Hydro Turbine and Governor Modeling and Scripting for Small-Signal and Transient Stability Analysis of Power Systems

September 23, 2011

WEI LI

KTH Electrical Engineering

Master's Degree Project
Stockholm, Sweden 2011

XR-EE-ES
2011:016

February 2011 to September 2011

Hydro Turbine and Governor Modeling and Scripting for Small-Signal and Transient Stability Analysis of Power System

— Diploma Thesis —

Wei Li[1]

Electrical Power Systems Division

School of Electrical Engineering, KTH Royal Institute of Technology, Sweden

Supervisor
Dr. Luigi Vanfretti
Yuwa Chompoobutrgool
KTH Stockholm

Examiner
Dr. Luigi Vanfretti
KTH Stockholm

Stockholm, September 23, 2011

[1] liwei.emma@gmail.com

Abstract

Today, hydro-power accounts for one half of Sweden's electricity net production of electrical power generation. As one of the most important renewable energy sources, the exploitation of hydro-power naturally attracts more and more attention worldwide. In hydro power production systems, we can not neglect turbine and governor's functions, which participate in the primary frequency control of synchronous machines. This thesis studies accurate and detailed hydro turbine and governor models, and implements these models in two software: the Power System Analysis Toolbox (PSAT)—an open source software—and MATLAB/Simulink combined with the SimPowerSystems (SPS)—a proprietary software. How to implement models in power system software is also explained in detail. Then, with the developed hydro turbine and governor models, simulation studies are carried out on different scale test systems ranging from a single-machine infinite-bus (SMIB) system to larger systems including the KTH-NORDIC32 system. Furthermore, corresponding transient stability analysis, small signal stability analysis and frequency response analysis are provided.

Acknowledgments

This master thesis has been carried out at the Department of Electric Power System (EPS) at the Royal Institute of Technology (KTH).

Firstly, I would like to thank my two supervisors, Dr. Luigi Vanfretti and Yuwa Chompoobutrgool for giving me this chance to work with them in EPS, KTH, together with their guidance and advices to my thesis. I believe I learn a lot from them on both knowledge and the attitude forwards research.

I would also like to thank all members in our Smart Transmission Systems Lab for motivating discussionS, communication and fun time.

Finally, I always thank my parents and boyfriend for their support throughout my studies.

Wei Li
September 23, 2011

Contents

Notation

CAD Computer-Aided Design

CPF Continuation Power Flow

DAE Differential and Algebraic Equations

EMT Electro Magnetic Transient

EST Educational Simulation Tool

FFT Fast Fourier Transform

GUI Graphic User Interface

OPF Optimal Powr Flow

PAT Power Analysis Toolbox

PF Power Flow

PMU Phaser Measurement Unit

PSAT Powe System Analysis Toolbox

PST Power System Toolbox

SMIB Single Machine Infinite Bus

SPS SimPowerSystems

SSSA Small Signal Stability Analysis

TDS Time Domain Simulation

1 Introduction

1.1. Background

Power transmission networks are inherently complex systems which mainly consist of generators, transmission lines, transformers and loads. At the same time, they are critical infrastructures supporting social and industrial development. An increase in electricity demand, together with a need for reliability enhancement, heavily affects overall system performance. As a consequence, system stability is of importance and must be analyzed by means of appropriate tools. To guarantee the secure and reliable operation of actual power systems, computer-based simulation tools are used by engineers to assess the stability dynamic and performance of the power system.

In power systems, generators are commonly driven by mechanical power provided by turbines. Each turbine is equipped with a speed governing system called turbine governor to regulate its speed in order to meet a desired power output [1]. Turbine governors function as primary frequency controllers of synchronous machines when the system is subjected to disturbances. Figure 1.1 shows the basic elements of a turbine and a governor within the power system environment.

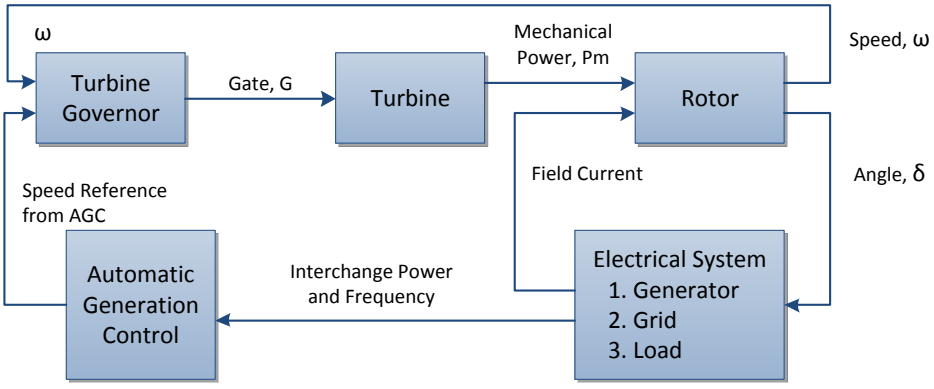


Figure 1.1.: Functional block diagram showing relationship of the turbine and governor with the overall power system

There are three main types of turbine and governors depending on their power sources: hydro, steam and thermal. Each type is different from one another, thus requiring different mod-

eling. For example, time delays in hydro turbine and governor models are due to transient droop compensation, pilot valves and gate servomotors. Moreover, the effect of water inertia in hydro turbines significantly influences the governor's requirements and must be compensated by a rate feedback, which must be reflected in the modeling of thesis devices. While in the steam turbine and governor models, time delays are due to steam chest and inlet piping, re-heaters, and crossover piping [3]. This thesis mainly focuses on modeling and scripting (i.e. software implementation) of hydro turbines and governors.

Implementing turbine and governor models in power system software is one important aspect of this thesis. With the advance in computer software, there has been a large number of power system simulation software packages developed, which can be basically divided into three categories: proprietary software, open source software and free software [12]. Proprietary software are conceived by the general public to be well-tested and computationally efficient. However, license agreements restrict their use imposing different conditions. Users are prohibited and generally have no possibility to modify the source code nor to distribute binaries (i.e. executables). In this case, implementing new components in proprietary software, although it is possible in some programs through specific modeling tools inside the proprietary package, is very difficult and error prone. Additionally, they seldom allow to import or export data in different formats [13].

On the other hand, open source software are usually freely distributed on line and less cumbersome for educational and research purposes. More importantly, they allow users to change the source code, add new algorithms, or implement new components. As for the free software, *"Free Software is a matter of liberty, not price."* [14] It supplies the users' freedom to run, copy, distribute, study, change and improve the software. GNU Operation System is established to preserve protect and promote this freedom, and to defend the rights of free software users. To this aim, the Free Software Foundation created the GNU General Public License (GPL), which guarantees that anyone distributing a complied program under GPL must also provide the source code. This is the significant difference between free software and open source software: once a program in free software is "opened", it cannot be "closed" anymore. Since open source software does not enforce copyleft, someone can develop closed source application based on the code [15], [16]. This thesis focuses on the implementation of hydro turbine and governor models in two simulation software: Power System Analysis Toolbox (PSAT)—an open source software—and SimPowerSystems (SPS)—a proprietary software.

1.2. Objectives

1.2.1. General Objectives

The main objective of this thesis is to develop models of hydro turbine and governors using MATLAB or GNU/Octave in PSAT. To this aim, a thorough study on hydro turbine and governor models shall be carried out. Following the study of turbine and governor models is a study of PSAT, e.g. how to use PSAT and how to build models in PSAT. Studied models shall be implemented in PSAT and their performance will be evaluated by simulations.

The second major objective is to implement the developed hydro turbine and governor models in PSAT. To this aim, simulation studies shall be carried out on different scale test systems

ranging from a single-machine infinite-bus(SMIB) system, to larger systems (including the KTH-NORDIC32 system). To verify their performance, the results shall be compared with a proprietary power system simulation software—SimPowerSystems (SPS).

1.2.2. Specific Objectives

- Study models of hydro turbine and governor systems.
- Implement each model of the turbine and governor systems in the Power System Analysis Toolbox (PSAT)—a free and open source software.
- Implement the same models in MATLAB/Simulink SimPowerSystems (SPS)—a proprietary software.
- Simulate the implemented models in PSAT and compare them with SPS. The simulation of the turbine and governor models will be performed on a single-machine infinite-bus (SMIB) system for both transient and small signal stability analyses.
- Perform additional simulations on other larger systems, e.g. the KTH-NORDIC32 system, using PAST.

1.3. Literature Review

There has been a plethora of work done on hydro turbine and governor modeling in power system studies, especially by IEEE committees and IEEE working groups. The report [4] written by an IEEE committee includes the representation of steam and hydraulic turbines and their speed-governing (governors) systems for power system stability studies. The separation of hydro turbine and governor into two parts is introduced. Both general and detailed models of the hydro turbine and governor are given in this report.

At one time, the turbine and governor models proposed in [4] had been widely used. However, modeling accuracy requirements increased greatly and more detailed models had to be developed and implemented in simulation programs. As a consequence, the Working Group on Prime Mover and Energy Supply Models for System Dynamic Performance Studies developed enhanced models in [5] to meet the increase demand in modeling accuracy. For the modeling of turbine conduit dynamics (turbine models), the report introduces a non-linear model assuming a non-elastic water column, linear models, traveling wave models, a non-linear model including surge tank effects, non-elastic water columns and other three models. For hydro turbine controls, proportional control with transient droop governors, PID governor including pilot and servo dynamics and enhanced governor mode are presented.

Apart from the background literature on hydro turbine and governor modeling, knowledge and experience with a power system software tool is essential. [7] and [8] supply a basic introduction about PSAT, and the PSAT manual [2] provides a reference for mastering the software and further applications.

SimPowerSystems, the proprietary simulation software used in this study, extends MATLAB Simulink with tools for modeling and simulating the generation, transmission, distribution, and consumption of electrical power. It provides necessary components used in power systems studies and analyses. The introduction and data sheet are supplied in [18]; demos and examples can be found in SimPowerSystems library in Simulink.

1.4. Outline

The rest of this thesis is organized as follows. Chapter 2 presents the modeling of hydro turbines and governors, including their main characteristics and the models studied in detail. In Chapter 3, after a general introduction of simulation software, namely, PSAT and SPS, detailed model implementation is discussed. Model validation with a SMIB and the KTH-NORDIC32 systems are presented in Chapter 4. Chapter 5 discusses the comparison of model simulation performance between PSAT and SPS, respectively, and manual data translation between these two software. Finally, the thesis finalizes with conclusions and future work in Chapter 6.

2 Hydro Turbine and Governor Modeling

This chapter includes a general introduction on hydro turbine and turbine governors, four different types of hydro turbine and governor models and their relevant diagram, equations and initializations. The parameters for all the models are provided at the end of this chapter.

2.1. Main Characteristics of Hydro Turbine and Governor

When water flows from high elevation to the hydro turbine, gravitational potential energy is converted into kinetic energy. Then, the turbine shaft, obtaining mechanical energy from the conversion, drives the machine to generate electricity. In a turbine, the power is controlled by regulating the flow into the turbine using the position of the gates or nozzles. This regulation is realized by the turbine governor, which is also called the *speed governing system*, or *turbine governing system*. Generally, hydro turbine governors can be classified in two types: mechanical hydraulic or electro hydraulic, depending on if there are electronic apparatus participating in sensing and measuring work in the turbine governor. Figure 2.1 shows the relationship of the turbine and governor, indicated by the red block, with the power system.

