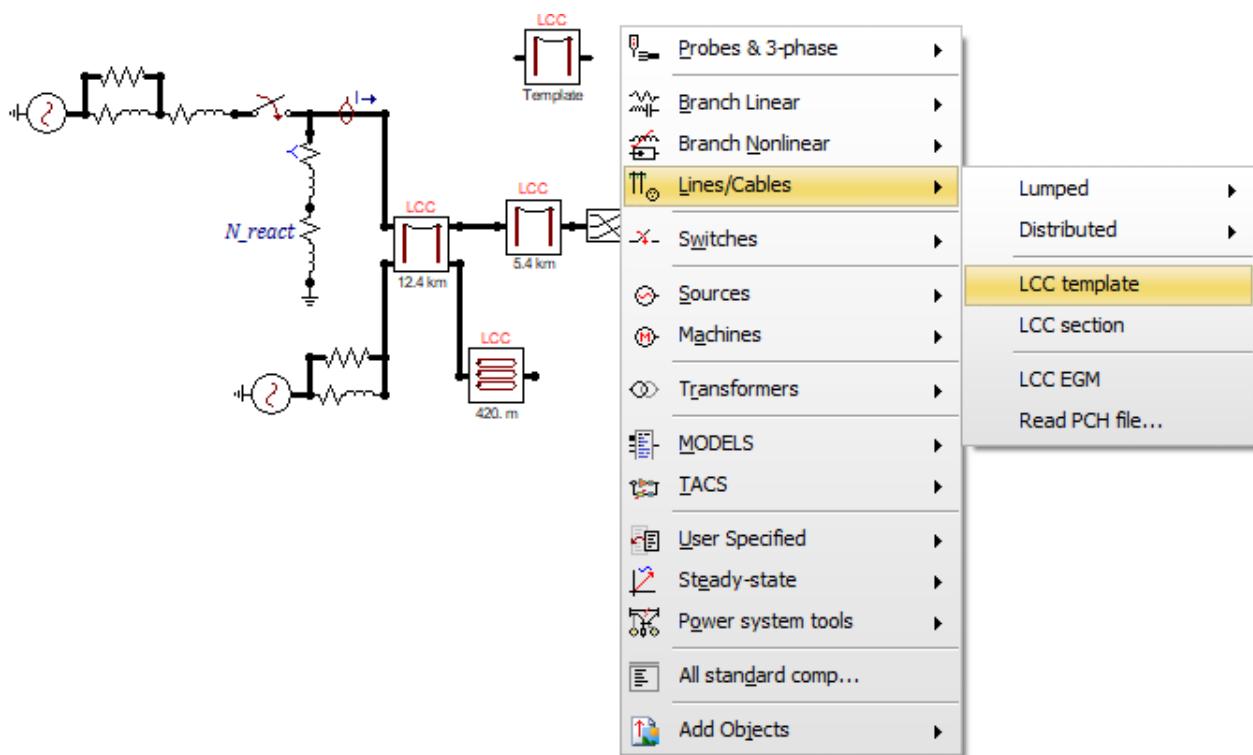


Fig. 5.20 - Selecting a line or cable and connecting the LCC object to the rest of the circuit.

Clicking on the LCC component with the right mouse button will show special input dialog box called the *Line/cable dialog* as shown in Fig. 5.21. This window contains three sheets: one for the various model specifications, one for the data (geometry and materials), and one for nodes where experts can change node sequence. The user specifies if the component should be Overhead Line, Single Core, Enclosing Pipe cables, or Matrix Import, and the number of phases (and cables) under the *System type* group. This choice will directly influence the grounding conditions in cable systems. The icon adapts setting of overhead line/single core cable/enclosing pipe and the number of phases. Under *System type* the user can also check *Template*. A *Template* is not written to the ATP-file, but its data can be used by *LCC Section* components. This is very useful when the same cross section and model is used in many sections of variable length.

Under *Standard data*, the ground resistivity (Rho), Starting frequency (Freq) and Length of the transmission line is given. Here is also the new option Internal calculation that let ATPDraw use internal routines for the parameters (bypass ATP) and enables the ULM model. It will also enable Matrix Import of series impedance and shunt admittance matrices. *Embed* is used to insert the line/cable model directly into the final ATP-file without \$INSERT. *Set length in icon* and *Single phase icon* influence the appearance of the object.

When the required data have been specified, the user can close the dialog by clicking on *OK*. The user is also asked if ATP (or internal routines) should be executed to produce the required punch-files. If the user answers *No* to this question, ATP is not executed, and the user is prompted again later when creating the final ATP-file under *ATP / run ATP*. You must give a name to the component. Template components must have unique names since this name is used as

identification by the LCC Sections. Otherwise, unique naming is recommended as debugging is easier in this case.

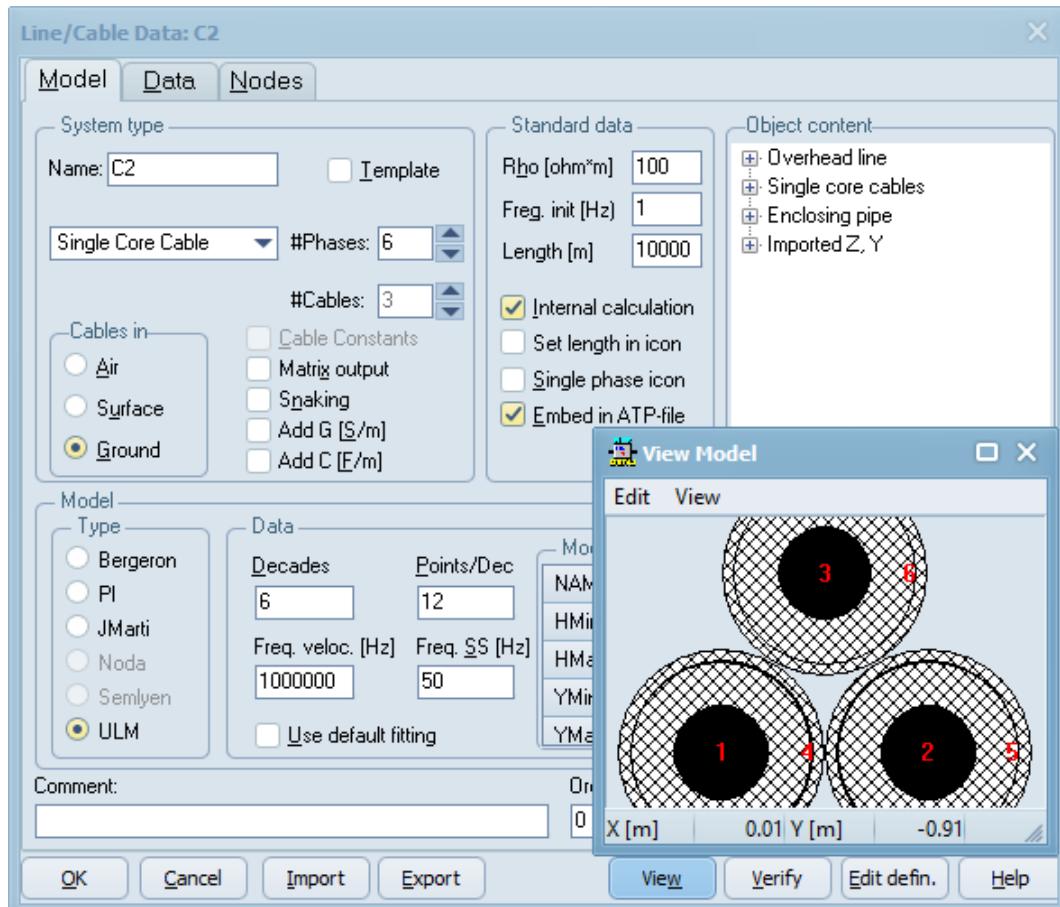


Fig. 5.21 - Line/Cable dialog box: Model specification. View- feature.

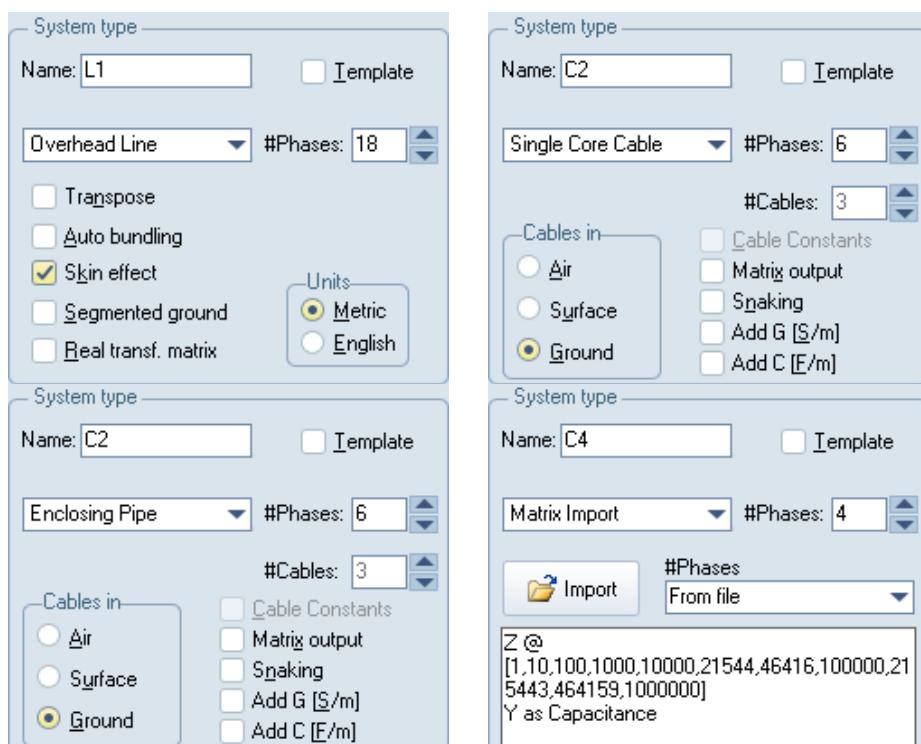


Fig. 5.22 – System type options.

The punch-file created by ATP with LINE CONSTANTS or CABLE PARAMETERS/CONSTANTS is immediately read from disc back into the component and stored in the project. When the final ATP-file is created the actual node-names are substituted in memory. Unless *Embed* is checked, the resulting punch (.pch) file will get the specified name followed by a unique counter number and stored in the Result Directory (same as the final ATP-file).

If something goes wrong in the generation of an electrical model an error message appears as shown in Fig. 5.23. Typical problems are missing or incorrect data (overlapping conductors for instance). You can inspect the intermediate files in the Result Directory (c:\atpdraw\atp in this case). File with extensions .dat (LINE/CABLE CONSTANTS or CABLE PARAMETER file), and .pch (result that is transformed into a .lib file) and the same name as the line/cable component should be present.



Fig. 5.23 – Model generation messages.

The data is stored internally in memory and the user can choose to export this data to an external library (typically the /LCC folder) by clicking the *Export* button. This data file is on a binary format and have extension .alc. You can click the *Import* button to load external data from disk. The Line&Cable component can also be copied between project as all other components. Clicking on the *View* button, displays the cross section of the line/cable as shown in Fig. 5.21. The phase numbers (with zero as ground) can be displayed in a red color via *View/Numbering*. For cables, the grounded conductors are drawn with a gray color, while the ungrounded conductors are black. The phase number is according to the rule of sequence: first comes the cable with the highest number of conductors and the lowest cable number. The thick horizontal line is the ground surface. Zooming and copying to the Windows clipboard is supported in metafile formats. The *Verify* button of the LCC dialog box helps the user to get an overview of the performance of the model in the frequency domain. This feature is described separately in sub-section 5.4.

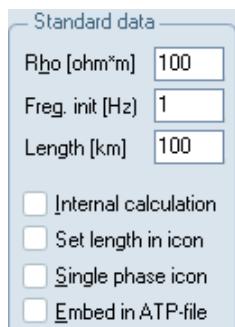
When creating a Noda line/cable model the Armafit program is executed automatically to create the required lib-file. The *Armafit command* is specified under *Tools / Options / Preferences*. The batch file runAF.bat is distributed with ATPDraw.

ATPDraw supports all the various electrical models: Bergeron (KCLee and Clarke), PI-equivalents, JMarti, Noda, Semlyen, and ULM. It is straightforward to switch between different models. Under *System type* the user can select between *Overhead Line* and *Single Core Cable* or *Enclosing Pipe*.

In the Line/Cable dialog the user can select between:

<b>System type:</b>	<b>Model / Type:</b>
<i>Overhead Line</i> : LINE CONSTANTS	<i>Bergeron</i> : Constant parameter KCLee or Clark
<i>Single Core Cables</i> :	<i>PI</i> : Nominal PI-equivalent (short lines)
CABLE PARAMETERS or CABLE CONSTANTS	<i>JMarti</i> : Frequency dependent model with constant transformation matrix
<i>Enclosing Pipe</i> :	<i>Noda</i> : Frequency dependent model (not supported in CABLE CONSTANTS)
CABLE PARAMETERS or CABLE CONSTANTS	<i>Semlyen</i> : Frequency dependent simple fitted model
<i>Matrix Import</i> :	<i>ULM</i> : Frequency dependent phase domain model, required Internal Calculation
Internal Calculation only	

The *Line/Cable Data* dialog of Fig. 5.21 really consists of three pages: *Model* page, *Line* or *Cable* page and *Node* page. The parameter names used in the LCC dialog boxes are identical with that of in Chapter XXI - LINE CONSTANTS and Chapter XXIII - CABLE CONSTANTS parts of the ATP Rule Book [3]. The *Standard data* of the Model page is common for all line and cable types and has the following parameters:



**Rho:** The ground resistivity in ohmm of the homogeneous earth (Carson's theory).  
**Freq. init:** Frequency at which the line parameters will be calculated (Bergeron and PI) or the lower frequency point (JMarti, Noda and Semlyen) of parameter fitting.  
**Length:** Length of overhead line in [m]/[km] or [miles]. Set length as a text in icon option.  
**Internal calculation:** Use internal routines for parameter calculations. Enables Matrix Import and ULM.  
**Embed in ATP-file:** Insert model directly in main ATP-file.

Fig. 5.24 - Standard data for all line/cable models.

### 5.3.1 Model and Data page settings for Overhead Lines

For overhead transmission lines the *System type* settings are as follows. High accuracy (FCAR=blank) is used in all cases. Specify the number of phases in the #Ph combo box.

**Transposed:** The overhead line is assumed to be transposed if the button is checked. Disabled for PI model type.

**Auto bundling:** When checked this enables the automatic bundling feature of LINE CONSTANTS.

**Skin effect:** If the button is checked skin effect is assumed (IX=4), if unchecked no skin effect correction. REACT option is set IX=0.

**Metric/English:** Switching between the Metric and English unit systems.

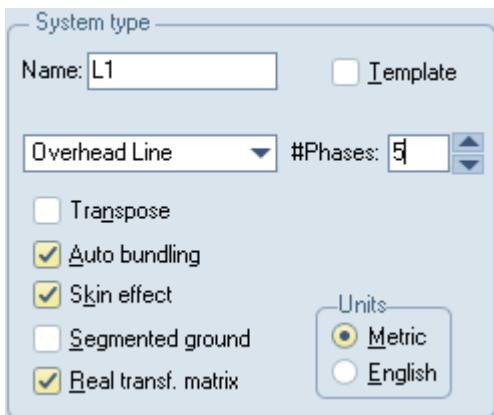


Fig. 5.25 - System type options for overhead lines.

### 5.3.1.1 Model Type settings

**Bergeron:** No additional settings are required.

**PI:** For nominal PI-equivalent (short) lines the following optional settings exist under *Data*:

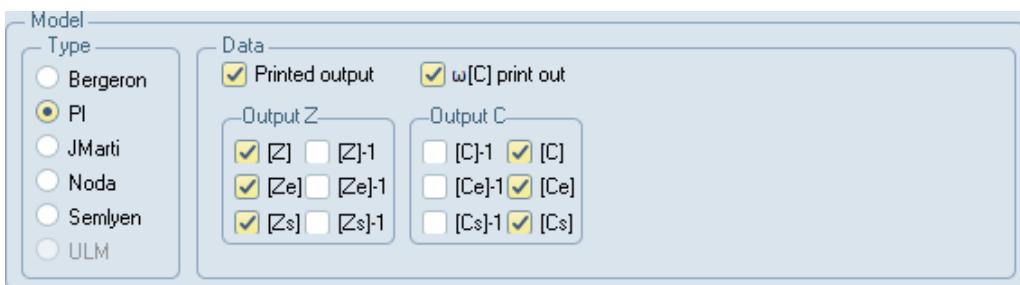


Fig. 5.26 - Optional settings for PI line models.

**Printed output:** If selected the shunt capacitance, series impedance/admittance matrix of the unreduced system, and/or of the equivalent phase conductor system (after elimination of ground wires and the bundling of conductors), and/or of the symmetrical components will be calculated.

**$\omega[C]$  print out:** Selection between the capacitance matrix and the susceptance matrix ( $\omega C$ ).

**JMarti:** The JMarti line model is fitted in a frequency range beginning from the standard data parameter *Freq. init* up to an upper frequency limit specified by the mandatory parameters number of *Decades* and the number of sample points per decade (*Points/Dec*). The model also requires a frequency (*Freq. matrix*) where the transformation matrix is calculated and a steady state frequency (*Freq. SS*) for calculation of the steady state condition. *Freq. matrix* parameter should be selected according to the dominant frequency component of the transient study. The JMarti model needs in some cases modification of the default fitting data under the optional *Model fitting data* field, that can be made visible by unselecting the *Use default fitting* check box. For further details please read in the ATP Rule Book [3].

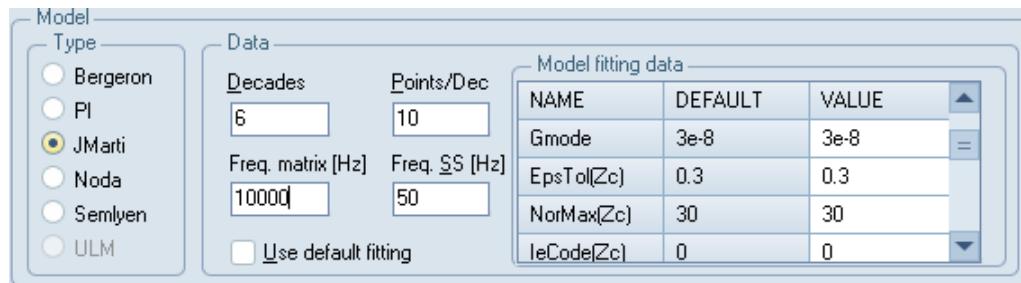


Fig. 5.27 - Parameter settings for the JMarti line model.

**Noda:** The Noda line model is fitted in a frequency range beginning from the standard data parameter *Freq. init* up to an upper frequency limit specified by the number of *Decades* with the resolution of *Points/Dec*. The model needs a frequency (*Freq. veloc.*), where the wave velocities of the natural modes of propagation are calculated. A value higher than the highest frequency of the frequency scan is usually appropriate. The Noda model needs in some cases modification of the default fitting data under the optional *Model fitting data* field, that can be made visible by unselecting the *Use default fitting* check box. For further details please read in the ATP Rule Book [3].

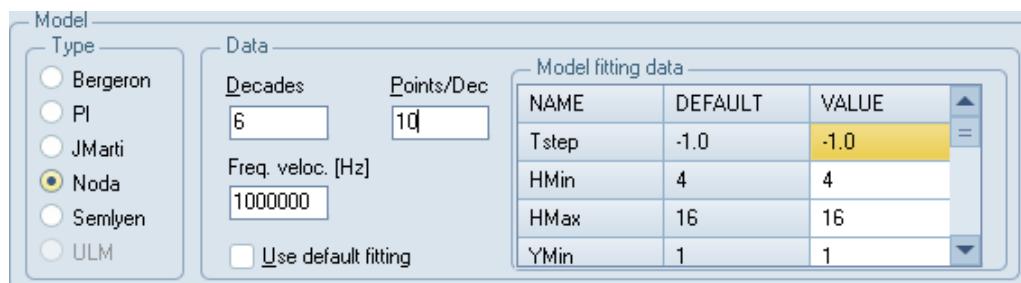


Fig. 5.28 - Parameter settings for the Noda line model.

**Semlyen:** The Semlyen line model is frequency dependent simple fitted model. Fitting range begins at the standard data parameter *Freq. init* and runs up to an upper frequency limit specified by the parameter number of *Decades*. The model also requires a frequency (*Freq. matrix*) where the transformation matrix is calculated and a steady state frequency (*Freq. SS*) for calculation of the steady state condition. *Freq. matrix* parameter should be selected according to the dominant frequency component of the transient study. The Semlyen model needs in some cases modification of the default fitting data under the optional *Model fitting data* field, that can be made visible by unselecting the *Use default fitting* check box. For more details please read in the ATP Rule Book.

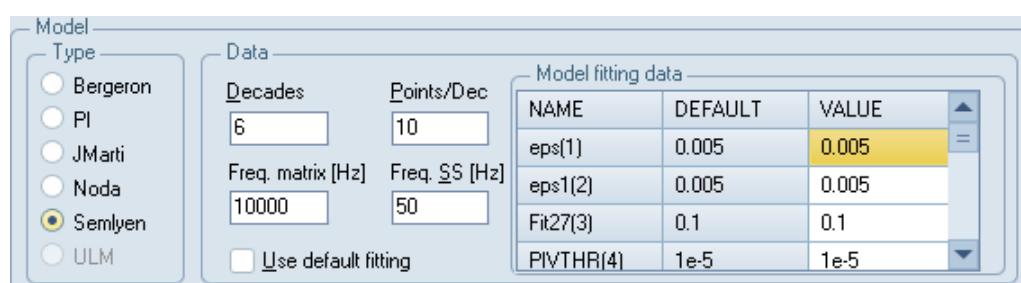


Fig. 5.29 - Parameter settings for the Semlyen line model.

**ULM:** The Universal Line Model needs Internal Calculations to be checked under Standard data. This model is a frequency dependent phase domain model baster of Vector Fitting to obtain a rational function approximation of characteristic admittance and propagation matrix with real and complex conjugated poles. This is the only model that enables accurately calculation of cable

screen voltages at wide-band frequencies. The fitting range begins at the standard data parameter *Freq. init* and runs up to an upper frequency limit specified by the parameter number of *Decades*. The frequency range 1 Hz to 1 MHz is recommended. Freq. veloc. is where the modes of the propagation is extracted (see Chapt. 7.6.3) and should be a high frequency ~1MHz. The Model fitting data (Fig. 5.30) sometimes needs to be adjusted to obtain a very good fit and are now explained in detail:

**Hmin:** The vector fitting starts with this number of poles for the propagation matrix ( $H$ ) and increments in steps of two until the accuracy **epsH** (%) or **Hmax** is reached.

**Ymin:** The vector fitting starts with this number of poles for the characteristic admittance matrix ( $Y_c$ ) and increments in steps of two until the accuracy **epsY** (%) or **Ymax** is reached.

**epsDeg:** Mode of  $H$  are merged if their phase angle at *Freq. veloc.* deviates with less than **epsDeg** [degrees]. The average of the merged modes are used in the fitting of each element in  $H$ .

**Niter:** Vector fitting runs this number of steps in the pole identification step.

For some simple (typically overhead line) cases the number of merged modes tends to be low, so the number of poles (*Hmin*) should be increased to obtain a good fit for the elements  $H$  at low frequencies. Alternatively increase the number of *Points/Dec* or lower *epsDeg*.

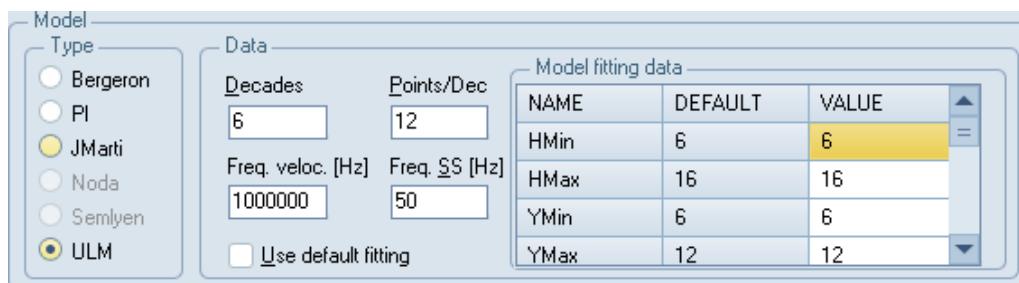


Fig. 5.30 - Parameter settings for the Universal Line Model.

### 5.3.1.2 Line Data page settings

The data page contains input fields where the user can specify the geometrical or material data. For overhead lines, the user can specify the phase number, conductor diameters, bundling, conductor positions, as shown in Fig. 5.31. The number of conductors is user selectable. ATPDraw set the grounding automatically or gives warnings if the grounding conditions do not match the fixed number of phases. You can *Delete last row* of the table using the gray buttons below or add a new one by clicking on the *Add row* command. Rows inside the table can also be deleted, but it must first be dragged down as last row. To drag a row click on its # identifier in the first column, hold the button down and drag the selected row to a new location or use the  $\uparrow$  and  $\downarrow$  arrows at right.

Line Data										
#	Ph.no.	Rin [cm]	Rout [cm]	Resis [ohm/km DC]	Horiz [m]	Vtower [m]	Vmid [cm]	Separ [cm]	Alpha [deg]	NB
1	1	0.55	1.55	0.0585	-17.5	27.9	13	60	45	4
2	2	0.55	1.55	0.0585	0	27.9	13	60	45	4
3	3	0.55	1.55	0.0585	17.5	27.9	13	60	45	4
4	0	0.3	0.8	0.304	-13.2	41.05	26.15	0	0	0
5	0	0.3	0.8	0.304	13.2	41.05	26.15	0	0	0

↑ ↓

Fig. 5.31 - Line Data dialog box of a 3-phase line. 4 conductors/phase + 2 ground wires.

**Ph.no.**: phase number. 0=ground wire (eliminated by matrix reduction).

**Rin**: Inner radius of the conductor. Only available if *Skin effect* check box is selected on the *Model* page (see in Fig. 5.25). If unselected, the *Rin* column is removed and a *React* column appears, where the user specifies the AC reactance of the line in ohm/unit length.

**Rout**: Outer radius (cm or inch) of the conductor.

**RESIS**: Conductor resistance (ohm/unit length) at DC (with *Skin effect* checked) or AC resistance at Freq. init (if no *Skin effect* selected).

**Horiz**: Horizontal distance (m or foot) from the centre of bundle to a user selectable reference line.

**Vtower**: vertical bundle height at tower (m or foot).

**Vmid**: vertical bundle height at mid-span (m or foot). The average conductor height calculated from the eq.  $h = 2/3 \cdot Vmid + 1/3 \cdot Vtower$  is used in the calculations.

If *System type / Auto bundling* is checked on the *Model* page (see Fig. 5.25):

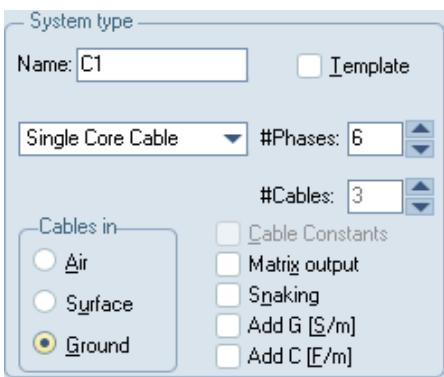
**Separ**: Distance between conductors in a bundle (cm or inch)

**Alpha**: Angular position of one of the conductors in a bundle, measured counter-clockwise from the horizontal line.

**NB**: Number of conductors in a bundle.

### 5.3.2 Model and Data page settings for Single Core Cable systems

Support of CABLE CONSTANTS and CABLE PARAMETERS has been added to the LCC module of ATPDraw recently and the user can select between the two supporting programs by a single button switch. This enables a more flexible grounding scheme, support of Semlyen cable model instead of Noda and the cascade PI section. On the other hand, in CABLE CONSTANTS enabled state ATPDraw does not support additional shunt capacitance and conductance input and Noda model selection. The CABLE CONSTANTS and CABLE PARAMETERS support in ATPDraw does not extend to the special overhead line part and the multi-layer ground model. For Class-A type cable systems which consists of single-core (SC) coaxial cables without enclosing conducting pipe the *System type* settings are as follows. Specify the number of phases in the #Ph combo box.



**Cables in:** Select if the cables are in the air, on the earth surface or in ground.

**Number of cables:** Specify the number of cables in the system.

**Cable constants:** Selects between Cable Constants and Cable Parameters option. If checked, the additional conductance and capacitance option will be switched off and the *Ground* options on the *Cable Data* page will be activated. The Semlyen model is supported only with Cable Constants and the Noda model only with Cable Parameters.

Fig. 5.32 - System type options for SC cables.

**Matrix output:** Check this button to enable printout of impedance and admittance matrix data ( $R$ ,  $\omega L$  and  $\omega C$ ).

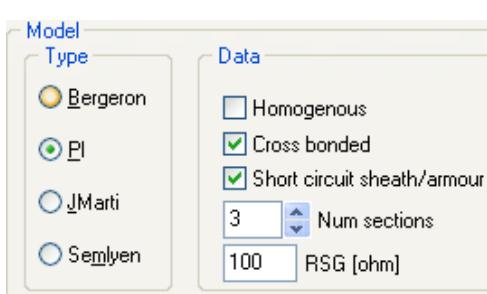
**Snaking:** If checked the cables are assumed to be transposed.

**Add G:** Check this button to allow conductance between conductors.  
Not supported for Cable Constants.

**Add C:** Check this button to allow additional capacitance between conductors.  
Not supported for Cable Constants.

### 5.3.2.1 Model Type settings for SC cables

**Bergeron, JMarti, Noda, Semlyen and ULM:** The *Model/Type* and *Data* settings for these SC cable models are identical with that of the overhead transmission lines as described in section 5.3.1.1. Users are warned however, that the frequency dependent models may produce unrealistic results, due to neglecting the frequency dependency of the transformation matrix, which is acceptable in overhead line modeling but not for cables.



Cascade **PI** model:

If the *Cable Constants* option is selected under the *System type* field, the PI model supports additional input parameters to produce cascade PI-equivalents. The cascade PI model is described in the ATP Rule Book [3]. The *Homogenous* type can be used with all grounding schemes.

Fig. 5.33 - SC cable data for cascade PI output.

### 5.3.2.2 Cable Data page settings for SC cables

The data page contains input fields where the user can specify the geometrical or material data for cables. The user can turn on sheath/armor by a single button and allowed to copy information between the cables. The cable number is selected in the top combo box with a maximum number specified in *Number of cables* in the Model page.

For CABLE PARAMETERS (*Cable Constants* unselected) the *Ground* options are inactive, and the number of grounded conductors is calculated internally in ATPDraw based on the total number of conductors in the system and the number of initially selected phases. For CABLE CONSTANTS (*Cable Constants* check box is On) the user must specify which conductor is grounded by checking the appropriate *Ground* buttons. A warning will appear if a mismatch between the number of phases and the number of ungrounded conductors is found. Grounded conductors are drawn by gray color under *View*. Selecting *View/Numbering* will show the phase number in red color (0=grounded). The cables will be sorted internally according to the sequence rule of ATP;

the cable with most conductors comes first. To avoid confusion and mismatch between expected phase number and conductors the user should try to follow this rule also in the Cable/Data dialog. The *Nodes* page allows the user to rearrange the phase sequence.

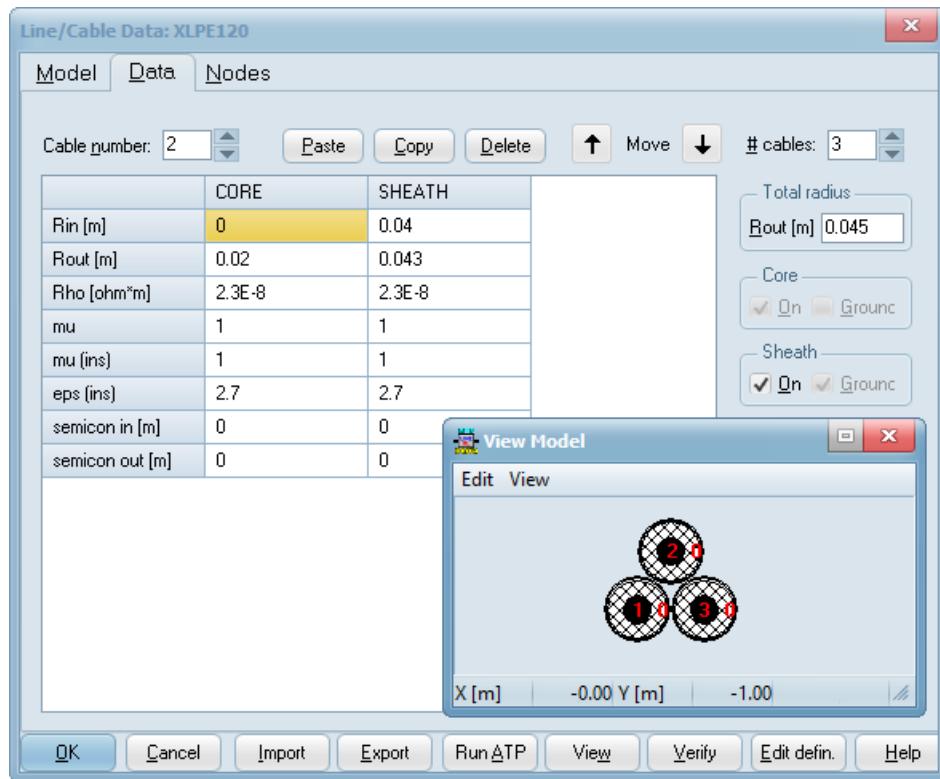


Fig. 5.34 - Cable Data dialog box for a 3-phase SC type cable system.

For each of the conductors Core, Sheath and Armor the user can specify the following data:

**R<sub>in</sub>:** Inner radius of conductor [m].

**R<sub>out</sub>:** Outer radius of conductor [m].

**Rho:** Resistivity of the conductor material.

**mu:** Relative permeability of the conductor material.

**mu (ins):** Relative permeability of the insulating material outside the conductor.

**eps (ins):** Relative permittivity of the insulating material outside the conductor.

**semicon in:** Thickness of inner semiconductor layer in [m].

**semicon out:** Thickness of outer semiconductor layer in [m].

The semicon parameters are used to calculate an equivalent permittivity.

**Total radius:** Total radius of the cable (outer insulator) [m].

**Sheath/Armor On:** Turn on optional Sheath and Armor conductors.

**Position:** Vertical and horizontal positions relative to ground surface and to a user selectable reference line for single core cables.

### 5.3.3 Model and Data page settings for Enclosing Pipe type cables

This selection specifies a cable system consisting of single-core (SC) coaxial cables, enclosed by a conducting pipe (referred as Class-B type in the ATP Rule Book [3]). The cable system might be located underground or in the air. The *System type* settings are identical with that of the Class-A type cables (see in sub-section 5.3.2). When the button *Cable Constants* is checked the shunt conductance and capacitance options are disabled and a new check box *Ground* controls the grounding condition of the pipe. Transposition of the cables within the pipe is available via the *Snaking* button. Cascade PI options can be specified similarly to SC cables (see Fig. 5.33). For cables with enclosing pipe, the following *Pipe data* are required:

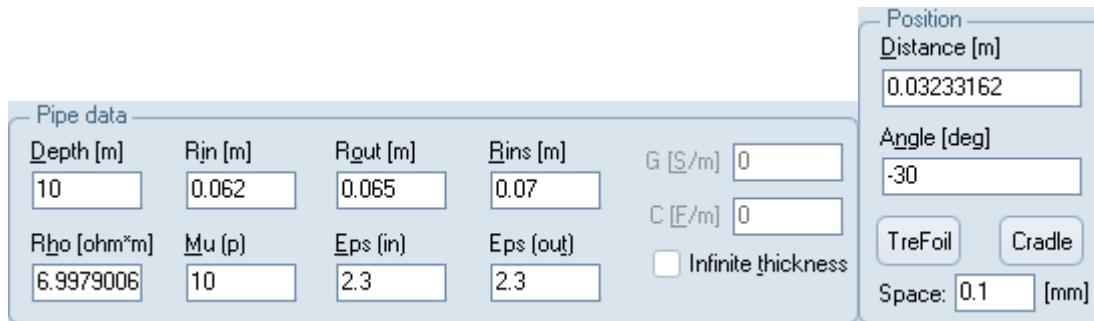


Fig. 5.35 - System type and Pipe data settings for an Enclosing Pipe cable.

**Depth:** Positive distance in meter between pipe center and ground surface.

**Rin:** Inner radius of the pipe in meter.

**Rout:** Outer radius of the pipe in meter.

**Rins:** Outer radius of outer insulation (total radius) in meter.

**Rho:** Resistivity of the pipe conductor.

**Mu:** Relative permeability of the pipe conductor.

**Eps (in):** Rel. permittivity of the inner insulation (between cables and pipe).

**Eps (out):** Rel. permittivity of the outer insulation (around pipe).

**G** and **C:** Additional shunt conductance and shunt capacitance between the pipe and the cables.

**Infinite thickness:** Infinite thick pipe. ISYST=0 and (uniform grounding).

The cable *Data* page input fields for Enclosing Pipe type cable systems are identical with that of the SC cables (see sub-section 5.3.2.2). The only difference is the meaning of *Position*:

**Position:** Relative position to pipe center in polar coordinates (distance and angle).

ATPDraw offers two position routines for 3-cable system in pipe (and also for single core cables). *TreFoil* centers the three cables in the pipe center and *Cradle* at the pipe bottom. The three cables are placed in a tight configuration with the spacing as given in *Space*.

### 5.3.4 Matrix Import

This option requires usage of Internal Calculation. Series impedance and shunt admittance matrices as function of frequency can then be imported in a XML-format given below.

The user must select *NumPhases* action before import with three possibilities:

- The imported file determines the number of phases.
- The model as specified above determines the number of phases and the matrices in the file are either reduced (normal case, conductors above *NumPhases* given by model are assumed to be on ground potential carrying current) or
- ignored (the conductors above *NumPhases* are assumed to be on ground potential carrying zero current (same as segmented ground in Line Constants)).

The following (case sensitive) XML-format is defined:

```
<ZY NumPhases="3" Length="1" ZFmt="R+Xi" YFmt="C">
<Z Freq="10">
  0.001+0.001i, 0.001+0.001i, 0.001+0.001i
  0.001+0.001i, 0.001+0.001i, 0.001+0.001i
  0.001+0.001i, 0.001+0.001i, 0.001+0.001i
</Z>
```

```

<Z Freq="1000">
  0.1+0.1i, 0.1+0.1i, 0.1+0.1i
  0.1+0.1i, 0.1+0.1i, 0.1+0.1i
  0.1+0.1i, 0.1+0.1i, 0.1+0.1i
</Z>
//Etc. in increasing order of frequency.
//Typically 3 points per decade are needed in the frequency range of interest.
<Y Freq="1">
  0.1e-6, -0.01e-6, -0.01e-6
  -0.01e-6, 0.1e-6, -0.1+0.1
  -0.01e-6, -0.01e-6, 0.1e-6
</Y>
</ZY>

```

The property ZFMt can be ["R+Xi", "R+iX", "R+Xj", "R+jX"],

The property YFmt can be ["C", "G+Bi", "G+iB", "G+Bj", "G+jB"].

With YFmt="C" the (Maxwell) capacitance is given in Farad and frequency does not matter.

The frequency points must be the same for Z and Y (unless YFMt="C" is used).

Length="1" means data per meter length, and Length="1000" means per km length.

The data for Z and Y are shown on the Data page and plotted under View.

The models PI, Bergeron, JMarti and ULM can be selected. JMarti and ULM will require Z and Y given at user selected points in the frequency range of interest. This would also be needed for PI and Bergeron unless the exact desired frequency point is provided. Vector Fitting is used to approximate the elements in Z and Y with rational functions  $\sum_n c_{ij,n} / (s - a_n) + d_{ij}$  for resampling.

### 5.3.5 Node page settings

The *Node* page was introduced in ATPDraw version 5.3. Normally, the user does not need to specify anything on this page. It gives, however, access to the node names of the LCC component and offers the user to assign conductor numbers to the nodes. Conductor numbering can be desirable for cables since ATP requires a special sequence in this case; first comes the cores, then the sheaths then the armors. The cables with most conductors must be numbered internally in ATP as the first cable. To avoid too much confusion the user should also try to follow this rule. For overhead line the user specifies the conductor number directly in the data grid and there should be no need to alter this.

A cable system consisting of 3 single core cables with sheaths and a fourth ground wire will as default receive an "unexpected" phase sequence. The core of the three cables will be numbered 1-2-3, then the ground wire will be numbered 4, and finally the three sheaths will be numbered 5-6-7. This does not fit well with the 3-phase layout used for this 7-phase system. The core of the cables will all be a part of IN1/OUT1-ABC, but then the ground wire will become IN2A/OUT2A, the cable sheaths 1 and 2 will be IN2B/OUT2B and IN2C/OUT2C and the third cable sheaths will be connected to the single-phase nodes IN3/OUT3. To let the ground wire be connected to the single-phase node the conductor sequence 1-2-3-5-6-7-4 can be assigned in the grid.

The *View* module has a *Number* feature that displays the conductor numbers.

### 5.3.6 LCC Section

This component (LCC\_) requires an LCC Template to be defined first. The component uses the data of the template but can change its Standard data; length, frequency and ground resistance. The *Frequency* and *Ground resistivity* can also be chosen to be inherited from the template. Particularly *Frequency* could be easily misunderstood for JMarti and ULM models as it is the

starting frequency only. The component is very useful when the same line/cable cross section is used in several sections of variable length as the data can be changed only in the template to affect all sections. In the *LCC Section dialog* the user selects the name of the template to use (Use As not really needed) and usually sets the length of the section.

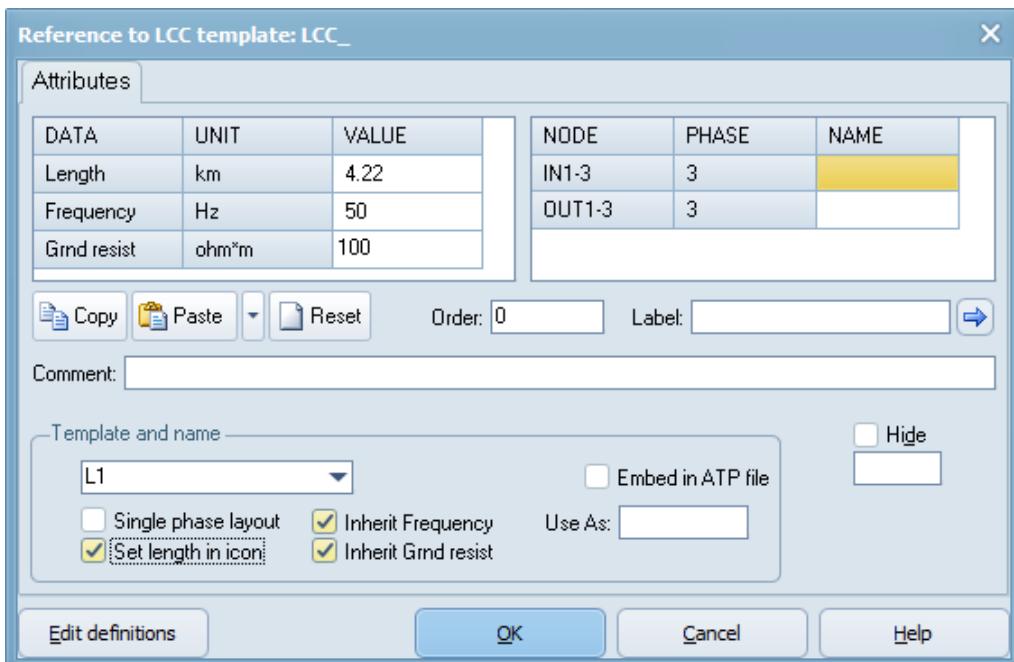


Fig. 5.36 – LCC Section input dialog

### 5.3.7 Selection of model type

As shown above there are many models available to choose from. One part of the reason for the variety comes from developments over time, another reason is model accuracy. The choice of model is also linked to application area and calculation speed.

**PI:** This model uses parameters calculated at a fixed frequency and with shunt admittance concentrated at each terminal. Such model is the preferred choice for pure power system frequency studies. For long lengths, the transmission line can be split up in sub-sections to eliminate the effect of concentrated shunt admittance (capacitance). Cables need shorter lengths of sub-sections than overhead lines. Sections of a few kilometers length is sufficient for power system frequency studies.

**Bergeron:** This model calculates the parameters at a fixed frequency but distributes the shunt admittance to create a travelling wave model. However, there is a transformation to modal domain involved that reduces the accuracy compared to cascaded PI-equivalents for fixed frequencies. This model is the preferred choice for many lightning studies when investigation the dominant mode of the voltages is sufficient. In such studies there are also many uncertain parameters related to the lightning strike that limits the value of correct damping and worst-case scenarios are suitable. The model is fast and thus enables comprehensive Monte Carlo studies.

**JMarti:** This is a frequency dependent, distributed parameter model. The model is implemented in the modal domain and assumes constant transformation matrices. The fitting of the modal quantities sometimes results in poor accuracy at low frequency that gives some problems when both power frequency and transient modeling are required. The JMarti model is beneficial when losses/damping of transients is important (lightning and switching transient fault analysis). The model is highly accurate if the frequency of the transient is known since the transformation matrix can be calculated at this frequency. In ATPDraw 7.4 JMarti models could also be fitted with Vector Fitting for somewhat improved low-frequency accuracy.

**Semlyen:** This model was the first frequency dependent model supported, but is nowadays not much use. It is included for historical and backward compatibility reasons.

**Noda:** This model is a phase domain, frequency dependent, distributed parameter model. It does not depend on a fixed transformation matrix and is in theory more accurate than the JMarti model. The fitting is not performed with rational functions but with polynomials in the z-domain using the Arma fitter. Stability of the fitting is thus the main concern that has reduced the applicability of the model. This model might not be optimally supported in ATPDraw either since symmetry specification of conductors is a manual procedure not covered.

**ULM:** This model is as the Noda model a phase domain, frequency dependent, distributed parameter model. The model is fitted with rational functions that ensures stability even with high order fit. The model depends on internal parameter calculations with a somewhat different ground return modeling for overhead lines. Presently, only the transient part of the model is implemented via a Type94 foreign model and no steady-state initialization is involved. This disables usage of *LineCheck* and makes power frequency parameter verification more difficult. The model is generally highly accurate and is the only choice for studies of transients in cable systems when induced voltages or screen potentials are of interest.

Examples Exe\_28-abc illustrate and compare the JMarti and ULM models for both overhead and cable configurations. For overhead lines, there are generally very small differences, while for cables systems the difference can be unacceptable with JMarti models. Fig. 5.37 shows the comparison between original JMarti model, JMarti with Vector Fitting and ULM models for a simple overhead line configuration (Exa\_28a). The observed differences are minimal.

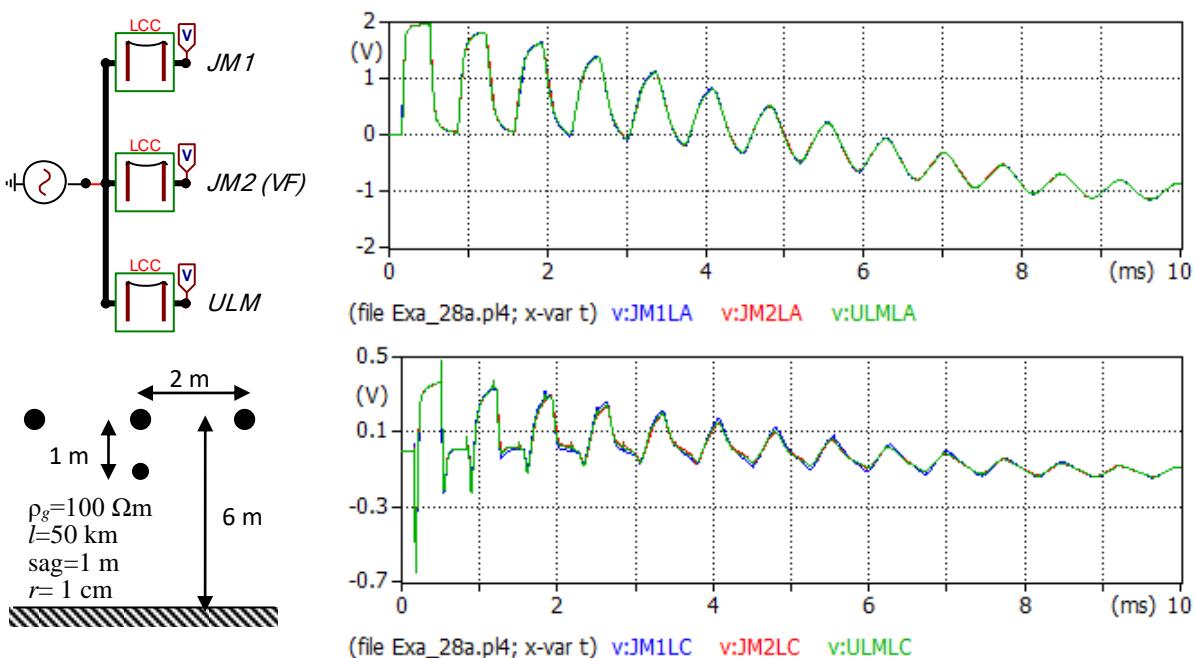


Fig. 5.37 – Comparison of JMarti and ULM line models for overhead lines.

Fig. 5.38 shows the comparison between original JMarti model, JMarti with Vector Fitting and ULM models for a simple underground cable configuration (Exa\_28c). When the screens are floating the original JMarti line produce highly incorrect induced voltages at the far end. In the figure the axis are cut and the peak of the induced voltage calculated by the JMarti model is 0.48 V. Fitting the model instead with Vector Fitting improves the situation, but voltages are still not as accurate as what is given by the ULM model.

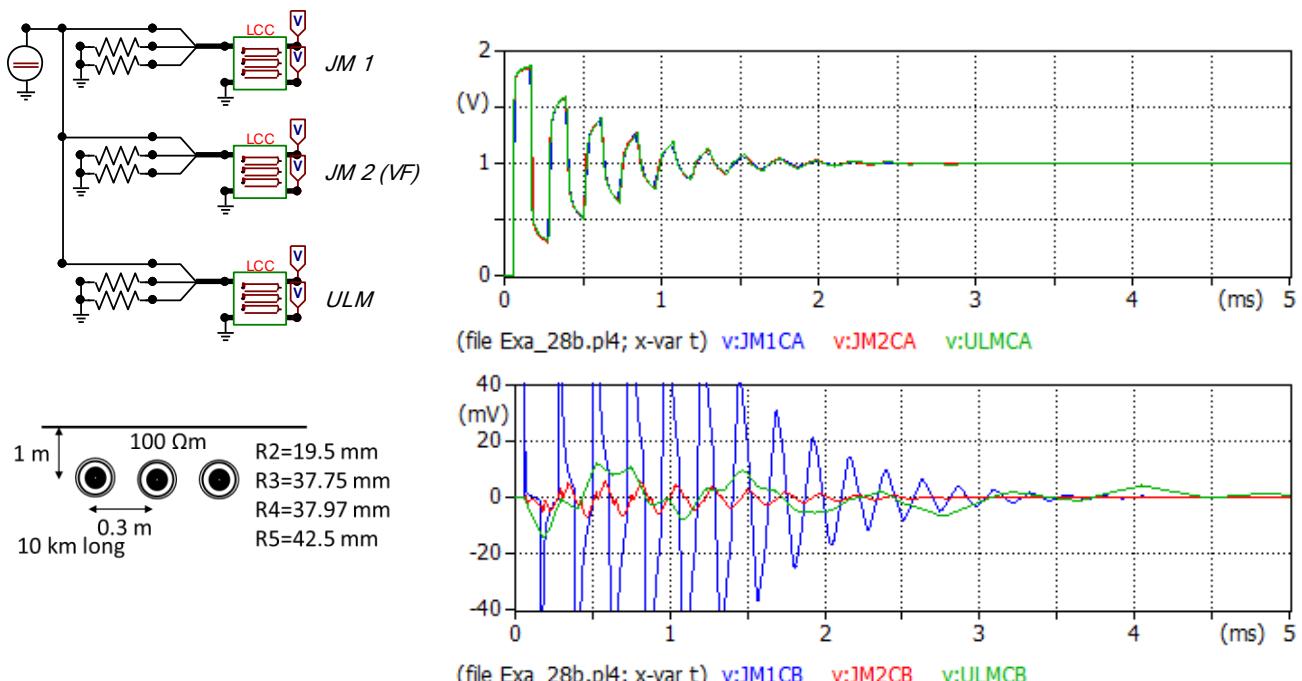


Fig. 5.38 – Comparison of JMarti and ULM line models for underground cables.

## 5.4 Verification of the Line/Cable model performance

A line or cable model can be verified in two different ways. Internally in the Line/Cable dialog there is a Verify module that supports both a frequency scan option and a power frequency calculation. Externally under ATP|Line Check there is a module that enables the user to select several sequential line section (including transposition) and perform power frequency calculations of series impedance and shunt admittance. This model is better for long lines.

### 5.4.1 Internal Line/Cable Verify

The *Verify* button of the LCC dialog box helps the user to get an overview of the performance of the model in the frequency domain. This feature of ATPDraw enables the user to compare the line/cable model with an exact PI-equivalent as a function of frequency, or verify the power frequency benchmark data for zero/positive short circuit impedances, reactive open circuit line charging, and mutual zero sequence coupling. The *Verify* module supports two types of frequency tests:

- 1) LINE MODEL FREQUENCY SCAN (LMFS) as documented in the ATP benchmark files DC51/52.dat. The LMFS feature of ATP compares the punched electrical model with the exact frequency dependent PI-equivalent as a function of a specified frequency range.
- 2) POWER FREQUENCY CALCULATION (PFC) of zero and positive short circuit impedances and open circuit reactive line charging, and mutual zero sequence impedance for multi circuit lines.

In the *Verify* dialog box as shown in Fig. 5.39 the user can choose between a LINE MODEL FREQUENCY SCAN (LMFS) or a POWER FREQUENCY CALCULATION (PFC) case. Under *Circuit specification*, each phase conductor is listed for which the user should assign a circuit number. The phase order for overhead lines is from the lowest phase number and up to the one assigned under *Data* in the Line/Cable dialog box. For cables, the cable with the highest number of conductors and the lowest cable number comes first (rule of sequence, ATP Rule Book - Chapter XXIII). A circuit number zero means that the conductor is grounded during the frequency test. For

the LMFS test the user must specify the frequency range (*Min freq* and *Max Freq*) along with the number of points per decade for the logarithmic space frequencies. For the PFC test, the input parameters are the power frequency and the voltage level (used to calculate the reactive line charging). There are ATP restrictions related the LMFS approach. If the ATP simulation hangs in the verification process, the user can press ESC.

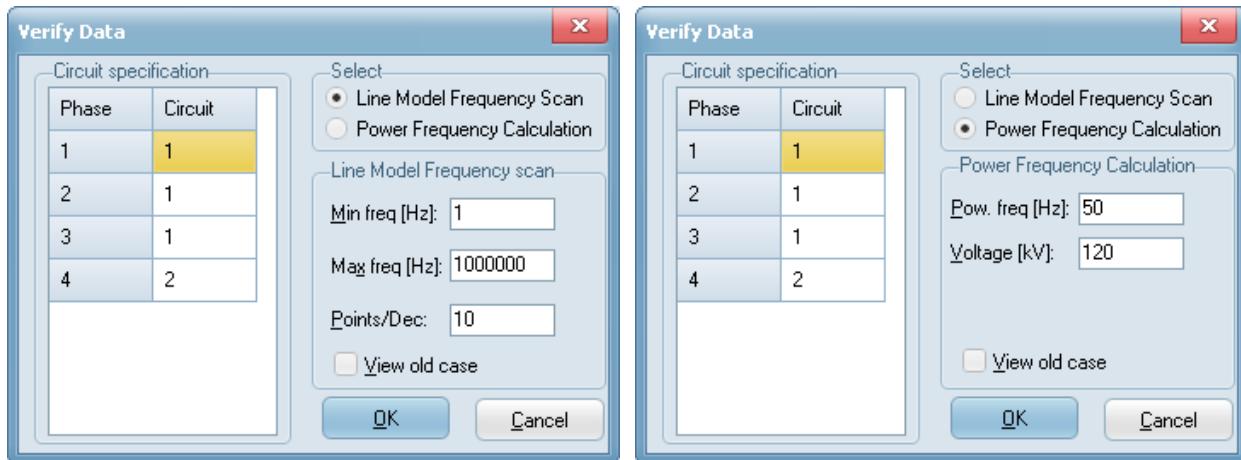


Fig. 5.39 - Frequency range specification for the LMFS run (left) and selecting the line voltage and system frequency for the PFC run (right).

- a) Select LMFS: Clicking on *OK* will result in the generation of a LMFS data case called `xVerify.dat` and execution of ATP based on the settings of the default ATP command (*Tools/Options/Preferences*). The sources are specified in include files called `xVerifyZ.dat`, `xVerifyP.dat`, and `xVerifyM.dat` for the zero, positive and mutual sequence respectively. The individual circuits are tested simultaneously. The receiving ends are all grounded (over 0.1 mΩ) and all sending ends (if *Circuit number > 0*) attached to AC current sources of 1 Amps. The phase angle of the applied current source for the  $i^{th}$  conductor is  $-360 \cdot (i-1)/n$  where  $n$  is the total number of conductors belonging to that circuit. Phase angle for the zero sequence tests are zero. The mutual coupling works only for 6-phase lines. For circuit one all phases are supplied with zero phase angle sources, while the phase conductors of the other circuit at the sending end are open. The *View old case* button will skip creation of the LMFS data case and trace the program directly to the procedure that reads the `xVerify.lis` file, which contains the input impedances of the electrical model compared to the exact PI-equivalent as function of frequency under various conditions. ATPDraw can read this file and interpretation of the results is displayed in the *LMFS results* window as shown in Fig. 5.41 for the 4-phase JMarti line-model specified in Fig. 5.40. LMFS relies on simulations in ATP and there are restrictions on how many phases (six?) can be managed. If the ATP simulation hangs for some reason, ATPDraw will wait in an eternal loop for it to finish. Break this loop by pressing ESC.

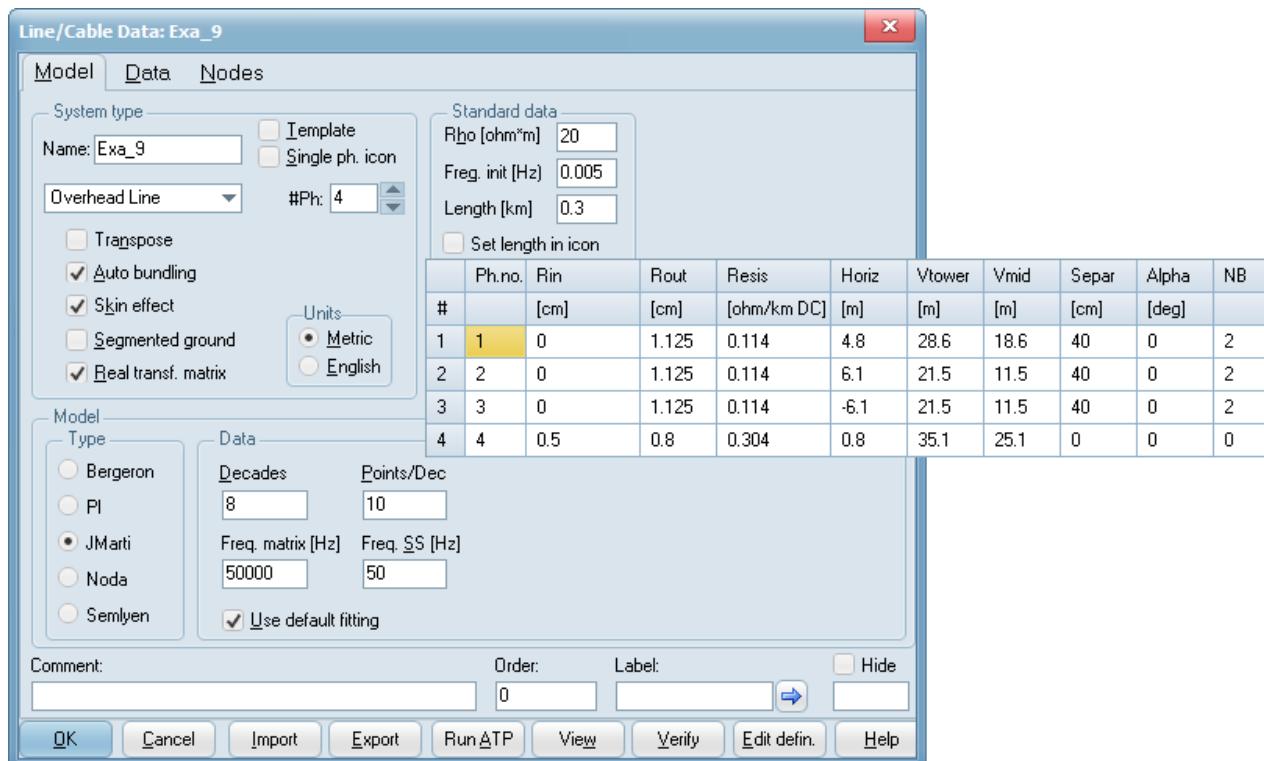


Fig. 5.40 - Specification of a 4-phase JMartí line model.

In Fig. 5.41, the user can select the *Mode* and the *Phase number* of which the absolute value of the input impedance is displayed to the left in a log-log plot. It is also possible to copy the curves to the windows clipboard in metafile format (*Copy wmf*). The absolute value of the input impedance of the model and the exact pi-equivalent can be compared for the following cases:

**Zero-sequence:** AC currents of 1 A with zero phase angle is applied to all phases simultaneously while the other end of the line/cable is grounded. The zero-sequence impedance is thus equal to the voltage on the sending end of each phase.

**Positive sequence:** AC currents of 1 A with a phase angle of  $-360*(i-1)/n$  is applied to all phases, where  $i$  is the current phase number in the specific circuit and  $n$  is the total number of phases in the circuit. (A 6-phase line/circuit will result in phase angles 0, -120, -240, 0, -120, -240 while a 4 phase circuit will result in 0, -90, -180, -270). The user specifies a circuit number for each phase under *Circuit specification* of *Verify Data* dialog. The receiving end is grounded.

**Mutual sequence:** AC currents of 1 A with zero phase angle is applied to all phases of the first circuit, while the other circuit is open. The receiving ends of all phases are grounded. Apparently, this works only for 6-phase lines.

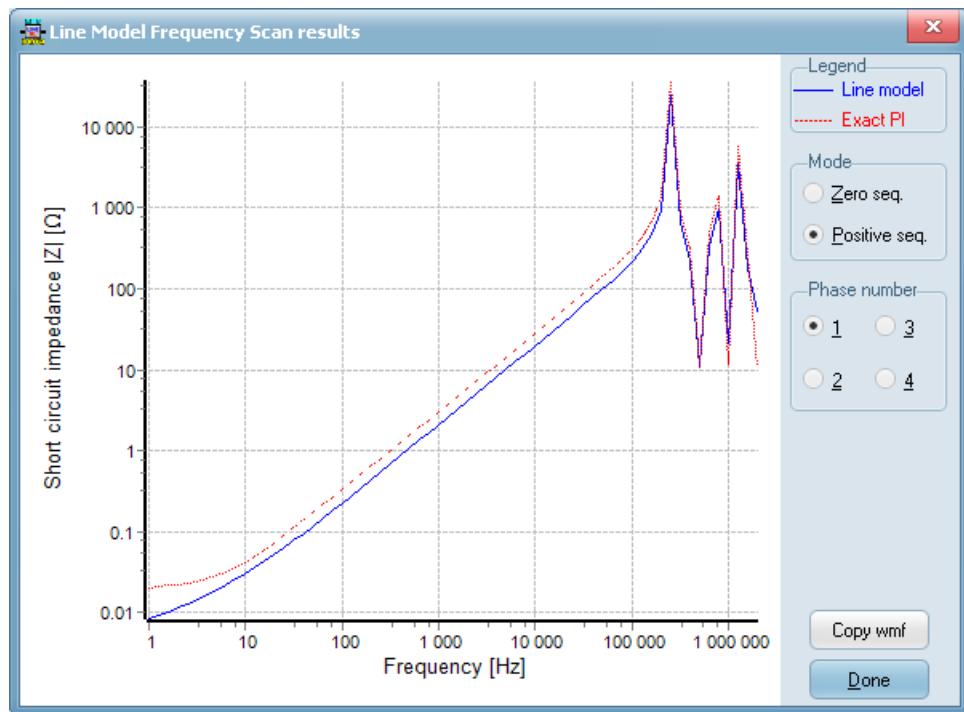


Fig. 5.41 - Verifying a JMarti line model 1 Hz to 1MHz. Model is OK for  $f > 25$  Hz.

b) Select PFC: For the PFC test the user must specify the power frequency and the base voltage level for scaling of the reactive charging. Clicking on *OK* will result in the generation of a PFC data case called `xVerifyF.dat` and execution of ATP based on the settings of the ATP-Command (*Tools / Options / Preferences*). In this case, each circuit is tested individually (all other phases are left open while a specific circuit is tested). The library file describing the electrical model of the line/cable is included in a new ATP case and supplied by unity voltage or current sources in order to calculate the steady state short circuit impedances and open circuit reactive line charging. The file `xVerifyF.lis` is read by ATPDraw and the short circuit impedances together with the open circuit line charging is calculated in the zero-sequence and positive-sequence mode. The results of the calculations are displayed in Fig. 5.42 .

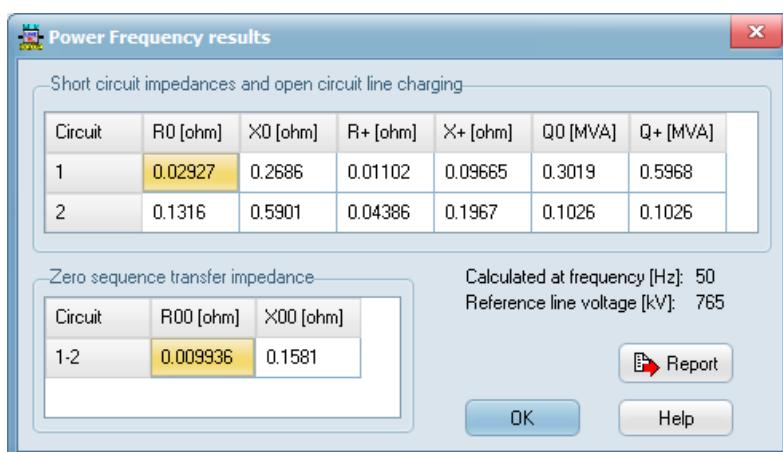


Fig. 5.42 - Results of the PFC run.

If the user clicks on *Report* the content in the string grids of Fig. 5.42 will be dumped to a user selectable text file. Further details about the operation of the *Verify* feature and PFC option can be found in the Appendix part of the Manual.

### 5.4.2 Tuning internal calculation of JMarti and ULM models

When Internal calculation is chosen and JMarti or ULM models selected, the *Verify* button will show plots of the fitting of the characteristic impedance/admittance ( $Z_c/Y_c$ ) and propagation ( $H$ ) matrices. First the fitting of the modes is shown (this is normally very accurate), then the fitting of the full matrices is shown. JMarti models will typically show some deviations at low frequencies and for small elements in  $H$ , while ULM shows much less deviations. These plots can be used to improve the fitting of the elements. For both, Vector Fitting [28-29] is used to calculate poles and residues for the rational approximations of the matrices. For JMarti the approach in [31] is used while for the ULM the original paper [30] is used and the practical implementation in [32] called.

For JMarti models the *Freq. init* value should be around 1 Hz. It makes little sense to go much below this since the accuracy of the fitting anyway will be poor in this region. Vector Fitting does not benefit from a low frequency range, and the risk could be incorrect calculations of the parameters instead worsen the situation. The number of decades should be in the range of 6 so that 1 MHz is reached. It is normally not beneficial to go much above 1 MHz, with exception of very short lines. But again, calculation of the parameters might be incorrect at very high frequencies, especially for magnetic materials. The parameter *Freq. matrix* is where the constant transformation matrix is calculated and should for JMarti models be at the dominant frequency of the transient. The number of points per decade should be in the range of 10-20. Default fitting could typically be used, since up to 30 poles are allowed for modes in  $Z_c$  and  $H$ . The accuracy of the mode fitting is set by default to 0.3%. The JMarti fitting is for cable systems typically not accurate at low frequency so there is limited value in fine tuning the parameters.

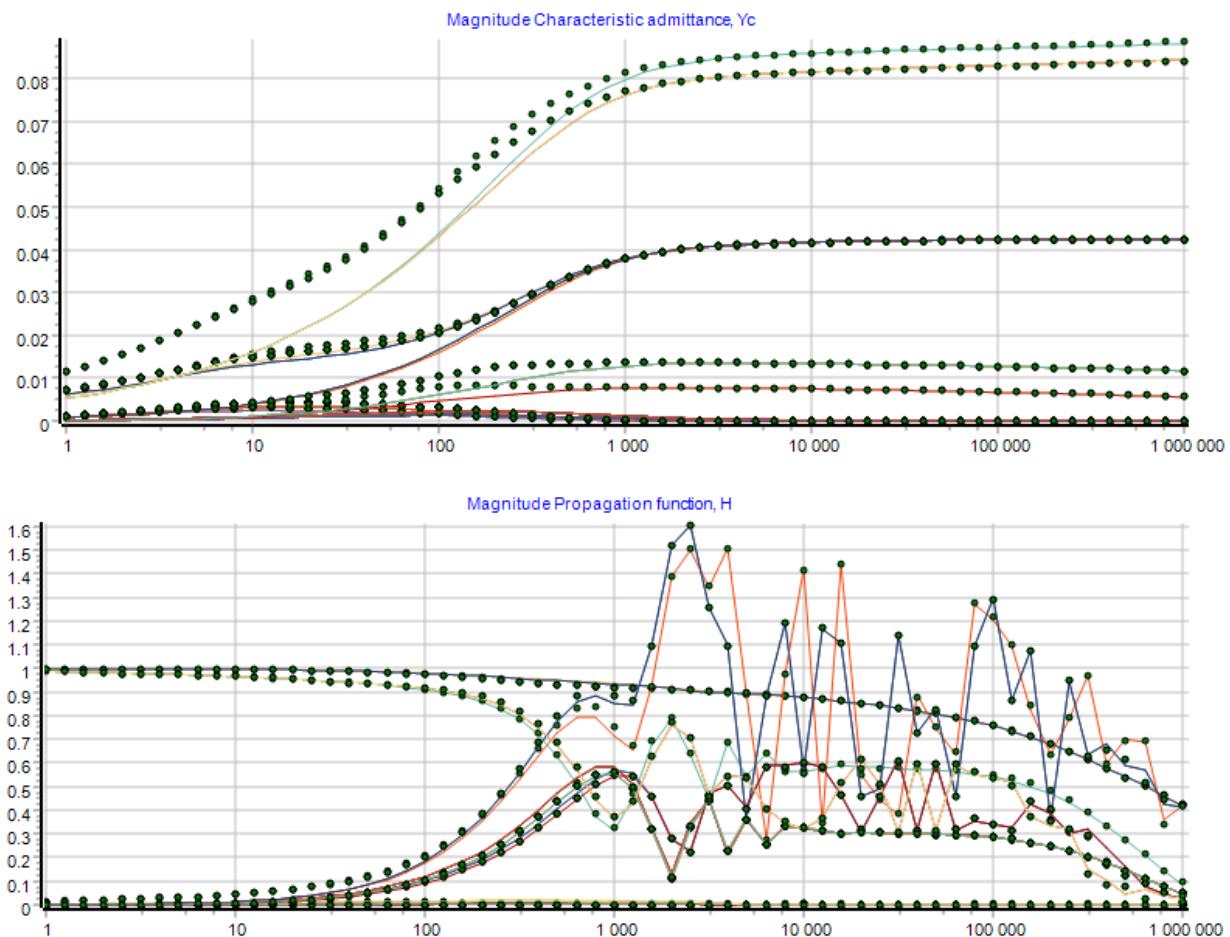


Fig. 5.43 JMarti model of the 10 km cable model from [30] by Vector Fitting. The phase cable with open screens. Modes in  $Y_c$  (top) {10-30-14-6-6-6}. Modes in  $H$  (bottom) {12-8-10-8-12-8}. *Freq-matrix* = 1000 Hz.

For ULM models, the same frequency range 1 Hz – 1 MHz should be considered. Now, the *Freq. veloc.* parameters should be a high frequency around 1 MHz. This is where the transformation matrix for the modes of  $H$  is calculated (trick to avoid eigenvalue switchover issues, see Chapt. 7.6.3). This results in somewhat inaccurate modes at low frequencies, but this is compensated by the accurate calculation of the residues of the full  $H$ -matrix. The points per decade can also in this case be 10-20. A higher number improves the accuracy of the fitting, especially at low frequencies, but slows down the fitting process somewhat. For ULM, the characteristic admittance matrix  $Y_c$  is fitted with a common pole set, and the accuracy of this is typically very high. The modes of  $H$  are merged based on an angle difference criterion at high frequencies and the total number is thus typically reduced. The accuracy of fitting the magnitude of  $H$  is typically high. Deviations are mostly seen in the angles at low frequencies where the magnitudes of off-diagonal elements tends to zero with low numerical accuracy as result. The accuracy can typically be improved by increasing the number of points per decade.

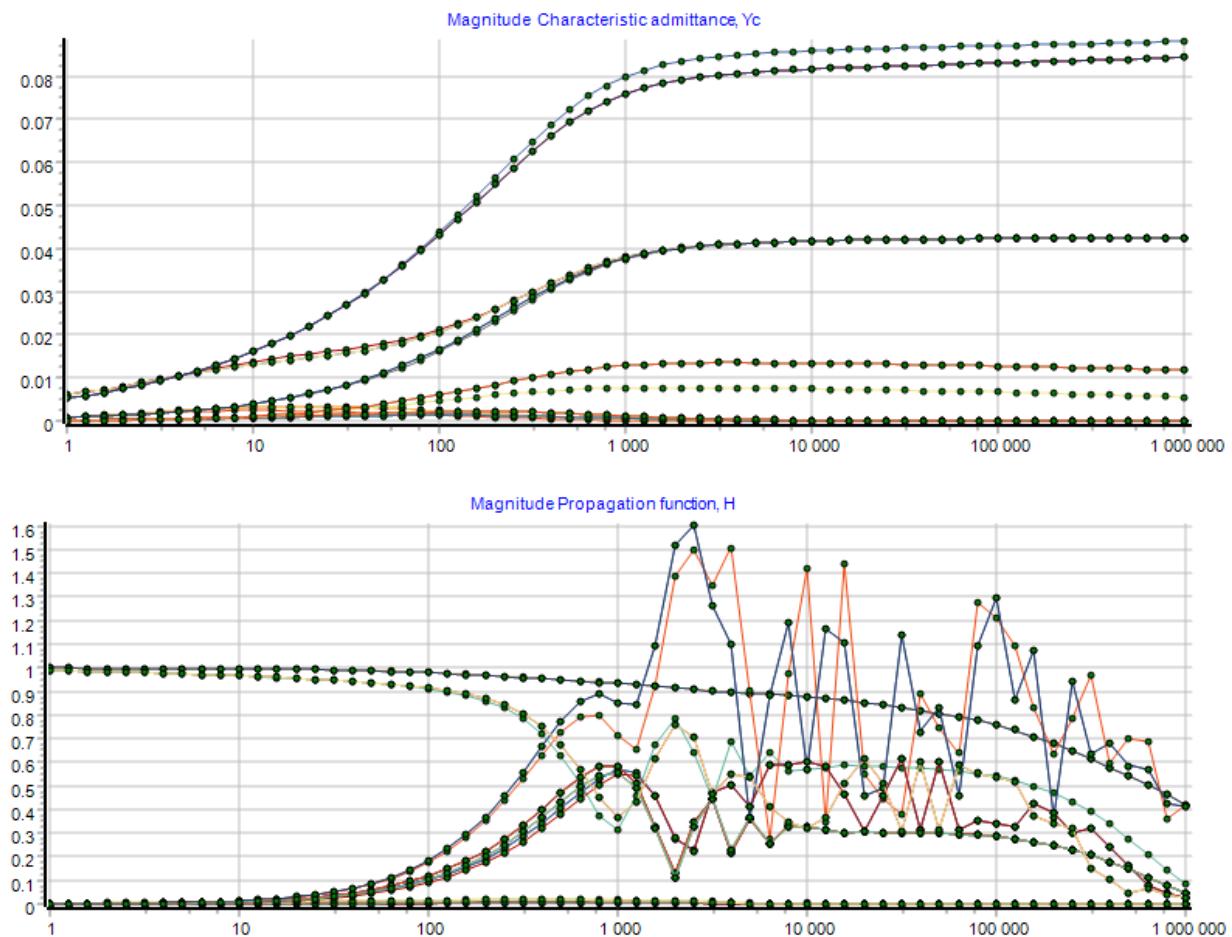


Fig. 5.44 ULM model of the 10 km cable model from [30] by Vector Fitting. Three phase cable with open screens. Modes in  $Y_c$  (top) {12}. Modes in  $H$  (bottom) {10-14-16-12}.

Sometimes the fitting parameters need to be adjusted as well and increasing the poles in  $H$  could be necessary. If the number of modes of  $H$  is low and the overall fitting of  $H$  bad the minimum number of poles ( $Hmin$ ) could be increased or the tolerance lowered from 0.1 to 0.01%. This could for instance be the situation in some trivial overhead line cases where only a single mode of  $H$  is uniquely identified. Another parameter is  $epsTol$  with a default value of 10 degrees used in the mode merging process. Reducing this could enforce more modes to stand out resulting in improved low frequency fit. Very bad fitting at low frequency could give passivity violations in the ULM model.

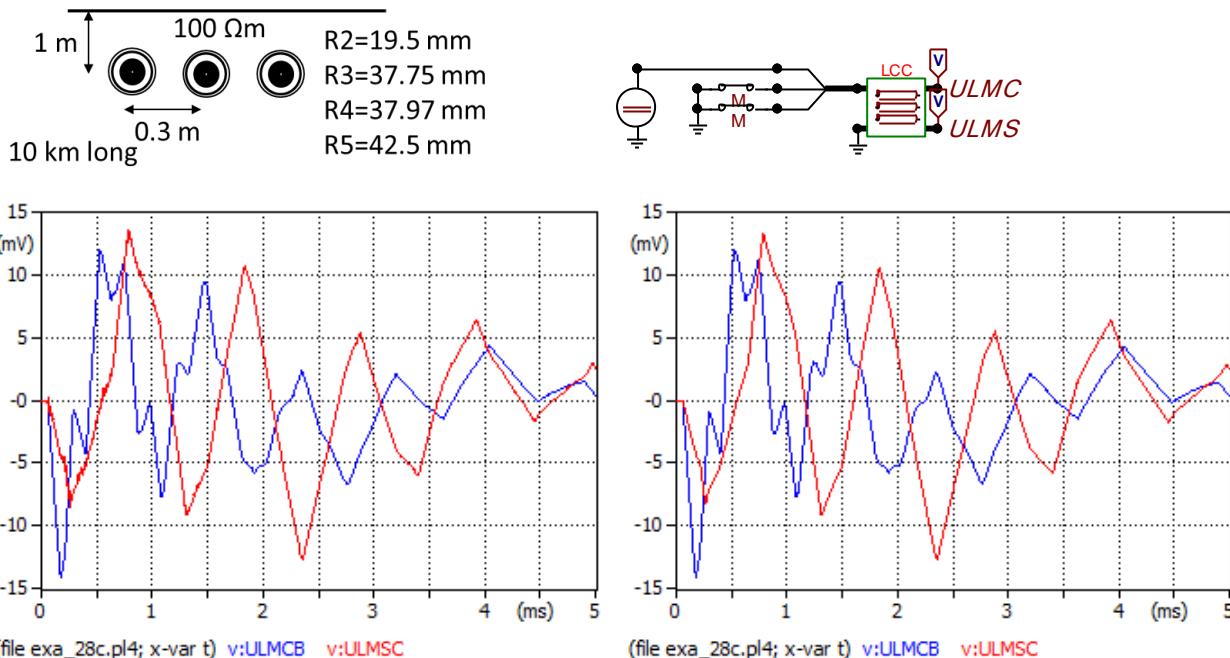


Fig. 5.45 Calculated induced voltage by ULM model [30] with 10 points per decade (left) and 20 points per decade (right). Small high-frequency oscillations removed in the figure to the right.

### 5.4.3 External Line Check

First, the user selects the line he wants to test and then clicks on *ATP/Line Check* as shown in Fig. 5.46. Then the input/output selection dialog box shown in Fig. 5.47 appears.

The LineCheck feature in ATPDraw supports up to 3 circuits. ATPDraw suggests the default quantities. The leftmost nodes in the circuit are suggested as the input nodes, while the rightmost nodes become the output. The circuit number follows the node order of the objects. For all standard ATPDraw components the upper nodes have the lowest circuit number. The user also has to specify the power frequency where the line/cable is tested. Finally, the user can check the *Exact phasor equivalent* button which will result in a slightly better results for long line sections.

When the user clicks on OK in Fig. 5.47 an ATP-file (/LCC/LineCheck.dat) is created and ATP executed. For a 3-phase configuration 4 sequential data cases are created ( $Z_+$ ,  $Y_+$ ,  $Z_0$ ,  $Y_0$ ) while for a 9-phase configuration 24 cases are created ( $Z_{11+}$ ,  $Y_{11+}$ ,  $Z_{110}$ ,  $Y_{110}$ ,  $Z_{12\dots}$ ,  $Z_{22\dots}$ ,  $Z_{13\dots}$ ,  $Z_{23\dots}$ ,  $Z_{33\dots}$ ), since symmetry is assumed. Finally, the entire LIS-file is scanned. The calculated values are then presented in result window as shown in Fig. 5.48. The user can switch between polar and complex coordinates and create a text-file of the result. The mutual data are presented on a separate page. The unit of the admittances is given in Farads or Siemens (micro or nano) and the user can scale all values by a factor or by the length.

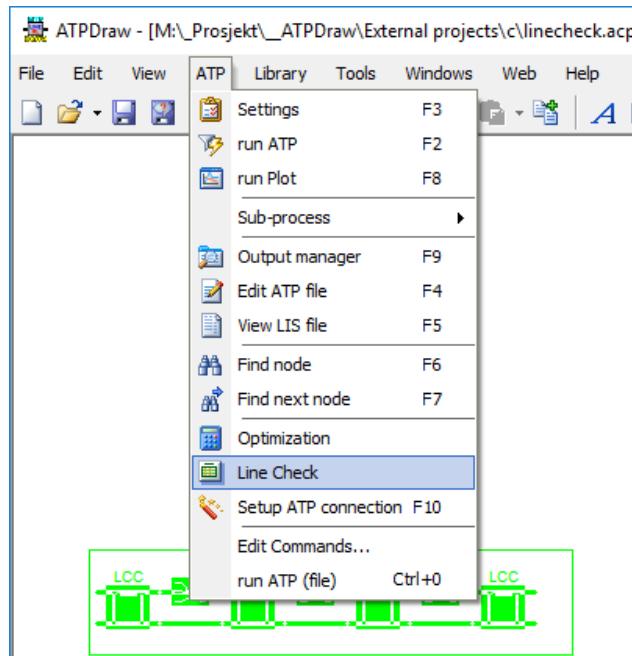


Fig. 5.46 – Select a line/cable sequence

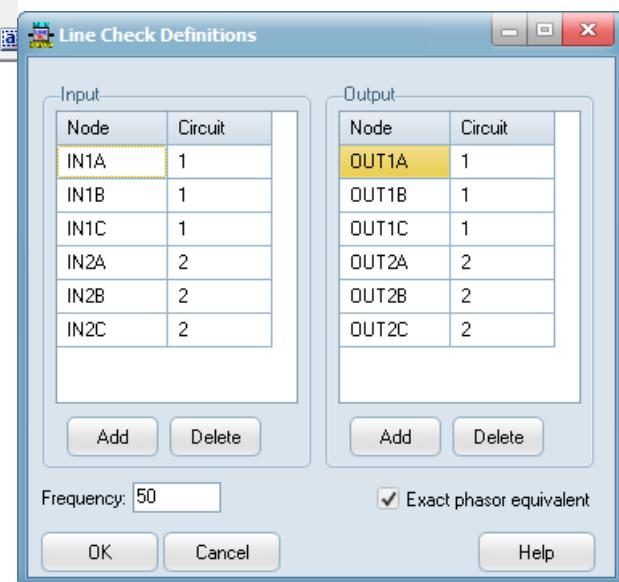


Fig. 5.47 – Specify inputs and outputs

The series impedances are obtained by applying 1 A currents on the terminals and the output ends are grounded (the other circuits are left open and unenergized). For mutual coupling, 1 A is applied at both circuits. On the other hand, the shunt admittances are obtained by applying a voltage source of 1 V at one terminal leaving the output end open. For mutual coupling, 1V is applied at one circuit while a voltage of 1E-20 is applied at the other.

Special attention must be paid to long lines and cables. This applies in particular to PI-equivalents. Usage of Exact phasor equivalent is recommended but is no guarantee of success. No attempt is made in ATPDraw to obtain a better approximation since the line/cable system to be tested in general is unknown. The mutual coupling in the positive sequence system is in symmetrical cases very small and vulnerable to the approximations made.

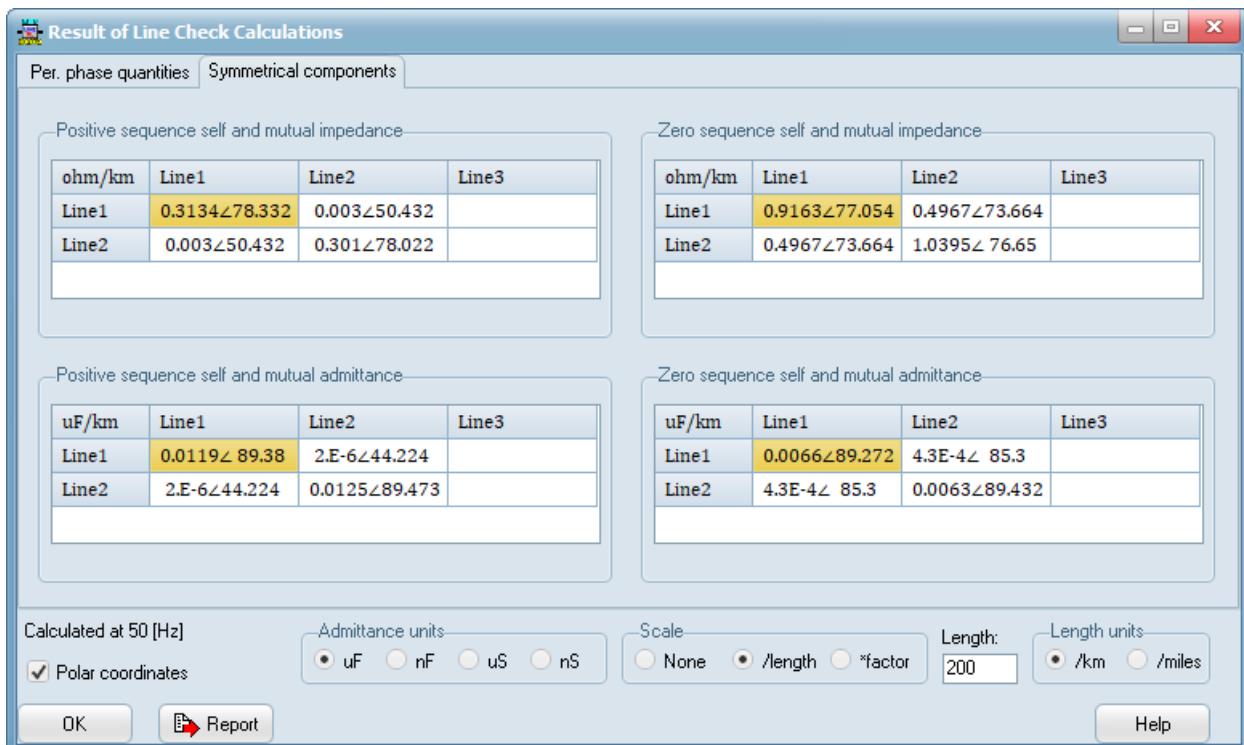


Fig. 5.48 – Presentation of the LineCheck results.

