CHAPTER 7

A TRANSIENT STABILITY SIMULATION PACKAGE (TSSP) UNDER SIMULINK

This chapter describes the Transient Stability Simulation Package (TSSP) under SIMULINK environment. It introduces the package under DOS operating system. It then points out the areas in TSSP where improvement can be made in order to better the performance of the package. For example, handling generator saliency is difficult in stability simulation studies. This difficulty will be overcome in the new software to be developed. Building blocks of various power apparatuses are developed and a rich library including such blocks and MATLAB functions is formed. Basic instructions on how to construct a simulation model for a power system are given, including an illustrative example.

7.1 TSSP UNDER DOS

Time domain simulation of power systems has been used for stability studies for decades[7–1]~[7–4]. Various large scale simulation packages such as the EPRI–ETMST (Extended Transient/Mid Term Stability Program) and the

PTI–PSS/E (Power System Simulation/E Program) are in wide use. A number of educational softwares are also reported in the literature [7–5]~[7–8].

Every simulation package has its own advantages and shortcomings. Most packages do not have all the features and flexibility that one needs for either education or research purposes. The possibility of changing the capability of the package is often limited if not impossible. There has been an urgent need to have a package that has the capability of fulfilling most requirements in both teaching and research. To answer this need, a transient stability simulation package(TSSP) implemented on an IBM PC has been developed[7–9]. This package has been utilised to perform stability studies of the WSCC system [7–10] and the New England Test system[7–11]. As a research tool, the package has also been used to conduct a stability investigation of a longitudinal power system having 69 buses and 12–machines [7–12]. The following is a list of tasks that can be performed using the TSSP:

- Machine modelling of varying complexity;
- Transient simulation of induction motors;
- Modelling of symmetrical and asymmetrical disturbances;
- Various load modelling;
- Modelling of AVR, AGC and PSS;
- Interfacing of user–defined modules;
- Load flow studies; and

- Short circuit calculations.

The transient behaviour of a power system component or a controller is modelled by a set of first order differential equations. Based on the implicit integration method [7–13], this set of differential equations can be simulated by building a block diagram with integrators and DC gains. Each block diagram forms an independent module to be called upon by the co–ordinator in the TSSP. This package has modules for four different synchronous machine models, one induction motor model, all the current IEEE recommended excitation system models, the standard power system stabiliser model[7–14], and speed governing and turbine system models[7–15]. Different load models can also be included [7–16].

A maximum of three input data files are needed to perform a transient stability simulation. Each data file consists of a number of data blocks. In each block, data items are entered as a record in free format with a comma between items. A dash line (/) signifies the end of a record. Preceding each block, there are comments and definitions specifying what data are to be entered and how. Any length of comments can be added into the files as long as an exclamation mark (!) precedes each comment line.

Data file *pqlf.inp* contains all the information required to perform a load flow study. It also provides the freedom to use different system base MVA, per unit or nominal unit system. Data file *ntwk.inp* contains fault information, sequence network parameters and branch incidence of the zero

sequence network. Data file *mach.inp* contains parameters for machine, excitation system (AVR), speed governor and turbine system (AGC) and power system stabiliser (PSS) in four blocks. Each controller is identified in the corresponding block by its machine number. If a controller is absent from a machine, a record 888/ is entered to signify this fact.

Three sample data files are provided in TSSP. The user can create his/her own data files by copying and modifying them with a DOS editor or any word processor according to the system under study. Any error in the data files will be automatically detected and located when the programs are started. The format of the data files is designed to provide maximum flexibility and minimize frustration in data input. This design was inspired after experiencing the "sophisticated" and "non user–friendly" data input in EMTP [7–17].

The overall structure of the package is shown in Figure 7.1. To initialise the TSSP, the fast decoupled load flow (FDLF) is run first. Then the short circuit calculation (STCC) program is executed. The results of the two runs are passed onto the TSSP co–ordinator internally. Finally the transient simulation program (TSSP) is run. The end result of a simulation is stored in the file *simu.out*. A data sorting and plotting program (WPLOT) can be employed to plot a maximum of five curves on one screen and up to fifteen variables can be sorted out at a time.