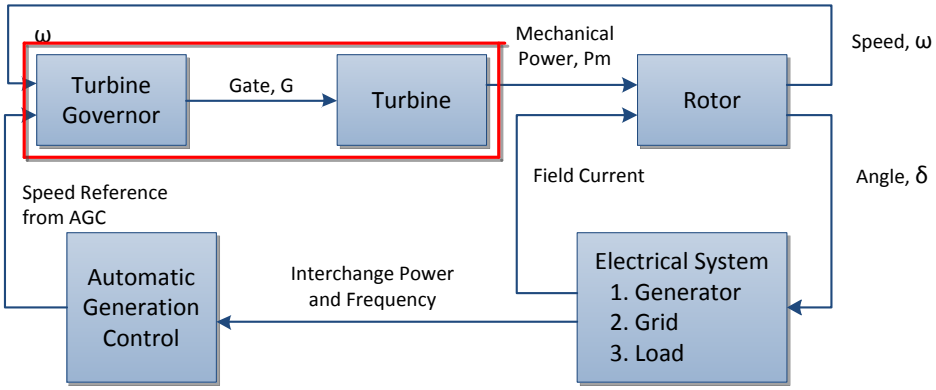


Figure 2.1.: Functional block diagram showing relationship of the turbine and governor with the overall power system

2.1.1. Hydro Turbine

As shown in the hydro turbine schematic diagram Fig. 2.2, the water flows from the higher reservoir to the lower level turbine through a penstock. At steady state, the water speeds and

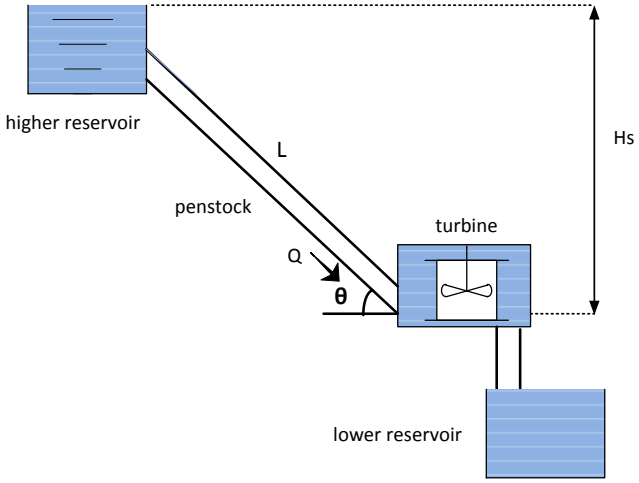


Figure 2.2.: Schematic diagram of hydro turbine

water pressures inside the penstock at various points are constant [1]. Assuming the flow is incompressible, the net force on the water in the penstock equals to the rate of momentum change of the water according to Newton's Second Law:

$$\begin{aligned} F_{net} &= Ma = \rho\, LAa = \rho\, LA\frac{dv}{dt} \\ F_{net} &= \rho\, L\frac{dQ}{dt} \end{aligned} \tag{2.1}$$

where M is the water's mass in the penstock, a the rate of velocity change (acceleration), ρ the mass density of water, L the length of penstock, A the cross-sectional area of the penstock, and Q the volumetric flow rate.

On the other hand, the net force on the water in the penstock can be obtained by considering the pressure head [1]:

$$\begin{aligned} F_{net} &= F_s - F_l - F = \rho\, LAa - F_l - F \\ &= \rho\, \frac{H_s}{\sin\theta} Ag\sin\theta - F_l - F = \rho\, H_s Ag - F_l - F \\ F_{net} &= \rho\, (H_s - H_l - H)Ag \end{aligned} \tag{2.2}$$

where F_s is the force on the water at the entry of the penstock, F_l the force of friction effect in the penstock, F the force on the water at the gate position of turbine. For convenience of representation, F_l and F are converted to H_l and H with the same proportion. H_s, H_l and H are static head, head loss, and head at turbine gate, respectively. Combining the equations (2.1) and (2.2), the rate of change of the volumetric flow rate can be described as:

$$\frac{dQ}{dt} = \frac{(H_s - H_l - H)Ag}{L} \tag{2.3}$$

We make use of h_{base} and q_{base} as common bases to normalize the above equation. h_{base}

stands for static pressure head at the top of water surface H_{base}, while q_{base} represents the flow rate through the turbine with the gate fully open at the head h_{base}. In this case, Q=$q_{base}q$, H_l=$h_{base}h_l$, H=$h_{base}h$. Equation (2.3) is normalized into

$$\frac{dq}{dt} = \frac{(1 - h_l - h)Agh_{base}}{Lq_{base}} = \frac{(1 - h_l - h)}{T_w} \tag{2.4}$$

where T_w=$\dfrac{Lq_{base}}{Agh_{base}}$ is the water starting time or water time constant. T_w defines the period it takes for water at the head h_{base} to obtain the flow rate of q_{base}. h_l is proportional to the flow rate square and penstock friction factor, as described by $h_l = k_f q^2$. The pressure head across the turbine is related to the flowrate as $h = \dfrac{q^2}{G^2}$, where G describes the gate position from 0 (closed) to 1 (fully open).

The mechanical power developed by the hydro turbine is proportional to pressure head and flow rate. As there is no-load flow q_{nl}, it should be subtracted from the actual flow rate,

$$P_m = A_t h(q - q_{nl}) \tag{2.5}$$

where A_t accounts for the difference in per units for both sides of the equation. The suffix 'r' indicates the parameter's value at rated load.

$$A_t = \frac{\text{turbine power (MW)}}{\text{generator rating (MVA)}} * \frac{1}{h_r(q_r - q_{nl})} \tag{2.6}$$

If damping effects are also taken into account, then

$$P_m = A_t h(q - q_{nl}) - DG\Delta\omega. \tag{2.7}$$

The nonlinear turbine model encompassing equations (2.1)—(2.7) is shown in Fig. 2.3.

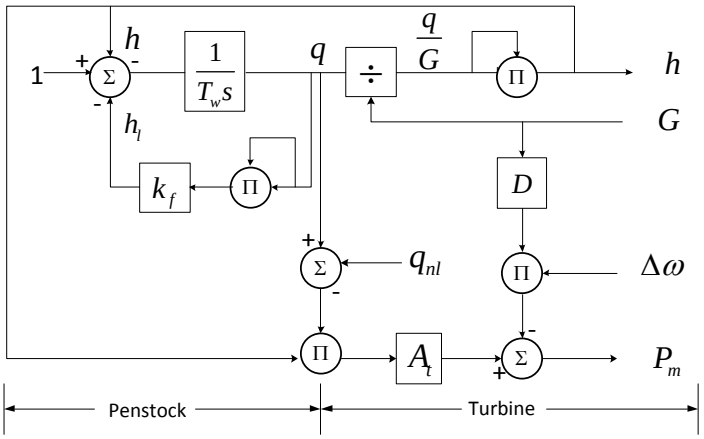


Figure 2.3.: Nonlinear turbine model (Adapted from [1])

Linearizing the turbine characteristics around an operating point can aid in simplifying the turbine model. In hydro-turbine systems, the mechanical torque increment Δm and the water

flow increment Δq are dependent on the gate opening increment ΔG, the rotational speed increment Δn, and the head increment Δh. The following linearized equations are valid to represent the turbine for small perturbations around a steady state condition.

$$\Delta q = a_{11}\Delta h + a_{12}\Delta n + a_{13}\Delta G \tag{2.8}$$

$$\Delta m = a_{21}\Delta h + a_{22}\Delta n + a_{23}\Delta G \tag{2.9}$$

where

$$a_{11} = \frac{\partial q}{\partial h}, a_{12} = \frac{\partial q}{\partial n}, a_{13} = \frac{\partial q}{\partial G}$$
$$a_{21} = \frac{\partial m}{\partial h}, a_{22} = \frac{\partial m}{\partial n}, a_{23} = \frac{\partial m}{\partial G}.$$

As mechanical power is produced by turbine, which drives machine to generate electricity, turbine dynamics has a relationship with machine dynamics.

$$P_m = \frac{n}{\omega} m \tag{2.10}$$

where P_m represents mechanical power, n rotational speed of machine, ω speed of turbine and m mechanical torque. Generally, n is assumed equal to ω at synchronous speed. As a consequence, $P_m = m$. Thus, the equations (2.8), (2.9) can transfer into

$$\Delta q = a_{11}\Delta h + a_{12}\Delta\omega + a_{13}\Delta G \tag{2.11}$$

$$\Delta P_m = a_{21}\Delta h + a_{22}\Delta\omega + a_{23}\Delta G. \tag{2.12}$$

When the rotational speed deviation is small ($\Delta\omega \approx 0$), the turbine transfer function which relates mechanical power increment in attaining gate position increment is

$$\frac{\Delta P_m}{\Delta G} = \frac{a_{23} + (a_{11}a_{23} - a_{13}a_{21})T_w s}{1 + a_{11}T_w s} \tag{2.13}$$

where ΔP_m=$P_m - P_{m0}$, and ΔG=$G - G_0$. Normally, we assume

$$\frac{P_{m0}}{G_0} = \frac{a_{23} + (a_{11}a_{23} - a_{13}a_{21})T_w s}{1 + a_{11}T_w s} \tag{2.14}$$

As a consequence,

$$\frac{P_m}{G} = \frac{\Delta P_m}{\Delta G} = \frac{a_{23} + (a_{11}a_{23} - a_{13}a_{21})T_w s}{1 + a_{11}T_w s} \tag{2.15}$$

and the corresponding transfer function block diagram is shown in Fig. 2.4.

Typical values for a turbine at full load are [9]

$$a_{11} = 0.58, a_{12} = 0, a_{13} = 1.10$$
$$a_{21} = 1.40, a_{22} = 0, a_{23} = 1.5.$$

For an ideal lossless turbine,

$$a_{11} = 0.5, a_{12} = 0, a_{13} = 1$$
$$a_{21} = 1.5, a_{22} = 0, a_{23} = 1.$$

Substituting these parameters in (2.15), we have

$$\frac{P_m}{G} = \frac{\Delta P_m}{\Delta G} = \frac{1 - T_w s}{1 + 0.5T_w s}. \tag{2.16}$$

This is the classical hydro turbine model in power system stability analysis, corresponding to ideal (lossless) turbine and inelastic (stiff) penstock with water inertial effect considered.

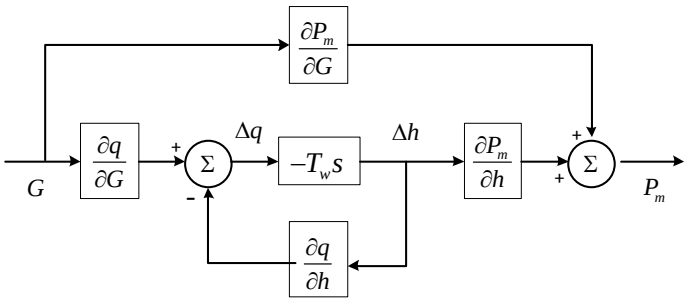


Figure 2.4.: Transfer Function Block Diagram of the Linearized Turbine Model

2.1.2. Hydro Turbine Governor

A simplified schematic of a mechanical hydraulic governor is shown in Fig. 2.5. In steady state, the shaft speed ω is compared to the reference speed ω_{ref}, and is modified by the permanent droop compensation $\sigma\Delta G$. When the gate position is changing, a transient droop compensation is developed to oppose fast changes in the gate position. In the mechanical hydraulic governor, these signals are transmitted from mechanical motion to the operation of the pilot valve through the floating levers system.