## 5.5 Using MODELS simulation language

MODELS is a general-purpose description language supported by a set of simulation tools for the representation and study of time-variant systems. This chapter of the Manual builds on *MODELS IN ATP -Language Manual*, February 1996 [4]. Please consult this manual for more detailed information related to the MODELS language.

The MODELS language focuses on the description of the structure of a model and on the function of its elements. There is a clear distinction in MODELS between the description of a model, and the use of a model. Individual models can be developed separately, grouped in one or more libraries of models, and used in other models as independent building blocks in the construction of a system. The description of a model is intended to be self-documenting. A system can be described in MODELS as an arrangement of inter-related sub models, independent from one another in their internal description and in their simulation (e.g. individual models can have different simulation time step). Description of each model uses a free-format, keyword-driven syntax of local context, and does not require fixed formatting in its representation.

The main description features of the MODELS language are the following:

- The syntax of MODELS allows the representation of a system according to the system's functional structure, supporting the explicit description of composition, sequence, concurrence, selection, repetition, and replication;
- The description of a model can also be used as the model's documentation;
- The interface of a model with the outside world is clearly specified;
- The components of a model can be given meaningful names representative of their function
- A system can be partitioned into individual sub models, each with a local name space;
- The models and functions used for describing the operation of a system can be constructed in programming languages other than the MODELS language.

ATPDraw supports only a simplified usage of MODELS. In general, ATPDraw takes care of the interface between MODELS and the electrical circuit (INPUT and OUTPUT of the MODELS section) and the execution of each model (USE). There can thus not be any expressions in the USE section. The type of input (current, voltage, output from model, tacs etc.) to a Model is a property of the Model's node. The user must click on the nodes to set or verify correct input type.

Creating a new Model in ATPDraw can follow two approaches:

1. The default approach. Select the *Models/Default model* or open an existing .mod file and let ATPDraw take care the component definitions with icon and node connections. This is the preferred option and the best approach as a model typically will change during the study. The icon and node positions can be edited under *Edit definitions* also here.
2. The manual approach. Select *Models/Files mod/sup* and choose a pre-existing support file (accompanied with a compatible .mod file). This is a relevant choice if the model is fixed during the study and the icon and node locations are crucial.

The new MODELS object created in this chapter is part of the ATPDraw's example file Exa\_14.acp. In this example the harmonic content of the line current on the 132 kV supply side of an industrial plan using a 24 pulse AC/DC converter is calculated by MODELS.

### 5.5.1 The default approach

Add a new Model to your circuit by selecting *MODELS/Default model* from the selection menu. A simple Model will appear with the standard Model dialog shown as shown in Fig. 5.49. Now, click on the *Edit* button and type in your model script, import a text from file with *File/Import* or paste in a text from the Windows clipboard. Anyway, this is the hard part of the process. All INPUT, OUTPUT, DATA and VAR can be indexed. For INPUTS and OUTPUTS there is a maximum upper index limit of 26 (A..Z phase extension of node names). The low index has to be unity. Indexed data is also allowed and these are then split in x[1], x[2] etc. There is no limitation on the number of data. Instead of starting with the Default model, the user could also paste in a Model from the same or another circuit.

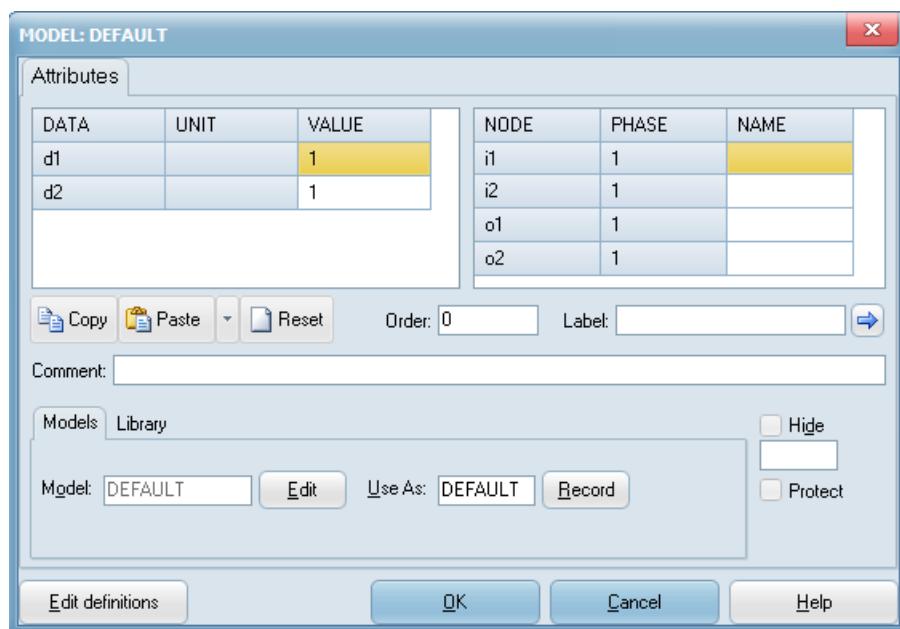


Fig. 5.49 – Component dialog of the Default Model.

Click on *Done* when the edit process is completed. ATPDraw will then examine the Model description and identify the Input/Output/Data declarations. If the number of input or outputs have

changed the icon is recreated. Inputs are positioned on the left side and Outputs on the right side (from top to bottom). A message box then appears as shown in Fig. 5.50. Typically, you should choose not to edit the file, but if you choose Yes the *Edit definitions* dialog appears where you can relocate the nodes and change the icon. This might be a tricky process though. Anyway, you can whenever click on *Edit definitions* and do this job later. If you click on No, you will return to an updated Component dialog box as shown in Fig. 5.51.

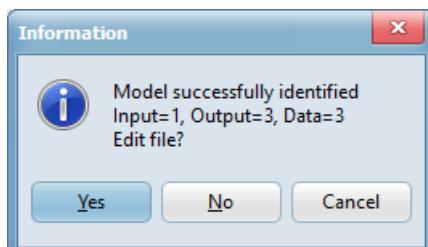


Fig. 5.50 – Identification of the Model text.

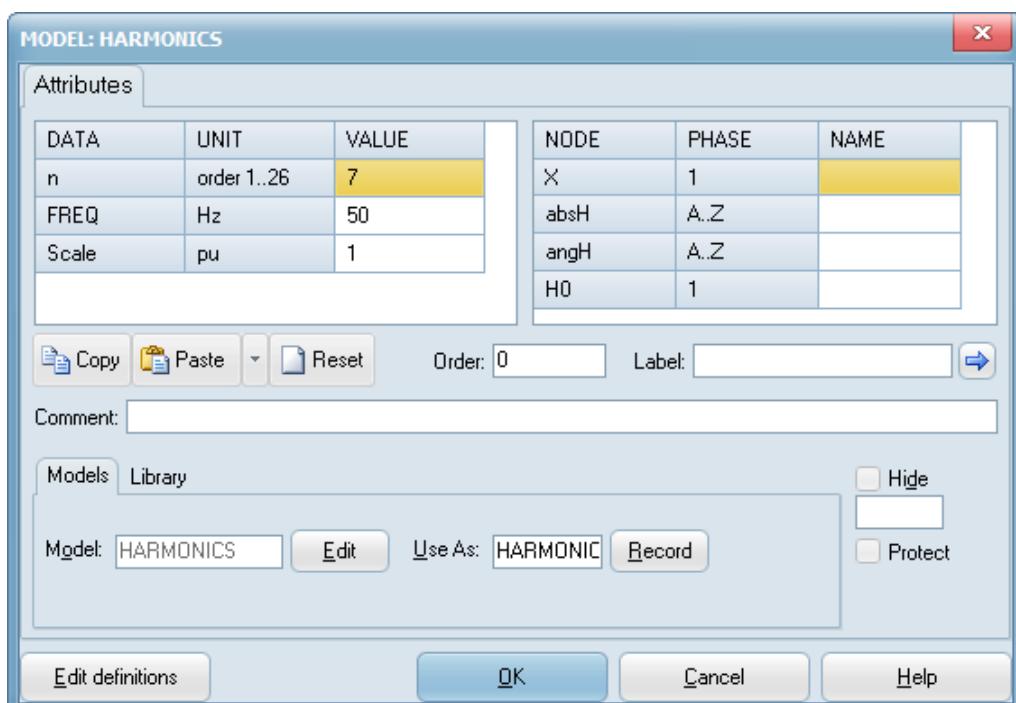


Fig. 5.51 – Component dialog of the FOURIER model.

In the Models section in Fig. 5.51 you must also specify the *Use As* name for USE model AS model\_name statement of MODELS. Record of local variable is also available in this section.

### 5.5.2 The MODELS editor

The Models Editor is built on the commercial TMS-software TadvMemo component. This provides better syntax highlighting features as the notoriously slow native Windows control RichEdit is bypassed. The syntax of the MODELS language is described in ATPDraw as required by the component. The MODELS editor shown in Fig. 5.52 has now a gutter with line numbers and a grey line at column 80 for visualization of maximum ATP line width (ATPDraw will wrap long lines, though).

### 5.5.2.1 Highlighter

The highlighter differentiates between

Comments:	<i>green italic</i>	(--this is a comment)
Keywords:	<b>black bold</b>	( <b>INPUT</b> , <b>EXEC</b> , <b>for</b> , <b>if</b> ...)
Functions:	<b>blue</b>	(cos, recip, sqrt ...)
Numbers:	<b>red</b>	(2, 5.0, 1e3...)
Operators/parenthesis:	<b>purple</b>	(:=, *, +, (, ]...)
Texts:	<b>teal</b>	('BEGIN WRITE @W1RELAY21P')

In ATPDraw v7.0 the user is only allowed to turn on/off highlighting and not edit the appearance.

```

1  MODEL HARMONICS
2  comment-----
3  This model calculates the harmonics up to maximum order 26
4  of a time varying signal X based on a DFT algorithm in a
5  moving window. Scale=1; output of peak quantities.
6  -----
7  INPUT X          --input signal to be transformed
8  DATA  FREQ {DFLT:50}    --power frequency
9      n {DFLT:26}        --number of harmonics to calculate
10     Scale {DFLT:1}       --scaling of the harmonics values
11  OUTPUT absH[1..26], angH[1..26],H0 --Harmonic outputs. H0 is DC comp.
12  VAR   absH[1..26], angH[1..26],H0,reH[1..26], imH[1..26],i,NSAMPL,OMEGA
13      D,F1,F2,F3,F4
14
15  HISTORY
16  X {DFLT:0}
17
18  DELAY CELLS DFLT: 1/ (FREQ*timestep)+2
19
20  INIT
21  OMEGA:= 2*PI*FREQ
22  NSAMPL:=1/ (FREQ*timestep)
23  H0:=0
24  FOR i:=1 to 26 DO
25      -----

```

Fig. 5.52 – MODELS editor in ATPDraw v7.

### 5.5.2.2 Undo/Redo

The undo/redo mechanism is substantially extended by default in the TadvMemo component with an infinite number of undo/redo steps compared to just a single step in ATPDraw v6.

### 5.5.2.3 Indent and code-folding

Auto-indent is to start the new line in the same column as the previous after a carriage return. There are also menu options to indent or unindent a (selected) block of code in steps of two characters.

Code-folding is an option to collapse or open a group of code lines between sections like INIT-ENDINIT, if-endif, for-endfor EXEC-ENDEXEC etc. This can be useful for large MODELS codes. Code-folding can be turned on/off by the user.

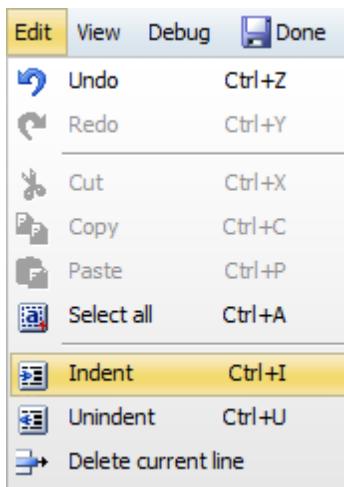


Fig. 5.53 – Edit menu in MODELS editor.

```

20  INIT
21      OMEGA:= 2*PI*FREQ
22      NSAMPL:=1/ (FREQ*timestep)
23      HO:=0
24  FOR i:=1 to 26 DO
25      reH[i]:=0
26      imH[i]:=0
27      absH[i]:=0
28      angH[i]:=0
29  ENDFOR
30  ENDINIT
31  EXEC ...
32  ENDMODEL

```

The code in the MODELS editor is shown with code folding. The 'INIT' block is collapsed, indicated by a minus sign icon before the line number 20. The code within the INIT block is visible. The status bar at the bottom shows '58:9 Modified'.

Fig. 5.54 – Code folding in MODELS editor.

#### 5.5.2.4 Insert menu

The MODELS editor contains a complete context menu (right mouse click) for insertion of MODELS controls and functions.

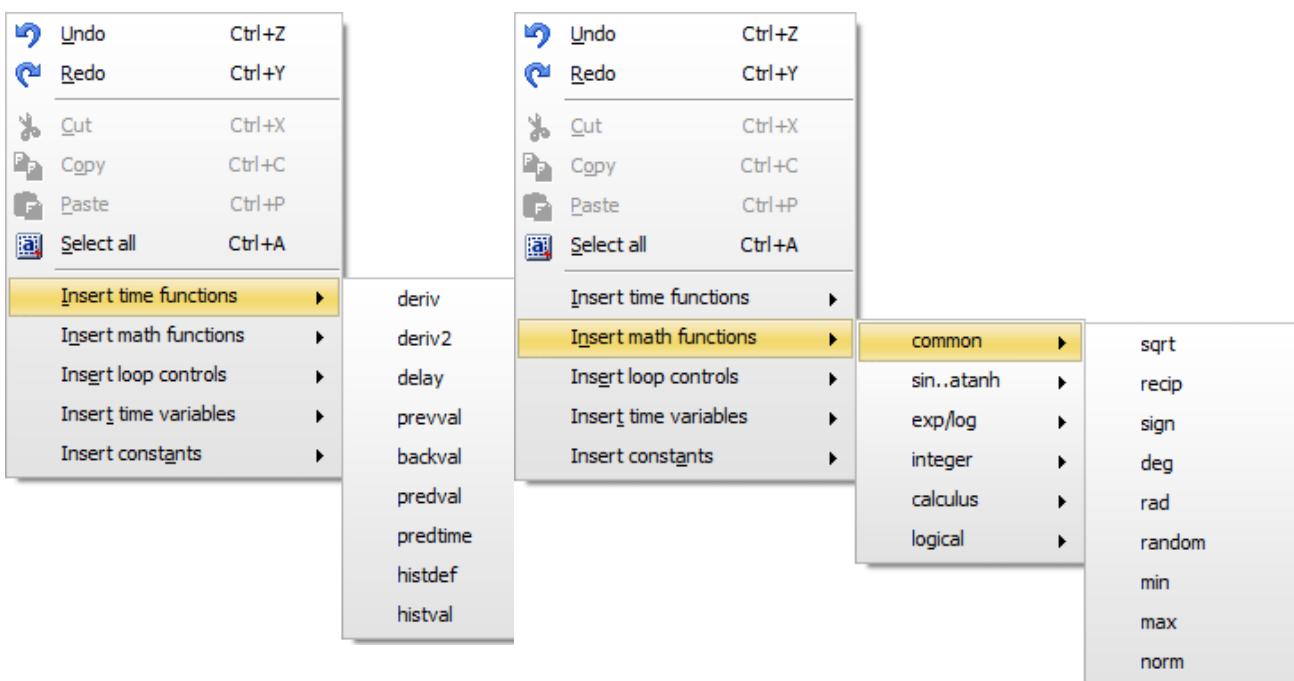


Fig. 5.55 – Insert function examples (context menu, right-click in Editor)

Among the most useful Insert is the Calculus in the Math functions. This contains the special limiter syntax and a reminder about History declaration

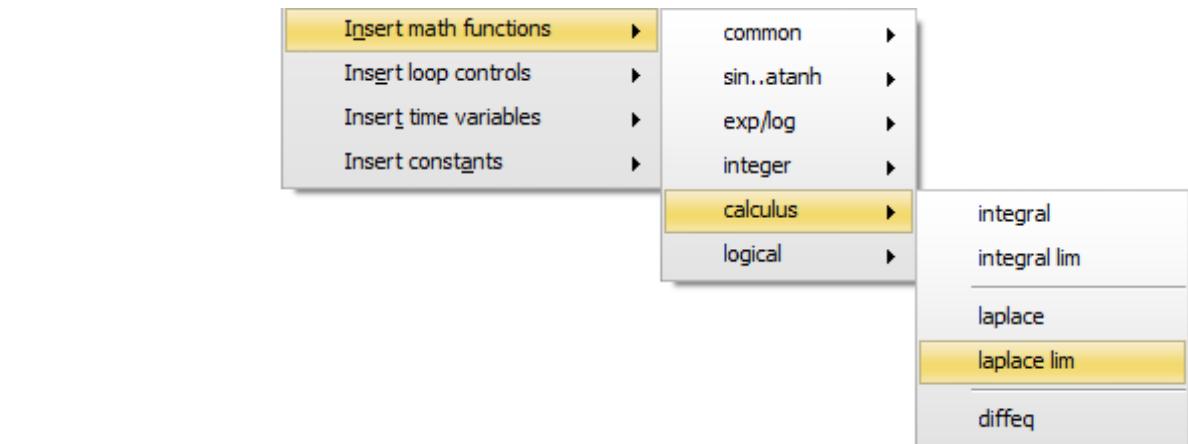


Fig. 5.56 – Insert calculus functions with illustration of limited laplace

### 5.5.2.5 Debugger

ATPDraw does not implement a MODELS parser since this would be a large and very difficult task. Instead the existing debugging feature of ATP is utilized.

The Debugger is built on adding a model called TESTER in the background. This TESTER model acts as a source with number of outputs corresponding to the number of inputs of the model being tested. In the *Debug/Test* setup, each of the inputs and data can be controlled by the user, while under *Debug/Syntax* default values are used.

A test case with the name of the model plus ‘\_debug.dat’ is created in the *ResultDir* folder and run in ATP. ATP will then in the LIS-file report error messages that usually can be identified by ATPDraw. For syntax errors, ATP will write a string of compact interpreted code and stop just prior to an error. ATPDraw will compare this code with its own compacted version of the tested model to identify the (for now) faulty line number. Only the first syntax error is detected.

Difficulties in syntax error checking is that ATP does not report the exact error but only part of the correct code *in front of* the error. It is particularly difficult to detect missing end parenthesis or end of sections like **endfor**, **endif** etc. properly. In models with repetitive sections, the debug information provided by ATP does not uniquely define the error and misinterpretation is theoretically possible.

### 5.5.2.6 Debug example

This example shows how to utilize the Debug feature in the new Models Editor. A model used for a voltage source inverter control is used as an example. The header of the model is shown in the listing below. The full model is shown in appendix. It has 1 input and 3 data. The TESTER model will thus have 1 output.

```
MODEL HARMONICS
comment-----
This model calculates the harmonics up to maximum order 26
of a time varying signal X based on a DFT algorithm in a
moving window. Scale=1; output of peak quantities.
-----endcomment
INPUT X           --input signal to be transformed
DATA FREQ {DFLT:50} --power frequency
```

```

n {DFLT:26}           --number of harmonics to calculate
Scale {DFLT:1}          --scaling of the harmonics values
OUTPUT absH[1..26], angH[1..26], H0 --Harmonic outputs. H0 is DC comp.
VAR     absH[1..26], angH[1..26], H0, reH[1..26], imH[1..26], i, NSAMPL, OMEGA
        D, F1, F2, F3, F4

HISTORY
X {DFLT:0}

```

### 5.5.2.6.1 Debug|Syntax

The TESTER model will consist of 1 output cosine signal, with amplitude of unity and a phase displacement of  $360/1=360$  degrees. The entire TESTER model is shown in the listing below. The data of the model will be set to the values specified, otherwise default values are used (zero if nothing is specified and the model is not initialized).

```

MODEL TESTER
OUTPUT out[1..1]
VAR out[1..1]
EXEC
for i:=1 to 1 do
    out[1]:=1*cos(2*pi*(t*50+0/360))
endfor
ENDEXEC
ENDMODEL

```

Fig. 5.57 shows the case where a syntax error is introduced by using a wrong parenthesis ‘(‘ instead of ‘[‘, and how the debugger responds to this by underlining line 28 and displaying ‘Syntax error found in line 28’ in the footer bar.

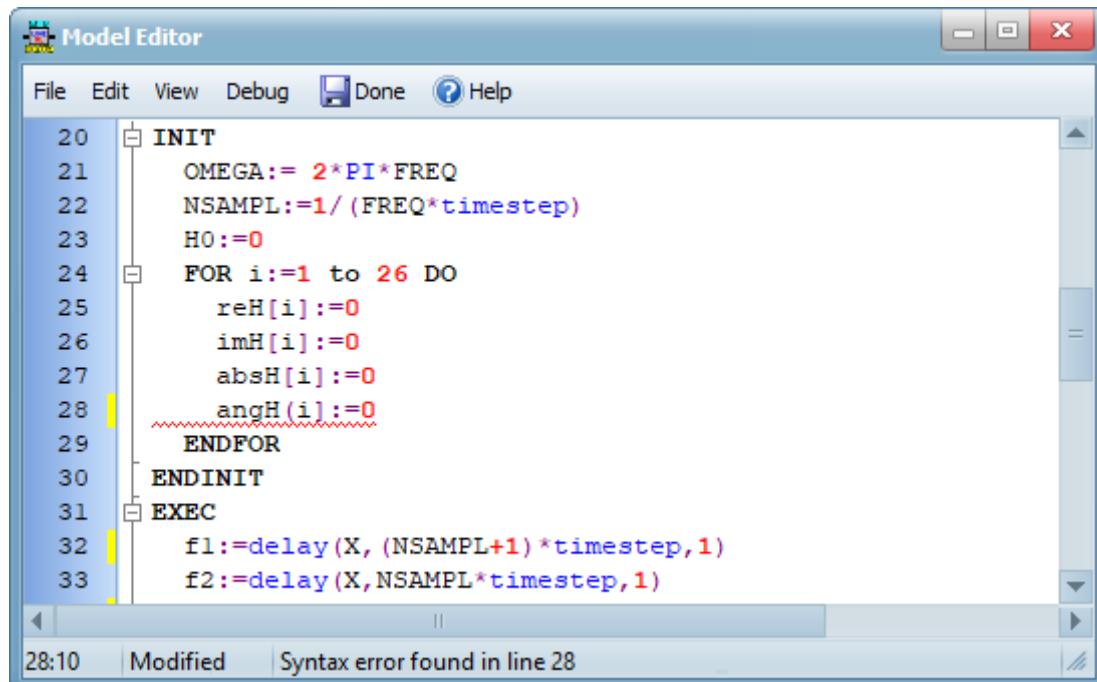


Fig. 5.57 – Debugger indicate line with syntax error.

### 5.5.2.6.2 Debug|Test

In *Debug/Test* all inputs and data of the models are automatically listed as shown in Fig. 5.58 . For Inputs the user can specify a cosine function with amplitude, frequency and phase angle. For Data the user can specify a constant. Default values equal to what is used under *Debug/Syntax* are initially suggested.

After specifying the Inputs and Data the user should click on *Run*. This will display possible error messages from the LIS-file in the field below (not only syntax errors, but also undefined variables or functions). Clicking *Plot* will open the plotting program with the generated pl4 file containing all outputs. Clicking *Show syntax error* will go back to the Models Editor and underline the line with an error.

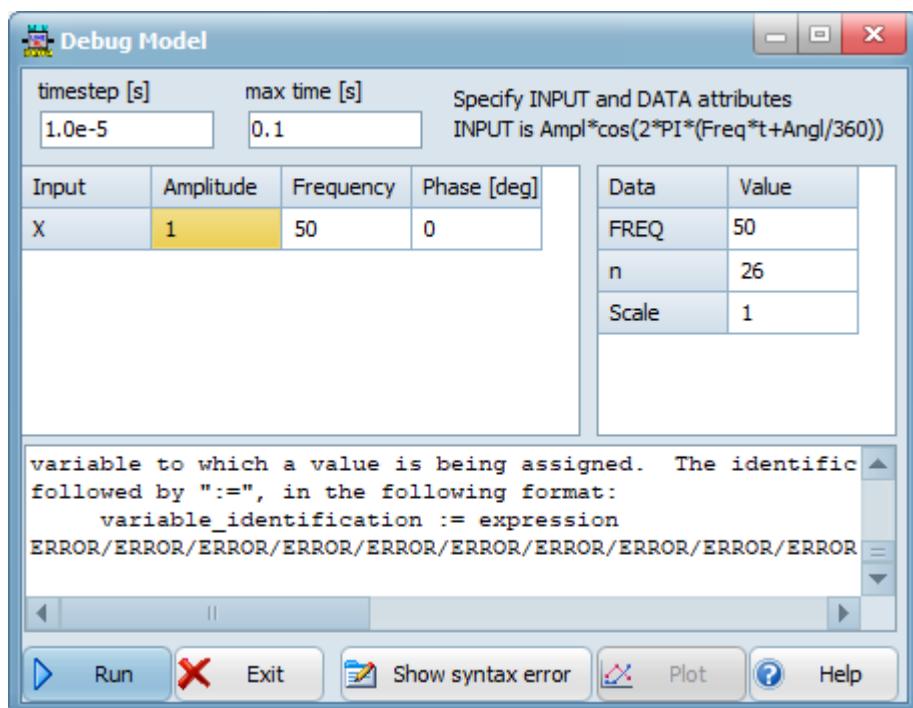


Fig. 5.58 – Model Debug/Test dialog. Specification of inputs and data.

The actual model file describing the calculation of harmonics is shown below:

```

MODEL HARMONICS
comment-----
This model calculates the harmonics up to maximum order 26
of a time varying signal X based on a DFT algorithm in a
moving window. Scale=1; output of peak quantities.
-----endcomment

INPUT X                                --input signal to be transformed
DATA FREQ {DFLT:50}                     --power frequency
      n {DFLT:26}                         --number of harmonics to calculate
      Scale {DFLT:1}                        --scaling of the harmonics values
OUTPUT absH[1..26], angH[1..26], H0 --Harmonic outputs. H0 is DC comp.
VAR    absH[1..26], angH[1..26], H0, reH[1..26], imH[1..26], i, NSAMPL, OMEGA
        D, F1, F2, F3, F4
HISTORY X {DFLT:0}
DELAY CELLS DFLT: 1/(FREQ*timestep)+2
INIT
  OMEGA:= 2*PI*FREQ
  NSAMPL:=1/(FREQ*timestep)
  H0 :=0

```

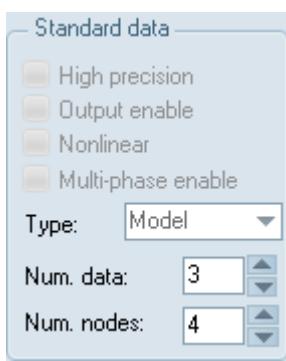
```

FOR i:=1 to 26 DO
  reH[i]:=0
  imH[i]:=0
  absH[i]:=0
  angH[i]:=0
ENDFOR
ENDINIT
EXEC
f1:=delay(X, (NSAMPL+1)*timestep,1)
f2:=delay(X, NSAMPL*timestep,1)
f3:=delay(X, timestep,1)
f4:=X
H0:=H0+(f4+f3-f2-f1)/(2*NSAMPL)
FOR i:=1 to n DO
  D:=1/(i*PI)*((f4-f2)*sin(i*OMEGA*T)-(f3-f1)*sin(i*OMEGA*(T-timestep)))
  + (f4-f3-f2+f1)/(timestep*i*OMEGA)*
    (cos(i*OMEGA*T)-cos(i*OMEGA*(T-timestep))))
  reH[i]:=reH[i]+D
  D:=1/(i*PI)*((f4-f2)*cos(i*OMEGA*T)-(f3-f1)*cos(i*OMEGA*(T-timestep)))
  - (f4-f3-f2+f1)/(timestep*i*OMEGA)*
    (sin(i*OMEGA*T)-sin(i*OMEGA*(T-timestep))))
  imH[i]:=imH[i]+D
  absH[i]:=sqrt(reH[i]**2+imH[i]**2)*Scale
  IF abs(imH[i])<1E-10
  THEN
    angH[i]:=0
  ELSE
    angH[i]:=atan2(imH[i],reH[i])
  ENDIF
ENDFOR
ENDEXEC
ENDMODEL

```

### 5.5.3 The manual approach

You can create an external support file in two ways. Either by click on *Edit definitions* is the *Component dialog* of your Model and then click on *Save As* (preferable to the /MOD directory). This will simply give you a copy of your Model component. The other way is to go via *Library/New object/Model sup-file* and create a support file from scratch. Both these options use the *Edit definitions dialog*. The result is a support file that you load via *MODELS/Files (sup/mod)*.



The manual approach requires that you have the mod file finished, or at least you need to know the number and name of all input, outputs and data. Enter the *Library* menu and select the *New objects/Model sup-file*. This menu item will perform the *Edit definitions dialog*. In the *Standard data* field, you specify the size of the model: number of nodes and number of data as shown in Fig. 5.59.

The *FOURIER.MOD* text has four nodes (1 input + 3 outputs) and two data, (FREQ, n), so you must enter 4 and 2 in the *Num.* fields.

Fig. 5.59 - Specify the size of the model.

After you have specified the node and data values go to the tabbed notebook style part of the dialog box. Select the *Data* page where you specify the values shown in Fig. 5.60. The *Name* of the data must be the same as those used in the DATA declaration part of the .mod file. The

*Default* value appears initially in the models dialog. The default values are taken from the Use Model statements in DC68.DAT (you can of course change these values individually for each use of the model). *Min* and *Max* restrict the legal input range. No restriction is applied here to data values, so *Min*=*Max*. *Param* is set to 1, which means that variable text string can be assigned to the data value. *Digits* is the maximum number of digits allowed in the ATP input file.

Name	Default	Units	Min	Max	Param	Page	Digits
FREQ	50		0	0	1	0	8
n	26		0	0	1	0	8
Scale	1		0	0	1	0	8

Fig. 5.60 - Specify Data parameters.

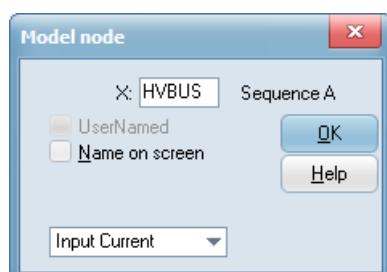
After you have specified the data values click on the *Nodes* tab to enter to the node window as shown in Fig. 5.61. The *Name* identifies the node in the Node and Component input dialogs. The name you enter here must be the same as those used in the INPUT and OUTPUT declaration sections of the .mod file. The *Position* field is the node position on the icon border as shown at the right (Alt+F1..F12 are short keys), but other positions (-120..120) is possible. The *Kind* value specifies the input/output type of the node. Number of #Phases must be set to match the array size of the input/outputs.

Name	Kind	#Phases	Pos.x	Pos.y	Internal
X	2	1	-20	-10	0
absH	0	26	-20	10	0
angH	0	26	20	-10	0
H0	0	1	20	10	0

Fig. 5.61 - Specifying Node attributes.

Supported *Kind* values for MODELS objects are:

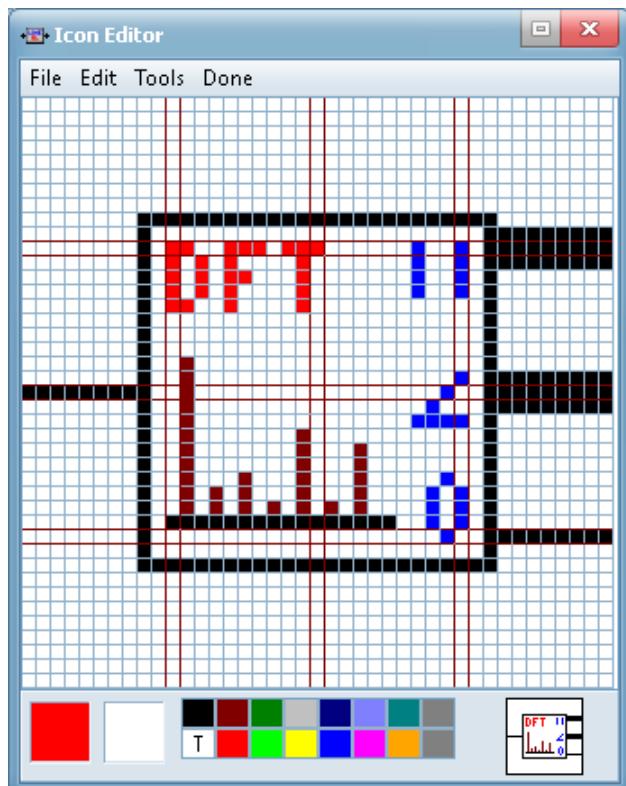
- |    |  |    |                              |
|----|--|----|------------------------------|
| 0: | Output node.   | 3: | Switch status input node.    |
| 1: | Current input node.                                    | 4: | Machine variable input node. |
| 2: | Voltage input node.                                    | 5: | TACS variable (tacs).        |
| 6: | Imaginary part of steady-state node voltage (imssv).   |    |                              |
| 7: | Imaginary part of steady-state switch current (imssi). |    |                              |
| 8: | Output from other model.                               | 9: | Global ATP variable.         |



The *Kind* parameter of model object nodes can be changed later in the Node dialog box (input field *Type*), as shown in Fig. 5.62. This window appears when the user clicks on a Model node with the right mouse button.

Fig. 5.62 - Model node dialog box.

**Note!** If a model output is used as input to another model, the outputting model must be USED in the ATP file before the receiving model. This can be done by selecting *Edit/Arrange/Sort all Models*, sorting manually in the *Sidebar/Project/Object tree*, or by specifying a lower *Order* number for the outputting model and selecting the *Sorting by Order* option under *ATP / Settings / Format* (or *Sidebar/Simulation*).



Model objects also have an icon, which represents the object on the screen and an optional help, which describes the meaning of parameters. If no user supplied help text was given, the *Help Viewer* displays the model definition file (.mod) automatically. If you really need a help text, this feature can be overridden by opening the *Help Editor* with the  button at the right-hand side of the dialog box.

The *Icon Editor* appears similarly, by clicking  on the  button. In this case Bitmap icon style is chosen. Here you can be creative and draw a suitable icon for the new model object. When you finished select the *Done* menu item.

Fig. 5.63 - The icon of the new model objects.

The *Save* or *Save As* buttons can be used to save the new support file to disk. Default location of Model support files is the \MOD folder. The .sup file does not need to have the same name as the model file, but it is recommended.

The new model object has now been created is ready for use. You can reload and modify the support file of the model objects whenever you like.

Selecting *MODELS | Files (sup/mod)...* in the component selection menu performs an *Open Model* dialog box where you can choose a model support file. If you select the file FOURIER.SUP the icon of the new model appears immediately in the circuit window and it can be connected with other object in normal way.

The input and output interface for MODELS objects, the use of the model and interfacing it with the rest of the circuit are handled automatically by ATPDraw. The model description is written directly in the ATP input file. Blank lines are removed when inserting the .mod file. The general structure of the MODELS section in an .atp input file is shown below:

```

MODELS
INPUT
M0001A {i(HVBUSA) }
OUTPUT
X0027A
X0027B
...
X0027Z
X0028A

```

```

X0028B
...
X0028Z
XX0029
-----
MODEL FOURIER
...
Description of the model.
Complete copy of the
FOURIER.MOD is pasted here.
...
ENDMODEL
-----
USE FOURIER AS FOURIER
INPUT
  X:= M0001A
DATA
  FREQ:=      50.
  N:=          26.
OUTPUT
  X0027A:=ABSF[1]
  X0027B:=ABSF[2]
  ...
  X0027Z:=ABSF[26]
  X0028A:=ANGF[1]
  X0028B:=ANGF[2]
  ...
  X0028Z:=ANGF[26]
  XX0029:=F0
ENDUSE

```

#### 5.5.4 Recording internal MODELS variables

ATPDraw supports the RECORD feature of MODELS to record any internal variable of a model object in the .pl4 output. The selection of internal variables is done by clicking the *Record* button in Fig. 5.51. This will bring up the Record dialog shown in Fig. 5.64. The available variables (VAR+OUTPUT) is shown in the list to the left. Select the desired variable and click the >> button. The *Record* field to the right is a free format text field that allows you to easily edit the AS name. In the case of indexed variables you also need to specify the index as well (shown as reH[5]). Remove the variable from the *Record* list by the << button. The Outputs from a Model can alternatively be recorded with the Model Probe as shown to the right in Fig. 5.64.

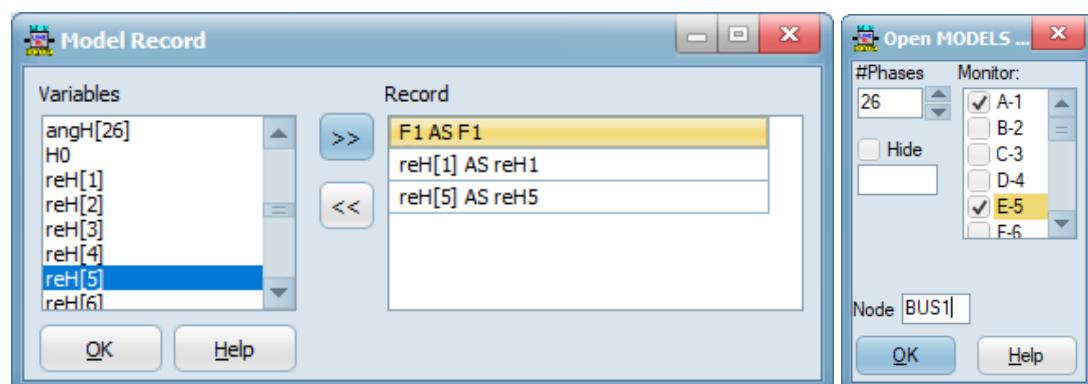


Fig. 5.64 - Record of model variables. Right: Models Probe connected to Output node.

## 5.6 BCTRAN support in ATPDraw

ATPDraw provides a user-friendly interface for the BCTRAN transformer matrix modeling, to represent single and three-phase, two and three winding transformers. After the user has entered the open circuit and short circuit factory test data, the ATPDraw calls ATP and executes a BCTRAN supporting routine. Finally, ATPDraw includes the punch-file into the ATP-file. The windings can be Y, D or Auto coupled with support of all possible phase shifts. The nonlinear magnetization branch can optionally be added externally.

Fig. 5.65 shows the *BCTRAN* dialog box, which appears when the user selects *BCTRAN* under *Transformers* of the component selection menu. Under *Structure*, the user specifies the number of phases, the number of windings, the type of core (not supported yet, except for single phase cores, triplex and three-phase shell type), and the test frequency. The dialog box format adapts the number of windings and phases. The user can also request the inverse L matrix as output by checking *AR output*. An *Auto-add nonlinearities* button appears when an external magnetizing branch is requested.

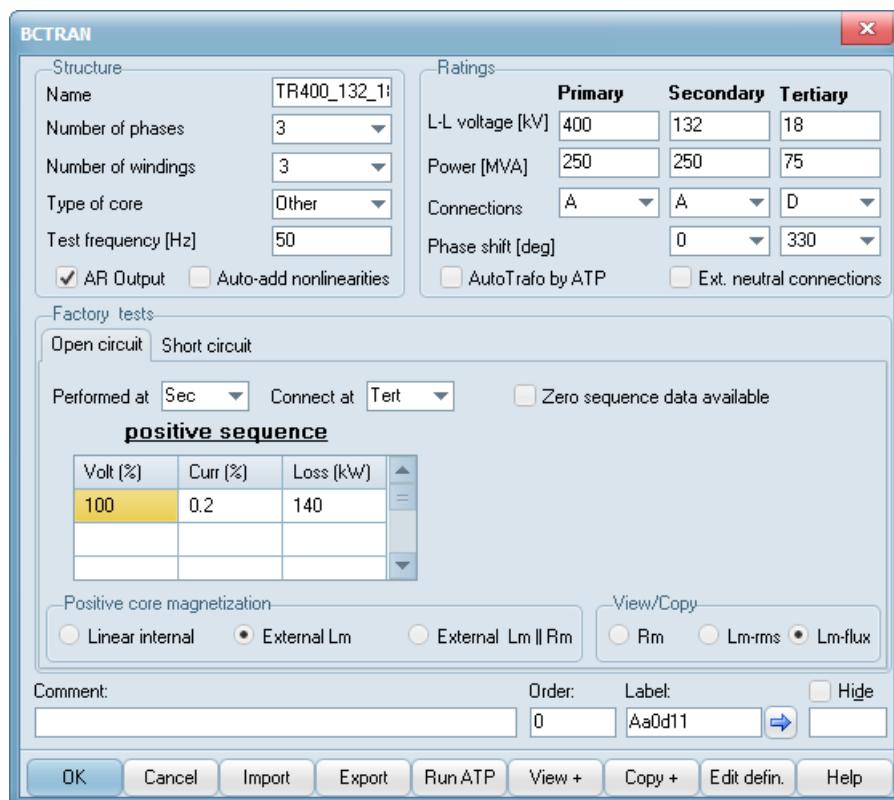


Fig. 5.65 - The BCTRAN dialog box.

Under *Ratings* the line-voltage, rated power, and type of coupling must be specified. Supported winding *Connections* are: A (auto-transformer), Y (wye) and D (delta). The *Phase shift* menu adapts these settings with all types of phase shifts supported. If the connection is A or Y, the rated voltage is automatically divided by  $\sqrt{3}$  to get the winding voltage VRAT. *Auto Trafo by ATP* should be checked in case of Auto-transformers using a new (>2015) ATP version, as ATP was at some point modified to auto-correct auto-transformer test reports.

Under *Factory tests*, the user can choose either the *Open circuit* test or the *Short circuit* test. Under the *Open circuit* tab the user can specify where the factory test has been performed and where to connect the excitation branch. In case of a three winding transformer one can choose

between the HV, LV, and the TV winding. Normally the lowest voltage is preferred, but stability problems for delta-connected nonlinear inductances could require the lowest Y-connected winding to be used. Up to 6 points on the magnetizing curve can be specified. The excitation voltage and current must be specified in % and the losses in kW. With reference to the ATP Rule Book, the values at 100 % voltage is used directly as `IEXPOS=Curr [%]` and `LEXPOS=Loss [kW]`. One exception is if *External Lm* is chosen under *Positive core magnetization*. In this case only the resistive current is specified resulting in  $IEXPOS=Loss/(10 \cdot SPOS)$ , where `SPOS` is the *Power [MVA]* value specified under *Ratings* of the winding where the test has been performed. If zero-sequence open circuit test data are also available, the user can similarly specify them to the right. The values for other voltages than 100 % can be used to define a nonlinear magnetizing inductance/resistance. This is set under *Positive core magnetization*:

- Specifying *Linear internal* will result in a linear core representation based on the 100 % voltage values.
- Specifying *External Lm//Rm* the magnetizing branch will be omitted in the BCTRAN calculation and the program assumes that the user will add these components as external objects to the model.
- Specifying *External Lm* will result in calculation of a nonlinear magnetizing inductance first as an  $I_{rms}$ - $U_{rms}$  characteristic, then automatically transformed to a current-fluxlinked characteristic (by means of an internal SATURA-like routine). The current in the magnetizing inductance is calculated as

$$I_{rms}[A] = \sqrt{(10 \cdot Curr[\%] \cdot SPOS[MVA]/3)^2 - (Loss[kW]/3)^2} / V_{ref}[kV]$$

where  $V_{ref}$  is actual rated voltage specified under *Ratings*, divided by  $\sqrt{3}$  for Y- and Auto-connected transformers.

The user can choose to *Auto-add nonlinearities* under *Structure* and in this case the magnetizing inductance is automatically added to the final ATP-file as a Type-98 inductance. ATPDraw connects the inductances in Y or D dependent on the selected connection for actual winding for a 3-phase transformer. In this case, the user has no control on the initial state of the inductor(s). If more control is needed (for instance to calculate the fluxlinked or set initial conditions) *Auto-add nonlinearities* should not be checked. The user is free to create separate nonlinear inductances, however. The *Copy+* button at the bottom of the dialog box allows the user to copy the calculated nonlinear characteristic to an external nonlinearity. What to copy is selected under *View/Copy*. To copy the fluxlinked-current characteristic used in Type-93 and Type-98 inductances *Lm-flux* should be selected.

	Open circuit	Short circuit	
	<b>positive sequence</b>		
	Imp. (%)	Pow. (MVA)	Loss (kW)
Prim-Se	15	250	710
Prim-Te	41.6667	250	188
Sec-Te	24	250	159

The *Short circuit* data can be specified as shown in Fig. 5.66. With reference to the ATP Rule Book; *Imp [%]* is equal to `ZPOS`, *Pow. [MVA]* is equal to `SPOS`, and *Loss [kW]* is equal to `P`. These three values are specified for all the windings. If zero-sequence short circuit factory test data are also available, the user can similarly specify them to the right of the positive sequence values after selecting the *Zero sequence data available* check box.

Fig. 5.66 - Short circuit factory test data.

If Auto-transformer is selected for the primary and secondary winding (HV-LV) the impedances must be re-calculated according to Eq. 6.45, 6.46, 6.50 of the EMTP Theory Book [5]. This task is performed by ATPDraw and the values  $Z_{H-L}^*$ ,  $Z_{L-T}^*$ , and  $Z_{H-T}^*$  are written to the BCTRAN-file automatically.

$$z_{H-L}^* = z_{H-L} \left( \frac{V_H}{V_H - V_L} \right)^2, \quad z_{L-T}^* = z_{L-T}, \quad z_{H-T}^* = z_{H-L} \frac{V_H \cdot V_L}{(V_H - V_L)^2} + z_{H-T} \frac{V_H}{V_H - V_L} - z_{L-T} \frac{V_L}{V_H - V_L}$$

where  $Z_{L-H}$ ,  $Z_{L-T}$ , and  $Z_{H-T}$  are the short-circuit impedances *Imp. [%]* referenced to a common *Pow.[MVA]* base. ATP was at some point modified to auto-correct auto-transformer test reports, and in this case *Auto Trafo* by ATP should be checked, to bypass the ATPDraw's correction.

When the user clicks on *OK* the data is stored in memory with the project. Then the user is offered to generate a BCTRAN model via execution of ATP. This is optional, since often a new BCTRAN model will be required anyway during the final ATP-file generation. Trying to run ATP is a good practice however, since this will quickly warn the user about possible problems. The button *Run ATP* requests an ATP execution without leaving the dialog box. If the BCTRAN-file is correct, a punch-file will be created. This file is directly included in the final ATP-file and there is no conversion to a library file as for lines/cables. This means in practice that a new BCTRAN model will be created and ATP executed automatically (when creating the final ATP-file) each time the transformer's node names change.

If the user clicks *Export* the data is stored in a binary disk file with extension *.bct* preferably in the */BCT* folder. There is also an *Import* button available to import existing BCT-files. The user can also store the BCT-file with a different name (*Save As*), which is useful when copying BCTRAN-objects. The *View+* and *Copy+* buttons are for the nonlinear characteristic. *Copy+* transfers the selected characteristic to the Windows clipboard in text format with 16 characters fixed columns (the first column is the current). *View+* displays the nonlinear characteristic in a standard *View Nonlin* window. The *Help* button at the lower right corner of the dialog box displays the help file associated with the BCTRAN object. This help text briefly describes the meaning of input data values.

#### 1. Excitation test data

Specified under *Factory test/Open circuit*.

The data required by BCTRAN are:

FREQ = Test frequency under *Structure*  
IEXPOS = Curr for the 100% voltage value in *Open circuit*, Positive sequence.  
= Loss for the 100% voltage value divided by 10\*SPOS when External Lm requested.  
SPOS = Power under *Ratings* for winding specified under *Performed at*.  
LEXPOS = Loss for the 100% voltage value in *Open circuit*, Positive sequence.  
IEXZERO= Curr for the 100% voltage value in *Open circuit*, Zero sequence.  
SZERO = Power under *Ratings* for winding specified under *Performed at*.  
LEXZERO= Loss for the 100% voltage value in *Open circuit*, Zero sequence.

The above input values can be derived from the factory test data as shown next:

```
IEXPOS= Iex*V*100/SPOS for single phase,
IEXPOS= Iex*√3*V*100/SPOS for 3-phase
  where Iex [kA] = excitation current,
        V [kV] = excitation voltage.
  SPOS[MVA]= power base
IEXZERO= 0 for single phase
IEXZERO= 1/3*Iexh*√3*V*100/SZERO for 3-phase
  where Iexh [kA]= zero-sequence excitation current,
        SPOS[MVA]= power base (normally equal to SPOS)
  Y-connected windings (typical values):
    3-leg core type: IEXZERO= IEXPOS
    5-leg core type: IEXZERO= 4*IEXPOS
```

#### 2. Winding cards

Specified under *Ratings*. The data required by BCTRAN are:

VRAT = L-L voltage [kV] for D-connection or single phase transformers  
 L-L voltage [kV] divided by  $\sqrt{3}$  for A (Auto) and Y connections.  
 3-phase only.  
 BUS1- = The present node names of the transformer component in ATPDraw  
 BUS6 taking the connection and Phase shift [deg] into account.  
 Renaming the nodes will require a new BCTRAN execution performed  
 automatically upon ATP|Run ATP or Make File.

### 3. Short circuit test data

Specified under *Factory test / Short circuit*. The data required by BCTRAN are:

$P_{ij}$  = Loss (kW) under *Short circuit*, Positive sequence  
 $Z_{POSij}$  = Imp (%) under *Short circuit*, Positive sequence  
 $S_{POS}$  = Pow (MVA) under *Short circuit*, Positive sequence  
 $Z_{ZEROij}$  = Imp (%) under *Short circuit*, Zero sequence  
 $S_{ZERO}$  = Pow (MVA) under *Short circuit*, Zero sequence

The short circuit input data can be derived from the factory test reports, as shown next:

$Z_{POSij} = U_{si}/I_{si} * S_{POS}/V_{ri}^2 * 100$  for single phase,  
 $Z_{POSij} = U_{sh}/\sqrt{3} * I_{sh} * S_{POS}/(V_{ri}^2) * 100$  for 3-phase  
 where  
      $U_{si}$  [kV] = short-circuit voltage at winding i  
      $I_{si}$  [kA] = nominal current at winding i  
      $S_{POS}$  [MVA] = power base  
      $V_{ri}$  [kV] = rated line voltage at winding i  
 $Z_{ZEROij} = 0$  for single phase  
 $Z_{ZEROij} = U_{sh}/I_{sh} * S_{ZERO}/(V_{ri}^2) * 300$  for 3-phase  
 where  
      $S_{ZERO}$  [MVA] = power base  
     Zero-sequence tests must be performed with open Delta-windings.

The BCTRAN component is found under *Transformers BCTRAN* in the component selection menu and it can be edited and connected to the main circuit as any other component.

 The data specified in Fig. 5.65 will result in an icon at left with 3 three-phase terminals and one single-phase neutral point common to the primary and secondary autotransformer windings. The label shows the transformer connection.

## 5.7 Hybrid Transformer, XFMR

This component called XFMR was first added to version 4.2 of ATPDraw in June 2005. The model is further improved in several steps by extensive debugging. The XFMR component is an implementation and extension of the work performed by Prof. Bruce Mork at Michigan Tech and his co-workers Francisco Gonzalez-Molina and Dmitry Ishchenko. This project called "Parameter Estimation and Advanced Transformer Models for EMTP Simulations" was sponsored by Bonneville Power Administration. A series of report documents this work and is here used as references MTU4, MTU6 and MTU7. The implementation in ATPDraw was also funded by BPA.

### 5.7.1 Overview

The principle of the modeling is to derive a topologically correct model with the core connected to an artificial winding on the core surface. Individual magnetizing branches are established for the yokes and legs dependent on their relative length and area (normally a value within limited range). A key feature is that magnetization is assumed to follow the Frolich equation which is fitted to Test Report data (using the Gradient Method optimization). This improves extreme saturation behavior since linear extrapolation above the Test Report data is avoided. The leakage inductance

is modeled with an inverse inductance matrix (A-matrix), following the BCTRAN approach as documented in the Theory Book p. 6.21. Shunt capacitances and frequency dependent winding resistance is also considered.

The transformer model consists of four parts (as shown in Fig. 5.67) :

- Inductance. Leakage reactance -> A-matrix
- Resistance. Winding resistance -> R(f)
- Capacitance. Shunt capacitance - C-matrix
- Core. Individual magnetization and losses for legs and yokes.

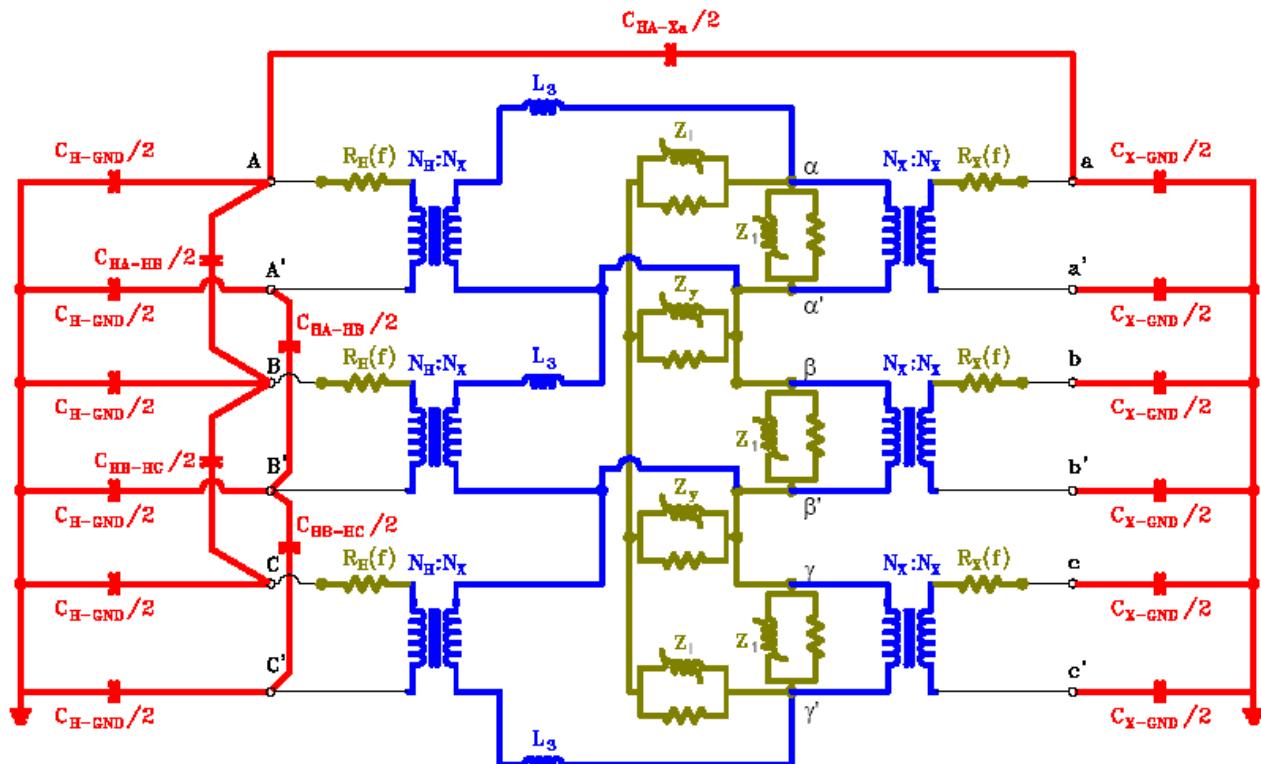


Fig. 5.67 - Duality model for a 3-phase, two-winding transformer from MTU4.

The XFMR component support three sources of data:

- Design parameters. Winding and core geometry and material properties.
- Test report. Standard Test Report data like in BCTRAN. Capacitances and frequency dependent resistance added.
- Typical values. Typical text book values based on transformer ratings. Be careful with this as both design and material properties have changed a lot the last decades.

The overall node structure of the XFMR component in the final ATP file is shown in Fig. 5.68.

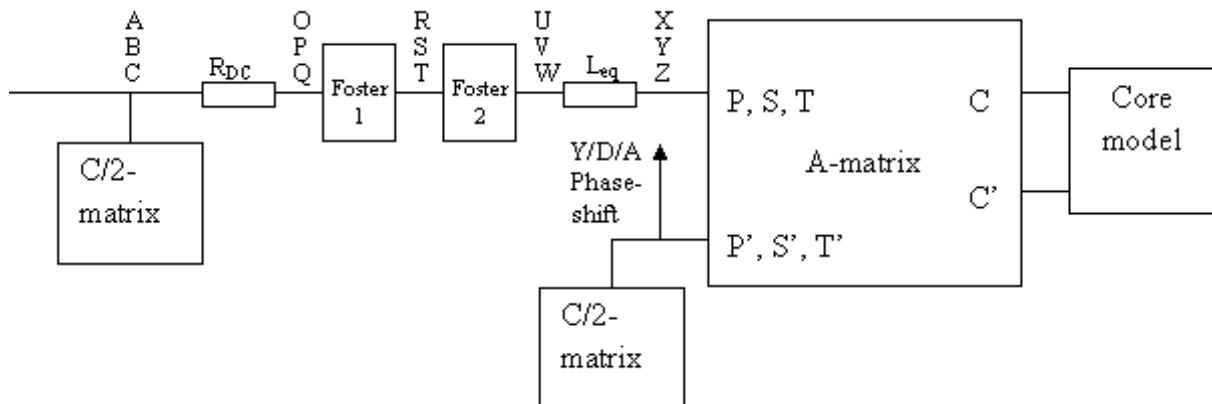


Fig. 5.68 - Node structure in the ATP-file.

This component can be connected as any other component in the circuit with the following exceptions. In both these cases switches should be used in order to maintain unique node names.

- It is not legal to ground nodes directly
- It is not correct to connect several components to the same bus.

### 5.7.2 XFMR dialog box

The advance Hybrid Transformer component, XFMR, is found under *Transformers* in the selection menu. The model support 3-phase transformers with 2 to 4 windings coupled as Wye, Delta, Auto, or Zigzag. All possible phase shifts are supported. Triplex (single phase bank), 3- and 5-legged stacked cores and shell form cores are supported. The dialog box is shown in Fig. 5.69.

All the input fields in the dialog box change dynamically with the user's selection of the number of windings and type of core.

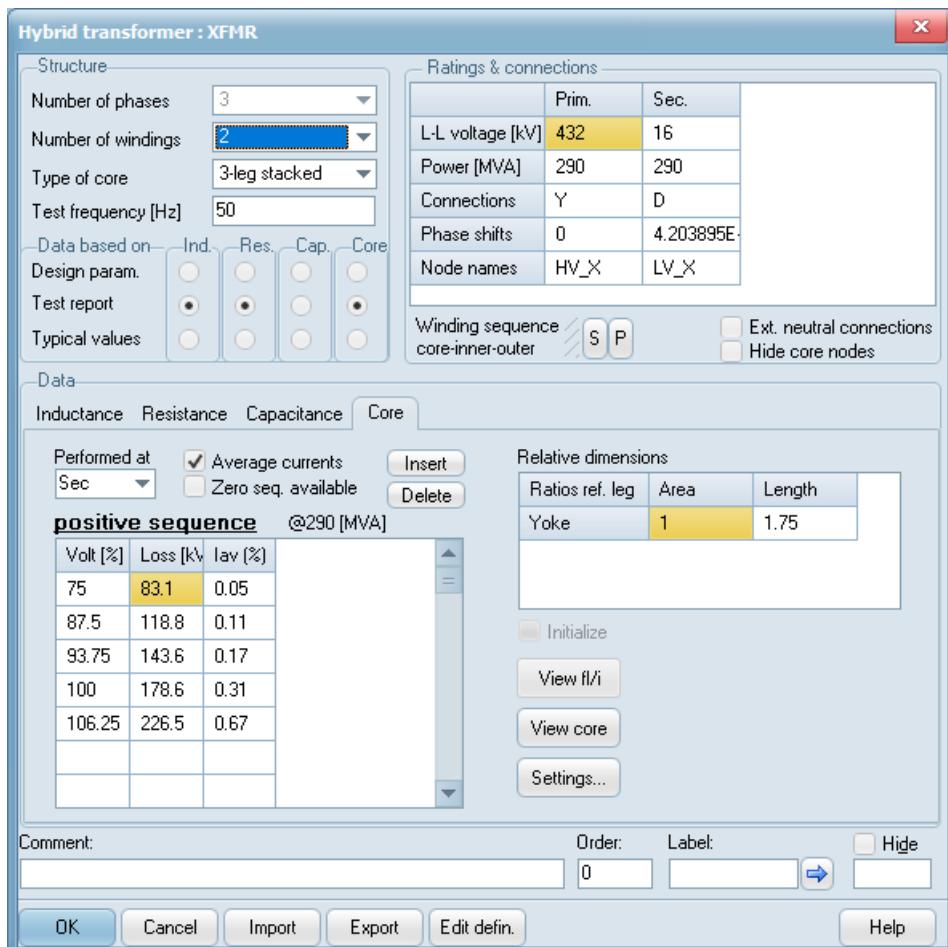


Fig. 5.69 – The XFMR component dialog box

When the user presses *OK* the electrical model data (A and C matrices, R, and Core) are calculated and stored internally. The calculation of the core model might take up to one minute and a progress bar is shown (the user can press ESC to stop the calculation). The data can be exported (*Export* button) to an external library file (.xfm) for later import, but also copied between projects. Using the *Import* button, it is possible to load a previously created .xfm file.

Twelve radio buttons are available under *Structure* and *Data based on* that enables the user to set the source of data individually for each part of the model. Click the right mouse button to omit the part completely (inductance cannot be omitted). *Inductance*, *Resistance*, *Capacitance* and *Core*. Under *Type of core* the user can select the core configuration. Triplex (single phase bank), 3- and 5-legged stacked, and shell form cores (Shellform B) are supported. The type of core will influence the structure and calculation process of the core model. A 5-legged core will have a saturation characteristic also for the outer legs, while in the case of a 3-legged core this is replaced by a constant inductance representing the zero-sequence behavior.

Under *Ratings & Connections* the user must specify the line-to-line voltage in [kV], the rated power of the transformer [MVA] and the type of coupling and phase shift for each winding. These settings all refer to the Primary (P), Secondary (S), and Tertiary (T) notation. P is on the left side, S on the right side, and T on the top side of the transformer icon. There is no restrictions on the voltage levels here.

The phase-shift referred to the primary winding is specified in the drop-down list. Only possible phase-shifts are listed, except for Zigzag transformer with arbitrary phase shifts in <-60,0> and <0,60>.

The sequence of the winding on the core leg is set in the combo box *Winding sequence*. This is used establish the artificial winding where the core should be connected. If this sequence is unknown then remember that the inner winding usually has the lowest voltage. When the *Ext. neutral connections* button is checked, all neutral points become 3-phase nodes that the user has to connect in the circuit manually (allows measurement of individual phase currents in the neutral).

For design data, the user must input the geometry and material data of the winding and core. For the core the user must choose a magnetic material. The list of available material data is very limited and only relatively new characteristics are included. This means that a modeling of an old transformer using this approach would result in too low core losses. Uncertain aspects of the design data are the core losses and the zero-sequence data especially for 3-legged transformers.

For test report data, ATPDraw has an embedded BCTRAN-like routine for calculation of the A-matrix and winding resistance R. The core model is established by fitting the measured excitation currents and losses. The user can specify 9 points on an excitation characteristic. Some *Insert* and *Delete* buttons are available. ATPDraw will also sort the points by increasing voltage level. If the current and core loss do not increase with voltage an error message is displayed.

For typical values some estimation is made based on textbook tables using the rated voltage and power. In the Typical data page there is a button *Edit reactances*, *Edit resistances*, *Edit capacitances*, or *Edit magnetization*. When the user check this button, ATPDraw calculates the typical values based on the rated quantities and display the typical values. The values are then locked. To update the values based on a new setting of rated values the user must uncheck the button. There are basically two levels of sophistication available.

- The default level requires no user input at all; the inductance, resistance, capacitance and core data is calculated based on typical values from tables. The user is allowed to specify a few data to improve the guessing; type of cooling for inductances (unknown=forced air), coupling factor for capacitances, and rated magnetic field intensity Bmax, loss density Pmax, and basic insulation level for core modeling. The user can examine the internally calculated data by checking an Edit button this also enables the second level. Once the button is checked the data are no longer updated when the rated voltage or power is changed.
- At the second level the user can directly specify the data.

Some buttons are available for viewing the winding and core design. If these buttons are checked a separate on-top window pops up with the information required to specify the input correctly. The Configuration image changes with the number and type of winding and the core type. The figures are fixed and are not scaled with the user specified dimensions.

Click on the *Settings* button on the core page to set some parameters for the core model. This will bring up the Advanced core settings dialog. An important setting is the #points in saturation; the internal core model based on the Frolich equation (2 or 3 parameter option) is fitted to the test report with a fast Gradient optimization method by minimizing the different between the measured and calculated rms currents. This is then converted to a piecewise linear characteristic (type 93 or 98 inductors) assuming a certain number of points. Type 96 hysteretic inductors are also supported, and in this case half the core loss is assumed to be hysteresis losses and the core loss is in general assumed to be proportional to the square of the flux density. Initialization is challenging for the type 96 inductors and ramping up the power supply with a controlled source

might be necessary at least for a 5-legged core. A very important parameter for inrush studies is the final slope inductance  $L_a$ . Design parameters are required here and  $L_a = \mu_0 \cdot N^2 \cdot A_{leg} / l_{leg}$ .

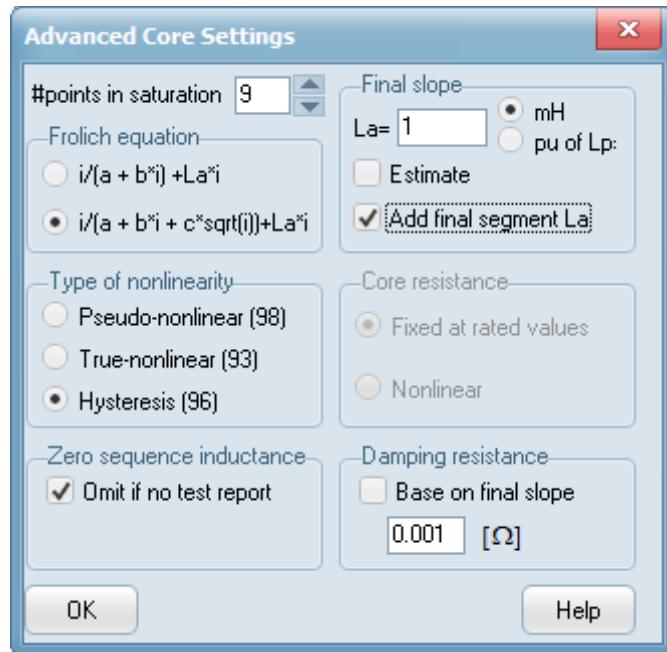


Fig. 5.70 – The Advanced core settings dialog.

## 5.8 Creating new circuit objects in ATPDraw

The user specified objects (USP) are either customized standard objects or objects created for the use of \$INCLUDE and DATA BASE MODULARIZATION feature of ATP-EMTP. The *Objects | User Specified / New sup-file* menu enables the user to create a new support file for such a user specified object or customize data/node properties and the icon or the help text of an existing one. The number of nodes and data specified in the *Edit Object* dialog box for USP objects must be in line with the ARG and NUM declarations in the header section of the Data Base Module (DBM) file. The number of data must be in the range of 0 to 36, and the number of nodes in the range of 0 to 12. The USP support files are normally located in the /USP folder.

Two new circuit objects will be created in this section: a 6-pulse controlled thyristor-rectifier bridge that is used as building block for simulating a 12-pulse HVDC station (*Exa\_6.acp*) in section 6.3 of the Application Manual, and a generator step-up transformer model with winding capacitances and hysteretic core magnetism included. The latter object is used in a transformer inrush current study (*Exa\_11.acp*) in section 6.5.2 of the Application Manual.

### 5.8.1 Creating a 6-phase rectifier bridge

The Data Base Module (DBM) file shown next describes a 6-pulse thyristor rectifier bridge (based on exercise 54 in [2]). The process of creating a DBM-file is certainly the most difficult part of adding new circuit objects to ATPDraw. The input file to the DBM supporting routine of ATP begins with a header declaration followed by the circuit description. The ATP Rule Book [3] chapter XIX-F explains in detail how to create such a file. The output punch-file of the DBM supporting routine can actually be considered as an external library file which is included to the ATP simulation at run time via a \$INCLUDE call.

```

BEGIN NEW DATA CASE --NOSORT--
DATA BASE MODULE
$ERASE
ARG,U_____,POS_____,NEG_____,REFPOS,REFNEG,ANGLE_,Rsnub_,Cssub_
NUM,ANGLE_,Rsnub_,Cssub_
DUM,PULS1_,PULS2_,PULS3_,PULS4_,PULS5_,PULS6_,MID1_,MID2_,MID3_
DUM,GATE1_,GATE2_,GATE3_,GATE4_,GATE5_,GATE6_,VAC_____,RAMP1_,COMP1_
DUM,DCMP1_,DLY60D
/TACS
11DLY60D .002777778
90REFPOS
90REFNEG
98VAC_____=REFPOS-REFNEG
98RAMP1_58+UNITY 120.00 0.0 1.0VAC_____
98COMP1_= (RAMP1_-ANGLE_/180) .AND. UNITY
98DCMP1_54+COMP1_ 5.0E-3
98PULS1_= .NOT. DCMP1_.AND. COMP1_
98PULS2_54+PULS1_ DLY60D
98PULS3_54+PULS2_ DLY60D
98PULS4_54+PULS3_ DLY60D
98PULS5_54+PULS4_ DLY60D
98PULS6_54+PULS5_ DLY60D
98GATE1_= PULS1_.OR. PULS2_
98GATE2_= PULS2_.OR. PULS3_
98GATE3_= PULS3_.OR. PULS4_
98GATE4_= PULS4_.OR. PULS5_
98GATE5_= PULS5_.OR. PULS6_
98GATE6_= PULS6_.OR. PULS1_
/BRANCH
$VINTAGE,0
    POS U A Rsnub_ Cssub_
    POS U BPOS U A
    POS U CPOS U A
    U ANEG POS U A
    U BNEG POS U A
    U CNEG POS U A
/SWITCH
11U____APOS____ GATE1_
11U____BPOS____ GATE3_
11U____CPOS____ GATE5_
11NEG____U____A GATE4_
11NEG____U____B GATE6_
11NEG____U____C GATE2_
BEGIN NEW DATA CASE
C <= "C" in the 1st column is mandatory here!
$PUNCH
BEGIN NEW DATA CASE
BLANK

```

The header section of the DBM-file starts with an ARG declaration after the special ATP request card DATA BASE MODULE. It's function is to specify the external variables (numerical + node names) and the sequence of arguments for the \$INCLUDE procedure. The NUM card tells what arguments are numerical. DUM card lists the dummy or local variables, which are typically internal node names. ATP gives dummy nodes a unique name and thus let you use the same DBM-file several times in a data case avoiding node name conflicts. The rest of the DBM-file describes the rectifier bridge in a normal ATP data structure, except that sorting cards /TACS, /BRANCH, /SWITCH etc., are used in a special way. Sorting cards are required, but no BLANK TACS, BLANK BRANCH, etc. indicators are needed.

The 3-phase thyristor bridge has a 3-phase AC input node and two single phase DC output nodes. The firing angle is taken as input data and the snubber parameters are also practical to consider as numerical input to the model. The model created here accepts external reference signals for the

zero crossing detector (alternatively the DBM module file could have detected its own AC input), thus the new USP object will have 5 nodes and 3 data:

- U\_\_\_\_\_ : The AC 3-phase node
- POS\_\_\_\_\_ : The positive DC node
- NEG\_\_\_\_\_ : The negative DC node
- REFPOS: Positive reference node.
- REFNEG: Negative reference node.
- ANGLE\_ : The firing angle of the thyristors.
- Rsnub\_ : The resistance in the snubber circuits.
- Csnub\_ : The capacitance in the snubber circuits.

Note the importance of the number of characters used for each parameter. The U\_\_\_\_\_ parameter has only 5 characters, because it is a 3-phase node and the extensions A, B and C are added inside the DBM-file. Underscore characters ‘\_’ has been used to force the variables to occupy the 6 characters space for node names and 6 columns (\$VINTAGE, 0) for the snubber data. Running the DBM-file through ATP will produce a .pch punch file shown below:

```

KARD 3 4 5 6 6 6 7 7 8 8 8 9 9 10 10 10 11 11 11 12 12 12 12 13 13 13
      14 14 14 15 15 15 16 16 16 17 17 17 18 18 18 19 19 19 19 20 20 20 20 21 21 21 24
      24 24 24 25 25 25 26 26 26 26 27 27 27 27 28 28 28 28 28 29 29 29 29 29 31 31
      31 32 32 32 33 33 33 34 34 34 35 35 35 36 36 36
KARG-20 4 5 4 5-16-16-17 6-17-18-18-19 -1-18-19 -1 -2-20 -2 -3-20 -3 -4-20
      -4 -5-20 -5 -6-20 -1 -2-10 -2 -3-11 -3 -4-12 -4 -5-13 -5 -6-14 -1 -6-15 1
      2 7 8 1 1 2 2 1 1 2 2 1 1 2 3 1 1 2 3 1 1 2 3 1 1 2 3 1 2
      -10 1 2-12 1 2-14 1 3-13 1 3-15 1 3-11
KBEG 3 3 3 12 19 3 69 3 20 13 3 12 3 3 32 19 12 3 69 12 3 69 12 3 69
      12 3 69 12 3 69 13 25 3 13 25 3 13 25 3 13 25 3 13 25 3 25 13 3 9
      3 27 39 9 21 3 15 9 21 3 15 3 21 15 9 3 21 15 9 3 21 15 9 3 21 15 9 3 9
      65 3 9 65 3 9 65 9 3 65 9 3 65 9 3 65
KEND 8 8 8 17 24 8 74 8 25 18 8 17 8 8 37 24 17 8 74 17 8 74 17 8 74
      17 8 74 17 8 74 18 30 8 18 30 8 18 30 8 18 30 8 18 30 8 30 18 8 13
      8 32 44 13 25 8 20 13 25 8 20 7 25 20 14 7 25 20 14 7 25 20 14 7 14
      70 7 14 70 7 14 70 13 8 70 13 8 70 13 8 70
KTEX 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
      1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
$ERASE
/TACS
11DLY60D .002777778
90REFPOS
90REFNEG
98VAC____ =REFPOS-REFNEG
98RAMP1_58+UNITY                               120.00   0.0   1.0VAC_____
98COMP1_ =(RAMP1_-ANGLE_/_180) .AND. UNITY
98DCMP1_54+COMP1_                               5.0E-3
98PULS1_ = .NOT. DCMP1_ .AND. COMP1_
98PULS2_54+PULS1_                               DLY60D
98PULS3_54+PULS2_                               DLY60D
98PULS4_54+PULS3_                               DLY60D
98PULS5_54+PULS4_                               DLY60D
98PULS6_54+PULS5_                               DLY60D
98GATE1_ = PULS1_ .OR. PULS2_
98GATE2_ = PULS2_ .OR. PULS3_
98GATE3_ = PULS3_ .OR. PULS4_
98GATE4_ = PULS4_ .OR. PULS5_
98GATE5_ = PULS5_ .OR. PULS6_
98GATE6_ = PULS6_ .OR. PULS1_
/BRANCH
$VINTAGE,0

```

```

POS ____ U ____ A           Rsnub_
POS ____ U ____ BPOS ____ U ____ A
POS ____ U ____ CPOS ____ U ____ A
U ____ ANEG ____ POS ____ U ____ A
U ____ BNEG ____ POS ____ U ____ A
U ____ CNEG ____ POS ____ U ____ A
/SWITCH
11U     APOS
11U     BPOS
11U     CPOS
11NEG   U     A
11NEG   U     B
11NEG   U     C
$EOF    User-supplied header cards follow.      31-May-02  15.46.06
ARG, U____, POS____, NEG____, REFPOS, REFNEG, ANGLE____, Rsnub____, Csnu_____
NUM, ANGLE____, Rsnub____, Csnu_____
DUM, PULS1____, PULS2____, PULS3____, PULS4____, PULS5____, PULS6____, MID1____, MID2____, MID3_____
DUM, GATE1____, GATE2____, GATE3____, GATE4____, GATE5____, GATE6____, VAC____, RAMP1____, COMP1_____
DUM, DCMP1____, DLY60D

```

This file is very similar to the DBM input file, but with a different header and with the original DBM-file header given at the bottom instead. This file is ready to \$INCLUDE into an ATP input file by ATPDraw. The file must be given a name and extension .LIB and stored in the default \USP directory. The name HVDC\_6.LIB is used here as an example.

When the punch-file from the DBM-file has been created, the next step is to create a support file for the new HVDC\_6 object in the the *Objects / User Specified* menu. The process of creating a new object consists of two steps: create parameter support and create the icon.

First select the *New sup-file* in the popup menu. A notebook-style dialog box shown in Fig. 5.71 appears where you specify the number of data and nodes. The number of arguments on the NUM card(s) of the DBM-file tells you the *Number of data*, which is 3 in this example. The number of arguments on the ARG card(s) minus number of arguments on the NUM card(s) specifies the total *Number of nodes*, which is 5 in this example.

On the *Data* tab, you specify the names of the data parameters, number of digits (it must be less or equal the space used in the DBM-file, which is 6 in this case) a default value, and the *Min/Max* values. The name of data need not be equal to the names used in the DBM punch-file, but the sequence of data must be the same as on the ARG and NUM card(s). After specifying data properties, click on the *Node* tab and set the node control parameters as shown in Fig. 5.71. The *Name* of nodes, the number of *Phases (1/3)* and the node position on the icon border (1-12) are to be given here. Codes for the available node positions are shown in the icon at right. *Kind* is not used here. It must be left unity (default) for all nodes. The name of the nodes need not be identical with the names used in the DBM-file, but the node sequence must be the same as on the ARG card.

ATPDraw writes all three names of a 3-phase node in the \$INCLUDE statement. In this example only the core name of the 3-phase node is expected on the argument list, because the phase identifiers A-B-C are added internally in the DBM-file. This option requires the *Internal phase seq.* checked box be selected in the component dialog box of the HVDC\_6 object, as shown in Fig. 5.74. If it is selected, ATPDraw writes only the 5-character long core names in the \$INCLUDE statement and let the extensions A, B and C be added inside the DBM library file.

Note that ATPDraw does not perform any diagnosis of the include file before sending the node names. Moreover, the *Internal phase seq.* option may result in conflict with transposition objects. As a result, this option should in general not be used in transposed circuits. To avoid the conflict

use three input names for 3-phase nodes in DATA BASE MODULE files.

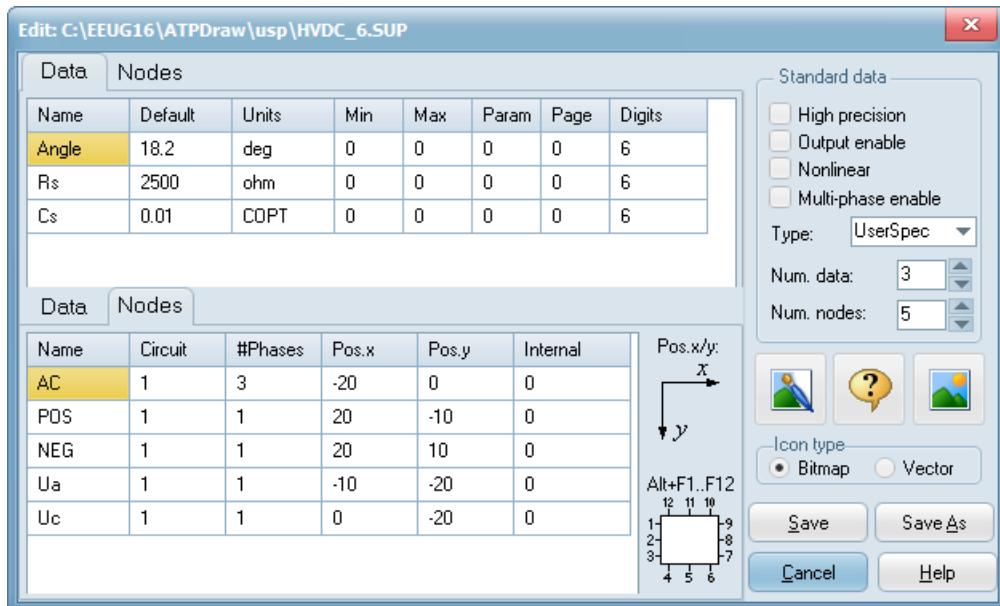


Fig. 5.71 - Properties of the new HVDC\_6 object.

Each user specified objects might have a unique icon, which represents the object on the screen and an optional on-line help, which describes the meaning of parameters. These properties can be edited using the built in *Help* and *Icon Editors*. Fig. 5.72 shows an example file that can be associated with the user specified 6-phase rectifier bridge.

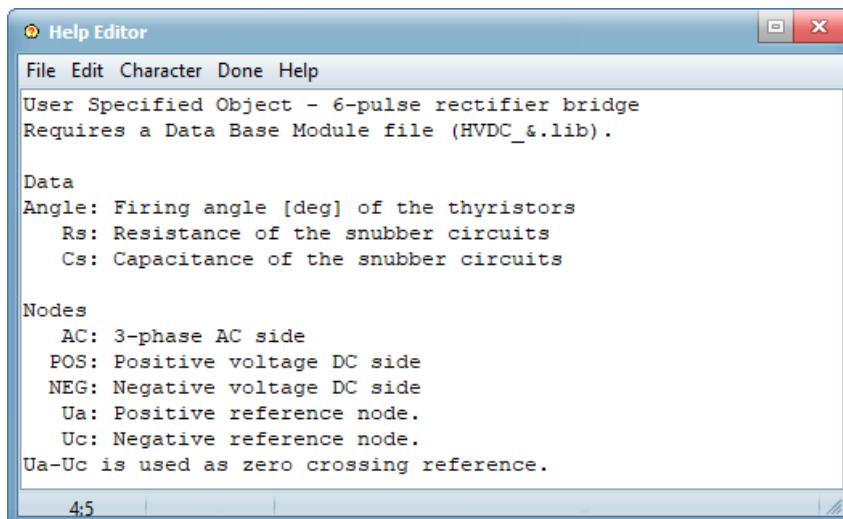


Fig. 5.72 - Help file of the HVDC\_6 object.

Fig. 5.73 shows the icon editor window. The red lines in the background indicate the possible node positions on the icon border. Connecting lines to the external nodes of the object should be drawn from the symbol in the middle and out to the node positions specified in Fig. 5.71. The completed icon of the 6-pulse rectifier bridge is shown in Fig. 5.73.

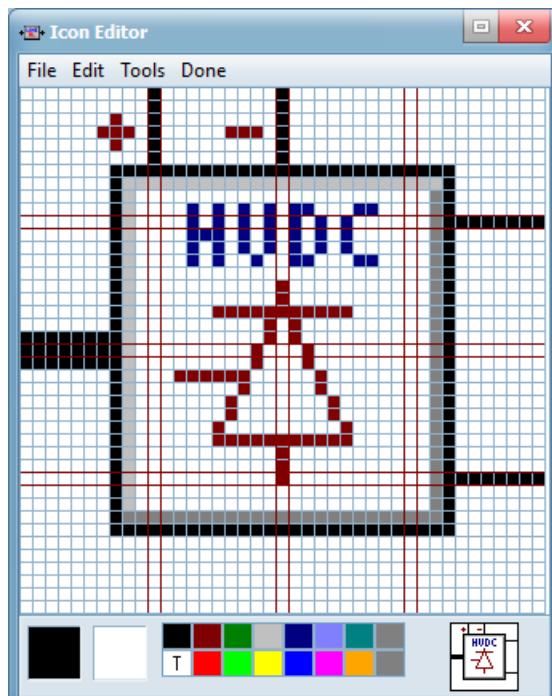


Fig. 5.73 - The icon associated with the new HVDC\_6 object.

Finally, the just created support file must be saved to disk using the *Save* or *Save As* buttons. User specified sup-files are normally located in the \USP folder and their default extension is .sup. You can reload the support file of any user specified objects whenever you like, using the *User Specified / Edit sup-file* option of the *Objects* menu.

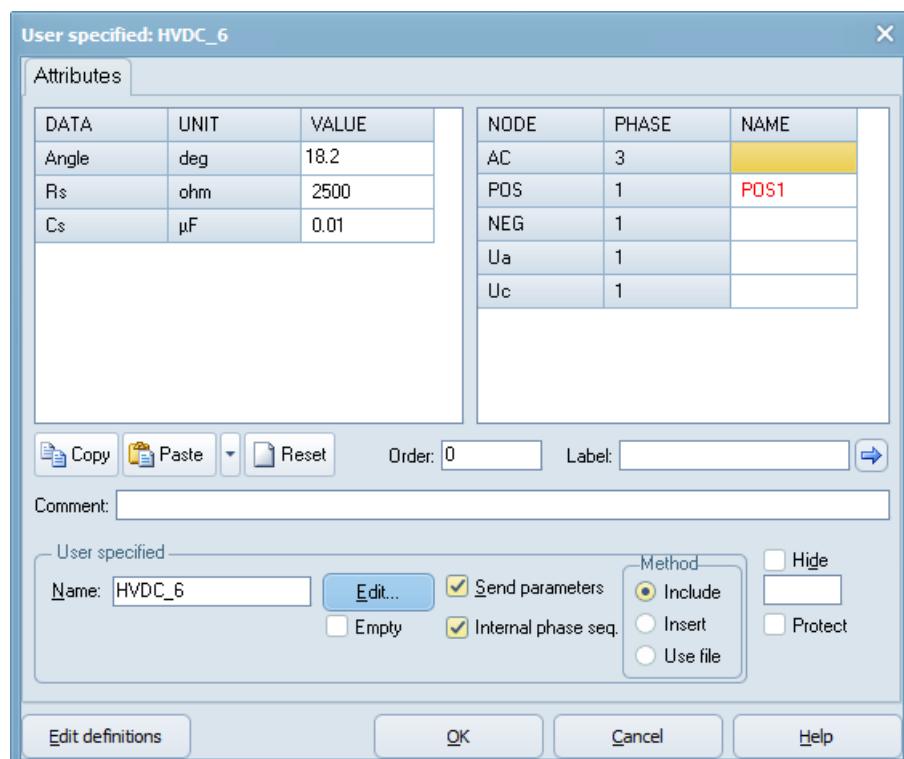


Fig. 5.74 - Component dialog box of the new user specified HVDC\_6 object.