7.2 NECESSITY OF IMPROVEMENT

TSSP is a numerical realisation of solving a large mathematical system composed of both differential and algebraic equations. Let x denote the system state variable, and y the auxiliary variable. Then the mathematical system describing the dynamic behaviour of a power system can be expressed as

$$\frac{d}{dt}(x) = f(x, y) \tag{7.1}$$

$$0 = g(x, y) \tag{7.2}$$

where

x = the state variables, and

y = the auxiliary variable.

A numerical solution to the system can be obtained by different algorithms, assuming that an initial condition is known. This initial condition is obtained by a load flow program. The solution procedure in TSSP consists of the following steps;

1. With the initial values $x^{(0)}, y^{(0)}$ known, and with a given disturbance in the system, a new value of the auxiliary variable $y^{(1)}$ is calculated from (7.2), keeping the state variable at $x^{(0)}$.
2. The derivative of the state variable is calculated from (7.1).
3. The new value of the state variable $x^{(1)}$ is calculated using a numerical method such as the implicit integration.

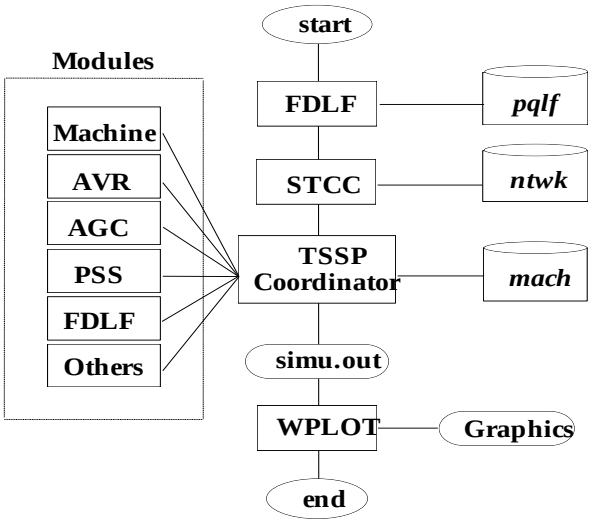

Fig. 7.1 Overall Structure of the TSSP

This calculation process is continued until the simulation time is reached. At each time step, the calculated values of $x^{(i)}$ and $y^{(i)}$ are stored so that they can be plotted at the end of simulation.

Because of the fact that there exists saliency in synchronous machines, the solution of (7.2) is accomplished by an iterative procedure in TSSP[7–2]. Depending on the structure of the saliency, this iterative procedure may lead to divergence and no solution can be obtained at certain times and with certain network configurations. For real time studies, this iterative process may impose unacceptable computation effort. Another disadvantage of this package is that to add any new features to the program, such as displaying a variable which is not listed in the plotting utility, modelling a new disturbance, or interfacing a user–defined model, part of the package has to

be recompiled. This compilation process has to be continued until the desirable results are produced.

7.3 TSSP UNDER SIMULINK

7.3.1 Introduction

SIMULINK is an interactive block–diagram environment for modeling, analyzing, and simulating dynamics of physical and mathematical systems. It provides point–and–click interface for all model building. Block–diagram and differential equation models, continuous–time, discrete–time, hybrid models, linear and nonlinear models, trained neural networks and fuzzy logic models, can all be simulated in this environment.

Though SIMULINK is powerful in simulating the above systems, it can not be directly applied to power system simulations. The reason is that power systems run a three–phase complex number system and their control systems are modeled by real number transfer functions. In order to comply with the rules of SIMULINK, a power system has to be decoupled from complex representation to real number representations which are asymmetrical when salient synchronous machines are present in the system. In the section that follows, the basics of SIMULINK are discussed. Then detailed building blocks for the modules in TSSP will be constructed using SIMULINK library and MATLAB functions, forming the TSSP Library.