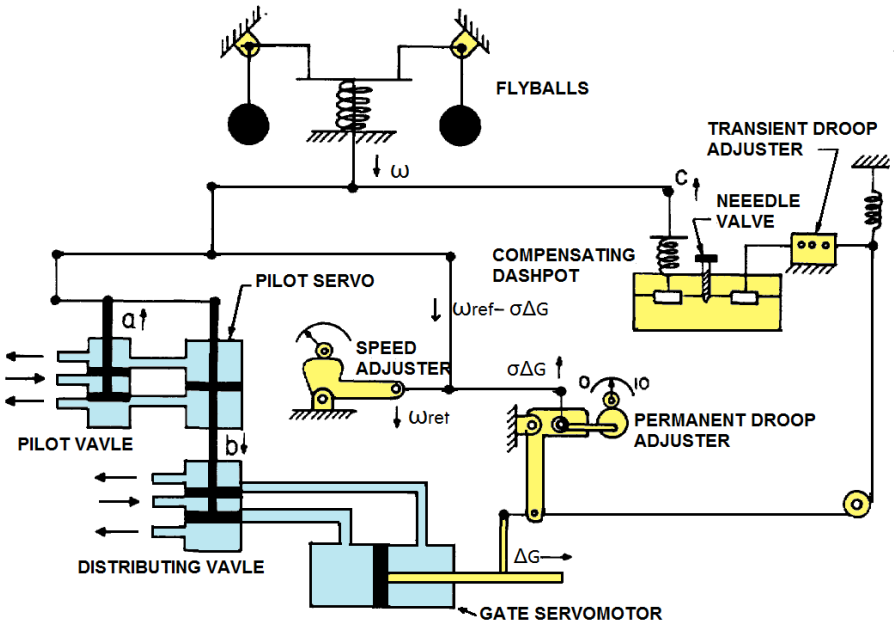


Figure 2.5.: A Simplified Schematic of a Mechanical Hydraulic Governor (Adapted from [9])

According to the schematic, the derived transfer function of the pilot valve a and pilot servo b is

$$\frac{b}{a} = \frac{K_1}{1 + T_p s} \tag{2.17}$$

where K_1 is determined by the feedback lever ratio, and T_p by the pilot valve port areas and K_1. The transfer function between the disturbing valve b and gate position change ΔG is

$$\frac{\Delta G}{b} = \frac{K_2}{s}. \tag{2.18}$$

Combining (2.17) and (2.18), we get

$$\frac{\Delta G}{a} = \frac{K_1 K_2}{(T_p s + 1)s} = \frac{1}{T_g} \frac{1}{(T_p s + 1)s} \tag{2.19}$$

where $\frac{1}{T_g} = K_1 K_2$. T_g is equal to the time in seconds for a 1 $p.u.$ change in frequency to produce a 1 $p.u.$ change in gate position. Assuming the flow of dashpot fluid through the needle valve is proportional to the dashpot pressure, the compensating dashpot transfer function is

$$\frac{c}{\Delta G} = \frac{\delta T_r s}{1 + T_r s} \tag{2.20}$$

where the temporary droop δ is determined by the selection of pilot point for the lever connected to the input piston. Through the action of a system's floating levers, the pilot valve input signal a is determined by the reference speed ω_{ref}, shaft speed ω, permanent droop $\sigma \Delta G$, and temporary droop signal c:

$$a = \omega_{ref} - \omega - \sigma \Delta G - c \tag{2.21}$$

The relationship between the speed difference $\omega_{ref} - \omega$ and the gate position change ΔG can be described in Fig. 2.6. The rate limit may occur for large, rapid speed deviation, and the position limit may correspond to wide open valves or the setting of a load limiter.

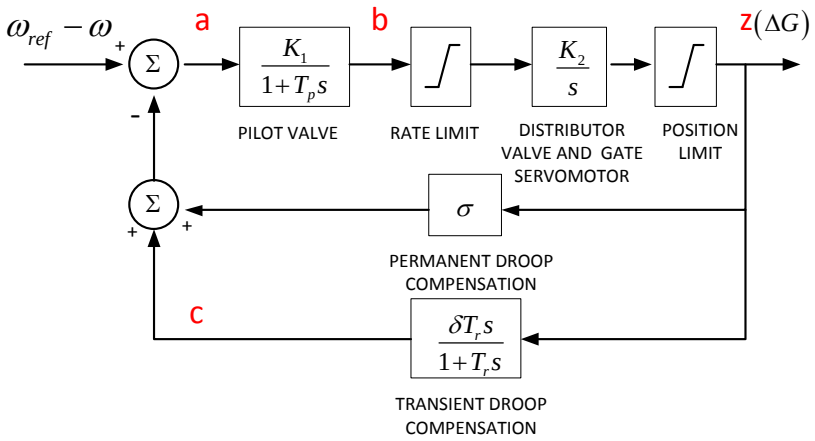


Figure 2.6.: Transfer Function Block Diagram of a Typical Hydro Turbine Governor Model

Electro hydraulic governor is more widely used in modern hydro speed governors. The dynamic behavior, structure, and operation are essentially similar to that of the mechanical hydraulic governor except that:

- Speed sensing, permanent droop, temporary droop, and other measuring and computer functions are performed electrically.

- The electric components provide more flexibility and better performance.

Three-terms controller with proportion-integral-derivative(PID) action is often implemented in electro hydraulic governors [6]. It calculates the "error" values between the measured process variable and a desired set point of $\Delta\omega$. The controller attempts to minimize the error by adjusting the process control input.

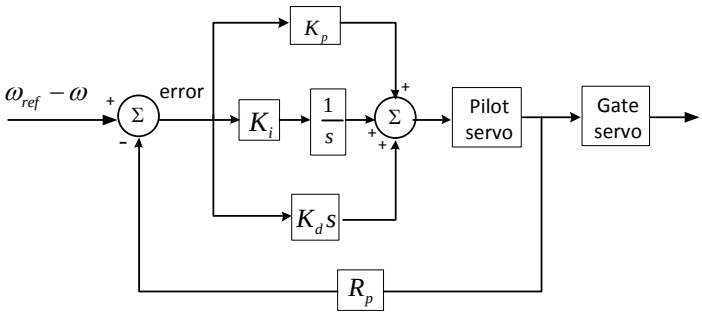


Figure 2.7.: PID Controller (Adapted from [6])

The proportional term controls the present state by means of multiplying the error with a negative constant K_p and adding to the desired set point. The integral term controls the past state. The sum of the instantaneous error over time multiplies with a negative constant K_i, and adds to the desired set point, which eliminates the error between the process variable and the desired set point. As a result, the PID system can reach the steady state of the desired set point.

The derivative term controls the future state by multiplying the error with a negative constant K_d and adding it to the desired set point. The derivative controller responds to system changes; the larger the derivative value is, the faster the response becomes. The derivative action is beneficial for isolated operation, particularly for plants with larger water starting time ($T_w = 3s$ or more) [6]. However, higher derivative gain may result in excessive oscillations and governor loop instability. This is the reason for the minimum limit imposed on the value of K_p/K_d or directly setting K_d to zero. The following formulas from reference [5] illustrate how to choose suitable parameters for PID controller; typical values are $K_p = 3.0$, $K_i = 0.7$, $K_p = 0.5$ [6].

$$\frac{1}{K_p} = \frac{0.625T_w}{H}$$

$$\frac{K_p}{K_i} = 3.33T_w$$

$$\frac{K_p}{K_d} > 3T_w$$

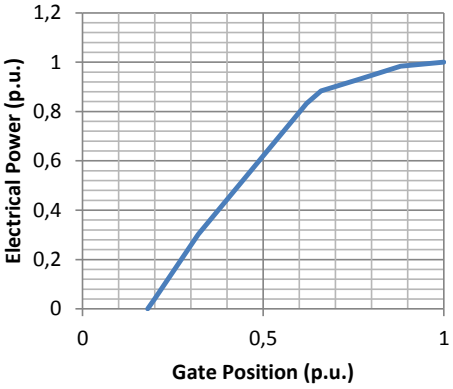


Figure 2.8.: St.Lawrence Unit 32, Electrical Power versus Gate Position from Tests [19]

2.2. Hydro Turbine and Governor Models

The hydro turbine and its governor are normally combined together for representation. However sometimes the output of turbine governor is the derivative of gate position, which does not match the input of the turbine. Therefore, a desired gate position reference is applied to add to the gate position derivative.

Figure 2.8 reveals a nonlinear relationship between electrical power and gate position over the entire range of gate position. This data is obtained from earlier tests and can be applied in models by adding a corresponding nonlinear function between the governor model and turbine model. But this nonlinear relationship varies with different types of turbines. What's more, the nonlinear feature mainly arises when the gate position is relatively small or large. In other words, usually, electrical power keeps a good linear relationship with gate position in the middle range (around 20% to 80%), which is the main working range of a turbine. In this case, this thesis does not account the nonlinear relationship so the power reference can be regarded as a gate position reference.

The following section presents four developed hydro turbine and governor models. Since PAST already has two turbine and governor models, shown in Fig.2.9 Model 1 (thermal turbine and governors) and Fig.2.10 Model 2, the new implemented models will be ordered sequentially afterwards. As shown in Table 2.1, the first new model, namely, Model 3, consists the typical hydro turbine and governor model. Model 4 contains a simple PI controller in front of Model 3. A nonlinear turbine model is used in Model 5, and the nonlinear hydro turbine governor model (HTG) available in SPS and implemented in PSAT, is called Model 6.

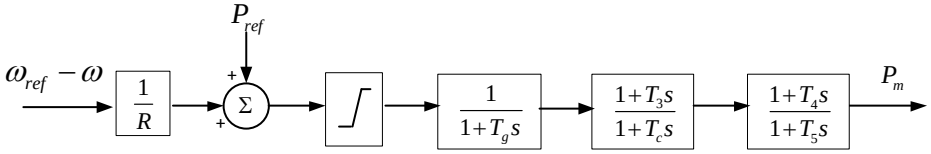


Figure 2.9.: Turbine and Governor Model 1 implemented in PSAT

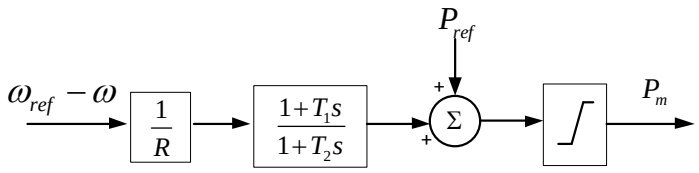


Figure 2.10.: Turbine and Governor Model 2 implemented in PSAT

Table 2.1.: Summary of governor models and turbine models combinations

Model	Turbine governor model selection	Turbine model selection
Model 3	Typical model	Linear model
Model 4	Typical mode with PI controller in front	Linear model
Model 5	PI controller combined with servomotor	Nonlinear model
Model 6	PID controller combined with servomotor	Nonlinear model

A general device model can be described by a set of differential algebraic equations (DAE) as:

$$\dot{\mathbf{x}}_\mathbf{i} = \mathbf{f}_\mathbf{i}(\mathbf{x}_\mathbf{i}, \mathbf{y}_\mathbf{i}, \mathbf{x}_\mathbf{e}, \mathbf{y}_\mathbf{e}, \mathbf{p}_\mathbf{i}, \mathbf{u}_\mathbf{i}); \tag{2.22}$$

$$\mathbf{0} = \mathbf{g}_\mathbf{i}(\mathbf{x}_\mathbf{i}, \mathbf{y}_\mathbf{i}, \mathbf{x}_\mathbf{e}, \mathbf{y}_\mathbf{e}, \mathbf{p}_\mathbf{i}, \mathbf{u}_\mathbf{i})$$

where the sub-index i indicates specific device internal variables and the sub-index e indicates external variables from other devices; $\mathbf{f}$ are the differential equations, $\mathbf{g}$ are the algebraic equations, $\mathbf{x}$ are the state variables, $\mathbf{y}$ are the algebraic variables, $\mathbf{p}$ are device parameters assigned in the data file, and $\mathbf{u}$ are controllable variables (such as reference signals in control loops) [2].