The *User Specified | Files* in the component selection menu provides access to the user specified objects. The component dialog box of the HVDC\_6 object is very similar to that of the standard objects, as shown in Fig. 5.74. The name of the DBM-file which is referenced in the final ATP input file must be specified in the *\$Include* field under *User specified*. The *Send parameters* check box is normally selected, if the USP object has at least one input node or data. The inserting method can be either *\$Include* (data base module) or *\$Insert* or simply *Use file* which does not write anything to disk but instead relies on a file coming from a different source.

### 5.8.2 Creating a user specified, nonlinear transformer model

Supporting routine BCTRAN can be used to derive a linear representation of a single or 3-phase multi-winding transformer, using excitation and short circuit test data. If the frequency range of interest does not exceed some kHz, the inter-winding capacitances and earth capacitance of the HV and LV windings can be simulated by adding lumped capacitances connected to the terminals of the transformer. Although BCTRAN produces only a linear representation of the transformer, connecting nonlinear inductances to the winding closest to the iron core as external elements, provides an easy way to take the saturation and/or hysteresis into account. It is noted that the BCTRAN object is now supported by ATPDraw in a user friendly way (see in section 5.6), but the procedure described here gives more flexibility in handling of the iron core nonlinearities and allows incorporation of winding capacitances in the USP object, if needed. Further advantage of the USP based modeling is that users do not need to run the BCTRAN supporting routine as many times as such kind of transformers present in the circuit before the execution of the time domain simulation. Creating such a user specified component however requires some experience in two ATP supporting routines: DATA BASE MODULE and BCTRAN.

The BCTRAN model requires easily available input data only, like the name-plate data of a generator step-up transformer shown below:

Voltage rating V <sub>high</sub> /V <sub>low</sub>	132/15 kV
Winding connection:	Ynd11
Power rating:	155 MVA
Excitation losses:	74 kW
Excitation current:	0.3% / 2.67 A
Short circuit losses:	461 kW
Short circuit reactance:	14 %

The zero-sequence excitation current and losses are approximately equal to the positive sequence measurements because the presence of delta connected secondary winding. Taking that the nonlinear magnetizing inductance is going to be added to the model as an external element, only the resistive component of the excitation current (0.05%) must entered in the BCTRAN input file shown next:

```

BEGIN NEW DATA CASE
ACCESS MODULE BCTRAN
$ERASE
 2      50.      0.05      155.      74.      0.05      155.      74. 0 2 2
 1    76.21
 2    15.0
 1 2   461.      14.0      155.      14.0      155. 0 1
BLANK
$PUNCH
BLANK
BEGIN NEW DATA CASE
BLANK
BLANK

```

Running this file through ATP will produce an output punch-file that can be used as input for the Data Base Module (DBM) run. The process of creating a DBM-file is certainly the most difficult part of adding new circuit objects to ATPDraw. The input file to the DBM supporting routine of ATP begins with a header declaration followed by the circuit description. The ATP Rule Book [3] chapter XIX-F explains in detail how to create such a file. The output of the DBM supporting routine is a *.lib* file, that can actually be considered as an external procedure which is included to the ATP simulation at run time via a \$INCLUDE call.

### 5.8.2.1 Creating a Data Base Module file for the BCTRAN object

The DBM-file begins with a header declaration followed by the ATP request card DATA BASE MODULE and ends with a \$PUNCH request. The ARG declaration together with the NUM card (if needed) specifies the external variables (numerical + node names) and the sequence of arguments for the \$INCLUDE procedure. The rest of the file describes the BCTRAN model. Note that data sorting card /BRANCH is part of the file, but no BLANK BRANCH indicator is required.

The ARG declaration of the DBM-file includes 7 node names in this example:

HVBUSA, HVBUSB, HVBUSC: The 3-phase node of the high voltage terminal  
 LVBUSA, LVBUSB, LVBUSC: The 3-phase node of the low voltage terminal  
 STRPNT: The 1-phase node of the HV neutral

The rest of the DBM-file is the transformer model description as produced by the BCTRAN supporting routine of ATP. The structure of the DBM input file is shown below:

```
BEGIN NEW DATA CASE --NOSORT--
DATA BASE MODULE
$ERASE
ARG,HVBUSA,HVBUSB,HVBUSC,LVBUSA,LVBUSB,LVBUSC,STRPNT
<<< The .PCH file generated by the >>>
<<< BCTRAN supporting routine must >>>
<<< be inserted here >>>
BEGIN NEW DATA CASE
C           !!! This comment line here is mandatory !!!
$PUNCH, MYTRAFO.LIB
BEGIN NEW DATA CASE
BLANK
BLANK
```

Running the DBM-file through ATP will produce a file *mytrafo.lib* that must be stored in the \USP folder of ATPDraw.

```
KARD  3   3   4   4   6   6  10  10  11  11  13  13  16  16  20  20  25  25
KARG  4   6   4   5   5   6   1   7   4   6   2   7   4   5   3   7   5   6
KBEG  3   9   9   3   9   3   3   9   3   9   3   9   9   3   3   9   9   3
KEND  8  14  14   8  14   8   8  14   8  14   8  14  14   8   8  14  14   8
KTEX  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
$ERASE
C  <++++++> Cards punched by support routine on 28-Jan-02 14.10.13 <++++++>
C ACCESS MODULE BCTRAN
C $ERASE
C  2      50.     0.05      155.       74.     0.05      155.       74.  0 2 2
C  1      76.21          HVBUSASTRPNTHBUSBSTRPNTHVBUASCSTRPNT
C  2      15.0          LVBU SALVBU CLVBU SBLVBU SALVBU CLVBU SBB
C  1 2    461.        14.0      155.       14.0      155.  0 1
C BLANK
$VINTAGE, 1,
1LVBUSALVBUSC          9121.6157726436
```

```

2LVBUSBLVBUSA          0.0
                        9121.6157726436
3LVBUSCLVBUSB          0.0
                        0.0
                        9121.6157726436
USE AR
1HVBUSASTRPNT          19.966704093183 .16716783247242
2LVBUSALVBUSC          -101.4441679294      0.0
                        515.41471986794 .00647606659729
3HVBUSBSTRPNT          0.0
                        0.0
                        0.0
                        19.966704093183 .16716783247242
4LVBUSBLVBUSA          0.0
                        0.0
                        -101.4441679294      0.0
                        515.41471986794 .00647606659729
5HVBUSCSTRPNT          0.0
                        0.0
                        0.0
                        0.0
                        0.0
                        19.966704093183 .16716783247242
6LVBUSCLVBUSB          0.0
                        0.0
                        0.0
                        0.0
                        -101.4441679294      0.0
                        515.41471986794 .00647606659729
$VINTAGE, 0,
$UNITS, -1.,-1.
USE RL
C ----- << case separator >> -----
$EOF User-supplied header cards follow.           28-Jan-02 14.28.28
ARG,HVBUSA,HVBUSB,HVBUSC,LVBUSA,LVBUSB,LVBUSC,STRPNT

```

### 5.8.2.2 Creating new support file and icon

Next step is to create a new user specified object via the *Object / User Specified / New sup file* menu of ATPDraw. The process of creating a new object consists of two steps: creating parameter support and creating an icon. Since no NUM card exists in the DBM header the number of data is 0, the number of nodes is 3 in this example as shown in Fig. 5.75.

On the *Nodes* tab, a *Name* can be assigned to each node. The number of phases and the node position on the icon border must also be specified here. The name of the nodes may differ from the name used in the *.lib* file, but the node sequence must be the same as specified on the ARG list. Each user specified component might have an icon and an optional on-line help, which describes the meaning of input parameters. The appearance of this icon is up to the users' creativity, but it is recommended to indicate three phase nodes with thick lines and to locate them according to the *Pos (1..12)* setting on the *Nodes* tab. Finally, the support file of the object must be saved to disk using the *Save* button (the default location is the */USP* folder), to make the new USP object accessible via the *User Specified | Files* option of the component selection menu.

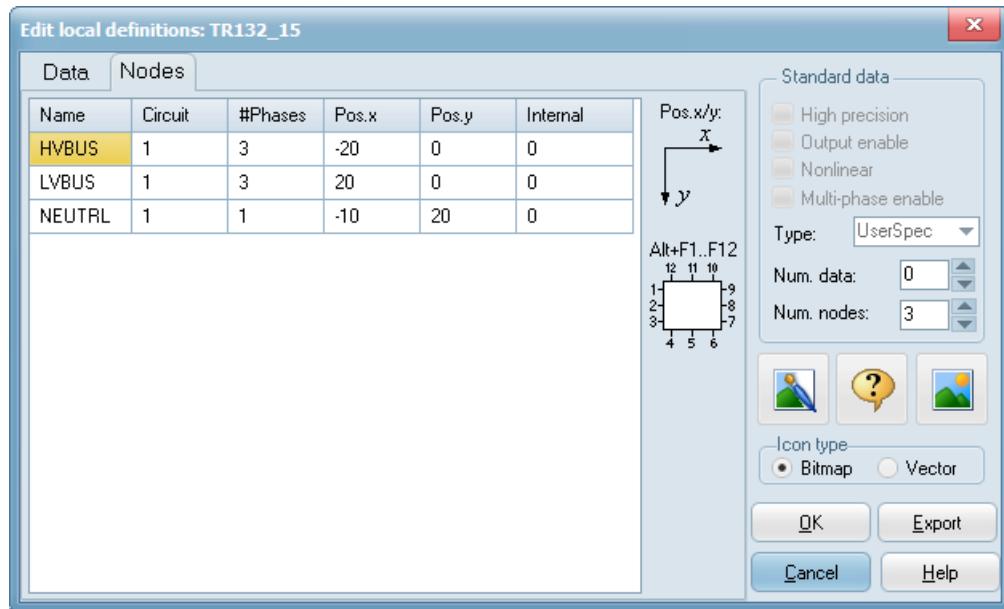


Fig. 5.75 – Creating support file for the new BCTRAN object.

The user specified components can be used in combination with the *Compress* feature of ATPDraw as shown in

Fig. 5.76. In this example, the linear part of the transformer model has been completed with winding capacitances as external components and three nonlinear Type-96 hysteretic inductors in delta connection at the 15 kV terminals, which represent the nonlinear magnetic core.

The *Compress* feature of ATPDraw supports single icon replacement of these 7 objects. The inter-winding and winding-to-earth capacitances are input parameters to the group object. As shown below, the group object's icon can be customized, as well. An artistic icon may improve the readability of the circuit and help in understanding of the circuit file for others.

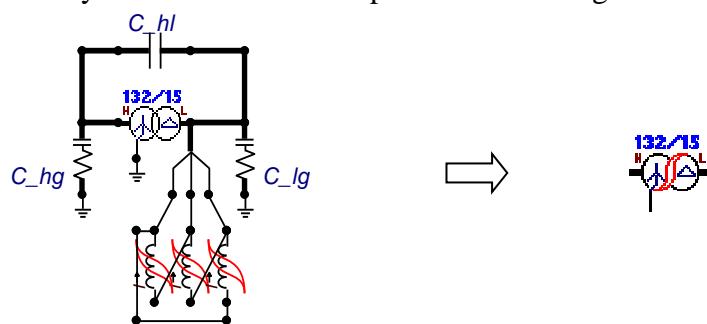


Fig. 5.76- Compressing the transformer model into a single object.

## 5.9 Systematic parameter variations

Very often the user wants to study the maximum voltage at a point as function of a certain parameter. ATPDraw offers features to do this quite easily. To utilize this, Variables must be introduced as function of the simulation number (KNT), and an extremal value extractor WriteMaxMin added. If you turn on *Internal Parser* you can vary most parameters.

As an example, let us say you want to study the maximum lightning overvoltage at a transformer as function of the length of the feeding cable from the busbar and arrester. Fig. 5.77 shows the complete circuit with the Variable LEN declaration in the Sidebar.

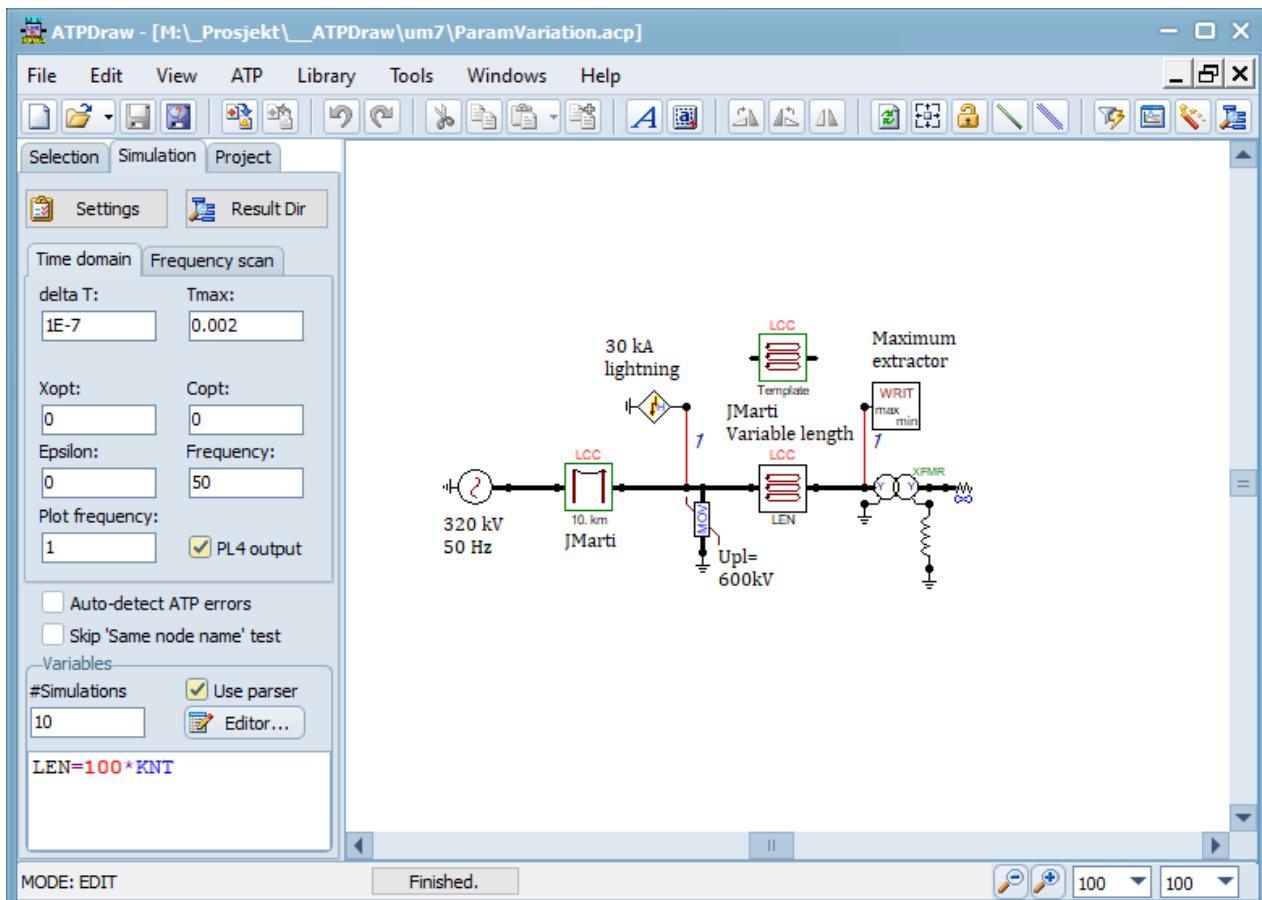


Fig. 5.77- Setup to vary cable (JMarti-model) from 100 to 1000 m in ten steps.

The cable is first modeled as an *LCC template* and an *LCC section* component is connected to it with the *Length* data declared as *LEN* as shown in Fig. 5.78.

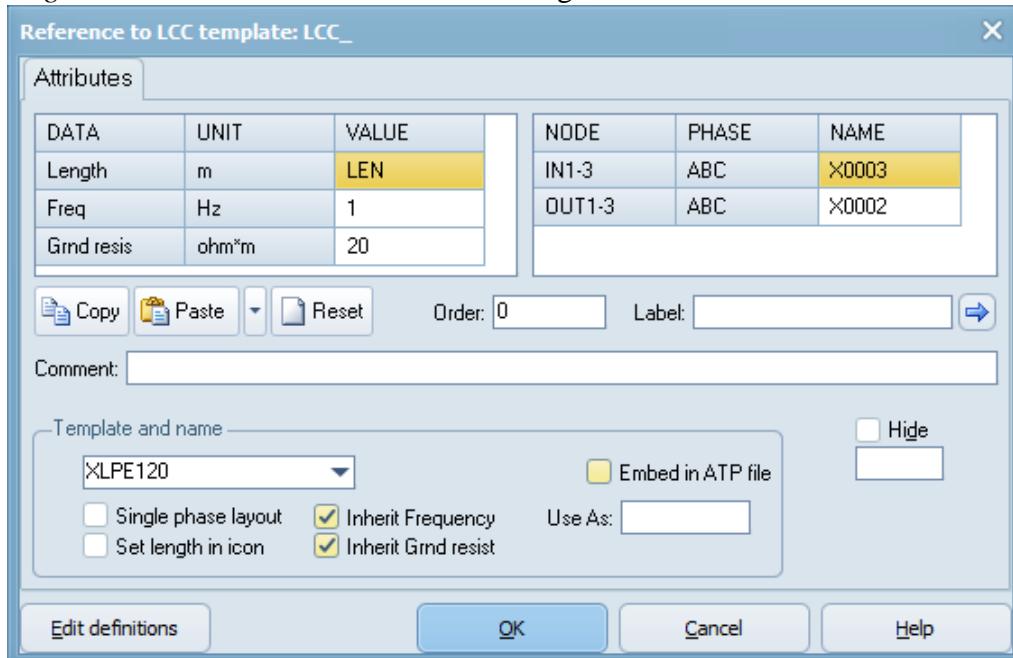


Fig. 5.78- Input of *LCC section* component for length variation.

The component *MODELS/WriteMaxMin* is then connected to record the voltage extremal of interest. This will be phase A of the transformer terminal since the lightning strike also is in phase A on the feeding line. In the input dialog of *WriteMaxMin* as shown in Fig. 5.79, the *LEN*

variable is also assigned to the *AsFuncOf* data. *MaxOrMin* is unity (default). *Tlimit* is the time when the model start looking for extremal values. The user must also click the input node of the component to make sure node voltage is selected (branch current is the default).

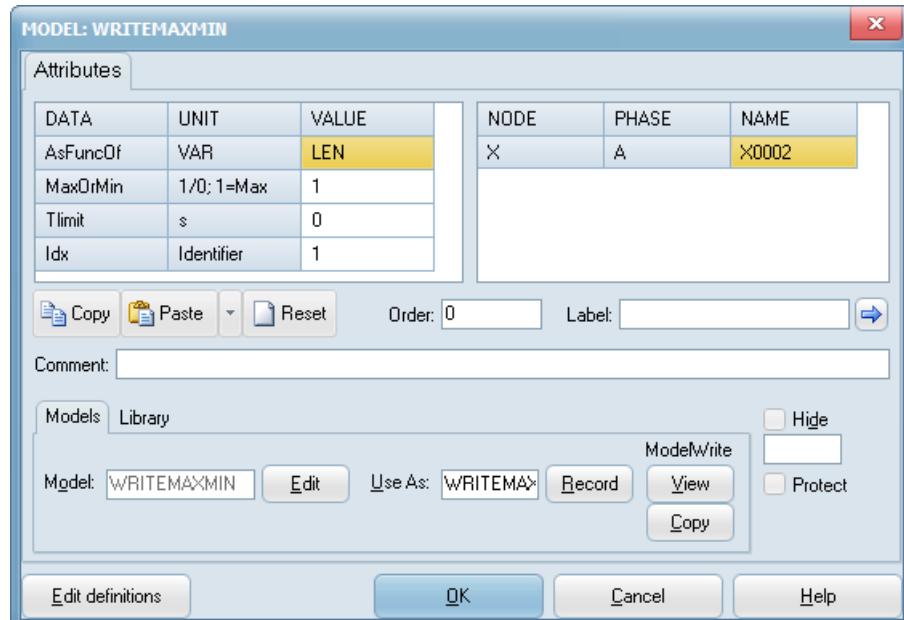


Fig. 5.79- Input of the WriteMaxMin component.

The Variable LEN now will be systematically varied. This can simply be done by declaring it as function of the simulation number (1-10 in this case) represented by the default variable KNT. LEN=100\*KNT means that LEN=100 m in the first run, 200 m in the second run etc. ATPDraw will make sure to run CABLE PARAMETERS every run to create the new cable model with a new length. The WriteMaxMin component will examine the LIS-file every run and extract the extremal value written to it by the MODELS code. If the system is set up correctly with ATP execution in hidden mode and NODISK=1 in graphics.aux, the process runs without any interruption. After the 10 runs counted in the main window's progress bar, the extremal values are available under View in Fig. 5.79 as shown in Fig. 5.80.

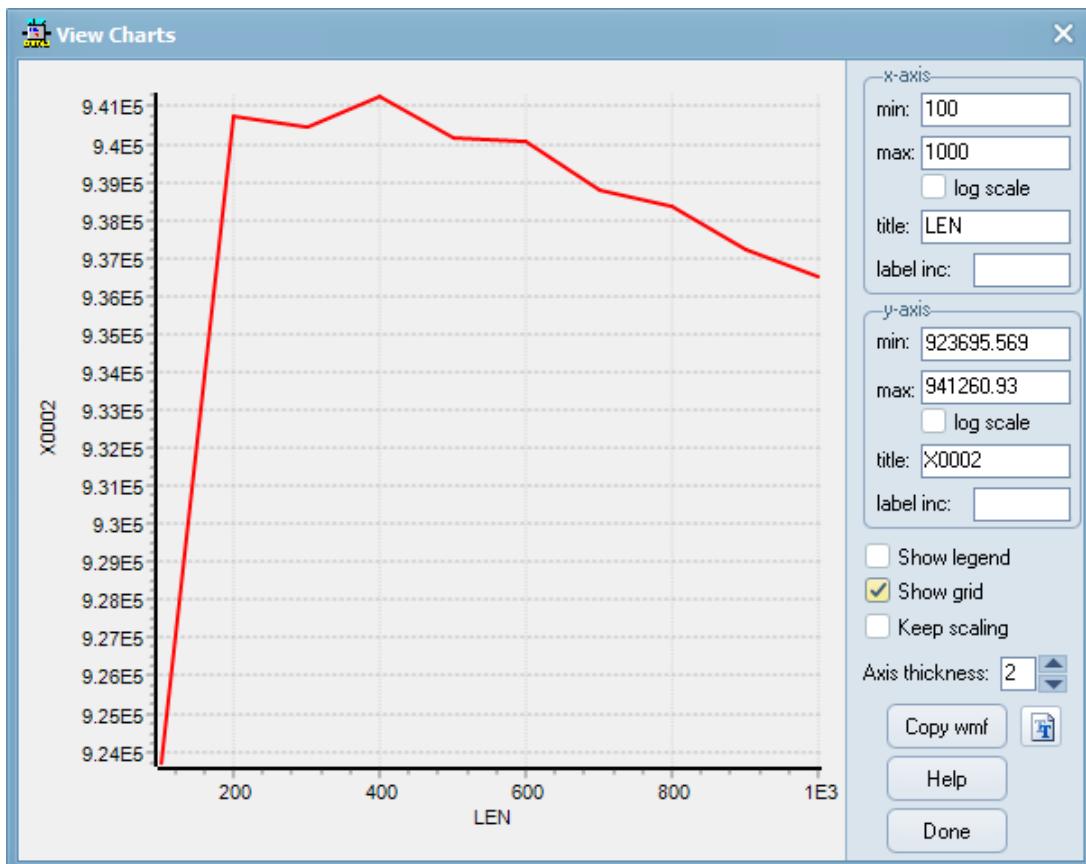


Fig. 5.80- View of the WriteMaxMin component.

The number of runs is not restricted to 10. Note that the maximum simulation time would need to be adjusted to capture the maximum of the longest cable. A text file ParameterVariation.log (same name as created ATP-file) will be created in the Result Folder with information of the variables at each run. Note that a lot of things are happening behind the scene here, so the user is advised to first get the correct behavior of the circuit before jumping into parameter variations.

### 5.9.1 Optimization

This module was added to ATPDraw version 5.6 as part of a co-operation with Schneider Electric. The user must add a cost function object found in the selection menu under *MODELS/Write MaxMin*. This component will extract a single value from the simulation. In addition, variables must be assigned to data in the circuit. These variables can then be tuned to optimize the cost function. The optimization problem is defined as the minimization or maximization of the object function  $OF$  in  $n$  dimensions with variables  $x$ .

$$\begin{aligned} & \max_{\text{min}} OF(x_1, x_2, \dots, x_n) \end{aligned}$$

The variables  $x$  can be selected by the user among the global variables.

#### 5.9.1.1 Optimization routines

Three different optimization routines are supported:

The **Gradient Method** (GM) is the **L-BFGS-B** routine [16] (limited memory algorithm for bound constrained optimization) which is a quasi-Newton method with numerical calculation of the gradient. The gradient is calculated based on the two-point formula:

$$\frac{\partial f}{\partial x} \approx \frac{f(x+h) - f(x-h)}{2h}$$

where the discretization point  $h$  is calculated as  $h = \max(|x|, 10^{-6}) \cdot dx$  where  $dx$  is a user selectable parameter (*delta X*).

If  $n$  is the number of variables in the optimization problem the cost function thus has to be evaluated  $2n+1$  times for each solution point. This is calculated in a single ATP run utilizing PCVP. The iteration number is somewhat loosely defined in the Gradient Method. If the solution is poorer than the previous point the algorithm steps backwards along the gradient until an improved solution is found and only then the iteration number is incremented.

The **Genetic Algorithm** (GA) is based on the RiverSoft AVG package ([www.RiverSoftAVG.com](http://www.RiverSoftAVG.com)), but modified to better handle the variable constraints. This optimization routine might need further improvement and development. The evolution of the solution with GA is to more or less randomly select solutions (individuals) and mate these to obtain new solutions. The selection process can be Random, Roulette (using cumulative distribution), Tournament (competition between a user selectable number of randomly selected rivals), Stochastic Tournament (combination of Roulette and Tournament), and Elitism (select only the user defined best percentage of the population). Tournament with 5-10 rivals is a reasonable starting point. The user has to select the size of the population (maximum 1000) and this is a critical parameter which depends on the problem and the number of variables. The user must also select the resolution with 8, 16 and 32 bits available. This part needs further development to allow integer values and arbitrary resolutions. Up to twenty cost function evaluations are performed in parallel using PCVP of ATP.

The **Simplex Annealing** (SA) method is implemented from Numerical Recipes [17]. It is based on the Nelder-Mead simplex algorithm with an added random behavior gradually reduced (simulated annealing). The algorithm also uses a possible larger set of points (called population) and can support mutation. With all control parameters set to zero the algorithm simply reduces to the classical Nelder-Mead simplex method. The method relies only on function evaluations and POCKET CALCULATOR of ATP is thus not used. Since a single case is run through ATP for each cost function evaluation, the method thus has potential to be extended to include other variables than those defined within the global variables.

### 5.9.1.2 Cost function

A general-purpose Cost Function in MODELS called WRITEMAXMIN is introduced in ATPDraw version 5.6. The idea is to extract a single value from a simulation and write this to the lis-file and read it back when the simulation is finished. The single value is either the maximum or minimum of the signal  $xout$  from time  $Tlimit$  and out to the end time of the simulation. The Model has one input but this can be expanded. The Model also takes in one DATA parameter *AsFuncOf* and if this is assigned to a variable WRITEMAXMIN writes output as function of this data parameter. If *AsFuncOf* is a number, it is simply replaced by the simulation number. WRITEMAXMIN supports multiple run though POCKET CALCULATOR. The selection of the component and its input dialog is shown in Fig. 5.81.

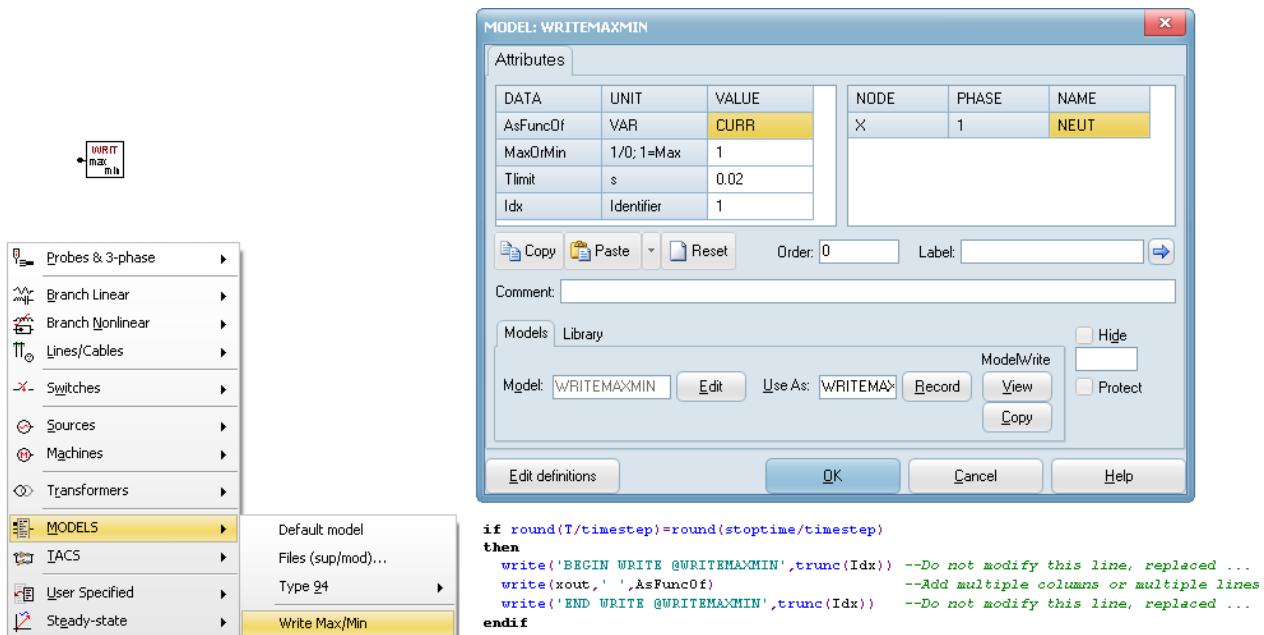


Fig. 5.81- Cost Function WRITEMAXMIN.

### 5.9.1.3 Optimization dialog

The Optimization dialog is found under *ATP/Optimization*. The user has to set up the data case which is not stored with the project. The variables  $x_1..x_n$  are chosen by clicking in the *Variables* column and selecting the available variable in the appearing combo box as shown to the left in Fig. 5.82. The user also must specify the constraints Minimum and Maximum. The Object function must be selected among the available WRITEMAXMIN components in the circuit. The user can then select to minimize or maximize and select a solution method (Genetic Algorithm, Gradient Method or Simplex Annealing). The *Max iter* field is the maximum number of iterations in the solution algorithm.

For the Genetic Algorithm there are several, special selections. The size of the *Population* is a critical parameter. A low number will produce a degenerated result, while a too high number will waste computation time. The maximum allowed number is 1000. The required *Resolution* depends on the selected range (Max-Min). Since it anyhow is recommended to switch to the Gradient Method for fine tuning a 8-bit resolution (255 steps) is normally sufficient. The *Population* count and *Resolution* cannot be changed in the optimization process (Continue). The *Crossover* probability should be set to a high number (<1) as the alternative is cloning. The *Inversion* and *Mutation* probabilities should be set to low numbers, but this depends on the complexity of the problem. High numbers will slow down the convergence considerably. The *Rival count* for Tournaments should be set to a medium value (2-10). A large number here will approach strong elitism and possible degenerated solutions. The *Preserve fittest* option will simply copy the fittest individual to the next generation (weak elitism). The preferred *Selection method* is one of the Tournament types. Elitism can be selected towards the end of the optimization process.

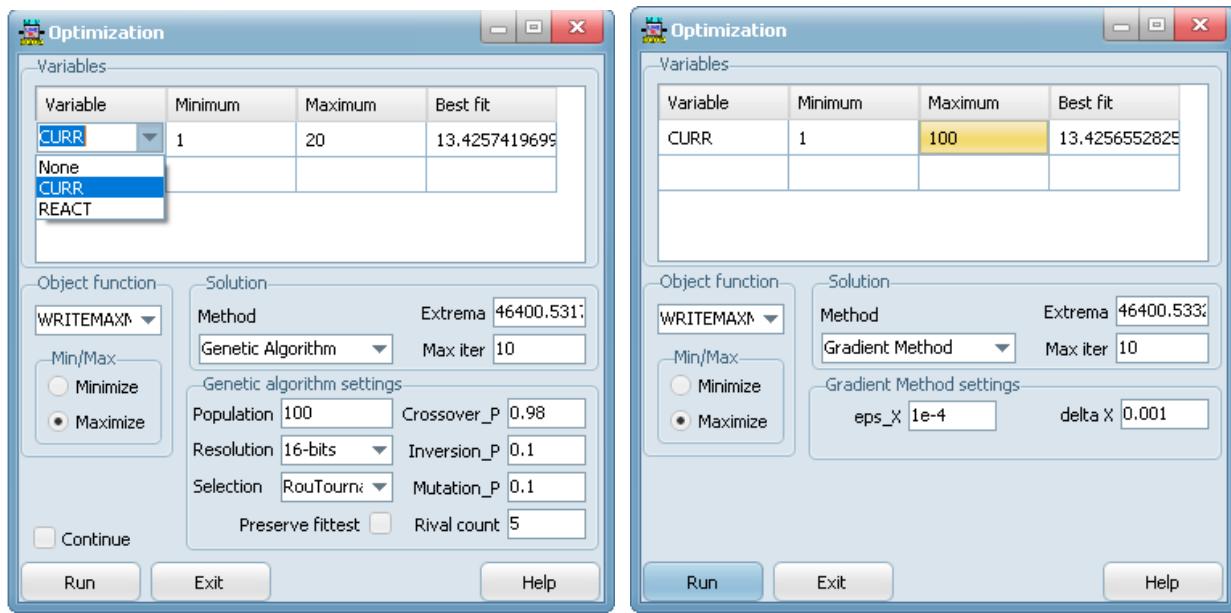


Fig. 5.82- Optimization dialogs.

For the Gradient Method the user has to specify a convergence limit *eps\_x* and a discretization step in per unit *delta X*. Intermediate trial steps do not count as part of the *Max iter*. The user also has to specify the starting point in the *Best fit* column (if blank the average of Minimum and Maximum is assumed).

For the Simplex Annealing method, the user must choose the *Population* (number of points evaluated for membership in the simplex) which is internally restricted to Population=max(Population, n+1). The *Mutation* probability parameter controls if the new points in the simplex is found at random or with the classical methods reflection, expansion or contradiction. The *Max Climbs* parameter controls how many steps in a negative direction that is accepted by the method. This should be a moderate value 0-3. The parameter *beta* (<1) controls annealing schedule (temperature reduction), and the parameter *ratio* (controls the annealing schedule when a local minimum is found. For a rough surface with many local minima the beta and ration parameters need to be increased. *Ftol* is the convergence criterion (the downside of this method). The iteration stops if  $F_{tol} > \frac{2|f_{\max} - f_{\min}|}{|f_{\max}| + |f_{\min}|}$ . With all the other parameters set to zero the Simplex Annealing method becomes equal to the Nelder-Mead simplex method.

The user can press ESC to stop the optimization algorithms. When the user clicks on Exit the result of the optimization are written back to the VALUE field in ATP/Settings/Variables.

#### 5.9.1.4 Example: Resonance grounding (Exa\_18.acp)

Fig. 4 shows a resonance grounding circuit which could be extended to any complexity. The variable REACT is assigned to the neutral inductor and the unit is set to ohms as XOPT is 50. An intermediate variable CURR is used in Fig. 5.84 to vary the current linearly between 1 and 20 Amps (simply by setting CURR=KNT) as this is the standard way of quantifying a resonant grounding.

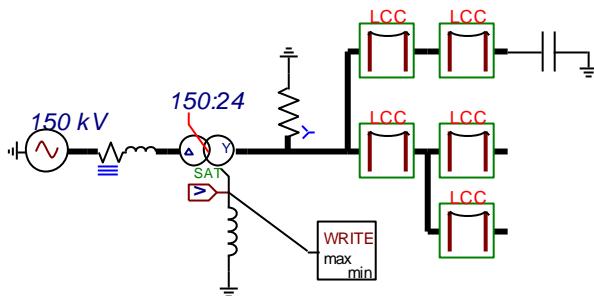


Fig. 5.83- Resonant grounding circuit.

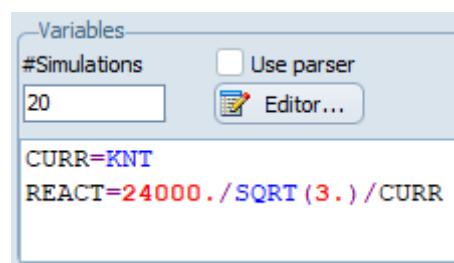


Fig. 5.84- Variable specification

The new, special Model component WRITEMAXMIN is used to extract the maximum value of the neutral voltage as function of the neutral current CURR for all the 20 simulations specified in Fig. 5.84. The input dialog of the Model component is shown in Fig. 5.85. It takes one input and writes the max or min value of this after an onset-time *Tlimit* to the lis-file. After the simulation the results are automatically read back from the lis file and a View button is available for charting the results as shown in Fig. 5.86.

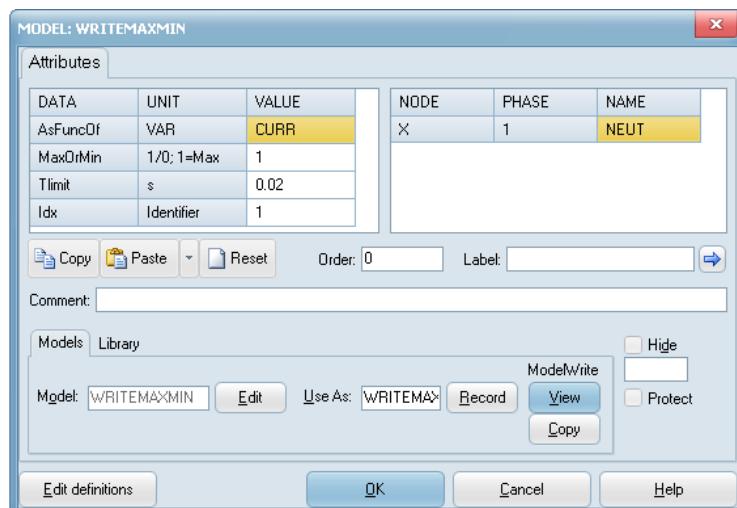


Fig. 5.85- Input dialog of the new WRITEMAXMIN component.

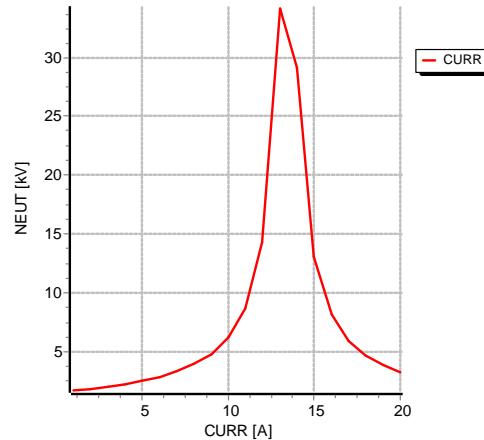


Fig. 5.86- Neutral voltage as function of neutral current.

The exact value of current that corresponds to resonance can be found via the new Optimization module of ATPDraw. This is obtained under *ATP/Optimization* with an input dialog as shown in Fig. 5.82. Fig. 5.82 shows the optimum value found for the GA and GM solution methods. This case with a single variable involved, and a pure convex object function as shown in Fig. 5.86 is simple to solve.

### 5.9.2 MonteCarlo simulations (Exa\_21.acp)

Systematic variations of parameters can be also be made based on statistical functions. ATPDraw offers a series of probability density functions as described in Chapt. 4.2.5. Among these are some special distribution functions suitable for lightning amplitude statistical variations:

- LACIGRE(a b);  $cfd = \begin{cases} 0.5 \cdot (1 + erf\left(\frac{\ln(x/61)}{\sqrt{2} \cdot 1.33}\right)), & a \leq x \leq 20 \\ 0.5 \cdot (1 + erf\left(\frac{\ln(x/33.3)}{\sqrt{2} \cdot 0.605}\right)), & 20 < x < b \end{cases}$ , a and b in kA, answer in A
- LACIGRE1(a,b);  $cfd = 0.5 \cdot \left(1 + \frac{erf\left(\ln(x/31.1)\right)}{\sqrt{2} \cdot 0.484}\right)$ ,  $a \leq x < b$ , a and b in kA, answer in A
- LAIEEE(a,b);  $cfd = 1/(1 + (31/x)^{2.6})$ ,  $a \leq x < b$ , a and b in kA, answer in A

Furthermore, ATPDraw has an overhead line model LCC\_EGM that can be connected to the LCC template component for calculation of lightning stroke hits within multiple segments. The overhead line cross section can be of arbitrary format and tower models, grounding and insulator string flashover modeled in any level of details. The LCC\_EGM model input dialog is shown in Fig. 5.87, where the user must declare variables for the lightning amplitude (Im), the lightning position (x, y), and specify the length and start (Ys) of the segment. The segments are then connected as shown in Fig. 5.88 with a lightning current source connected to all the top nodes (HIT).

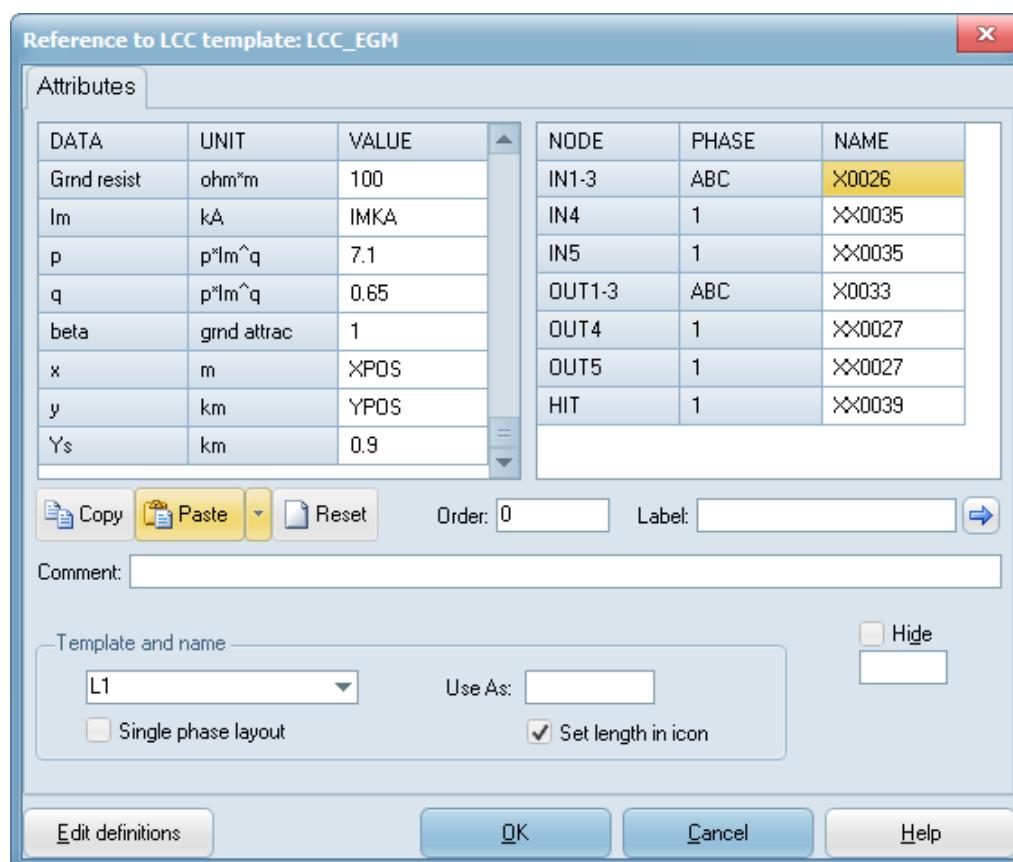


Fig. 5.87- LCC\_EGM input dialog.

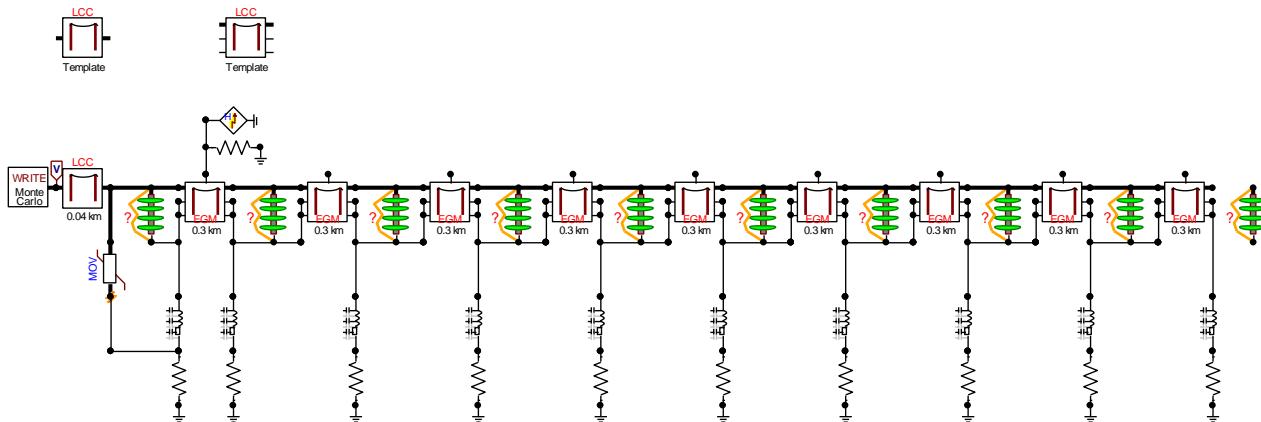


Fig. 5.88- MonteCarlo study case of lightning strikes to overheat line (Exa\_21.acp)

The parameters to vary statistically are the lightning amplitude and position. The Variables in the Sidebar/Simulations are defined as shown in Fig. 5.89. The lightning current source has an amplitude following the CIGRE two-slope log-Normal statistics, assigned to the variable IM, while the LCC\_EGM model is assigned to IMKA. The lightning position is uniform within an area with 100 m width on each side of the line center and 3 km length (10 spans in this case). The Internal Parser must be used to enable the statistical functions. A total number of 8000 simulations is chosen for the MonteCarlo study.

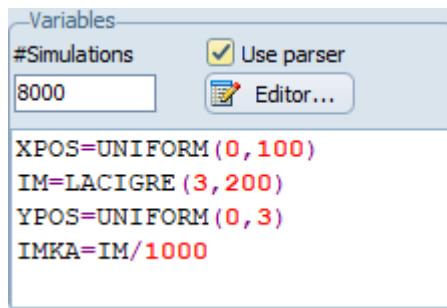


Fig. 5.89- Declarations of Variables

To extract the extremal simulation values and calculate statistical overvoltage distributions and breakdown probability a new component WriteMonteCarlo is added. This builds on the ModelWrite technology and reads in extremal three-phase voltages from the LIS-file like the WriteMaxMin component but has a different View module. Instead of showing the extremal value for each run, the model will show the probability distribution of the voltage. A message box will also show the ratio of voltage exceeding Withstand over the total number of runs. Fig. 5.90 shows the dialog box of the WriteMonteCarlo component and Fig. 5.91 the View module. If Polarity is set to 0 the absolute value of the voltages is used, if Polarity is positive only positive voltages are tested, and with -1 only negative. Resolution and Withstand can be changed before or after the simulation and the View will adapt.

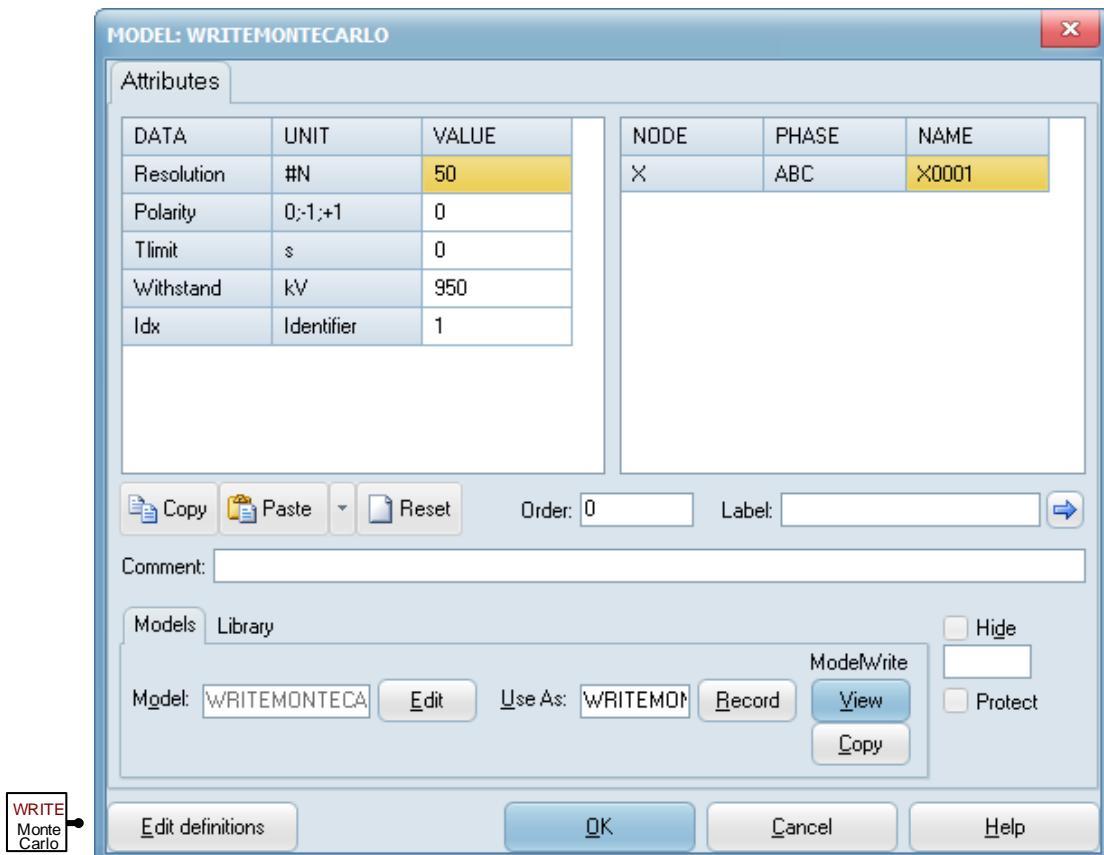


Fig. 5.90- WRITEMONTECARLO icon and input dialog

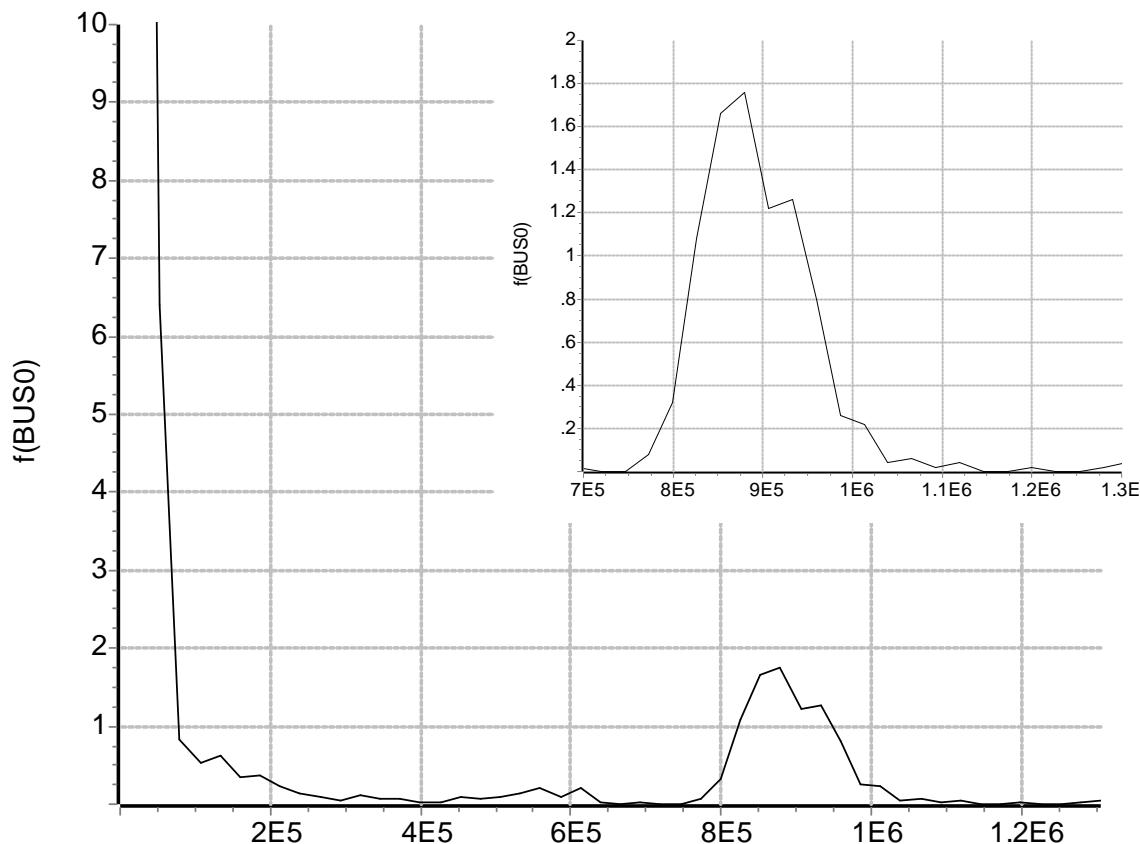


Fig. 5.91- WriteMonteCarlo's View with plotting of the maximum voltage density.



## **6. Application Manual . . .**

.....



ATPDraw™  
for Windows  
**7.6**



This chapter begins with some simple examples. You will not be shown how to create these circuits, but the circuits files `Exa_*.acp` are part of the ATPDraw distribution. To load these example circuits into the circuit window of ATPDraw, use the *File / Open* command (or *Ctrl + O*) and select the file name in the *Open Project* dialog. The resulting ATP-files will be given at the end of each description. Simulation results and/or comparison with measurements are also presented in some cases. These figures have been obtained by processing the `.pl4` output file or field test records with post-processors PlotXY or ATP\_Analyzer.

## 6.1 Switching studies using JMarti LCC objects

The LCC modeling features of ATPDraw are described in detail in section 5.3 of the Advanced Manual. Line modeling by LCC objects means that user specifies the geometrical arrangement and material constants, then ATPDraw executes ATP's Line/Cable Constants routine and converts the output punch-file to DBM library format. The resulting LIB-file will then be included in the final ATP-file via a `$Include` call. The JMarti option is one out of the five alternatives supported by ATPDraw's LCC object. Here two switching transient simulation examples are presented.

### 6.1.1 JMarti model of a 750 kV line

The JMarti line models introduced in this section will be used in the subsequent single-line-to-ground fault study on a 750 kV shunt compensated transmission line with total length of 487 km. Transpositions separate this line into four sections. Each section of the line is represented by 3-phase un-transposed LCC object with JMarti option enabled. The ATPDraw project of the SLG study includes four such objects with name `LIN750_x.ALC`, where `x` runs from 1 to 4. The line configuration is shown in Fig. 6.1.

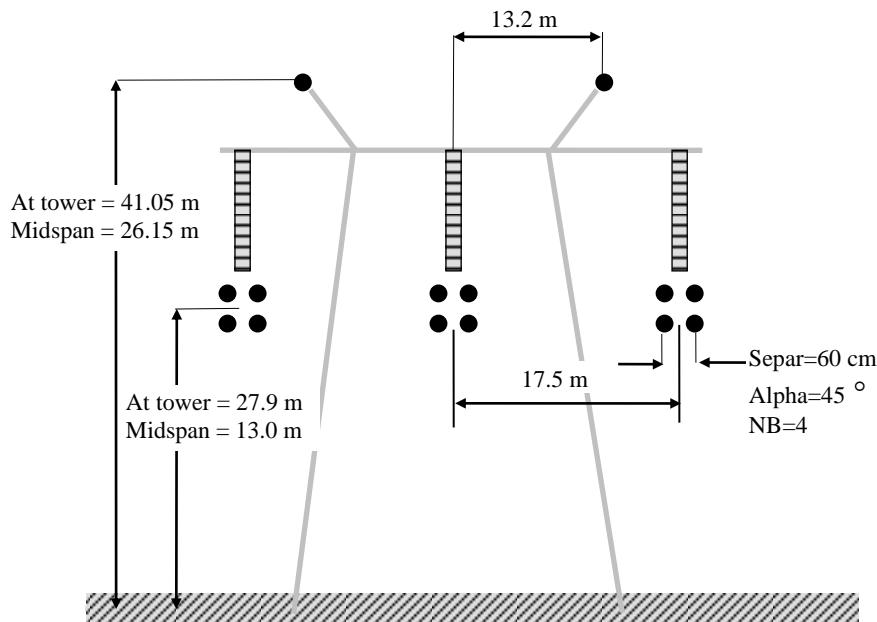


Fig. 6.1 - Tower configuration of the 750 kV line.

The line parameters are given in Metric units. The *Auto bundling* option is enabled to simplify the data entry for this 4 conductor/phase in rectangular arrangement system. Tubular assumption has been applied as in the previous example with the following parameters:

DC resistance =  $0.0585 \Omega/\text{km}$

Outside diameter of the conductors = 3.105 cm.

Inner radius of the tube = 0.55 cm  
ATPDraw calculates the thickness/diameter value internally ( $T/D = 0.32$ ).  
Sky wires are made from steel reinforced conductors, thus tubular assumption applies here, too:

DC resistance =  $0.304 \Omega/\text{km}$

Outside diameter of the sky wire = 1.6 cm

Inner radius of the tube = 0.3 cm

ATPDraw calculates the thickness/diameter value internally ( $T/D = 0.187$ ).

The resistivity of the soil equals to  $20 \Omega\text{m}$ . The conductor separation in the bundle is 60 cm.

Entering the geometrical, material data and model options of the line, then executing *Run ATP* will produce a LIB-file in the /LCC folder. Since the length of each section is different, four LCC objects with different name are needed. The *Save As* button of the LCC dialog box can be used to save the .ALC file with the new length, thus the line parameters need not be entered from scratch.

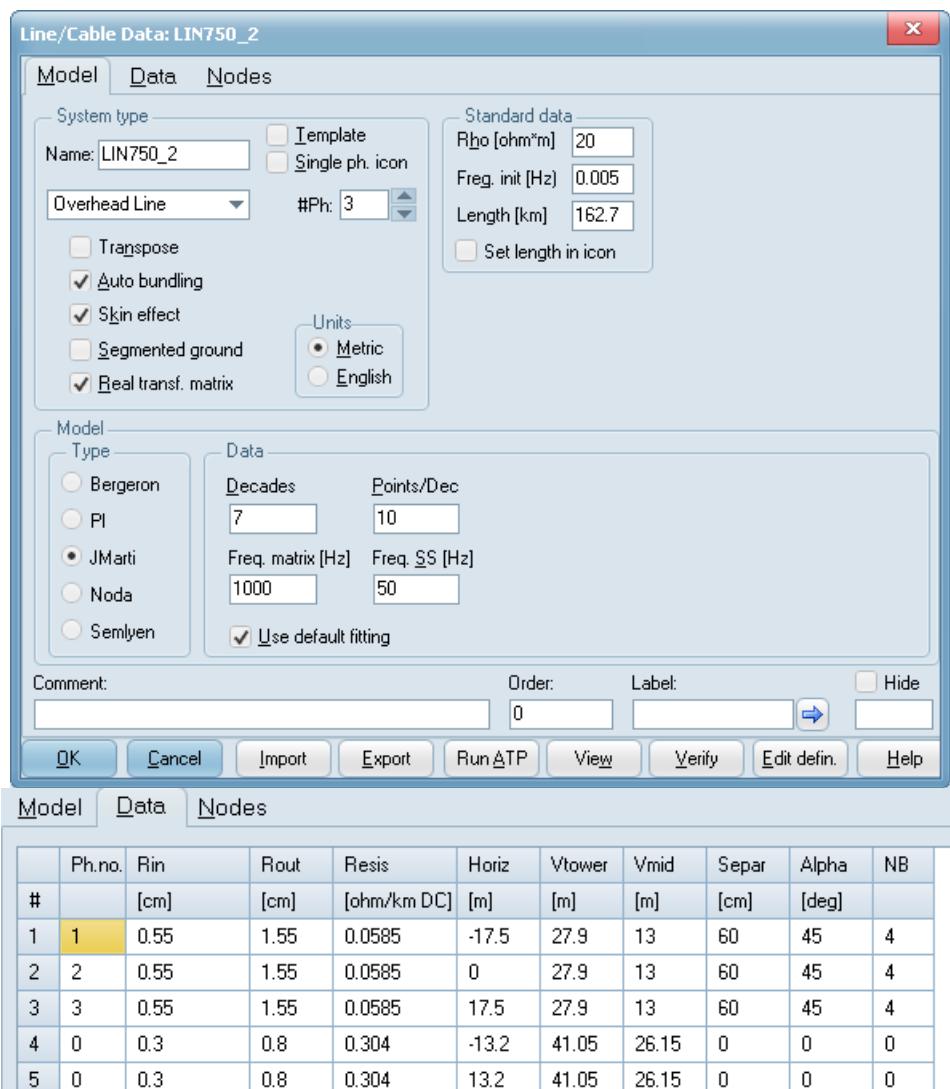


Fig. 6.2- LCC Model and Data tab of the 1<sup>st</sup> section of the 750 kV line.

```
BEGIN NEW DATA CASE
JMARTI SETUP
$ERASE
BRANCH IN__AOUT__AIN__BOUT__BIN__COUT__C
LINE CONSTANTS
```

```

METRIC
10.323 0.0585 4      3.1   -17.5    27.9    13.    60.    45.    4
20.323 0.0585 4      3.1     0.0    27.9    13.    60.    45.    4
30.323 0.0585 4      3.1    17.5    27.9    13.    60.    45.    4
00.313 0.304 4       1.6   -13.2   41.05   26.15   0.0    0.0    0
00.313 0.304 4       1.6    13.2   41.05   26.15   0.0    0.0    0
BLANK CARD ENDING CONDUCTOR CARDS
20.      1.E3           84.6          1
20.      50.            84.6          1
20.      0.005          84.6          7 10 1
BLANK CARD ENDING FREQUENCY CARDS
BLANK CARD ENDING LINE CONSTANT
DEFAULT
$PUNCH
BLANK CARD ENDING JMARTI SETUP
BEGIN NEW DATA CASE
BLANK CARD

```

### 6.1.2 Line to ground fault and fault tripping transients (*Exa\_7a.acp*)

Single-phase to ground fault transients on a 750 kV interconnection are investigated in this study. The one-line diagram of the simulated network is shown in Fig. 6.3. At the sending end of the line shunt reactors are connected with neutral reactors to reduce the secondary arc current during the dead time of the single-phase reclosing. The staged fault has been initiated at the receiving end of the line.

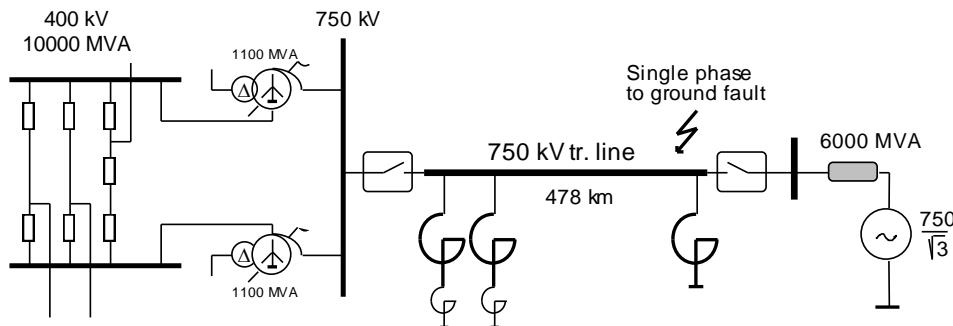


Fig. 6.3 - One line diagram of the faulted line.

The layout of the completed ATPDraw circuit is shown in Fig. 6.4. Along the route three transposition exist, so each LCC object represents a line section between two transpositions with length 84.6 km, 162.7 km, 155.9 km, 75.7 km, respectively.

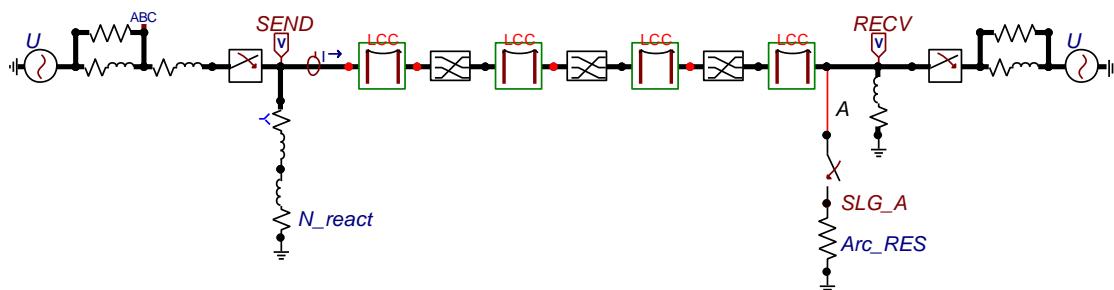


Fig. 6.4 - Line-to-ground fault study (*Exa\_7a.acp*)

The supply network model is rather simple: a Thevenin equivalent 50 Hz source and a parallel resistor representing the surge impedance of the lines erected from the 400 kV bus. An uncoupled series reactance simulates the short circuit inductance of the 400/750 kV transformer bank. The single-phase shunt reactors are represented by linear RLC components. Nonlinearities need not

been considered here, because the predicted amplitude of the reactor voltage is far below the saturation level of the air gapped core. The impedance of the fault arc is considered as 2 ohm constant resistance.

The ATPDraw generated ATP-file for this 750 kV example circuit is shown next:

```
BEGIN NEW DATA CASE
C -----
C Generated by ATPDRAW August, Monday 26, 2019
C A Bonneville Power Administration program
C by H. K. Høidalen at NTNU/SEfAS - NORWAY 1994-2019
C -----
$DUMMY, XYZ000
C dt >< Tmax >< Xopt >< Copt >
 2.E-5   .5
 500     3    0    0    1    0    0    1    0
C 1       2    3    4    5    6    7    8
C 34567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R >< L >< C >
  SLG_A          2.          0
    XX0008        1.  300.          0
  X0012CX0014C  5.  180.          0
  X0012AX0014A  5.  180.          0
  X0012BX0014B  5.  180.          0
  X0012CX0014C  150.          0
  X0012AX0014A  150.          0
  X0012BX0014B  150.          0
  X0022CX0021C  5.  300.          0
  X0022AX0021A  5.  300.          0
  X0022BX0021B  5.  300.          0
  X0022CX0021C  150.          0
  X0022AX0021A  150.          0
  X0022BX0021B  150.          0
    RECVC        20.  6.E3          0
    RECVA        20.  6.E3          0
    RECVB        20.  6.E3          0
  X0014CX0017C  2.  200.          0
  X0014AX0017A  2.  200.          0
  X0014BX0017B  2.  200.          0
  SENDC XX0008  10.  3.E3          0
  SENDA XX0008  10.  3.E3          0
  SENDB XX0008  10.  3.E3          0
$INCLUDE, D:\ATPDRAW3\LCC\LIN750_2.LIB, TRAN1B, TRAN1C, TRAN1A, TRAN2B $$
, TRAN2C, TRAN2A
$INCLUDE, D:\ATPDRAW3\LCC\LIN750_1.LIB, LN1C##, LN1A##, LN1B##, TRAN1C $$
, TRAN1A, TRAN1B
$INCLUDE, D:\ATPDRAW3\LCC\LIN750_3.LIB, TRAN2A, TRAN2B, TRAN2C, TRAN3A $$
, TRAN3B, TRAN3C
$INCLUDE, D:\ATPDRAW3\LCC\LIN750_4.LIB, TRAN3C, TRAN3A, TRAN3B, RECVC# $$
, RECVA#, RECVB#
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CLOP >< type >
  RECVC SLG_A  .0285  .225  10.          0
  X0017CSENDC -1.  .075          0
  X0017ASENDA -1.  1.          0
  X0017BSENDB -1.  1.          0
  SENDC LN1C          MEASURING  1
  SENDA LN1A          MEASURING  1
  SENDB LN1B          MEASURING  1
  RECVC X0022C -1.  .075          0
  RECVA X0022A -1.  1.          0
  RECVB X0022B -1.  1.          0
/SOURCE
C < n 1><>< Ampl. >< Freq. ><Phase/T0>< A1 >< T1 >< TSTART >< TSTOP >
14X0012C 0  612300.      50.          -1.  1.
14X0012A 0  612300.      50.     -120.          -1.  1.
14X0012B 0  612300.      50.      120.          -1.  1.
14X0021C 0  612300.      50.       10.          -1.  1.
14X0021A 0  612300.      50.     -110.          -1.  1.
14X0021B 0  612300.      50.      130.          -1.  1.
```

```

/INITIAL
/OUTPUT
SENDC SENDA SENDB RECVC RECVB RECVB
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK INITIAL
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK

```

Fig. 6.5 shows the results of the simulation. The upper curve is the phase-to-ground voltage at the receiving end of the line. Following the secondary arc extinction an oscillating trapped charge appears on the faulty phase, which is the characteristics of the shunt compensated lines. The blue (lower) curve shows the line current at the faulty phase during the fault and henceforth.

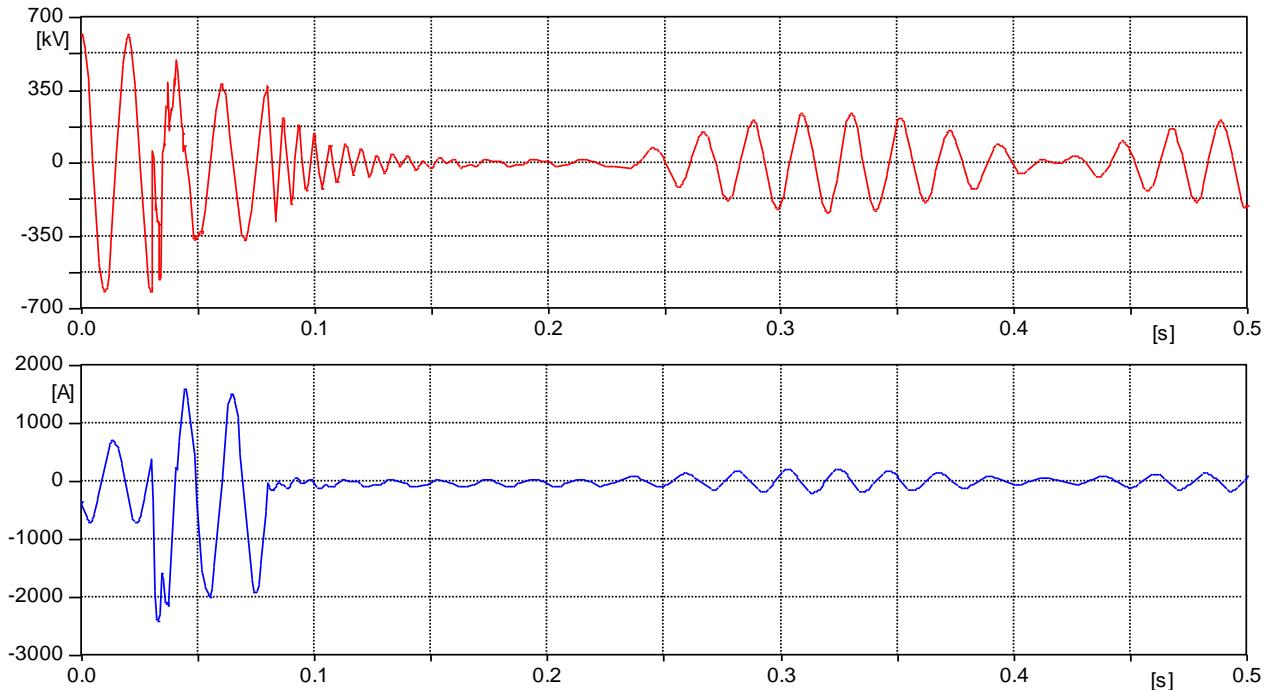


Fig. 6.5 - SLG fault and fault clearing transients (simulation).  
upper curve: phase to ground voltage, lower curve: line current

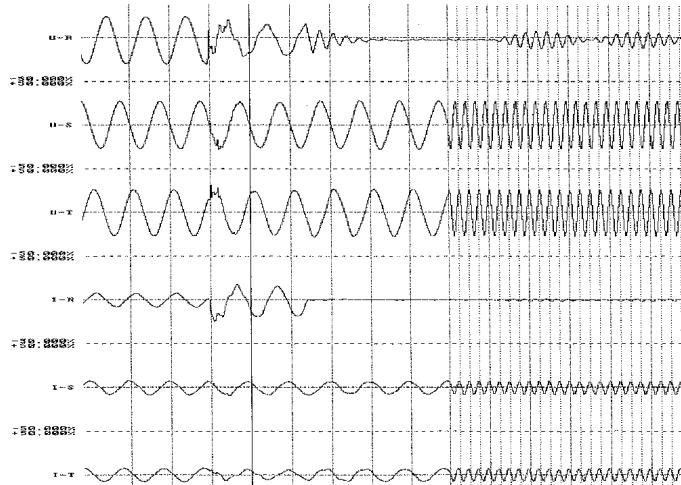


Fig. 6.6 - SLG fault and fault clearing transients. Phase currents and voltages recorded at a staged fault test by a variable sampling frequency disturbance recorder.

Fig. 6.6 shows the recorded phase voltages and line currents obtained by a high-speed transient recorder at a staged fault tests of the same 750 kV line.

## 6.2 Lightning overvoltage study in a 400 kV substation (*Exa\_9.acp*)

This example demonstrates the use of ATPDraw in a lightning protection study. The one-line diagram of the investigated 400 kV substation is drawn in

Fig. 6.7. The numbers written on the top of the bus sections specify the length in meters. The simulated incident is a single-phase back-flashover caused by a lightning strike to the tower structure 900 m away from the substation. Severe lightning parameters were chosen with 120 kA amplitude and 4/50  $\mu$ s front/tail times. In the investigated cases, only Line1 and Line2 are connected with the transformer bus. The transformer is protected by conventional SiC arresters.

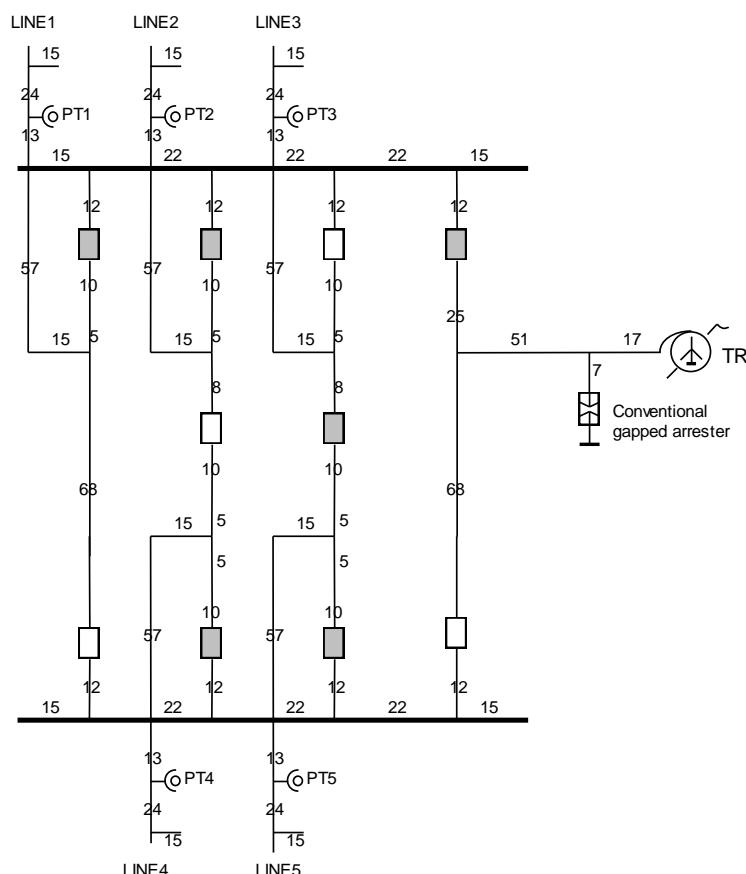


Fig. 6.7 - One-line diagram of the substation

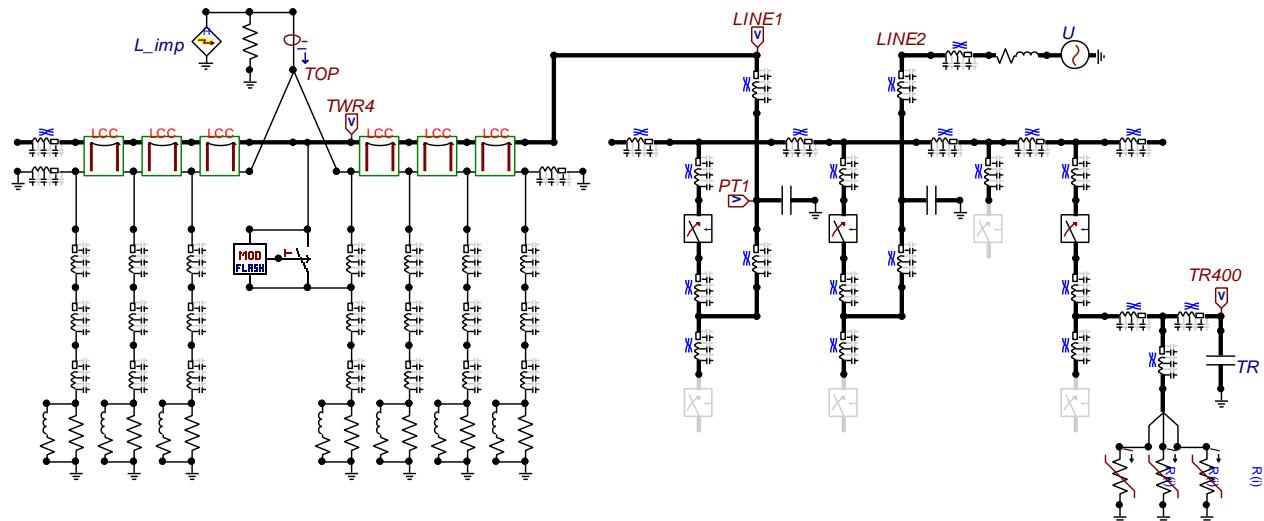


Fig. 6.8 - Example circuit (Exa\_9.acp)

The ATPDraw circuit of the complete network (substation+incoming line) is shown in Fig. 6.8. The *Copy&Paste* or *Grouping (Compress)* feature of ATPDraw could be used effectively when creating such a model because the circuit has many identical blocks. I.e. the user needs to define the object parameters only once and copy them as many times as needed.

Close to the lightning strike, the line spans are represented by 4-phase JMarti LCC objects (phase conductors + sky wire). The surge propagation along the tower structure has been taken into account in this model by representing the vertical pylon sections as single-phase constant parameter transmission lines. The R-L branches below the tower model simulate the tower grounding impedance. The front of wave flashover characteristic of the line insulators plays a significant role in such a back-flashover study. It can be simulated quite easily using a MODELS object - like the Flash of this example-, which controls a TACS/MODELS controlled switch. The influence of the power frequency voltage on the back-flashover probability can't be neglected either at this voltage level. In this study case, it was considered by a Thevenin equivalent 3-phase source connected to the remote end of Line2.

The ATP-file created by ATPDraw is shown below. Note! This case exceeds the storage cell limit of ATP if the program runs with DEFAULT=3.0 table size (default LISTSIZE.DAT setting). To run the simulation successfully the user must increase this limit from 3.0 to 6.0.

```

BEGIN NEW DATA CASE
C -----
C Generated by ATPDRAW August, Monday 26, 2019
C A Bonneville Power Administration program
C by H. K. Høidalen at NTNU/SEFAS - NORWAY 1994-2019
C -----
$DUMMY, XYZ000
C dT >< Tmax >< Xopt >< Copt >
  5.E-9  2.5E-5
      500      3       0       0       1       0       0       1       0
MODELS
/MODELS
INPUT
  IX0001 {v(TWR4A ) }
  IX0002 {v(XX0016) }
OUTPUT
  XX0048
MODEL Flash
comment-----

```

```

| Front of wave flashover characteristic      |
| of the HV insulator.                      |
| Input: Voltage across the insulator.      |
| Output: Close command for the TACS switch |
-----endcomment
INPUT UP, UN
OUTPUT CLOSE
DATA UINF {DFLT:650e3}, UO {DFLT: 1650e3}, TAU {DFLT:8.e-7}, UINIT {DFLT:1E5}
VAR CLOSE, TT, U, FLASH
INIT
  CLOSE:=0
  TT:=0
  FLASH:=INF
ENDINIT
EXEC
  U:= ABS(UP-UN)
  IF (U>UINIT) THEN
    TT:=TT+timestep
    FLASH:=(UINF + (UO-UINF) * (EXP(-TT/TAU)) )
    IF (U>FLASH) THEN CLOSE:=1 ENDIF
  ENDIF
ENDEXEC
ENDMODEL
USE FLASH AS FLASH
INPUT
  UP:= IX0001
  UN:= IX0002
DATA
  UINF:=     1.4E6
  UO:=       3.E6
  TAU:=      8.E-7
  UINIT:=    3.5E5
OUTPUT
  XX0048:=CLOSE
ENDUSE
RECORD
  FLASH.U AS U
  FLASH.CLOSE AS CLOSE
ENDMODELS
C   1          2          3          4          5          6          7          8
C 34567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R >< L >< C >
C < n 1>< n 2><ref1><ref2>< R >< A >< B ><Leng><><>0
-1XX0010XX0167      10. 200. 2.5E5 .008 1 0          0
-1XX0012XX0010      10. 200. 2.5E5 .007 1 0          0
-1XX0014XX0012      10. 200. 2.5E5 .018 1 0          0
-1XX0016TOP         10. 200. 2.5E5 .008 1 0          0
-1      XX0019        20. 600. 2.9E5 .3 1 0          0
-1XX0020XX0016      10. 200. 2.5E5 .007 1 0          0
  XX0014            40.                         0
  XX0014            13. .005                     0
-1XX0026XX0171      10. 200. 2.5E5 .008 1 0          0
-1XX0028XX0020      10. 200. 2.5E5 .018 1 0          0
-1X0032AX0033A      20. 650. 2.4E5 3. 1 0          0
-2X0032BX0033B      2. 400. 2.9E5 3. 1 0          0
-3X0032CX0033C      XX0028          40.                         0
  XX0028            20. 600. 2.9E5 .3 1 0          0
  XX0028            13. .005                     0
-1XX0040XX0179      10. 200. 2.5E5 .008 1 0          0
-1XX0042XX0040      10. 200. 2.5E5 .007 1 0          0
-1XX0044XX0042      10. 200. 2.5E5 .018 1 0          0
  XX0044            40.                         0
  XX0044            13. .005                     0
-1XX0054XX0183      10. 200. 2.5E5 .008 1 0          0
-1XX0056XX0026      10. 200. 2.5E5 .007 1 0          0
  LIGHT             400.                         0
-1XX0060XX0054      10. 200. 2.5E5 .007 1 0          0
-1XX0062XX0056      10. 200. 2.5E5 .018 1 0          0
-1XX0064XX0060      10. 200. 2.5E5 .018 1 0          0
  XX0064            40.                         0

```

-1XX0069XX0019	10.	200.	2.5E5	.008	1	0	0
XX0064	13.	.005					0
-1X0073AX0074A	20.	400.	2.4E5	.008	1	0	0
-2X0073BX0074B	2.	260.	2.9E5	.008	1	0	0
-3X0073CX0074C							0
-1XX0075XX0036	10.	200.	2.5E5	.008	1	0	0
-1X0078AX0211A	20.	400.	2.4E5	.012	1	0	0
-2X0078BX0211B	2.	260.	2.9E5	.012	1	0	0
-3X0078CX0211C							0
-1X0257AX0081A	50.	650.	2.4E5	.015	1	0	0
-2X0257BX0081B	10.	360.	2.9E5	.015	1	0	0
-3X0257CX0081C							0
-1X0082AX0083A	20.	400.	2.4E5	.068	1	0	0
-2X0082BX0083B	2.	260.	2.9E5	.068	1	0	0
-3X0082CX0083C							0
-1X0271ALINE2A	20.	650.	2.4E5	.024	1	0	0
-2X0271BLINE2B	2.	360.	2.9E5	.024	1	0	0
-3X0271CLINE2C							0
-1X0086AX0269A	20.	400.	2.4E5	.012	1	0	0
-2X0086BX0269B	2.	260.	2.9E5	.012	1	0	0
-3X0086CX0269C							0
-1X0088AX0293A	20.	650.	2.4E5	.015	1	0	0
-2X0088BX0293B	2.	360.	2.9E5	.015	1	0	0
-3X0088CX0293C							0
-1X0074AX0090A	20.	400.	2.4E5	.015	1	0	0
-2X0074BX0090B	2.	260.	2.9E5	.015	1	0	0
-3X0074CX0090C							0
-1X0074AX0271A	20.	400.	2.4E5	.085	1	0	0
-2X0074BX0271B	2.	260.	2.9E5	.085	1	0	0
-3X0074CX0271C							0
X0271A			.0005				0
X0271B			.0005				0
X0271C			.0005				0
-1X0269AX0211A	20.	650.	2.4E5	.022	1	0	0
-2X0269BX0211B	2.	360.	2.9E5	.022	1	0	0
-3X0269CX0211C							0
-1X0211AX0257A	20.	650.	2.4E5	.022	1	0	0
-2X0211BX0257B	2.	360.	2.9E5	.022	1	0	0
-3X0211CX0257C							0
99SICC	1.1E6	1.					1
100.	6.5E5						
1.E3	7.6E5						
2.E3	8.E5						
4.E3	8.34E5						
5.E3	8.5E5						
1.E4	9.35E5						
2.E4	1.082E6						
3.E4	1.2E6						
9999							
-1X0104AX0105A	20.	400.	2.4E5	.068	1	0	0
-2X0104BX0105B	2.	260.	2.9E5	.068	1	0	0
-3X0104CX0105C							0
-1X0106AX0257A	20.	400.	2.4E5	.012	1	0	0
-2X0106BX0257B	2.	260.	2.9E5	.012	1	0	0
-3X0106CX0257C							0
-1X0108ATR400A	20.	650.	2.4E5	.017	1	0	0
-2X0108BTR400B	2.	360.	2.9E5	.017	1	0	0
-3X0108CTR400C							0
-1X0105AX0110A	20.	400.	2.4E5	.025	1	0	0
-2X0105BX0110B	2.	260.	2.9E5	.025	1	0	0
-3X0105CX0110C							0
99SICB	1.1E6	1.					1
100.	6.5E5						
1.E3	7.6E5						
2.E3	8.E5						
4.E3	8.34E5						
5.E3	8.5E5						
1.E4	9.35E5						
2.E4	1.082E6						
3.E4	1.2E6						
9999							
-1PT1A LINE1A	20.	650.	2.4E5	.024	1	0	0

```

-2PT1B LINE1B          2.  360. 2.9E5 .024 1 0          0
-3PT1C LINE1C          20. 400. 2.4E5 .012 1 0          0
-1X0118AX0293A         20. 400. 2.4E5 .012 1 0          0
-2X0118BX0293B         2.  260. 2.9E5 .012 1 0          0
-3X0118CX0293C         20. 400. 2.4E5 .015 1 0          0
-1X0083AX0120A         20. 400. 2.4E5 .015 1 0          0
-2X0083BX0120B         2.  260. 2.9E5 .015 1 0          0
-3X0083CX0120C         20. 400. 2.4E5 .015 1 0          0
    TR400A               .003          0
    TR400B               .003          0
    TR400C               .003          0
-1X0105AX0108A         20. 650. 2.4E5 .051 1 0          0
-2X0105BX0108B         2.  360. 2.9E5 .051 1 0          0
-3X0105CX0108C         20. 400. 2.4E5 .007 1 0          0
-1SICA X0108A          20. 400. 2.4E5 .007 1 0          0
-2SICB X0108B          2.  260. 2.9E5 .007 1 0          0
-3SICC X0108C          20. 400. 2.4E5 .007 1 0          0
99SICA                 1.1E6   1.          1
    100.                6.5E5          0
    1.E3                7.6E5          0
    2.E3                8.E5           0
    4.E3                8.34E5         0
    5.E3                8.5E5           0
    1.E4                9.35E5         0
    2.E4                1.082E6        0
    3.E4                1.2E6           0
    9999
X0132AX0133A          1.  50.          0
X0132BX0133B          1.  50.          0
X0132CX0133C          1.  50.          0
-1XX0135XX0075         10. 200. 2.5E5 .007 1 0          0
-1X0083APT1A          20. 400. 2.4E5 .085 1 0          0
-2X0083BPT1B          2.  260. 2.9E5 .085 1 0          0
-3X0083CPT1C          20. 400. 2.4E5 .085 1 0          0
    PT1A               .0005          0
    PT1B               .0005          0
    PT1C               .0005          0
-1X0293AX0269A         20. 650. 2.4E5 .022 1 0          0
-2X0293BX0269B         2.  360. 2.9E5 .022 1 0          0
-3X0293CX0269C         20. 650. 2.4E5 .022 1 0          0
-1XX0143XX0135         10. 200. 2.5E5 .018 1 0          0
    XX0062             40.            0
    XX0062             13. .005          0
-1XX0149XX0069         10. 200. 2.5E5 .007 1 0          0
-1XX0151XX0149         10. 200. 2.5E5 .018 1 0          0
    XX0151             40.            0
    XX0151             13. .005          0
    XX0143             40.            0
    XX0143             13. .005          0
-1LINE2AX0132A         20. 650. 2.4E5   3. 1 0          0
-2LINE2BX0132B         2.  360. 2.9E5   3. 1 0          0
-3LINE2CX0132C         20. 650. 2.4E5   3. 1 0          0
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, X0033A, X0033B, X0033C, XX0019, X0166A $$
, X0166B, X0166C, XX0167
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, X0166A, X0166B, X0166C, XX0167, X0170A $$
, X0170B, X0170C, XX0171
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, X0170A, X0170B, X0170C, XX0171, TWR4A# $$
, TWR4B#, TWR4C#, TOP###
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, TWR4A#, TWR4B#, TWR4C#, TOP###, X0178A $$
, X0178B, X0178C, XX0179
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, X0178A, X0178B, X0178C, XX0179, X0182A $$
, X0182B, X0182C, XX0183
$INCLUDE, D:\ATPDRAW\LCC\EXA_9.LIB, X0182A, X0182B, X0182C, XX0183, LINE1A $$
, LINE1B, LINE1C, XX0036
/ SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CLOP >< type >
    LIGHT TOP                                     MEASURING      1
    X0090AX0086A     -1.   1.001          0
    X0090BX0086B     -1.   1.001          0
    X0090CX0086C     -1.   1.001          0
    X0110AX0106A     -1.   1.001          0
    X0110BX0106B     -1.   1.001          0

```

```

X0110CX0106C      -1.      1.001          0
X0120AX0118A      -1.      1.001          0
X0120BX0118B      -1.      1.001          0
X0120CX0118C      -1.      1.001          0
13XX0016TWR4A           XX0048   0
/SOURCE
C < n><>< Ampl. >< Freq. ><Phase/T0>< A1 >< T1 >< TSTART >< TSTOP >
15LIGHT -1      1.2E5    4.E-6    5.E-5      5.          1.
14X0133A 0     -3.3E5    50.          -1.          1.
14X0133B 0     -3.3E5    50.     -120.        -1.          1.
14X0133C 0     -3.3E5    50.      120.        -1.          1.
/INITIAL
/OUTPUT
LINE1ALINE1BLINE1CTWR4A TWR4B TWR4C TR400ATR400BTR400CPT1A PT1B PT1C
BLANK MODELS
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK INITIAL
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK

```

Some results of the simulation are drawn in Fig. 6.9. The blue line is the voltage stress appearing at the transformer terminal, the red line shows the incoming surge measured at the voltage transformer of Line1 (node PT1 of the circuit). The discharge current of the gapped arrester is drawn at the bottom if the figure. As it can be seen, the instantaneous value of the power frequency voltage was set opposite to the polarity of the lightning surge in the simulation.