7.3.2 Basics of SIMULINK

To simulate a dynamic system, we need to create its model in *a block diagram* window using mouse driven commands. As SIMULINK uses the metaphor of a block diagram to represent dynamic systems, creating a model is much like drawing a signal flow graph. The individual blocks need not be drawn, but are represented by input/out blocks from the SIMULINK standard library. Once the model is defined, its dynamic performance can be simulated by built–in analysis tools. The progress of the simulation can be viewed while the simulation is running, and the final results can be made available in the MATLAB workplace when the simulation is complete. Specifically, the following steps can be followed to create and analyse the model for a given dynamic system.

1. At MATLAB prompt, type **`simulink`** and press return. This will display the standard library of block diagram.
2. Pull down the MATLAB file menu and open up a new model file which is a blank *block diagram* window.
3. Copy individual blocks from the standard library to your *block diagram* window; and repeat this step until the model is complete.
4. Save that model you just defined as a *m*–file which you can modify or run later on.

7.3.3 Building Blocks

(1) *Synchronous Generators*

In Chapter 6 various synchronous generator models are presented. In this section, their simulation models will be built. This is done by showing one detailed example, i.e., building of the *two–axes model with subtransient*. The other three models will be listed in Appendix B. Figure 7.2 shows the simulation model of a synchronous generator. A functional block for both an excitation system and a speed governing and turbine system is included. These functional blocks have to be replaced by the simulation models of their corresponding physical equipment.

The input to the model is a vector output from a MATLAB function that solves the network equations of the power system under study. It can be expressed as follows, for generator number 1:

$$in\_1 = \left[i_d, i_q, V_x, V_y, i_x, i_y \right]^{\tau} \tag{7.3}$$

The output of the model is a vector that can be expressed as follows:

$$out\_1 = \left[E_q^{'}, E_q^{''}, E_d^{''}, \delta \right]^{\tau} \tag{7.4}$$

The output of the excitation system is E_{fd}. And the output of the speed governing and turbine system is P_m. The rest of the model is to simulate the differential equations (6.18) – (6.22). Note that the memory blocks(M01–M03) in the model function as a breakdown of a complete loop for simulation purpose. Values of parameters are passed to their corresponding variables in

the various blocks from the workplace where these variables are initialized before simulation is started.

(2) *Excitation Systems*

As an example to show how to build a simulation model for an excitation system, we choose the DC Type 1 excitation system. Figure 6.3 shows its block diagram. Figure 7.3 shows a simulation model of the system. The input to the model has two variables, i.e., the terminal voltage and the output signal from a power system stabiliser. The output of an excitation system is always the excitation field voltage, E_{fd}. There are two MATLAB functions; i.e., VT block converts a voltage in complex components to its magnitude and SE calculates the saturation factor of the excitation curve at an operating point. Other types of excitation systems are given in the TSSP library.

(3) *Prime Movers*

The hydro turbine and governor system, shown in Figure 6.5, can be accurately simulated in TSSP, since nonlinear elements of a dynamic system can be easily modelled in SIMULINK. Figure 7.4 shows the simulation model in which there are two subsystems. Each of these subsystems simulates a transfer function with initial variable values. The input to the model is the machine speed in per unit and the output is the mechanical power in per unit exerted to the shaft of generating system.

Fig. 7.2 Simulation Model of A Synchronous Generator

Fig. 7.3 Simulation Model of the DC Type 1 Excitation System

The steam turbine system, shown in Figures 6.7 and 6.8, is simulated by the model shown in Figures 7.5 and 7.6. The input to the governor model is the incremental change of the shaft speed of the generator. The output is the valve positioning that controls the inlet of steam and is equivalent to the total power input to the turbine which converts the heat energy to mechanical energy and outputs it as P_m. Other steam turbines are also included in the TSSP library.