For the turbine and governor model, equation (2.22) can be simplified:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{y}, \mathbf{u}); \tag{2.23}$$

$$\mathbf{0} = \mathbf{g}(\mathbf{x}, \mathbf{y}, \mathbf{u})$$

where the most important state variables $\mathbf{x}$ are generator rotor angles δ and generator rotor speeds ω. The vector of algebraic variables $\mathbf{y}$ includes system variables (e.g. bus voltage magnitudes), internal variables (e.g. internal input and output signals), and output variables (e.g. mechanical torques), and $\mathbf{u}$ are controllable and/or specified signals (e.g.voltage or speed references). The DAEs are provided for each model in next sections.

2.2.1. Model 3

The structural diagram of Model 3 is shown in Fig.2.11. It consists of a typical hydro turbine governor model and a linearized hydro turbine model. The output of turbine governor is the gate position derivative (ΔG), while the input of the turbine is the gate position (G). Consequently, a position reference G_{ref}, which is regarded as equal to P_{ref}, is required between them.

To implement models in computer software and determine its future behavior, normally, a set of state variables that consists of coupled first-order differential equations are necessary

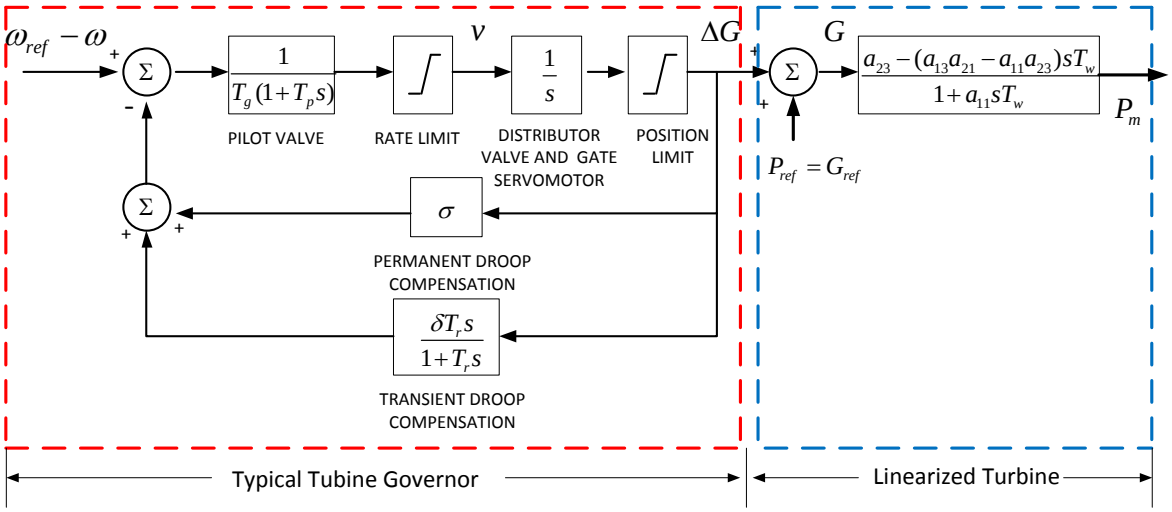


Figure 2.11.: Diagram showing the general structure of Model 3

for a dynamic system. However, it is hard to figure out the state variables in the structural diagram. The solution is redrawing the models only by integrators and gain blocks. In this way, along the signal flow each state variable is located behind each integrator, while the derivative of state variable in front, as shown in Fig.2.12. The number of state variables is equal to the total number of integrators in a system.

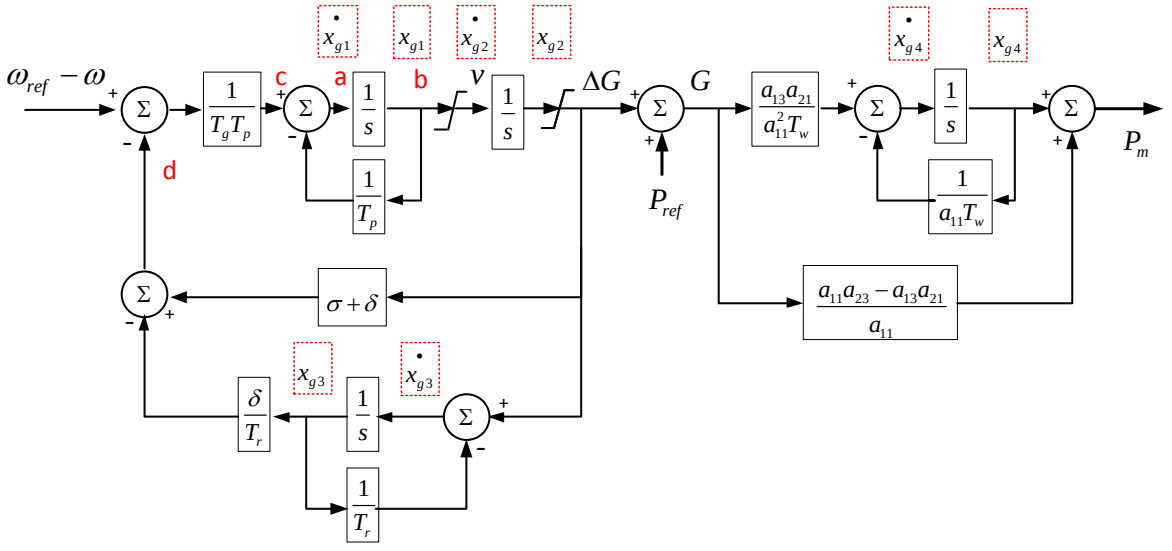


Figure 2.12.: Block Diagram Realization of Model 3

From the block diagram realization in Fig.2.12, we can derive a differential algebraic equation (DAE) set, which is utilized for implementing the model in PSAT and calculating integrators'

initial values. The derived DAEs are

$$
\begin{aligned}
\dot{x}_{g1} &= \frac{1}{T_g T_p}[(\omega_{ref} - \omega) - (\sigma + \delta)\Delta G + \frac{\delta}{T_r} x_{g3}] - \frac{1}{T_p} x_{g1} \\
\dot{x}_{g2} = v &= \begin{cases} x_{g1} & \text{if } v_{max} \geq x_{g1} \geq v_{min} \\ v_{max} & \text{if } v_{max} < x_{g1} \\ v_{min} & \text{if } v_{min} > x_{g1} \end{cases} \\
\dot{x}_{g3} &= \Delta G - \frac{1}{T_r} x_{g3} \\
\dot{x}_{g4} &= \frac{a_{13} a_{21}}{{a_{11}}^2 T_w}(\Delta G + P_{ref}) - \frac{1}{a_{11} T_w} x_{g4} \\
\Delta G &= \begin{cases} x_{g2} & \text{if } G_{max} - P_{ref} \geq x_{g2} \geq G_{min} - P_{ref} \\ G_{max} - P_{ref} & \text{if } G_{max} - P_{ref} < x_{g2} \\ G_{min} - P_{ref} & \text{if } G_{min} - P_{ref} > x_{g2} \end{cases} \\
P_m &= x_{g4} + \frac{a_{11} a_{23} - a_{13} a_{21}}{a_{11}}(\Delta G + P_{ref}).
\end{aligned}
$$

When the system is in steady state, $\omega = \omega_{ref}$, the rate of the gate movement $v = 0$, and the gate is fixed as $\Delta G = 0$. The initial values for Model 3 can be obtained by setting the derivatives variables in differential equations to zero.

$$
\begin{aligned}
x_{g1} &= x_{g2} = x_{g3} = 0 \\
x_{g4} &= \frac{a_{13} a_{21}}{a_{11}} P_{ref} \\
P_m &= a_{23} P_{ref}.
\end{aligned}
$$

This initial values can also be determined by analyzing Fig.2.12. When the system is in steady state, $\omega = \omega_{ref}$, all the derivatives of all state variables are zero. Consequently, as we can see, point a and b are zero. Because $c - b = a$, c is accordingly equal to zero. Similarly, point d is also zero as $(\omega_{ref} - \omega)$ and c are zero. With this recurrence method, we can get all the state variables' initial values that are as same as DAEs get and this method can be used to check if the DAE calculation is correct.

2.2.2. Model 4

As shown in Fig.2.13, the difference between Model 4 to Model 3 is a simple PI controller added in the front of the turbine governor. The integrator in the PI controller computes the integral of the error between ω_{ref} and ω, which affects the input of turbine governor model. In this case the error is no longer zero in steady state, and the output becomes the gate position itself, which is also the input of the turbine model.

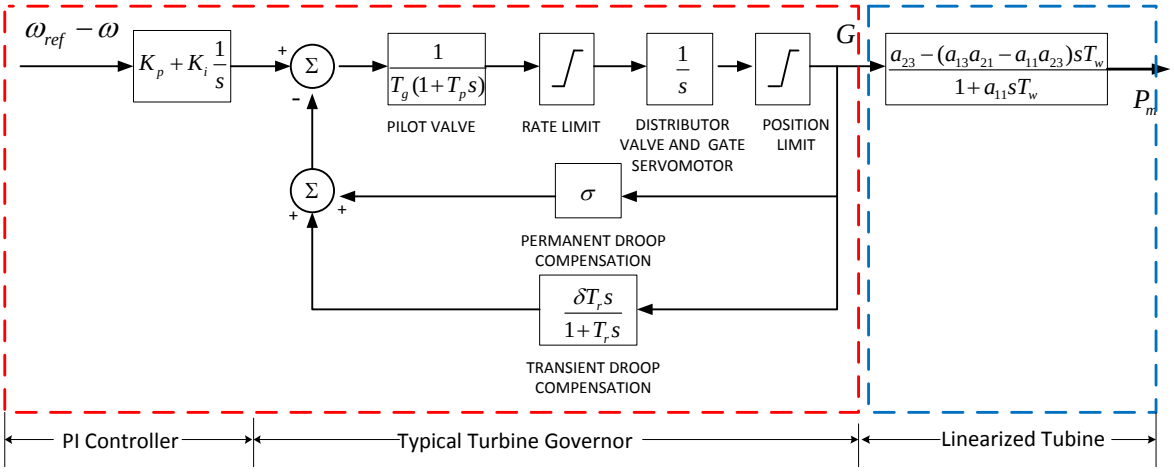


Figure 2.13.: Diagram Showing the General Structure of Model 4

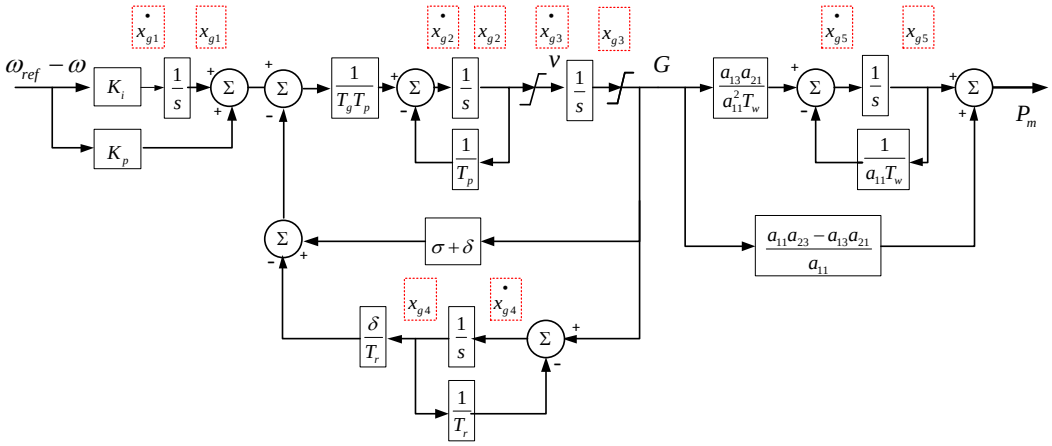


Figure 2.14.: Block Diagram Realization of Model 4

The diagram in Fig.2.14 can be described by DAEs as

$$\dot{x}_{g1} = K_i(\omega_{ref} - \omega)$$

$$\dot{x}_{g2} = \frac{1}{T_g T_p}[x_{g1} + K_p(\omega_{ref} - \omega) - (\sigma + \delta)G + \frac{\delta}{T_r}x_{g4}] - \frac{1}{T_p}x_{g2}$$