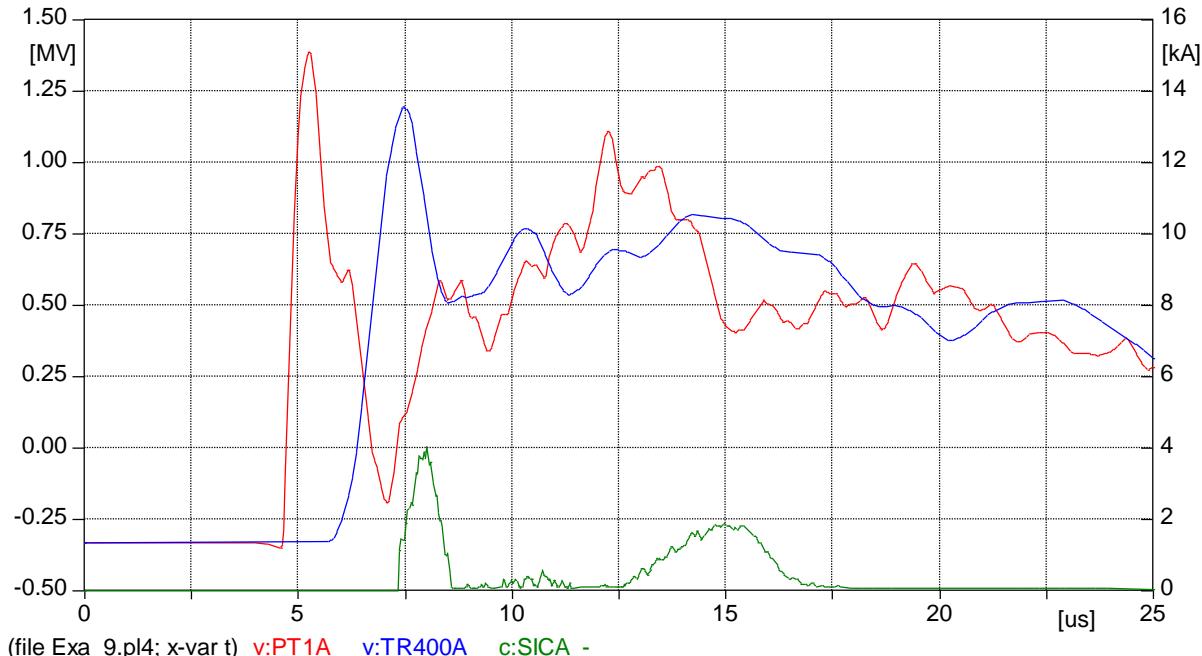


Fig. 6.9 - ATP simulation results. Red: incoming surge at the substation entrance. Blue: voltage stress at the transformer terminal. Green: arrester discharge current.

### 6.3 Modeling Rectifiers, zigzag transformers and analysis of Harmonics (Exa\_14.acp)

In section 5.8.1 of the Advanced Manual, it is shown how to create a 6-pulse controlled thyristor-rectifier bridge and make it available in ATPDraw as a user specified single object. In this part of the manual a diode rectifier will be used instead, and the focus shifted to harmonics in the supplying line currents. The case is an industrial plant consisting of AC/DC converters and consuming 55 MW for aluminum production. The plant is supplied by a 132 kV high voltage AC

system and there are concerns about the harmonics in the current on the high voltage side. This example shows how to model an equivalent 24 pulse diode rectifier and calculate the harmonics in currents in Models. The harmonics could alternatively have been calculated as a part of a post-processing. Fig. 6.10 shows the example circuit.

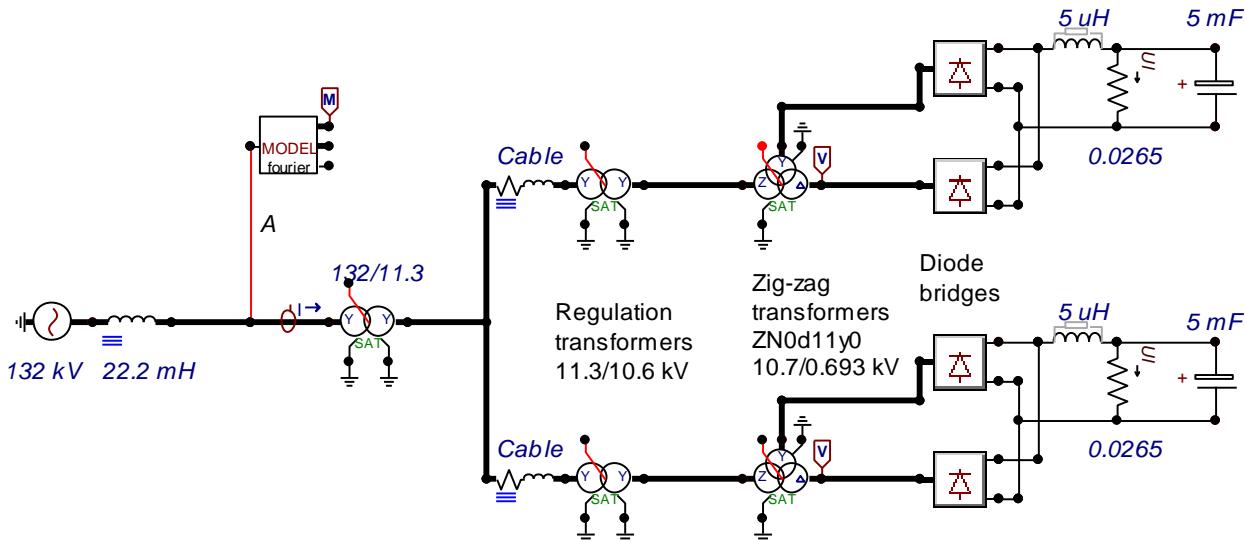


Fig. 6.10 – Example circuit (Exa\_14.acp).

The diode bridge is modeled and compressed into a group as shown in Fig. 6.11. Note the need for small resistors ( $1 \mu\Omega$ ) to decouple the diodes and added snubber circuits. The R and C data for all six snubbers are added to the External parameter group but will appear as only two parameters in the compressed object. A bitmap icon is created for diode bridge.

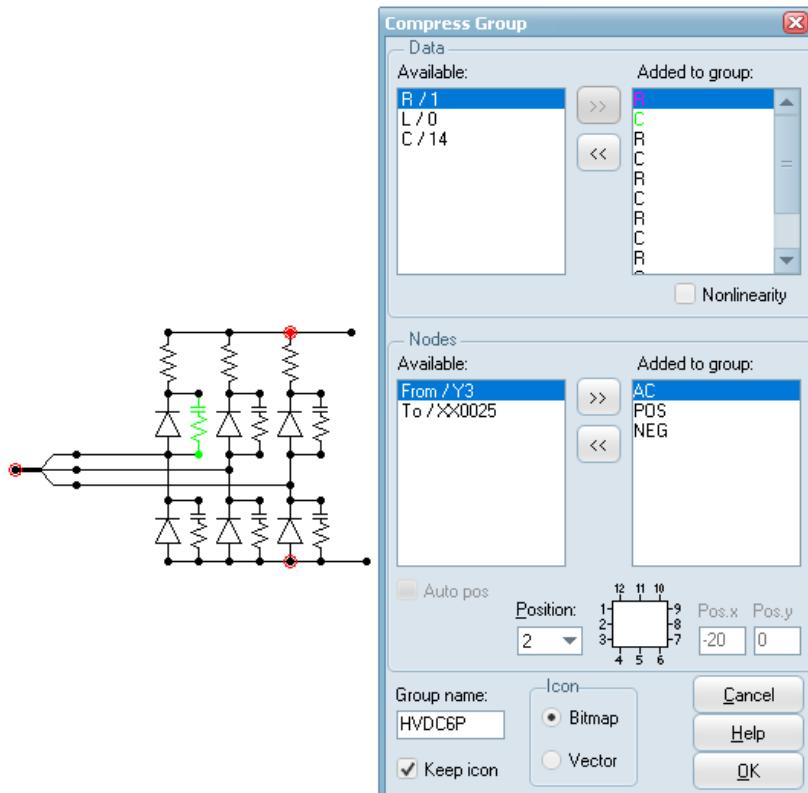


Fig. 6.11 – Compress a 3-phase diode bridge.

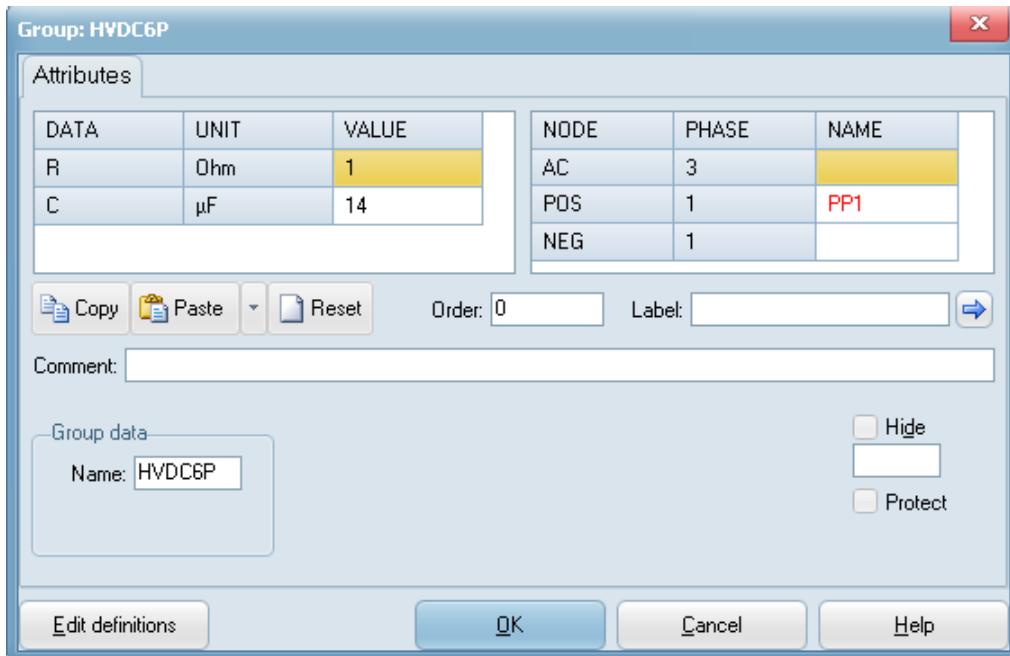


Fig. 6.12 – Component dialog of the compressed group ACDC.

The key unit to produce the 24-pulse system are the two supplying transformers phase shifted 15 degrees and with a Y and Δ coupling on the secondary side. This is accomplished by using the Saturable Transformer component with a zigzag coupling on the primary winding. The input dialog of the upper transformer is shown in Fig. 6.12. The Saturable Transformer requires direct input of electrical quantities, so recalculation of Test Report data is required. The transformers had the following test report data:

Coupling:	ZN0d11y
Rated power:	24.8 MVA
Rated primary voltage:	10.735 kV
Rated secondary voltage:	693 V
Rated tertiary voltage:	693 V
Rated frequency:	50 Hz
Open circuit current:	0.0056 pu
Short circuit impedance 1-2:	0.0084 + j0.1015 pu
Short circuit impedance 1-3:	0.0084 + j0.1015 pu
Short circuit impedance 2-3:	0.0210 + j0.1887 pu
Phase shift Z (ref. 3):	±7.5 deg.

This will result in the standard per unit equivalent circuit for the short circuit impedances

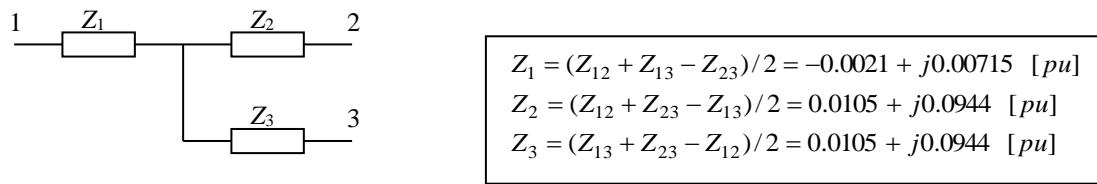


Fig. 6.13 – Per unit equivalent circuit of the 3-winding transformer.

Note the negative resistance in the primary winding. This could result in a stability problem in the simulations, but fortunately this didn't seem to be the case in this example. The input dialog of the Saturable transformer with the electrical parameters is shown in Fig. 6.14.

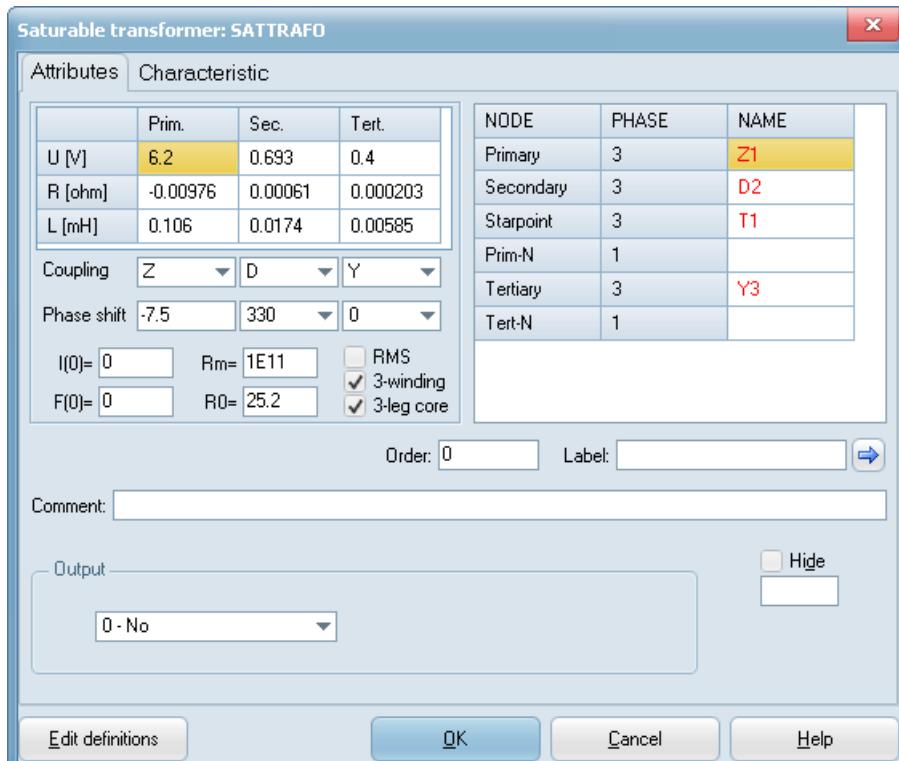


Fig. 6.14 – Component dialog of the Saturable Transformer component.

The total winding voltage is  $U_A = 10.735 / \sqrt{3} \text{ kV} = 6.2 \text{ kV}$

The short circuit impedance is

$$Z_1 = (-0.0021 + j0.00715) \cdot (10.735 \text{ kV})^2 / 24.8 \text{ MVA} = -0.00976 + j0.0332 [\Omega]$$

The zigzag winding 1 is further split in Z and Y parts with  $n = \frac{\sin(7.5^\circ)}{\sin(60 - 7.5^\circ)} = 0.165$ .

The voltages across each winding part and the individual leakage impedances are automatically calculated by ATPDraw as:

$$U_{1z} = \frac{10.735 / \sqrt{3}}{\cos(7.5^\circ) + 0.165 \cdot \cos(60^\circ - 7.5^\circ)} \text{ kV} = 5.68 \text{ kV}$$

$$U_{1y} = 5.68 \text{ kV} \cdot 0.165 = 0.934 \text{ kV}$$

$$R_{1z} = -0.00976 \cdot \frac{1}{1 + 0.165} [\Omega] = -8.4 [\text{m}\Omega]$$

$$R_{1y} = -0.00976 \cdot \frac{0.165}{1 + 0.165} [\Omega] = -1.4 [\text{m}\Omega]$$

$$L_{1z} = \frac{0.0332}{2\pi \cdot 50} \cdot \frac{1}{1 + 0.165^2} [\text{H}] = 0.103 [\text{mH}]$$

$$L_{1y} = \frac{0.0332}{2\pi \cdot 50} \cdot \frac{0.165^2}{1 + 0.165^2} [\text{H}] = 2.79 [\mu\text{H}]$$

If the HV winding 1 is chosen as the primary winding, the magnetizing branch will be added to the first winding part (Z) of the zigzag winding. This is probably not a good choice, and alternatively the magnetizing branch should be added to the low-voltage Y-coupled winding. This could be done externally or by choosing winding 3 as the primary.

The measured inductance is

$$L_m = \frac{1/\sqrt{3}}{2\pi \cdot 50 \cdot 0.0056} \text{ pu} = 0.328 \text{ pu} = 0.328 \cdot (10.735 \text{ kV})^2 / 24.8 \text{ MVA} = 1.52 \text{ [H]}$$

and the inductance that should be added to winding 1Z in ATP:

$$L_{mz}^{ATP} = \frac{L_m}{1+n+n^2} = 1.28 \text{ [H]}$$

Saturation is of no importance in this example and a single point is set on the characteristic page  $(i, \lambda) = (1, 1.28)$ .

If a measurement of the zero sequence impedance is missing a reasonable assumption for this particular transformer is to set it to 2/3 of the positive sequence magnetizing current. Further, the zero sequence inductance added in ATP is one half of the real value. This gives

$$R_0 = \frac{U_{z0}^2}{3 \cdot L_{0z}^{ATP}} \approx 2 \cdot \frac{5.68^2}{2 \cdot L_{mz}^{ATP}} = \frac{5.68^2}{1.28} = 25.2 [\Omega]$$

The Delta- winding:

The total winding voltage is  $U_{A2} = 0.693 \text{ kV}$

The short circuit impedance is

$$Z_2 = (0.0105 + j0.0944) \cdot (\sqrt{3} \cdot 0.693 \text{ kV})^2 / 24.8 \text{ MVA} = 0.61 + j5.48 \text{ [m}\Omega\text{]}$$

$$R_2 = 0.61 \text{ [m}\Omega\text{]} \text{ and } L_2 = 17.5 \text{ [\mu H]}$$

The Wye- winding:

The total winding voltage is  $U_{A3} = 0.693 / \sqrt{3} = 0.4 \text{ kV}$

The short circuit impedance is

$$Z_3 = (0.0105 + j0.0944) \cdot (\sqrt{3} \cdot 0.4 \text{ kV})^2 / 24.8 \text{ MVA} = 0.203 + j1.83 \text{ [m}\Omega\text{]}$$

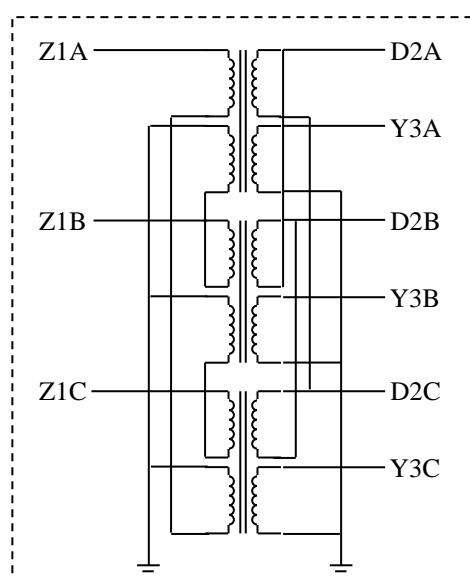
$$R_3 = 0.203 \text{ [m}\Omega\text{]} \text{ and } L_3 = 5.85 \text{ [\mu H]}$$

The ATP file format and connectivity of the transformer specified in is:

```

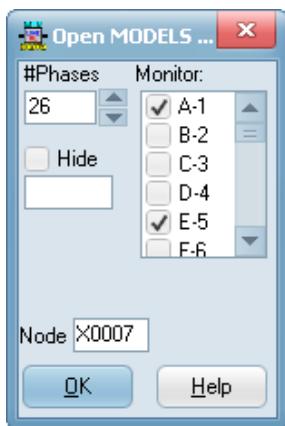
TRANSFORMER THREE PHASE TX0001 25.2
TRANSFORMER
1. 1.28
9999
1Z1A T0002C -.0084.103215.6797
2 T0002A -.0014.00279.93446
3D2A D2C .00061 .0174 .693
4Y3A .0002.00585 .4
TRANSFORMER T1A
1Z1B T0002A
2 T0002B
3D2B D2A
4Y3B
TRANSFORMER T1A
1Z1C T0002B
2 T0002C
3D2C D2B
4Y3C

```



The example shown in Fig. 6.10 also includes a stepdown transformer and regulating transformer (regulation not modeled) that also are modeled as Saturable Transformer components. Alternatively, the BCTRAN or Hybrid Transformer models could have been used as they have an internal conversion of test report data. These models do not support Zigzag transformers, however.

The harmonics are calculated by an algorithm in MODELS. This is shown in chapter 5.5.1 in this manual. The automatic approach is assumed. A default model is used and the Models text is typed in under Edit. The output of absolute value and angle are declared as 26-phase (ABSF and ANGF) while the input X is single phase. The user can select the type of input (switch current in this case) by clicking on the left input node of the model and select *Input Current* in the Node dialog box. The Model will output all harmonics 0..N (where N is a data parameter) as a function of time. The calculation is performed by integration of a sliding window of size 1/FREQ [sec]. The selection of variables to plot is made from a models probe connected to the ABSF node.



The probe is set to 26-phases and the phases of special interest 1, 5, 7, 11, 13, 23, 25 are checked under Monitor.

Fig. 6.15 – Model probe dialog.

The line current in phase A at the 132 kV side is selected as input. A connection is drawn from the left 3-phase side of the switch to the single phase Model input node. In the Connection dialog that then pops up phase A is selected. The simulated phase A current is shown in Fig. 6.16 and the 5<sup>th</sup>, 7<sup>th</sup>, 23<sup>rd</sup> and 25<sup>th</sup> harmonics calculated in Models shown in Fig. 6.17.

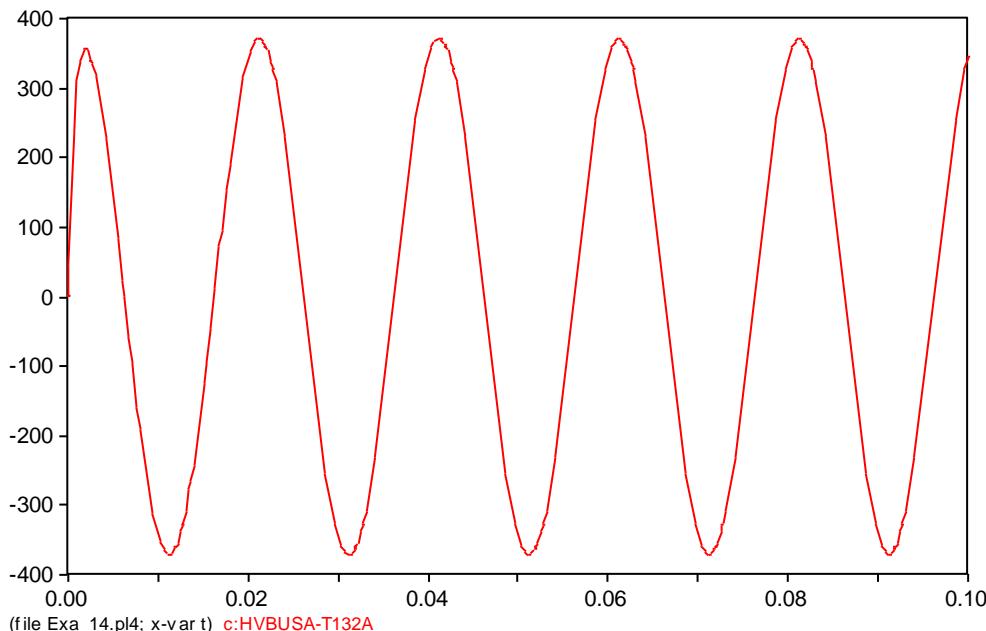


Fig. 6.16 – Simulated line current phase A at the 132 kV side.

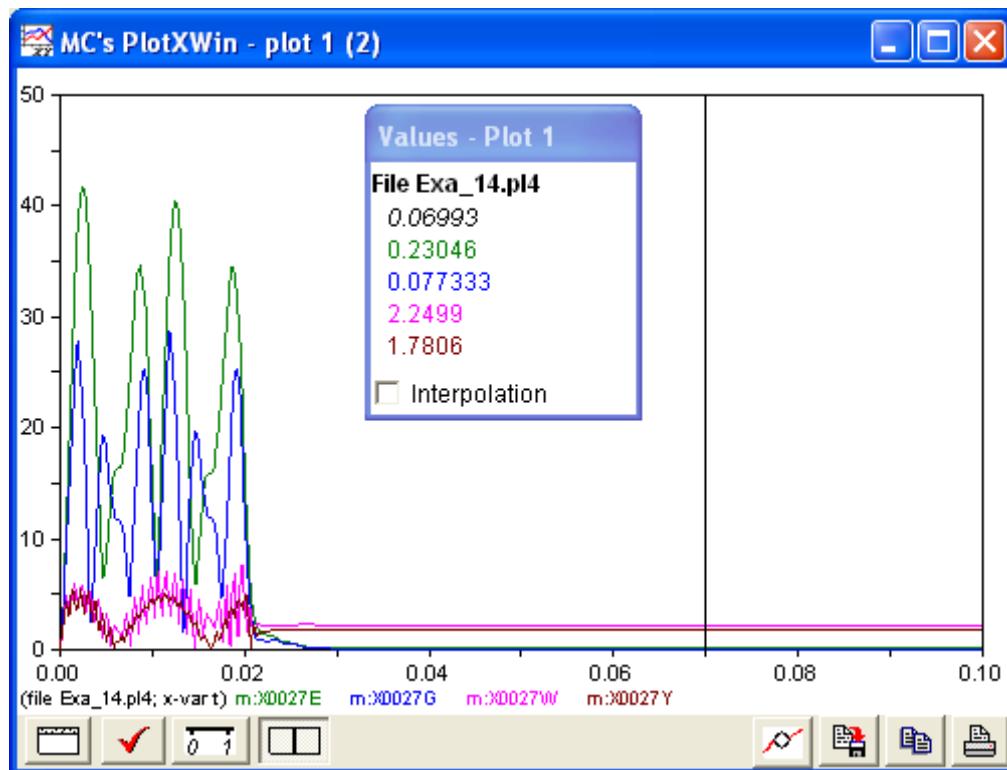


Fig. 6.17 – The harmonics of the current in Fig. 6.16.

The harmonics can also be calculated in for instance PlotXY as shown in Fig. 6.18, but not as a function of time.

#### MC's PlotXY - Fourier chart(s). Copying date: 28.01.2009

File Exa\_14.pl4 Variable c:HVBUSA-T132A [peak]  
Initial Time: 0.08 Final Time: 0.1

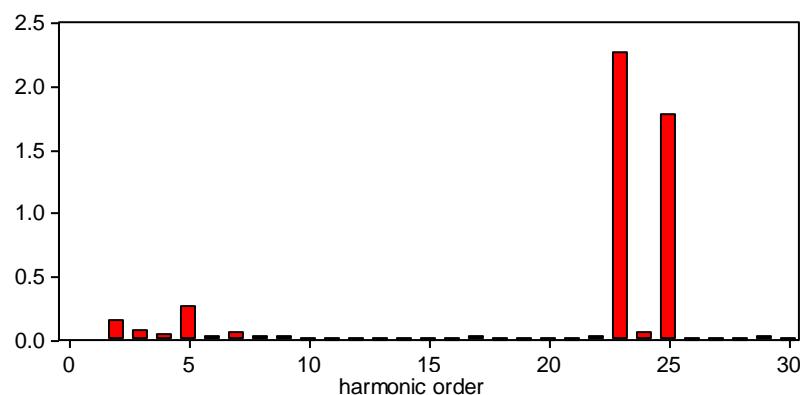


Fig. 6.18 – Harmonics calculated by FFT in PlotXY.

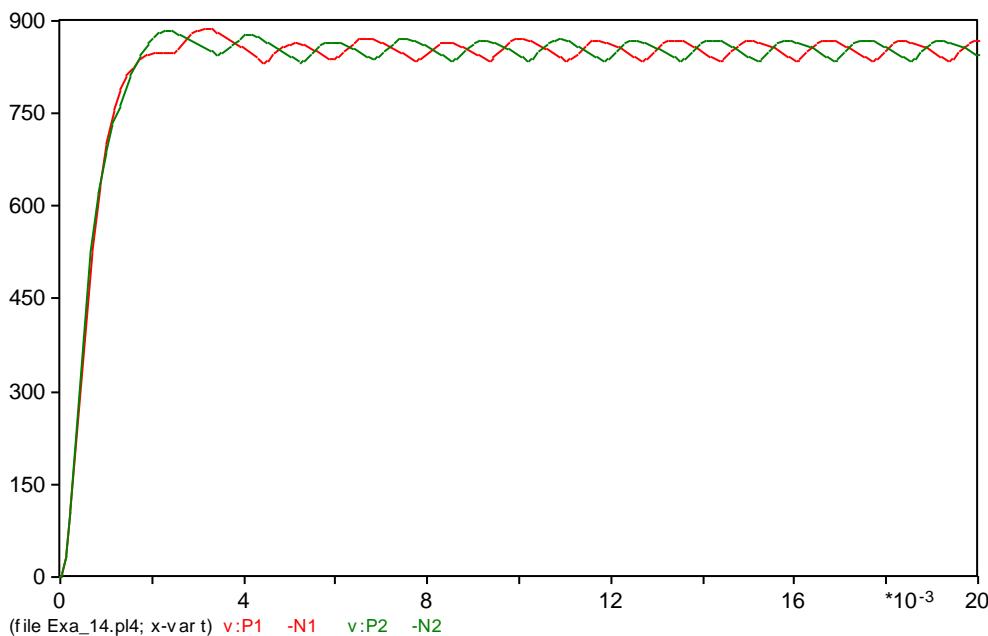


Fig. 6.19 – DC voltages on the LV side.

## 6.4 The Controlled Electric Rotating Machines

Power system studies often require the simulation of transients operations of 3-phase electrical rotating machines, including their control schemes, with cases like synchronous generator's stability after external faults, induction motor stalling due to voltage dips, power quality issues due to the aeolic generators dynamics, etc.. This section illustrates on how to represent these machines (synchronous, induction and double-fed mainly) and their controls in order to properly reproduce the group behavior in realistic cases.

Electric Rotating machinery representations for simulation of transients include both, electromagnetic models for the windings and magnetic core and, mechanical models for the rotating body, so; their input/output possibilities include electromagnetic magnitudes like voltages, currents and fluxes along with mechanical magnitudes like torques, angles and speeds.

Regarding the control schemes and due to their extensive functions/topologies, neither ATPDraw nor ATP-EMTP offers specific single objects that represent control schemes of the rotating machines, instead; with the TACS or MODELS object(s), the user may assemble or “code” the particular control arrangement.

Due to the above and as mentioned in the ATPDraw Reference Manual chapter 4.11.7, ATPDraw offers three classes of rotating machines objects:

- Objects that include rotating machines with both, the electromagnetic and the mechanical models: SM (ATP/EMTP source 59/58, multi-mass synchronous machine) and IM56A (ATP/EMTP source 56, single-mass induction machine). The input data for these objects are the nameplate/manufacturer data, so: they can be readily incorporated in a simulation case. The user must add the proper control schemes for the power/frequency and voltage/reactive power controls for the SM object and, the mechanical load/prime mover scheme for the IM56A object.
- Objects that include the universal machine (UM) with only the electromagnetic modelling: UM\_1 (ATP/EMTP source type 19 code 1, synchronous machine), UM\_3 (ATP/EMTP

source type 19 code 3, single-fed induction machine), UM\_4 (ATP/EMTP source type 19 code 4, double-fed induction machine). The input data for these objects are the electrical parameters' values that the user must obtain from the nameplate/manufacturer data. Also, the user must add the analog electric network for the mechanical rotating body representation and, the proper power/frequency and voltage/reactive power control schemes, too.

- Objects that include the universal machine (ATP/EMTP source type 19) with the electromagnetic modelling and a single-mass rotor (electric analogy), with built-in measurements and some optional controllers in TACS: UMSYN (ATP/EMTP source type 19 code 1, synchronous machine), UMIND (ATP/EMTP source type 19 codes 3 and 4, single-fed and double-fed induction machines). The input data for these objects are mainly the nameplate/manufacturer data; with them, ATPDraw obtains the electrical parameters' values required by the ATP/EMTP sources. Also, in case that the represented unit have a different control scheme, the user can inhibit the built-in controls and, to include the proper power/frequency and voltage/reactive power control schemes.

In order to have the case with the machines –controlled or not- at the desired steady-state conditions, the case must be solved with a few seconds in TMAX, without intentional transients. Then, the user must verify that the initial operational magnitudes of the machines (voltages, angles, powers, torques, etc.), have the correct values. For this purpose, the analysis of the solution text output file (LIS) is highly recommended, too.

Note that for the UM variants (codes 1, 3, 4) the control input magnitudes to the ATPDraw objects are applied in physical units and incremental regarding the initial values; currents in Ampere, as analogy of mechanical torques ( $1\text{ A} = 1\text{ N.m}$ ), are applied to the masses' analog RC circuits and; in the case of code 1 (synchronous machine), the DC excitation voltage in Volt, is applied to the field winding (first winding in rotor's data of UM) first terminal, while the other is 'grounded'. This is explained in the "Help" text at every ATPDraw's object and must be taken into account when the user set the required scaling and limits for the output of the machines' control schemes.

For the SM and IM56A objects, the control input magnitudes are applied in PU units of the initial values of the torques and the DC excitation voltage then, it is convenient that a previous steady-state solution be analyzed before the main case simulation is done:

- a) Solve the steady-state of the case until the initial values of voltages, angles, powers, etc. of the simulated network (machines and other elements) met the desired conditions;
- b) Analyze the steady-state solution text output (LIS file) for each SM and IM56 machine and note the reported values for mechanical and electromagnetic torques and the DC excitation voltage at the field winding terminals. For the SM 59/58 machines, these values are under their "Data Parameter and initial conditions.." section; for the IM56A machines, their torque values are at their "The initial torque calculation for a Type-56 IM at bus "X00YYA" .." report.
- c) With the initial values as bases for the PU magnitudes and in order to obtain a proper controlled machines response, the user must obtain and set the required scaling and limits for the machines' control schemes.

#### **6.4.1 Synchronous machine control (*Exa\_22a.acp*)**

The ATP/EMTP sources types 59 (solved in the d-q-0 reference frame) and 58 (solved in the ABC reference frame), whose rules of application and input data formats are described in the

ATP/EMTP Rule Book chapter VIII and their formulation is illustrated in the EMTP Theory Book chapter 8, are preprocessed with the ATPDraw's object SM that properly considers these features/characteristics:

- A three-phase stator winding, with possible delta or wye (star) internal connection, with only a three-phase terminal for connection to the rest of the power system representation. In case of wye-connected machines, the user has no access to the neutral terminal, but he/she can specify an RL grounding impedance –possible zero- in PU of the machine MVA and kV ratings;
- Up to four rotor windings, field and damper in the d-axis and damper and parasitic –eddy currents- in the q-axis-; even when the rotor has no electrical network connection nodes, the user can specify the DC voltage –in PU of the steady-state value- to apply to the field winding;
- Field current at rated voltage in the air-gap line, with optional saturation characteristics in the rotor's direct and quadrature axis. Only two points (S1 at rated voltage and S2 > S1) are required for each characteristic.
- A multi-mass rotating body, with masses specified with constant moment of inertia and viscous damping coefficient and, shaft sections specified only with the stiffness constant; individual mechanical torque -in PU of the steady-state value- can be applied at any mass; the electromagnetic torques –internal- application is done to the mass designated as the generator rotor and to the exciter rotor if any;

Then, for the graphic representation and pre-processing of the source types 59/58, the ATPDraw's object named SM has this appearance and node description:



Fig. 6.20 – Graphic appearance and connection nodes of the ATPDraw object SM.

**BUS** – a three-phase node, for the stator winding connection to the rest of the power system representation;

**EXFD** – a single-phase node, for the DC voltage –in PU of the value at the steady-state solution-, control input to the rotor's field winding of the from the connected TACS or MODELS output variable;

**POWER** – a multiphase node, for the mechanical torques –in PU of the value at the steady-state solution-, control input to every specified turbine's mass from the connected TACS or MODELS output variables;

**EXOUT** – up to five outputs for ‘passing’ to TACS or MODELS objects, the values of the specified internal machine’s electrical and mechanical magnitudes, in engineering units: stator and rotor currents, EM torques, masses speeds and angles, shaft torques, etc.

These machine nodes can be named and the machine magnitude for EXOUT nodes can be selected as shown:

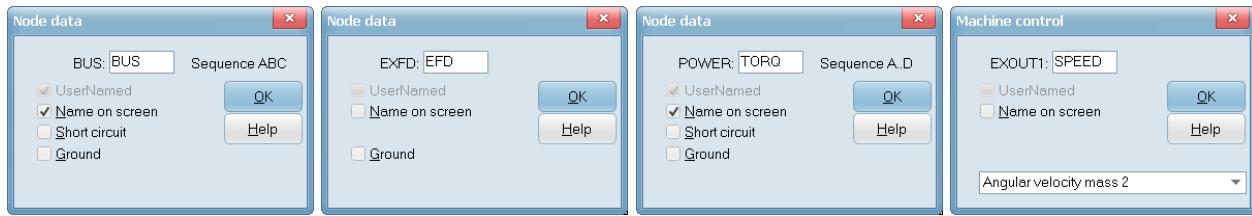


Fig. 6.21 – SM Connection Nodes Data.

For the usage of a machine magnitude in a TACS process, an ATPDraw's EMTP\_OUT Circuit Variable -Internal Variable- (ATP/EMTP TACS signal source type 92) object must be connected to the EXOUT node that have the selected electromagnetic or mechanical magnitude. The user must be aware that in the transient simulation, the initial (at  $\text{TIME} = 0$  seconds) value of the signal source is zero, because TACS solve its "circuit" before the electrical network and MODELS objects are solved in the time loop calculation.

For the usage of a machine magnitude in a MODELS object, its input node type must be selected as "Input Machine" and connected to the EXOUT node that have the selected electromagnetic or mechanical magnitude, but; if a TACS signal source is also connected to that EXOUT node, the MODELS input node must be declared as "Input TACS", otherwise the values that the MODELS object receives will be zero at all the simulation steps, without any warning from ATP/EMTP.

On other hand, the object SM accepts its data input only in the "manufacturers" format described in the ATP/EMTP Rule Book chapter VIII, the same for both source types 58 and 59. The following figure shows the data sections of this format: Electrical Attributes (rated values, reactances,...), General data (steady AC, source 58/59,...), Field current (saturation characteristic,...), Masses (number, ids, inertias,...) and Output (machine's magnitudes for the PL4 file and the number of EXOUT nodes).

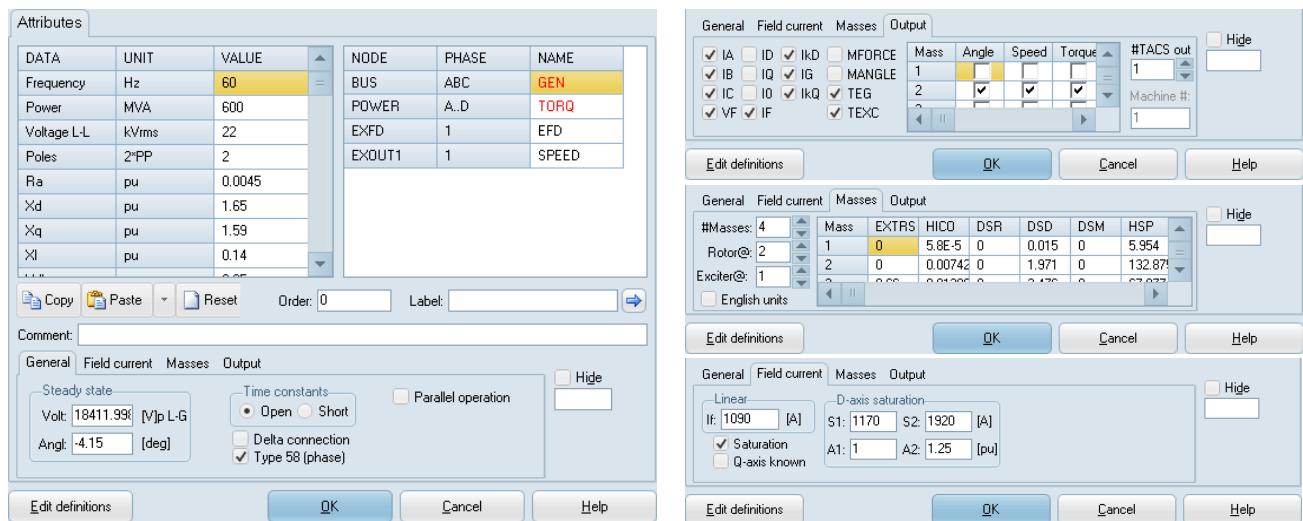


Fig. 6.22 – SM input data: Electrical Attributes, General data, Field Current, Masses and Output.

For the illustration of the controlled synchronous machine represented with the ATPDraw's standard object SM, the companion file [SM58.ACP] contains the simulation case of a generating unit rated 600MVA, 22kV, 60Hz, 2 pole generator and its step-up transformer, that is delivering 500 MW PF=0.0 at 1.025 rated voltage in its steady-state, and; in the transient simulation, it experiences a single-line-to-ground fault at  $t = 1.0$  seconds and a trip/reclose event on phase C of

its 300 km transmission overhead line at 400 kV. The data of the generator, the controls and the rest of the system appears in the Chapt. 6.4.3.

The following figure shows the entire simulated system with the controlled synchronous machine's scheme and its flow of signals:

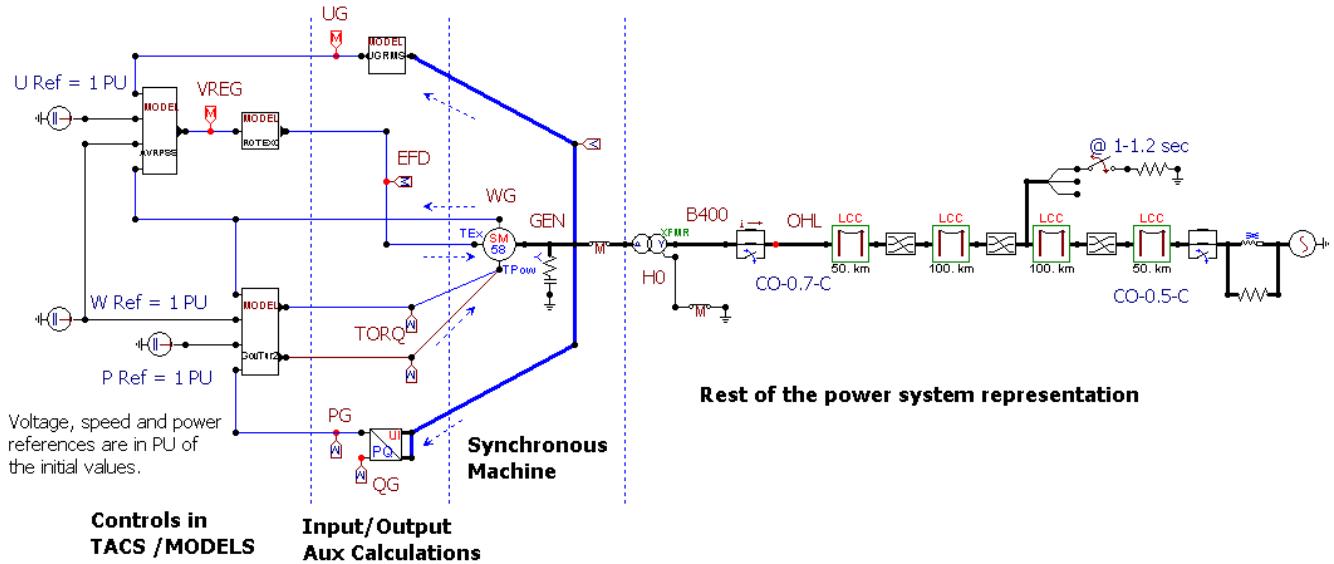


Fig. 6.23 – Power System Representation for the Controlled Synchronous Machine Simulation.

The simplified models of the machine's controls and prime mover in this case are all in MODELS objects, see these scripts in Chapt. 6.4.3. The controls' scheme objects have with the following definitions:

TABLE 6.4.1 Synchronous Machine Controls (MODELS Objects) of the Example Case

MODEL	PURPOSE	INPUT	OUTPUT
UGRMS	Calculation of the L-L rms 3-phase voltage in kV	3-phase set of node voltages	UG – representative L-L rms voltage from generator's stator winding terminals, in kV
UI2PQ3 (ATPDraw's Power System Tools object)	Calculation of the 3-phase P, Q powers in VA	3-phase set of node voltages 3-phase set of switch currents	PG, QG - 3-phase P, Q powers in Watt and VAr
AVRPSS	Representation of the Generator's Automatic Voltage Regulator, with a Power System Stabilizer Function	UG – L-L rms Voltage from generator's stator winding terminals, in kV RefV – Desired generator voltage, in PU of its steady-state value RefW - Desired generator rotor speed, in PU of its steady-state value WG – generator rotor speed, in rad/s	VREG – Regulation voltage for the Generator's Rotating Exciter field winding, in PU of its steady-state value
ROTEXC	Representation of the Generator Rotating Exciter	VREG – Field winding regulation voltage in PU of its steady-state value	EFD – Excitation Voltage for the synchronous machine's field winding, in PU of its steady-state value.
GOVT2	Representation of the Generator's Governor and Turbine (LP and HP)	WG – Generator rotor speed, in rad/s RefW - Desired generator	TORQC, TORQD – Mechanical Torque for the masses 3 (LP) and 4 (HP)

	sections)	rotor speed, in PU of its steady-state value RefP – Desired generator Power, in PU of its steady-state value PG – Generator's 3-phase active power, in Watt;	of the generator rotor, in PU of its steady-state value. Note the single-phase connection lines for the corresponding masses (C = 3, LP; D = 4, HP) directly to the multi-phase node POWER.
--	-----------	--	---

The case was solved with  $\text{deltat}=5.0\text{E-}5$  seconds and  $T_{\text{max}}=20.0$  seconds. The SLG fault is applied from  $t = 1.0$  second to  $t = 1.2$  second; the faulted transmission line phase C is disconnected at the generation end from  $t = 1.05$  second to  $t = 1.75$  second and, from  $t = 1.05$  second to 1.55 second at the system (Thévenin) equivalent end.

Because the synchronous machine model in the ATPDraw object is very easy to select from the ATP/EMTP source type 59 to type 58, both solutions are illustrated and they show a very different machine behavior:

- a) with source type 58, solved in the ABC reference frame, whose results are believed closer to the real behavior, the simulation shows a machine's stable swing, without an out-of-step operation; and
- b) with the source type 59, solved in the d-q-0 reference frame, the simulation shows a machine's 14-pole slip (out-of-step) event that supposes a more severe operational stress on all components of the generating unit and the rest of the power system.

The graphical results of simulations with both sources, type 59 (14-pole slip) and type 58 (no pole slip) are shown next:

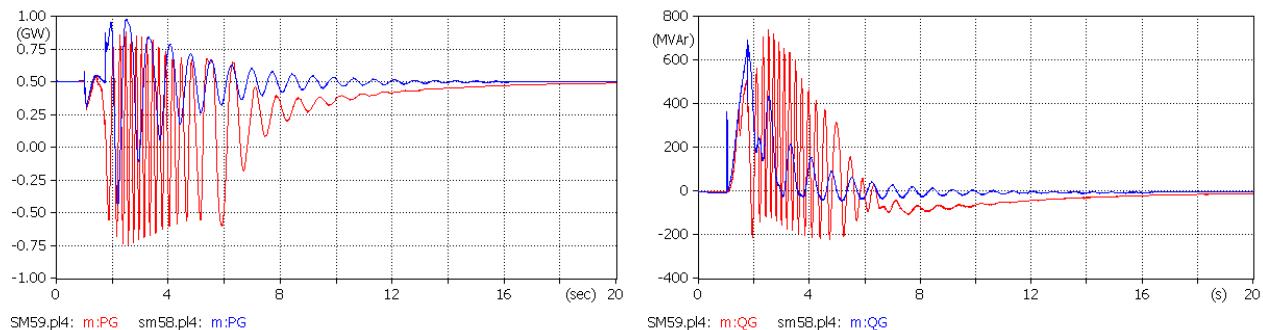


Fig. 6.24 – Active (P) and Reactive (Q) powers delivered by the generator to its step-up transformer.

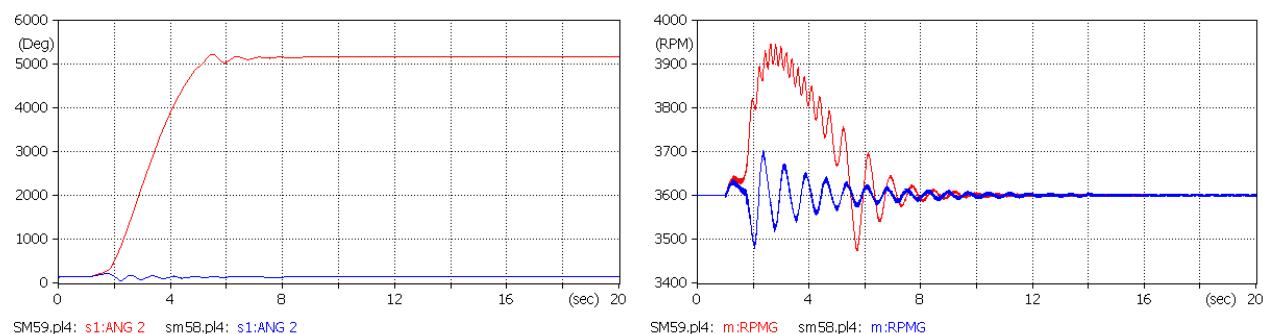


Fig. 6.25 – Generator rotor mass angle and speed.

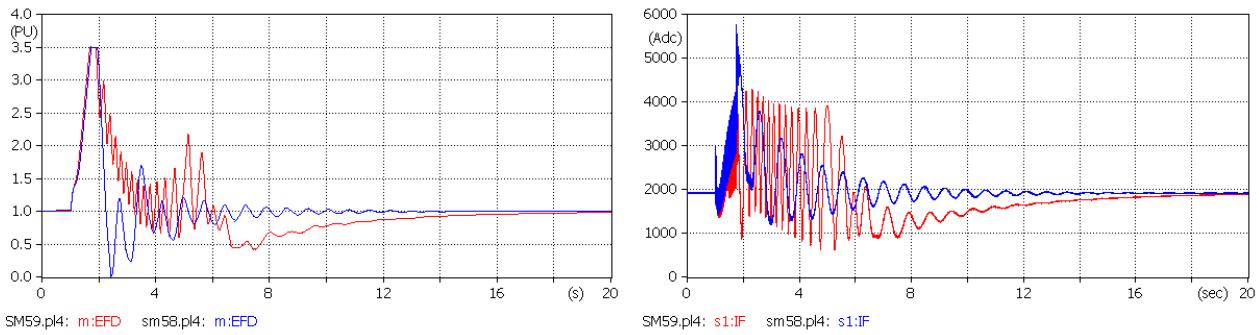


Fig. 6.26 – Rotating exciter output voltage and the current at the generator field winding terminals.

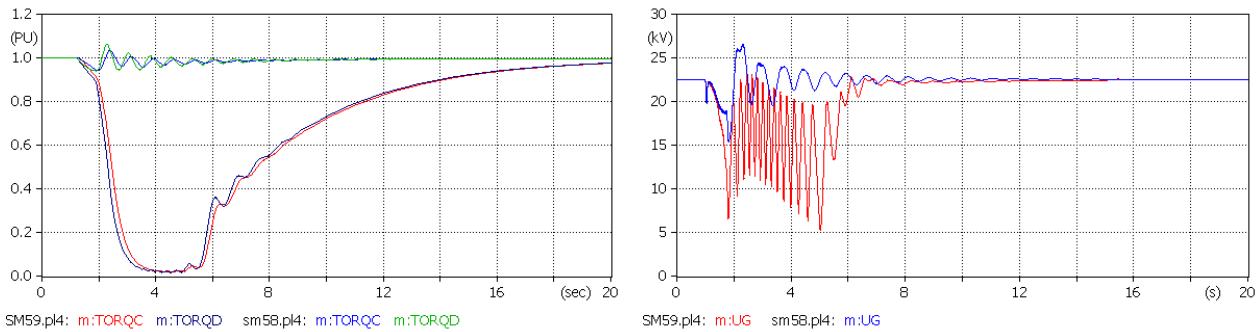


Fig. 6.27 – Torque applied to turbine masses and L-L rms voltage at the stator winding terminals.

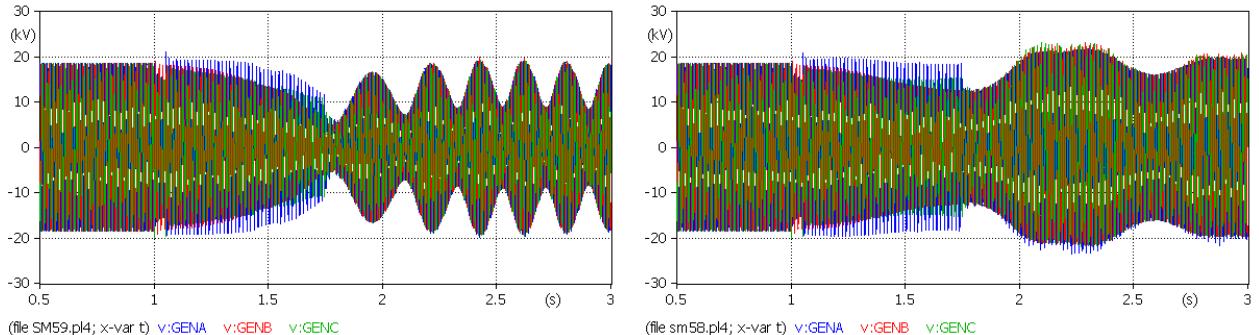


Fig. 6.28 – L-G voltages at the stator winding terminals.

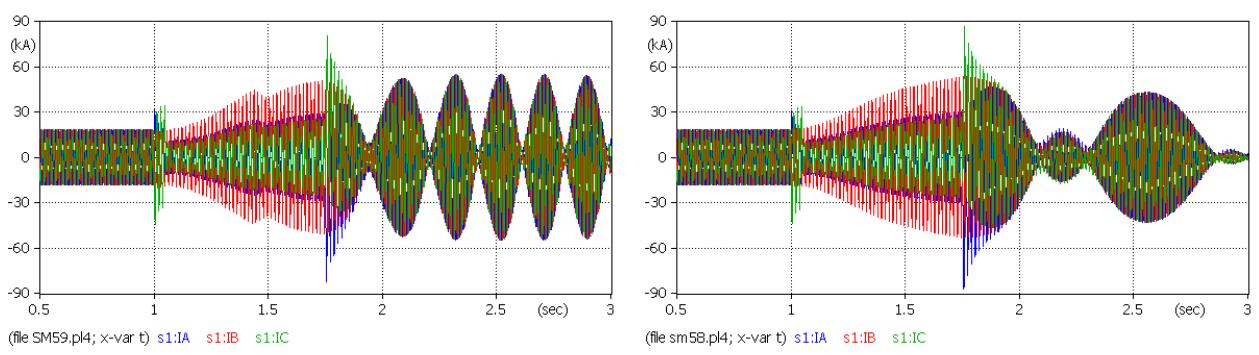


Fig. 6.29 – Currents at the stator winding terminals.

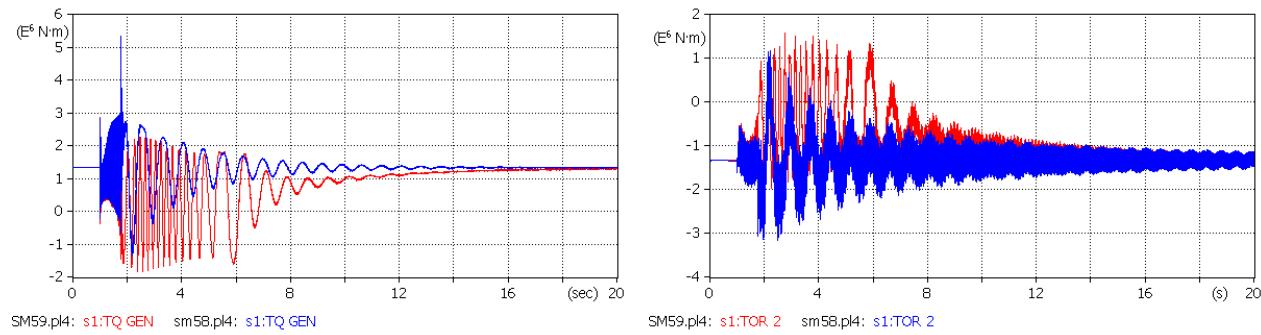


Fig. 6.30 – Torques: electromagnetic and mechanical at the generator - turbine shaft.

#### 6.4.2 Universal machine control (*Exa\_22b.acp*)

The Universal Machine requires input in electrical quantities. This makes it rather difficult to use in practice. The WINDYN tool offers calculation of these parameters based on manufacturers data. The example in this chapter starts with a UMSYN machine with the same parameters as in Chapt. 6.4.1. The UMSYN component support a single mass internally.

The Universal Machine needs control inputs in physical quantities (field voltage in [V] in torque in [Nm] with current in [A] used as analog). Moreover, the Universal Machine can optionally be automatically initialized. The recommended setup is to use Automatic-initialization and record the physical quantities by inspecting the LIS-file. Then the field and torque controls must be modified by TRQ0\*(TRQpu-1.0) and EFD0\*(EFDpu-1.0).

- Run the case with Automatic initialization (no need for tuned controls at this point)
- Inspect the LIS-file, look for ‘Solution at nodes with known voltage’:

Solution at nodes with known voltage. Nodes that are shorted together by switches...						
Node name	Source Rectangular	node Polar	voltage	Injected Rectangular	source Polar	current NA
WGEN	376.99111843078 0.00	376.99111843078 0.0		-.13426960819E7 54.480216294013	.134269608304E7 179.9976752	NA NA
EXFD						
IX0019	23.053311654305 0.0	23.053311654305 0.0		29768.502209823 0.0	29768.502209823 0.0	NA NA

- Take note of the calculated torque current (injected into the node) and the field voltage. In this example these are; WGEN=1.342696E6 [A=Nm] (positive for generator) and EXFD=23.053312 [V]. These values must be used in a final, physical scaling of the controls.
- Use the same control blocks as in the SM case (based on pu quantities) but modify the output via TACS. Use the TFORTTRAN block with one input and one data. Specify the output as shown below. The input to the TACS scaling block comes from MODELS so add a TMODVAR block on the input. The scaling could have been embedded into the MODELS block also.

**TACS:TFORTRAN**

Attributes		
DATA	UNIT	VALUE
#INPUT		1
#DATA		1
DATA1		1342696

NODE	PHASE	NAME
OUT	1	WGEN
IN1	1	TMPU

Order: 0 Label:

Comment: Initial Tmech = 1342696.0957 N.m = 1.0 PU, From Steady-state of LIS file.

**FORTRAN**

```
OUT= $D1 * ($I1-1.0)
```

**TACS:TFORTRAN**

Attributes		
DATA	UNIT	VALUE
#INPUT		1
#DATA		1
DATA1		23.053311654

NODE	PHASE	NAME
OUT	1	EXFD
IN1	1	EFDPU

Order: 0 Label:

Comment: Initial EFD = 23.053311654305 V = 1.0 PU, from Steady-State in LIS file

**FORTRAN**

```
OUT= $D1 * ($I1-1.0)
```

Fig. 6.31 – Scaling of control output to physical quantities. Upper: Torque, lower: Field.

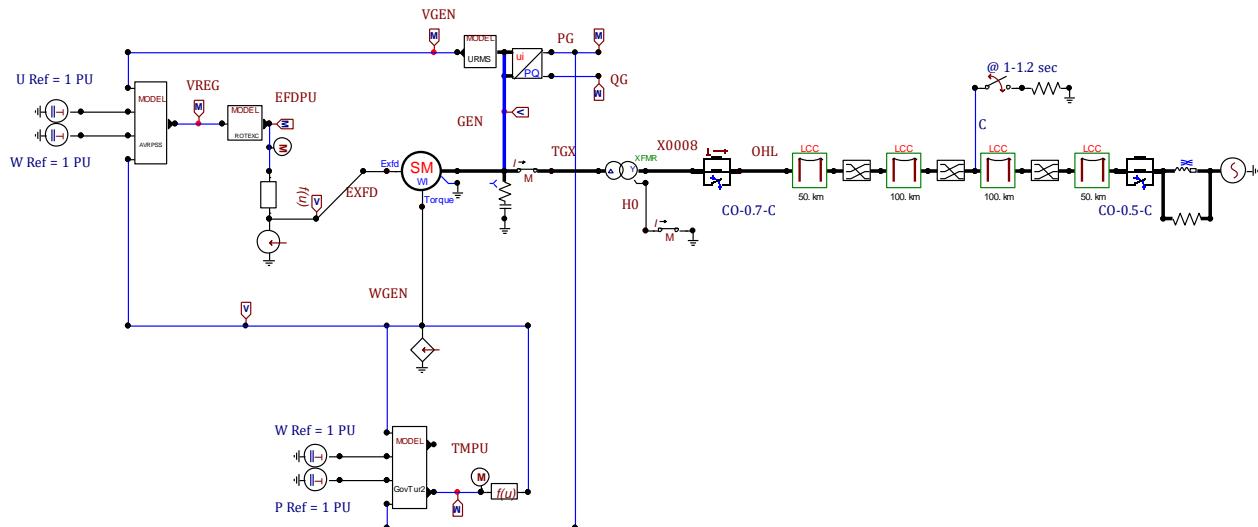


Fig. 6.32 – Same case as in Fig. 6.23 with UMSYN model, *Exa\_22b.acp*

The UMSYN component can be used to calculate the input to an UM1 machine model. This is done by inspecting the ATP-file creating from the UMSYN model and typing in the created coil data manually. The UM1 model needs two TYPE14 AC sources for initialization and the mechanical network added manually as capacitor-resistor equivalents. The control part can be identical.

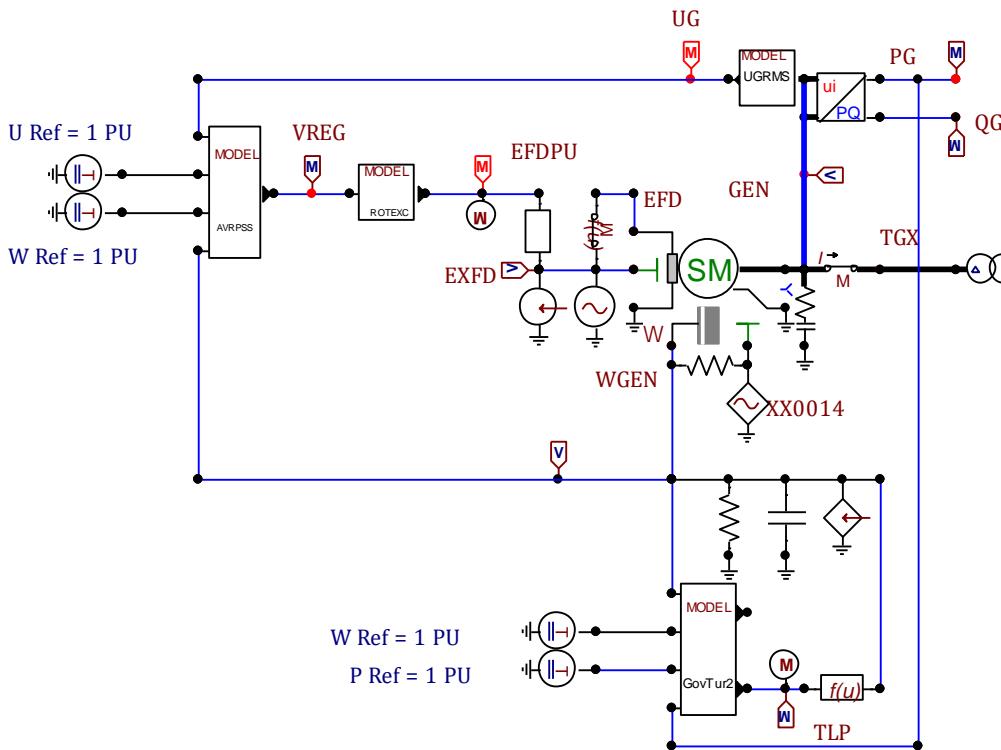


Fig. 6.33 – Same case as in Fig. 6.23 with UM1 model, *Exa\_22c.acp*

Fig. 6.34 compares the active power response of the type 58 synchronous machine in Exa\_22a with the universal machine in Exa\_22b (Exa\_22c is identical to Exa\_22b). We clearly see that the active power delivered is equal in the beginning but damping of the type 58 machine is higher.

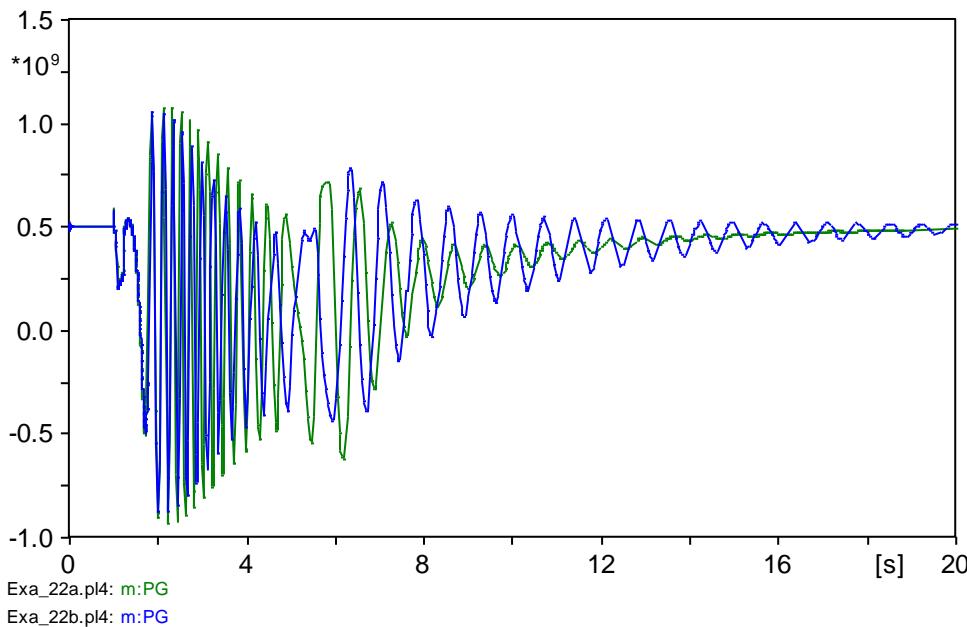


Fig. 6.34 – Active power delivered by the generators in *Exa\_22a* and *Exa\_22b*.

#### 6.4.3 Data used in the ATPDraw cases described Chapt. 6.4.1-6.4.2 (*Exa\_22*).

The generator has the following data.

TABLE 6.4.2 Synchronous machine data.

Electrical data:					
Rated Capacity, Sn in MVA	600.00				Adc
Rated L-L Voltage, Un in kV	22.00	Rated Load (Sn, PFn, Un) Field Current, Ifn			3035.0
Rated Frequency, Fn in Hz	60.00	No-Load Rated (Un) Field Current, AGline			1090.0
Rated Speed, in RPM	3600.0	No-Load Field Current (1.0 Un), S1			1170.0
Rated Power Factor, PFn in PU	0.90	No-Load Field Current (1.25 Un), S2			1920.0
Full Load Generator Efficiency	99.03%				
Full Load Generator+Turbine Efficiency	98.47%				
	PU				PU
D-Axis Synchronous Reactance, Xd	1.650	Stator Winding Resistance, Ra			0.0045
Q-Axis Synchronous Reactance, Xq	1.590	Stator Winding Leakage Reactance, XL			0.1400
D-Axis Transient Reactance, X'd	0.250	Zero Sequence Reactance, Xo			0.0802
Q-Axis Transient Reactance, X'q	0.460	Neutral Grounding Resistance, Rn			900.00
D-Axis SubTransient Reactance, X"do	0.200	Neutral Grounding Reactance, Xn			64.533
Q-Axis SubTransient Reactance, X"qo	0.200				
	Seconds				
D-Axis Transient Time Const, T'do	4.5	Generator L-G Capacitance, Cg [uF]			0.2325
Q-Axis Transient Time Constant, T'qo	0.550				
D-Axis SubTransient Time Constant, T"do	0.040				
Q-Axis SubTransient Time Constant, T"qo	0.090				
Mechanical data:					
Mass No.	Rotor	EXTRS [% Tmec]	HICO [E6 kg.m <sup>2</sup> ]	DSD [N.m/(rad/s)]	HSP [E6 N.m/rad]
1	Exciter	- - -	0.000058	0.015	5.954
2	Generator	- - -	0.007425	1.971	132.875
3	Turbine LP	66%	0.013094	3.476	67.977
4	Turbine HP	34%	0.002103	0.558	- - -

The Step-up transformer data is: 650 MVA 22kV:400kV DY1g Zt = 11% Ploss@Sn = 5520.0 kW. This step-up transformer was represented with the ATPDraw's Hybrid Transformer object.

The 300 km overhead transmission line at 400 kV has three 1113 kCM ACSR/AW conductors per phase and two guards galvanized steel conductors with this configuration data:

TABLE 6.4.3 Overhead line data.

	Rin [cm]	Rout [cm]		Rdc[ohm/km]	Horiz [m]	Vtower [m]	Vmid [m]
Phase A	0.399	1.599		0.0511	10.5	25.0	12.5
Phase B	0.399	1.599		0.0511	0.0	25.0	12.5
Phase C	0.399	1.599		0.0511	-10.5	25.0	12.5
Guard 1	0.000	0.4890		1.463	11.0	34.3	23
Guard 2	0.5695	0.7325		1.460	-11.0	34.3	23

It is represented with four segments of  $1/6 - 1/3 - 1/3 - 1/6$  of the total length and three ABC-BCA transpositions; every segment has electrical parameters obtained with the Bergerón model, traveling wave constant parameters, at the rated frequency of 60 Hz.

The generator delivers power to a remote power system represented with a balanced Thévenin equivalent ( $S_{sc3ph} = 6642.8$  MVA,  $S_{sc1ph} = 7909.9$  MVA) with these data:

TABLE 6.4.4 Remote source data.

	Rthev [ohm]	Xthev [ohm]
Zero sequence	0.56344	13.7529
Positive sequence	1.62208	26.4452

The MODELS user objects in this example case, with the data applied in the simulations, have these codes:

```

-----  

MODEL UGRMS -- Representative 3-PH L-L RMS VOLTAGE  

DATA Freq{dflt: 60.0} -- Rated Frequency, in Hz  

  Uini{dflt: 22.55} -- Initial Voltage, in kV  

INPUT Ubus[1..3]  

OUTPUT Urms  

VAR Urms, Tf11, Tf12, Ualp, Ubet, Uraw  

HISTORY Urms{dflt: Uini}  

INIT  

  Tf11 := 1.0/Freq  

  Tf12 := (Tf11/2.0)**2.0  

ENDINIT  

EXEC  

  Ualp := (2.0*Ubus[1]-Ubus[2]-Ubus[3])/3.0  

  Ubet := (Ubus[2] - Ubus[3])/1.7320508076  

  Uraw := sqrt(1.5*(Ualp**2 + Ubet**2))  

  CLAPLACE(Urms/Uraw) := (0.001|S0)/(1.0|S0 + Tf11|S1 + Tf12|S2)  

ENDEXEC  

ENDMODEL  

-----  

MODEL AVRPSS -- An Automatic Voltage Regulator + PSS  

DATA RVgn{dflt: 22.000} -- Rated voltage, in kV  

  Vini{dflt: 22.550} -- Initial Voltage, in kV  

  Kreg{dflt: 400.00} -- Regulator Amplifier Gain, in PU  

  MaxV{dflt: 5.000} -- Exciter Max limit, in PU  

  RPMi{dflt: 3600.0} -- Initial Speed, in RPM  

INPUT Vgen, RefV, RefW, Wgen  

OUTPUT EFD  

VAR EFD, Wini, LimV, DWgn, DWps  

  DVps, DVgn, ErrV, ErfV, DVfb  

  Tled, Tlag, Treg, Kfbk, Tfbk  

  Twsh, Tlgw, Tldw, LVps  

HISTORY DWps{dflt: 0.0}, DVps{dflt: 0.0}, ErfV{dflt: 1.0/Kreg}

```

```

EFD{dflt: 1.0}, DVfb{dflt: 0.0}
INIT
Tled := 0.400 -- AVR lead time constant
Tlag := 0.050 -- AVR lag time constant
Treg := 0.020 -- AVR lead time constant
Kfbk := 0.050 -- AVR lead time constant
Tfbk := 2.000 -- AVR lag time constant
Twsh := 5.000 -- PSS Washout time constant
Tldw := 0.150 -- PSS lead time constant
Tlgw := 0.050 -- PSS wash lag time constant
LVps := 0.100 -- Pss Min/Max limit
Wini := PI*RPMi /30.0
LimV := MaxV*RVgn/Vini
ENDINIT
EXEC
DWgn := (RefW - Wgen/Wini) -- Speed Error in PU
CLAPLACE(DWps/DWgn) := (Twsh|S1)/(1.0|S0 + Twsh|S1)
CLAPLACE(DVps/DWps) {dmin:-LVps dmax:LVps} := (1.0|S0+Tldw|S1)/(1.0|S0+Tlgw|S1)
DVgn := (Kreg/(Kreg-1.0))*RefV - Vgen/Vini -- Voltage Error
ErrV := DVgn - DVps - DVfb -- Sum point
Claplace(ErfV/ErrV) := (1|S0+Tled|S1)/(1|S0+Tlag|S1)
Claplace(EFD/ErfV) {dmin:-LimV dmax:LimV} := (Kreg|S0)/(1|S0+Treg|S1)
Claplace(DVfb/EFD) := (Kfbk|S1)/(1|S0+Tfbk|S1)
ENDEXEC
ENDMODEL
-----  

MODEL ROTEXC -- Simple Rotating Exciter Model
DATA
Texc{dflt: 0.600} -- Exciter time constant, in s
A{dflt: 1.440} -- Frolich Saturation constants for
B{dflt: 0.440} -- Ug(Ur) = A*Ur/(1+B*Ur)
Vemx{dflt: 3.500} -- Maximum EFD output
Vemn{dflt: 0.010} -- Minimum EFD output
INPUT VREG
OUTPUT EFD
VAR EFD, Vsum
HISTORY EFD{dflt: 1.0}
EXEC
Vsum := VREG - A*EFD/(1.0 + B * EFD) -- Sum point
CLAPLACE(EFD/Vsum) {dmin:Vemn dmax:Vemx} := (1.0|S0)/(Texc|S1)
ENDEXEC
ENDMODEL
-----  

MODEL GOVT2 -- SM's Power/Speed Governor + Turbine
DATA RPMi {dflt: 3600.0} -- Stead-state speed, in RPM
Pmax {dflt: 600.0} -- Max Turbine power, in MW
Pini {dflt: 500.0} -- Initial gen power, in MW
Drop {dflt: 0.050} -- Regulation (Droop) factor, in PU
TtHP {dflt: 0.350} -- HP Turbine time constant, in s
TtLP {dflt: 0.150} -- LP Turbine time constant, in s
INPUT Wgen, RefW, RefP, Pgen
OUTPUT PmLP, PmHP
VAR PmHP, PmLP, RWgn, LimP, DWgn
DPWg, DPgn, DPPg, DPgt, Pval
Tldw, Tlgw, Kgnp, Tldp, Tlgp, RPMG
HISTORY DPWg{dflt: 0.0}, DPPg{dflt: 0.0}, Pval{dflt: 1.0}
PmHP{dflt: 1.0}, PmLP{dflt: 1.0}
INIT
RWgn := PI * RPMi / 30.0 -- Rated Speed in rad/s
LimP := Pmax/Pini -- Turbine Power limit, in PU
Tldw := 0.250 -- Gov W Lead time constant, in s
Tlgw := 0.010 -- Gov W Lagg time constant, in s
Kgnp := 0.250 -- Power Error Gain, in PU
Tldp := 0.250 -- Gov P Lead time constant, in s
Tlgp := 0.010 -- Gov P Lagg time constant, in s
ENDINIT
EXEC
RPMG := 30.0 * Wgen / PI -- Gen Speed in RPM

```

```

DWgn := (RefW - Wgen/RWgn)/Drop           -- Speed error in PU
CLAPLACE(DPWg/DWgn) := (1.0|S0 + Tldw|S1)/(1.0|S0 + Tlgw|S1)
DPgn := RefP - Pgen/(1.0E6*Pini)          -- Power error in PU
CLAPLACE(DPPg/DPgn) := (1.0|S0 + Tldp|S1)/(1.0|S0 + Tlgp|S1)
DPgt := DPWg + DPPg * Kgnp                -- Total Power Error
CLAPLACE(Pval/DPgt){dmin:0.0 dmax:LimP} := (1.0|S0)/(1.0|S1)    -- Valve
CLAPLACE(PmHP/Pval) := (1.0|S0)/(1.0|S0 + TtHP|S1)              -- HP Turbine
CLAPLACE(PmLP/PmHP) := (1.0|S0)/(1.0|S0 + TtLP|S1)              -- LP Turbine
ENDEXEC
ENDMODEL

```

#### 6.4.4 The Controlled Induction Machines (*Exa\_23.acp*)

The simulation of the transient events in the operation of induction machines is needed in many studies. The results of these simulations provide the assessment of the electrical and mechanical performance of the induction machines regarding an event in the electrical system the machine is connected to, or; the impact in the performance of that electrical system due to a transient event in the mechanical load of the induction machine. In both situations, the control on mechanical torque that the load or the prime mover applies to the machine is an essential part of the rotating machine representation.

As exposed in chapter 6.4.1, an induction machine (motor or generator) can be represented in ATPDraw circuit projects with these elements:

- The ATP/EMTP source type 19 (Universal Machine, UM) code 3, the single-fed induction machine solved in the d-q formulation frame, represented with the ATPDraw objects: UMIND with the controlled torque applied as an incremental value in physical units (N.m), additional to its value determined in its steady-state solution; and UM\_3, that only represents the electromagnetic performance of the induction machine, while the mechanical rotational body must be modelled as an electrical analog network and controlled torques - in physical units (N.m)- are applied as TACS-controlled current sources; and
- The ATP/EMTP source type 56 (TEPCO's Machine), the single-fed induction single-mass machine solved in the ABC formulation frame, represented with the ATPDraw objects IM56A, with the controlled torque applied as a PU magnitude of its value determined at the steady-state solution.

The mechanical load in the machine's shaft is represented by a torque to be applied to the rotating mass of the machine and, this torque can be a function of time according to the machine's operational duty or, a speed-dependent function according to the load type (fan, compressor, mill, etc.). In both cases, the load must be represented by a TACS or MODELS variable with the torque value in the proper unit (N.m or PU of steady-state value) that is "connected" to the input node of UMIND, UM\_3 or IM56A.

In order to illustrate on the controlled-torque induction machine, the example case *Exa\_23.acp* contains an induction motor represented, in three unconnected sub-circuits with all three UMIND, UM\_3 and IM56A objects and, this motor is subject to an operational sequence that includes the start-up at full-voltage, the mechanical full-load application and, a Line-to-Line fault for 0.5 seconds at its feeder. The motor and feeder have the following nameplate/catalog rated data and operative specifications.

TABLE 6.4.5 Induction Motor and Feeder Data for the Example Simulation.

Induction Motor:			
Rated Output Load	6500 HP, 4847.05 kW	Rated Frequency	60 Hz
Rated Voltage	6.6 kV	Speed at Rated Load, Voltage And Frequency	1787 RPM
Current at Rated Load, Voltage And Frequency	490 A	Rated Starting Current and No-load Current	5.1 PU and 0.25 PU of Rated Current
Efficiency at Rated Load, Voltage And Frequency	96.8%	Maximum and Starting Torque	250% and 20% of Rated Torque
Rated F&W Losses	1% of Rated Load	Moment of Inertia	1.15 kW.s/KVA
Feeder, represented with a Thévenin equivalent:			
Voltage	Rated 6.9 kV, operates at 6.85 kV	Zero sequence impedance	0.0233 + j0.1275 ohm
Frequency	60 Hz	Positive sequence impedance	0.0095 + j0.2937 ohm

The induction machine is represented with a single-cage rotor in all three objects, even when in UMIND and UM3 the machine can be specified with a more complex structure. Besides the lack of information, the machine's saturation was considered of minor relevance for this rated-voltage case and it was not represented.

In order to simulate the source type 56 with a floating-neutral stator winding, comparable to the UM3 and UMIND connection, an auxiliary 1:1 front-end Y-Yg transformer is included. Also, the simulation begins with the TEPCO's IM at 0.001% rated speed that yield a steady-state non-zero torque, that is the base for the required torque scaling to PU of the initial value. The UM3 and UMIND representations start at 0.0% rated speed and their initial load torque is 0.0 N.m, of course.

Now, the machine's load torque is represented with this speed ( $\omega$ ) function for the steady state operation:

$$T[\omega] = T_{min} + K_t \cdot |\omega - \omega_{min}|^\alpha \quad (6.1)$$

The function parameters can be determined with the full-load (6500 HP, 4847.050 kW) characteristic given by these three points:

TABLE 6.4.6 Torque speed characteristic.