(4) *Conventional Power System Stabilisers*

Three IEEE conventional power system stabilisers(PSSs) are presented in Chapter 6. Their simulation models in SIMULINK are constructed here. They are also included in the TSSP library. The input to a PSS can be a combination of different signals, as has been pointed out in Chapter 6. Figures 7.7, 7.8 and 7.9 give the details of these simulation models.

(5) *Induction Motors*

The block diagram of an induction motor is shown in Figure 6.13. As an induction motor is treated like a generator in TSSP, its simulation model in SIMULINK is similar to that of a generator too, as shown in Figure 7.10. The input to the model is a vector output from the network solution MATLAB function, as expressed in (7.3). The output of the model is a vector given by

$$out\_1 = \left[E_q^{'}, E_d^{'}, \delta \right]^{\tau} \tag{7.5}$$

The load of the induction motor is passed to the model from the workplace by the variable TM(1) which is calculated by either (6.60) or (6.61). There are two MATLAB functions; PE calculates the electric power the motor consumes and PL calculates the power loss in the stator windings. The rest of the model simulates the differential equations (6.5) – (6.59) of the system.

(6) *DC Transmission Lines*

A three–phase alternating current(AC) transmission line system is represented by the positive–sequence network of one of the three phases in transient stability studies(see next section for more detail). DC transmission lines can be simulated by three categories of models; (a) simple model, (b) response model, and (c) detailed model. Depending on its application, each DC transmission line is designed to meet specific requirements in the overall performance of the system, therefore, no standard model is developed to represent DC links in stability studies[7–18].

Fig. 7.4 Simulation Model of the Hydrogovernor Turbine System

Fig. 7.5 Speed Governing System for Steam Turbines

Fig. 7.6 Tandem–Compound Double–Reheat Steam Turbine

Fig. 7.7 Simulation Model of IEEE Standard PSS

Fig. 7.8 Simulation Model of IEEEST PSS

Fig. 7.9 Simulation Model of IEEESN PSS

Fig. 7.10 Simulation Model of An Induction Motor Model

(7) *Network Function*

In a power system, generators, loads and other power apparatuses are connected though transmission lines to form a composite dynamic system. As generators and their control systems are modelled by real number transfer functions while the network is represented by complex numbers, we need to decouple the complex number system. This is done in TSSP through a MATLAB function in which an AC system is modelled by

$$\begin{pmatrix} I_{xy1} \\ I_{xy2} \\ . \\ I_{xyn} \end{pmatrix} = \begin{pmatrix} Y_{11} & Y_{12} & . & Y_{1n} \\ Y_{21} & Y_{22} & . & Y_{2n} \\ . & . & . & . \\ Y_{n1} & Y_{n2} & . & Y_{nn} \end{pmatrix} \begin{pmatrix} V_{xy1} \\ V_{xy2} \\ . \\ V_{xyn} \end{pmatrix} \tag{7.6}$$

where $I_{xyi} = \begin{bmatrix} I_{xi} & I_{yi} \end{bmatrix}^{\tau}$, the injected current,

$V_{xyi} = \begin{bmatrix} V_{xi} & V_{yi} \end{bmatrix}^{\tau}$, the voltage at bus i, and

$Y_{ij} = \begin{pmatrix} G_{ij} & -B_{ij} \\ B_{ij} & G_{ij} \end{pmatrix}$, its elements being the real and imaginary parts of the element $\left(G_{ij} + jB_{ij}\right)$ at position (i, j) of the admittance matrix of the network.

The interfacing point between the network and generators, loads and other power apparatuses is the bus terminal where injected current and bus voltage hold true for both parties. With no approximation and no iteration, this interfacing is accomplished in the MATLAB function, using exact algebraic equations as derived in Chapter 6. For any generator or load model,

if its interfacing condition with the network is expressed in terms of voltage and current in $d-q-0$ quantities, they can be transformed into quantities in $x-y$ coordinates. For example, a current transformation is accomplished by

$$\begin{pmatrix} I_x \\ I_y \end{pmatrix} = \begin{pmatrix} \cos(\delta) & \sin(\delta) \\ \sin(\delta) & -\cos(\delta) \end{pmatrix} \begin{pmatrix} I_q \\ I_d \end{pmatrix} \tag{7.7}$$

where δ is the angle between the reference $x-$axis in $x-y$ coordinates and the $q-$axis of the device to be modelled. This transform applies to voltage too. Detailed information on the network function is given in Appendix B. This topic is further covered in the coming section where model construction for a power system is described.