$$\dot{x}_{g3} = v = \begin{cases} x_{g2} & \text{if } v_{max} \geq x_{g2} \geq v_{min} \\ v_{max} & \text{if } v_{max} < x_{g2} \\ v_{min} & \text{if } v_{min} > x_{g2} \end{cases}$$

$$\dot{x}_{g4} = G - \frac{1}{T_r}x_{g4}$$

$$\dot{x}_{g5} = \frac{a_{13}a_{21}}{a_{11}{}^2 T_w}G - \frac{1}{a_{11}T_w}x_{g5}$$

$$G = \begin{cases} x_{g3} & \text{if } G_{max} \geq x_{g3} \geq G_{min} \\ G_{max} & \text{if } G_{max} < x_{g3} \\ G_{min} & \text{if } G_{min} > x_{g3} \end{cases}$$

$$P_m = x_{g5} + \frac{a_{11}a_{23} - a_{13}a_{21}}{a_{11}}G.$$

The calculation results for the integrators initial values are

$$
\begin{aligned}
G &= P_{ref} \\
x_{g1} &= \sigma P_{ref} \\
x_{g2} &= 0 \\
x_{g3} &= P_{ref} \\
x_{g4} &= T_r P_{ref} \\
x_{g5} &= \frac{a_{13}a_{21}}{a_{11}} P_{ref}.
\end{aligned}
$$

$$P_m = P_{ref}$$

2.2.3. Model 5

Model 5 is taken from [10], in which the main difference to the previous models is that a nonlinear turbine model is used. For nonlinear turbine models, when calculating Jacobian matrices, the partial derivatives of differential function f with respect to state variables x still contain state variables, which means that the partial derivatives changes with time. This translates to different performances when compared with linear models. In Chapter 3, there is a detailed introduction on how to calculate Jacobian Matrices and implement them in PSAT.

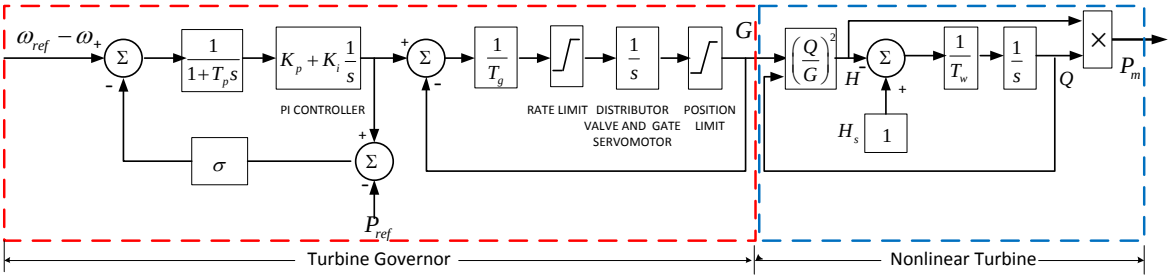


Figure 2.15.: Diagram Showing the General Structure of Model 5

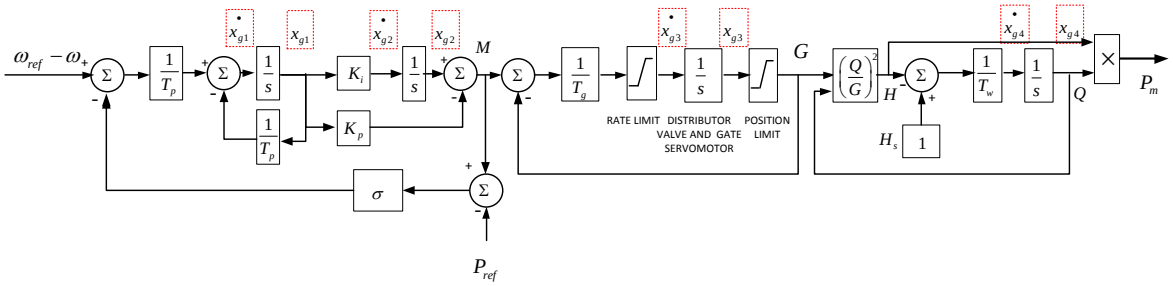


Figure 2.16.: Block Diagram Realization of Model 5

The block diagram realization in Fig.2.16 can be described by DAEs as

$$
\begin{aligned}
\dot{x}_{g1} &= \frac{1}{T_p}[(\omega_{ref} - \omega) + (\sigma K_p - 1)x_{g1} - \sigma x_{g2} + \sigma P_{ref}] \\
\dot{x}_{g2} &= K_i x_{g1} \\
\dot{x}_{g3} = v &= \begin{cases} \frac{1}{T_g}(-K_p x_{g1} + x_{g2} - x_{g3}) & \text{if } v_{max} \geq \frac{1}{T_g}(-K_p x_{g1} + x_{g2} - G) \geq v_{min} \\ v_{max} & \text{if } v_{max} < \frac{1}{T_g}(-K_p x_{g1} + x_{g2} - G) \\ v_{min} & \text{if } v_{min} > \frac{1}{T_g}(-K_p x_{g1} + x_{g2} - G) \end{cases} \\
\dot{x}_{g4} &= \frac{1}{T_w}[1 - (\frac{x_{g4}}{G})^2] \\
G &= \begin{cases} x_{g3} & \text{if } G_{max} \geq x_{g3} \geq G_{min} \\ G_{max} & \text{if } G_{max} < x_{g3} \\ G_{min} & \text{if } G_{min} > x_{g3} \end{cases} \\
P_m &= x_{g4}(\frac{x_{g4}}{G})^2.
\end{aligned}
$$

When the system is in steady state, $\omega = \omega_{ref}$, all differential variables are equal to zero, thus M and G are equal to P_{ref}. The calculation results for integrators' initial values are

$$
\begin{aligned}
G &= P_{ref} \\
x_{g1} &= 0 \\
x_{g2} &= P_{ref} \\
x_{g3} &= P_{ref} \\
x_{g4} &= P_{ref} \\
P_m &= P_{ref}.
\end{aligned}
$$

2.2.4. Model 6

The hydro turbine and governor model in SPS is encapsulated into one block named HTG, which contains a nonlinear hydro turbine model, a PID governor system, and a servomotor as shown in Fig.2.17. Normally, the switch chooses input 3 ($P_e - P_{ref}$) because the input 2 (d_{ref}) is below the threshold (default value is 0.5). An additional distinction is that this model makes use of $\dot{\omega}$ as another input to the turbine, which can accelerate the system reaction when subject to a large transient movement. The turbine and governor model can be transfered into the diagram shown in Fig.2.18.

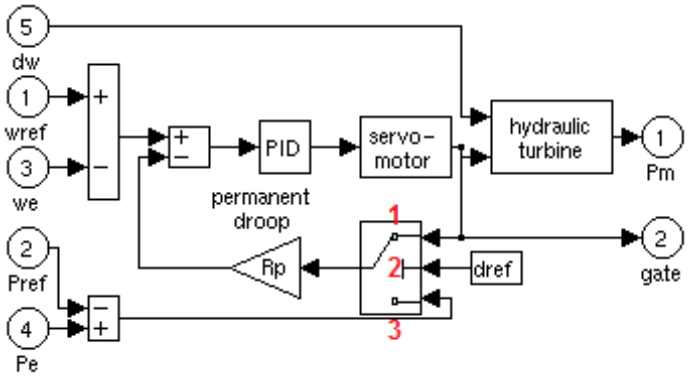


Figure 2.17.: Block Diagram Showing Model 6 (Taken from [11])

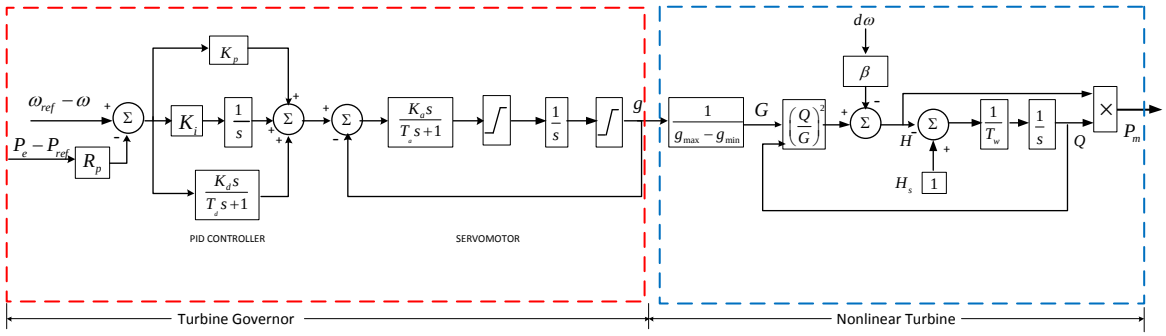


Figure 2.18.: Block Diagram Showing the General Structure of Model 6

The diagram in Fig.2.19 can be described by DAEs as

$$\dot{x}_{g1} = K_i[\omega_{ref} - \omega - R_p(P_e - P_{ref})]$$

$$\dot{x}_{g2} = \frac{K_d}{T_d^2}[\omega_{ref} - \omega - R_p(P_e - P_{ref})] - \frac{1}{T_d}x_{g2}$$

$$\dot{x}_{g3} = \frac{K_a}{T_a}\{x_{g1} - x_{g2} + (K_p + \frac{K_d}{T_d})[\omega_{ref} - \omega - R_p(P_e - P_{ref})]\} - \frac{1}{T_a}x_{g3}$$

$$\dot{x}_{g4} = v = \begin{cases} x_{g3} & \text{if } v_{max} \geq x_{g3} \geq v_{min} \\ v_{max} & \text{if } v_{max} < x_{g3} \\ v_{min} & \text{if } v_{min} > x_{g3} \end{cases}$$

$$g = \begin{cases} x_{g4} & \text{if } G_{max} * (g_{max} - g_{min}) \geq x_{g4} \geq G_{min} * (g_{max} - g_{min}) \\ g_{max} & \text{if } G_{max} * (g_{max} - g_{min}) < x_{g4} \\ g_{min} & \text{if } G_{min} * (g_{max} - g_{min}) > x_{g4} \end{cases}$$

$$G = \frac{g}{g_{max} - g_{min}}$$

$$\dot{x}_{g5} = \frac{1}{T_w}[1 - (\frac{x_{g5}}{G})^2 + \beta d\omega]$$

$$P_m = x_{g5}[(\frac{x_{g5}}{G})^2 - \beta d\omega].$$

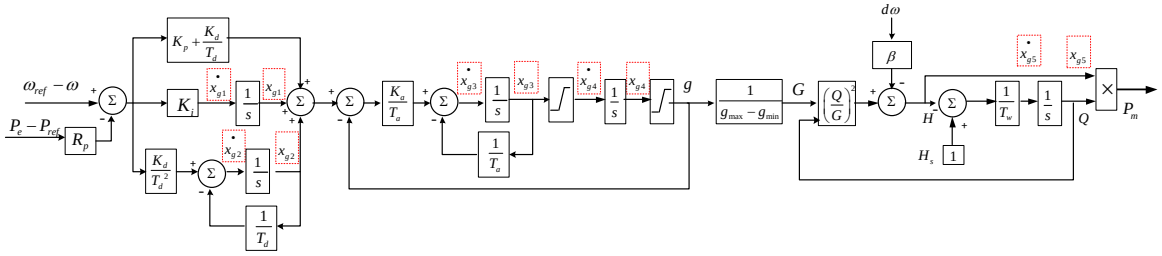


Figure 2.19.: Block Diagram Realization of Model 6

The integrators' initial values are calculated from

$$
\begin{aligned}
G &= P_{ref} \\
x_{g1} &= x_{g2} = x_{g3} = 0 \\
x_{g4} &= g = (g_{max} - g_{min}) * P_{ref} \\
x_{g5} &= P_{ref} \\
P_m &= P_{ref}.
\end{aligned}
$$

2.3. Model Parameters

Parameters in $p.u.$ for four models are presented below, these parameters were obtained for Model 3 and Model 4 from [4], Model 5 from [10] and Model 6 from [11].