CONDITION	1 - Stand Still		2 - Minimum Torque		3 - Full-load	
Speed, S	0.000 PU	0 RPM 0 rad/s	0.200 PU	357.4 RPM 37.427 rad/s	1.00 PU	1787 RPM 187.134 rad/s
Torque, T	0.150 PU	3885.22 N.m	0.075 PU	1942.61 N.m	1.00 PU	25901.465 N.m

With:

$$T_{min} = T_2 \quad (6.2)$$

$$\alpha = [\ln(T_3 - T_2) - \ln(T_1 - T_2)] / [\ln(S_3 - S_2) - \ln(S_2)] \quad (6.3)$$

$$K_t = \exp(\ln(T_3 - T_2) - \alpha * \ln(S_3 - S_2)) \quad (6.4)$$

With the speed in rad/s and the torque in N.m, the parameters of  $T[\omega]$  are:

$$T_{\min} = 1942.61 \text{ N.m}$$

$$K_t = 2.7377361$$

$$\alpha = 1.8122454$$

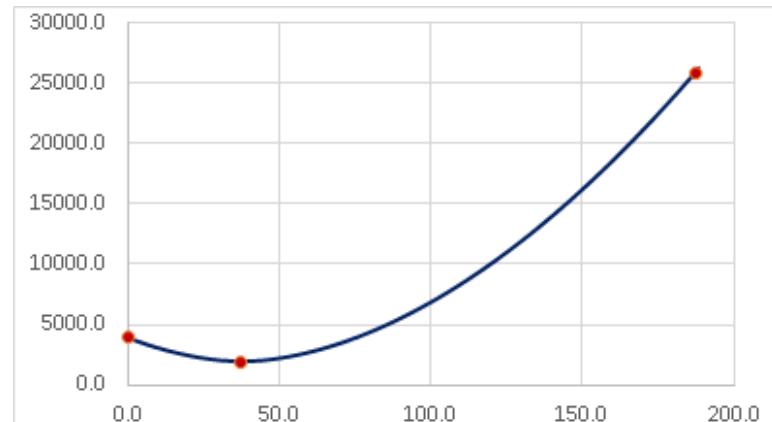


Fig. 6.35 – Torque-Speed function.

In addition to this steady-state behavior, a low-pass transfer function, with a unitary gain and a time constant of 0.3 seconds, is added in order to represent the possible load torque delay response to a sudden speed change.

Now, for the induction motor simulation with its UMIND representation, the circuit has this appearance:

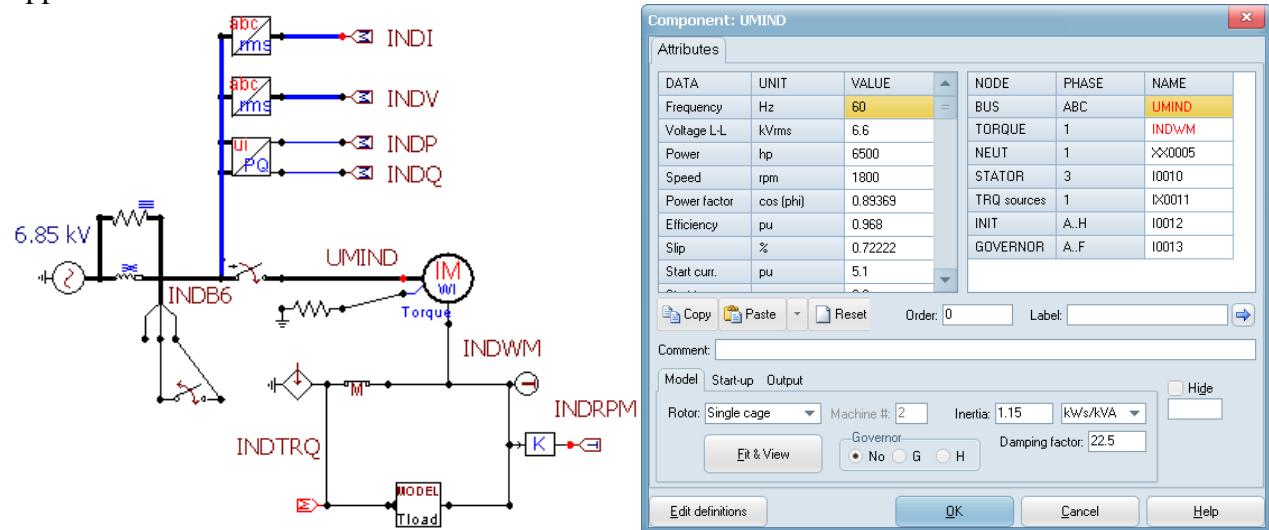


Fig. 6.36 – Circuit for the simulation of a single-cage induction motor with UMIND and its

Attributes window.

In the UMIND data window, the MODEL was chosen as “Single-cage” rotor with a damping factor of 22.5; the STARTUP was specified at a “Slip” of 100% respect to synchronous speed and, the invoked “Fit & View” reported this result from the ATPDraw fitting process for the UM code 3 data:

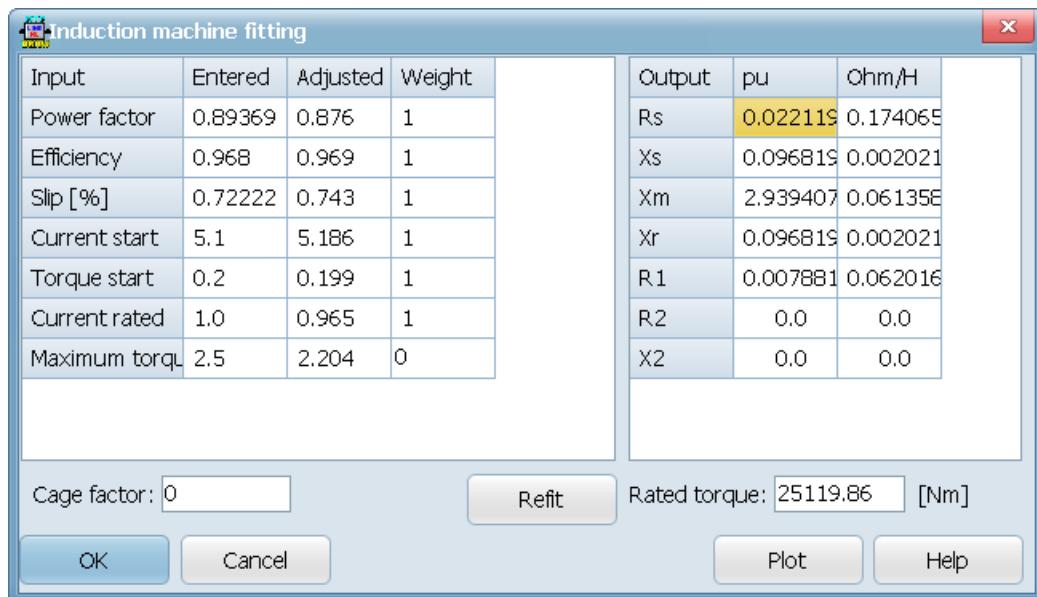


Fig. 6.37 – Data for the UM code 3, obtained with the UMIND “Fitting & View” feature.

Now, the load torque in N.m for its application at the UMIND object is coded in MODELS as:

```

MODEL T4UMIND -- Mech Load Torque for UMIND

DATA
  Snom{dflt: 1787.0} -- Rated speed, in RPM
  Tnom{dflt: 25901.465} -- Rated torque, in N.m
  Tat0{dflt: 0.150} -- Torque at zero speed, in PU of Tnom
  Smin{dflt: 0.200} -- Speed at minimum torque, in PU of Snom
  Tmin{dflt: 0.075} -- Minimum torque, in PU of Tnom
  Tauc{dflt: 0.300} -- Load time constant, in second
  Tini{dflt: 0.000} -- Initial motor torque, in N.m
  Tsta{dflt: 10.000} -- Load torque start time, in second

INPUT Smotr
OUTPUT Tload

VAR Tatz, Tcon, Srat, Scon, Alph, Ktrq, Tstdy, Tload
HISTORY Tload{dflt: 0.0}

INIT
  Tatz := Tnom * Tat0
  Tcon := Tnom * Tmin
  Srat := Snom * Pi / 30.0 -- rated speed in rad/s
  Scon := Srat * Smin
  Alph := (ln(Tnom-Tcon)-ln(Tatz-Tcon)) / (ln(Srat-Scon)-ln(Scon))
  Ktrq := exp(ln(Tnom-Tcon)-Alph*ln(Srat-Scon))
ENDINIT

EXEC
  Tstdy := Tini - (t>Tsta) * (Tcon + Ktrq*(abs(Smotr-Scon))**Alph) -- Steady
  torque
  CLAPLACE(Tload/Tstdy) := (1.0|S0)/(1.0|S0 + Tauc|S1) -- plus inertia
ENDEXEC
ENDMODEL

```

Note that the calculated torque is defined negative because it represents a withdrawal of power from the machine. For cases with non-zero initial speed, the Tini torque can be obtained from the steady-state report in the LIS file and, the Tini value must be positive because this initial torque is

already applied by an internal current source in the UMIND object and the external current source only must apply the additional torque.

The Data entry window and the Help text of the T4UMIND MODEL is:

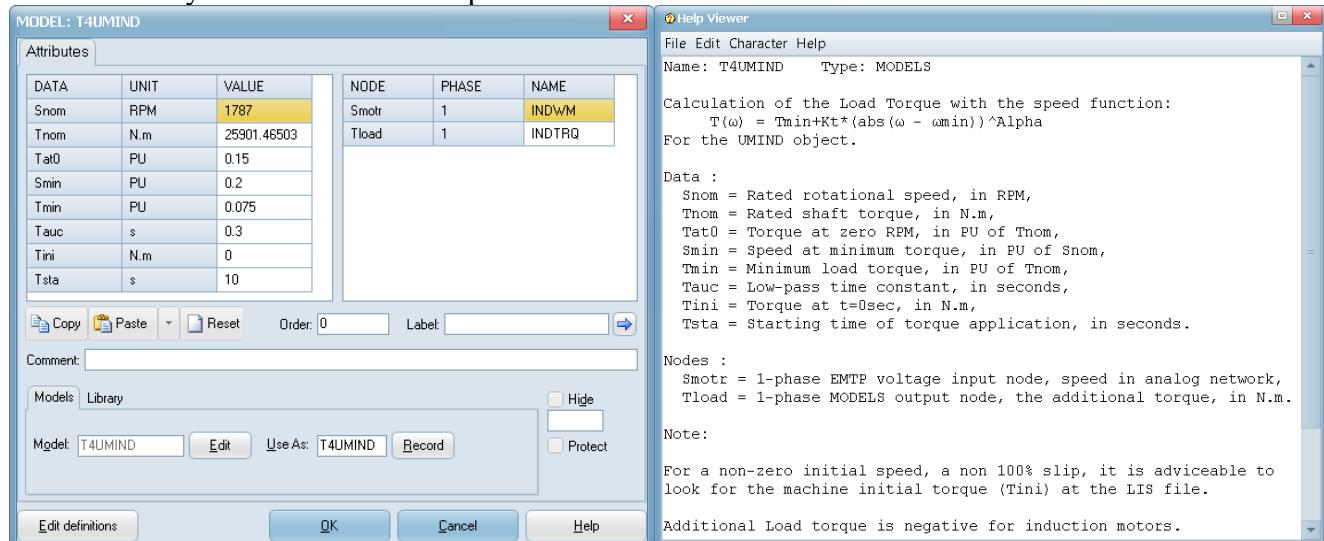


Fig. 6.38 – Data Entry Window and Help text of MODEL T4UMIND.

On other hand, the simulation circuit with the UM\_3 representation, with the same data as obtained with the “Fit & View” feature and the RC electric analog network derived of the UMIND object, obtained from the ATP file or LIS file, has this appearance:

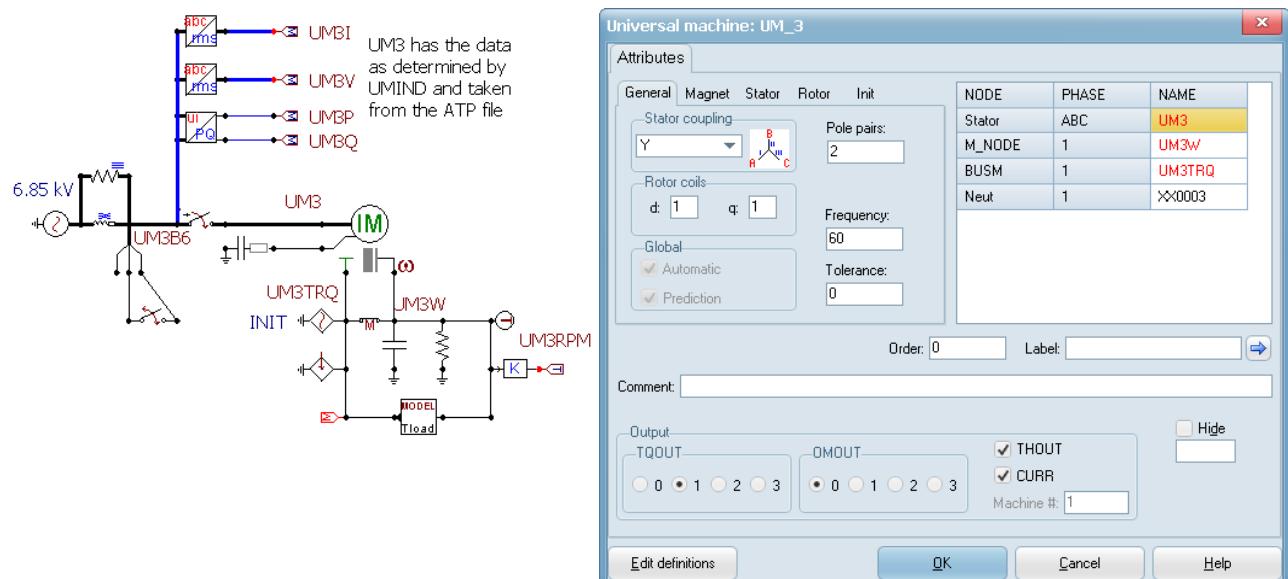


Fig. 6.39 – Circuit for the simulation of a single-cage induction motor with UM\_3 and its Attributes window.

Note the type 14 AC current source -required by the UM code 3 at BUSM- has Tsto set to 1.0E-9 seconds and, the TACS-controlled type-60 current source then apply the full calculated load torque passed from the MODEL named T4UM3. The code of this MODEL differs from that of T4UMIND only in these sections:

```

MODEL T4UM3 -- Mech Load Torque for UM3
...
VAR Tatz, Tcon, Srat, Scon, Alph, Ktrq, Tstrt, Tstdy, Tload
HISTORY Tload{dflt: -Tini}
...
EXEC
  Tstrt := -(t<=Tsta)*Tini
  Tstdy := Tstrt - (t>Tsta)*(Tcon + Ktrq*(abs(Smotr-Scon))**Alph)
  CLAPLACE(Tload/Tstdy) := (1.0|S0)/(1.0|S0 + Tauc|S1)
ENDEXEC
ENDMODEL

```

The Data entry window, the Help text and the Tini recommendation of the T4UM3 MODEL are essentially the same as for T4UMIND.

At last, the simulation circuit with the IM56A representation is shown in the Fig. 6.40, where an auxiliary 1:1 front-end Y-Yg transformer is included in order to have a stator winding neutral point available for the same floating neutral connection of the UMIND and UM\_3 objects. Also, the data for the IM56A object is obtained with a simplified calculation in a spreadsheet in a file included in the example case *Exa\_23.acp*.

The circuit appearance and the IM56A data entry window are:

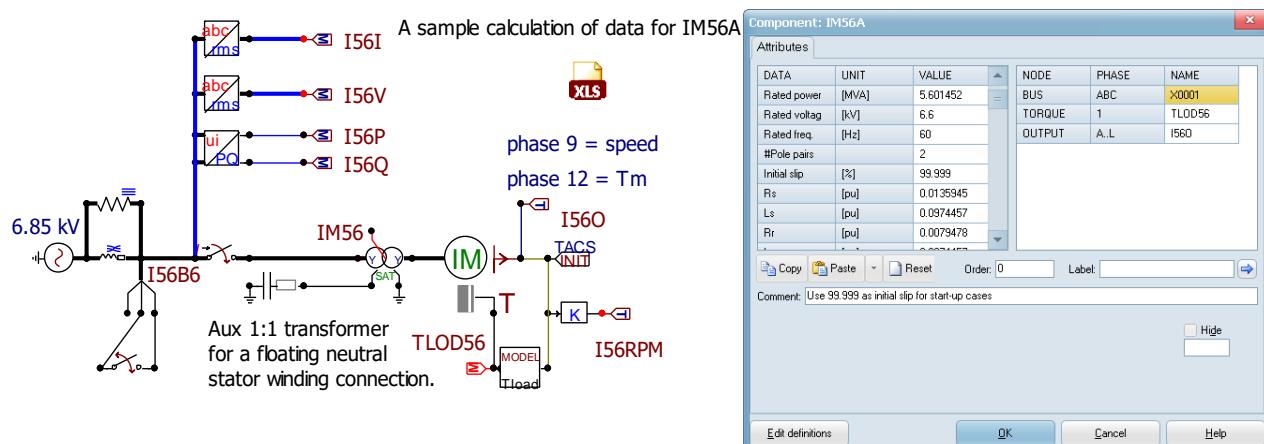


Fig. 6.40 – Circuit for the simulation of an induction motor with IM56A and its Attributes window.

Note that the 12-phase IM56A output node, named I56O, provides in phase 9 the speed value -in rad/s- that is admitted with the proper connection 9-I as a TACS input into the T4IM56 MODELS object, whose output is the calculated load torque that is connected to the TORQE node of the IM56A object, here named TLOD56. The code in this T4IM56 object only differs from that of T4UMIND in the following:

```

MODEL T4IM56 -- Mech Load Torque for IM56A
...
HISTORY Tload{dflt: 1.0}
...
XEC
  Tstrt := (t<=Tsta)
  Tstdy := Tstrt - (t>Tsta)*(Tcon + Ktrq*(abs(Smotr-Scon))**Alph)/Tini
  CLAPLACE(Tload/Tstdy) := (1.0|S0)/(1.0|S0 + Tauc|S1) -- plus inertia
ENDEXEC
ENDMODEL

```

Also note that if the ATP/EMTP source type 56 is specified starting at zero speed, the executable tpbig.exe aborts the simulation. To avoid this in the example, the initial slip is 99.999% regarding the synchronous speed, as if the simulated machine is a very-slow-moving generator, requiring a mechanical torque = 2.97166258E-03 N.m, as it is read in the steady-state report of the LIS file:

```
The initial torque calculation for a Type-56 IM at bus "X0001A" differs by 1% or more from the scheduled input value. Within
overlay 11 SUBROUTINE IMINIT, the two values are, respectively: TM, TMINIT = 2.97166258E-03 0.0000000E+00
```

Exactly this initial torque value is entered as  $T_{ini}$ , the base torque value in the data entry window of the MODELS object named T4IM56, that it is shown next along with its Help text.

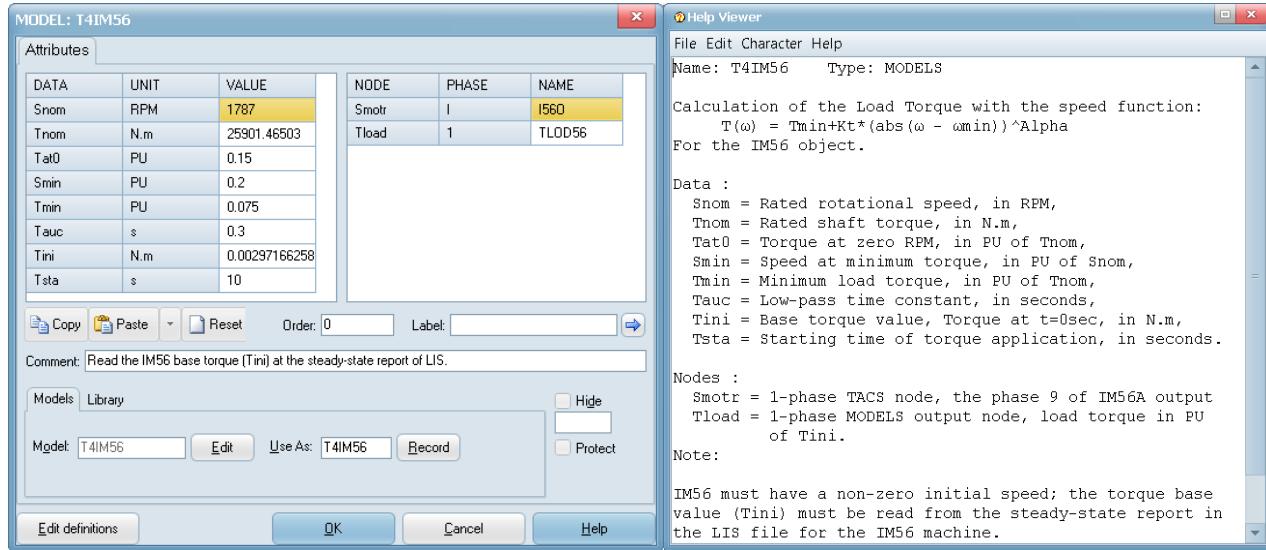


Fig. 6.41 – Data Entry Window and Help text of MODEL T4UMIND.

Now and all circuits in the same example, with  $\Delta t = 5.0E-5$  seconds and  $T_{max} = 12$  seconds, all three UMIND, UM\_3 and IM56A replicated well the rated data of no-load, rated load and starting currents, efficiency and slip at full load in a previous rated voltage, rated frequency case. Now, in the example case *Exa\_23.acp* with the feeder's Thévenin equivalent data from table 6.4.5, the simulated sequence is:

- 0.000s      UM3/UMIND at zero rpm initial speed and IM56A at  $(1800 * 0.00001)$  0.018 RPM.
- 0.050s      Start-up transient, full-voltage application on unloaded motor,
- 10.000s      Full-load application to the motor(s),
- 12.000s      L-L fault at the feeder's terminals,
- 12.500s      L-L Fault is cleared,
- 15.000s      End of simulation.

This case results show the voltage sag in amplitude and duration time due to the no-load start-up operation of the motor and the starting current magnitude; due to the L-L fault, the variations on the motor's speed, power demand and currents. The results also show that both circuits with UM (UMIND and UM\_3) yield the same magnitudes, with slight differences regarding the magnitudes from the circuit with the IM56A object.

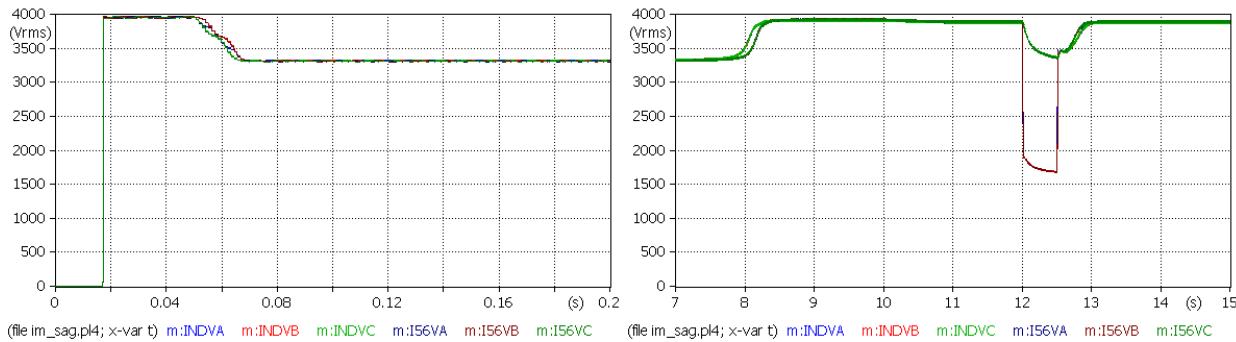


Fig. 6.42 – Variations on the L-G RMS voltage at the motors' terminals.

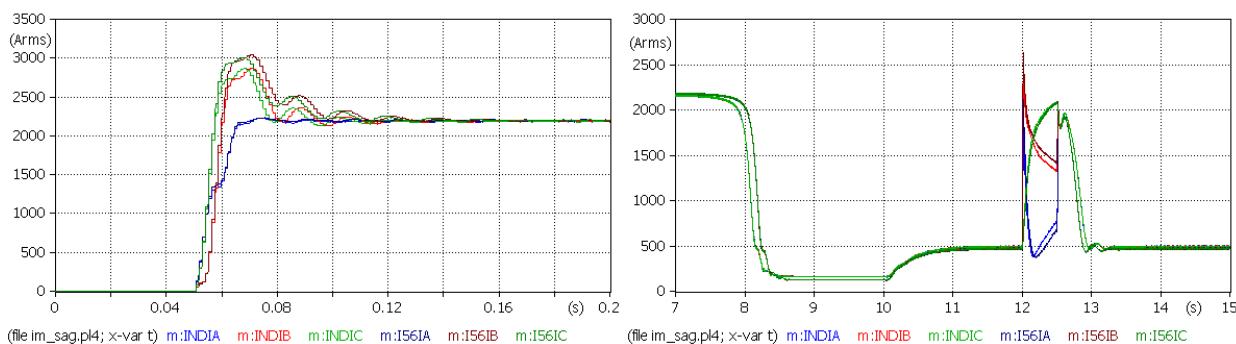


Fig. 6.43 – RMS Motor Currents: energization, full-load application and L-L fault.

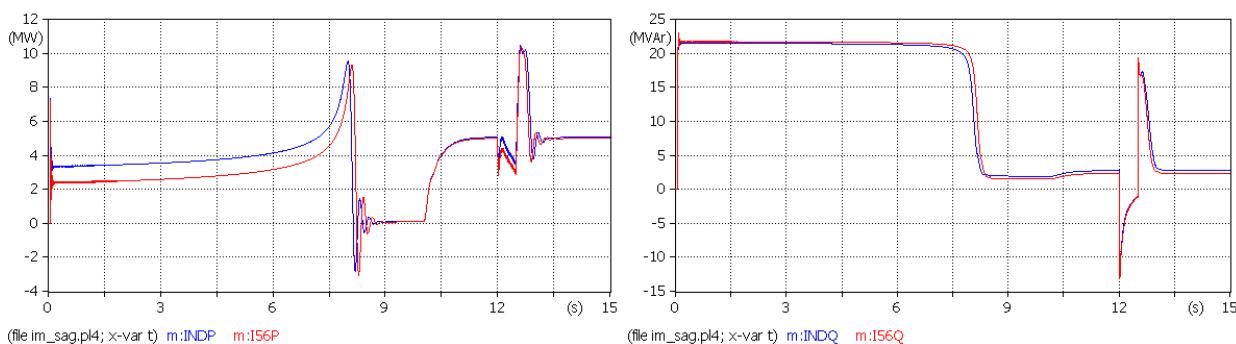


Fig. 6.44 – Motor demand in the simulated sequence: Active (P) and Reactive (Q) Powers.

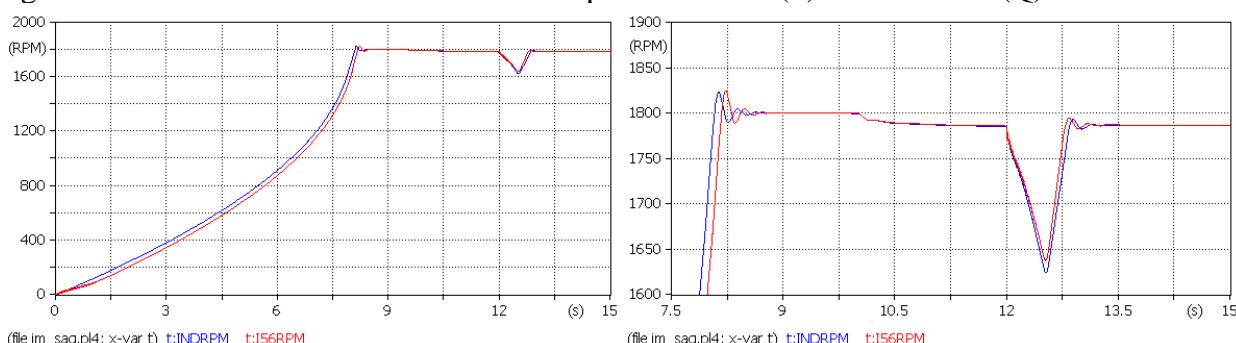


Fig. 6.45 – Rotational Speed (RPM) of the motor, detail at the full-load and L-L fault.

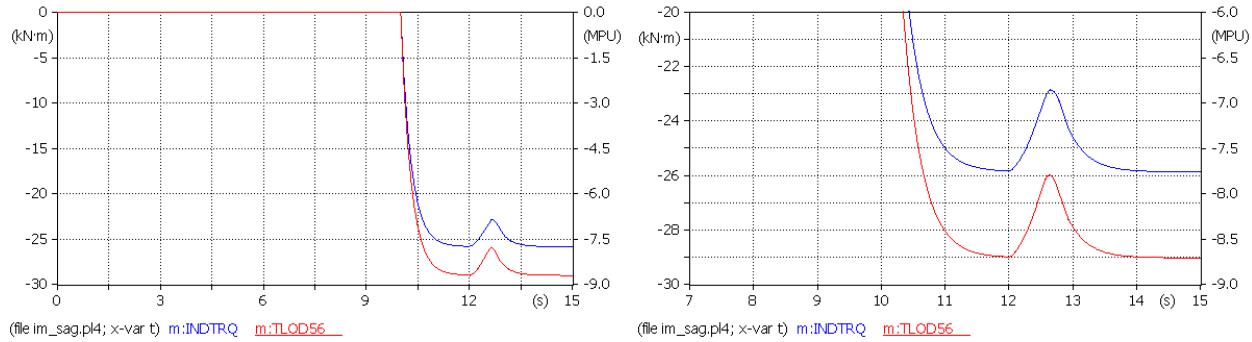


Fig. 6.46 – Mechanical Load torque applied to the motor, detail at the full-load and L-L fault.

#### 6.4.5 Windsyn machine control (Exa\_17.acp)

Machine control is typically of minor importance in an electromagnetic transients program as the time constants involved are much larger than the electrical time constants. Nevertheless, in some situations it might be of interest. The Fig. 6.47 shows a simple example where the Windsyn synchronous machine model is being controlled by a governor and an exciter. The loads of the machine double at 2 seconds and go back to the initial 500 kW at 10 seconds. The Windsyn generator is auto-initialized and this involves two sources hidden inside its lib-file. Initialization of the control units can thus be a challenge. To control the machine additional external sources must be adjusted. MODELS is here used for convenience, but TACS components will result in much faster performance. The Windsyn component requires the special request card ‘UM TO TACS’ so be able to do calculation performance parameters in TACS. This is added as a *User Specified/Additional* component. The parameters used and the type of controls may certainly be discussed, but the point here is to illustrate the interface between machine and control.

The speed control takes as input the actual speed of the machine (voltage at the TORQUE node of the machine) and gives out the torque to an additional current source connected to the same node.

The voltage control takes as input the phase A voltage to ground and gives out the field voltage to an additional voltage source. The example shows how to get the field current and initial field voltage into the ST1A exciter model. A separate model is used to calculate the rms value.

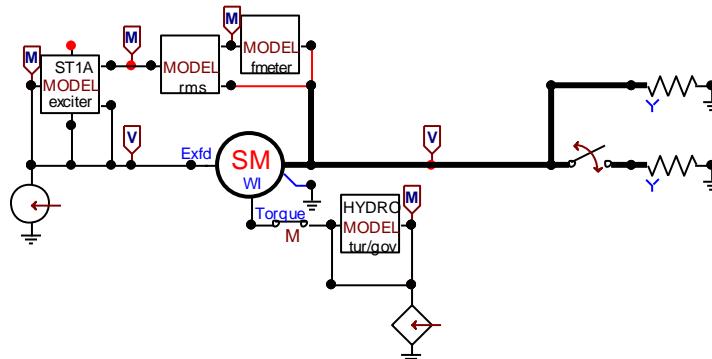


Fig. 6.47 – Machine control of Winsyn, auto-initialize synchronous machine (Exa\_17.acp).

#### 6.4.5.1 Hydro turbine governor

The gate opening limits must be adjusted to take the steady-state condition into account and  $G_{min}=-1$  is set in this case to allow 1 pu increase and reduction in torque. Also the initial head  $h_0$  is set to zero here.

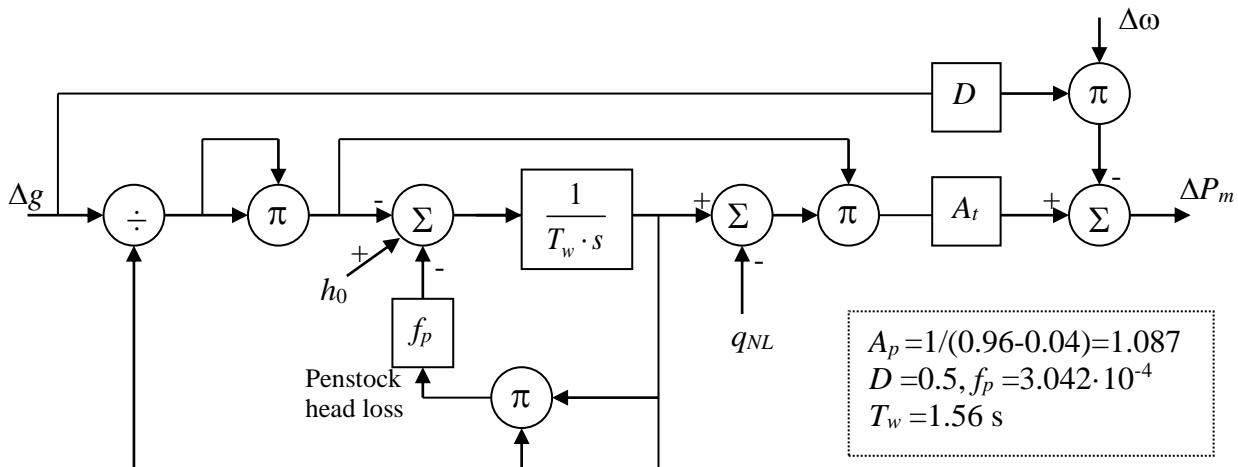
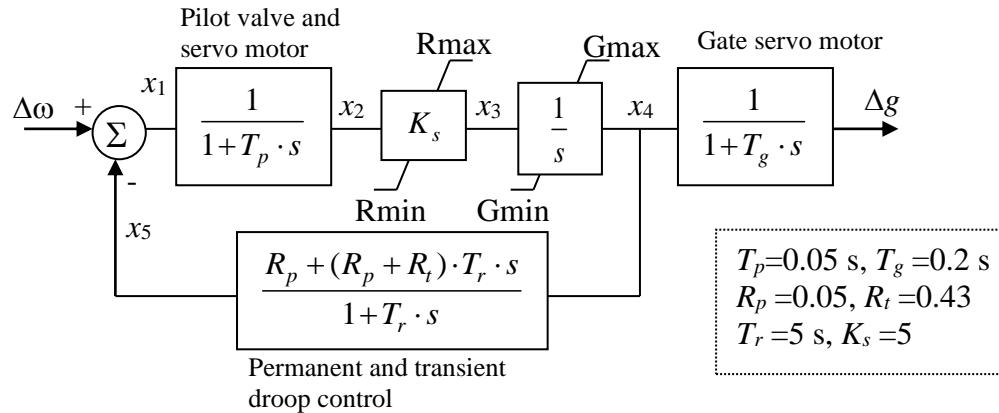


Fig. 6.48 – Hydro turbine governor

```

MODEL TUR_GOV
DATA Tw,D,gFL,fp,Rp,Tr,Rt,Tg,Tp,Ks
      Rmaxclose,Rmaxopen,Gmax,Gmin,MW,Wrated, Wref
INPUT W
OUTPUT Torque
VAR x1,x2,x3,x4,Pmech,At,x5,h,q,qNL,h0,s
      y1,y2,g,h1,Wrefpu,Wpu,torque
HISTORY
x1 {dflt:0},x2 {dflt:0},x3 {dflt:0},x4 {dflt:0}, x5 {dflt:0},
q {dflt:0},h {dflt:0}, y1 {dflt:0},y2 {dflt:0},g {dflt:0}
INIT
h0:=0 --Initial head. Set to zero in case of auto-initiation of generator.
At:=recip(gFL-gNL)
qNL:=(gNL)*sqrt(h0)
Wrefpu:=(Wref/Wrated)
ENDINIT
EXEC
Wpu:=(W/Wrated)*(30/pi)
--Governor hydraulic turbin
x1:=Wrefpu-Wpu-x5
cLaplace(x2/x1):=(1/s0)/(1+s0+Tp/s1)
x3:=(Ks*x2) {min:Rmaxclose max:Rmaxopen} --Gate opening/closing rate
cLaplace(x4/x3) {dmin:Gmin dmax:Gmax}:=(1/s0)/(1/s1) --Gate position
cLaplace(g/x4):=(1/s0)/(1+s0+Tg/s1) --Gate servo motor
cLaplace(x5/x4):=(Rp*s0+((Rp+Rt)*Tr)|s1)/(1+s0+Tr|s1) --Permanent, transient droop
--Hydraulic turbin

```

```

cLaplace(q/y1):=(1/s0)/(Tw/s1)                                --q=Flow
h:=(q*recip(g))**2                                         --h=Head
h1:=(q*q)*fp                                              --Penstock head loss
y1:=(h0-h-h1)                                            --Change in head
y2:=(q-qNL)*h                                           --Change in mechanical power
Pmech:=At*y2+(g*D*(Wrefpu-Wpu))                         --Uncomment to turn off turbine
Pmech:=g
Torque:=(Pmech*recip(W))*MW*1e6
ENDEXEC
ENDMODEL
    
```

#### 6.4.5.2 Exciter model

The Exciter is of type IEEE ST1A with inputs; terminal voltage  $V_T$ , field current  $I_{FD}$ , reference voltage  $V_{ref}$  and stabilizer signal  $V_S$  (all signals in pu). The Exciter IEEE DC1A is also implemented for comparison.

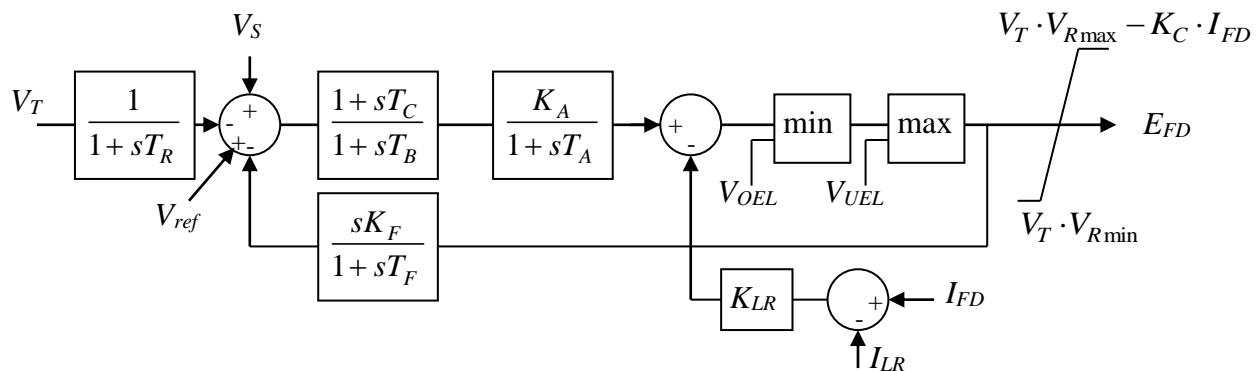


Fig. 6.49 – IEEE ST1A exciter. Parameters used;  $T_R=0.04$ ,  $T_B=10$ ,  $T_C=1$ ,  $K_A=190$ ,  $T_A=0$ ,  $T_F=1$ ,  $K_F=0$ ,  $K_{LR}=0$ ,  $I_{LR}=5$ ,  $V_{R\max}=7.8$ ,  $V_{R\min}=-6.7$ ,  $K_C=0.08$ .

The exciter model ST1A requires the field current as input. This variable can be obtained by specifying the MODELS node as Input Current and connect it directly to the EXFD terminal of the machine since there is a switch inside that measures the total field current.

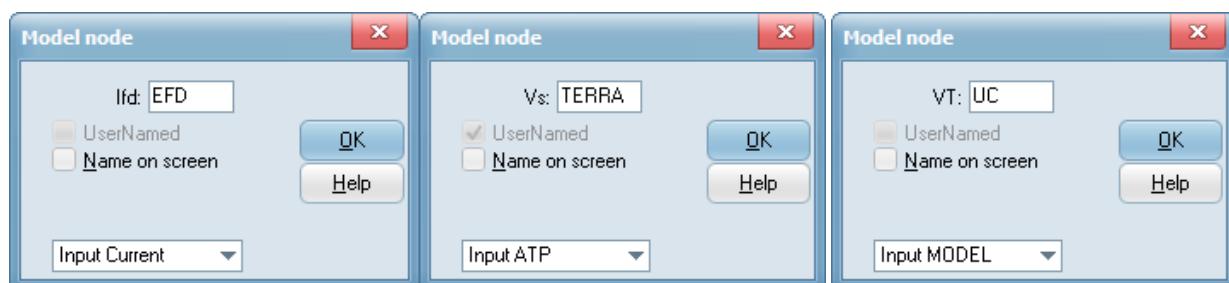


Fig. 6.50 – How to get the field current into Models, and how to specify the Vs and VT nodes.

```

MODEL EX_ST1A
DATA Vref, VTpu, Tr, Tc, Tb, Ka, Ta, Vuel, Voel, Klr, Ilr, Kf, Tf, VRmax, VRmin, Kc,
EFDref, IFDref
INPUT VT, Ifd, Vs, If0
OUTPUT Efd
VAR x1, x2, x3, x4, x5, x6, Efd, Vc, IFDpu, Efd0
HISTORY
x1 {dflt:0}, x2 {dflt:0}, x3 {dflt:0}, x4 {dflt:0}, x5 {dflt:0}, x6 {dflt:0},
Vc {dflt:0}, VT {dflt:0}
INIT
Efd:=0
    
```

```

IFDpu:=1
ENDINIT
EXEC
if T<2*timestep then --Special trick to obtain the initial field voltage
    Efd0:=-If0*0.01
else
    IFDpu:=-IFD/IFDref
    -Vc:=VT/(1+Tr)
    cLaplace(Vc/VT):=(1/VTpu*s0)/(1*s0+Tr*s1)
    cLaplace(x6/x5):=(Kf*s1)/(1*s0+Tf*s1)
    x1:=Vref-Vc-Vs-x6
    cLaplace(x2/x1):=(1*s0+Tc*s1)/(1*s0+Tb*s1)
    cLaplace(x3/x2):=(Ka*s0)/(1*s0+Ta*s1)
    x4:=x3- (IFDpu-Ilr)*Klr
    x5:=max(x4,Vuel)
    x5:=min(x5,Voel)
    Efd:=x5 {min:VT/VTpu*VRmin max:VT/VTpu*VRmax+Kc*IFDpu}
    Efd:=Efd*EFDRref+Efd0
endif
ENDEXEC
ENDMODEL
    
```

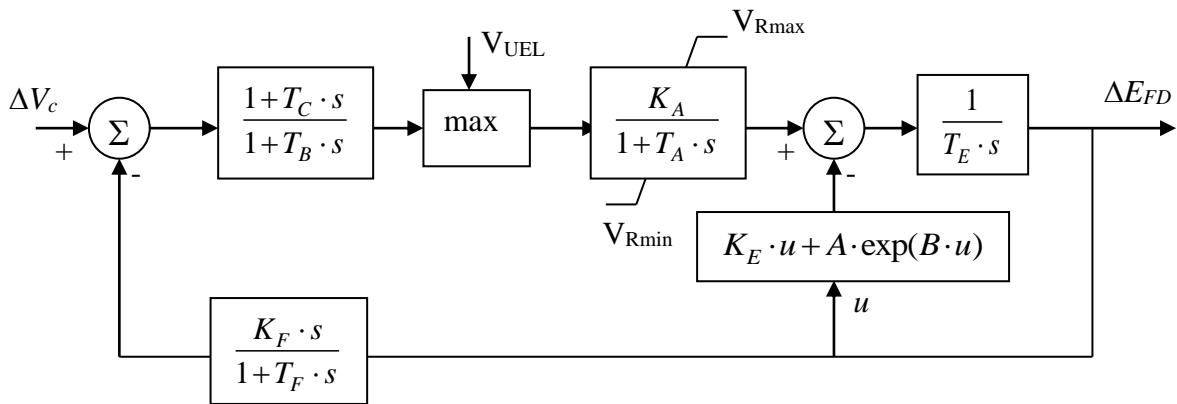


Fig. 6.51 – IEEE DC1A exciter. Parameters used;  $T_B = 0.06$ ,  $T_C = 0.173$ ,  $K_A = 400$ ,  $T_A = 0.89$ ,  $T_E = 1.15$ ,  $K_E = 1$ ,  $A = 0.014$ ,  $B = 1.55$ ,  $K_F = 0.058$ ,  $T_F = 0.62$ .

```

MODEL EX_DC1A
DATA Vref,Tc,Tb,Ka,Ta,VRMAX,VRMIN,Kf,Tf,Te,Ke,Vuel,A,B, Efdbase
INPUT Vc
OUTPUT Efd
VAR x1,x2,x3,x4,x5,x6, Vfe,Vf,Efd,Vcpu
HISTORY
x1 {dfilt:0},x2 {dfilt:0},x3 {dfilt:0},x4 {dfilt:0},x5 {dfilt:0}, x6 {dfilt:0}, Vfe {dfilt:0}
Vf {dfilt:0}
INIT
Efd:=0
ENDINIT
EXEC
Vcpu:=(Vc*sqrt(3) / (Vref*1000)) --Phase voltage measured so scale to line voltage
x1:=(1-Vcpu-Vf)
cLaplace(x2/x1):=(Tc*s1+1*s0) / (Tb*s1+1*s0)
x3:=max(x2,Vuel)
cLaplace(x4/x3) {dmin:VRMIN dmax:VRMAX}:=(Ka*s0) / (Ta*s1+1*s0)
x5:=(x4-Vfe)
cLaplace(x6/x5):=(1*s0) / (Te*s1)
Vfe:=x6*Ke+A*exp(B*x6)
cLaplace(Vf/Vfe):=(Kf*s1) / (Tf*s1+1*s0)
Efd:=x6*Efdbase
ENDEXEC
ENDMODEL
    
```

### 6.4.5.3 RMS value calculation

The RMS value is calculated by a standard models provided by Laurent Dube. Since the speed of the generator changes the frequency is calculated by another model. The *MODELS/Default model* option was used and the text simply pasted into the Model component. *Edit/Flip* was used to switch the input and outputs. As this model gives its output to another model it must be written first to the ATP file. This is managed by giving it a lower *Order* number than the receiving model and then choose *ATP/Settings/Format – Sort by Order*. In the receiving model the input node must be set to *Input MODEL*.

```

MODEL rms_meter
DATA xrms_ini {dflt:-1} -- initial rms value
INPUT freq -- monitored frequency
      x -- monitored signal
VAR xrms -- rms value of monitored signal
      x2 -- internal, x*x
      ix2 -- internal, integral of x2
      period -- 1/freq
OUTPUT xrms
DELAY CELLS(ix2): 1/2/timestep +1
INIT
  histdef(ix2) := 0
  integral(x2) := 0
  IF xrms_ini <0 THEN xrms:=0 ELSE xrms:=xrms_ini ENDIF
ENDINIT
EXEC
  period := recip(freq)
  x2 := x*x
  ix2 := integral(x2)
  IF t>period THEN
    xrms:= sqrt((ix2 - delay(ix2, period))/period)
  ENDIF
ENDEXEC
ENDMODEL

```

The frequency is calculated by another model based on zero-crossing detection.

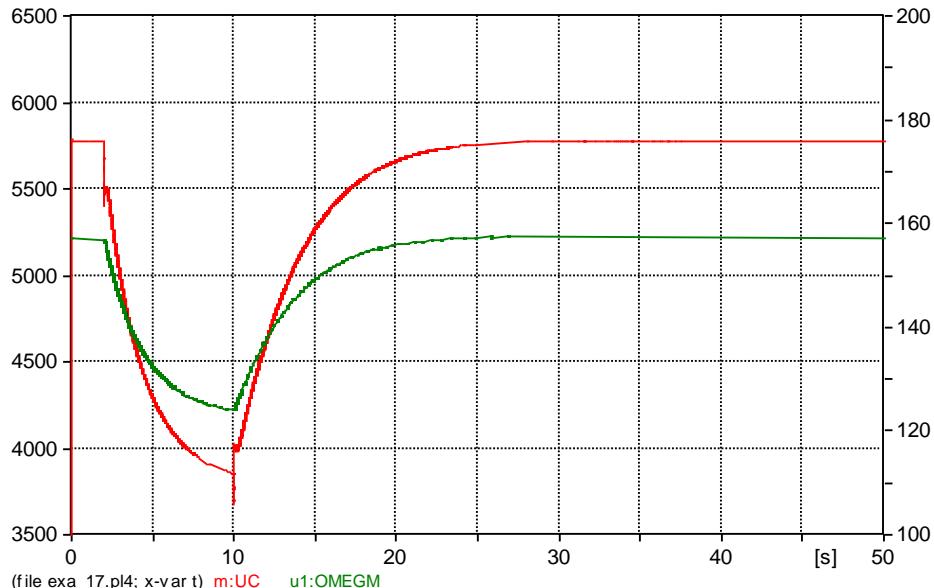


Fig. 6.52 – Machine response with no regulation

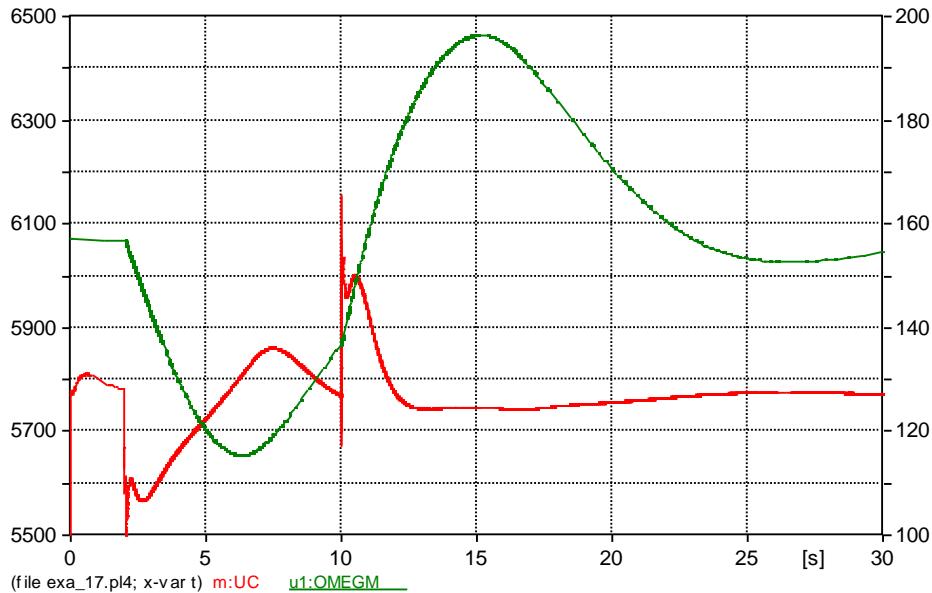


Fig. 6.53 – Machine response with exciter (DC1A) and governor (no hydro turbine).

## 6.5 Simulating transformer inrush current transients

The magnetic coupling between the windings and the nonlinear characteristic of the magnetizing reactance are the most important factors in transformer energizing transient studies. The BCTRAN supporting routine of ATP can be used to derive the R L or ( $L^{-1} R$ ) matrix representation of a single or 3-phase multi-winding transformer. ATPDraw now provides a similar interface to the BCTRAN supporting routine like the one provided for the LCC objects. The BCTRAN input data are the excitation and short circuit factory test data, which can easily be obtained from the transformer manufacturers. Additionally, the user can select between several options for modeling the nonlinear magnetizing branch.

The first example circuit of this section demonstrates the use of BCTRAN objects for transformer energization studies. In the second example, readers are familiarized with the application of *user specified objects* and the *Grouping* feature for transformer modeling.

### 6.5.1 Energization of a 400/132/18 kV auto-transformer (Exa\_10.acp)

The study case is the energization of a 3-phase, three-winding Yyd coupled transformer. The wye connected 132 kV windings and the delta coupled 18 kV windings are unloaded in this study. The schematic diagram of the simulated case is shown in Fig. 6.54, the corresponding ATPDraw circuit is depicted in Fig. 6.55.

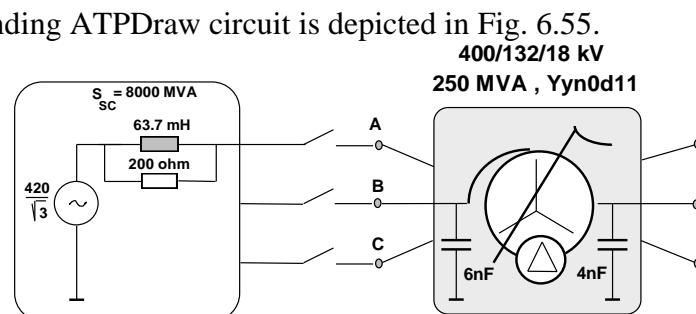


Fig. 6.54 - One-line scheme of the transformer and the 400 kV source.

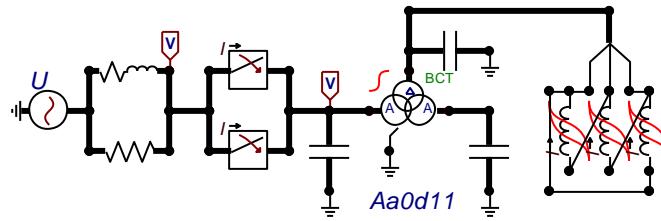


Fig. 6.55 - ATPDraw circuit (Exa\_10.acp).

The nameplate data of the transformer are as follows:

Voltage rating $V_{\text{high}}/V_{\text{low}}/V_{\text{tertiary}}$ :	400/132/18 kV, Yyn0d11	
Power rating:	250 MVA (75 MVA tertiary)	
Positive seq. excitation loss/current:	140 kW / 0.2 %	
Positive seq. reactance:	High to Low:	15 % ( $S_{\text{base}}=250\text{MVA}$ )
	High to Tertiary:	12.5 % ( $S_{\text{base}}=75\text{MVA}$ )
	Low to Tertiary:	7.2 % ( $S_{\text{base}}=75\text{MVA}$ )
Short circuit loss:	High to Low:	15 % ( $S_{\text{base}}=250\text{MVA}$ )
	High to Tertiary:	41.6667 % ( $S_{\text{base}}=250\text{MVA}$ )
	Low to Tertiary:	24 % ( $S_{\text{base}}=250\text{MVA}$ )
	High to Low:	710 kW
	High to Tertiary:	188 kW
	Low to Tertiary:	159 kW

In the *BCTRAN* dialog box, you specify first the number of phases and the number of windings per phase under *Structure* (see Fig. 6.56). Under *Ratings*, the nominal line-to-line voltage, power ratings, the type of coupling of windings and the phase shift must be entered. For auto-transformers, the nominal voltage of the *windings* (which is the required input for *BCTRAN*) is calculated automatically by ATPDraw and the short-circuit impedances are also re-defined according to the Eq. 6.45, 6.46, 6.50 of the EMTP Theory Book [5]. The zero sequence excitation and short circuit parameters are approximately equal to the positive sequence values for an auto-transformer having tertiary delta winding, so the *Zero sequence data available* check boxes are unselected in this example. The *External Lm* option is chosen under *Positive core magnetization* because external Type-96 hysteretic inductors are used to represent the magnetizing inductance. Accordingly, only the resistive component of the magnetizing current will be entered as *IEXPOS* in the *BCTRAN* input file.

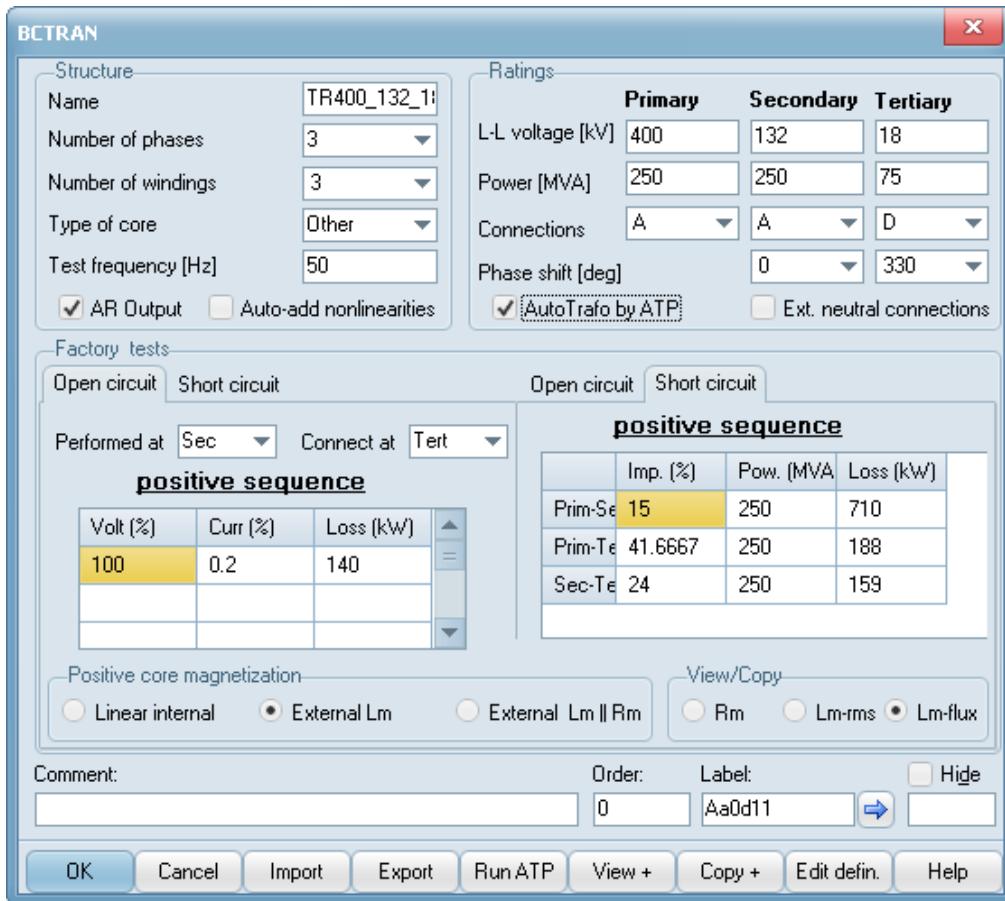


Fig. 6.56 - BCTRAN dialog box of the 400/132/18 kV transformer.

Following data specification the program offers to generate a BCTRAN input file and run ATP. It can either be performed by a *Run ATP* requests, (without leaving the dialog box), or selecting *OK*. If the BCTRAN-file is correct, a punch-file will be created. This file is directly included in the final ATP-file and there is no conversion to a library file as for lines/cables. The BCTRAN input file generated by ATPDraw is shown next. This file is given extension .atp and stored in the /BCT folder.

```

BEGIN NEW DATA CASE
ACCESS MODULE BCTRAN
$ERASE
C Excitation test data card
C < FREQ >< IEXPOS >< SPOS >< LEXPOS ><IEZERO >< SZERO ><LEXZERO ><><><><>
  3      50. .05600056    250.      140.          0 2 3 0
C Winding data cards
C <>< VRAT >< R ><>< PHASE1 >< PHASE2 >< PHASE3 >
  1 154.729872      H_BUSAH_BUSAH_BUSBL_BUSBH_BUSCL_BUSC
  2 76.2102355     L_BUSA      L_BUSB      L_BUSC
  3   18.          T_BUSAT_BUSCT_BUSBT_BUSAT_BUSCT_BUSB
C Short-circuit test data cards
C <>< PIJ >< ZPOSIJ >< SPOS ><ZZEROIJ >< SZERO ><><>
  1 2      710.33.4150145    250.33.4150145    250. 0 1
  1 3      188.61.3951637    250.61.3951637    250. 0 1
  2 3      159.           24.          250.           24.        250. 0 1
BLANK card ending short-circuit test data
$PUNCH
BLANK card ending BCTRAN data
BEGIN NEW DATA CASE
BLANK CARD

```

The nonlinear magnetizing branch of the 400/132/18 kV auto-transformer is represented by delta

coupled Type-96 hysteretic inductors in this study. The flux-current characteristic of these inductors can be obtained by means of the HYSDAT supporting routine of ATP. Fig. 6.57 shows the hysteresis loop of the Itype-1 material of ATP and of the magnetic core of the transformer.

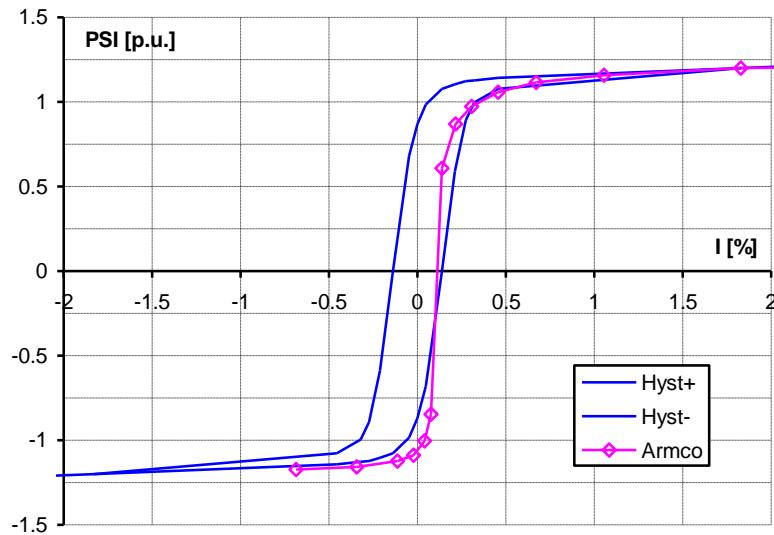


Fig. 6.57 - The shape of the hysteresis loop of the transformer magnetic core compared with the material type 1 of ATP's HYSDAT supporting routine.

The output file generated by the HYSDAT supporting routine is listed below. In this example the file is given a name HYSTR400.LIB and stored in the /USP folder.

```
C <++++++> Cards punched by support routine on 21-Jul-02 14.08.23 <++++++>
C HYSTERESIS
C $ERASE
C C ITYPE    LEVEL   { Request Armco M4 oriented silicon steel -- only 1 availab
C      1        4   { That was ITYPE=1. As for LEVEL=2, moderate accuracy outp
C      98.2     97.2   { Current and flux coordinates of positive saturat
-3.6825000E+01 -9.49129412E+01
-2.4550000E+01 -9.43411765E+01
-1.10475000E+01 -9.23400000E+01
-4.91000000E+00 -9.03388235E+01
-1.84125000E+00 -8.86235294E+01
 6.13750000E-01 -8.51929412E+01
 2.14812500E+00 -8.11905882E+01
 3.55975000E+00 -7.43294118E+01
 4.29625000E+00 -6.28941176E+01
 4.91000000E+00 -4.57411765E+01
 6.13750000E+00  3.05894118E+01
 6.75125000E+00  4.23105882E+01
 8.59250000E+00  5.71764706E+01
 1.10475000E+01  6.86117647E+01
 1.33797500E+01  7.43294118E+01
 1.74918750E+01  8.00470588E+01
 2.39362500E+01  8.51929412E+01
 3.28356250E+01  8.91952941E+01
 4.29625000E+01  9.20541176E+01
 6.13750000E+01  9.49129412E+01
 9.82000000E+01  9.72000000E+01
1.35025000E+02   9.77717647E+01
 9999.
```

Such a nonlinear characteristic can be connected to the Type-96 inductor in two ways: include as an external file, or enter flux-current data pairs directly in the *Characteristic* page as shown in Fig. 6.58. The *Copy* and *Paste* buttons of the dialog box provide a powerful way to import the

whole characteristic from an external text file via the Windows clipboard or export it to another Type96 objects. It is thus possible to bring a HYSDAT punch-file up in a text editor, mark the characteristic, copy it to the clipboard and paste it into the *Characteristic* page.

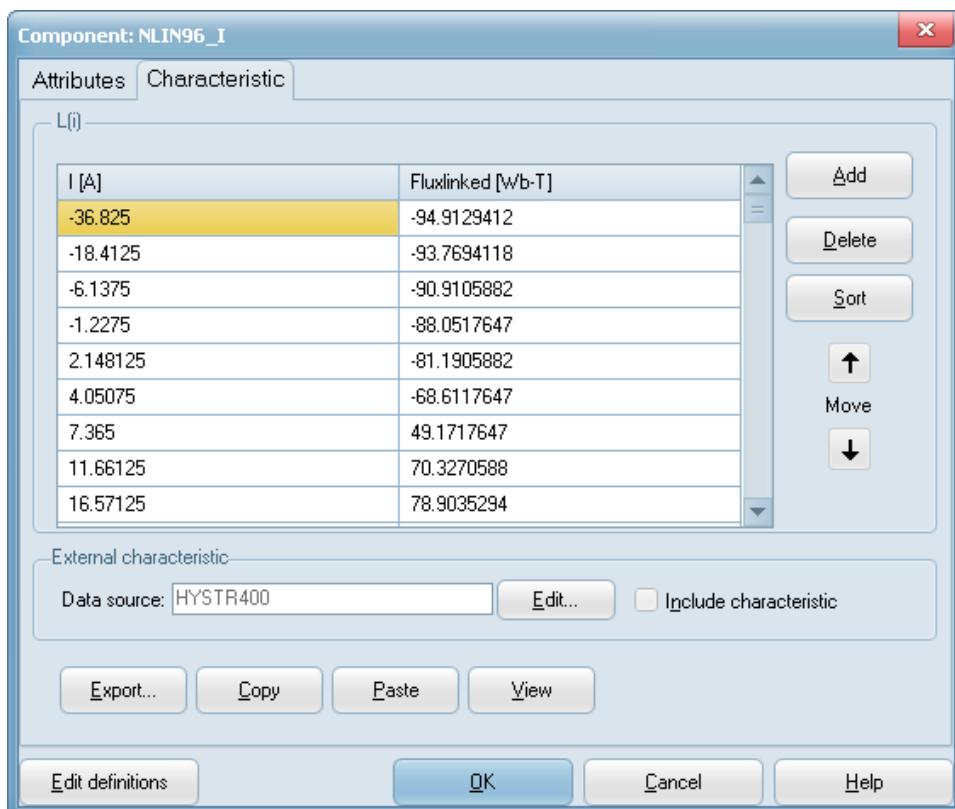


Fig. 6.58 - Importing the nonlinear characteristic from a HYSDAT punch-file.

The complete ATP input file generated by ATPDraw for this study case is listed next:

```
BEGIN NEW DATA CASE
C Generated by ATPDRAW August, Monday 26, 2019
C A Bonneville Power Administration program
C by H. K. Høidalen at SefAS/NTNU - NORWAY 1994-2019
C -----
$DUMMY, XYZ000
C   dT >< Tmax >< Xopt >< Copt >
  5.E-6    .15
      500      5     0     0      1     0     0      1     0
C       1      2     3     4      5     6     7     8
C 34567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R >< L >< C >
  L_BUSA          .004
  L_BUSB          .004
  L_BUSC          .004
SOURCASUPLA      2.  63.7
SOURCBSUPLB      2.  63.7
SOURCCSUPLC      2.  63.7
SOURCASUPLA      200.
SOURCBSUPLB      200.
SOURCCSUPLC      200.
T_BUSA           .01
T_BUSB           .01
T_BUSC           .01
96T_BUSBT_BUSC  8888.  0.0
      -36.825    -94.9129412
      -18.4125    -93.7694118
```

-6.1375	-90.9105882	
-1.2275	-88.0517647	
2.148125	-81.1905882	
4.05075	-68.6117647	
7.365	49.1717647	
11.66125	70.3270588	
16.57125	78.9035294	
24.55	85.7647059	
36.21125	90.3388235	
56.465	93.7694118	
98.2	97.2	
135.025	97.7717647	
9999		
96T_BUSAT_BUSB	8888. 0.0	1
-36.825	-94.9129412	
-18.4125	-93.7694118	
-6.1375	-90.9105882	
-1.2275	-88.0517647	
2.148125	-81.1905882	
4.05075	-68.6117647	
7.365	49.1717647	
11.66125	70.3270588	
16.57125	78.9035294	
24.55	85.7647059	
36.21125	90.3388235	
56.465	93.7694118	
98.2	97.2	
135.025	97.7717647	
9999		
96T_BUSCT_BUSA	8888. 0.0	1
-36.825	-94.9129412	
-18.4125	-93.7694118	
-6.1375	-90.9105882	
-1.2275	-88.0517647	
2.148125	-81.1905882	
4.05075	-68.6117647	
7.365	49.1717647	
11.66125	70.3270588	
16.57125	78.9035294	
24.55	85.7647059	
36.21125	90.3388235	
56.465	93.7694118	
98.2	97.2	
135.025	97.7717647	
9999		
H_BUSA	.006	0
H_BUSB	.006	0
H_BUSC	.006	0
\$VINTAGE, 1,		
1T_BUSAT_BUSC	6942.8436268432	
2T_BUSBT_BUSA	0.0	
	6942.8436268432	
3T_BUSCT_BUSB	0.0	
	0.0	
	6942.8436268432	
USE AR		
1H_BUSAL_BUSA	3.2888630659697 .42462348721612	
2L_BUSA	-7.231251366149 0.0	
	34.681001957452 .09492595191772	
3T_BUSAT_BUSC	2.3450004639366 0.0	
	-84.67537379274 0.0	
	338.34949508527 0.0	
4H_BUSBL_BUSB	.1936225317E-15 0.0	
	-.677127449E-15 0.0	
	.1202491824E-14 0.0	
	3.2888630659697 .42462348721612 0.0	
5L_BUSB	-.677127449E-15 0.0	
	.2041578689E-14 0.0	

```

          -.282318606E-14      0.0
          -7.231251366149      0.0
          34.681001957452 .09492595191772
6T_BUSBT_BUSA          .1202491824E-14      0.0
          -.282318606E-14      0.0
          -.6542678427E-4      0.0
          2.3450004639366      0.0
          -84.67537379274      0.0
          338.34949508527      0.0
7H_BUSCL_BUSC          .1936225317E-15      0.0
          -.677127449E-15      0.0
          .1202491824E-14      0.0
          .1936225317E-15      0.0
          -.677127449E-15      0.0
          .1202491824E-14      0.0
          3.2888630659697 .42462348721612
8L_BUSC               -.677127449E-15      0.0
          .2041578689E-14      0.0
          -.282318606E-14      0.0
          -.677127449E-15      0.0
          .2041578689E-14      0.0
          -.282318606E-14      0.0
          -7.231251366149      0.0
          34.681001957452 .09492595191772
9T_BUSCT_BUSB          .1202491824E-14      0.0
          -.282318606E-14      0.0
          -.6542678427E-4      0.0
          .1202491824E-14      0.0
          -.282318606E-14      0.0
          -.6542678427E-4      0.0
          2.3450004639366      0.0
          -84.67537379274      0.0
          338.34949508527      0.0
$VINTAGE, 0,
$UNITS, -1.,-1.
    USE RL
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CLOP >< type >
    SUPLA H_BUSA      -1.      .045      1.                  1
    SUPLB H_BUSB      -1.      .045      1.                  1
    SUPLC H_BUSC      -1.      .045      1.                  1
    SUPLA H_BUSA      .0735     1.
    SUPLB H_BUSB      .0785     1.
    SUPLC H_BUSC      .0785     1.
/SOURCE
C < n 1><>< Ampl. >< Freq. ><Phase/T0>< A1 >< T1 >< TSTART >< TSTOP >
14SOURCA 0   326600.      50.              -1.      1.
14SOURCB 0   326600.      50.     -120.              -1.      1.
14SOURCC 0   326600.      50.      120.              -1.      1.
/INITIAL
/OUTPUT
    SUPLA SUPLB SUPLC H_BUSAH_BUSBH_BUSC
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK INITIAL
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK

```

Some results of the simulation are shown in Fig. 6.59. In the reported case, the steady state magnetizing current of the unloaded transformer is interrupted at 45 ms producing high residual flux in two phases. As a result, a high amplitude inrush current may occur at a subsequent transformer energization.

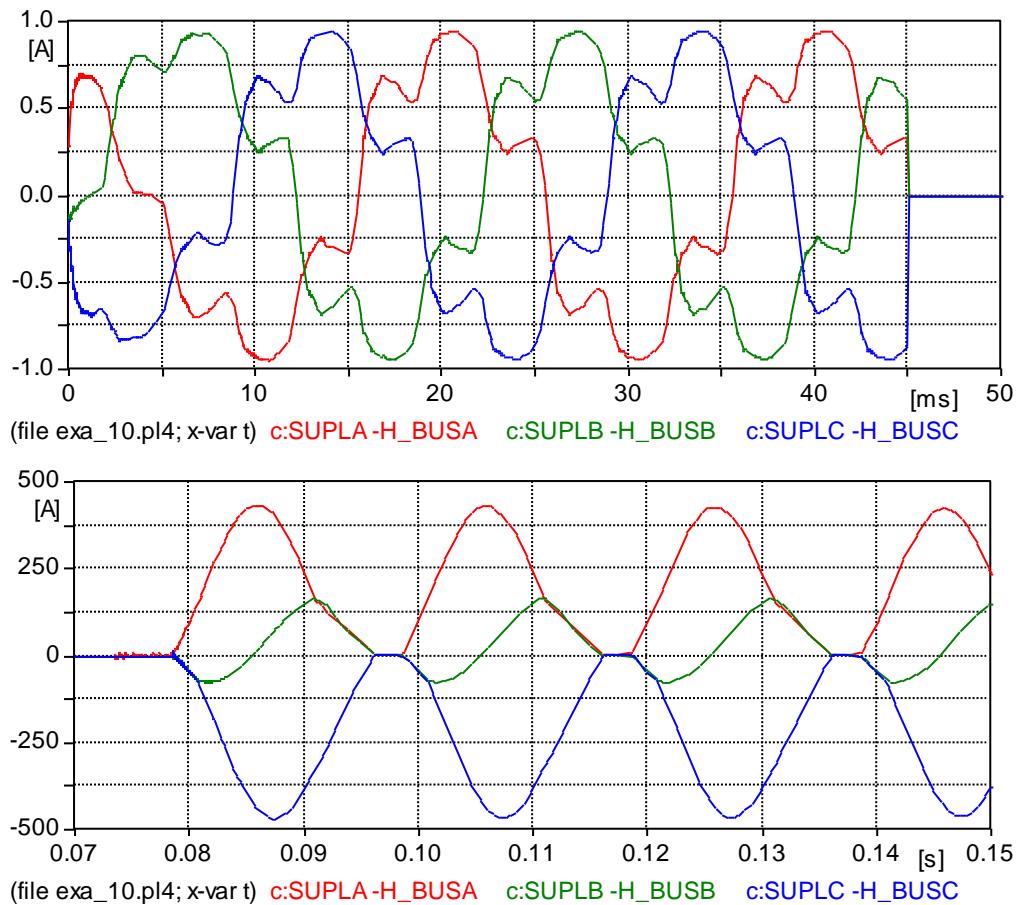


Fig. 6.59 – Steady-state magnetizing current (upper curves) and the inrush current (lower curves) at a subsequent energization.

### 6.5.2 Energization of a 132/15 kV generator step-up transformer (Exa\_11.acp)

The use of the icon customization and the advantages of the grouping feature of ATPDraw are demonstrated in this example. The simulated case is again a transformer switching study, in which a 155 MVA 132/15 kV Y/d coupled step-up and a 4 MVA 15/6.9 kV D/d coupled auxiliary transformer are energized together. The fast start gas turbine plant is located near to a 400/220/120 kV substation and the transformers are connected with the substation by a 120 kV single core XLPE cable. During the step-up transformer energization, the generator is still disconnected, so it need not to be considered in this study. The ATPDraw circuit of the simulation is shown in Fig. 6.60.

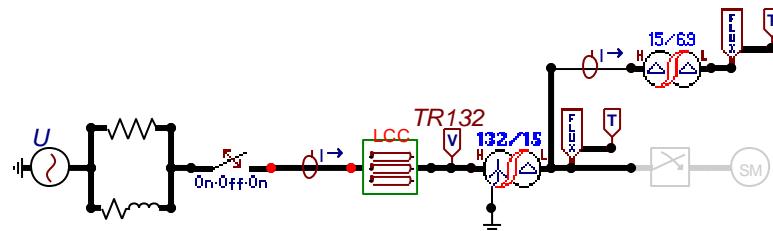


Fig. 6.60 – ATPDraw circuit (Exa\_11g.acp)

Fig. 6.60 shows several customized ATPDraw objects created by the *Edit / Compress* command. If you are not familiar with this grouping feature, please read in section 5.1 of this Advanced Manual. This feature provides a powerful tool in advanced modeling. On Fig. 6.60 the nonlinear,

hysteretic transformer objects, the parallel connected 3-phase breakers and the TACS objects for flux measurement were compressed into single objects, and the icon of each group has been customized, as well. The icon of some non-group objects were also customized, e.g. the LCC object of the XLPE cable. The uncompressed version of this case is also part of the ATPDraw's example collection and is shown in Fig. 6.61. Therefore, you can see how the grouping feature makes the circuit more readable.

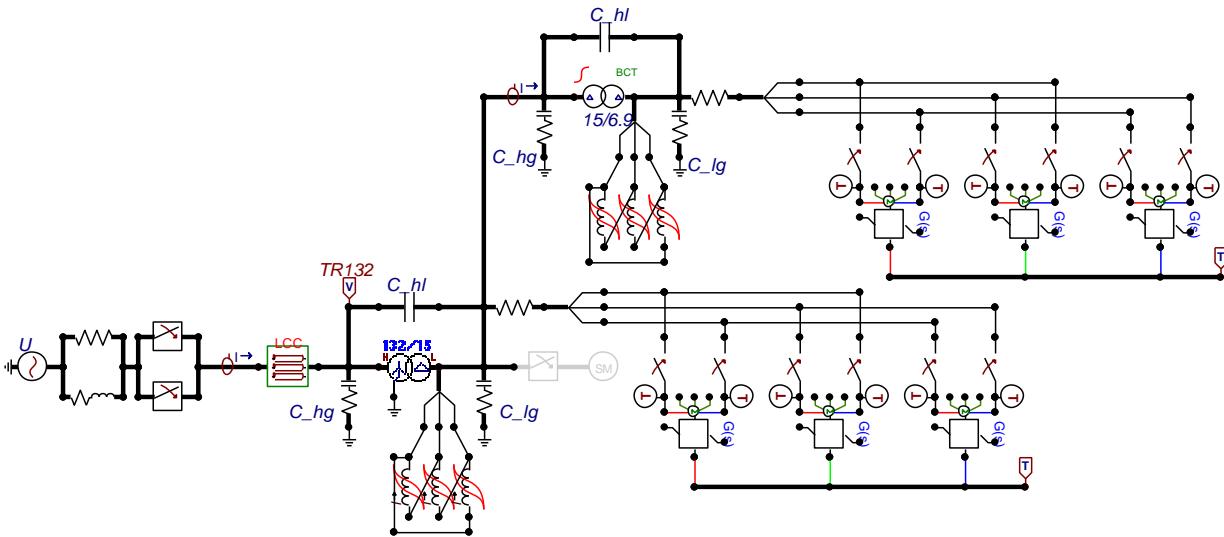


Fig. 6.61 – ATPDraw circuit without using compress (Exa\_11.acp).

The model of the Ynd11 and the Dd0 transformers consists of a linear part (user specified library object or BCTRAN object) and a nonlinear hysteretic inductor. The capacitances between the transformer windings and ground have been considered, as well. These capacitances do not influence the inrush current significantly, but they need to be considered especially at delta coupled transformer terminals to avoid “floating subnetwork found” simulation errors. For more details about the model parameters please read in section 5.8.2 of the Advanced Manual.

The compress option of ATPDraw can be used effectively to create new probe-type objects, as well. The 3-phase *Flux probe* of this example has been constructed by integrators (*TACS / Transfer functions / General*) objects, time-controlled switches (to set zero initial conditions) and coupling to TACS objects. The output of the *Flux probe* (the instantaneous flux linkage of the transformer windings) can be used to analyze the operation of the model during steady state no-load conditions, and during the transformer de-energization/re-energization, as shown in Fig. 6.62.

The circuit breaker of the transformer has a common drive with mechanical phase shift of 60 electrical degrees. The making sequence is A-C-B with 3.33 ms delay between the poles and the breaking sequence is B-C-A. Some results of the simulation obtained by the elaborated model are shown next. Fig. 6.63 shows the flux linkage and the phase-to-ground voltages of the step-up transformer during the no-load breaking process. The residual flux is quite low in all phases, thus a subsequent energization will not produce high amplitude inrush current even if the making is done at the voltage zero crossing. When synchronizing the first pole to close with the bus voltage and energize the transformer close to the voltage peak, the inrush current amplitude will not exceed the peak value of the nominal load current of the transformer (see in Fig. 6.64).

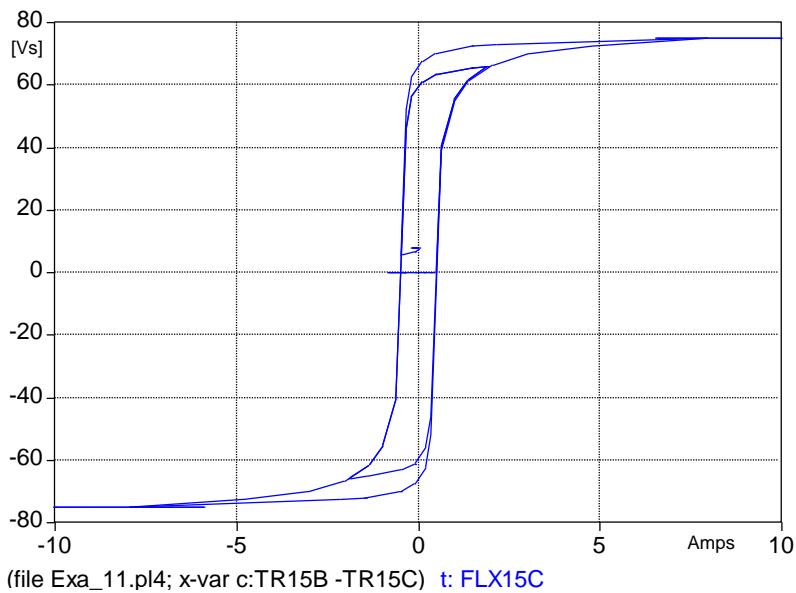


Fig. 6.62 – Roaming of the operating point on the hysteresis loop in steady-state and during the subsequent non-sinusoidal oscillations at transformer de-energization.

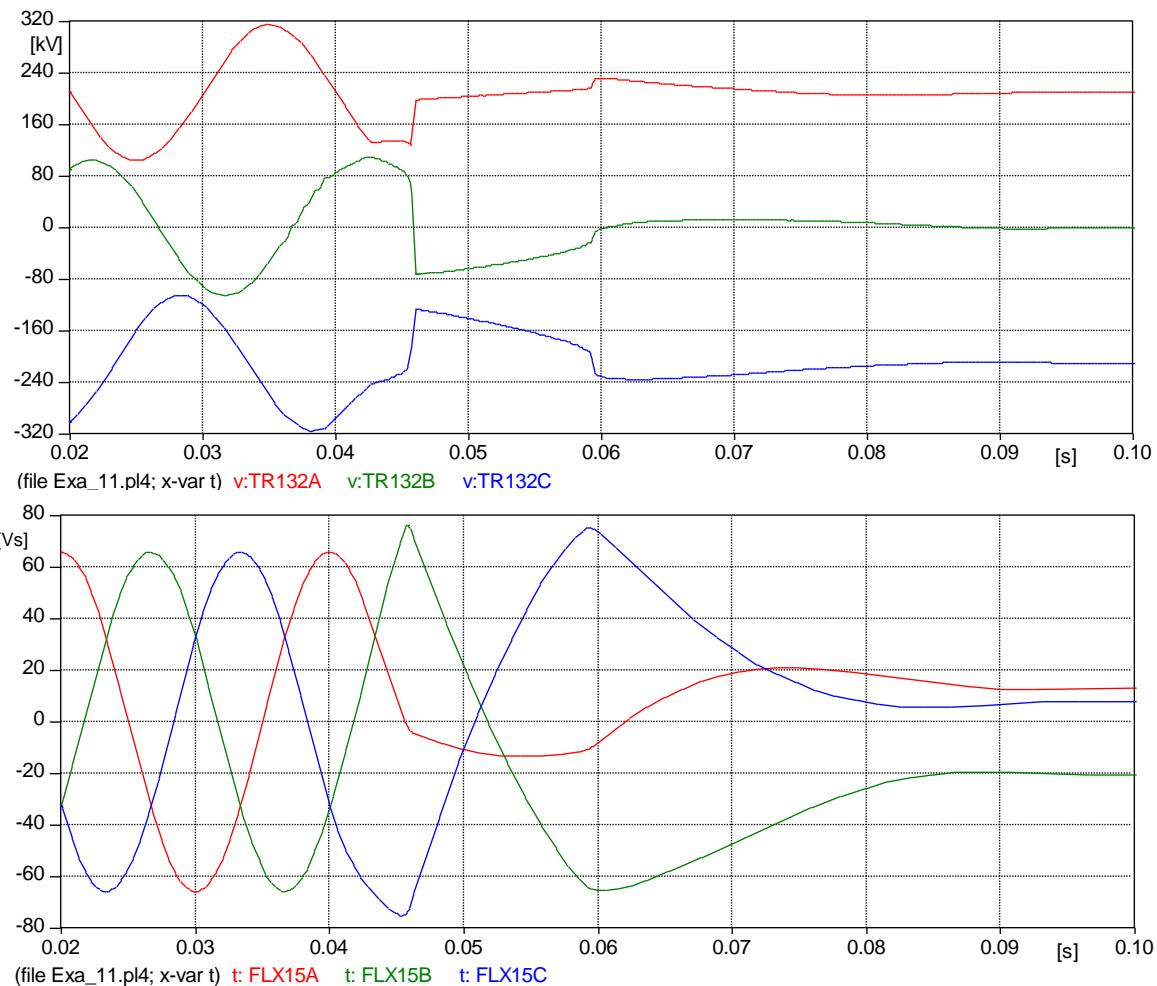


Fig. 6.63 – Non-sinusoidal voltage oscillations appear after de-energizing the step-up transformer (upper curves). The residual flux is less than 30% in each phases (lower curves).