7.3.4 Model Construction

In this section, we demonstrate how to construct a simulation model for a simple one–machine infinite bus power system. The procedure can be extended to the case of multimachine systems which will be covered in Chapter 8. Figure 7.11 shows a single–line diagram of the system. Assume that the generator can be simulated by the *two–axes with transient* model(GEN_3). The excitation system is simulated by ST_4. The speed governing and turbine system is simulated by GOV_221. The load is an induction motor modelled by IndMotor block from the TSSP library. The infinite bus is simulated by GEN_0 block. Figure 7.12 illustrates the final simulation model of the power system. Note that the MATLAB function is a

m–file referred as find_vif.m, meaning to find voltage(v) and current(i). This file is listed in Appendix B. The input vector to a MATLAB function can have only a single argument with a number of elements. This allows access to the numerous variables output from various devices each of which may have an arbitrary number of outputs. The output of the function is also a vector with the number of components equal to a multiple of the number of the devices modeled. This is necessary because the elements of the output vector have to be accessed by each device appropriately. A simulation function referred to as Demux can divide its input only into equal parts.

7.4 TSSP'S POTENTIAL WITH SIMULINK

We have seen some merits of TSSP under SIMULINK environment, and other potential applications is of interest. With the new DSP Blockset and a real–time C code generator[7–19], we can develop real–time software for target hardware and test the generated C code in actual applications. This C code generator is useful for batch file simulation studies too, as the size of a power system increases, model simulation in SIMULINK may be slow. The generated portable C code can be used in different situations. Another advantage is that we can use advanced control techniques such as ANNs and FL to design intelligent control systems. With TSSP in SIMULINK, we can prototype, simulate and implement a control on a single personal computer.

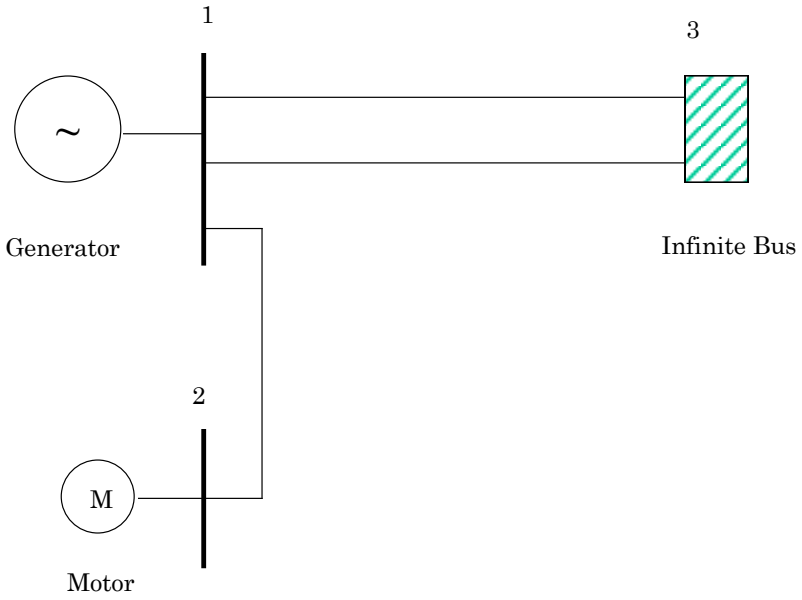

Fig. 7.11 Single–Line Diagram of the One–Machine System

Fig. 7.12 Simulation Model of the One–Machine System