Table 2.2.: Parameters for Model 3 to Model 6

Parameters	Model 3 ($p.u.$)	Model 4 ($p.u.$)	Model 5 ($p.u.$)	Model 6 ($p.u.$)
ω_{ref}	1	1	1	1
T_g	0.2	0.2	0.2	
T_p	0.04	0.04	0.05	
G_{max}	1.0	1.0	1.0	0.97518
G_{min}	0	0	0	0.01
v_{max}	0.1	0.1	0.1	0.1
v_{min}	-0.1	-0.1	-0.1	-0.1
T_r	5.0	5.0		
T_w	1.0	1.0	1.0	2.67
σ	0.04	0.04	0.04	
δ	0.3	0.3		
a_{11}	0.5	0.5		
a_{13}	1.0	1.0		
a_{21}	1.5	1.5		
a_{23}	1.0	1.0		
K_p		1.163	3.0	1.163
K_i		0.105	0.5	0.105
K_a				10/3
T_a				0.07
β				0
R_p				0.04
K_d				0
T_d				0.01

3 Software Implementation

Now that the hydro turbine and governor models have been established, they now can be implemented, simulated and studied using different software. This thesis focuses on the implementation of two simulation software: Power System Analysis Toolbox (PSAT) — an open source software and SimPowerSystems (SPS) — a proprietary software. This chapter deals with how to implement the new turbine and governor models in PSAT and SPS.

3.1. PSAT Implementation

3.1.1. General Introduction about PSAT

PSAT is an open source MATLAB and GNU/Octave-based software package for analysis and design of small to medium size electric power systems. It is developed by Federico Milano and currently used in more than 50 countries. PSAT is a very flexible and modular tool for power flow (PF), continuation power flow (CPF), optimal power flow (OPF), small signal stability analysis (SSSA) and time domain simulation. Additionally, a variety of static and dynamic models are provided. Both graphic user interface (GUI) and command line execution can be utilized for calculations and simulations. Since MATLAB is a proprietary software, PSAT can also run on the latest GNU/Octave release to realize its full free software potential [2], [8], [7], [12]. The greatest advantage of PSAT is that it is free and open source, allowing the user to thoroughly know about components' inside structures and power flow calculations, even to develop their own models.

Table 3.1 depicts several Matlab-based commercial, research and educational power system

Table 3.1.: MATLAB-based packages for power system analysis (taken from [2])

Package	PF	CPF	OPF	SSSA	TDS	EMT	GUI	CAD
EST	√			√	√			√
MatEMTP					√	√	√	√
MATPOWER	√		√					
PAT	√			√	√			√
PSAT	√	√	√	√	√		√	√
PST	√	√		√	√			
SPS	√			√	√	√	√	√
VST	√	√		√	√		√	

tools such as Power System Toolbox (PST), MatPower, Toolbox(VST), MatEMTP, SimPowerSystems (SPS), Power Analysis Toolbox (PAT), and the Educational Simulation Tool (EST). Among these, only MatPower, PSAT and VST are open source and freely distributed.

The core of PAST is its power flow algorithm, which contains the initialization of state variables . After completing power flow, the following routines can be executed for further static and dynamic analyses[2].

- Continuation power flow (CPF);
- Optimal power flow (OPF);
- Small signal stability analysis (SSSA);
- Time domain simulations (TDS);
- Phaser measurement unit (PMU) placement.

PSAT supports a variety of static and dynamic component models, as follows [2]:

- *Power Flow Data*: Bus bars, transmission lines and transformers, slack buses, PV generators, constant power loads, and shunt admittances.
- *CPF and OPF Data*: Power supply bids and limits, generator power reserves and ramp data, and power demand bids, limits and ramp data.
- *Switching Operations*: Transmission line faults and transmission line breakers.
- *Measurements*: Bus frequency and phasor measurement units (PMU).
- *Loads*: Voltage dependent loads, frequency dependent loads, ZIP (impedance, constant current and constant power) loads, exponential recovery loads, voltage dependent loads with embedded dynamic tap changers, thermostatically controlled loads, Jimma's loads, and mixed loads.
- *Machines*: Synchronous machines (dynamic order from 2 to 8) and induction motors (dynamic order from 1 to 5).
- *Controls*: Turbine Governors, Automatic Voltage Regulators, Power System Stabilizer, Over-excitation limiters, Secondary Voltage Regulation (Central Area Controllers and Cluster Controllers), and a Supplementary Stabilizing Control Loop for SVCs.
- *Regulating Transformers*: Load tap changer with voltage or reactive power regulators and phase shifting transformers.
- *FACTS*: Static Var Compensators, Thyristor Controlled Series Capacitors, Static Synchronous Source Series Compensators, Unified Power Flow Controllers, and High Voltage DC transmission systems.
- *Wind Turbines*: Wind models, Constant speed wind turbine with squirrel cage induction motor, variable speed wind turbine with doubly fed induction generator, and variable speed wind turbine with direct drive synchronous generator.

- *Other Models*: Synchronous machine dynamic shaft, sub-synchronous resonance (SSR) model, and Solid Oxide Fuel Cell.

Besides mathematical algorithms and models, PSAT includes the following utilities (taken from [2]):

- One-line network diagram editor (Simulink library);
- GUIs for settings system and routine parameters;
- GUI for plotting results;
- Filters for converting data to and from other formats;
- Data file conversion to and from other formats;
- User defined model editor and installer;
- Command logs.

PSAT can be freely downloaded from [26] and this thesis makes use of Version 2.1.6 for PSAT developing. Download this version into your computer and add the "psat_Matlab" folder as a MATLAB path. Then you could start to use PSAT by GUI or command logs.

3.1.2. Hydro Turbine and Governor Models Implementations

PSAT is written by using classes and object-oriented programming techniques. Each device is defined by a class with attributes and methods [2]. The advantages of using classes include: easing the maintenance of the code, facilitating modularity and reusing well assessed code. Moreover, the class modularity allows one to quickly master the code.

Model Properties

Common properties for all devices are (taken from [2]):

1. con: device data in the form of a matrix. Each row defines a new instance of the device, while each column defines a parameter. The format of the matrix con is described in detail for each device.
2. format: a string containing the format of each row of the con matrix. Used for printing PSAT data to files.
3. n: total number of devices. This is also the amount of rows in con file.
4. ncol: maximum number of columns of the matrix con.
5. store: data backup. This is basically a copy of the con matrix. This attribute can be used for automatically modifying some input data and can be useful for command line usage.
6. u: vector containing the status of the devices. 1 means on-line, 0 means off-line. Some devices can lack this attribute (i.e. buses).

For the turbine and governor models, "***con***" is unique for each model. The TGcalss in PSAT already provides two turbine and governor models named Model 1 and Model 2. And the new ones Model 3 to Model 6 are presented in Table 3.2, 3.3, 3.4, 3.5.

Table 3.2.: Turbine and Governor Model 3 Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	3	Turbine governor type	int
3	ω_{ref}	Referenced speed	p.u.
4	T_g	Pilot valve droop	p.u.
5	G_{max}	Maximum gate opening	p.u.
6	G_{min}	Minimum gate opening	p.u.
7	v_{max}	Maximum gate opening rate	p.u.
8	v_{min}	Minimum gate opening rate	p.u.
9	T_p	Pilot valve time constant	s
10	T_r	Dashpot time constant	s
11	σ	Permanent speed droop	p.u.
12	δ	Transient speed droop	p.u.
13	T_w	Water starting time	s
14	a_{11}	$\partial q/\partial h$	p.u.
15	a_{13}	$\partial q/\partial g$	p.u.
16	a_{21}	$\partial m/\partial h$	p.u.
17	a_{23}	$\partial m/\partial g$	p.u.
18	u	Connection status	0,1

Table 3.3.: Turbine and Governor Model 4 Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	4	Turbine governor type	int
3	ω_{ref}	Referenced speed	p.u.
4	T_g	Pilot valve droop	p.u.
5	G_{max}	Maximum gate opening	p.u.
6	G_{min}	Minimum gate opening	p.u.
7	v_{max}	Maximum gate opening rate	p.u.
8	v_{min}	Minimum gate opening rate	p.u.
9	T_p	Pilot valve time constant	s
10	T_r	Dashpot time constant	s
11	σ	Permanent speed droop	p.u.
12	δ	Transient speed droop	p.u.
13	T_w	Water starting time	s
14	a_{11}	$\partial q/\partial h$	p.u.
15	a_{13}	$\partial q/\partial g$	p.u.
16	a_{21}	$\partial m/\partial h$	p.u.
17	a_{23}	$\partial m/\partial g$	p.u.
18	K_p	Proportional droop	p.u.
19	K_i	Integral droop	p.u.
20	u	Connection status	0,1

Table 3.4.: Turbine and Governor Model 5 Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	5	Turbine governor type	int
3	ω_{ref}	Referenced speed	p.u.
4	T_g	Servomotor droop	p.u.
5	G_{max}	Maximum gate opening	p.u.
6	G_{min}	Minimum gate opening	p.u.
7	v_{max}	Maximum gate opening rate	p.u.
8	v_{min}	Minimum gate opening rate	p.u.
9	T_p	Pilot valve time constant	s
10	T_w	Water starting time	s
11	σ	Permanent speed droop	p.u.
12	K_p	Proportional droop	p.u.
13	K_i	Integral droop	p.u.
14	u	Connection status	0,1

Table 3.5.: Turbine and Governor Model 6 Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	6	Turbine governor type	int
3	ω_{ref}	Referenced speed	p.u.
4	K_a	Servomotor gain	p.u.
5	G_{max}	Maximum gate opening	p.u.
6	G_{min}	Minimum gate opening	p.u.
7	v_{max}	Maximum gate opening rate	p.u.
8	v_{min}	Minimum gate opening rate	p.u.
9	T_a	Pilot valve time constant	s
10	T_w	Water starting time	s
11	$beta$	Transient speed droop	p.u.
12	K_p	Proportional droop	p.u.
13	K_i	Integral droop	p.u.
14	K_d	Derivative droop	p.u.
15	T_d	Derivative droop time constant	s
16	R_p	Permanent droop	p.u.
17	u	Connection status	0,1

Model Methods

Common methods for for all devices are (taken from [2]):

1. add: adds one or more instances of the device.
2. base: converts device parameters to system power and voltage bases.
3. block: defines special operations of the device mask (used only for Simulink models).
4. display: prints the class properties in a structure-like format.
5. dynidx: assigns indexes to the state variables of the device.
6. fcall: computes differential equations **f** of the device.
7. Fxcall: computes Jacobian matrices f_x, f_y and g_x of the device.
8. gcall: computes algebraic equations **g** of the device.
9. getxy: returns indexes of state and algebraic variables of the device.
10. Gycall: computes the Jacobian matrix g_y of the device.
11. init: cleans up all device properties.
12. mask: coordinates of the black mask (used in Simulink models and for drawing system).
13. remove: removes one or more instances of the device.
14. restore: restores device properties as given in the original data file.
15. setup: initializes the main device properties.
16. setx0: compute the initial value of state variables of the device after power flow analysis.
17. subsasgn: assigns device properties. Properties that are not listed in this function cannot be assigned from outside the class.
18. subsref: returns device properties. Properties that are not listed in this function cannot be get from outside the class.
19. XXclass: class constructor. XX is the device specific code.
20. xfirst: assigns an initialization value to state and algebraic variables of the device.
21. warn: prints some warning messages.
22. windup: applies the anti-windup limiter if necessary.