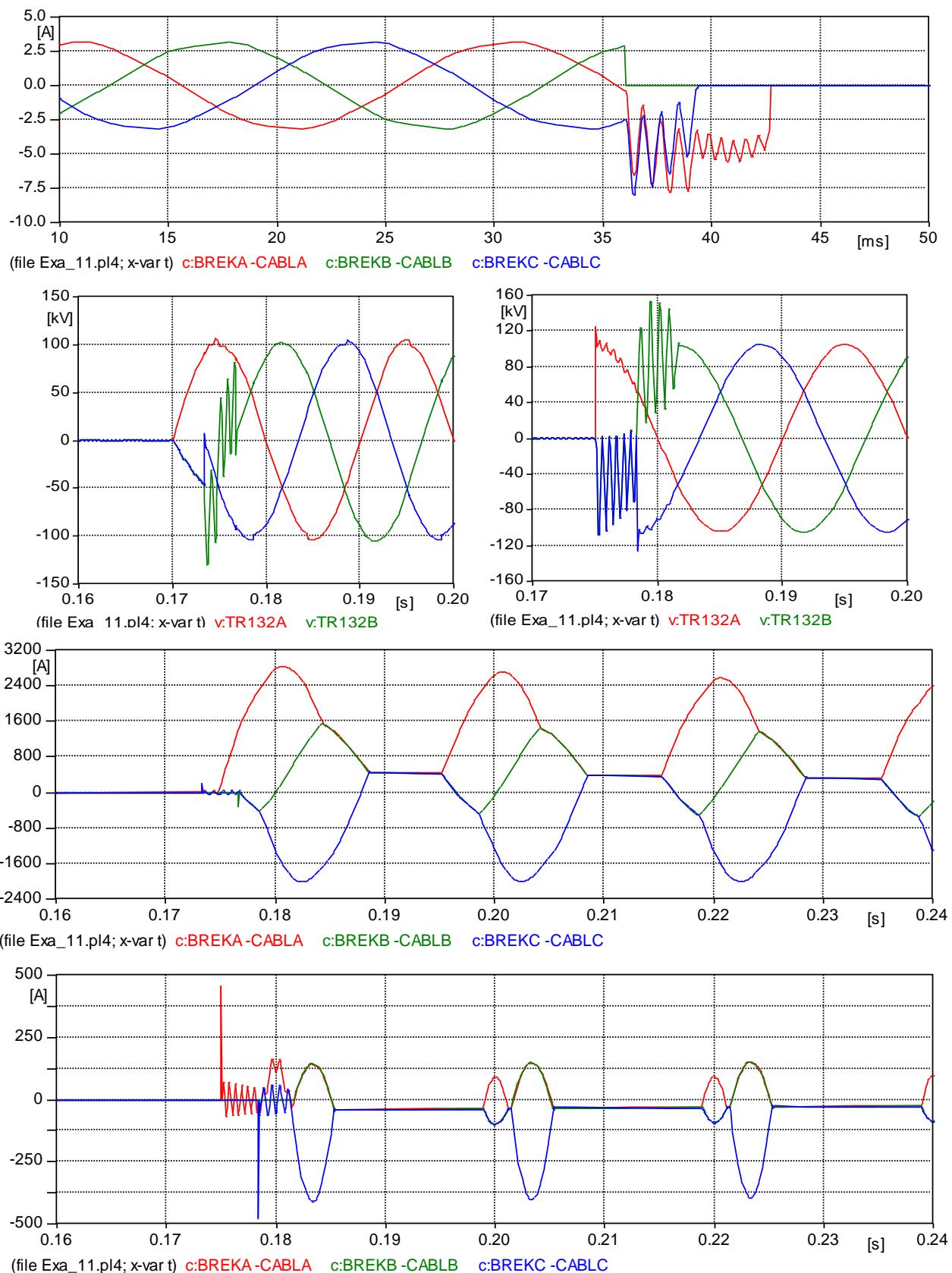


Fig. 6.64 – Interrupting the steady state no-load current of the step-up transformer (upper curves) and the inrush current amplitude (below) when energizing the first pole of the breaker:  
a) at the voltage zero crossing, b) close to the voltage peak.

### 6.5.3 Using the Hybrid Transformer component (Exa\_16.acp)

The Hybrid Transformer component (XFMR) provides a topologically correct core model with individual saturation characteristics in legs and yokes calculated based on relative core dimensions. Further the saturation characteristic is based on the Frolich equation with an additional, optional air-core inductance thus improving the response above the last test report value. This is of great importance when it comes to over-excitation situations like inrush current simulations. The XFMR component in version 5.6 offers type 96 inductances even if these are not recommended for transient studies. This gives on the other hand residual flux in the core after de-energization. In general, advance Models controlled hysteretic inductors are needed to give good inrush current predictions.

Fig. 6.65 shows the XFMR input dialog for the example Exa\_16.acp. A 3-legged stacked core is selected and this requires relative yoke dimensions to be given under Core data. A Triplex core (single phase units) does not require relative dimensions. Under *Inductance* and *Core* the short and open circuit test report data are given, respectively (*Resistance* automatically follows *Inductance* for Test Report data). The *Winding sequence* is set with the low-voltage winding as the inner. The XFMR dialog can work test report data directly. Creation of the saturation characteristics is automated (for type 96 half of the core losses is assigned to hysteresis losses with a Steinmetz coefficient  $n=2$ , and a uniform width of the hysteresis).

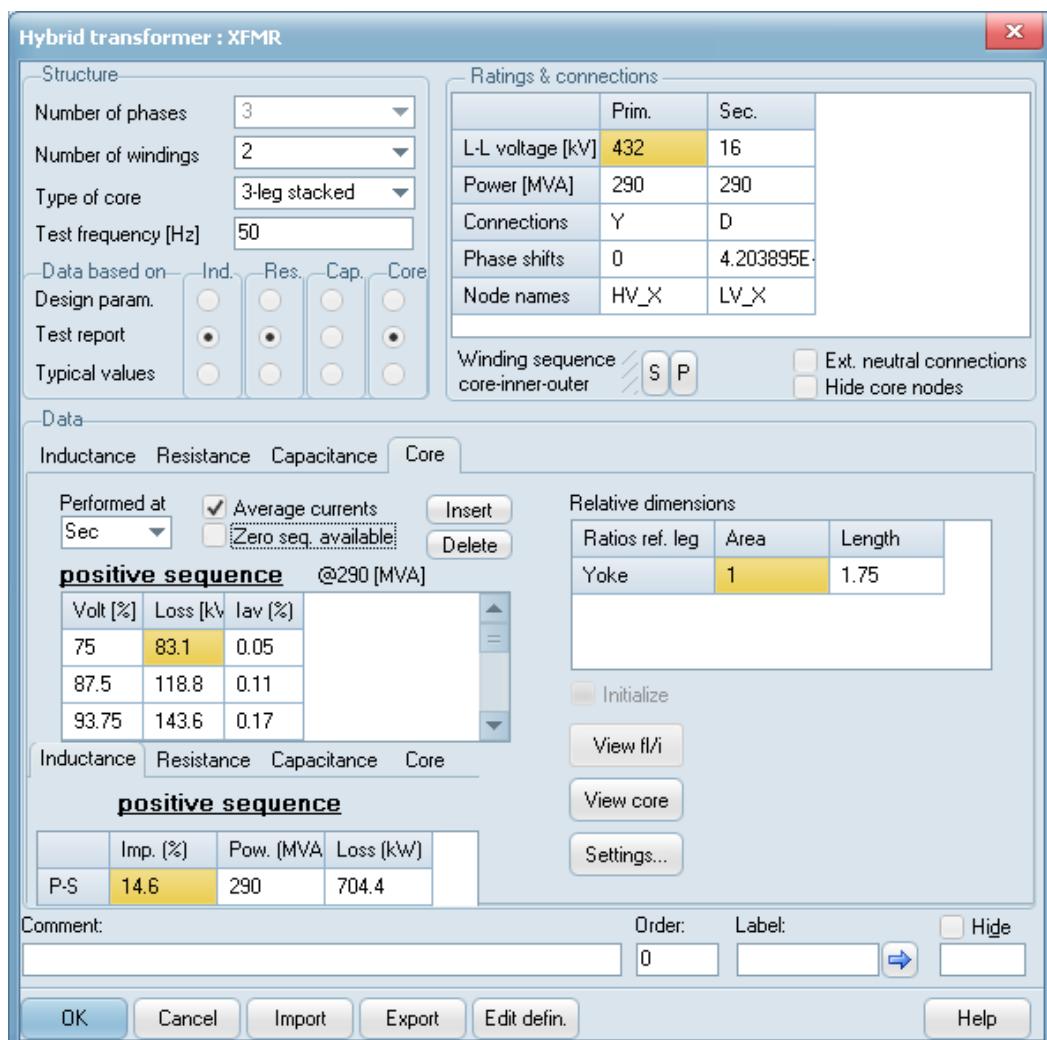
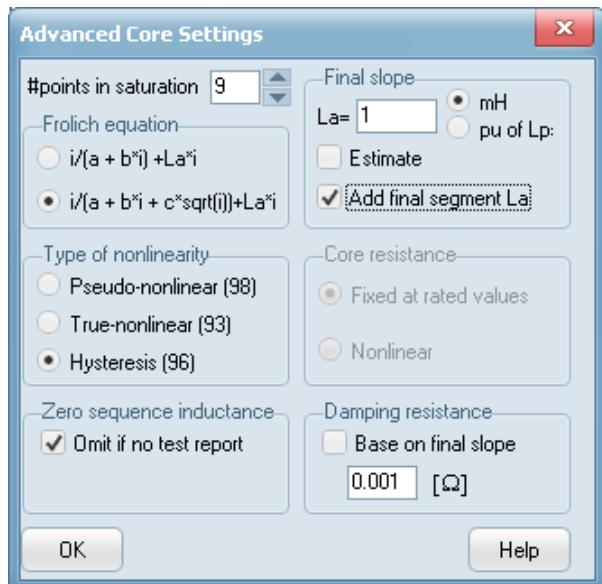


Fig. 6.65 – XFMR model example Exa\_16.acp

When the user clicks on OK ATPDraw performs an internal calculation of the leakage inductance in the same way as BCTRAN. The winding resistances are connected outside the A-matrix, however. The core model is fitted to the Test Report rms values by a Gradient Method optimization routine.



The user should also click on the *Settings...* button on the *Core* page to select the type of nonlinear inductance (98, 93, or 96) and the number of points on linearized Frolich equation (maximum 9). A high number is required to get good inrush current estimates. The final slope inductance (part of the air-core inductance) is set to zero in this case. Design data really required to estimate it. Using the *Estimate* check box will estimate  $La=\mu_0 \cdot 6/a'$  where the factor  $a=6$  is typical for core material M4 and  $a'$  is found from the optimization (with ' $=0$ ')

Fig. 6.66 – Core settings.

Fig. 6.67 shows a simulated inrush currents switching in a 290 MVA transformer from the 16 kV side with zero residual flux. The same transformer is modeled both in BCTRAN and XFMR and the comparison shows that the XFMR gives about four times higher inrush currents. This is because the BCTRAN model incorrectly assumes linear extrapolation of the magnetization characteristic above the Test Report data. In addition the currents into the XFMR model have more reasonable waveshapes and attenuation.

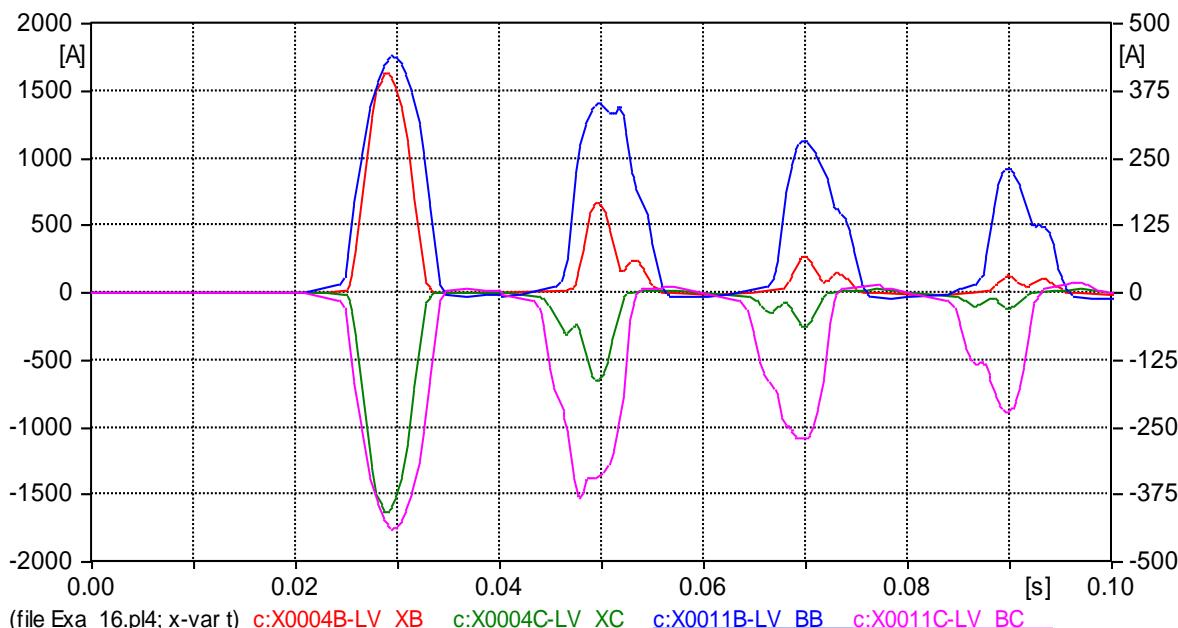


Fig. 6.67 – Comparison of inrush currents (zero residual flux) for a 290 MVA transformer modeled in BCTRAN and XFMR.

## 6.6 Switching overvoltage studies with statistical approach (Exa\_12.acp)

The switching impulse withstand level of EHV line insulators are generally lower than the lightning impulse withstand level. Therefore, some measures are needed to protect the line against switching overvoltages, especially when the insulation level is rather low, like in case of line uprating. One or more of the following measures could be applied to reduce these overvoltages:

- mounting surge arresters at the line terminals and along the line
- application of circuit breaker with closing resistors
- synchronizing the breaker operations at line energization and reclosing
- limiting or eliminating the trapped charge at dead time of the 3-phase reclosing

The influence of the latter two measures to the switching overvoltage distribution is analyzed in this example. The use of the master/slave feature of ATP's statistical switches is also introduced.

The EMTP model shown in Fig. 6.68 has been elaborated for a line upgrading feasibility study to analyze the switching performance of a 400 kV compact line. The clearances, the location of the phase- and ground wires, and the length of the composite insulator strings are assumed known in this example.

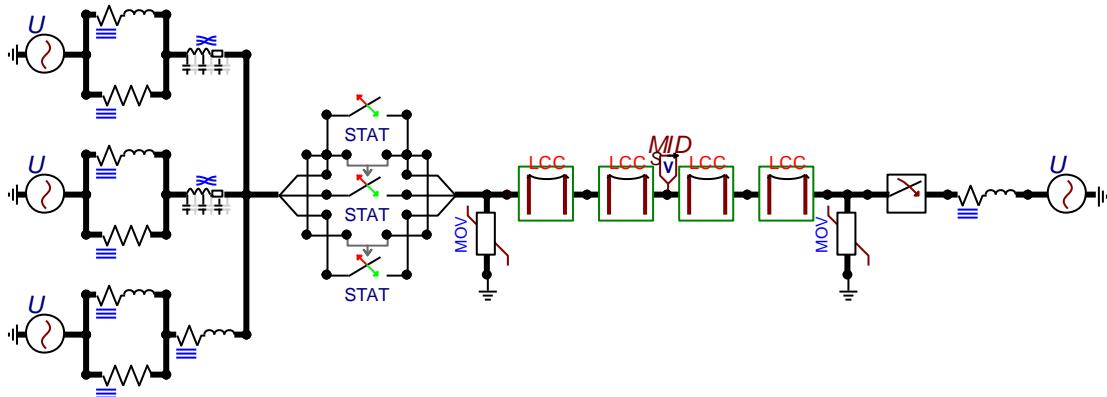


Fig. 6.68 – ATPDraw circuit for the statistical switching study (Exa\_12.acp).

The investigated line has been divided into four sections, each of them represented by an LCC JMartí object. To set up the initial conditions of the line easily, a 3-phase voltage source is connected to the line at right having voltage amplitude equal to the desired trapped charge. This source is disconnected before the operation of the statistical switches to make the line unloaded. It is worth to mention that some care is needed when constructing the EMTP model for such a statistical simulations, because the unnecessary over-complication of the model may increase the overall simulation time of that many statistical runs significantly.

### 6.6.1 Setting program options for the statistical simulation

The simulated switching incidence is a 3-phase reclosing in this study. Statistical switches of Gaussian-type represent the reclosing breaker. The master/slave dependency is now supported by ATPDraw, thus phase A is specified as *master* and the remaining two as *slave*. ATP requires the master switch be specified earlier in the ATP-file than a slave. ATPDraw ensured automatically this ordering. This is why the closing of the dialog box of a master switch is somewhat delayed.

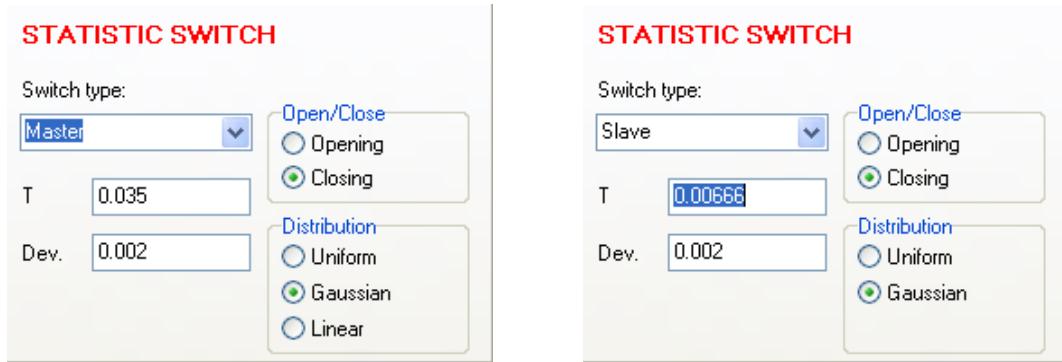
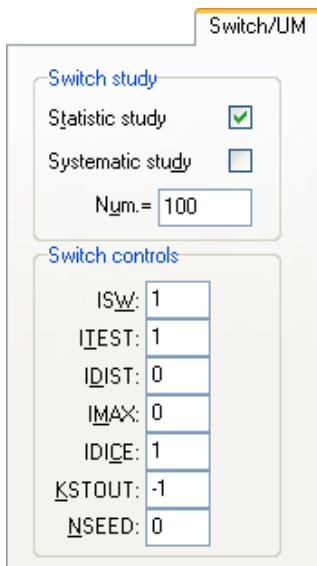


Fig. 6.69 – Input parameters of master and slave statistical switches.



The rest of program options and circuit parameter settings for a statistical study is very similar to that of any other time domain simulations. There is one addition however. You need to specify the *Switch study* and *Switch controls* under ATP / Settings / Switch before generating the ATP-file.

Unless you need special settings, the *Switch controls* parameters need not be modified.

Fig. 6.70 – Setting the parameters of the statistical study.

The **Output Manager** found under ATP/Output Manager (F9) enables the user to select those output requests to be added to the statistical tabulation. The user can also group and scale the output requests. Example 12 requests as default only output of the MID voltage, but the terminal voltages and for instance surge arrester energy can be added. The selection of alternative statistical tabulation is shown in Fig. 6.72.

## 6.6.2 Results of the statistical study

As worst-case assumption the fault, which precedes the 3-phase reclosing in one or more phases has not been considered here. Taking that the inductive voltage transformers play a significant role in eliminating the trapped charge in the healthy phases during the dead time of reclosing, but CVTs or CCVT has no such effect, two different cases have been considered:

- a1) the trapped charge is equal to the phase to ground voltage peak
- a2) the trapped charge is 30% of the phase to ground voltage peak.

The reclosing operations are synchronized to the bus voltage in this simulation. It means that the master switch is closed when the instantaneous value of the phase-to-ground bus voltage is equal to zero. The average delay for the slave switches in phase B and C is set 120 and 60 electrical degrees, respectively. The standard deviation of the operating time of the synchronous controller and the breaker has been considered as an additional parameter in the study:

- b1) accumulated deviation of the breaker and the controller operating time is 1 ms  
 b2) accumulated deviation of the breaker and the controller operating time is 2 ms.

The statistical tabulation of the overvoltage distribution will be part of the LIS-file, as shown next:

```

1 ) -----
Statistical output of node voltage 0.3430E+06 |0 MIDA MIDB MIDC
Statistical distribution of peak voltage at node "MIDA".
The base voltage for per unit printout is V-base = 3.43000000E+05
  Interval      voltage      voltage in      Frequency   Cumulative      Per cent
  number     in per unit    physical units (density)   frequency   .GE. current value
    51        1.2750000    4.37325000E+05       0           0          100.000000
    52        1.3000000    4.45900000E+05       2           2          98.000000
  .....
    87        2.1750000    7.46025000E+05       1           99         1.000000
    88        2.2000000    7.54600000E+05       1          100         .000000
Summary of preceding table follows:      Grouped data      Ungrouped data
                                         Mean = 1.66850000E+00  1.66882696E+00
                                         Variance = 3.85116162E-02 3.81739314E-02
                                         Standard deviation = 1.96243767E-01 1.95381502E-01
  .....
4 ) -----
  SUMMARY      SUMMARY
4 ) -----
The following is a distribution of peak overvoltages among all output nodes of the last data card that have
the same base voltage.
This distribution is for the maximum of the peaks at all output nodes with V-base = 3.43000000E+05
  Interval      voltage      voltage in      Frequency   Cumulative      Per cent
  number     in per unit    physical units (density)   frequency   .GE. current value
    51        1.2750000    4.37325000E+05       0           0          100.000000
    52        1.3000000    4.45900000E+05       1           1          99.000000
  .....
    91        2.2750000    7.80325000E+05       1           99         1.000000
    92        2.3000000    7.88900000E+05       1          100         .000000
Summary of preceding table follows:      Grouped data      Ungrouped data
                                         Mean = 1.77125000E+00  1.77305706E+00
                                         Variance = 5.25173611E-02 5.27332819E-02
                                         Standard deviation = 2.29166667E-01 2.29637283E-01

```

Finally, a brief summary of the simulation results is given next. Considering the metal-oxide arresters with 2 p.u. protection level at both ends of the line, the highest overvoltages appear in the inner points of the line. As an example, Fig. 6.71 shows the probability distribution functions of the switching overvoltages arising in the middle of the line. The four curves correspond to the following cases:

- Three phase reclosing with 30% trapped charge. Standard deviation of the accumulated operating time of the synchronous controller and the breaker is 1 ms.
- Three phase reclosing with 100% trapped charge. Standard deviation of the accumulated operating time of the synchronous controller and the breaker is 1 ms.
- Three phase reclosing with 30% trapped charge. Standard deviation of the accumulated operating time of the synchronous controller and the breaker is 2 ms.
- Three phase reclosing with 100% trapped charge. Standard deviation of the accumulated operating time of the synchronous controller and the breaker is 2 ms.

As it can be seen, the reclosing overvoltages are quite low even if the trapped charge is close to the voltage peak, if the reclosing operations are synchronized to the bus-side voltage zero by a point on wave controller.

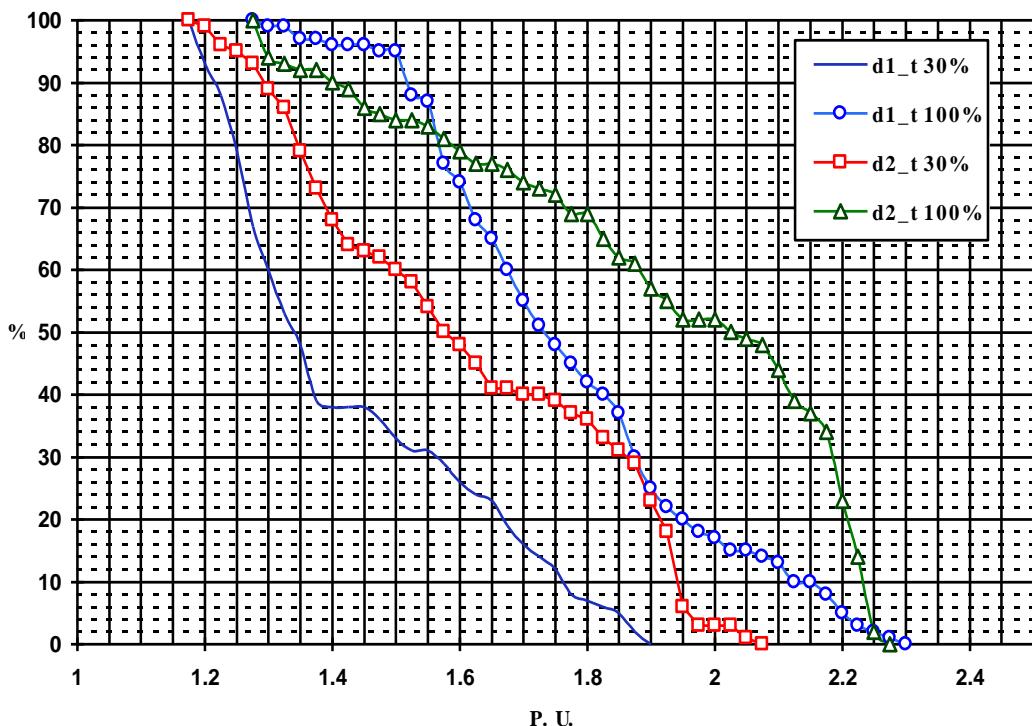


Fig. 6.71- Probability distribution function of the 3-phase reclosing overvoltages.

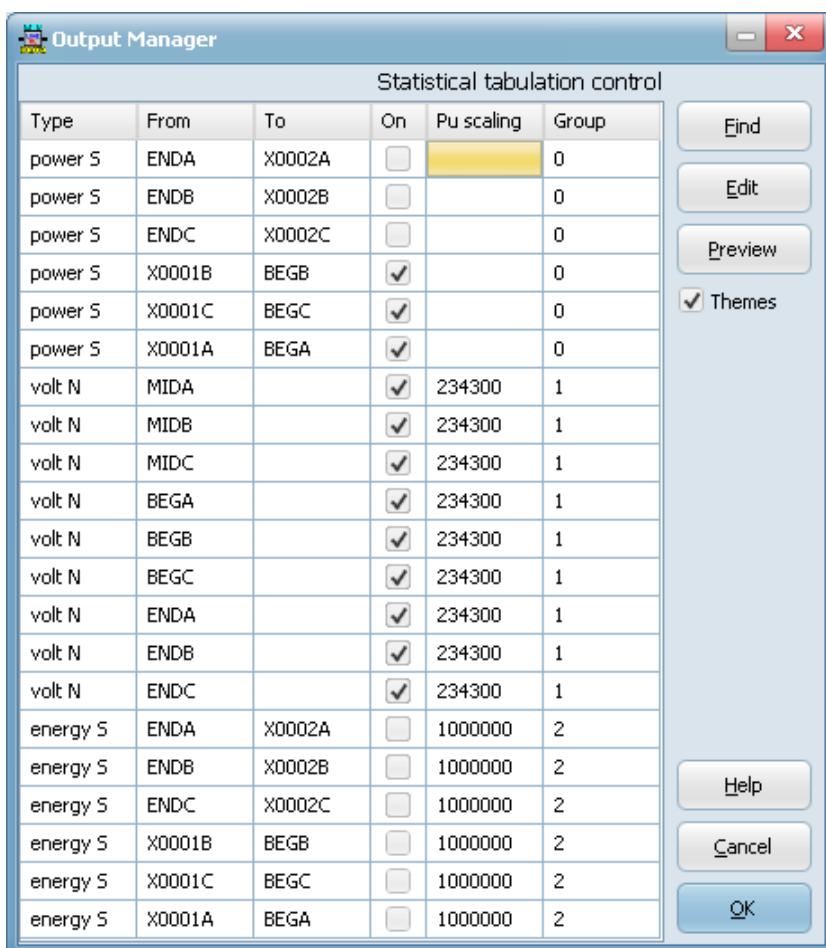


Fig. 6.72 – Output Manager and alternative request of Statistical Tabulation.

## 6.7 Power system protection (*Exa\_24.acp*)

This example illustrates the usage of the Power System Toolbox in ATPDraw with modeling of the IEEE 9 Bus system with distance protection units. The LINE3 component is used to represent the lines as PI-equivalent and various fault scenarios. Voltage and current probes are used to show steady state voltage and load flow.

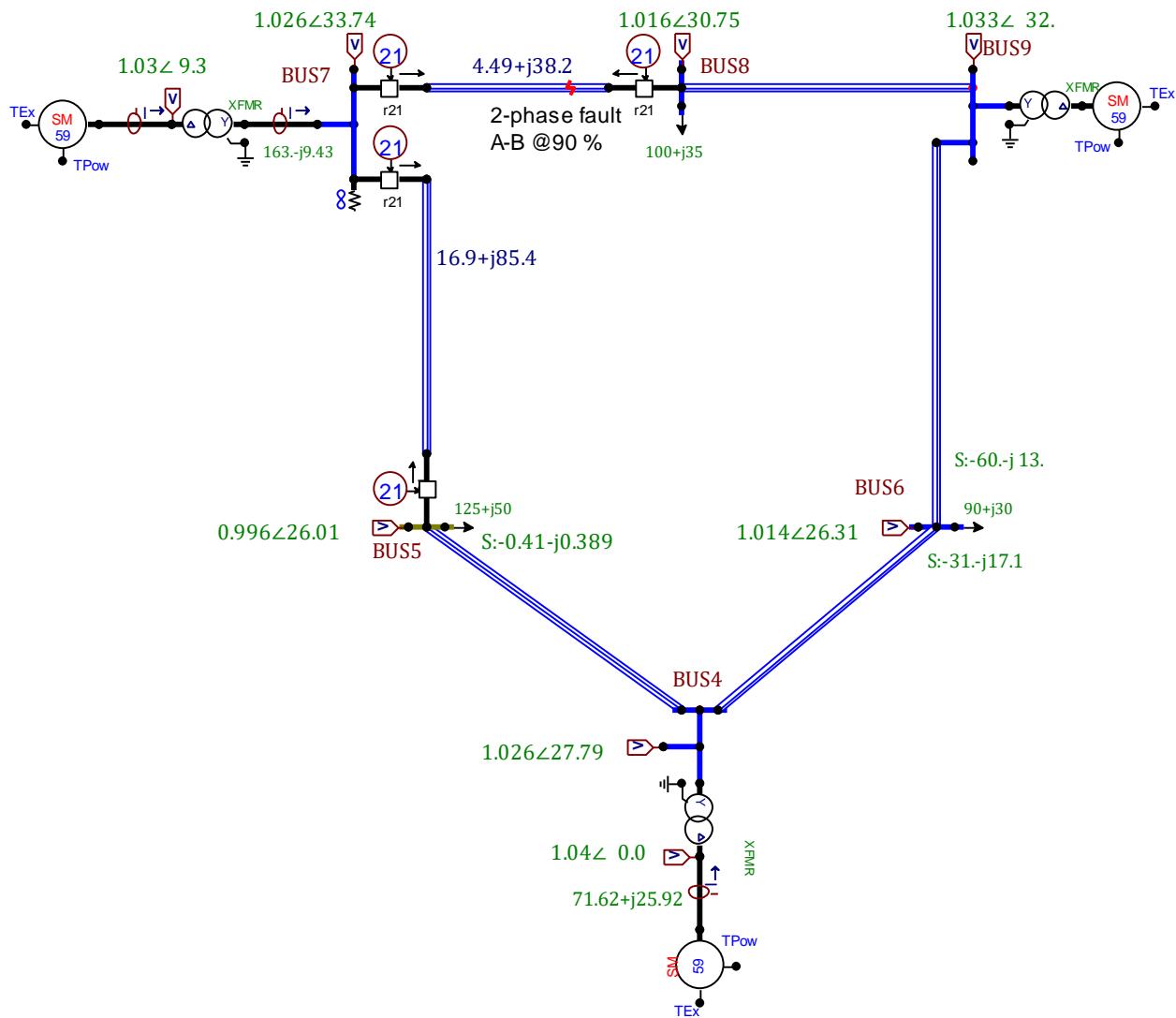


Fig. 6.73 – IEEE 9 Bus system modeled in ATPDraw with Power System Toolbox, *Exa\_24.acp*.

The most sophisticated part of the model is the protective relays represented by a group R21. No data is surfaces so all settings and readout must be done inside the group. Selecting the relay at Bus7 facing Bus 8 and *Edit/Edit group*, gives the content shown in Fig. 6.74. All the relays are equal in topology. The two relays at each side of the line are also equal in parameters but flipped in direction (take note of the important icon arrow). Relays have different settings for each line. The relay group R21 consists of four Models and a Group that must come in the correct sequence (*Sidebar's Object Inspector Tree* used to arrange the sequence correctly) starting with the two low-pass filters for the measured voltages and currents, then an RX-calculator for the loop impedance estimation, next a distance relay, and then finally a Group consisting of a controlled circuit breaker (current zero detection).

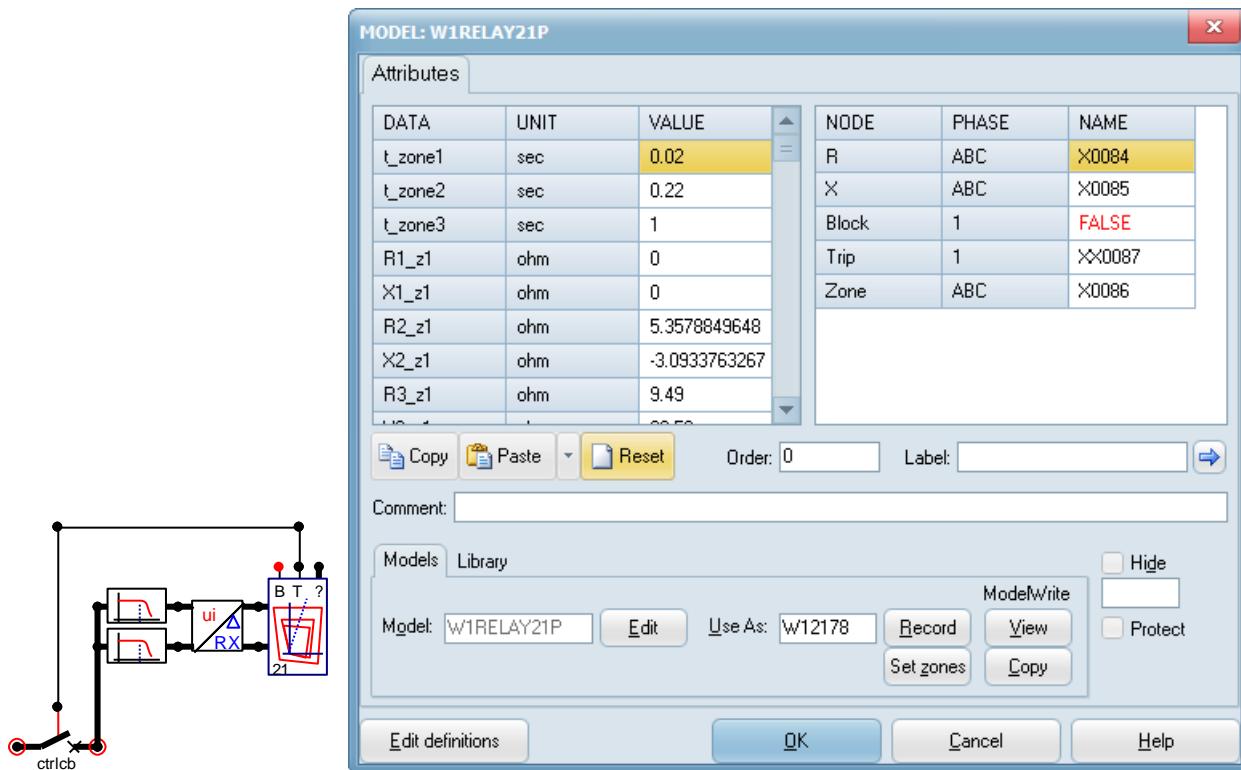


Fig. 6.74 – Content of distance relay group (left) with input dialog of the distance relay (right).

The relay settings are the most complicated, but a Zone Helper, shown in Fig. 6.75, is available via *Set zones* that let the user work with reach and blinder angles instead of RX values in a polygon. To take full advantage of the Zone Helper the RL and XL settings in the relay should correspond to the actual positive sequence impedance of the involved line. Default settings is based on using 80 and 120 % of the line length for reach X of zone 1 and 2, the rightmost blinder angle equal to the line impedance angle and the other angles equal to 30 degrees.

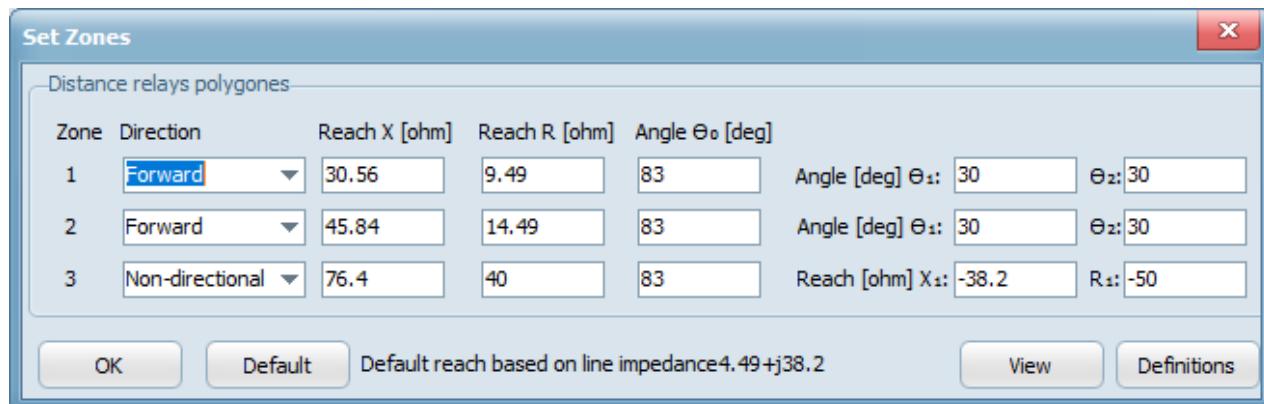


Fig. 6.75 – Zone Helper, distance relay W1RELAY21P

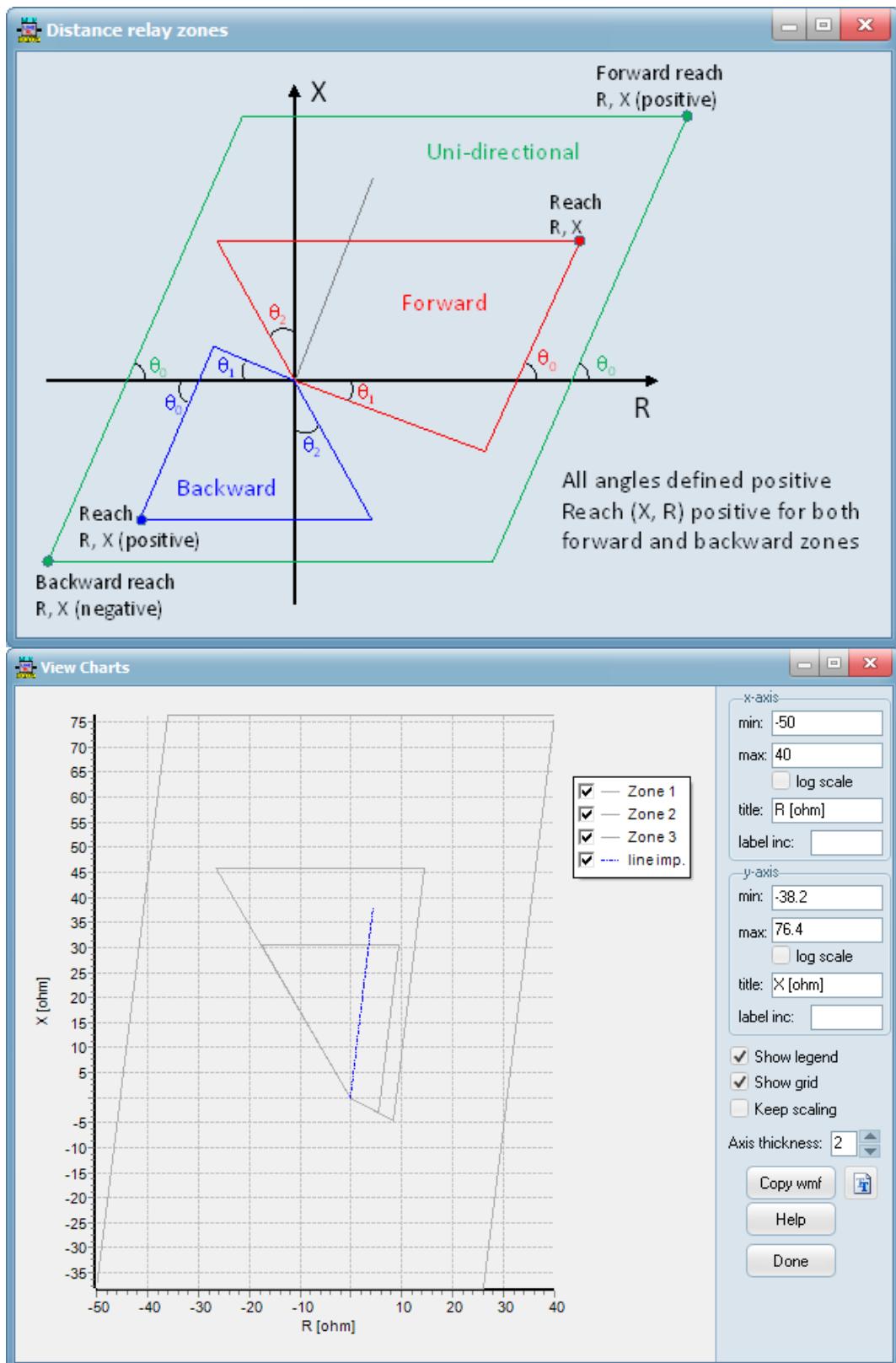


Fig. 6.76 – Zone definition and view from the Zone Helper for relay W1RELAY21P

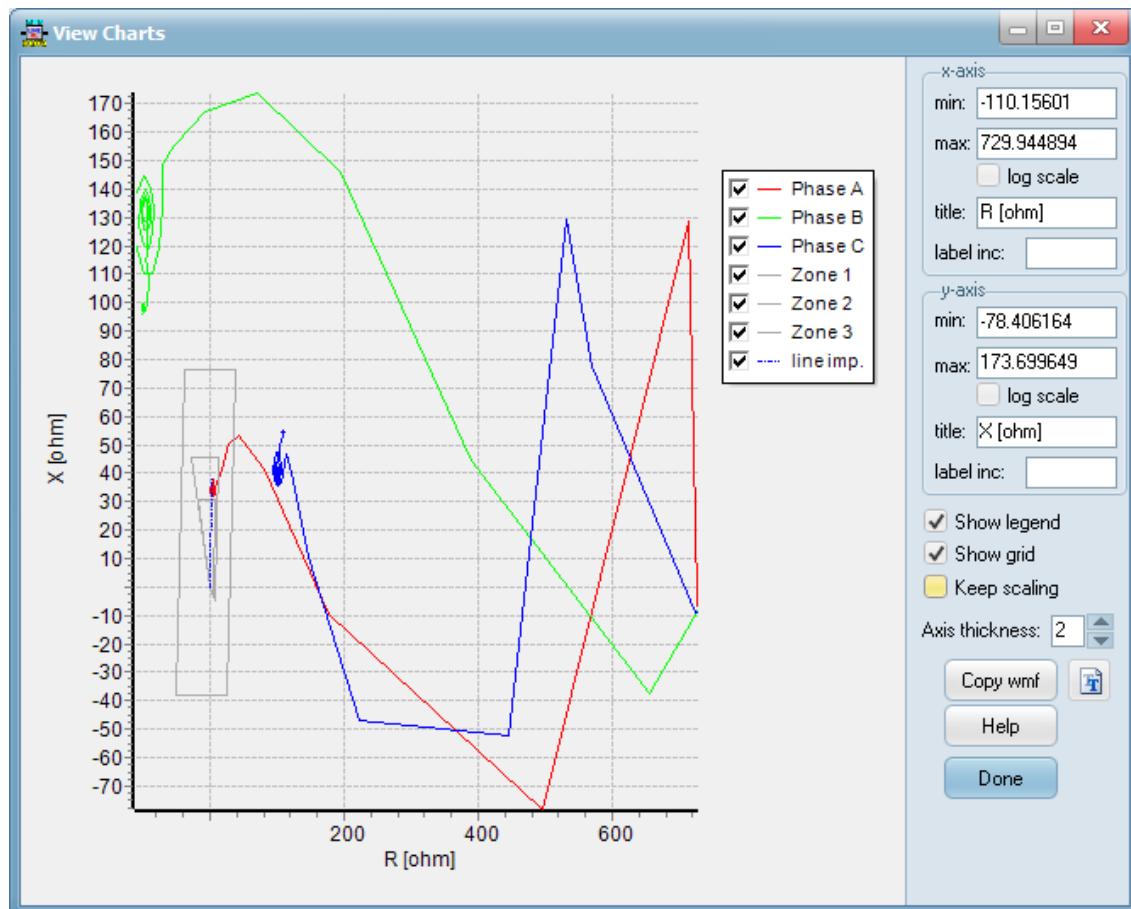


Fig. 6.77 –Relay @Bus7 towards Bus8 impedance trajectories, click on *View* in Fig. 6.74. Phase A, B, C notation in legend means loop 1, 2, 3. The figure shows that the impedance of loop 1 enters zone 2, while the others stays outside. This means a phase A to B fault.

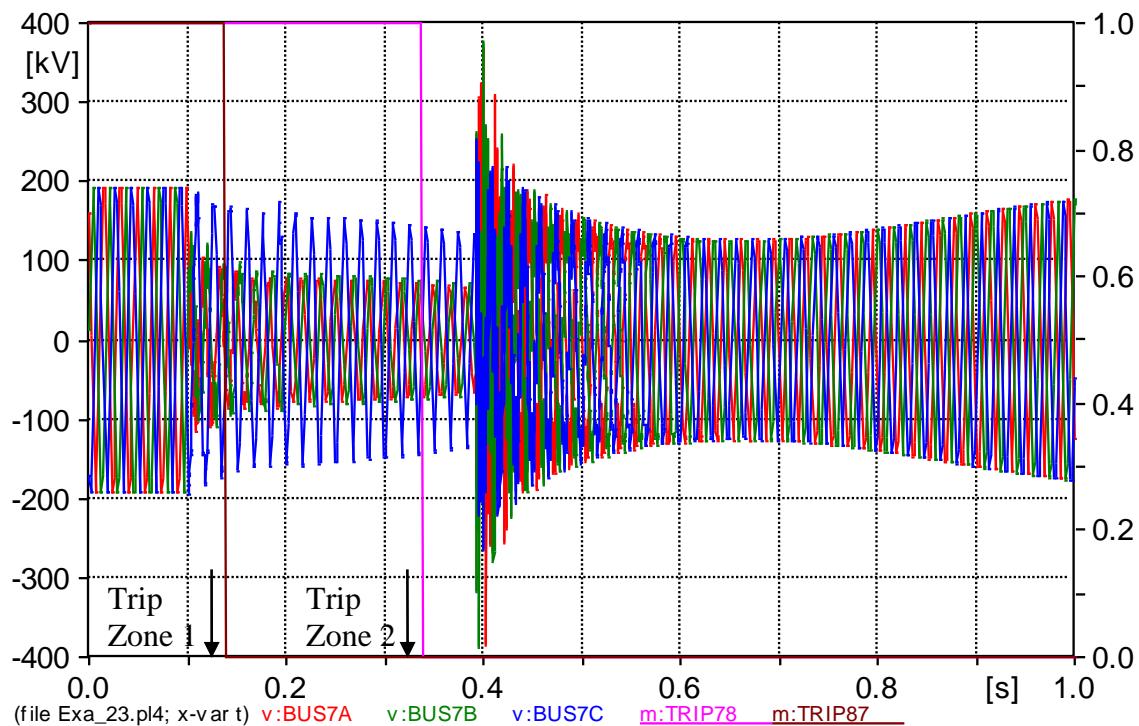


Fig. 6.78 – Simulated voltage at Bus 7 and trip signals (for each side) sent to circuit breaker.

## 6.8 Solar power interface via PWM controlled inverter (Exa\_25.acp)

This example shows how ATPDraw can be used to study the integration of solar energy in the power grid. The example is based on case #43 (by Francisco Peñaloza) at [www.atpdraw.net](http://www.atpdraw.net) for the PV part, but adds a fully switched igbt inverter controlled by MODELS in the dq-plane. The system is shown in Fig. 6.79 with the PV model to the left with sun radiation variations as input and a temperature corrected MODELS controlled current source output. To the right is the 22 kV system where a possible 3-phase fault can be applied at the middle of the feeding line.

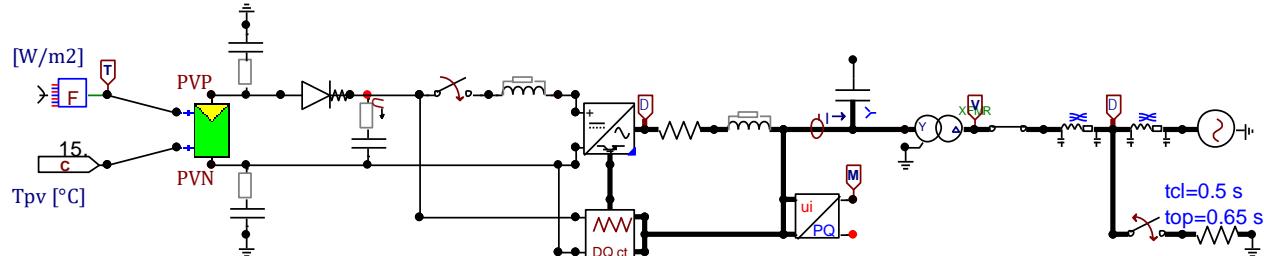


Fig. 6.79 – Photo voltaic source interfaced with igbt PWM-controlled inverter, Exa\_25.acp.

In the middle of Fig. 6.79 is the inverter in a Group shown below with a Models control. Because ATP has no special treatment for the simulation of power electronics switching, it is important to use a small timestep (1  $\mu$ s used here) and add snubbers across the switches. The nonlinear diode DIODEN component is used as it has embedded all needed functionality. The six fire pulses are collected by a 6-phase node and controlled by a PWM scheme.

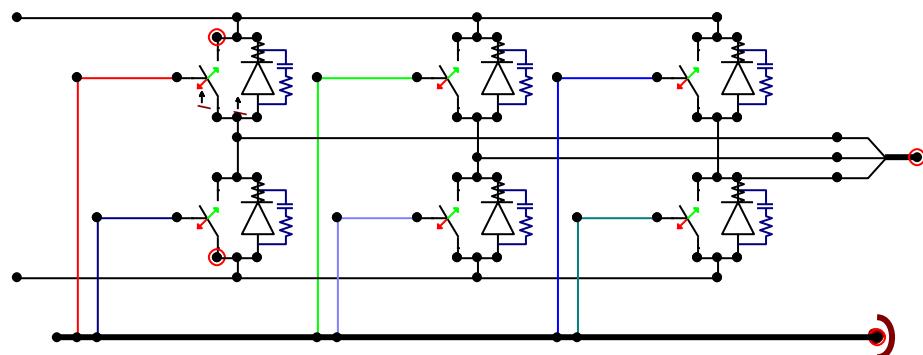


Fig. 6.80 – Switched igbt inverter

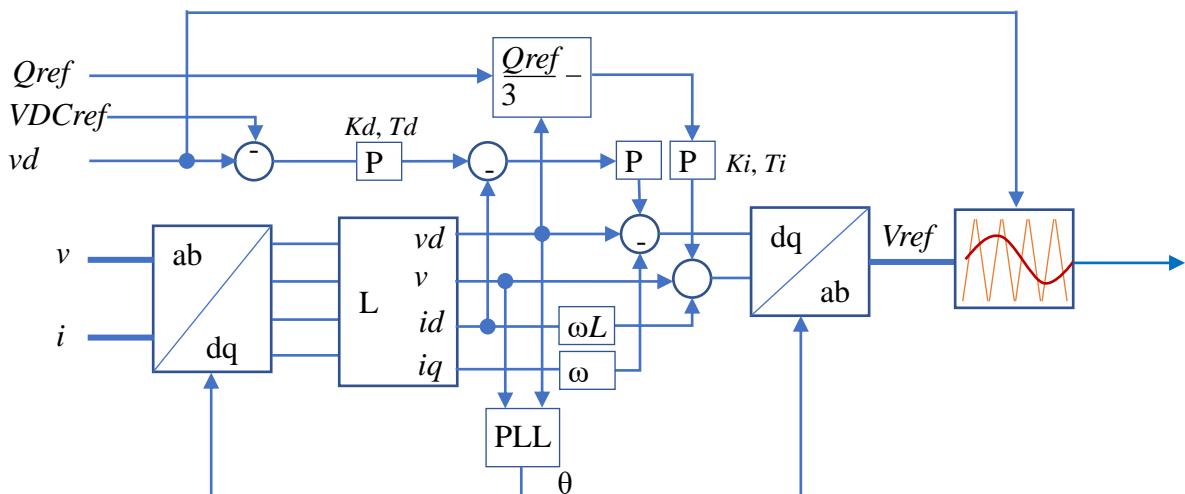


Fig. 6.81 – Control scheme of igbt inverter

The input dialog of the voltage source inverter control is shown in Fig. 6.82. The Data attributes are manipulated to show all data values.

The control strategy is to force reactive power output to zero and keep the DC-bus voltage constant at 1800 V. The PV is first ramped up from zero and the inverter-switch on the DC-side is connected at 0.02 sec. The response of the control to a change in sunlight (Irad) is shown in Fig. 6.83.

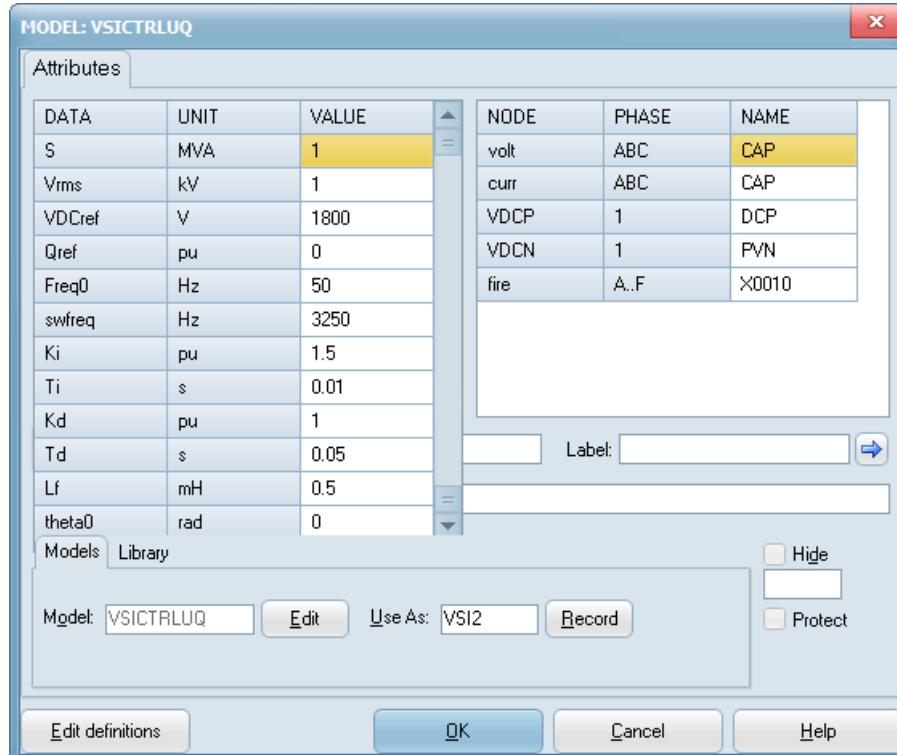


Fig. 6.82 – Input dialog of VSICTRLUQ Model.

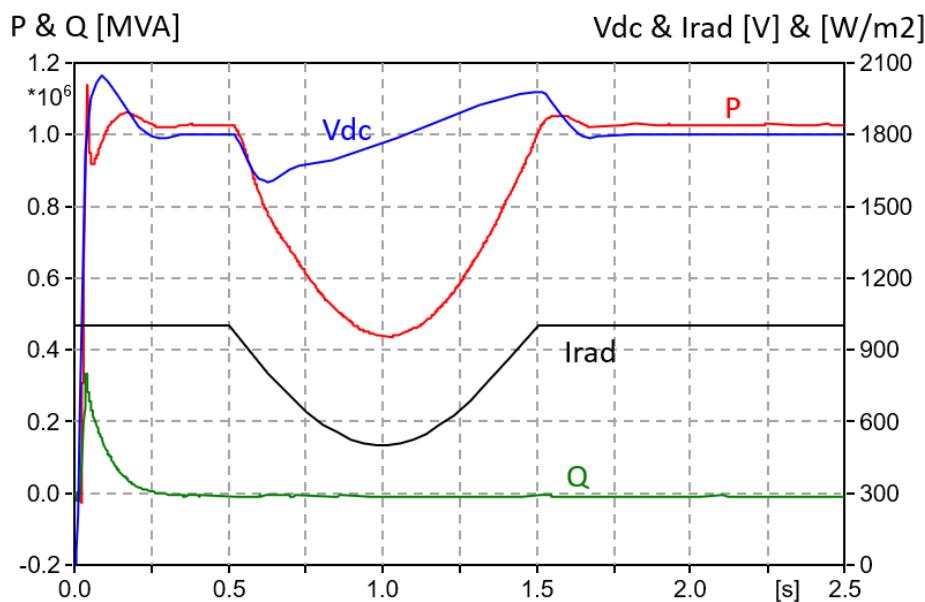


Fig. 6.83 – Calculated power (P & Q) on the AC-side of the inverter and DC-bus voltage (Vdc) as response to irradiated power (Irad).

An ideal 3-phase fault at the middle of the feeding AC line is next applied at 0.5 sec lasting for 150 ms. The response of the control is shown in Fig. 6.84. There is a large overshoot in the DC-bus voltage when clearing the fault at 0.65 sec as power is fed into the DC side for a short period.

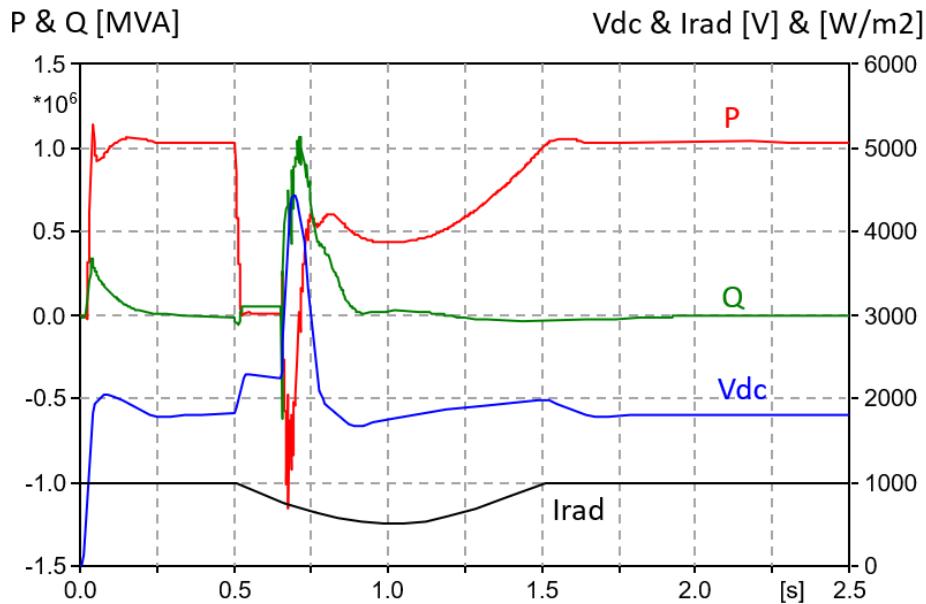


Fig. 6.84 – Calculated power (P & Q) on the AC-side of the inverter and DC-bus voltage (Vdc) as response to irradiated power (Irad) and a 3-phase AC fault from 0.5 to 0.65 sec.

The MODELS script for reactive power and DC-bus control is shown below:

```

MODEL VSICTRLUQ
COMMENT =====
  05-Oct-2018, Prof. H. K. Hoidalen
=====
INPUT  volt[1..3] -- phase voltages
       curr[1..3] -- line current out of inverter
       VDCP, VDCN -- Positive and negative DC-bus voltage
OUTPUT fire[1..6]
CONST
  twopi {VAL:6.283185307}
DATA
  S      {dflt:1}      --MVA
  Vrms  {dflt:1}      --kV L-L rms
  VDCref {dflt:1}      --V
  Qref   {dflt:0}      --pu
  Freq0 {dflt:50}      --Hz
  swfreq {dflt:3250}    --Hz
  Ki     {dflt:1}      --pu
  Ti     {dflt:0.003}   --sec
  Kd     {dflt:1}      --pu
  Td     {dflt:0.1}     --sec
  Lf     {dflt:0.5}     --mH filter inductance
  theta0 {DFLT:0.0}     --initial phase betw. d-axis and alpha-axis
VAR
  UA,UB,UD,UQ,IA,IB,ID,IQ, UDF, UQF, IDF, IQF, IDerr, IQerr, VDCor, VQcor, tx, triang
  VDref, VQref, Vref[1..3], freq, omega, theta, dth, tau, Vpu, Ipu, Fire[1..6], i
  P, Q, U0, IDref, IQref, Vlim, Lfpu, IQmax, Vdc, VDCerr
HISTORY
  INTEGRAL(omega) {DFLT:theta0*PI/180}
  freq {DFLT:Freq0}
  UDF {dflt:1}
  UQF {dflt:0}
  IDF {dflt:0}

```

```

IQF {dflt:0}
VDcor {dflt:0}
VQcor {dflt:0}
IDref {dflt:0}

INIT
omega:=2*PI*Freq0
theta := theta0*PI/180.0
tau:=recip(2*PI*swFreq/5)
Vpu:=Vrms*1000*sqrt(2/3) --peak ref.
Ipu:=S*1000/Vrms*sqrt(2/3) --peak ref
Lfpu:=Lf/1000*Ipu/Vpu --scale to H and pu
Vlim:=5.0

ENDINIT
EXEC
Vdc:=VDCP-VDCN
--Park transforms
UD:=volt[1]*cos(theta)+volt[2]*cos(theta-2*pi/3)+volt[3]*cos(theta+2*pi/3) --Park
UQ:=-volt[1]*sin(theta)-volt[2]*sin(theta-2*pi/3)-volt[3]*sin(theta+2*pi/3) --Park
ID:=curr[1]*cos(theta)+curr[2]*cos(theta-2*pi/3)+curr[3]*cos(theta+2*pi/3) --Park
IQ:=-curr[1]*sin(theta)-curr[2]*sin(theta-2*pi/3)-curr[3]*sin(theta+2*pi/3) --Park
claplace(UDF/UD):=(sqrt(2/3)/Vpu|s0)/(1|s0+sqrt(2)*tau|s1+tau**2|s2) --LP filter
claplace(UQF/UQ):=(sqrt(2/3)/Vpu|s0)/(1|s0+sqrt(2)*tau|s1+tau**2|s2) --LP filter
claplace(IDF/ID):=(sqrt(2/3)/Ipu|s0)/(1|s0+sqrt(2)*tau|s1+tau**2|s2) --LP filter
claplace(IQF/IQ):=(sqrt(2/3)/Ipu|s0)/(1|s0+sqrt(2)*tau|s1+tau**2|s2) --LP filter

P:=(UDF*IDF+UQF*IQF)*2/3
Q:=-UDF*IQF-UQF*IDF)*2/3
VDCerr:=(Vdc-VDCref)/(2*Vpu)
claplace(IDref/VDCerr){dmin:-1.3 dmax:1.3} :=(Kd/Td|s0+Kd|s1)/(1.0|s1) --PI regulator
--IDref:=Pref/UDF*3/2 {min:-1.3 max:1.3}
IQmax:=sqrt(max(0,(3/2*S)**2-IDref**2))
IQref:=-Qref/UDF*3/2 {min:-IQmax max:IQmax}
IDerr:=(IDref-IDF)
IQerr:=(IQref-IQF)

claplace(VDcor/IDerr):=(Ki/Ti|s0+Ki|s1)/(1.0|s1) --PI regulator
claplace(VQcor/IQerr):=(Ki/Ti|s0+Ki|s1)/(1.0|s1) --PI regulator
VDref:=UDF-IQF*omega*Lfp+VDcor {min:-Vlim max:Vlim}
VQref:=UQF+IDF*omega*Lfp+VQcor {min:-Vlim max:Vlim}

for i:=1 to 3 do
  Vref[i]:=sqrt(2/3)*(VDref*cos(theta-(i-1)*2*pi/3)-VQref*sin(theta-(i-1)*2*pi/3))
endfor

tx:=t mod recip(swfreq)
triang:=Vdc/(2*Vpu)*(tx*4*swfreq-1 - 2*(tx*4*swfreq-2)*(2*tx>recip(swfreq))) --triangular PWM
for i:=1 to 3 do
  fire[i]:=Vref[i]>triang
  fire[i+3]:=not fire[i]
endfor

--PLL-----
dth:=atan2(UQF,UDF)
claplace(Freq/dth):=(0.00277|s0+2.5|s1)/(1|s1)
omega := 2*pi*Freq
theta := INTEGRAL(omega)
ENDEXEC
ENDMODEL

```

## 6.9 Post-processing

ATPDraw offers a few options for post-processing as summarized in Chapt. 3.9. Here the COMTRADE and embedded plotting features are explained.

### 6.9.1 COMTRADE generation

The example Exa\_26.acp illustrates how to use the COMTRADE objects to generate comtrade files. The case is a single-pole reclosing after a ground fault phase A. Only the COMTRADE1 object is shown in Fig. 6.85 because the other options are similar. The timestep of the simulation is 125  $\mu$ s and the power frequency is 60 Hz. The sampling frequency chosen in the COMTRADE1 object is on the contrary 1920 Hz, meant to illustrate how this can be selected independent from the timestep. A special trick using the COMTRADE objects is that a packing object is needed to stack first the analog then the digital channels. The user must specify the MODELS code for each case as shown below. The COMTRADE1 object requires declaration of a single output and 13 phases is chosen in this case with first 10 analog, then 3 digital channels. Inside the packing object, zero sequence voltage and current are calculated. Click on the packing object's input nodes to select the type of input (voltage, current, switch status, MODELS).

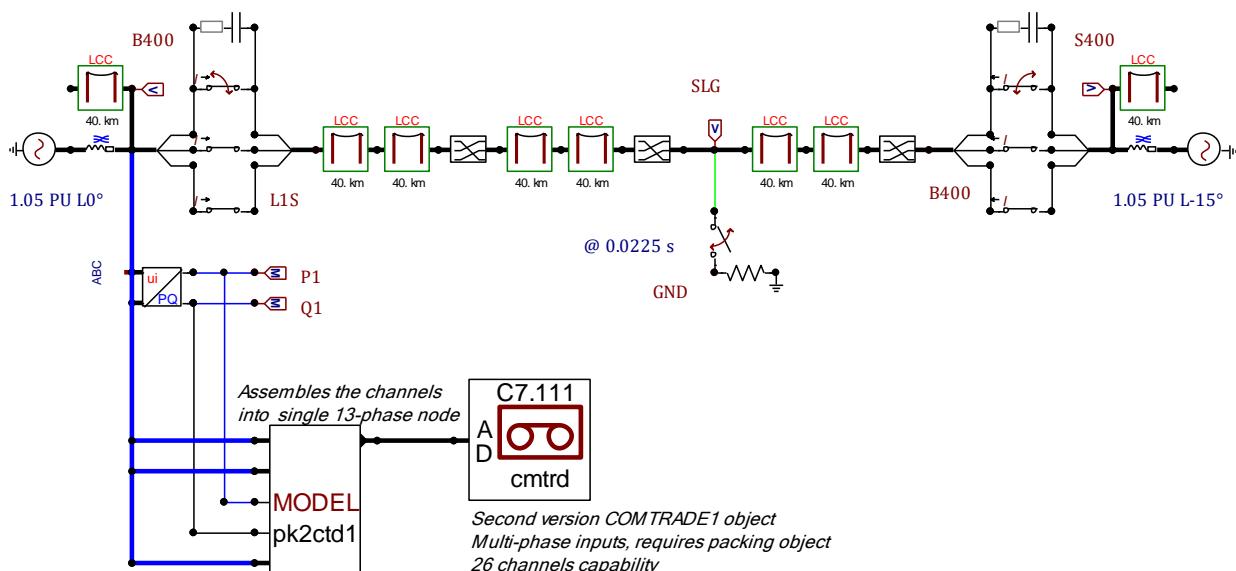


Fig. 6.85 – Setting up creation of COMTRADE files with packing object.

In Fig. 6.85 there are three Models component in series, so the sequence of these objects must be correct. Control the sequence manually from the *Sidebar/Project* (click Update) or simply select *Edit/Arrange/Sort all Models*.

Packing object MODELS code.

```

MODEL PACK2CTD1      -- * * * User defined COMTRADE Signal 'Packer' * * *
INPUT V1ABC[1..3]    -- 1st 3-ph Analog signals: L-N Voltages
      I1ABC[1..3]    -- 2nd 3-ph Analog signals: phase currents
      P1, Q1        -- 1-ph Analog signal: P, Q powers
      CBST [1..3]    -- digital signals: line's CB pole status
OUTPUT CHA[1..13]   -- Packed
VAR CHA[1..13]    -- Signals
EXEC
  CHA[1] := V1ABC[1] -- First pack signals into analog channels
  CHA[2] := V1ABC[2]
  CHA[3] := V1ABC[3]
  CHA[4] := V1ABC[1] + V1ABC[2] + V1ABC[3] - Zero seq voltage 3*V0
  CHA[5] := I1ABC[1]
  CHA[6] := I1ABC[2]
  CHA[7] := I1ABC[3]
  CHA[8] := -(I1ABC[1] + I1ABC[2] + I1ABC[3]) -- 3I0 return

```

```

CHA[9] := P1
CHA[10] := Q1
CHA[11] := CBST[1] -- Then pack signals into digital channels
CHA[12] := CBST[2]
CHA[13] := CBST[3]
ENDEXEC
ENDMODEL

```

The COMTRADE object's input dialog is shown in Fig. 6.86. The user can select an arbitrary sampling frequency and both up and down sampling is supported. The COMTRADE objects consist of open MODELS code that can be modified by the user, but the WRITE section must not be changed. The sum of the number of analog (NumA) and digital (NumD) channels must match the number of outputs of the packing object (13 in this case). Trigger is specified in the IEEE C37.111 standard and Tinit is additionally used to start the sampling.

On the Comtrade page (Fig. 6.87) the channel information should be provided according to the IEEE C37.111-1999 standard. The Comtrade file format can be ascii or binary, but in addition an option to store the data in a .mat MatLab v4 file is available. ATPDraw reads in the written data from the LIS-file and perform the necessary scaling to create appropriate .cfg and .dat files stored in the Result Directory with the name given as *Filename*. The time multiplier is added as an option since vendors tend to interpret the time scale a bit differently. The COMTRADE objects have a View option that plots the raw data read from the LIS-file. In a multiple run situation, only the last run is stored this was unless Merge results in multiple runs is checked. The configuration file .cfg is on standard format and can be loaded into external software for further processing and documentation like shown in Fig. 6.88.

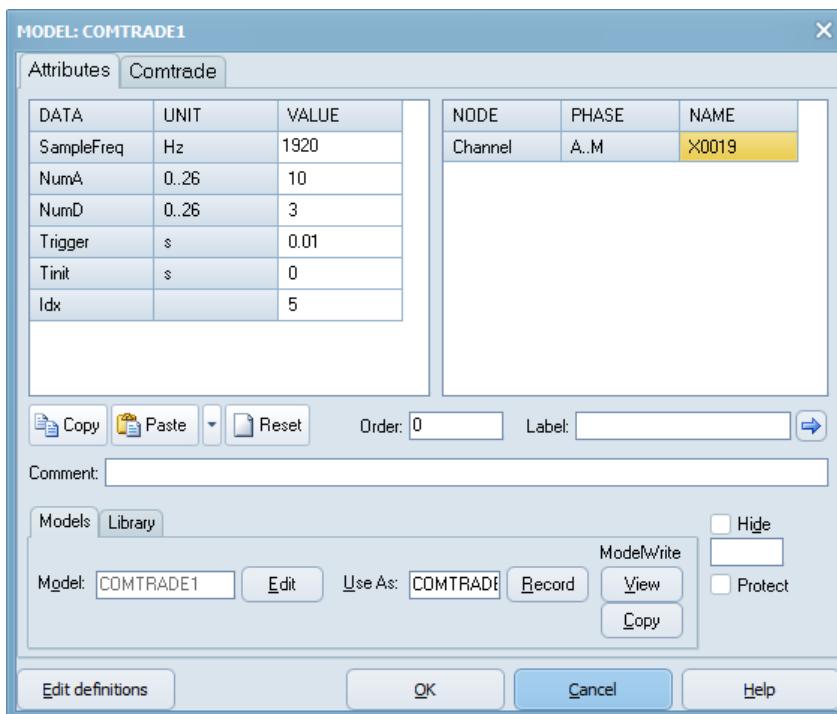


Fig. 6.86 – COMTRADE object's dialog. Standard MODELS page.

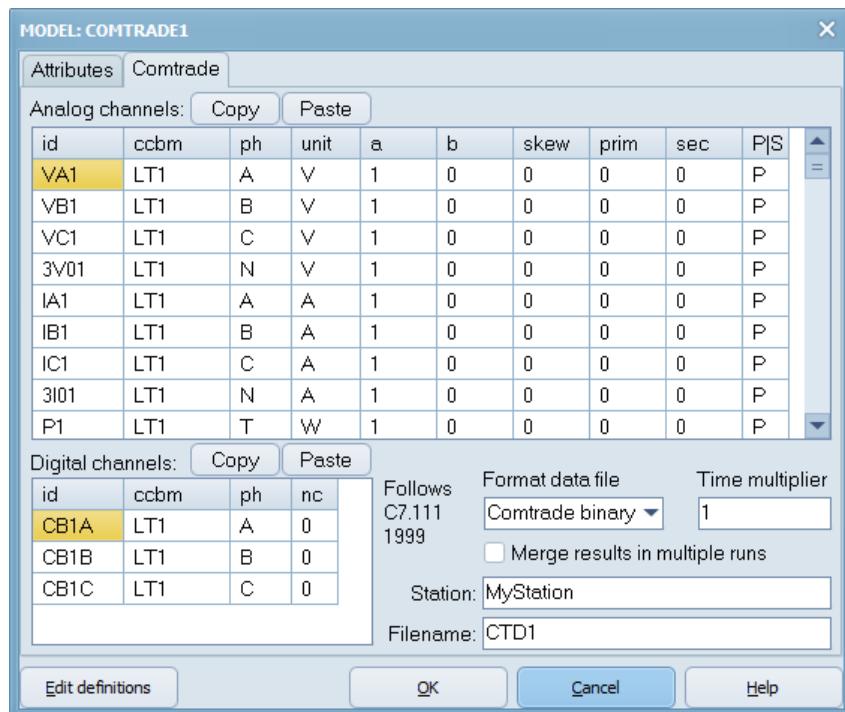


Fig. 6.87 – COMTRADE object's dialog, Comtrade page.

CDT1.cfg file:

```

MyStation,ATPDraw,1999
13,10A,3D
1,VA1,A,LT1,V,100.,0.0,0.0,-3664.9697,3664.9697,0.0,0.0,P
2,VB1,B,LT1,V,100.,0.0,0.0,-3509.9676,3509.9676,0.0,0.0,P
3,VC1,C,LT1,V,100.,0.0,0.0,-3475.72287,3475.72287,0.0,0.0,P
4,3V01,N,LT1,V,10.,0.0,0.0,-10277.314,10277.314,0.0,0.0,P
5,IA1,A,LT1,A,1.,0.0,0.0,-5078.8188,5078.8188,0.0,0.0,P
6,IB1,B,LT1,A,0.1,0.0,0.0,-10775.6608,10775.6608,0.0,0.0,P
7,IC1,C,LT1,A,0.1,0.0,0.0,-14554.9193,14554.9193,0.0,0.0,P
8,3I01,N,LT1,A,1.,0.0,0.0,-4971.95969,4971.95969,0.0,0.0,P
9,P1,T,LT1,W,1.E5,0.0,0.0,-6409.297,6409.297,0.0,0.0,P
10,Q1,T,LT1,VAr,1.E5,0.0,0.0,-6332.661,6332.661,0.0,0.0,P
1,CB1A,A,LT1,0
2,CB1B,B,LT1,0
3,CB1C,C,LT1,0
60.
1
1920.,1152
17/12/2020, 20:57:15.417000
17/12/2020, 20:57:15.427000
BINARY
1.

```

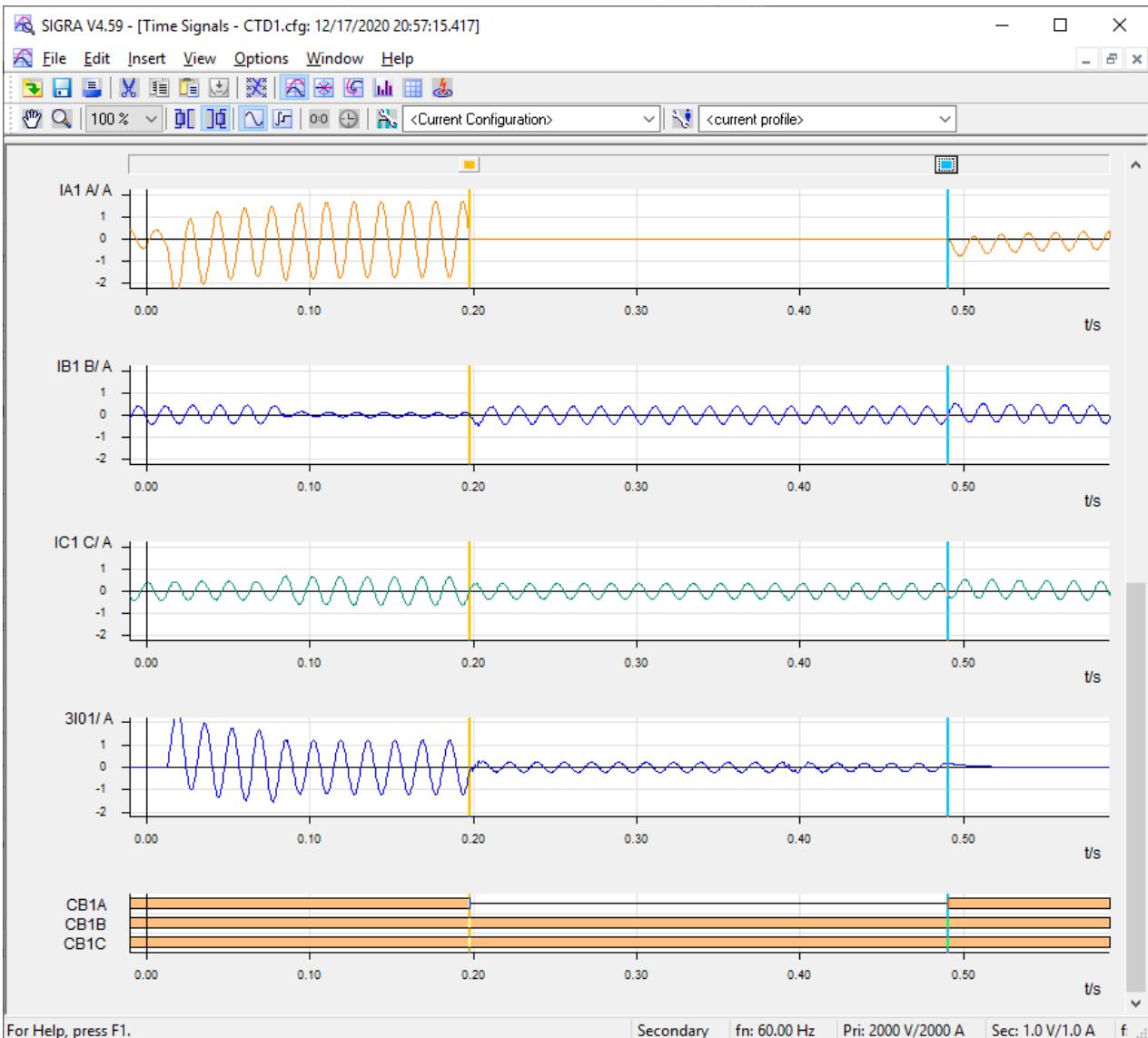


Fig. 6.88 – Loading of the CTD1.cfg (and corresponding CDT1.dat) in SIGRA.

### 6.9.2 Embedded plotting

Embedded plotting was added to ATPDraw v7.1 and improved in steps into v7.2. In contrast to the other post-processing technologies, the Plot object reads directly from the PL4-file. This is based on sub-scripting to specific curve names and runs. The functionality of the plotting object is explained in Chapt. 3.9.4. In Exa\_27.acp it is used to directly compare the inrush current as function of the switching instant, dependent on the switching strategy.

The main work is to select the curves to plot as shown in Fig. 6.90. Clicking in the column *Series name*, a list of all plotting variables is available in a dropdown combo box (obtained from Output Manager (F9)). The naming convention follows that of PlotXY. The selected name will then be stored in the object as a subscription to curves in the PL4-file. If node names change in the process, there will be no error, just missing curves. Consequently, the Plot object must be updated as well. It is therefore recommended to only plot curves whose nodes are user named and fixed. Development in the Plot object will likely be related to the selection of curves to plot.

The result from the investigation in Fig. 6.89 shows that it is possible to reduce the maximum inrush current from 1200 A to around 500 A with the switching strategy to the left.

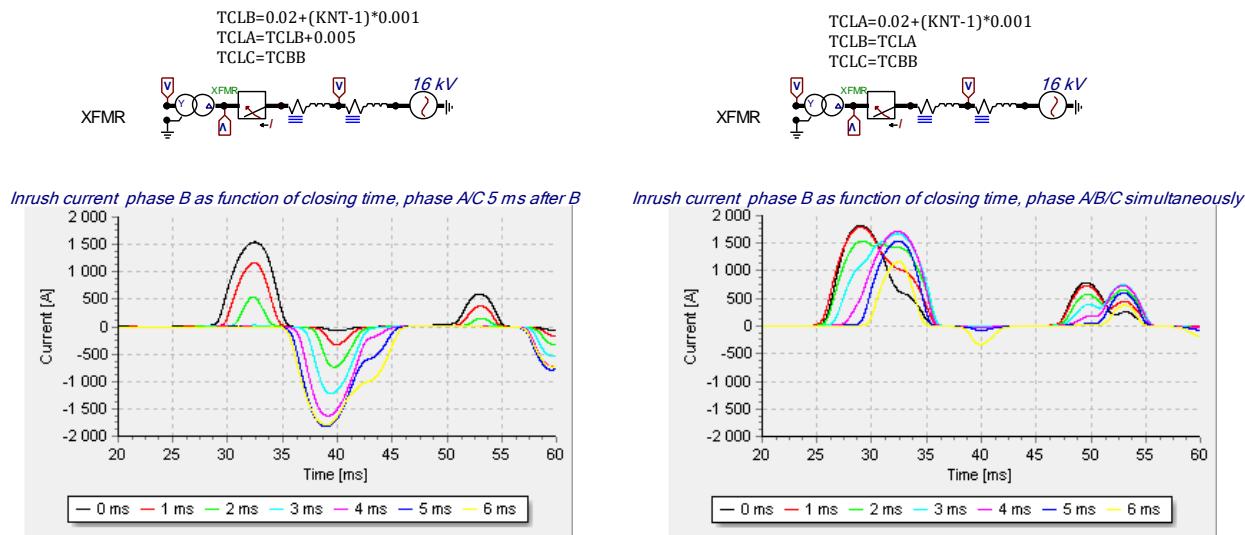


Fig. 6.89 – Embedded plotting objects for testing of inrush current.

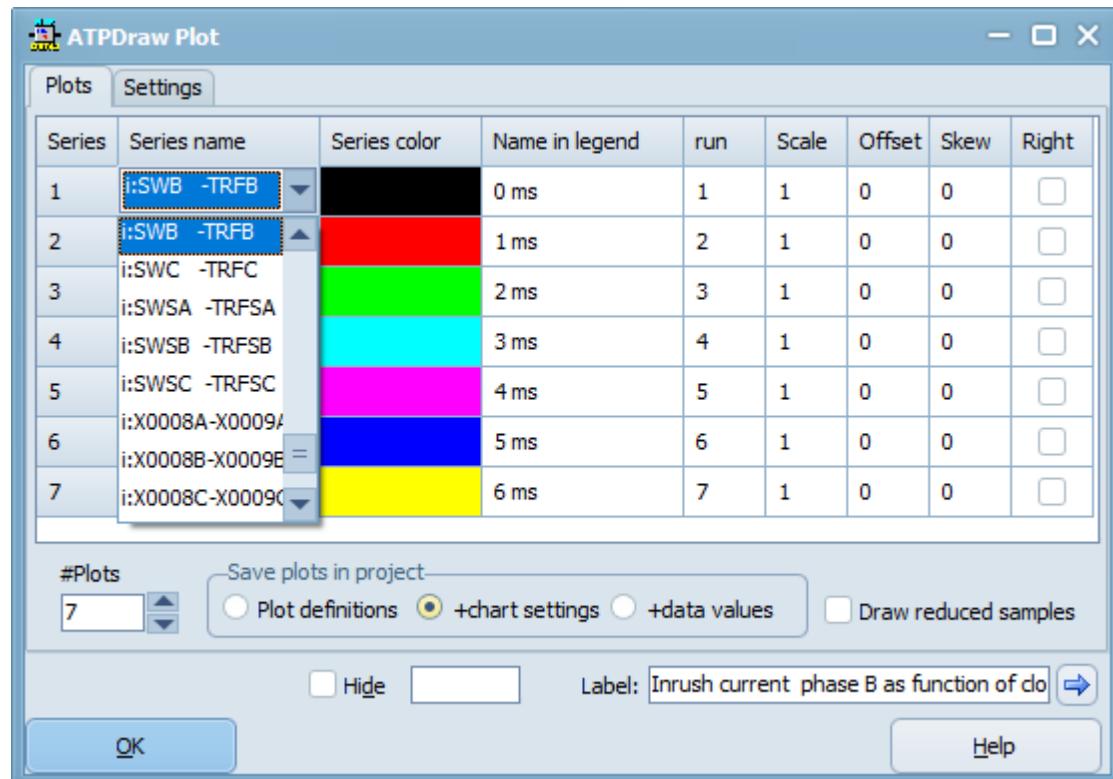
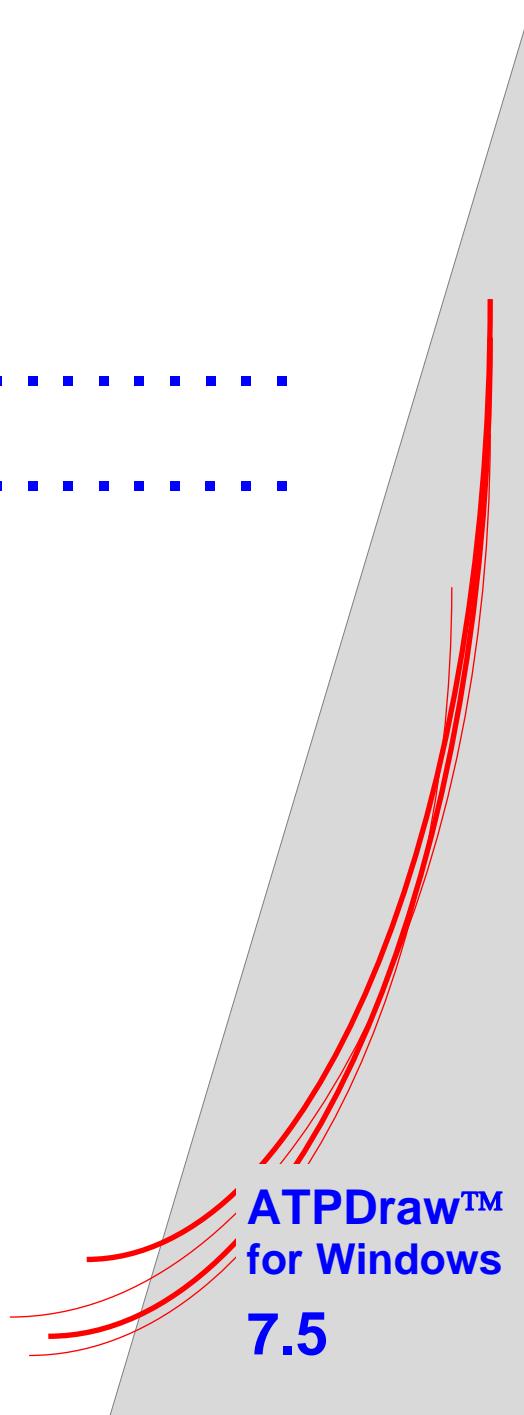


Fig. 6.90 – Selecting curves to plot.



## **7. Appendix .....**

A graphic element consisting of a light gray parallelogram containing three red wavy lines that converge towards the top right corner. To the right of this shape, the text 'ATPDraw™ for Windows' is written in blue, and below it, the number '7.5' is also in blue.

**ATPDraw™  
for Windows**

**7.5**



## 7.1 PFC simulations in ATPDraw

The *Verify* feature of ATPDraw enables the user to compare the line/cable model with an exact PI-equivalent as a function of frequency, or verify the power frequency benchmark data for zero/positive short circuit impedances, reactive open circuit line charging, and mutual zero sequence coupling. The *Verify* module supports the POWER FREQUENCY CALCULATION (PFC) of zero and positive short circuit impedances and open circuit reactive line charging, along with mutual zero sequence impedance for multi circuit lines.

The supporting programs LINE CONSTANTS and CABLE CONSTANTS calculate the series impedance and the shunt admittance from geometrical data and material properties. These electrical parameters are part of the printout file (.lis). The power frequency calculations give in principle the short circuit impedances and the open circuit reactive power. The line/cable may be a single circuit component with an arbitrary number of phases or a multi-circuit component where all circuits normally are three-phase. The following parameters are calculated for a single circuit in a line/cable with  $n$  conductors:

### a) Short circuit impedances

All terminals at one end of the line/cable are connected to ground. A positive sequence symmetrical voltage is applied to the terminals at the other end and the positive sequence impedance is calculated:  $Z_+ = E_+ / I_+$

The voltage applied to the terminal  $I$  is:

$$E_i = E_+ \cdot \exp(-j \cdot 2\pi \cdot (i-1)/n), \text{ where } n \text{ is the number of phases in the circuit.}$$

The positive sequence current is obtained from the terminal currents by the formula:

$$I_+ = \frac{1}{n} \cdot [I_1 + I_2 \cdot \exp(j2\pi/n) + \dots + I_i \cdot \exp(j2\pi(i-1)/n) + \dots + I_n \cdot \exp(-j2\pi/n)]$$

The zero sequence impedance is calculated in a similar way:

$$Z_0 = E_0 / I_0$$

The voltage  $E_0$  here is applied to all terminals and  $I_0$  is the average current supplied by the source.

### b) Open-circuit reactive power

All terminals at one end of the component are open (except the conductors which are specified to be grounded). A positive sequence symmetrical voltage is applied to the terminals at the other end and the positive sequence current component is calculated by the same formula as for the positive sequence impedance. The positive sequence open-circuit reactive power is then calculated by the formula:

$$Q_+ = \text{Im}(n \cdot E_+ \cdot I_+^*), \text{ where } E_+ \text{ is the line to line voltage.}$$

Using the voltage between two adjacent phases for an  $n$ -phase circuit gives  $E_+ = V / [2 \cdot \sin(\pi/n)]$ . The calculation  $I_+$  is based on an ATP calculation with  $E_+ = 1.0$ . Using this value for  $I_+$  implies that

$$Q_+ = \frac{-V^2 \cdot n}{4 \cdot \sin^2(\pi/n)} \text{ Im}(I_+)$$

ATP also automatically calculates the reactive power supplied by the source ( $Q_1..Q_n$ ). The open-circuit reactive power can thus also be calculated by taking the average of these quantities for all phases and multiply by a factor 2 (since a peak value 1.0 is used in the calculation and the line-to-line voltage is specified as rms):

$$Q_+ = \frac{-V^2 \cdot 2}{n} (Q_1 + Q_2 + \dots + Q_n)$$

The zero sequence open-circuit reactive power is calculated as well. The same voltage is then applied to all terminals at one end of the line. The zero sequence current is the average value of the current injected into the terminals. This current  $I_0$  is calculated by ATP with  $E_0 = 1.0$ . Using this value for  $I_0$  implies that

$$Q_0 = \frac{-V^2 \cdot n}{4 \cdot \sin^2(\pi/n)} \operatorname{Im}(I_0)$$

In this case ATP automatically calculates the reactive power  $Q$ , injected into the circuit from the source. Similarly to the positive sequence values, the zero sequence open-circuit reactive power is also equal to

$$Q_0 = \frac{-V^2 \cdot 2}{n} (Q)$$

For a line/cable with several circuits, each circuit is tested separately. For short-circuit calculation the other circuit(s) is/are also grounded at one end, while for open-circuit calculations all terminals are open. The mutual coupling between the circuits is calculated as well and called *zero sequence transfer impedance*. This is done by connecting all phases of each individual circuit to a common node. A current  $3 \cdot I_0$  is then applied to one of these common nodes circuit and the voltage on the other node is measured. All terminals at the other end of the component is grounded. The procedure is repeated for all circuits except the last one. Below is listed the xVerifyF.dat file for a 6-phase line.

```

BEGIN NEW DATA CASE
1.667E-9      -1.0
    1           1           1
$PREFIX, D:\ATPDraw3\lcc\
$INCLUDE, LCC_6.lib, INZO1_, INZO1_, INZO1D, INZO1E, INZO1F $$
, OUTO1A, OUTO1B, OUTO1C, OUTO1D, OUTO1E, OUTO1F
$INCLUDE, LCC_6.lib, INZO2A, INZO2B, INZO2C, INZO2_, INZO2_, INZO2_ $$
, OUTO2A, OUTO2B, OUTO2C, OUTO2D, OUTO2E, OUTO2F
$INCLUDE, LCC_6.lib, INZS1_, INZS1_, INZS1D, INZS1E, INZS1F $$
, #####, #####, #####, #####, #####
$INCLUDE, LCC_6.lib, INZS2A, INZS2B, INZS2C, INZS2_, INZS2_, INZS2_ $$
, #####, #####, #####, #####, #####
$INCLUDE, LCC_6.lib, INPO1A, INPO1B, INPO1C, INPO1D, INPO1E, INPO1F $$
, OUPO1A, OUPO1B, OUPO1C, OUPO1D, OUPO1E, OUPO1F
$INCLUDE, LCC_6.lib, INPO2A, INPO2B, INPO2C, INPO2D, INPO2E, INPO2F $$
, OUPO2A, OUPO2B, OUPO2C, OUPO2D, OUPO2E, OUPO2F
$INCLUDE, LCC_6.lib, INPS1A, INPS1B, INPS1C, INPS1D, INPS1E, INPS1F $$
, #####, #####, #####, #####
$INCLUDE, LCC_6.lib, INPS2A, INPS2B, INPS2C, INPS2D, INPS2E, INPS2F $$
, #####, #####, #####, #####
$INCLUDE, LCC_6.lib, INMS11, INMS11, INMS11, INMS12, INMS12, INMS12 $$
, #####, #####, #####, #####
BLANK BRANCH
BLANK SWITCH
14INZO1_+1      1.0      50.      0.0      -1.0
14INZO2_-+1     1.0      50.      0.0      -1.0
14INPO1A+1      1.0      50.      0.0      -1.0
14INPO1B+1      1.0      50.     -120.     -1.0
14INPO1C+1      1.0      50.     -240.     -1.0
14INPO2D+1      1.0      50.      0.0      -1.0
14INPO2E+1      1.0      50.     -120.     -1.0
14INPO2F+1      1.0      50.     -240.     -1.0
14INZS1_-+1     1.0      50.      0.0      -1.0
14INZS2_-+1     1.0      50.      0.0      -1.0
14INPS1A+1      1.0      50.      0.0      -1.0
14INPS1B+1      1.0      50.     -120.     -1.0
14INPS1C+1      1.0      50.     -240.     -1.0
14INPS2D+1      1.0      50.      0.0      -1.0
14INPS2E+1      1.0      50.     -120.     -1.0
14INPS2F+1      1.0      50.     -240.     -1.0

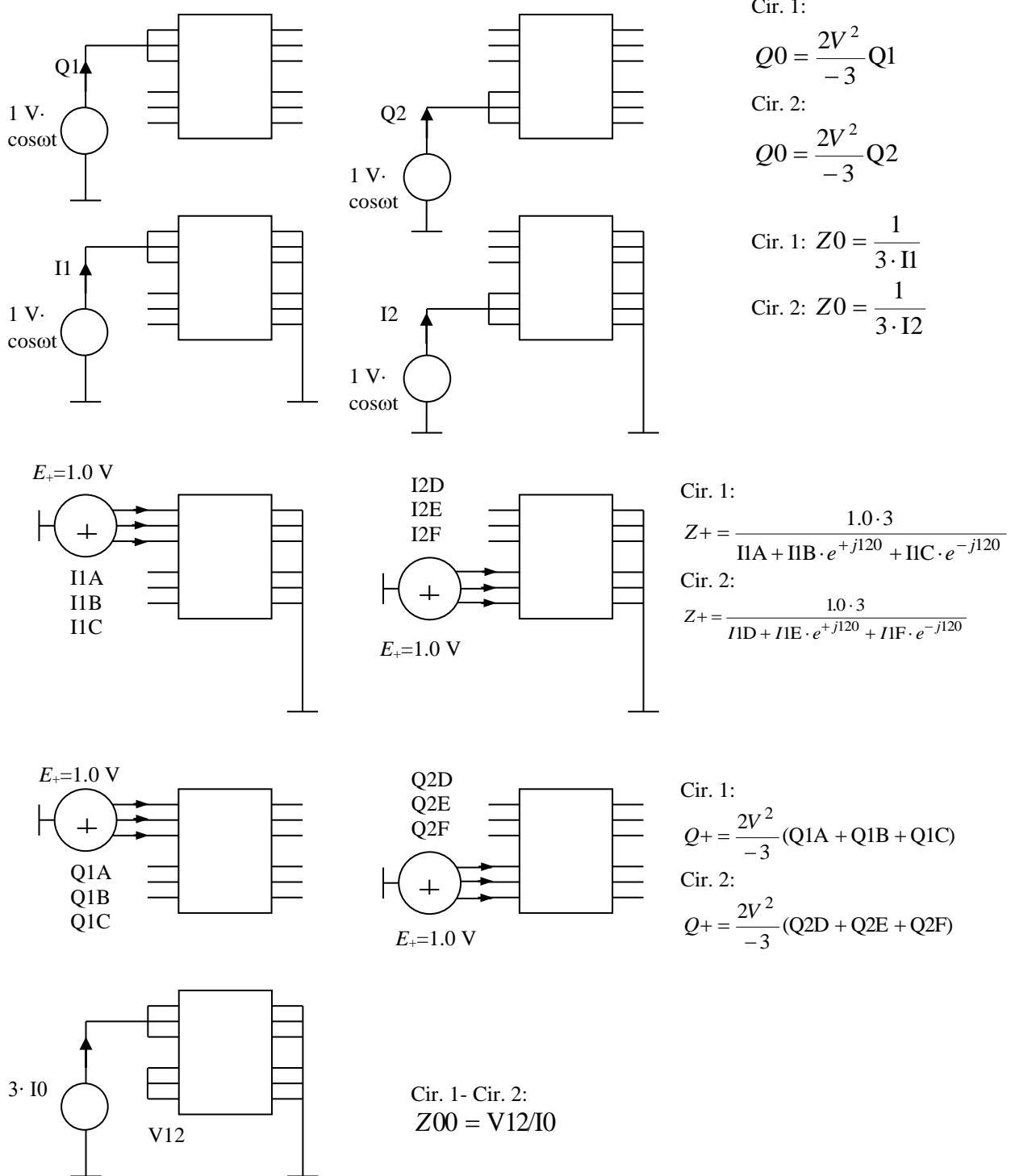
```

```

14INMS11-1      3.      50.      0.0      -1.0
BLANK SOURCE
INMS12
BLANK OUTPUT
BLANK CARD PLOT
BEGIN NEW DATA CASE
BLANK

```

The xVerifyF.dat file describes the following 9 cases:



*Zero sequence short circuit impedance:* (real and imaginary part).  $Z_0 = R_0 + jX_0$ .

Fig. 7.1 – LCC-Verify; Power Frequency Calculations.

Each phase of a circuit is connected to a 1 V amplitude voltage source with zero phase angle. The other end of the line is grounded.  $Z_0$  is calculated as the inverse of the injected current divided by the number of phases in the circuit. All phase conductors of other phases are open.

*Positive sequence short circuit impedance:* (real and imaginary part).  $Z_+ = R_+ + jX_+$ .

The phases of a circuit are connected to a 1 V amplitude voltage source with phase angle  $-360*(i-1)/n$  where  $I$  is the phase number (1,2,3..) and  $n$  is the number of phases of the tested circuit. The other end of the line is grounded.  $Z_+$  is calculated as the inverse of the positive sequence current. All phase conductors of other phases are open.

*Zero sequence line charging:*  $Q_0$

Each phase of a circuit is connected to a 1 V amplitude voltage source with zero phase angle. The other end of the line is open.  $Q_0$  is the injected reactive power multiplied by the square of the user specified base voltage (multiplied with  $2/n$ ). All phase conductors of other phases are open.

*Positive sequence line charging:*  $Q_+$

The phases of a circuit are connected to a 1 V amplitude voltage source with phase angle  $-360*(i-1)/n$  where  $I$  is the phase number and  $n$  is the number of phases of the tested circuit. The other end of the line is open.  $Q_+$  is calculated as the average injected reactive power multiplied by the square of the user specified base voltage (multiplied with  $2/n$ ). All phase conductors of other phases are open.

*Mutual zero sequence impedance:* (real and imaginary part).  $Z_{00} = R_{00} + jX_{00}$ .

Each phase of the  $i^{th}$  circuit is connected to a 1 A amplitude current source with zero phase angle. The receiving end of the circuits  $I$  and  $j$  is grounded. The  $j^{th}$  circuit is short-circuited and open in the sending end.  $Z_{00}$  is calculated as the voltage at the sending end of the  $j^{th}$  circuit. The process is repeated for all circuits. All phase conductors of phases not belonging to the  $i^{th}$  and  $j^{th}$  circuit are open.

## 7.2 Line Check

When performing transient analysis of power systems, high frequency models of overhead transmission lines and underground cables must be developed. In this process, parameters like ground and conductor conductivity, cross-section geometry, and average overhead line height could be uncertain and questionable. Very often the only reliable benchmark data are sequential parameters at power frequency. It is thus of great interest to be able to verify the developed line/cable model at power frequency before simulating and analyzing transients. The present version of ATPDraw has in the LCC-module a built-in option to verify a line segment [1]. This is done by calculating the short circuit input impedances and the open circuit reactive power consumption. In addition, a frequency scan is supported. However, data for each line segment is rarely available, and in addition one would prefer to verify an entire line/cable length including the effect of transpositions. Instead of calculating the short circuit input impedance and the open circuit reactive power consumption it would be better to obtain the serial impedance and the shunt admittance along with the average mutual impedance and admittance between circuits in 6-phase and 9-phase cases. The new module integrated in ATPDraw involves an improved handling of the equivalent mutual coupling between circuits.

### 7.2.1 Single phase systems

Initially, consider a single-phase circuit of length  $l$  with frequency domain distributed series impedances and shunt admittances, as shown in Fig. 7.2. The line is spited in segments of length  $dx$ .

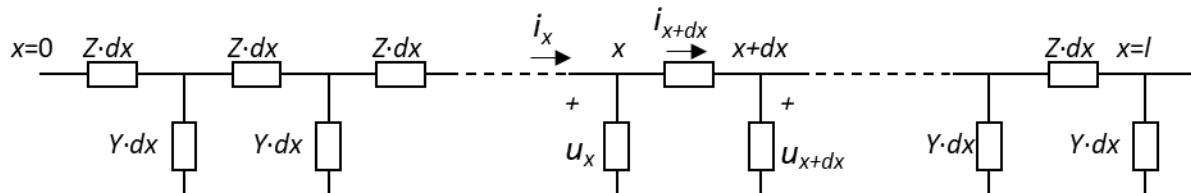


Fig. 7.2 – Single phase representation of transmission line.  $Z = R + j\omega L$  [ $\Omega/m$ ],  $Y = G + j\omega C$  [ $S/m$ ].

The currents and voltages at the sending and receiving ends will not be equal. The idea is further to use the measured quantities at both terminals to obtain the series impedance and shunt capacitance. Current balance at point  $x$  results in  $-\frac{\partial i}{\partial x} = Y \cdot u$ . The voltage drop between  $x$  and  $x+dx$  gives  $-\frac{\partial u}{\partial x} = Z \cdot i$ . These two equations result in the wave equation  $\frac{\partial^2 u}{\partial x^2} = Z \cdot Y \cdot u$  with the solution  $u(x) = A \cdot e^{\gamma \cdot x} + B \cdot e^{-\gamma \cdot x}$ , where the constants  $A$  and  $B$  are determined from the boundary conditions and  $\gamma = \sqrt{Z \cdot Y}$ . The current is  $i(x) = -Z^{-1} \cdot \frac{\partial u}{\partial x} = -Z^{-1} \cdot \gamma \cdot (A \cdot e^{\gamma \cdot x} - B \cdot e^{-\gamma \cdot x})$

#### 1. Short circuit case:

This is the typical configuration for obtaining the series impedance. A sinusoidal voltage or current is applied at the sending end while the receiving end is grounded.

$$u(0) = U_0 \text{ and } u(l) = 0 \text{ gives}$$

$$A + B = U_0 \text{ and } A \cdot e^{\gamma l} + B \cdot e^{-\gamma l} = 0 \text{ which result in}$$

$$u(x) = U_0 \cdot \frac{\sinh \gamma \cdot (l - x)}{\sinh \gamma \cdot l} \text{ and } i(x) = U_0 \cdot Z^{-1} \cdot \gamma \cdot \frac{\cosh \gamma \cdot (l - x)}{\sinh \gamma \cdot l} \quad (1)$$

The currents at the terminals are

$$i(0) = U_0 \cdot Z^{-1} \cdot \gamma \cdot \frac{\cosh \gamma \cdot l}{\sinh \gamma \cdot l} \approx U_0 \cdot (Z \cdot l)^{-1} \cdot \left( 1 + \frac{1}{3} \cdot (\gamma l)^2 - \frac{1}{45} \cdot (\gamma l)^4 \dots \right) \text{ and} \quad (2)$$

$$i(l) = U_0 \cdot Z^{-1} \cdot \gamma \cdot \frac{1}{\sinh \gamma \cdot l} \approx U_0 \cdot (Z \cdot l)^{-1} \cdot \left( 1 - \frac{1}{6} \cdot (\gamma l)^2 + \frac{7}{360} \cdot (\gamma l)^4 \dots \right) \quad (3)$$

where the approximation comes from a series expansion of the hyperbolic functions.

The second quadratic term is eliminated in the following combination:

$$\tilde{i} = \frac{i(0) + 2 \cdot i(l)}{3} = U_0 \cdot (Z \cdot l)^{-1} \cdot \left( 1 + \frac{1}{180} \cdot (\gamma l)^4 \dots \right) \quad (4)$$

The total series impedance can thus be approximated by the following combination of the measured inputs and outputs:

$$Z_s = \left. \frac{3 \cdot u(0)}{i(0) + 2 \cdot i(l)} \right|_{sc} = Z \cdot l \cdot \left( 1 - \frac{1}{180} \cdot (\gamma l)^4 \dots \right) \approx Z \cdot l \quad [\Omega] \quad (5)$$

The same result is obtained if a current is applied at the sending end instead of a voltage.

## 2. Open circuit case:

This is the typical configuration for obtaining the shunt admittance. A sinusoidal voltage or current is applied at the sending end while the receiving end is left open.

$u(0) = U_0$  and  $i(l) = 0$  gives

$A + B = U_0$  and  $A \cdot e^{\gamma l} - B \cdot e^{-\gamma l} = 0$  which result in

$$u(x) = U_0 \cdot \frac{\cosh \gamma \cdot (l - x)}{\cosh \gamma \cdot l} \text{ and } i(x) = U_0 \cdot Z^{-1} \cdot \gamma \cdot \frac{\sinh \gamma \cdot (l - x)}{\cosh \gamma \cdot l} \quad (6)$$

The unknown terminal quantities are:

$$i(0) = U_0 \cdot Z^{-1} \cdot \gamma \cdot \frac{\sinh \gamma \cdot l}{\cosh \gamma \cdot l} \approx U_0 \cdot Y \cdot l \cdot \left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{2}{15} \cdot (\gamma l)^4 \dots \right) \text{ and} \quad (7)$$

$$u(l) = U_0 \cdot \frac{1}{\cosh \gamma \cdot l} \approx U_0 \cdot \left( 1 - \frac{1}{2} \cdot (\gamma l)^2 + \frac{5}{24} \cdot (\gamma l)^4 \dots \right) \quad (8)$$

where the approximation again comes from a series expansion of the hyperbolic functions.

Similar to the short circuit case an equivalent voltage is defined as:

$$\tilde{u} = \frac{u(0) + 2 \cdot u(l)}{3} = U_0 \cdot \left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{5}{36} \cdot (\gamma l)^4 \dots \right) \quad (9)$$

The total shunt impedance can be approximated by the following combination of the measured inputs and outputs:

$$Y_s = \left. \frac{3 \cdot i(0)}{u(0) + 2 \cdot u(l)} \right|_{oc} = \frac{Y \cdot l \cdot \left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{2}{15} \cdot (\gamma l)^4 \dots \right)}{\left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{5}{36} \cdot (\gamma l)^4 \dots \right)} = Y \cdot l \cdot \left( 1 - \frac{1}{180} \cdot (\gamma l)^4 \dots \right) \approx Y \cdot l \quad [\text{S}] \quad (10)$$

The same result is obtained if a current is applied at the sending end instead of a voltage.

## 3. Comparison with input impedance/admittance

The short circuit input impedance and the open circuit input admittance (scaled to get reactive power in ATPDraw) is for comparison

$$Z_{in} = \left. \frac{u(0)}{i(0)} \right|_{sc} = Z \cdot l \cdot \left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{2}{15} \cdot (\gamma l)^4 \dots \right) \text{ and} \quad (11)$$

$$Y_{in} = \left. \frac{i(0)}{u(0)} \right|_{oc} = Y \cdot l \cdot \left( 1 - \frac{1}{3} \cdot (\gamma l)^2 + \frac{2}{15} \cdot (\gamma l)^4 \dots \right) \quad (12)$$

In these expressions there is a quadratic term present, but for short transmission lines the two approaches will give similar results.

#### 4. PI-circuits implications

So far only a distributed parameter model has been investigated. However, concentrated parameter models are often used. Besides, the distributed parameter models in ATP are replaced by PI-equivalents during steady state calculation. This sub-section briefly outlines the implications of this.

Fig. 7.3 shows a PI-equivalent under short- and open circuit testing.

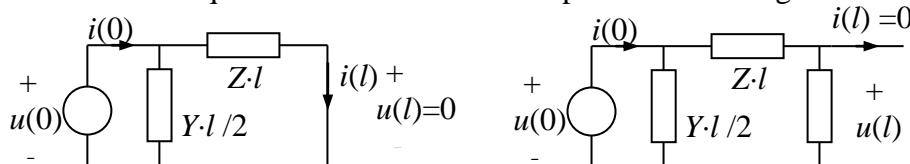


Fig. 7.3 – Testing a PI-circuit. Left: short circuit; serial impedance. Right: open circuit shunt admittance.

The procedure for calculation of the series impedance and shunt admittance in (5) and (10) will in this case result in

$$\begin{aligned} Z_s^{PI} &= \left. \frac{3 \cdot u(0)}{i(0) + 2 \cdot i(l)} \right|_{sc} = \frac{Z \cdot l}{1 + (\gamma l)^2 / 6} \approx Z \cdot l \cdot \left( 1 - \frac{(\gamma l)^2}{6} \right) \text{ and} \\ Y_s^{PI} &= \left. \frac{3 \cdot i(0)}{u(0) + 2 \cdot u(l)} \right|_{oc} = Y \cdot l \cdot \frac{1 + (\gamma l)^2 / 4}{1 + (\gamma l)^2 / 6} \approx Y \cdot l \cdot \left( 1 + \frac{(\gamma l)^2}{12} \right) \end{aligned} \quad (13)$$

Due to the present quadratic term, the result in (13) will be less accurate than for distributed parameters models. Care must be taken to prevent wrong results for long transmission lines. For example, by splitting the line up in smaller segments. In constant parameter distributed parameter line models the series resistance  $I$  is concentrated at each end ( $R/4$ ) and at the middle of the line ( $R/2$ ). This will result in some different formulations than in (13), with accuracy dependent on  $R$ . A solution to this problem is to request ‘EXACT PHASOR EQUIVALENT’ [2, 3] which prevents ATP from using lumped resistance. In such case the “exact pi” equivalent is used (as is also the case for frequency dependent transmission line models in ATP). The exact PI-equivalent is on the form shown in Fig. 7.4.

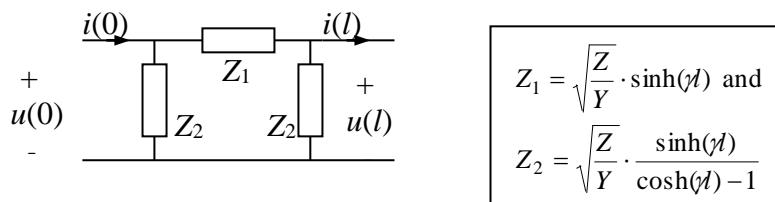


Fig. 7.4 – Exact PI-equivalent

With reference to (13) the calculated series impedance and shunt admittance become

$$Z_s^{Exact-PI} = \left. \frac{3 \cdot u(0)}{i(0) + 2 \cdot i(l)} \right|_{sc} = \sqrt{\frac{Z}{Y}} \cdot \frac{3 \cdot \sinh(\gamma l)}{2 + \cosh(\gamma l)} \approx Z \cdot l \cdot \left( 1 - \frac{(\gamma l)^4}{180} \right) \text{ and}$$

$$Y_s^{Exact-PI} = \left. \frac{3 \cdot i(0)}{u(0) + 2 \cdot u(l)} \right|_{oc} = \sqrt{\frac{Y}{Z}} \cdot \frac{3 \cdot \sinh(\gamma l)}{2 + \cosh(\gamma l)} \approx Y \cdot l \cdot \left( 1 - \frac{(\gamma l)^4}{180} \right) \quad (14)$$

We see that the exact-pi equivalent gives the same result as the distributed parameter model.

### 7.2.2 3-phase systems

#### 1. Positive and zero-sequence

A 3-phase circuit is tested with positive and zero sequence sources applied. In the positive sequence, phase number  $I$  is energized with a sinusoidal source with a phase angle  $-120^\circ \cdot (i-1)$ . In the zero-sequence system all phases are energized with a sinusoidal source with zero phase angle. In cases with several 3-phase circuits in parallel the other circuits are not energized and open. The series impedance and shunt admittance are calculated for each individual phase as deduced above. For example, in phase  $a$ :  $Z_{sa} = 3 \cdot u_a(0)/(i_a(0) + 2 \cdot i_a(l))$ .

#### 2. Self-impedance/admittance

The self-impedance and admittance of the 3-phase circuit  $j$  is defined as the average of the values for each individual phase:  $Z_{jj} = 1/3 \cdot (Z_{sa} + Z_{sb} + Z_{sc})$  and  $Y_{jj} = 1/3 \cdot (Y_{sa} + Y_{sb} + Y_{sc})$  in either the zero- and positive-sequence system.

#### 3. Mutual couplings

Mutual couplings are the equivalent impedance and admittance between circuits. The deduction of these quantities is based on an equivalent two-phase representation shown in Fig. 7.5. Each 3-phase circuit is equated by a single conductor with its self-impedance/admittance and with the average voltage and current distribution.

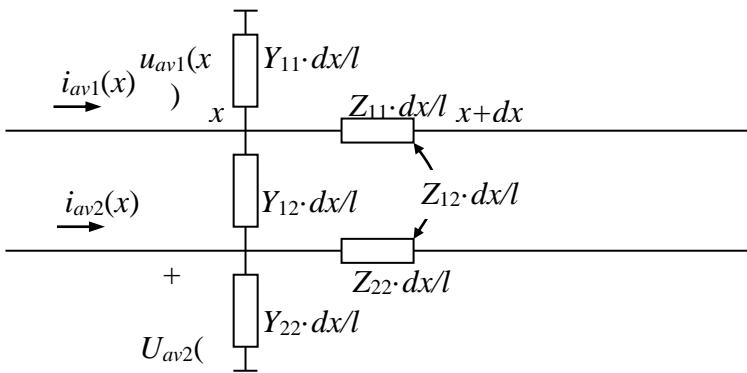


Fig. 7.5 – Two-phase representation

Like the single-phase case, matrix expressions are now developed and approximated by series expansions. The end-result is equal to the single-phase case:

$$u(0) = Z_s \cdot \tilde{r} \quad (15)$$

$$i(0) = Y_s \cdot \tilde{u}$$

with

$$Z_s = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12} & Z_{22} \end{bmatrix}, \quad Y_s = \begin{bmatrix} Y_{11} + Y_{12} & -Y_{12} \\ -Y_{12} & Y_{22} + Y_{12} \end{bmatrix}, \quad (16)$$

$$u(0) = \begin{bmatrix} u_{av1}(0) \\ u_{av2}(0) \end{bmatrix}, i(0) = \begin{bmatrix} i_{av1}(0) \\ i_{av2}(0) \end{bmatrix}, \quad (17)$$

$$\tilde{u} = \begin{bmatrix} \tilde{u}_1 \\ \tilde{u}_2 \end{bmatrix} = \frac{1}{3} \cdot \begin{bmatrix} u_{av1}(0) + 2u_{av1}(l) \\ u_{av2}(0) + 2u_{av2}(l) \end{bmatrix}, \tilde{i} = \begin{bmatrix} \tilde{i}_1 \\ \tilde{i}_2 \end{bmatrix} = \frac{1}{3} \cdot \begin{bmatrix} i_{av1}(0) + 2i_{av1}(l) \\ i_{av2}(0) + 2i_{av2}(l) \end{bmatrix} \quad (18)$$

The unknown mutual impedance and admittance becomes

$$Z_{12} = \left( \frac{u_{av1}(0)}{\tilde{i}_1} - Z_{11} \right) \cdot \frac{\tilde{i}_1}{\tilde{i}_2} \quad (19)$$

$$Y_{12} = \left( \frac{i_{av1}(0)}{\tilde{u}_1} - Y_{11} \right) \cdot \frac{\tilde{u}_1}{\tilde{u}_1 - \tilde{u}_2} \quad (20)$$

In the positive sequence system, the average currents and voltages tend to be very small, and for a perfectly symmetric and transposed systems exactly zero. In such situations the positive sequence coupling has no meaning. The typical test condition is to apply 1 pu current at both circuits with the other ends grounded to obtain the mutual impedance. For mutual admittance the test condition is to apply 1 pu at one and 0 (or -1) pu at the other circuit and leaving the other ends open.

### 7.3 Hybrid Transformer, XFMR

The modeling of the transformer is based on the magnetic circuit transformed to its electric dual [7, 8]. The leakage and main fluxes are then separated into a core model for the main flux and an inverse inductance matrix for the leakage flux. The copper losses and coil capacitances are added at the terminals of the transformer. The resulting electrical circuit is shown in Fig. 7.6. Only standard EMTP elements are used.

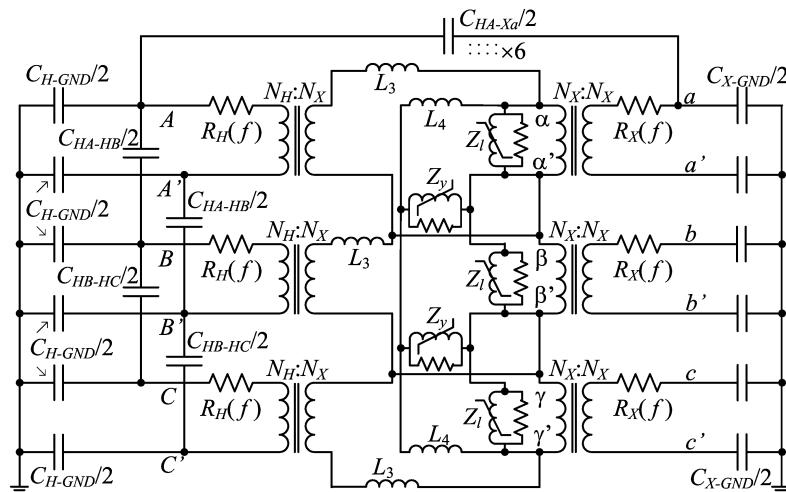
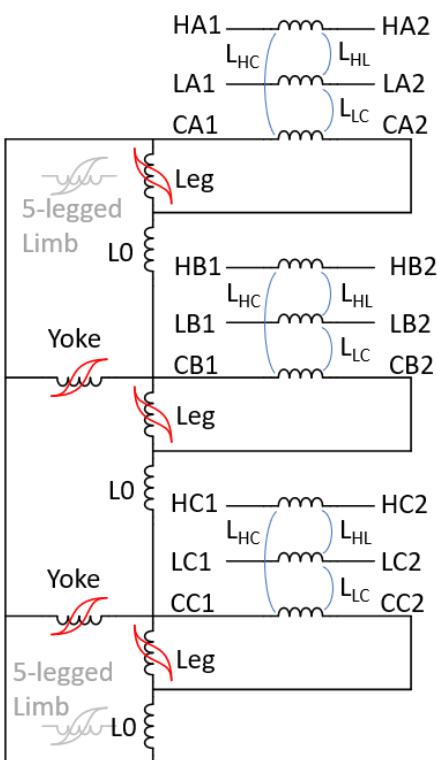


Fig. 7.6a – Electric model of the Hybrid Transformer [9], 2-windings (H and X), 3-phases, 3-legged core.



The figure Fig. 7.6a is not fully correct for the representation of leakage inductances. Instead of lumped elements and ideal transformers, the leakage is modeled via an inverse inductance matrix (A-matrix) just like in BCTRAN. Windings in each phase have zero self-inductance and is only modeled with mutual inductance to the other windings. HA1 and HA2 is the outer winding terminals in phase A. LA1 and LA2 are the inner winding terminals. CA1 and CA2 are the fictitious core winding terminals phase A.

$L_{HL}$  is the main leakage inductance and  $L_{LC}$  represents the space between the core and the inner winding and is set to  $L_{LC}=L_{HL}\cdot k$  (with  $k=0.5$ ) and  $L_{HC}$  represents the space between the outer winding and the core and is set to  $L_{HC}=L_{HL}\cdot(1+k)$ .

The mutual inductance between phases is ignored thus short circuit zero-sequence inductance is not considered.

The open circuit zero sequence inductance is represented by  $L_0$  (off-core flux path).

Fig. 7.6b – Implementation of mutual inductances via the BCTRAN A-matrix approach.

Transformer parameters can be based on three different data sources; typical values, test report, and design information. The three sources can be selected independently for resistance, inductance, capacitance, and core. Test report input is based on standard open- and short-circuits tests, with capacitance measurements as an additional option. This is the normal choice of data source for existing transformers. Design data requires the geometry and material parameters of the windings and the core. Such data are rarely available, so this option is more for research purposes. The Typical value option uses available text book tabulated values of leakage impedance, copper and core losses, and magnetizing current to estimate model parameters. This is suitable when the transformer is not purchased yet, or data is unavailable in an initial study. However, such model must be used with caution.

### 7.3.1 Leakage inductance

The leakage inductance is modeled with an inverse inductance matrix (A-matrix). The matrix has dimension  $(nw+1)\cdot np$  where  $nw$  is the number of physical windings, the core is connected to the  $nw + 1$  winding, and  $np$  is the number of phases [7-9]. The coupling (auto, Y, D), turns ratio, and phase shift are produced directly in the A-matrix. All possible phase shifts are supported. The A-matrix has the following structure for a three-winding, three-phase transformer:

$$A = \begin{bmatrix} A & B & C \\ A_w & 0 & 0 \\ 0 & A_w & 0 \\ 0 & 0 & A_w \end{bmatrix} \text{ where } A_w = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad (1)$$

where ABC are the three phases and PSTC stands for primary, secondary, tertiary, and the core ( $nw+1$ ) winding.

The  $A$ -matrix is assumed to have no mutual coupling between the phases. The entire zero-sequence effect is modeled in the attached core. The  $A_w$ -matrix is established according to the EMTP Theory Book [5] Section 6.4, and Section 5.2.4 p. 31 in [7].

### 7.3.1.1 Typical values

The leakage reactance is established from [11] using the lowest value in the typical range. In the case of a three-winding transformer the leakage reactance (in pu) between the inner and outer winding is approximated as the sum of the other two. In this case it is assumed that the medium voltage winding is the middle one.

### 7.3.1.2 Test report

The leakage reactance is calculated from the standard test report short circuit data (positive sequence).

$$X[\text{pu}] = \sqrt{Z[\%]^2 - (P[\text{kW}] / (10^* S[\text{MVA}]))^2} / 100 \quad (2)$$

In the case of an autotransformer the reactances are scaled according to the Theory Book [5] Section 6.7.

### 7.3.1.3 Design data

The leakage reactances are calculated according to classical MMF distribution theory as shown in [7, 8]. Both cylindrical and pancake windings are supported.

### 7.3.1.4 Handling of the core winding

The artificial core winding is related to the leakage channel between the inner physical winding and the core. A parameter  $K=a_1/a_2$  is defined in [7, 10] where  $a_1$  is the width of the inner leakage channel and  $a_2$  is the width of the leakage channel between the inner and the outer/middle winding. A fixed value  $K=0.5$  is used in ATPDraw. If the pu leakage reactances  $X_{ML}$ ,  $X_{MH}$ , and  $X_{HL}$  ( $L=\text{inner}$ ,  $M=\text{middle}$ ,  $H=\text{outer}$ ) for a three winding transformer are given then the leakage reactances to the core winding are assumed to be [7, 10]

$$\begin{aligned} X_{LC} &\approx K \cdot X_{ML}, X_{MC} \approx X_{LC} + X_{ML} = (K+1) \cdot X_{ML}, \text{ and} \\ X_{HC} &\approx X_{MC} + X_{HM} = (K+1) \cdot X_{ML} + X_{HM} \end{aligned} \quad (3)$$

## 7.3.2 Winding resistance

The winding resistances are added externally at the terminal of the transformer ( $A$ -matrix). Optionally, the resistances can be frequency dependent.

### 7.3.2.1 Typical values

The typical winding resistances (at power frequency) are in principle based on [12]. A function (4) is established that takes in the parameter  $u$  [kV] and  $s$  [MVA] and returns the resistance in %. Data for a 290 MVA/ 420 kV transformer (Table I) were used to extend the data given in [12]:

$$R_w = 0.7537 \cdot \left( \frac{u}{15} \right)^{0.0859} \cdot s^{-0.2759} \quad [\%] \quad (4)$$

### 7.3.2.2 Test report

The test report data are given at power frequency. The per unit short circuit resistances are calculated from short circuit power losses in the test report (positive sequence). The winding resistance (in pu) is assumed to be equally shared between the windings in the case of a two-winding transformer. In the case of a 3-winding transformer the traditional star-equivalent approach is used.

In the case of an auto-transformer the short circuit resistances are recalculated according to the power balance used in [10]. The approach used for reactances (from the Theory Book [5]) did not work out for the resistances.

### 7.3.2.3 Design data

The user can specify the winding conductivity  $\sigma$ , the equivalent cross section  $A$  of each turn, the average length  $l$  of each turn, number of turns of the inner winding  $N$ . The DC resistance is normalized to the power frequency. If the resistance is assumed to be frequency dependent the conductor area must be specified in height and width (which determines the stray losses).

### 7.3.2.4 Frequency dependency

The frequency dependent resistance is calculated between 0.1 to 10 kHz. The typical values and test report resistances are assumed to follow  $R_s(\omega) = R_0 \cdot \sqrt{\omega/\omega_0}$  where  $R_0$  is the resistance at the angular power frequency  $\omega_0$ . This expression results in considerably lower values than suggested in Fig. 26 in [7]. This needs to be further investigated. The design data resistances are assumed to follow eq. (37) in [7].

The calculated  $R(\omega)$  and  $\omega$  value pairs are fitted to the function (two-cell Foster equivalent)

$$R(\omega) = R_0 + \frac{R_1 \cdot \omega^2 \cdot L_1^2}{R_1^2 + \omega^2 \cdot L_1^2} + \frac{R_2 \cdot \omega^2 \cdot L_2^2}{R_2^2 + \omega^2 \cdot L_2^2} \quad (5)$$

$$L(\omega) = \frac{L_1 \cdot R_1^2}{R_1^2 + \omega^2 \cdot L_1^2} + \frac{L_2 \cdot R_2^2}{R_2^2 + \omega^2 \cdot L_2^2}$$

with the resistances  $R_1$  and  $R_2$ , and inductances  $L_1$  and  $L_2$  as unknowns. The fitting routine is based on a Genetic Algorithm implemented in ATPDraw. The object function is defined as  $OF = \min(|R(\omega) - R_s(\omega)| + |\omega L(\omega)|)$  constrained to positive unknowns. A negative inductance  $L_0 = -L_1 - L_2$  is added in series with the winding resistance to compensate for the inductance of the Foster cells. A constraint is put on the total inductance  $|L_0| < L_w$  where  $L_w$  is the inverse of the diagonal  $A_w$ -matrix element, [7] section 5.4.2. The constraint is handled simply by setting  $L_1 = L_2 = 0.5 \cdot L_w$  when the constraint is violated and then continue to obtain new optimized values for  $R_1$  and  $R_2$ .

### 7.3.3 Capacitance

The  $C$ -matrix is split in two halves and connected to each end of the physical windings. The capacitance matrix  $C$  is based on the following two matrices:

$$C_w = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \text{ and } C_p = \begin{bmatrix} C_{AA} & C_{AB} & C_{AC} \\ C_{BA} & C_{BB} & C_{BC} \\ C_{CA} & C_{CB} & C_{CC} \end{bmatrix} \quad (6)$$

The  $C_w$  matrix contains the capacitances between windings 1-3 equal in all phases. The capacitance matrix  $C_w$  is built up like a nodal admittance matrix. The  $C_p$  matrix contains capacitances that are specific to phase A, B, or C. These are typically connected to the outer windings. The total  $C$ -matrix is then built on these two symmetrical matrices dependent on the type of winding (pancake/cylindrical). The concept “outer winding” will be different for pancake and cylindrical windings.

### 7.3.3.1 Typical values

A capacitive coupling factor  $K_c$  can be specified by the user with a default value of 0.3. The concept of transient recovery voltage (TRV) is used to calculate the effective capacitance when the inductance is known [13]. IEEE standard C37, Fig. B2 [14] is used to obtain the typical frequency of the TRV for a known voltage level and fault current.

$$C_{eff}(U, S, X_{pu}, f) = \frac{f}{2\pi} \cdot \frac{3 \cdot I}{U \cdot (f_{TRV}(U, I))^2} \quad [\mu F] \quad (7)$$

with  $U$  in [kV],  $S$  in [MVA] and  $I = \frac{S}{3 \cdot U \cdot X_{pu}}$  [kA]

In the case of typical values, the  $C_p$  matrix (between phases) is always set to zero for lack of any better choice. For a two-winding transformer the  $C_w$  matrix is calculated as

$$\begin{aligned} C_w[1,2] &= C_{PS} = K_c \cdot C_{eff}(U_S, S, X_{PS,pu}, f) \\ C_w[1,1] &= C_{PP} = C_{eff}(U_P, S, X_{PS,pu}, f) - C_w[1,2] \\ C_w[2,2] &= C_{SS} = (1/K_c - 1) \cdot C_w[1,2] \end{aligned} \quad (8)$$

For a three winding transformer the typical capacitance is more complicated with several coupling factors involved. Here a simple approach is used:

$$\begin{aligned} C_w[1,3] &= C_{PT} = 0 \\ C_w[2,3] &= C_{ST} = C_{eff}(U_S, S, X_{ST,pu}, f) - C_w[2,2] \\ C_w[3,3] &= C_{TT} = C_{eff}(U_T, S, X_{ST,pu}, f) - C_w[2,3] \end{aligned} \quad (9)$$

This approach could be further discussed and improved.

### 7.3.3.2 Test report

In the test report the capacitances between each winding and ground and between all windings is assumed to be directly specified while the  $C_p$  matrix is set to zero. All values must be specified per phase.

### 7.3.3.3 Design data

The calculation of design data capacitances are based on [7] chapt. 5.3, p. 33-42. The user has to specify the winding geometry as well as the various equivalent permittivities of insulation system. Standard formulas for calculating the capacitance between cylinders and for cylinders over planes are used with end effect and tank effect adjustments.

### 7.3.4 Core

The core model is connected to the “core winding” terminals of the A-matrix. Triplex (single phase cores), stacked cores with three and five legs, and shell form cores are supported. Basically the inductive and resistive core parts are treated independently, but this is a point that requires more research particularly for 3- and 5-legged cores where harmonics in the flux creates additional losses. The core losses are represented by a linear resistor and the nonlinear inductances are modeled by the Frolich equation (10). Each part of the core is modeled with its own core loss resistance and nonlinear inductance using information about their relative cross section and length to scale the values. Fig. 7.7 shows the core model for a 5-legged transformer.

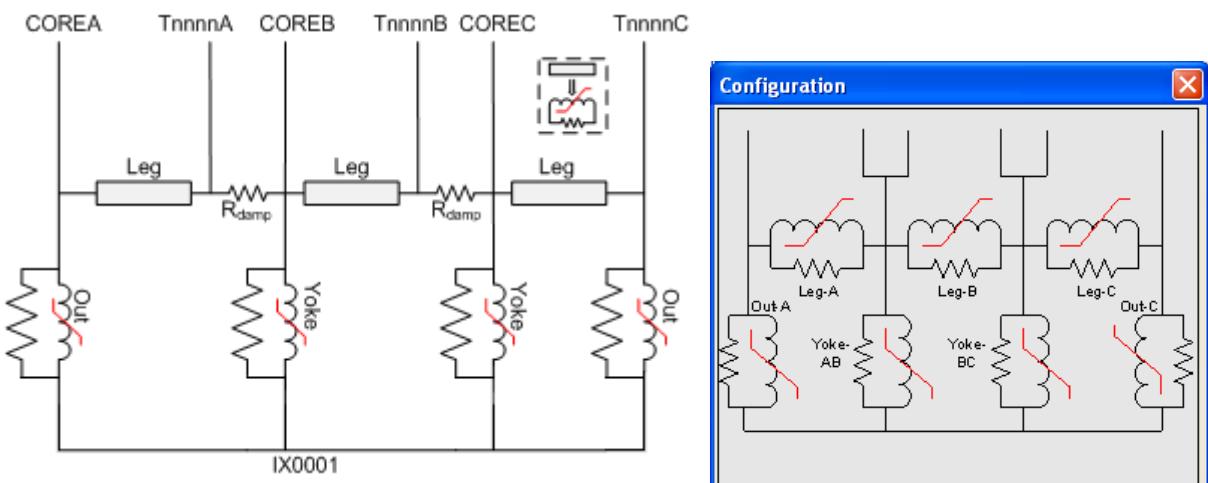


Fig. 7.7 – 5-legged stacked core model. The  $\alpha\beta\gamma$  terminals are the  $n_w+1$  winding.  
Left: Practical ATPDraw implementation. Right: Topologically correct model.

It is assumed that the magnetic material is characterized by five parameters  $a$ ,  $b$ ,  $c$ ,  $d$  and  $e$ . A list of typical steel materials is developed based on fitting the manufacturer’s data from state of the art catalogues. Older steel materials will have a different characteristic and the losses are typically higher. The material list is only used for design data and typical values.

The  $B/H$  relationship is assumed to follow the Frolich equation where the optional parameter  $c$  (introduced in [15]) improves the fitting to test report data around rated voltage

$$B = \frac{H}{a + b \cdot |H| + c \cdot \sqrt{|H|}} + \mu_0 \cdot H \quad (10)$$

The specific loss is assumed to follow

$$P [W/kg] = \left( \frac{f}{50} \right)^{1.5} \cdot (d \cdot B^2 + e \cdot B^{10}) \quad (11)$$

where  $f$  is the power frequency.

The specific loss is traditionally (for instance Westinghouse T&D reference book, 1964) assumed to be  $P = K_e \cdot (f \cdot t \cdot B_{\max})^2 + K_e \cdot f \cdot B_{\max}^x$  with  $x$  said to be 3 for modern materials in the year of 1964. In the above expression  $t$  is the thickness of the laminates. The traditional expression was tested on modern material data with little success.

Fig. 7.8 shows the fit of the specific losses and DC-magnetization curve of ARMCO M4 steel. The Frolich fitting is not very good, and in Fig. 7.8b fitting around the knee point (nominal flux) was preferred at the sacrifice of high field fitting ( $B=1.9$  T). Similar fitting is performed for the other core materials.

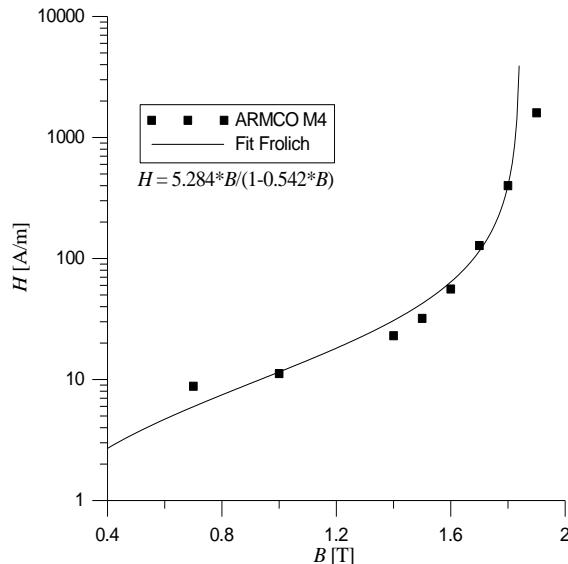


Fig. 7.8a) – Core loss curves ( $c=0$ )

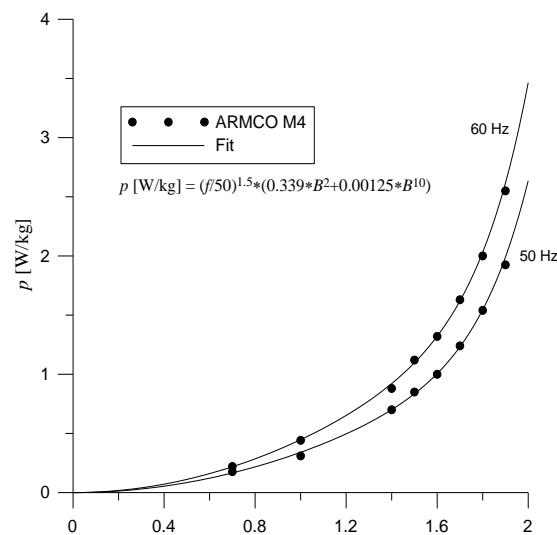


Fig. 7.8b) DC-magnetization curve

#### 7.3.4.1 Inductance modeling:

The basic Frolich equation in (10) is reformulated as a current flux-linkage characteristic by introducing the flux linkage  $\lambda = B \cdot A \cdot N$  and the current  $i = H \cdot l / N$  where  $N$  is the number of turns of the inner winding,  $A$  is the cross section, and  $l$  is the length of the involved core section. This gives

$$\lambda = \frac{i \cdot A \cdot N^2 / l}{a + b \cdot |i| \cdot N / l + c \cdot \sqrt{N / l \cdot |i|}} + \mu_0 \cdot i \cdot A \cdot N^2 / l \Rightarrow \frac{\lambda}{A_r} = \frac{i / l_r}{a' + b' \cdot |i| / l_r + c' \cdot \sqrt{|i| / l_r}} + L_a \cdot i / l_r \quad (12)$$

where the constants  $a' = a \cdot l_L / (N^2 \cdot A_L)$ ,  $b' = b / (N \cdot A_L)$  and  $c' = c \cdot \sqrt{l_L / (A_L^2 \cdot N^3)}$ , based on the absolute length ( $l_L$ ) and cross section area ( $A_L$ ) of the core leg, are determined in an optimization process;

$$\min OF(a', b', c') = \sum_{i=1}^n (I_{meas,rms}(U_{i,rms}) - I_{calc,rms}(U_{i,rms}, a', b', c'))^2 \text{ for } n \text{ excitation levels.}$$

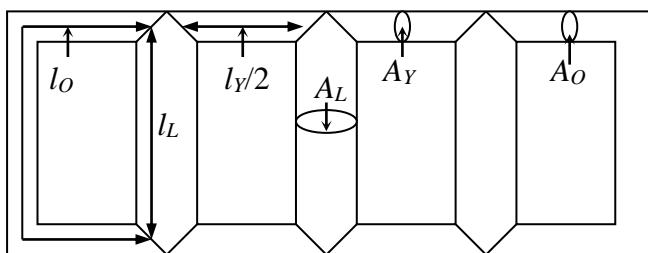


Fig. 7.9 Core dimensions, 5-legged stacked core. The user must provide  $A_Y/A_L$ ,  $A_O/A_L$ ,  $l_Y/l_L$ ,  $l_O/l_L$

The final characteristics are determined by using the relative (to the main leg) dimensions for the corresponding section,  $A_r$  and  $l_r$ . The nonlinear inductances are implemented as optional type 98, 93, or 96 inductances in ATP.

### 7.3.4.2 Core loss modeling

The core loss is split in parts associated with individual core sections. It is assumed that the core loss is proportional to the core volume and to the square of the rms voltage across each section of the electric dual. The voltage,  $U_o$ , in the neutral point in Fig. 7.7 (node IX0001) is the time derivative of the neutral flux found during the Frolich optimization described above. It is assumed that the inductive current components determine the voltage distribution. For a 5-legged core

$$P_{loss} = 3P_{leg} + 2P_{yoke} + 2P_{out} = p \cdot (3 + 2 \cdot V_{ry} \cdot (U_y/U)^2 + 2 \cdot V_{ro} \cdot (U_o/U)^2) \quad (13)$$

where  $V_{ry}$  and  $V_{ro}$  are the relative volumes of the yoke and outer legs respectively.  
and where  $U_y$  and  $U_o$  are the rms value of the voltage across the sections.

For a 3-legged core the outer leg volume is zero and for triplex and shell form core the loss distribution is straight forward and determined only by the main leg voltage.

In the type 96 modelling, half of the loss is included as hysteresis loss scaled by a Steinmetz coefficient of 2. The hysteresis has a uniform width.

### 7.3.4.3 Typical values

The estimation of the magnetizing current ( $I_m$ ) is based on [12]. Some fitting of the data is performed which results in

$$I_m = 0.73 \cdot \left( \frac{BIL}{350} \right)^{0.2933} \cdot \left( \frac{s}{20} \right)^{-0.2154} [\%] \quad (14)$$

when the basic insulation level ( $BIL$ ) is known and

$$I_m = 0.855 \cdot \left( \frac{u}{150} \right)^{0.2283} \cdot \left( \frac{s}{20} \right)^{-0.2134} [\%] \quad (15)$$

when  $BIL$  must be estimated.  $BIL$  is in [kV],  $u$  is the rated voltage in [kV], and  $s$  is the rated power in [MVA].

For a typical core model, the user has to specify the maximum  $B$ -field (normally 1.5-1.7 Tesla) and the maximum core loss density. First a core material has to be guessed and this gives the  $a$  and  $b$  values in the Frolich equation (and possibly also the  $c$  and  $d$  values that would replace  $p$ ).

The following relationships are then assumed:

$$\lambda_{max} = \frac{\sqrt{2} \cdot U_{rms}}{\omega} = B_{max} \cdot A \cdot N \Rightarrow A \cdot N = \frac{\sqrt{2} \cdot U_{rms}}{\omega \cdot B_{max}} \quad (16)$$

$$H_{max} = \frac{a \cdot B_{max}}{1 - b \cdot B_{max}} \approx \sqrt{2} \cdot i_{rms} \cdot \frac{N}{l} \\ \Rightarrow \frac{N}{l} = \frac{a \cdot B_{max}}{(1 - b \cdot B_{max}) \cdot \sqrt{2} \cdot i_{rms}} \quad (17)$$

which simplistically assumes a sinusoidal magnetizing current.

This gives the parameter of the fluxlinkage-current characteristic:

$$a' = a \cdot \frac{l}{A \cdot N^2} \approx \omega \cdot (1 - b \cdot B_{max}) \cdot \frac{i_{rms}}{u_{rms}}, \\ b' = b \cdot \frac{1}{A \cdot N} \approx b \cdot \frac{\omega \cdot B_{max}}{\sqrt{2} \cdot u_{rms}} \quad \text{and} \\ c' = 0 \quad (18)$$

We see that the expressions for  $a'$  and  $b'$  are independent of the magnetic material property  $a$ . The typical value of  $b$  seems to be fairly constant for standard core materials and a value of 0.5 is assumed in ATPDraw.

The core loss is estimated as

$$P_{loss} = p \cdot \rho \cdot A \cdot l = p \cdot \rho \cdot \frac{(1 - b \cdot B_{max}) \cdot 2 \cdot u_{rms} \cdot i_{rms}}{\omega \cdot a \cdot B_{max}^2} \quad (19)$$

where  $p$  [W/kg] and  $\rho$  [kg/m<sup>3</sup>] are given and the volume  $A \cdot l$  is estimated from (16) and (17).

#### 7.3.4.4 Test report

The user specifies the excitation voltage in [%], the current in [%] and the core loss in [kW]. The core loss is used directly as explained above to obtain the core resistances. For now the core resistances are assumed to be linear and the core loss value at 100 % excitation is used.

The inductive magnetizing current for each point is calculated as

$$I_{rms} = \sqrt{I_0[\%]^2 - \left( \frac{P[kW]}{10 \cdot S[MVA]} \right)^2} [\%] \quad (20)$$

This results in a sequence of excitation points ( $U_{rms}$  and  $I_{rms}$ ). The magnetic circuit in Fig. 7.7 assuming sinusoidal fluxes is solved and the rms values of the line currents are calculated and compared to measured ones. Optimized values of  $a'$ ,  $b'$  and  $c'$  (optional) in (12) are found by a Gradient Method implemented in ATPDraw. If a single point is specified, the core model is linear.

#### 7.3.4.5 Design data

For design data the user directly specifies the core material with its B-H relationship ( $a$  and  $b$  values in (10)). The absolute core dimensions and the number of inner-winding turns  $N$  are known, so the inductances can be found directly from (12). Based on manufacturer data the core losses can be established from (11) with  $B = \frac{\sqrt{2} \cdot U_{rms}}{\omega \cdot A \cdot N}$  and known values of the core weight (volume and density) the core loss can be estimated.

### 7.4 Windsyn manufacturers data input and controls

The Windsyn program was developed by the late Gabor First as a stand-alone program for manufacturers data fitting of universal machines in ATP. First, and interface to this program was added to ATPDraw [18], but later the Windsyn approach was directly implemented in ATPDraw.

#### 7.4.1 Induction machine modeling

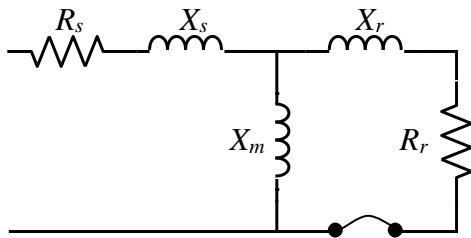
The Windsyn program supports four types of induction machines; single cage, double cage and deep bar rotors as well as wound rotors for doubly fed machines. The first three (single, double, deep-bar) is based on UM type 3 in ATP while the forth (wound) is based on UM type 4. The only difference between double cage rotor and deep-bar is the value of a certain cage factor. Windsyn assumes round rotors (same quantities in d and q axis) and ignores zero sequence parameters and saturation. Basically, a single mechanical mass is assumed, but multi masses can be added externally.

The input parameters (manufacturers data) are; Frequency  $f$  [Hz], line-to-line voltage  $V_L$  [kV], Power  $P$  [hp], Speed  $rpm$  [rpm], Power factor  $\cos\phi$  [pu], efficiency  $\eta$  [pu], slip [pu], start-current  $I_{st}$  [pu], start-torque  $T_{st}$  [pu], load-torque  $T$  [pu], maximum torque (optional)  $T_{max}$  [pu], cage factor  $m$  [pu]. In addition, it is assumed that the rated load current is 1 pu. Fig. 7.10 shows the equivalent circuit in the d-axis of the Induction machine supported by Windsyn. A common pu basis is chosen so that

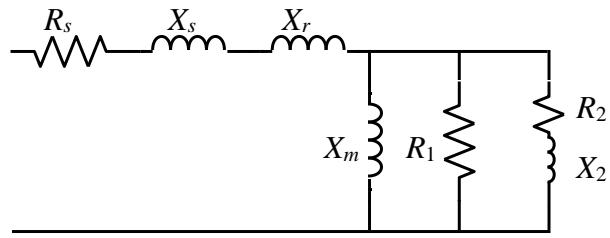
$$Z_{pu} = U^2 / S \quad (21)$$

where  $S$  is the rated power of the machine in [MVA] and  $U$  is the rated voltage in [kV].

The rated power of the machine is actually given in horse power in the input dialog so rated power is given by  $S = hp \cdot 0.746 \cdot 10^{-3} / \cos\phi$ .



Single cage and wound rotor



Double cage and deep bar rotor

Fig. 7.10 Windsyn Induction machine models in start-up.

The circuits above are considered in two different conditions; Startup where the equivalent is as shown and Rated conditions where the rotor resistances are divided by the slip  $s$ . Note that the rotor inductance is moved to the left side of magnetizing inductance for double cage and deep-bar rotors, because this is the only way the model fits to the ATP requirements.

When performing the fitting the stator current and electric torque is used, either at startup ( $s=1$ ) or at rated power ( $s$  given). The current is equal to the admittance seen in from the stator side, while the electric torque is equal to square of the absolute value of the rotor current times the real value of the rotor impedance. If in addition the maximum torque is required, the slip is varied in (23) until maximum is found in an iterative process.

$$I_s = \frac{Z_r / Z_m + 1}{Z_s \cdot (Z_r / Z_m + 1) + Z_r} \quad [pu] \quad (22)$$

$$T_e = \left| \frac{1}{Z_s \cdot (Z_r / Z_m + 1) + Z_r} \right|^2 \cdot \operatorname{Re}(Z_r) \quad [pu] \quad (23)$$

where

$$Z_s = \begin{cases} R_s + jX_s & , \text{single cage and wound rotor} \\ R_s + jX_s + jX_r & , \text{double cage and deep bar rotors} \end{cases}$$

is the equivalent stator impedance

$$Z_r = \begin{cases} R_r / s + jX_r & , \text{single cage and wound rotor} \\ (R_1 / s) \parallel (R_2 / s + jX_2) & , \text{double cage and deep bar rotors} \end{cases}$$

is the equivalent rotor impedance

$$Z_m = jX_m \text{ is the magnetization impedance}$$

The stator and rotor inductances are further assumed to be equal. The rotor secondary reactance  $X_2$  is related to the rotor resistance through the cage factor  $m$ ,  $X_2 = (R_1 + R_2)/m$

According to [19] the stator resistance, rotor resistance and magnetizing reactance are given by

$$\begin{aligned} R_s &= \cos \phi \cdot \left( 1 - \frac{\eta}{1-s} \right) \\ R_r &= \frac{s \cdot \eta}{(1-s) \cdot \cos \phi} \\ X_m &= \frac{\eta}{(1-s) \cdot \sin \phi} \end{aligned} \quad (24)$$

where  $\cos \phi$  is the power factor,  $\eta$  is the efficiency corrected for core loss, friction and windage, and  $s$  is the slip at rated load. The calculated values in (24) are used as an initial guess in the fitting. The expression for  $R_s$  in (24) implies the restriction  $\eta < 1-s$  in order to make the stator resistance positive.

The electrical parameters in Fig. 7.10 are not fitted directly for double cage or deep-bar rotors since they are linked together through the efficiency and power factor. Instead an equivalent rotor resistance  $R_{st}$  is used. This is the real part of the rotor impedance at startup. According to [19] we can then write

$$\begin{aligned} R_1 &= R_{st} (1+m^2) - R_r \cdot m^2 \\ R_2 &= R_1 \cdot R_r / (R_1 - R_r) \end{aligned} \quad (25)$$

An initial value for the equivalent rotor resistance is according to [19]

$$R_{st} = R_r \cdot T_{st} \cdot (\cos \phi / I_{st})^2 / s \quad (26)$$

An initial value for the stator and rotor reactances, which in contrast to [19] are assumed linear, are

$$X_s = X_r = \frac{1}{2} \cdot \sqrt{\frac{1}{I_{st}^2} - (R_s + R_r)^2} \quad (27)$$

The Windsyn program asks for the saturation current but this value isn't used.

#### 7.4.2 The ATPDraw fitting approach

ATPDraw lets the following variables be free in the optimization process; Stator reactance (=Rotor reactance)  $x[0]=X_s$ , Magnetization reactance  $x[1]=X_m$ , Equivalent rotor resistance  $x[2]=R_{st}$ , and Slip  $x[3]=s$ . The slip is generally a free variable, but a deviation from rated value penalty is also added to the cost function. Starting values are calculated according to (24), (26) and (27). Changes in the variables will result in new efficiency, power factor and stator resistance.

The following defined object function is to be minimized

$$\begin{aligned} OF(X_0..X_3) &= w_1 \cdot (1 - \cos \phi / \cos \phi_0)^2 + w_2 \cdot (1 - \eta / \eta_0)^2 \\ &\quad + w_3 \cdot (1 - X[3] / s_0)^2 + w_4 \cdot (1 - I_{st} / I_{st0})^2 \\ &\quad + w_5 \cdot (1 - T_{st} / T_{st0})^2 + w_6 \cdot (1 - I / I_1)^2 + w_7 \cdot (1 - T_{max} / T_{max0})^2 \end{aligned} \quad (28)$$

where  $w_1..w_7$  are user-definable weights.

In Windsyn a successive, non-convergent, iterative procedure was used with effectively  $w_1=w_2=w_3=w_7=0$ , and  $w_6$  set to a very high value. The rated torque is not used in the fitting process since

the related rated current  $I_0=1$  is used instead. Some deviations will be seen from the original Windsyn as the ATPDraw version converge at the exact minimum, while Windsyn stopped the iteration at a fixed accuracy.

To calculate the object function the following process is used:

- The estimate for stator and rotor resistances are updated from (24) using the new slip  $s=X[3]$  (but using the initial power factor and efficiency).
- The rotor resistances for double cage and deep-bar are calculated according to (25) using the equivalent rotor resistances  $R_{st}=X[2]$  and  $R_r$ .
- The equivalent circuit in fig. 1 is thus defined and the currents ( $I$  and  $I_{st}$ ) are calculated according to (22) and the torques ( $T$ ,  $T_{st}$  and  $T_{max}$ ) are calculated according to (23).
- New power factor and efficiency are calculated as  $\cos\phi=\text{Re}(I)/|I|$  and  $\eta=(1-s)\cdot(T/\text{Re}(I))$ .

The object function is called by the BSGF routine [20] in ATPDraw (same as used for Hybrid Transformer fitting and Optimization). A lower constraint is set on the variables to make them positive.

#### 7.4.3 ATPDraw input dialogs

The new machine model is found as ‘Induction WI’ under Machines in the selection menu. It has a single input dialog box as any other component as shown in Fig. 7.11 . Manufacturer’s data are inputted in the data grid (expanded here for illustrative purposes). Under *Model* the type of rotor is selected as well as the mechanical data. Also a Governor (G=General, H=Hydro) can be selected here. Under Start-up the initialization and extra load conditions are specified as shown in Fig. 7.12. The user must click the *Fit & View* button to perform the actual fitting as shown in Fig. 7.13. Here adjustment of parameters and weights can be performed. The corresponding electrical parameters are given to the right and the rated torque given below in Nm. ATPDraw can also calculate and present the torque-speed characteristic as shown in Fig. 7.14.

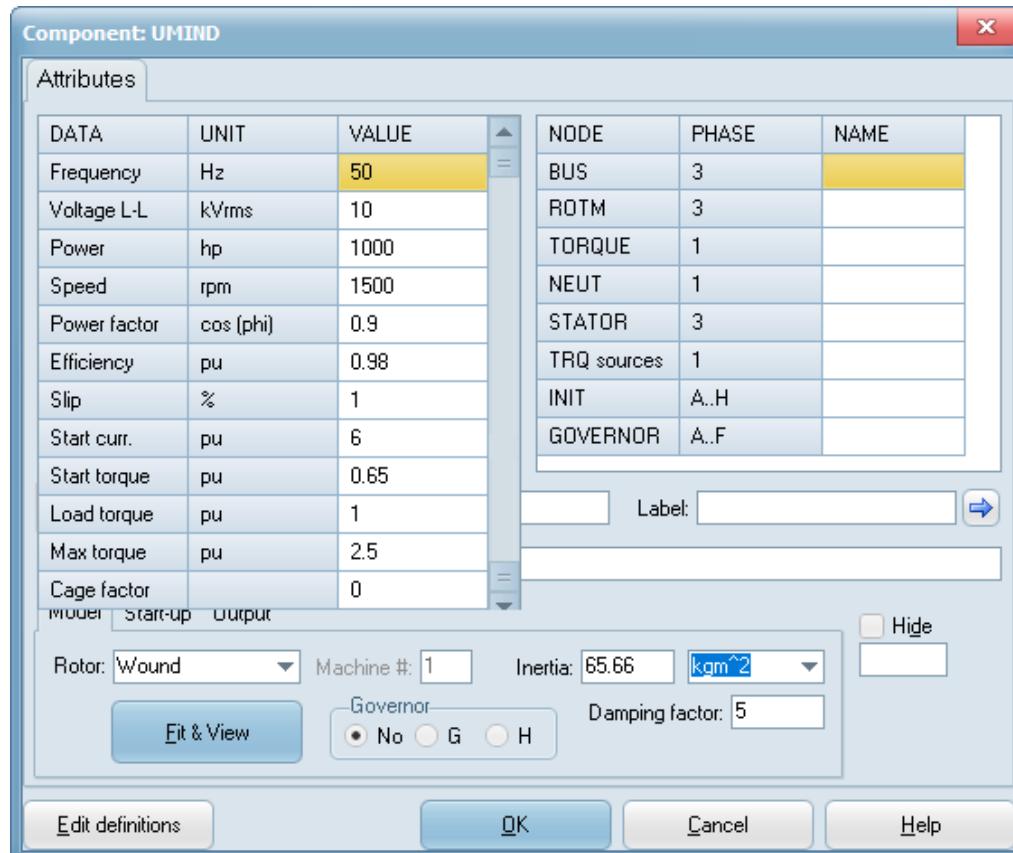


Fig. 7.11 Input Dialog of Induction machine WI.

For Automatic initialization (ATP|Settings/Switch/UM or Sidebar) the user should specify the slip as calculated by the Fitter to get the adjusted rated power out. For Manual initialization the initial torque in pu is specified instead. The user can also specify an optional extra load applied to the machine. Positive signs are for motors and negative sign for generators.

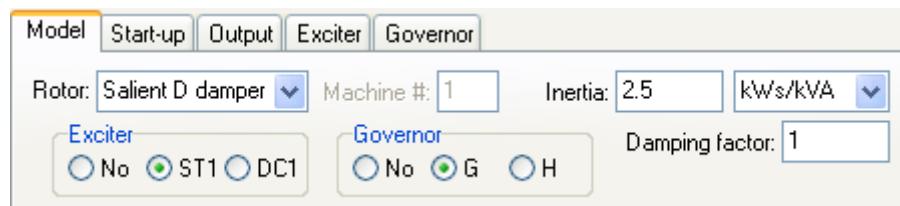


Fig. 7.12a Model selection page for synchronous machines

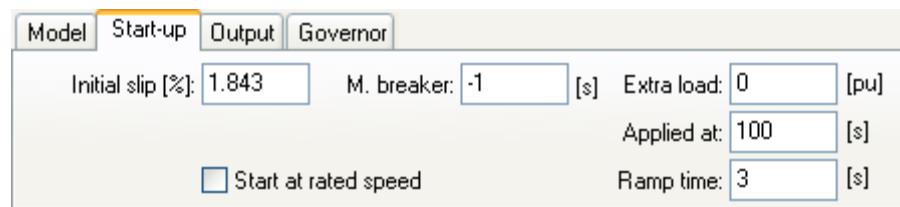


Fig. 7.12b Settings for Automatic start-up. For synchronous machines the initial voltage in [%] and angle must be specified instead of the slip.

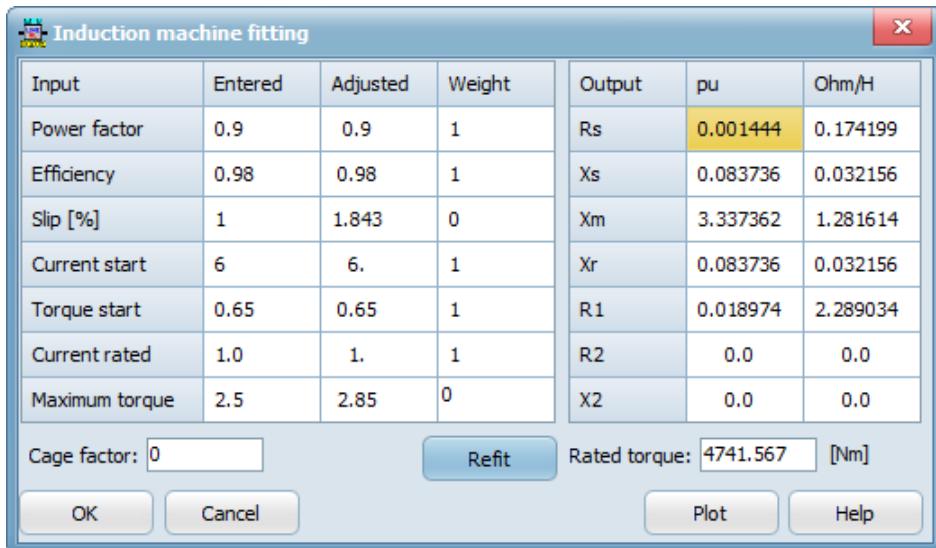


Fig. 7.13 Fitting dialog. Slip fitting relaxed ( $w_3=0$ ) here similar to Windsyn.

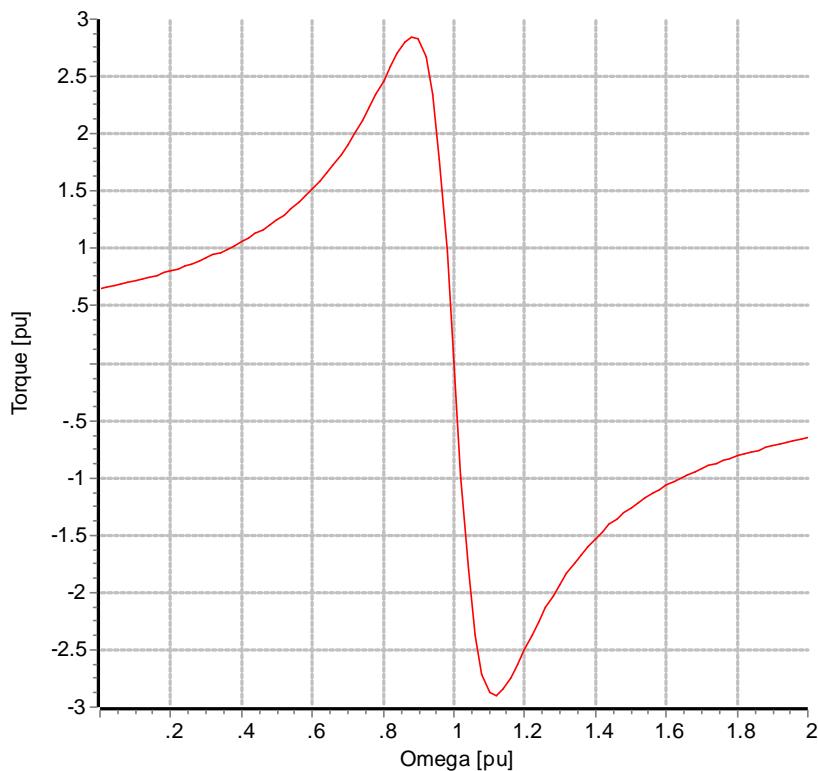


Fig. 7.14 Torque-speed curve obtained by clicking on Plot in Fig. 4.

#### 7.4.4 Synchronous machine modeling

The synchronous machine modeling process in Windsyn is more straight forward as there is not fitting involved. The zero-sequence and Canay impedances are ignored. Time constants are assumed to come from open circuit tests.

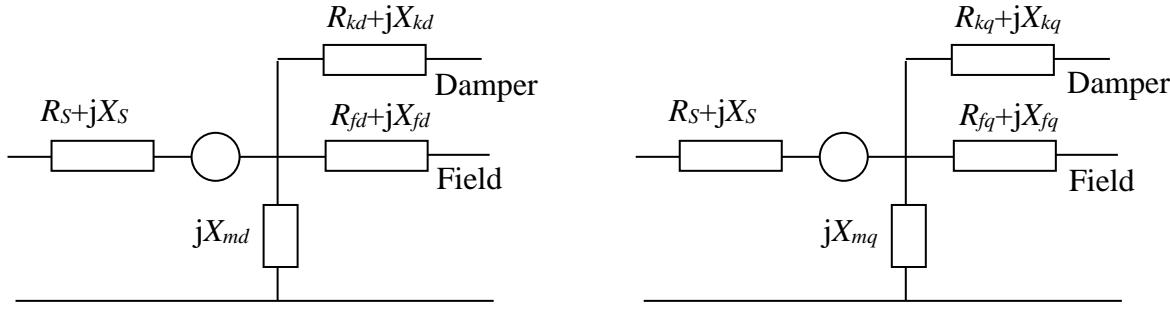


Fig. 7.15 Synchronous machine equivalent, d-axis (left) and q-axis (right).

Pu quantities are also used for synchronous machines according to (21). Further without any iteration or fitting:

Stator resistance  $R_s = 1 - \eta$

Stator reactance  $X_s = X_l$  (=leakage)

Magnetization reactance d-axis:  $X_{md} = X_d - X_l$

Magnetization reactance q-axis:  $X_{mq} = X_q - X_l$

$$\text{Field winding reactance d-axis: } X_{fd} = \frac{(X_d - X_l) \cdot (X_d' - X_l)}{X_d - X_d'}$$

$$\text{Damper winding reactance d-axis: } X_{kd} = \frac{(X_d' - X_l) \cdot (X_d'' - X_l)}{X_d' - X_d''}$$

$$\text{Field winding reactance q-axis: } X_{fq} = \frac{(X_q - X_l) \cdot (X_q' - X_l)}{X_q - X_q'}$$

$$\text{Damper winding reactance q-axis: } X_{kq} = \frac{(X_q' - X_l) \cdot (X_q'' - X_l)}{X_q' - X_q''}$$

$$\text{Field winding resistance d-axis: } R_{fd} = \frac{(X_{fd} + X_{md})}{\omega \cdot T_{d0}'} \text{ if not provided directly by user}$$

$$\text{Field winding resistance q-axis: } R_{fq} = \frac{(X_{fq} + X_{mq})}{\omega \cdot T_{q0}'}$$

$$\text{Damper winding resistance d-axis: } R_{kd} = \frac{(X_{kd} + X_{md})}{\omega \cdot T_{d0}''}$$

$$\text{Damper winding resistance q-axis: } R_{kq} = \frac{(X_{kq} + X_{mq})}{\omega \cdot T_{q0}''}$$

The presence of damper windings depends on the user's choice of rotor type.

#### 7.4.5 Machine controls

The machines can have embedded Governor control (torque) to maintain a set speed or active power. The Synchronous machine can also have an Exciter control (field voltage) for voltage or reactive power settings. All the controls are simplified with minimum TACS requirements.

Advanced machine studies need probably to extend the controls which is possible since the Torque and Field Voltage nodes are externally available. The TACS section also provides power, speed and rms voltage measurements.

#### 7.4.5.1 Governor

Both the Induction Machine and the Synchronous machine comes with an optional Governor of either General purpose or Hydro type. Both are simplified controls aimed at minimum TACS requirements.

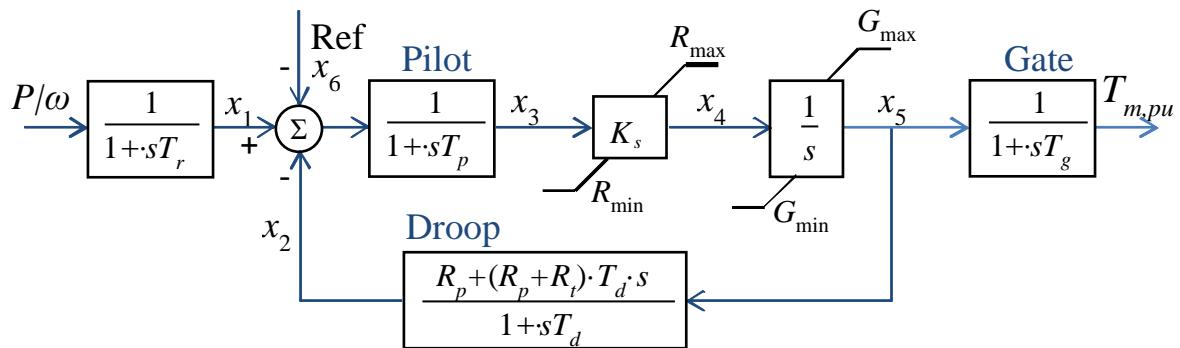


Fig. 7.16 Hydro type Governor [21]

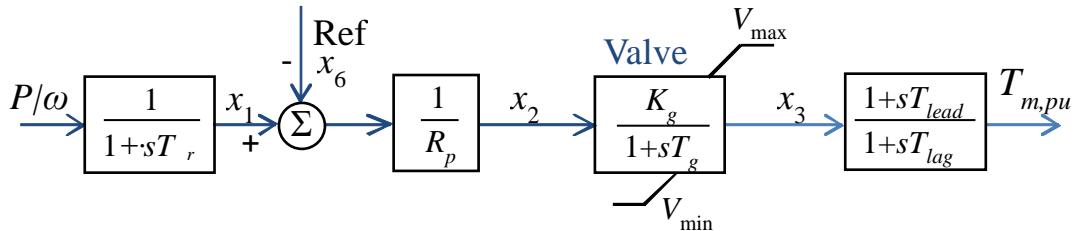


Fig. 7.17 General purpose Governor [21]

The state variables  $X_1..X_6$  are TACS variables given by the GOVERNOR node name with phase extension A..F respectively.

The torque node of the machines are available for manual or additional control of the machine. Torque is modeled as a current (1 A = 1 Nm) with positive current into the node for generators. The voltage at the torque node is the rotor velocity (1 V = rad/sec).

The rated torque is calculated as  $TQ = \begin{cases} W \cdot \eta / \Omega, & \text{for motor} \\ W / (\eta \cdot \Omega), & \text{for generator} \end{cases}$

where  $W$  is the active power of the machine calculated as  $hp \cdot 746$  for induction machine and  $kVA \cdot 1000 \cdot \cos \phi$  for synchronous machines,

$\eta$  is the machine efficiency, and

$\Omega$  is the rated speed equal to  $rpm \cdot \pi / 30 \cdot (1 \pm s)$  with positive  $s$  used for induction generator and negative  $s$  used for induction motor. For synchronous machines  $s$  is zero.

A damping resistor is connected at the mechanical side with a value  $D = \frac{\Omega \cdot DampFact}{0.3 \cdot TQ}$

where *DampFact* is user controllable as shown in Fig. 7.11 . The torque consumed by the damping is taken into account when setting the initial torque.

#### 7.4.5.2 Exciter

The Synchronous Machine comes with an optional Exciter of type IEEE DC or ST. The exciter can be set to voltage control or reactive power control.

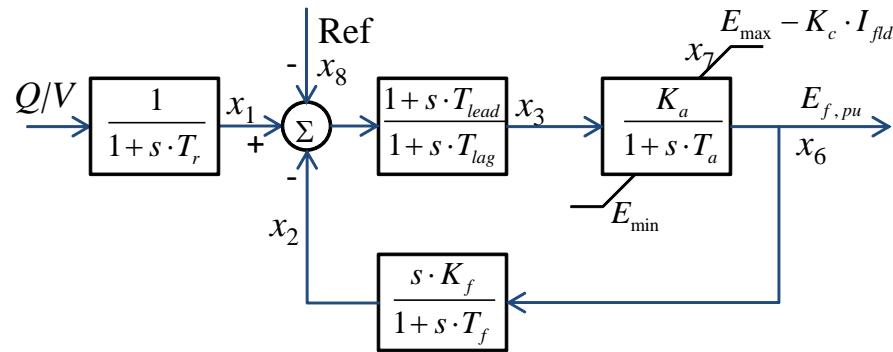


Fig. 7.18 Exciter ST1 [21]

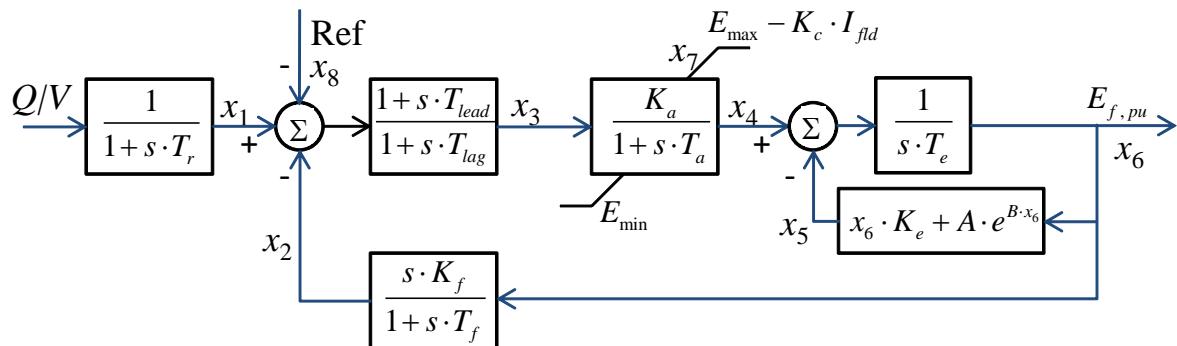


Fig. 7.19 Exciter DC1 [21]

The state variables  $X_1..X_8$  are TACS variables given the EXCITER node name with phase extension A..H respectively.