To add new turbine and governor models in the TGclass, the following files need some modification, in which some are particularly necessary only for turbine and governors, rather than common devices:

1. base: In PSAT, when defining the TG class data, the model droop (or gain) is given in p.u. with respect to the synchronous machine power rating. During initialization, the droop (or gain) is converted to the system power base in order to keep the same base with other variables in whole system, like electric and mechanical power.

   It is quite easy to see why this conversion is necessary if you think of two machines, called A and B. $S_A = 1$ MVA and $S_B = 100$ MVA are the power rates of machine A and B, respectively. When assuming the droops are $R_A = R_B$ on system basis, the two machines will vary the power production by the same amount following a frequency variation. Clearly, this is not what we want. What we want is, for example, that machine B provides 100 times the power variation provided by machine A. Hence, the droop of the two machines, must be $R_A = R_B$ on the machine base, and $R_A = 100\ R_B$ on the system base. Equation (3.1) and (3.2) show the droop and gain conversions, respectively.

$$\frac{1}{R_{system}} = \frac{S_{machine}}{S_{system}} * \frac{1}{R_{machine}} \tag{3.1}$$

   or

$$K_{system} = \frac{S_{machine}}{S_{system}} * K_{machine} \tag{3.2}$$

   The gain (droop) is defined as (the inverse of) the limit for $s \Rightarrow 0$ of F(s), where F(s) is the transfer function of hydro turbine governor. In this case, the first mission is to calculate the transfer function and figure out the gain (or droop). Sometimes, apart from gain (or droop), the mechanical power limits should be converted into system base in order to keep in same scale with mechanical power.

$$\begin{aligned} P_{max(system)} &= \frac{S_{machine}}{S_{system}} * P_{max(machine)} \\ P_{min(system)} &= \frac{S_{machine}}{S_{system}} * P_{min(machine)} \end{aligned} \tag{3.3}$$

   Model 1 and Model 2 —For Model 1 and Model 2, as the PSAT manual [2] indicated, the transfer functions of the governor are

$$F_1(s) = \frac{\Delta P_m}{\Delta\omega_{ref} - \Delta\omega} = \frac{1}{R} \tag{3.4}$$

$$F_2(s) = \frac{\Delta P_m}{\Delta\omega_{ref} - \Delta\omega} = \frac{1}{R} * \frac{T_1 s + 1}{T_2 s + 1} \tag{3.5}$$

   Obviously, the limit for $s \Rightarrow 0$ of F(s) is $\frac{1}{R}$, that is to say, R is the droop. According to equations (3.1) and (3.3), R, $P_{max(system)}$ and $P_{min(system)}$ are the one needed conversion in the *base.m* file. The following code describes these equations, where p.con(:,4), p.con(:,5) and p.con(:,6) represent R, $P_{max(system)}$ and $P_{min(system)}$, respectively. "getvar(Syn,p.syn,'mva')" is to get the machine power, and Settings.mva is usually set 100 as the system power base.

```
1 for i = 1:p.n
```

```
2    if (p.con(i,2) == 1) || (p.con(i,2) == 2),
3        p.con(i,4) = Settings.mva.*p.con(i,4)./getvar(Syn,p.syn(i),'mva');
4        p.con(i,5) = p.con(i,5).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
5        p.con(i,6) = p.con(i,6).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
6    end
```

Model 3 —The transfer function of Model 3 is given by

$$F_3(s) = \frac{\Delta G}{\omega_{ref} - \omega} = \frac{1 + T_r s}{T_g T_r T_p s^3 + T_g(T_r + T_p)s^2 + (T_g + \sigma T_r + \delta T_r)s + \sigma} \tag{3.6}$$

Parameter σ is the droop of $F_3(s)$ when $s \Rightarrow 0$, which is defined as the 11th parameter in "Tg.con". From the diagram in Fig.2.11, we can see that both gate position limits and rate limits need to match the system base because the whole loop of the governor will operate on the system base when performing a simulation.

```
1  if (p.con(i,2) == 3),
2      p.con(i,5) = p.con(i,5).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
3      p.con(i,6) = p.con(i,6).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
4      p.con(i,7) = p.con(i,7).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
5      p.con(i,8) = p.con(i,8).*getvar(Syn,p.syn(i),'mva')/Settings.mva;
6      p.con(i,11) = p.con(i,11).*Settings.mva./getvar(Syn,p.syn(i),'mva');
7  end
```

Model 4 —In Model 4, the turbine governor consists two parts, that are: PI controller and mechanical-hydraulic speed governing. The integrator in the PI controller in Fig.2.13 integrates the error of the rotor speed, which plays the role of driving the speed as close as possible to the speed reference, but does not make any contribution to the gain (or droop). Additionally, the proportional part in the PI controller has to keep the same the base with the integrator. As a consequence, it can not be considered as droop for base conversion as well. As shown in Fig.3.1, the mechanical-hydraulic speed governor actually is the typical turbine governor used in Model 3 . In this case, it is reasonable to set the same droop used in Model 3 for base conversion. The code is the same as in Model 3.

Model 5 —In Model 5, the relationship between ΔP (linear with ΔG) and $\Delta\omega$ is the one shown in Fig.3.2. As in multi-input systems, when calculating a transfer function of a particular input and output, there is no need to consider other inputs. This case is applicable to P_{ref}.

$$F_5(s) = \frac{\Delta P}{\omega_{ref} - \omega} = \frac{K_i + K_p s}{T_p s^2 + (\sigma K_p + 1)s + \sigma K_i} \tag{3.7}$$

According to equation (3.7), parameter σ is the droop of $F_5(s)$ when $s \Rightarrow 0$, and thus, the code is the same as the one used for Model 3.

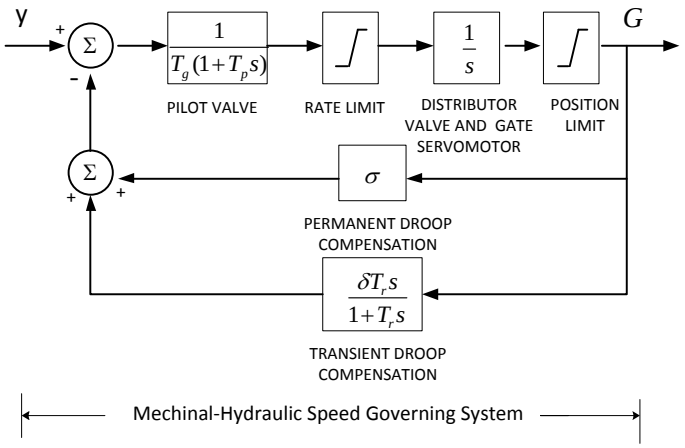


Figure 3.1.: Mechanical-hydraulic speed governor in Model 4

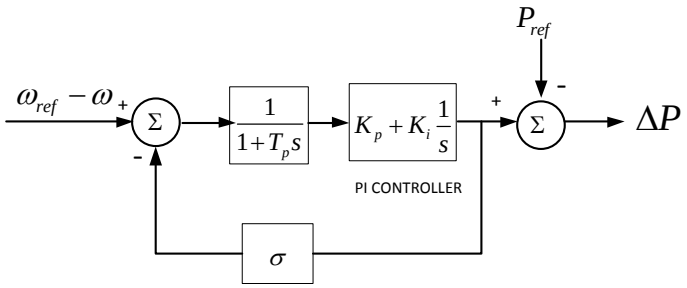


Figure 3.2.: Diagram of relationship between ΔP and $\Delta\omega$ in Model 5

Model 6 —As shown in Fig.2.18, firstly the permanent droop converts the power difference from system base to machine base by using (3.1) in order to operate in the same scale of the rotor speed scale. In the code, "p.con(i,16)" is where the permanent droop R_p is stored.

```
1 if (p.con(i,2) == 6),
2     p.con(i,16) = p.con(i,16).*Settings.mva./getvar(Syn,p.syn(i),'mva');
3 end
```

Then, the following part of governor is similar to Model 4. The PID controller has no impact on the droop for base conversion and the droop of the servomotor is 1. So, obviously, the droop for base conversion is $\frac{1}{g_{max}-g_{min}}$. To ensure that the output of mechanical power is in system base, the gate position must be transfered from machine base to system base by multiplying $\frac{S_{machine}}{S_{system}}$. That is achieved in "setx0" file through multiplying $\frac{1}{g_{max}-g_{min}}$ with that ratio.

2. block: This file defines the prompt strings in parameter box for turbine and governor models in PSAT simulation library. New strings for Model 3 to 6 are added in this file. The following is an example of Model 3.

```
case 3,
 prompts(idx5) = {'Maximum gate opening rate vmax [p.u.]'; ...
             'Minimum gate opening rate vmin [p.u.]';...
             'Pilot valve time constant Tp [s]';...
             'Dashpot time constant Tr [s]';...
             'Permanent speed droop sigma [p.u./p.u.]';...
             'Temporary speed droop ∆ [p.u./p.u.]';...
             'Water starting time Tw[s]';...
             'Patical derivative of turbine flow rate with respect to ...
                tubine head a11';...
             'Patical derivative of turbine flow rate with respect to ...
                gate position a13';...
             'Patical derivative of turbine torque with respect to ...
                tubine head a21';...
             'Patical derivative of turbine torque with respect to ...
                gate position a23'};
 enables(idx6) = {'off'; 'off'};
```

The *"idx5"* and *"idx6"* are indexes indicating the position of each parameter located in the parameter box. For this instance, all these listed parameter are located at positions 6 to 16 in parameter box, while positions 17 and 18 are vacant for other models utilizing more parameters.

```
idx5 = [6 7 8 9 10 11 12 13 14 15 16];
idx6 = [17 18];
```

The *block.m* file in the TGclass programs the parameter box for turbine and governor models in PSAT Simulink library. While *fm_lib* designs more general settings for the whole simulation library, which also contains some turbine and governor models specifications. This *fm_lib* file does not belong to TGclass, but acts as an independent file in the PSAT package. Right click on *fm_lib* and choose "Open as Text", then find the "Tg" block definition starting with:

```
    Block {
      BlockType       SubSystem
      Name            "Tg"
      Tag             "PSATblock"
```

As the amount of models changes, relevant variables from "MaskStyleString" until "MaskVariables" need to be modified. More strings should be added in "MaskPromptString" to make sure that the maximum parameters amount is the same that in *block.m*. These strings are only default, and await for specific strings set in the *block.m* file to cover them. "ModelHelp" is used to give a simple description of the model, so new models' are added here.