The field voltage node of the machines are available for manual or additional control of the machine. The 'EFD winding' node is the field winding terminal while the 'EXFD' node is where the initialization source AND the exciter source are connected (voltage sources in parallel are summed in ATP) as shown in Fig. 7.20. Any additional external field voltage source should be connected directly at the external field voltage node and NOT via resistors or switches.

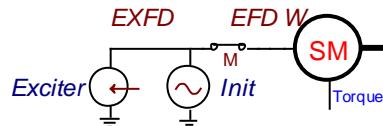


Fig. 7.20 Connection of field voltage sources

#### 7.4.5.3 IO and measurements

The rms voltage, active and reactive power is used in the controls but these variables are also available for output requests along with other machine quantities as shown in Fig. 7.21.

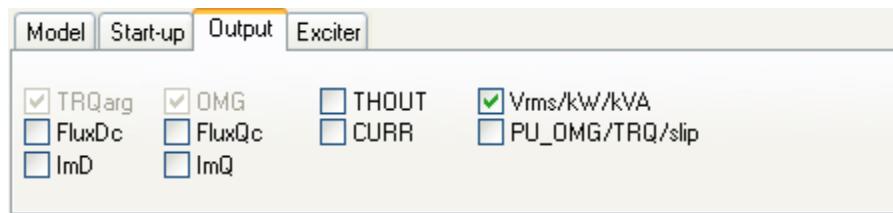


Fig. 7.21 Output selection

The stator winding is connected to the internal node 'STATOR' which is connected via the main machine breaker to the external node 'BUS'. The three phase voltages at the stator node and the current into the stator phase A are always available as variables in the TACS section. The internal INIT node is used as intermediate variables; INITA and INITB are 90 deg. phase shifts of the phase A stator voltage and current respectively. The active and reactive power and the rms voltage are calculated and filtered (with a  $\tau=10$  ms low-pass filter) as shown below:

```

90STATA
90STATB
90STATC
91BUSA
98INITA 53+BUSA          0.005 0.005
98INITB 53+STATA         0.005 0.005
98VARX1 = 1.5* (STATA *INITA -BUSA *INITB )
98WATX1 = 1.5* (STATA *BUSA +INITA *INITB )
1KVAR1 +VARX1             0.001 { kVar
1.
1.      0.01
1KWAT1 +WATX1            0.001 { kWatt
1.
1.      0.01
98VRMS1 =SQRT (STATA **2+STATB **2+STATC **2)
1PUVT1 +VRMS1            1.E-4 { pu volt
1.
1.      0.01
33PUVT1 KWAT1 KVAR1

```

#### 7.5 Power system toolbox calculators

The Power System Toolbox contains various components for converting transients to steady-state quantities. In addition to this, the voltage and current probes add models (WRITEPROBEI or WRITEPROBEV) for phasor calculations behind the scene if steady-state output beyond time zero

is requested. Common to all the Power System Tools is that they are mostly open and built on MODELS.

### 7.5.1 Filtering and down-sampling

Separate components for analogue filtering are available in the toolbox. These are 3-phase Butterworth low and high pass filters of user selectable order 1 to 3. A low-pass filter should be used for anti-aliasing, with a filter frequency less than the half of sampling frequency. The high-pass filter can be used to remove sub-harmonics. In both components the user sets a Gain, filter frequency *FilterFreq*, an order (1-3) *FilterOrder*, and a frequency for amplitude correction *ScaleFreq*. The low-pass algorithm in MODELS code is shown below:

```
INIT
tau:=recip(2*PI*FilterFreq)
beta:=ScaleFreq*recip(FilterFreq)
-- amplitude correction at ScaleFreq
alpha:=Gain*sqrt((1+beta** (2*FilterOrder)))
ENDINIT
EXEC
FOR n:=1 TO 3 DO
  if FilterOrder=1 then
    claplace(XF[n]/X[n]):=(alpha|s0)/(1|s0+tau|s1)
  elsif FilterOrder=2 then
    claplace(XF[n]/X[n]):=(alpha|s0)/(1|s0+sqrt(2)*tau|s1+tau**2|s2)
  elsif FilterOrder=3 then
    claplace(XF[n]/X[n]):=(alpha|s0)/(1|s0+2*tau|s1+2*tau**2|s2+tau**3|s3)
  else
    XF[n]:=Gain*X[n]
  endif
ENDFOR
ENDEXEC
```

After filtering the signal should be sampled. Often the time step of the simulation is small which gives memory overflow in the later processing. As a result, down-sampling must be used, also to mimic actual relay algorithms. The MODELS language provides an easy way of down sampling with a single line of code: `timestep min: 1/SampleFreq`. For this to work properly `1/SampleFreq` must be multiples of the global timestep in the simulation. Otherwise the MODELS internal time step will be slightly different during the simulation. Furthermore, as 8 sampler per period is used by the FFT algorithm, it is beneficial to also let `SampleFreq` be multiple of `8*FREQ`. It is difficult to meet the above requirements, especially in 60 Hz systems, so in v7.3 a new approach was introduced inspired by the interpolation technique in the COMTRADE objects. First, the MODELS method of down sampling is utilized to avoid memory overflow:

```
timestep min: recip(8*SampleFreq).
```

Next, internal interpolation in each MODEL is used with the variable `DTi:= T-Ti` where `Ti` is the actual sampling time managed inside each MODEL.

The following models have down-sampling functionality; ABC2RMS, ABC2SEQ, ABC2PHR, ABC2PHRI, ABC2PHRF, ABC2PHR2, UI2PQ, UI2PQ3, UI2RX, UI2RXL, UI2RXE, WRITEPROBEI, WRITEPROBEV. The last two are used with probes for  $T>0$  phasor calculations.

## 7.5.2 Phasor calculations

With down-sampling, the original recursive DFT algorithm [22] got substantially reduced accuracy and was enhanced by Radix2-8/16 FFT-routines [23-24]. Models containing this routine require *SampleFreq* input that should be multiples of  $8*FREQ$  for the Radix2-8 routine and  $16*FREQ$  for the Radix2-16 routine. Many of the models have also an algorithm selector where *Algorithm*=1 gives a recursive DFT algorithm [22], while *Algorithm*=0 gives the Radix2 algorithm. Both routines use a moving window covering one period. The DFT algorithm is recursive and adds increments as the window advances, while the FFT routine uses 8 or 16 samples of the entire period. The recursive DFT algorithm is the fastest, but errors accumulate more easily. Compared to [22], the low-pass filtering is now excluded as this must be performed before (down) sampling.

### 7.5.2.1 Recursive DFT routine

The recursive DFT algorithm is implemented in MODELS. Originally in [22], the algorithm assumed a linear signal within the time step, but is now simplified to a stepwise signal.

$$\begin{aligned} a_h(t) &= \frac{\omega}{\pi} \int_{t-T}^t y(\tau) \cdot \cos(h\omega \cdot \tau) \cdot d\tau \approx \frac{\omega}{\pi} \int_{t-T}^{t-\Delta t} y(\tau) \cdot \cos(h\omega \cdot \tau) \cdot d\tau + \frac{\omega}{\pi} \cdot \Delta t \cdot y(t) \cdot \cos(h\omega \cdot t) \\ a_h(t - \Delta t) &= \frac{\omega}{\pi} \int_{t-\Delta t-T}^{t-\Delta t} y(\tau) \cdot \cos(h\omega \cdot \tau) \cdot d\tau \approx \frac{\omega}{\pi} \int_{t-T}^{t-\Delta t} y(\tau) \cdot \cos(h\omega \cdot \tau) \cdot d\tau + \frac{\omega}{\pi} \cdot \Delta t \cdot y(t-T) \cdot \cos(h\omega \cdot (t-T)) \quad (29) \\ \Rightarrow a_h(t) &= a_h(t - \Delta t) + \frac{\omega}{\pi} \cdot \Delta t \cdot (y(t) - y(t-T)) \cdot \cos(h\omega \cdot t) \end{aligned}$$

In the MODELS syntax with  $h=1$ , this becomes ( $\omega \cdot \Delta t / \pi = 2/NSAMPL$ ):

```
EXEC
  DTi := T - Ti
  if DTi >= 0
  then
    for i := 1 to 3 do
      D := 2 * FREQ * dt * (delay(X[i], DTi, 2) - delay(X[i], 1 / FREQ + DTi, 2))
      re[i] := re[i] + D * cos(OMEGA * Ti) * Scale
      im[i] := im[i] - D * sin(OMEGA * Ti) * Scale
    endfor
    Ti := Ti + recip(SampleFreq)
  endif
ENDEXEC
```

The new algorithm is faster compared to [22], but the accuracy is lower, especially for low sampling rates 8 and 16 times the fundamental frequency. One side benefit with the simplified algorithm is that it is more stable when the sampling interval ( $1/SampleFreq$ ) becomes smaller and is not an integer number of the time step (as supposed to).

### 7.5.2.2 FFT Radix2 routine

The FFT Radix 2 algorithm was implemented based on [24]. This based on classical FFT theory and the Cooley-Tukey algorithm with decimation in time. A sample array x with length N being the power of 2 is sub-divided in even and odd terms. The discrete Fourier transformer thus becomes:

$$X_k = \sum_{m=0}^{N-1} x_m \cdot W^{mk} = \sum_{m=0}^{N/2-1} (x_{2m} + W^k \cdot x_{2m+1}) \cdot W^{2mk} \text{ with } W = e^{-j2\pi/N} \quad (30)$$

The sub-division in odd and even parts then continues until only two terms remains. With  $N=8$  (and 16) closed form analytical expressions can be obtained for the FFT:

$$X(k,8) = (-1)^{k+1} \left[ x_0 - x_4 + \alpha^{k/4} \cdot (x_2 - x_6) + \alpha^{k/8} \cdot ((x_1 - x_5) + \alpha^{k/4} \cdot (x_3 - x_7)) \right]$$

$$\text{with } \alpha^n = \cos(2\pi \cdot n) + j \sin(2\pi \cdot n)$$

The fundamental harmonic  $X(1,8)$  can further be written out in real and imaginary parts containing only one scaling factor coming from  $\alpha^{1/8} = \frac{1+j}{\sqrt{2}}$  as  $\alpha^{1/4} = j$ .

The corresponding MODELS code is shown below where delay is used to obtain  $x_1$  to  $x_7$  ( $x_0$  is the present value).

```

INIT
  OMEGA:= 2*PI*FREQ
  alpha:=1/sqrt(2)
  for n:= 0 to 7 do
    delta_T[n] := n/(FREQ*8)
  endfor
ENDINIT
EXEC
  DTi:=T-Ti
  if DTi>=0
  then
    for i:=1 to 3 do
      x1 := delay(x[i],delta_T[0]+DTi,2) - delay(x[i],delta_T[4]+DTi,2)
      x3 := delay(x[i],delta_T[2]+DTi,2) - delay(x[i],delta_T[6]+DTi,2)
      x5 := delay(x[i],delta_T[1]+DTi,2) - delay(x[i],delta_T[5]+DTi,2)
      x7 := delay(x[i],delta_T[3]+DTi,2) - delay(x[i],delta_T[7]+DTi,2)
      xre := x1 + (x5 - x7)*alpha
      xim := x3 + (x5 + x7)*alpha
      re[i] := (xre*cos(OMEGA*Ti)+xim*sin(OMEGA*Ti))/4*Scale
      im[i] := (xim*cos(OMEGA*Ti)-xre*sin(OMEGA*Ti))/4*Scale
    endfor
    Ti:=Ti+recip(SampleFreq)
  endif
ENDEXEC

```

A Radix2-16 ( $N=16$ ) algorithm [6] is used to achieve better accuracy for higher harmonics in the ABC2PHRH2 component. The FFT can still be written on a fairly closed form.

### 7.5.2.3 Initialization and variable frequency

Two new phasor calculation routines are introduced; ABC2PHRI and ABC2PHRF. In the former, the steady state information is utilized to properly initialize the phasor value without the need to wait for one period. In the latter, the frequency is not a fixed quantity but an input variable which can be calculated by the new PLLDQ component. In this case the FFT Radix2 algorithm is required.

The initialization algorithm has also the imaginary part of the steady-state phasor as input and performed an initialization as shown below (the real part is the normal input):

#### HISTORY

```

X[1] {DFLT:sqrt(reX[1]**2+imX[1]**2)*cos(t*omega+atan2(imX[1],reX[1]))}
X[2] {DFLT:sqrt(reX[2]**2+imX[2]**2)*cos(t*omega+atan2(imX[2],reX[2]))}
X[3] {DFLT:sqrt(reX[3]**2+imX[3]**2)*cos(t*omega+atan2(imX[3],reX[3]))}

```

### 7.5.3 Power and impedance calculations

The algorithm shown in the listing above is used also for PQ and RX calculations and in that case the phasors of two 3-phase inputs current  $I[1..3]$  and voltage  $V[1..3]$  are calculated. This gives

$$R + jX = \frac{V_{re} + jV_{im}}{I_{re} + jI_{im}} \quad \text{and} \quad P + jQ = (V_{re} + jV_{im}) \cdot (I_{re} - jI_{im}) \quad (31)$$

For distance relays the positive sequence impedance is needed and this is found by calculate line quantities, shown below for the (real part of the) voltage:

$$VL[n] := (V[n] - V[(n \bmod 3) + 1])$$

Ground fault relays also requires a zero-sequence correction. A new ABC2RXE component is added where the zero sequence impedance ratio  $K0 = \frac{Z_0}{Z_1} - 1$  must be provided by the user. The positive sequence impedance is then calculated as:

```

reI0:=(reI[1]+reI[2]+reI[3])/3
imI0:=(imI[1]+imI[2]+imI[3])/3
for n:=1 to 3 do
    reI[n]:=reI[n]+K0*reI0
    imI[n]:=imI[n]+K0*imI0
    D:=reI[n]**2+imI[n]**2
    R[n]:=(reV[n]*reI[n]+imV[n]*imI[n])*ScaleV*recip(D*ScaleI)
    X[n]:=(imV[n]*reI[n]-reV[n]*imI[n])*ScaleV*recip(D*ScaleI)
endfor

```

### 7.5.4 FFT/DFT algorithm test

The phasor calculation algorithms are compared in the figure below, where also the DFT algorithm from ATPDraw 7.2 is added. The purpose is to test the accuracy of the Power calculation ( $V^*I$ ) for various sample frequencies. Reactive power calculations in a 60 Hz system are shown because this is the most demanding case. The time step is in all the calculations 0.1 ms.

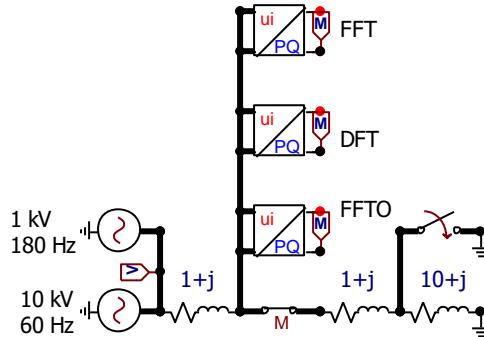


Fig. 7.22 Test circuit for RX calculations. 10% 3<sup>rd</sup> harmonic added.

Fig. 7.23 shows the comparison of the three algorithms (only FFT (Radix2-8) and DFT can be selected in the power system toolbox in ATPDraw v7.3). The FFT0 algorithm is from ATPDraw v7.2. We see that the new FFT algorithm is accurate in all cases, while the old FFT algorithm has problems when the sampling frequency is too large for the time timestep. To some minor extent this also applies to the new DFT algorithm. The old FFT algorithm also has potential problems for low sampling frequencies (shown for 480 S/s). Reducing the timestep will increase the accuracy.

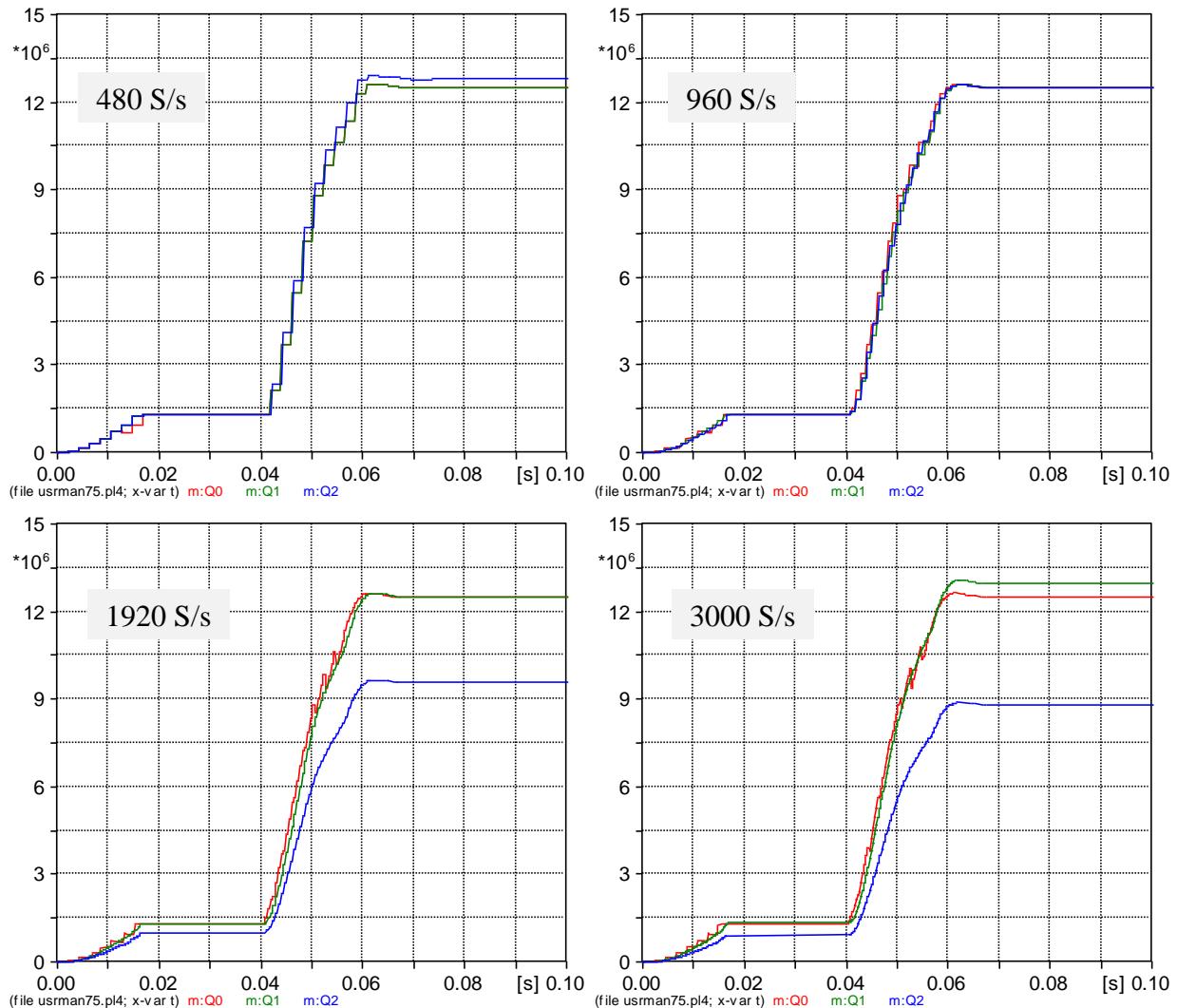


Fig. 7.23 Calculated reactive power. Q0 FFT (red), Q1 DFT (green), Q2 FFT-old (blue).  
Theoretical 1.307  $\Rightarrow$  12.5 MVar.  $\Delta t = 1\text{e-}4$ .

## 7.6 LCC Internal Calculation

ATPDraw 7.4 included Internal Calculation in the LCC object. This will enable extensive calculations behind the scene. Key units introduced are:

- Complex.pas: Manage complex numbers as records with operator overloading.
- Bessel.pas: Routines for various Bessel function evaluations.
- Eigen.pas: Eigenvalue and eigenvector calculations of non-symmetric matrices.
- VectorMatrix.pas: Large library of matrix and vector manipulation routines. LU and QR decompositions, square root and exponent of matrix, various multiplications and vector products. Matrix and Vector implemented as records with operator overloading.
- VectorFit.pas: VectFit3 implemented with management of JMarti and ULM models.

On top of this utility routines come the unit that calculate the parameters for overhead lines and cables. This follows closely [25-27]

### 7.6.1 Overhead line modelling

The complex image theory is used based on the complex penetration depth illustrated in Fig. 7.24 and defined as

$$p = \frac{1}{\sqrt{j\omega\mu_e \cdot (\sigma_e + j\omega\varepsilon_e)}} \quad (32)$$

The admittance is defined with  $p=0$  and potential coefficients

$$P_{ij} = \frac{1}{2\pi\varepsilon_0} \cdot \ln\left(\frac{D_{ij}}{d}\right) \text{ with} \quad (33)$$

$$d = \begin{cases} r_i, i = j \\ d_{ij} = \sqrt{(y_i - y_j)^2 + (x_i - x_j)^2}, i \neq j \end{cases} \quad \text{and} \quad D_{ij} = \sqrt{(y_i + y_j)^2 + (x_i - x_j)^2} \quad (34)$$

The series impedance is defined as  $z = \text{diag}(z_i) + z_e$  with

$$z_{e,ij} = \frac{j\omega\mu_0}{2\pi} \cdot \ln\left(\frac{D'_{ij}}{d}\right) = \frac{j\omega\mu_0}{2\pi} \cdot \ln\left(\frac{\sqrt{(y_i + y_j + 2p)^2 + (x_i - x_j)^2}}{d}\right) \text{ and} \quad (35)$$

$$z_i = R_{DC} \cdot \frac{\gamma r}{2} \cdot \begin{cases} \frac{I_0(\gamma r)}{I_1(\gamma r)}, \text{ solid conductor} \\ \frac{I_0(\gamma r) \cdot K_1(\gamma r_i) + I_1(\gamma r_i) \cdot K_0(\gamma r)}{I_1(\gamma r) \cdot K_1(\gamma r_i) - I_1(\gamma r_i) \cdot K_1(\gamma r)}, \text{ tubular conductor} \end{cases} \quad (36)$$

where  $r$  is the outer radius,  $r_i$  is the inner radius and  $\gamma = \sqrt{j\omega\mu_0(\sigma + j\omega\varepsilon_0)}$  with  $\sigma$  as the conductor conductivity. The Bessel functions in (36) must be series expanded for small and asymptotic expanded for large arguments.

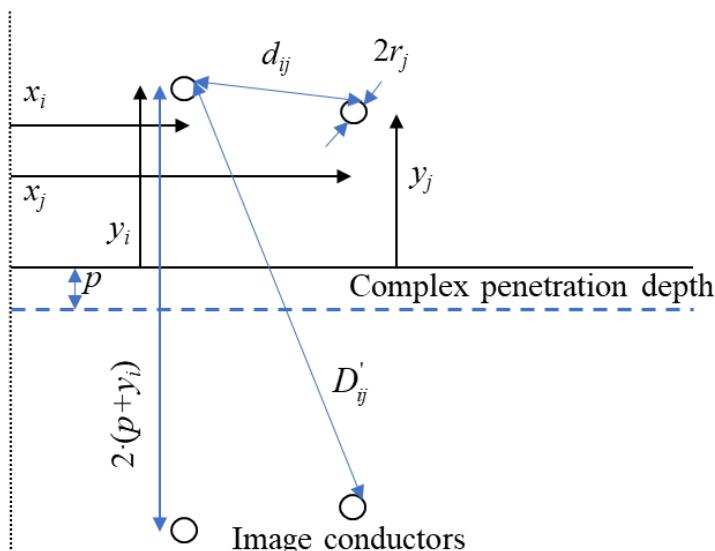


Fig. 7.24. Geometry of conductors above a complex image

### 7.6.2 Underground cable modelling

The formulation according to Ametani [25] is used for both single core cables and enclosing pipe cables. The cable impedances for individual cables are first formulated in loop quantities [26]. A loop current enters conductor  $i$  and leaves in next layer conductor  $i+1$ , as shown in Fig. 7.25. Loop voltages are measured between the conductors.

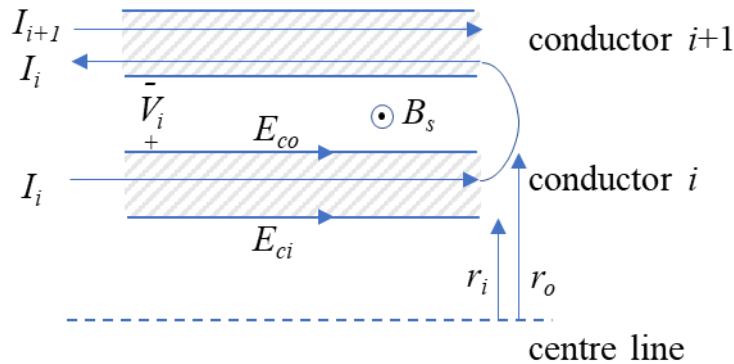


Fig. 7.25 Geometry of coaxial cables with loop quantities

The electric field in tubular geometries directed along the cable (z-direction) with a radial variation (r-direction) is generally written as

$$E_z(r) = A_1 \cdot I_0(\gamma r) + A_2 \cdot K_0(\gamma r) \quad (37)$$

With  $I_0$  and  $K_0$  being modified Bessel functions of first and second kind respectively of order zero with propagation coefficient  $\gamma = \sqrt{j\omega\mu_c(\sigma_c + j\omega\epsilon_c)} \approx \sqrt{j\omega\mu_c\sigma_c}$ .

With loop quantities of currents there exists simple boundary conditions on the form

$$\left. \frac{\partial E}{\partial r} \right|_{r=r_i \vee r_o} = 0 \text{ on inner or outer surfaces and the integral of the current density in the conductor is}$$

given by the known loop current  $\int_{r_i}^{r_o} 2\pi r \cdot \sigma \cdot E_z(r) \cdot dr = I_i$ . This gives constants  $A_1$  and  $A_2$  in (37).

Based on this, three different impedances (field on conductor surface over current) can be defined for a conductor's inner and outer surfaces and for the mutual between conductors.

$$Z_{ci} = \frac{j\omega\mu_c}{2\pi z_1} \frac{I_0(z_1)K_1(z_2) + K_0(z_1)I_1(z_2)}{I_1(z_2)K_1(z_1) - K_1(z_2)I_1(z_1)} \quad (38)$$

$$Z_{co} = \frac{j\omega\mu_c}{2\pi z_1} \frac{I_0(z_2)K_1(z_1) + K_0(z_2)I_1(z_1)}{I_1(z_2)K_1(z_1) - K_1(z_2)I_1(z_1)} \quad (39)$$

$$Z_{cm} = \frac{j\omega\mu_c}{2\pi z_1 z_2} \frac{1}{I_1(z_2)K_1(z_1) - K_1(z_2)I_1(z_1)} \quad (40)$$

with  $z_1 = r_i \cdot \sqrt{j\omega\mu_c\sigma_c}$  and  $z_2 = r_o \cdot \sqrt{j\omega\mu_c\sigma_c}$  with  $\mu_c$  and  $\sigma_c$  being the permittivity and conductivity and  $r_i$  and  $r_o$  being its inner and outer radius of the tubular conductor.

In addition, there are contributions from the magnetic field between conductors  $Z_s$  and the outside ground return  $Z_g$ . The ground return contribution

$$Z_{s,i} = \frac{j\omega\mu_0}{2\pi} \cdot \ln \frac{r_{i,i+1}}{r_{o,i}} \quad (41)$$

The loop-impedance matrix can now be established based on Fig. 7.25 and (38-41) starting from the inner conductor  $i=1$  (core) to the outer ( $i=n$ ). The loop-impedance matrix is a tridiagonal matrix (diagonal, and sub/super diagonals) defined as

$$\text{Diagonal elements: } Z_{Lii} = Z_{co,i} + Z_{ci,i+1} + Z_{s,i} \quad (42)$$

$$\text{Sub/super-diagonal: } Z_{L,i-1,i} = Z_{L,i,i-1} = -Z_{cm,i} \quad (43)$$

For the last layer the impedance  $Z_{c,n,n+1}$  can be set to the ground return impedance, or set to zero with handling of ground-return added later as a separate task.

Each cable ( $k=1..K$ ) is modelled separately with its loop-impedance and then converted to phase quantities by  $Z_{kk} = U \cdot Z_{Lk} \cdot U^T$  where  $U$  is an upper transformation matrix containing ones on and above the diagonal and zeros below.

The submatrices containing phase quantities are then stacked into a total series-impedance matrix:

$$Z = \begin{bmatrix} Z_{11} & 0 & \cdots & 0 \\ 0 & Z_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Z_{KK} \end{bmatrix} \quad (44)$$

The ground return contributions  $Z_{g,kl}$  given in (45) with simplification in (46), are now added to every element inside the blocks  $(k, l)$  in  $Z$ .

$$Z_{g,kl} = \frac{j\omega\mu_0}{2\pi} \left[ K_0(\gamma_g D_k) - K_0(\gamma_g D_l) + \int_0^\infty \frac{2e^{-H\sqrt{\lambda^2 + \gamma_g^2}}}{\lambda + \sqrt{\lambda^2 + \gamma_g^2}} \cos(x\lambda) \cdot d\lambda \right] \quad (45)$$

$$\text{with } \gamma_g = \sqrt{j\omega\mu_g(\sigma_g + j\omega\varepsilon_g)}$$

Several approximations are given for the ground return in (45). ATP uses a Carson formula series expansion whereas ATPDraw 7.4 uses what is given in (46):

$$Z_{g,kl} = \frac{j\omega\mu_0}{2\pi} \cdot \begin{cases} K_0\left(\gamma_g \sqrt{(x_k - x_l)^2 + (y_k - y_l)^2}\right) + \frac{2}{4 + ((x_k - x_l) \cdot \gamma_g)^2} e^{-(y_k + y_l) \cdot \gamma_g}, & k \neq l \\ K_0(\gamma_g r_k) + \frac{2}{4 + (r_k \cdot \gamma_g)^2} e^{-2y_k \cdot \gamma_g}, & k = l \end{cases} \quad (46)$$

where  $(x, y)$  are the coordinates of the cable centre with  $y$  positive into the ground.

This will now result in a matrix sorted according to the individual cable conductors. However, it is more practical to sort it with cores first, then sheaths followed by armour. This will make it more logical to connect it to the outside circuit and also to eliminate grounded conductors. In ATPDraw this accomplished by stacking the loop matrices in (44) and applying a large transformation matrix that both converts to phase quantities and sort the conductors.

The admittance can also be formulated with loop quantities, but since the admittance is much simpler this is not done in ATPDraw 7.4.

### 7.6.3 Transmission line models

ATPDraw 7.4 implements Vector Fitting [29] based on the publicly available VectFit3 routine. For ULM fitting [30] this is extended to enable inclusion of time delay factors in the residue identification. For Internal Calculation, Vector Fitting is also used for JMarti modeling according to [31].

The frequency domain transmission line model for a line between terminals A and B, established from the telegraph equations, is

$$(Y_C \cdot V - I)_A = H \cdot (Y_C \cdot V + I)_B \quad (47)$$

Where  $Y_C = z^{-1} \sqrt{zy} = \sqrt{(yz)^{-1}} \cdot y$  is the characteristic admittance

and  $H = \exp(-\sqrt{yz} \cdot l)$  is the propagation matrix

The series impedance  $z$  and shunt admittance  $y$  matrices are calculated as illustrated in Chaps. 7.6.1 and 7.6.2.

If the elements of the  $Y_C$  and  $H$  matrices are fitted with rational functions ( $c/(s-a)$ ), their inverse Fourier transform is analytical exponential functions. This results in recursive convolution integrals and practical implementation as state space models in the time domain.

For JMarti lines the modes of  $Y_C$  (actually the inverse  $Z_C$ ) and  $H$  are fitted with rational functions and combined with a constant transformation matrix. The extraction of the modes and transformation matrix comes from eigenvalue analysis of the complex, unsymmetrical matrix product  $y \cdot z$ ;  $[T, \lambda] = eig(y \cdot z)$ . The transformation matrix  $T$  is the eigenvectors and the modal quantities to be fitted defines as

$$\begin{aligned} Z_C^m &= diag(\sqrt{\lambda}) / diag(T \cdot y \cdot T^T) \\ H_0^m &= diag(\exp(-\sqrt{\lambda} \cdot l)) \cdot \exp(+s \cdot \tau_m) \end{aligned} \quad (50)$$

where  $\tau_m$  is the time delay extracted to make  $H_0^m$  a smooth function. The calculation of the time delay is according to [33] using the imaginary part of the eigenvalue at high frequency correction by minimum phase shift method and without optimization.

The modes in (50) are now easily fitted with Vector Fitting and passed on to ATP for simulation.

$$\begin{aligned} Z_C^m &\approx \sum_{n=1}^{NZ} \frac{c_{m,n}}{s - a_{m,n}} + d_m \\ H_0^m &\approx \sum_{n=1}^{NH} \frac{c_{m,n}}{s - a_{m,n}} \end{aligned} \quad (51)$$

Challenges in the calculation of the modes are eigenvalue switchovers and non-orthogonal eigenvectors when eigenvalues become identical. Measures to reduce the problem is implemented according to [26] and analytical eigenvalues are calculated for perfectly symmetrical cases. Still there are problems in near symmetrical cases. This is also the case for the original ATP JMarti model.

For ULM the characteristic admittance matrix is fitted directly in the phase domain with a common set of real and complex conjugated poles. For speed, ATPDraw uses the trace of the matrix for pole identification and then calculates the residues ( $c_{ij}$ ) and constant terms ( $d_{ij}$ ) for each element by stacking the upper half of the symmetrical  $Y_C$  matrix in the call to Vector Fitting.

$$Y_{C,ij} = \sum_{n=1}^{N_Y} \frac{c_{ij,n}}{s - a_n} + d_{ij} \quad (52)$$

To avoid issues with eigenvector degeneration and increase the robustness for near-symmetrical cases, the modes of the propagation matrix are approximated by calculating eigenvectors at a single high *Freq. veloc.* ( $f_v \approx 1$  MHz) as proposed in [30]:

$$H^m \approx T_{f_v}^{-1} \cdot H \cdot T_{f_v} \quad (53)$$

This gives slightly inaccurate modes at low frequencies, normally compensated by the large number of poles at this frequency range. Moreover, modes are also combined in groups according to [30] based on the deviation in angles at the highest frequency point.

The mode approximation and grouping can result in unexpected problems for simple overhead line cases if no precautions are taken. The number of poles and possible calculation points should be increased in such cases and possibly the merging criterion *epsDeg* reduced from the default 10 deg. This will improve the low frequency accuracy and avoid stability issues.

Now, all elements of the propagation matrix  $H$  are written as a linear combination of each mode:

$$H_{ij} \approx \sum_{m=1}^M \sum_{n=1}^N \frac{c_{ij,m,n}}{s - a_{m,n}} \cdot e^{-s\tau_m} \quad (54)$$

The time delay factors ( $\exp(-s\tau_m)$ ) are included in the Vector Fitting residue identification process as weighting factors.

The poles, residues, constant terms, and time delays in (52, 54) are passed in a text file to the ULM Foreign Model for the time loop implementation.

## 7.7 Internal Solver

ATPDraw version 7.5 introduced an internal solver for direct solution of transients with steady-state initialization. At this stage this is just for demonstration purposes. The following key elements are enabled in the core-part of the solver:

- Steady-state phasor solution for initialization of all components. Frequency Scan.
- Trapezoidal rule of integration used for first order differential equation approximation.
- Interpolation used for all state-change events (switching, nonlinear segments change). Half timestep scatter removal after switching events.
- Dynamic memory management for all matrices and vectors. No size restrictions.
- Sparse matrix storage. Partial LU-factorization for system changes. Scales well with size.
- Multiple-runs in thread-safe environment.
- Object oriented data structure that enables easier extensions and modularization. Slows down performance somewhat.

The solver code is very light-weighted. The main unit that defines the TSolverThread with frequency domain and time domain solutions, the TSolverController that manages multiple runs and the ancestor component class TSolverComp that initializes the data structure consists of only 2000 program lines. The 70 components supported in v7.5 are defined in 6000 code lines.

### 7.7.1 System description

The circuit is modelled in nodal form where each row of the system description consists of the current balance of one node. Both sources and switches are modeled with the augmentation technique. This enables an optional series resistance of those components. An open switch is modeled by changing the row of the switch to all zeros with unity on the diagonal. The system matrix is modeled as a sparse matrix based on linked lists, and with a topological structure shown in Fig. 7.26. The trapezoidal rule of integration is used for differential equation approximation and equivalent circuits according to [34] are established as the basis for the modeling.

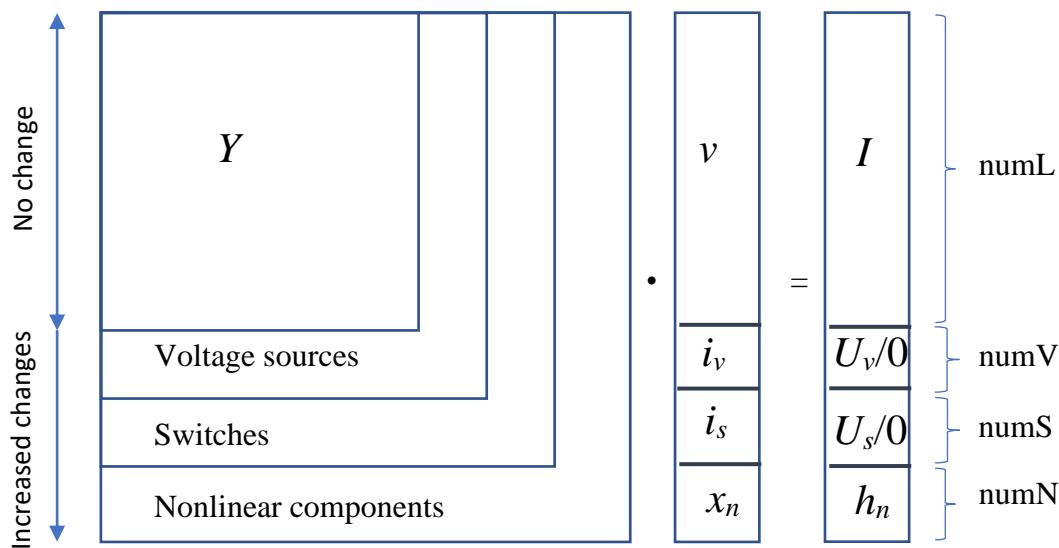


Fig. 7.26 Topology of the system description in the internal solver in v7.5

### 7.7.2 Interpolation

At each time step, switched components (switches and nonlinearities) are called to report exact switching instance within the remainder of the timestep. The components with the first switching requests are then allowed to change status at the requested time. All states of the system (solution vector, history sources, local state-space) are then interpolated to this time instance as shown in Fig. 7.27. Every component has a *DoInterpolate* procedure that does the interpolation [35] of their internal states based on previous state  $px$ , present state  $x$ , and the interval  $dh$  reported by the *CheckEvent* procedure. The time is then set to  $t-dh$  and a new solution is found. For this new solution, the  $dh$  time interval is available for new switching events. When no more switching events happens the interpolation back on the time step grid is performed. This point is then made available for plotting. Scatter removal is performed one time step afterwards with half time step back and forward interpolation.

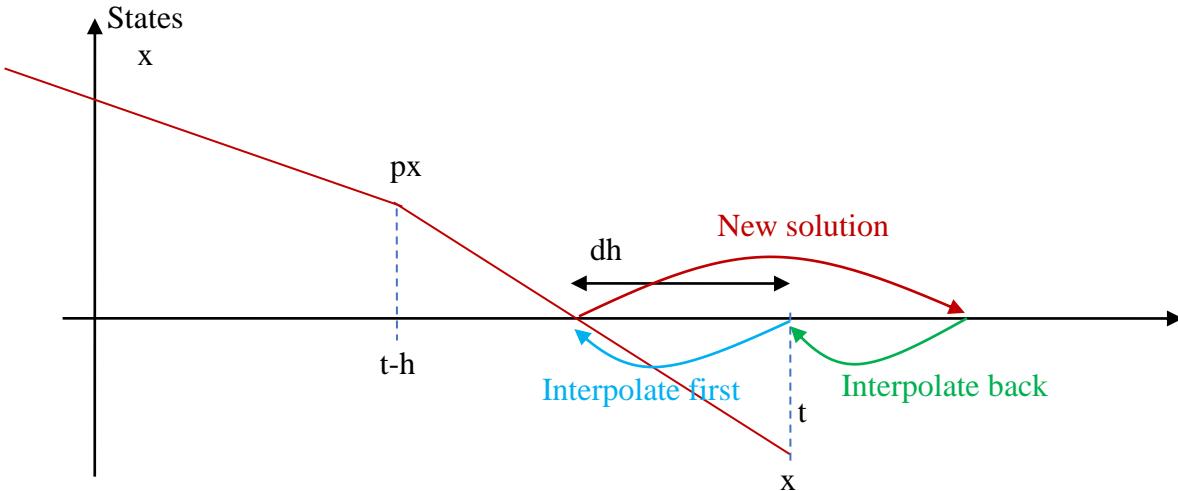


Fig. 7.27 Interpolation scheme in internal solver in v7.5 according to [35].

### 7.7.3 Object oriented structure

A separate, simplified data structure is used in the solver. It contains only active components and relevant, parsed data. Before the simulation starts all nodes are named. Then data are copied to the new object-oriented structure created in a separate thread. All nodes are registered and sorted with node names used most taken first in node array.

The following procedures are called for each component

- PreCalc. Before the calculation starts. Given timestep. Do time consuming calculations.
- GetFreqDomain. In the steady-state initialization or in Frequency Scan. Given frequency. Return the complex system description.
- Initialize. Called after the steady-state initialization. Complex phasor solution given. Return the history sources and initial states.
- GetTimeDomain. In the timestep loop for most components. Updates history sources and the system matrix (at least first time). Here, all trapezoidal rule based equivalent circuits are updated, as well as state-space formulations.
- CheckState. In the timestep loop for selected components (switches, non-linearities). Check if switching events or segment change could occur in timestep.
- SwapState. In the timestep loop for selected components. Perform the actual change in states that happens first in time (switches, nonlinear etc.)
- DoInterpolate. In the timestep loop. Perform interpolation for all components. Default updates the standard history sources. Some components implement their own states.
- GetPlot. In the timestep loop for selected components.

Due to interpolation, the GetTimeDomain routine might be called for instances outside the timestep grid. The consequence is that GetPlot and CheckState must be called separately.

### 7.8 XML data exchange

The open XML project-file format gives an opportunity to manually modify project content and share projects with other software. ATPDraw v7.0 supports limited features that covers standard components, connections and texts. The XML format is very flexible, and since the context is embedded, it can be dynamically developed over time with less concern about backward

compatibility. The XML format is expected to be extended more in the future and what is presented here in this manual is just a start. The XML-format does not follow a SIM-standard and the data and physical layout are not separated. The coordinates of components and nodes is the trickiest thing about data exchange. The coordinate system and model are explained next.

### 7.8.1 ATPDraw coordinate system

The coordinate system (world coordinates) has origin in the upper left corner and a resolution equal to screen pixels when the zoom factor is 100%. All positions have a resolution of 10 pixels. The x-axis is directed from left to right and the y-axis from top to bottom. All components have a positive position in this coordinate system, with default around (5000, 5000). Furthermore, every component can be rotated around this position with an Angle of 90, 180 or 270 deg. measured counter-clockwise with zero as default. The nodes have a position given relative to the component's position. An exception is the LINE3 component with Left and Right nodes given positions in absolute world coordinates. Connections have similarly node positions in world coordinates. Position of component nodes (except for LINE3) is optional in the XML-file as this information for standard components is given in ATPDraw.scl. When a component is rotated or flipped the relative node position remains unchanged, but the actual node position is recalculated internally. The component's icon can be of type vector or bitmap. The XML-file format offers also to set Icon=default and in this case the icon is obtained from ATPDraw.scl or reconstructed. The icon of type bitmap is centered around the component position ( $\pm 20, \pm 20$ ) containing 41·41 bytes of color information. The icon of type vector consists of shapes and texts in coordinates relative to the component position.

Fig. 7.28 shows a case with a component located at Comp.Pos=(4900,5020) having a node with relative position Node.RelPos(20,-10) (which is the default position 9). The node position without rotation will be Node.Pos=(4920,5010). This position must match what is given in overlapping nodes or connections in order to connect the circuit properly. If the component is rotated  $n \cdot 90$  deg., the node position becomes Node.Pos=Comp.Pos+Node.RelPos· $R^n$  with  $R = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

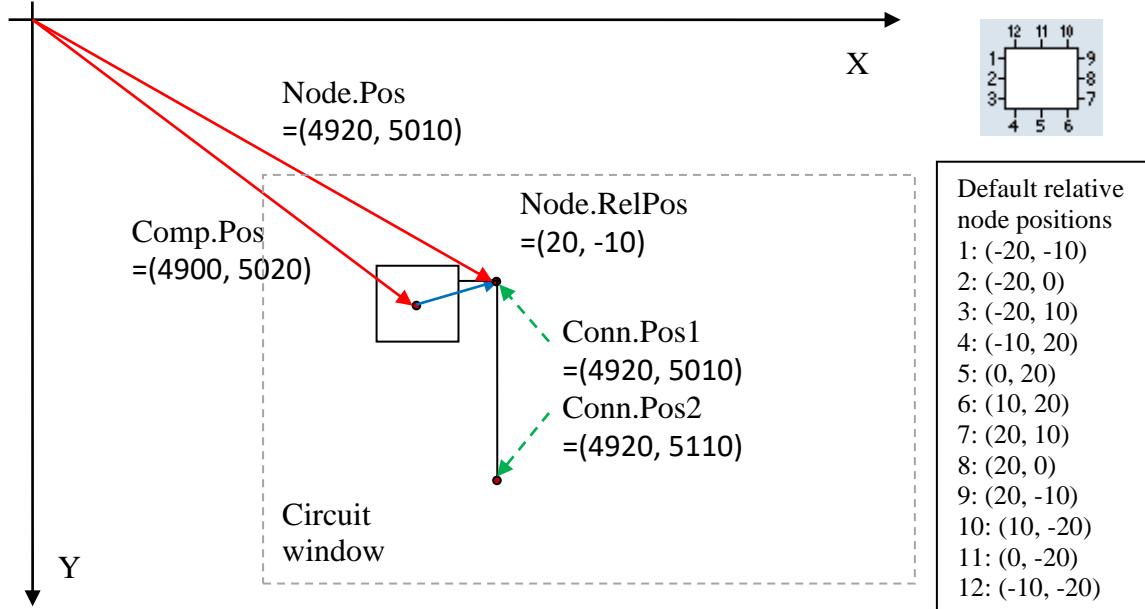


Fig. 7.28 Coordinate system in ATPDraw.

### 7.8.2 The XML format definition; DTD-file

The DTD-file describes the possible xml elements (tags) and their structure and properties. The names of element and attributes are case sensitive. For definitions of syntax consult for instance [https://en.wikipedia.org/wiki/Document\\_type\\_definition](https://en.wikipedia.org/wiki/Document_type_definition)

```

<!ELEMENT project (header, objects, variables)>
<!ATTLIST project Application CDATA #REQUIRED>
<!ATTLIST project Version CDATA #REQUIRED>
<!ATTLIST project VersionXML CDATA #REQUIRED>
<!ATTLIST header Timestep CDATA #REQUIRED>
<!ATTLIST header Tmax CDATA #REQUIRED>
<!ATTLIST header XOPT CDATA #REQUIRED>
<!ATTLIST header COPT CDATA #REQUIRED>
<!ATTLIST header TopLeftX CDATA #REQUIRED>
<!ATTLIST header TopLeftY CDATA #REQUIRED>
<!ELEMENT objects (comp*,conn*,text*)>           <!--Supported objects. -->
<!ATTLIST comp Name CDATA #REQUIRED>
<!ATTLIST comp Id CDATA #IMPLIED>
<!ATTLIST comp Hidden (true|false) #IMPLIED>
<!ATTLIST comp Caption CDATA #IMPLIED>
<!ATTLIST comp CapAngle (0| 90| 180 | 270) #IMPLIED>
<!ATTLIST comp CapPosX CDATA #IMPLIED>
<!ATTLIST comp CapPosY CDATA #IMPLIED>
<!--Important definition of various component extensions. objects used for groups-->
<!ELEMENT comp (comp_content,(Model|probe|BCTRAN|XFMR|LCC|SM|USP|BUS3|LINE3|objects))>
<!ATTLIST comp_content Protected (true|false) #IMPLIED>
<!ATTLIST comp_content PosX CDATA #REQUIRED>
<!ATTLIST comp_content PosY CDATA #REQUIRED>
<!ATTLIST comp_content NumPhases CDATA #IMPLIED>
<!ATTLIST comp_content SinglePhaseIcon (true|false) #IMPLIED>
<!ATTLIST comp_content Icon (default|vector|bitmap) #IMPLIED> <!--default:no data-->
<!ATTLIST comp_content ScaleIconX CDATA #IMPLIED> <!--1 default-->
<!ATTLIST comp_content ScaleIconY CDATA #IMPLIED>
<!ATTLIST comp_content Order CDATA #IMPLIED>
<!ATTLIST comp_content Angle (0| 90| 180 | 270) #IMPLIED>
<!ATTLIST comp_content FlitLR (true|false) #IMPLIED>
<!ATTLIST comp_content Output CDATA #IMPLIED>
<!ATTLIST comp_content HighRes (true|false) #IMPLIED>
<!ELEMENT comp_content (node*,data*,icon?, comment?,helpstring?,datastring?,nonlin?)>
<!ATTLIST node Name CDATA #IMPLIED>
<!ATTLIST node Value CDATA #IMPLIED>
<!ATTLIST node PosX CDATA #IMPLIED> <!--not needed, stored in atpdraw.scl -->
<!ATTLIST node PosY CDATA #IMPLIED>
<!ATTLIST node NamePosX CDATA #IMPLIED>
<!ATTLIST node NamePosY CDATA #IMPLIED>
<!ATTLIST node NumPhases CDATA #IMPLIED>
<!ATTLIST node Ground (0|1|2|3|4|) #IMPLIED>
<!ATTLIST node Kind CDATA #IMPLIED>
<!ATTLIST node Internal (true|false) #IMPLIED>
<!ATTLIST node Disabled (true|false) #IMPLIED>
<!--0=not inherited, otherwise = parent's node -->
<!ATTLIST node Inherited CDATA #IMPLIED>
<!ATTLIST data Name CDATA #IMPLIED>
<!ATTLIST data Value CDATA #IMPLIED>
<!--0=not inherited, otherwise = parent's data -->
<!ATTLIST data Inherited CDATA #IMPLIED>
<!ELEMENT comment (#PCDATA)>
<!ELEMENT helpstring (#PCDATA)>
<!ELEMENT datastring (#PCDATA)>
<!ATTLIST nonlin Kind CDATA #IMPLIED>
<!ELEMENT nonlin (point*)>
<!ATTLIST point X CDATA #REQUIRED>
<!ATTLIST point Y CDATA #REQUIRED>
<!ATTLIST icon BRectTop CDATA #IMPLIED>
<!ATTLIST icon BRectLeft CDATA #IMPLIED>
<!ATTLIST icon BRectBottom CDATA #IMPLIED>
```

```

<!ATTLIST icon BRectRight CDATA #IMPLIED>
<!ATTLIST icon ExtPX CDATA #IMPLIED>
<!ATTLIST icon ExtPY CDATA #IMPLIED>
<!ELEMENT icon (#PCDATA|icon_shape|icon_text)*> <!--#PCDATA holds Bitmaps as hex-->
<!ATTLIST icon_shape Kind CDATA #REQUIRED> <!--Vector in icon_shape and icon_text-->
<!ATTLIST icon_shape Tag CDATA #REQUIRED>
<!ATTLIST icon_shape Visible (true|false) #REQUIRED>
<!ATTLIST icon_shape PenColor CDATA #REQUIRED>
<!ATTLIST icon_shape PenWidth CDATA #REQUIRED>
<!ATTLIST icon_shape PenStyle CDATA #REQUIRED>
<!ATTLIST icon_shape BrushColor CDATA #REQUIRED>
<!ATTLIST icon_shape BrushStyle CDATA #REQUIRED>
<!ELEMENT icon_shape (point*)> <!-- relative position to comp, integer-->
<!ATTLIST icon_text Tag CDATA #REQUIRED>
<!ATTLIST icon_text Visible (true|false) #REQUIRED>
<!ATTLIST icon_text PosX (true|false) #REQUIRED>
<!ATTLIST icon_text PosY (true|false) #REQUIRED>
<!ATTLIST icon_text FontIdx CDATA #REQUIRED>
<!ATTLIST icon_text FontSize CDATA #REQUIRED>
<!ATTLIST icon_text FontColor CDATA #REQUIRED> <!--Bold+2*Italic+4*Underline-->
<!ATTLIST icon_text FontAttrib CDATA #REQUIRED>
<!ATTLIST icon_text Rotate (true|false) CDATA #IMPLIED>
<!ATTLIST icon_text Angle CDATA #REQUIRED>
<!ELEMENT icon_text (#PCDATA)> <!--#PCDATA holds the actual text string-->
<!ATTLIST Model UseAs CDATA #IMPLIED>
<!ELEMENT Model (#PCDATA)> <!--contains RecordStr. -->
<!ATTLIST probe CaptureSteadyState (true|false) #IMPLIED>
<!ATTLIST probe OnScreen (0|1|2) #IMPLIED>
<!ATTLIST probe ScreenFormat (0|1|2|3|4) #IMPLIED>
<!ATTLIST probe ScreenShow (0|1|2|3|4|5) #IMPLIED>
<!ATTLIST probe CurrNode (true|false) #IMPLIED>
<!ATTLIST probe FontSize CDATA #IMPLIED>
<!ATTLIST probe Precision CDATA #IMPLIED>
<!ATTLIST probe TimeOnScreen (true|false) #IMPLIED>
<!ELEMENT probe (monitor*)>
<!ATTLIST monitor Phase CDATA #IMPLIED>
<!ATTLIST BCTRAN NumPhases (1|3) #IMPLIED>
<!ATTLIST BCTRAN NumWindings (2|3) #IMPLIED>
<!ATTLIST BCTRAN Freq CDATA #IMPLIED>
<!ATTLIST BCTRAN OutputAR (true|false) #IMPLIED>
<!ATTLIST BCTRAN AutoAdd (true|false) #IMPLIED>
<!ATTLIST BCTRAN ExtNeutral (true|false) #IMPLIED>
<!ATTLIST BCTRAN AutoByATP (true|false) #IMPLIED>
<!ELEMENT BCTRAN (winding+,core,short1+,short0*,open1*,open0*)>
<!ATTLIST winding kV CDATA #IMPLIED>
<!ATTLIST winding MVA CDATA #IMPLIED>
<!ATTLIST winding Coupl CDATA #IMPLIED>
<!ATTLIST winding Shift CDATA #IMPLIED>
<!ATTLIST core Type CDATA #IMPLIED>
<!ATTLIST core CoreOutside CDATA #IMPLIED>
<!ATTLIST core ExcitationAt CDATA #IMPLIED>
<!ATTLIST core ConnectAt CDATA #IMPLIED>
<!ATTLIST short1 Zpc CDATA #REQUIRED>
<!ATTLIST short1 MVA CDATA #REQUIRED>
<!ATTLIST short1 kW CDATA #REQUIRED>
<!ATTLIST short0 Zpc CDATA #REQUIRED>
<!ATTLIST short0 MVA CDATA #REQUIRED>
<!ATTLIST short0 kW CDATA #REQUIRED>
<!ATTLIST open1 Vpc CDATA #REQUIRED>
<!ATTLIST open1 Ipc CDATA #REQUIRED>
<!ATTLIST open1 kW CDATA #REQUIRED>
<!ATTLIST open0 Vpc CDATA #REQUIRED>
<!ATTLIST open0 Ipc CDATA #REQUIRED>
<!ATTLIST XFMR NumPhases (1|3) #REQUIRED>
<!ATTLIST XFMR NumWindings (1|2|3|4) #REQUIRED>
<!ATTLIST XFMR ExcitationAt (1|2|3|4) #IMPLIED>
<!ATTLIST XFMR ExternalNeutral (true|false) #IMPLIED>
<!ATTLIST XFMR HideCore (true|false) #IMPLIED>

```

```

<!ATTLIST XFMR LBasedOn (0|1|2|3) #IMPLIED> <!--0:no, 1:design, 2:test, 3: typ -->
<!ATTLIST XFMR RBasedOn (0|1|2|3) #IMPLIED>
<!ATTLIST XFMR CBasedOn (0|1|2|3) #IMPLIED>
<!ATTLIST XFMR CoreBasedOn (0|1|2|3) #IMPLIED>
<!ELEMENT XFMR (winding+,core,short1+,short0*,open1*,open0*)>
<!ATTLIST core Linf CDATA #IMPLIED>
<!ATTLIST core Lzero CDATA #IMPLIED>
<!ATTLIST core RelYokeArea CDATA #REQUIRED>
<!ATTLIST core RelYokeLength CDATA #REQUIRED>
<!ATTLIST core RelOuterLegArea CDATA #IMPLIED>
<!ATTLIST core RelOuterLegLength CDATA #IMPLIED>
<!ATTLIST core RelLimbArea CDATA #IMPLIED>
<!ATTLIST core RelLimbLength CDATA #IMPLIED>
<!ATTLIST LCC Template (true|false) #IMPLIED>
<!ATTLIST LCC Embed (true|false) #IMPLIED>
<!ATTLIST LCC InternalLCC (true|false) #IMPLIED>
<!ATTLIST LCC NumPhases CDATA #REQUIRED>
<!ATTLIST LCC LineCablePipe (1|2|3|4) #REQUIRED>
<!ATTLIST LCC ModelType (0|1|2|3|4) #REQUIRED>
<!ATTLIST LCC IconLength (true|false) #IMPLIED>
<!ATTLIST LCC FreqMatrix CDATA #IMPLIED>
<!ATTLIST LCC FreqSS CDATA #IMPLIED>
<!ATTLIST LCC IPNT CDATA #IMPLIED>
<!ATTLIST LCC IDEC CDATA #IMPLIED>
<!ATTLIST LCC PolesY CDATA #IMPLIED>
<!ATTLIST LCC ModesH CDATA #IMPLIED>
<!ATTLIST LCC PolesH CDATA #IMPLIED>
<!ELEMENT LCC (line_header?,cable_header?,pipe?)>
<!ATTLIST cable_header InAirGrnd (1|0|1) #REQUIRED>
<!ATTLIST cable_header CableConstant (true|false) #IMPLIED>
<!ATTLIST cable_header Snaking (true|false) #IMPLIED>
<!ATTLIST cable_header MatrixOutput (true|false) #IMPLIED>
<!ATTLIST cable_header ExtraCG (true|false) #IMPLIED>
<!ELEMENT cable_header (cable*)>
<!ATTLIST cable NumCond CDATA #REQUIRED>
<!ATTLIST cable Rout CDATA #REQUIRED>
<!ATTLIST cable PosX CDATA #REQUIRED>
<!ATTLIST cable PosY CDATA #REQUIRED>
<!ELEMENT cable (conductor*)>
<!ATTLIST conductor Rin CDATA #REQUIRED>
<!ATTLIST conductor Rout CDATA #REQUIRED>
<!ATTLIST conductor rho CDATA #REQUIRED>
<!ATTLIST conductor muc CDATA #REQUIRED>
<!ATTLIST conductor mui CDATA #REQUIRED>
<!ATTLIST conductor epsI CDATA #REQUIRED>
<!ATTLIST conductor semicon1 CDATA #IMPLIED>
<!ATTLIST conductor semicon2 CDATA #IMPLIED>
<!ATTLIST conductor Cext CDATA #IMPLIED>
<!ATTLIST conductor Gext CDATA #IMPLIED>
<!ATTLIST conductor Grounded (true|false) #IMPLIED>
<!ATTLIST line_header RealMtrx (true|false) #IMPLIED>
<!ATTLIST line_header SkinEffect (true|false) #IMPLIED>
<!ATTLIST line_header AutoBundle (true|false) #IMPLIED>
<!ATTLIST line_header MetricUnit (true|false) #IMPLIED>
<!ELEMENT line_header (line*)>
<!ATTLIST line PhNo CDATA #REQUIRED>
<!ATTLIST line Rin CDATA #REQUIRED>
<!ATTLIST line Rout CDATA #REQUIRED>
<!ATTLIST line React CDATA #REQUIRED>
<!ATTLIST line Horiz CDATA #REQUIRED>
<!ATTLIST line Vtow CDATA #REQUIRED>
<!ATTLIST line Vmid CDATA #REQUIRED>
<!ATTLIST line NB CDATA #REQUIRED>
<!ATTLIST line Separ CDATA #REQUIRED>
<!ATTLIST line Alpha CDATA #REQUIRED>
<!ATTLIST pipe Infinite (true|false) #IMPLIED>
<!ATTLIST pipe RP1 CDATA #REQUIRED>
<!ATTLIST pipe RP2 CDATA #REQUIRED>

```

```

<!ATTLIST pipe RP3 CDATA #REQUIRED>
<!ATTLIST pipe rho CDATA #REQUIRED>
<!ATTLIST pipe mu CDATA #REQUIRED>
<!ATTLIST pipe eps1 CDATA #REQUIRED>
<!ATTLIST pipe eps2 CDATA # REQUIRED >
<!ATTLIST pipe C_P CDATA # IMPLIED >
<!ATTLIST pipe G_P CDATA # IMPLIED >
<!ATTLIST SM NumMasses CDATA #REQUIRED>
<!ELEMENT SM (mass*)>
<!ATTLIST mass Angle (true|false) #IMPLIED>
<!ATTLIST mass Speed (true|false) #IMPLIED>
<!ATTLIST mass Torque (true|false) #IMPLIED>
<!ATTLIST USP UspParams (true|false) #REQUIRED>
<!ATTLIST USP Usp3ph5s (true|false) #IMPLIED> <!--The include file is in datastring -->
<!ATTLIST BUS3 PQOnScreen (true|false) #IMPLIED>
<!ATTLIST BUS3 PQUser (true|false) #IMPLIED>
<!ATTLIST BUS3 PSColor (true|false) #IMPLIED>
<!ATTLIST BUS3 PQPosX CDATA #IMPLIED>
<!ATTLIST BUS3 PQPosY CDATA #IMPLIED>
<!ATTLIST BUS3 PQFontSize CDATA #IMPLIED>
<!ELEMENT BUS3 (#PCDATA)>                                <!--Stores the text on screen -->
<!ATTLIST LINE3 ScaleI CDATA #IMPLIED>
<!ATTLIST LINE3 ScaleS CDATA #IMPLIED>
<!ATTLIST LINE3 Fontsize CDATA #IMPLIED>
<!ATTLIST LINE3 Precision CDATA #IMPLIED>
<!ELEMENT LINE3 (Left,Right)>
<!ATTLIST Left Flag CDATA #IMPLIED>
<!ATTLIST Left Seq CDATA #IMPLIED>
<!ATTLIST Right Flag CDATA #IMPLIED>
<!ATTLIST Right Seq CDATA #IMPLIED>

<!ELEMENT conn (conn_content)>
<!ATTLIST conn_content NumPhases CDATA #IMPLIED>
<!ATTLIST conn_content PhaseIdx CDATA #IMPLIED>
<!ATTLIST conn_content Pos1X CDATA #REQUIRED>
<!ATTLIST conn_content Pos1Y CDATA #REQUIRED>
<!ATTLIST conn_content Pos2X CDATA #REQUIRED>
<!ATTLIST conn_content Pos2Y CDATA #REQUIRED>

<!ELEMENT text (text_content)>
<!ATTLIST text_content FontName CDATA #IMPLIED>
<!ATTLIST text_content FontSize CDATA #IMPLIED>
<!ATTLIST text_content FontStyle CDATA #IMPLIED>
<!ATTLIST text_content Color CDATA #IMPLIED>
<!ATTLIST text_content Orientation CDATA #IMPLIED> <!--In tenths of degrees -->
<!ATTLIST text_content BckCol CDATA #IMPLIED>
<!ATTLIST text_content FrmCol CDATA #IMPLIED>
<!ELEMENT text_content (#PCDATA)>

<!ATTLIST variables NumSim CDATA #IMPLIED>
<!ATTLIST variables IOPCVP CDATA #IMPLIED>
<!ELEMENT variables (var*)>
<!ATTLIST var Name CDATA #REQUIRED>
<!ATTLIST var Expr CDATA #REQUIRED>

```

### 7.8.3 XML skeleton

```

<?xml version="1.0"?>
<project Application="ATPDraw" Version="7" VersionXML="1">
    <header Timestep="1E-5" Tmax="1" XOPT="0" COPT="0" TopLeftX="4180" TopLeftY="4475"/>
    <objects>
        <comp Name="" >
            <comp_content PosX="4910" PosY="5270" Icon="default">
                <node Name="IN" Value="" /> %if Name present, sequence does not matter
                <node Name="OUT" Value="" /> %Value contains the node name
                <data Name="RES" Value="10" />

```

```

<data Name="R_P" Value="MyVar"/>
  <datastring>Text string to include, incl. model script</datastring>
</comp_content>
<probe> %used by probes
  <monitor Phase="1"/>
</probe>
<nonlin> %contains nonlinear characteristics
  <point/>
</nonlin>
<LCC> %contains lines/cables with header and conductors
  <cable_header>
    <cable>
      <conductor />
    </cable>
    <line_header>
      <line/>
    </line_header>
  </LCC>
<Model></Model> %contains UseAs and Record. Script in datastring
<SM NumMasses=""> %used by synchronous machines
  <mass /> %contains output requests, mass data in comp.data
</SM>
<LINE3></LINE3> %used by LINE3 components
<BCTRAN> %used by BCTRAN transformer
  <winding /> %voltage and rating, coupling per winding
  <core /> %core information
  <short1 /> %binary test data, positive sequence
  <short0 /> %binary test data, zero sequence
  <open1 /> %open circuit test data, positive sequence
  <open0 /> %open circuit test data, zero sequence
</BCTRAN>
<XFMR></XFMR> %used by hybrid transformer, same structure as BCTRAN
<objects> </objects> %used by GROUPS. Content list in hierarchical levels
</comp>
<conn>
  <conn_content Pos1X="4920" Pos1Y="5010" Pos2X="4920" Pos2Y="5110"/>
</conn>
<text>
  <text_content>This is the text on screen
  </text_content>
</text>
</objects>
<variables NumSim="1" IOPCVP="0">
  <var Name="MYVAR" Expr="3.14"/>
  <var Name="RES" Expr="12*(KNT-1)+10"/>
</variables>
</project>

```

## 7.9 ATPDraw data structure and object model

ATPDraw has an object-oriented data structure with full support of inheritance and polymorphism as supported in Object oriented Pascal in Delphi 10.4. The ATPDraw objects inherit from the basic TObject class, while the containers inherit from the TObjectList class, as shown in Fig. 7.29. The ancestor class TATPDrawObject defines the foundation of all objects in ATPDraw, visualization, editing, storing etc. The most important class is the TATPDrawComp that handles IO of all data (with variables) and nodes in a protected and dynamic way, supporting inheritance in groups (TATPDrawGroup). The main container class TATPDrawObjectList manage iteration (enumerations) of multi-level groups in two different ways.

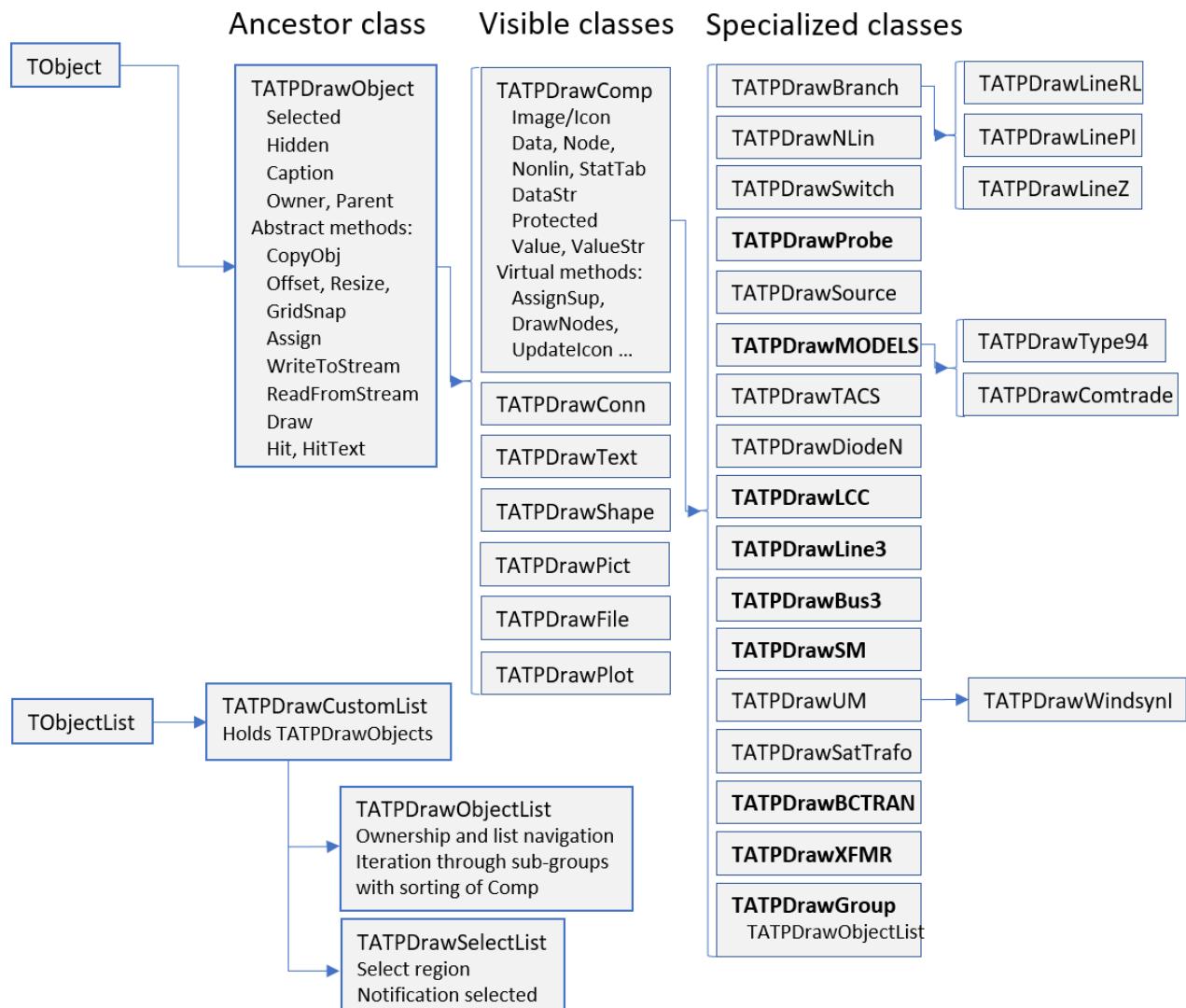


Fig. 7.29 ATPDraw data structure. Specialized classes in **bold** implements substantial content.

## 7.10 Examples project distributed with ATPDraw v7.5

- Exa\_1 MyFirst circuit. Single phase rectifier bridge.
- Exa\_2 Transpositions and 3-phase nodes.
- Exa\_3 Simple lightning study.
- Exa\_4 Induction machine fed by PWM source.
- Exa\_5 Reference cards and Library.
- Exa\_6 12-pulse rectifier, groups.
- Exa\_7 Switching study in 750 kV line.
- Exa\_8 Controlled MOV in FACTS device.
- Exa\_9 Detailed lightning study.
- Exa\_10 Inrush current study, BCTRAN.
- Exa\_11 Inrush current study, BCTRAN, groups.
- Exa\_12 Statistical switching overvoltages
- Exa\_13 Simple DC-DC converters. GIFU-switches.
- Exa\_14 24-pulse rectifier, harmonics.
- Exa\_15 Lightning induced overvoltages, adv. Models.
- Exa\_16 XFMR and BCTRAN inrush comparison.
- Exa\_17 Windsyn synchronous machine. (updated in v7)
- Exa\_18 Resonance grounding, scanning and optimization.
- Exa\_19 Double fed induction wind generator.
- Exa\_20 Power system toolbox, loads and relays.
- Exa\_21 Statistical lightning study, Monto Carlo, EGM. (new in v7)
- Exa\_22 Controlled synchronous machines (a, b, c). (new in v7)
- Exa\_23 Controlled induction machines. (new in v7)
- Exa\_24 IEEE 9BUS system with distance protection. (new in v7)
- Exa\_25 Controlled inverter interfaced solar plant. (new in v7)
- Exa\_26 Using the COMTRADE objects, packing objects, single pole reclosing (new in v7.1)
- Exa\_27 Embedded plotting example, inrush current display in multi-run (new in v7.2)
- Exa\_28 Three cases for illustration of Vector Fitting for JMarti and ULM (new in v7.4/v7.5)

## 7.11 References

- [1] *ATPDRAW version 3*, User Manual, TR A4389, EFI, Norway, 1996
- [2] Ned Mohan, *Computer Exercises for Power Electronic Education*, 1990, Department of Electrical Engineering, University of Minnesota.
- [3] *ATP-EMTP Rule Book*, Canadian-American EMTP Users Group, 1997
- [4] Lauren Dube, *MODELS in ATP*, Language manual, February 1996
- [5] H.W. Dommel, Electromagnetic Transients Program. Reference Manual (EMTP Theory Book), Bonneville Power Administration, Portland, 1986.
- [6] L. Prikler, Main Characteristics of Plotting Programs for ATP, EEUG News Vol. 6, No. 3-4, August-November 2000, pp. 28-33
- [7] B. A. Mork, F. Gonzalez, and D. Ishchenko: "Parameter estimation and advancements in transformer models for EMTP simulations. Task MTU-7: Model performance and sensitivity analysis", Bonneville Power Administration, Portland, OR, 2004.
- [8] B.A. Mork, F. Gonzalez, D. Ishchenko, D. L. Stuehm, J. Mitra." Hybrid Transformer Model for Transient Simulation-Part I: Development and Parameters", *IEEE Trans. Power Delivery*, Vol. 22, pp. 248-255, 2007.
- [9] B.A. Mork, F. Gonzalez, D. Ishchenko, D. L. Stuehm, J. Mitra, "Hybrid Transformer Model for Transient Simulation-Part II: Laboratory Measurements and Benchmarking", *IEEE Trans. Power Delivery*, Vol. 22, pp. 256-262, Jan. 2007
- [10] B. A. Mork, F. Gonzalez, D. Ichshenko, "Leakage inductance model for Autotransformer transient simulation", in *Proc. Int. Conf. on Power System Transients*, paper 248, 2005.
- [11] J. J. Graininger and W. D. Stevenson: *Power System Analysis*, McGraw-Hill 1994.
- [12] A. Greenwood: *Electrical Transients in Power Systems*, Wiley, 1991.
- [13] IEEE Working Group 15.08.09, Editors: A. M. Gole, J. Martinez-Velasco, A. J. F Keri, *Modeling and analysis of power system transients using digital programs*, IEEE 99TP133-0, pp. 4.12-4.13, 1998.
- [14] *IEEE Guide for Transient Recovery Voltage for AC High-Voltage Circuit Breakers Rated on a Symmetrical Current Basis*, ANSI/IEEE Standard C37.011-1994.
- [15] N. Chiesa, "Power Transformer Modelling: Advanced Core Model", M. SC. Thesis, Politecnico di Milano, Italy, 2005.
- [16] C. Zhu, R.H. Byrd and J. Nocedal : *L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization*, ACM Transactions on Mathematical Software, Vol 23, Num. 4, 1997 Page(s): 550 – 560.
- [17] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P, Flannery: *Numerical recipes*, 2<sup>nd</sup> Ed. 1992, Cambridge University Press.
- [18] Furst, G; Høidalen, H.K.: "Windsyn for ATPDraw", EEUG-meeting 2008, 22-24. Sept., Cesme, Turkey.
- [19] Rogers, G.J.; Shirmohammadi, D.: "Induction Machine Modelling for Electromagnetic Transient Program", *IEEE Trans. on Energy Conversion*, Vol. EC-2, Issue 4, pp. 622-628, Dec. 1987.
- [20] Zhu, C.; Byrd, R.H.; Nocedal, J.: "L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization", *ACM Transactions on Mathematical Software*, Vol 23, Num. 4, 1997 Page(s): 550 – 560.
- [21] Kundur, P: *Power System Stability and Control*, McGraw-Hill, 1996.
- [22] Hans Kr. Høidalen: "Power System Toolbox in ATPDraw 5.9 –Power Frequency Quantities and Relaying", Proc. EMTP Users Group meeting, Cagliari, Italy, Sept. 15-16, 2014.
- [23] Richard G. Lyons: "Understanding Digital Signal Processing", Prentice Hall PTR, Second Edition, 2004.
- [24] Z. Szadkowski: "16-point discrete Fourier transform based on the Radix-2 FFT algorithm implemented into cyclone FPGA as the UHECR trigger for horizontal air showers in the Pierre Auger Observatory", Nuclear Instruments and Methods in Physics Research A 560 (2006) 309–316.
- [25] A. Ametani: "A general formulation of impedance and admittance of cables", IEEE TR. of PAS, Vol. 99, No. 3, pp 902-910, 1980.

- [26] T. Noda: “Numerical techniques for accurate evaluation of overhead line and underground cable constants”, IEEJ Trans No. 3, pp.549-559, 2008.
- [27] J. A. Martinez-Velasco (Ed): “Power System Transients- Parameter determination”, *CRC Press*, 2010.
- [28] B. Gustavsen, A. Semlyen: “Simulation of transmission line transients using vector fitting and modal decomposition”, IEEE TRPWD, Vol. 13, No. 2, 1998.
- [29] B. Gustavsen, A. Semlyen: “Rational approximation of frequency domain responses by vector fitting”, IEEE TRPWD, Vol. 14, No. 3, 1999.
- [30] A. Morched, B. Gustavsen, M. Tartibi: “A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables”, *IEEE TRPWD*, Vol. 14, No. 3, pp. 1032-1038, 1999.
- [31] E. S. Bañuelos-Cabral et.al.: “Accuracy enhancement of the JMarti model by using real poles through vector fitting”, *Electrical Engineering*, Vol. 101, pp. 635-646, 2019
- [32] F. O. S. Zanon, O. E. S. Leal, A. De Conti: “Implementation of the universal line model in the alternative transients program”, *Electric Power Systems Research*, No. 197, 2021.
- [33] B. Gustavsen: “Optimal time delay extraction for transmission line modeling”, IEEE TRPWD, Vol. 32, No. 1, pp. 45-54, 2017.
- [34] H. W. Dommel: “EMTP Theory Book”, BPA, 1986.
- [35] P. Kuffel, K. Kent, G. Irwin: “The implementation and effectiveness of linear interpolation within digital simulation”, *Electric Power & Energy Systems*, Vol. 19, No. 4, pp. 221-227, 1997.

## 7.12 Index

\$

- \$Include ..... 121
- \$PARAMETER ..... 57, **83**, 86
- \$Prefix & \$Suffix ..... 81
- \$Vintage ..... 119

A

- action mode ..... 45
- Alternative Transients Program ..... 12
  - licensing ..... 21
  - on-line licensing ..... 21
- Armafit command ..... 186
- ATP
  - ATP file ..... 23
  - DBM-file ..... 226
  - lib-file ..... 183
  - punch-file ..... 183
  - Rule Book ..... 42
  - run ATP ..... 53
- ATP Connection Wizard* ..... **25**
- ATP menu ..... 27, 78
- ATP settings ..... 78
  - Output ..... 52
  - Simulation ..... 51
- ATPDraw ..... 11
  - ATPDraw.ini ..... 24
  - configuration ..... 24
  - Default view options ..... 113
  - Directories ..... 112
  - download ..... 21
  - Edit options ..... 113
  - Edit settings ..... 113
  - examples ..... 249
  - include files ..... 23
  - installation ..... 22
  - interface ..... 25, 33
  - on-line help ..... 28
  - options ..... 110
  - Preferences ..... 111
  - project file ..... 23
  - support file ..... 23
- ATP-EMTP-L mailing list ..... 28
- Attachment
  - Input dialog ..... 124
- Auto-detect errors ..... 80

B

- Bitmap editor ..... 102

C

- CABLE CONSTANTS ..... 139
- cable data page ..... 192
- CABLE PARAMETERS ..... 139

Canadian/American EMTP User Group

- Tsu-huei Liu ..... 12
- W. Scott Meyer ..... 12
- circuit files ..... 66
- circuit font ..... 77
- Circuit objects ..... 34
- Circuit window ..... 34
- command line options ..... 24
- Component
  - attributes ..... 38
  - Characteristic tab ..... 119
  - Hide button ..... 119
  - Input dialog ..... 38, 42, 117, **118**
  - nonlinear characteristic ..... 121
  - Order ..... 119
  - Output request ..... 119
- Component selection menu ..... 33, 37, 117, 129
- Compress ..... 35, 71, 74, **165**, 235
- COMTRADE ..... 60, 317
- Connection ..... 39
  - Input dialog ..... 54, 122, 264
  - creating ATP-file ..... 51

D

- DC machine ..... 175
- delta T ..... 51
- distributed line ..... 138
- download ..... 21
- drag and drop ..... 24, 162
- Drag and drop ..... 124
- duplicate ..... 46

E

- Edit
  - Arrange ..... 70
  - copy ..... 69
  - copy graphics ..... 69
  - cut ..... 69
  - duplicate ..... 69
  - LINE3 ..... 71

paste .....	69
polygon selection .....	70
select object.....	70
edit ATP-file .....	52, 90
Edit circuit.....	73, 169
Edit commands .....	94
Edit group .....	73, 169
Edit LIS-file .....	91
Edit operations overview .....	37
Electromagnetic Transients Program.....	11
EMTP.....	12
applications .....	15
Rule Book .....	371
TPBIG.EXE .....	25, 26
user group .....	21
enclosing pipe .....	186
export circuit .....	67
external programs .....	27
extract.....	73

**F**

File	
Input dialog .....	124
flux probe .....	300
Fortran.....	153

**G**

gridsnap.....	37
ground symbol .....	50
Group	
Input dialog .....	73, 169, 261
selection .....	45
Grouping, see Compress .....	165

**H**

HARMONIC FREQUENCY SCAN.....	30
Harmonic source .....	180
Help editor .....	99, 108
Help menu.....	115
Help topics .....	115
hierarchical modeling .....	11
Høidalen.....	11, 29

**I**

Icon editor .....	99
import circuit.....	67
Import Power System.....	67
Include characteristic .....	145
Induction machine.....	175
initial conditions .....	12, 153
Internal Parser.....	83

**J**

JMarti .....	139, 196
JMarti line .....	188

**L**

LCC object .....	183, 249
Library .....	95
ATPDraw.scl .....	<b>95</b>
edit component.....	95
User specified template .....	96
Library menu.....	39
LINE CONSTANTS .....	139
line data settings .....	190
LINE MODEL FREQUENCY SCAN .....	198
line/cable dialog .....	139, 184
linear branch.....	133
lines/cables .....	129, 136
load flow .....	83, 156

**M**

Machines .....	130, <b>266</b>
Main menu .....	34, 66
Main window .....	33, 65
Map window .....	34, 115
master/slave.....	305
metafile.....	67, 200
miscellaneous parameters .....	51
MODELS .....	12, 13, 130, 145, 206
Component dialog .....	207
Debugger .....	211
Editor.....	208
Input dialog .....	148
mod-file.....	101, 213
new object .....	214
record .....	217
script structure.....	148
sup-file .....	101
Modified flag.....	51
mouse operations.....	36
multi-layer circuit.....	165

**N**

new circuit.....	41, 66
Noda line .....	189
Node	
Input dialog .....	37, 125
Node attribute.....	215
node name restrictions .....	53
nonlinear branch.....	135

**O**

## Object

- Input dialog.....36
- Sequence.....58
- open project.....66
- Optimization.....92, 238, 240, 241, 242
- Output combo box.....49
- Output Manager.....88, 89, 90, 306, 308
- Output settings.....80

**P**

- phase sequence .....56
- Picture
  - Input dialog.....123
- Plotting
  - Embedded.....60, 320
  - PlotXY.....15
  - Programs.....14
- PlotXY .....15
- Post-processing.....59
- POWER FREQUENCY CALCULATION ..198, 325
- Power System Toolbox .....59, 157, 350
- Probes .....131
  - Curr.....43, 59
  - Input dialog.....127
  - Steady-state .....127
  - Volt.....44, 59
- Probes & 3-phase.....131
- project file.....51, 66
- public domain.....12

**R**

- redo.....68
- reference object .....56, 133
- refresh.....77
- reload icon .....101
- Result Directory .....23
- rubber band.....70, 71
- run ATP .....51, 88
- running simulation.....53

**S**

- save circuit.....51
- save project.....66
- select group.....38
- Selection dialog .....37, 117, 128
- Semlyen line.....189
- Shape
  - Input dialog.....123
- Shortcut menu.....117

Sidebar .....35, **74**

simulation settings .....79

single core cable .....186

## Solver

- ATP.....**25**
- Internal.....16, 53, **360**
- sorting cards.....81
- sources .....130, 141
- Splitter .....54, 132
- standard components .....29
- standard library .....161
- statistical switch.....179, 305
- Status bar .....35, 74
- Supporting routines .....13
  - BCTRAN.....26, 218, 232
  - CABLE CONSTANTS .....26
  - DATA BASE MODULE .....26
  - LINE CONSTANTS .....26
- switches .....129
- Synchronous machine.....143, 175, **267**

**T**

- TACS .....12, 13, 130
  - coupling to circuit.....151
  - devices .....152
  - menu .....151
  - transfer functions .....152
- Template dialog .....108
- Template editor.....96
- Text
  - Input dialog.....123
- Text editor .....91, 109
- Toolbar .....46, **76**
- Tools menu .....102
- Transformers.....130, 144
  - BCTRAN .....145, 218
    - auto-transformer .....293
    - dialog .....218, 293
    - Import/Export .....220
  - Input dialog .....262
  - inrush currents .....292
  - Saturable transformer .....173, 261, 264
  - Selection menu .....144
  - XFMR .....145, **221**, 303, **333**
- transposition .....56, 133, 138
- trapezoidal rule .....12
- Type94 .....149
  - Input dialog.....149

***U***

- ULM..... 139, 184, 189, 197, **202**
- undo ..... 46, 68
- Universal machine ..... 143, 175
- untransposed ..... 138
- User specified
  - Additional ..... 155
  - Component dialog ..... 231
  - create new objects ..... 226
  - DBM-file ..... 227
  - Library object ..... 155
  - nonlinear transformer ..... 232
  - Reference object ..... 156
  - Selection menu ..... 131, 155

***V***

- Variables ..... 39, 57, **83**, 84, 235
  - Internal Parser ..... 83, 86
- Vector graphic editor ..... 103
- Verify button ..... 186
- View options ..... 77

***W***

- Windsyn ..... **181**, 287
- WWW
  - www.eeug.org ..... 21
  - www.emtp.org ..... 21

***Z***

- zoom ..... 76