3. dynidx: New index for state variables in Model 3 to 6 are added. For Model 3, there are four state variables, so

```
   case 3
```

```
2       a.tg1(i) = DAE.n + 1;
3       a.tg2(i) = DAE.n + 2;
4       a.tg3(i) = DAE.n + 3;
5       a.tg4(i) = DAE.n + 4;
6       DAE.n = DAE.n + 4;
```

4. fcall: Differential equations for Model 3 to 6 introduced in Chapter 2 are added here. For example, for Model 3 we have

```
1    % define and name state variables
2    tg1 = DAE.x(p.dat3(:,1));
3    tg2 = DAE.x(p.dat3(:,2));
4    tg3 = DAE.x(p.dat3(:,3));
5    tg4 = DAE.x(p.dat3(:,4));
6    wref = DAE.y(p.wref(p.ty3));
7    % gate position limits
8    Δ_G = tg2;
9    Δ_G = max(Δ_G,p.con(p.ty3,6)-p.dat3(:,6));
10   Δ_G = min(Δ_G,p.con(p.ty3,5)-p.dat3(:,6));
11   p.dat3(:,18) = Δ_G;
12   % rate limits
13   v = tg1;
14   v = max(v,p.con(p.ty3,8));
15   v = min(v,p.con(p.ty3,7));
16   p.dat3(:,19) = v;
17   % differential equations
18   DAE.f(p.dat3(:,1)) = ...
         p.u(p.ty3).*(p.dat3(:,5).*(wref-DAE.x(p.dat3(:,17)))...
19       -(p.dat3(:,15)+p.dat3(:,14)).*Δ_G+p.dat3(:,15).*p.dat3(:,8).*tg3)...
20       -p.dat3(:,7).*tg1);
21   DAE.f(p.dat3(:,2)) = p.u(p.ty3).*v;
22   DAE.f(p.dat3(:,3)) = p.u(p.ty3).*(Δ_G-p.dat3(:,8).*tg3);
23   DAE.f(p.dat3(:,4)) = p.u(p.ty3).*(p.dat3(:,11).*p.dat3(:,12).*...
24       (p.dat3(:,6)+Δ_G)-tg4.* p.dat3(:,11));
```

p.dat3 is the index for Model 3 in *tg.con* and will be stated in the *setx0* file. DAE.x(p.dat3(:,1)) represents a state variable, which is the first element in the matrix *p.dat3*, namely x_{g1}. Similarly, other parameters' index can be found in *p.dat3* as well. While, DAE.f(p.dat3(:,1)) stands for the derivative of state variable x_{g1}. *p.u(p.ty3)* is a scale factor to change the parameters in per unit.

5. Fxcall: This file computes Jacobian matrix F_x, F_y and G_x (refer to (4.1) in Chapter 4). As (4.1) indicates, F_x represents the partial derivatives of function f with respect to state variables x, and F_y the partial derivative of function f with respect to algebraic variables, and G_x the partial derivative of algebraic function g with respect to state variables x.

   Sparse matrices often appear in science or engineering when solving differential equations. Sparse data is by nature easily compressed, and this compression almost always results in significantly less computer data storage usage. In MATLAB function "sparse" can create a sparse matrix as shown below.

```
[i,j,s] = find(S);
[m,n] = size(S);
S = sparse(i,j,s,m,n);
```

S is an m-by-n sparse matrix, where the real or complex entries vector, s, indicates the values of nonzero elements and two integer index vectors, i and j, represent each element's corresponding the position of rows and columns.

As shown in the code, for example, " sparse(tg1, p.dat3 (:, 17), u3.*p.dat3 (:, 5), DAE.n, DAE.n)" illustrates the partial derivative of $\dot{x}_{g1}$ with respect to state variables ω. The Jacobian matrix F_x is represented as DAE.Fx, whose size is DAE.n*DAE.n. The value in the x_{g1} row, ω column is u3.*p.dat3 (:, 5), which is the partial derivative of $\dot{x}_{g1}$ with respect to state variables ω according to the DAEs of Model 3. u_G and u_v are logically true (="1") only when the gate position and gate rate do not reach any limits. For example, as the 10th line in the code below shows, only when u_v is logically true, the partial derivative $\dot{x}_{g2}$ with respect to x_{g1} is 1, otherwise it is zero.

```
u_G = Δ_G < (p.con(p.ty3,5)−p.dat3(:,6)) & Δ_G >(p.con(p.ty3,6)...
   −p.dat3(:,6));% windup limiter
u_v = v < p.con(p.ty3,7) & v > p.con(p.ty3,8);% windup limiter
DAE.Fx = DAE.Fx − sparse(tg1,p.dat3(:,17),u3.*p.dat3(:,5),DAE.n,DAE.n);
DAE.Fx = DAE.Fx − sparse(tg1,tg1,p.dat3(:,7),DAE.n,DAE.n);
DAE.Fx = DAE.Fx − ...
    sparse(tg1,tg2,u_G.*u3.*(p.dat3(:,14)+p.dat3(:,15)).*...
 p.dat3(:,5),DAE.n,DAE.n);
DAE.Fx = DAE.Fx + sparse(tg1,tg3,u3.*p.dat3(:,5).*p.dat3(:,15).*...
    p.dat3(:,8),DAE.n,DAE.n);
DAE.Fx = DAE.Fx + sparse(tg2,tg1,u_v.*u3,DAE.n,DAE.n);
DAE.Fx = DAE.Fx + sparse(tg3,tg2,u_G.*u3,DAE.n,DAE.n);
DAE.Fx = DAE.Fx − sparse(tg3,tg3,p.dat3(:,8),DAE.n,DAE.n);
DAE.Fx = DAE.Fx + sparse(tg4,tg2,u_G.*u3.*p.dat3(:,12).* p.dat3(:,11),...
   DAE.n,DAE.n);
DAE.Fx = DAE.Fx − sparse(tg4,tg4,p.dat3(:,11),DAE.n,DAE.n);
DAE.Fy = DAE.Fy + sparse(tg1,p.wref(p.ty3),u3.*p.dat3(:,5),DAE.n, DAE.m);
DAE.Gx = DAE.Gx + sparse(pm3,tg2,u_G.*u3.*p.dat3(:,10).*p.dat3(:,13),...
   DAE.m,DAE.n);
DAE.Gx = DAE.Gx + sparse(pm3,tg4,u3,DAE.m,DAE.n);
```

The following matrices indicate the parts of the Jacobian matrix F_x which are related to the state variables in Model 3, Model 4, Model 5 and Model 6, respectively. Taking Model 3 as an example, the first element $-\frac{1}{T_p * T_g}$ represents the partial derivative of $\dot{x}_{g1}$ with respect to ω. These matrices are used for writing the "Fxcall" file. The same approach is applicable for writing the Jacobian matrix F_y and G_x related to turbine and governor models.

$$\text{Model 3}: \begin{array}{c} \\ \dot{x}_{g1} \\ \dot{x}_{g2} \\ \dot{x}_{g3} \\ \dot{x}_{g4} \end{array} \begin{array}{c} \begin{array}{ccccc} \omega & x_{g1} & x_{g2} & x_{g3} & x_{g4} \end{array} \\ \begin{pmatrix} -\frac{1}{T_p * T_g} & -\frac{1}{T_p} & -\frac{\sigma+\delta}{T_p * T_g} & \frac{\delta}{T_p * T_g * T_r} & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -\frac{1}{T_r} & 0 \\ 0 & 0 & \frac{a_{13}a_{21}}{a_{11}^2 T_w} & 0 & -\frac{1}{a_{11}T_w} \end{pmatrix} \end{array}$$

$$\text{Model 4}: \begin{array}{c} \\ \dot{x}_{g1} \\ \dot{x}_{g2} \\ \dot{x}_{g3} \\ \dot{x}_{g4} \\ \dot{x}_{g5} \end{array} \begin{array}{c} \begin{array}{ccccccc} \omega & & x_{g1} & x_{g2} & x_{g3} & x_{g4} & x_{g5} \end{array} \\ \begin{pmatrix} -K_i & 0 & 0 & 0 & 0 & 0 \\ -\frac{K_p}{T_p * T_g} & -\frac{1}{T_p * T_g} & -\frac{1}{T_p} & -\frac{\sigma+\delta}{T_p * T_g} & \frac{\delta}{T_p * T_g * T_r} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -\frac{1}{T_r} & 0 \\ 0 & 0 & 0 & \frac{a_{13}a_{21}}{a_{11}^2 T_w} & 0 & -\frac{1}{a_{11}T_w} \end{pmatrix} \end{array}$$

$$\text{Model 5}: \begin{array}{c} \\ \dot{x}_{g1} \\ \dot{x}_{g2} \\ \dot{x}_{g3} \\ \dot{x}_{g4} \end{array} \begin{array}{c} \begin{array}{ccccc} \omega & x_{g1} & x_{g2} & x_{g3} & x_{g4} \end{array} \\ \begin{pmatrix} -\frac{1}{T_p} & -\frac{\sigma K_p - 1}{T_p} & -\frac{\sigma}{T_p} & 0 & 0 \\ 0 & K_i & 0 & 0 & 0 \\ 0 & -\frac{K_p}{T_g} & 1 & -1 & 0 \\ 0 & 0 & 0 & \frac{2x_{g4}^2}{T_w x_{g3}^3} & -\frac{2x_{g4}}{T_w z^2} \end{pmatrix} \end{array}$$

$$\text{Model 6}: \begin{array}{c} \\ \dot{x}_{g1} \\ \dot{x}_{g2} \\ \dot{x}_{g3} \\ \dot{x}_{g4} \\ \dot{x}_{g5} \end{array} \begin{array}{c} \begin{array}{ccccccc} \omega & & x_{g1} & x_{g2} & x_{g3} & x_{g4} & x_{g5} \end{array} \\ \begin{pmatrix} -K_i & 0 & 0 & 0 & 0 & 0 \\ -\frac{K_d}{T_d^2} & -\frac{1}{T_d} & 0 & 0 & 0 & 0 \\ -\frac{K_a(K_p + \frac{K_d}{T_d})}{T_a} & \frac{K_a}{T_a} & \frac{K_a}{T_a} & -\frac{1}{T_a} & \frac{K_a}{T_a} & 0 \\ 0 & 0 & 0 & 1 & -\frac{1}{a_{11}T_w} & 0 \\ 0 & 0 & 0 & 0 & -\frac{2x_{g5}^2(G_{max}-G_{min})^2}{T_w x_{g4}^3} & -\frac{2x_{g5}(G_{max}-G_{min})^2}{T_w x_{g4}^2} \end{pmatrix} \end{array}$$

6. init: Initialization properties of Model 3 to 6 are added.
7. P_{mec}: Mechanical power equations for Model 3 to 6 are added.
8. remove: Model 3 to 6 turbine and governor models' removal.
9. setx0: The tg.dat defines index of parameters in each model. In the index, all parameters that will be implemented in the model equations are arranged with fixed positions so that they can be called.

```
a.dat3 = [a.tg1(a.ty3), ...                    %  1
          a.tg2(a.ty3), ...                    %  2
          a.tg3(a.ty3), ...                    %  3
          a.tg4(a.ty3), ...                    %  4
          K5, ...                              %  5
          Porder3, ...                         %  6
          ap, ...                              %  7
          ar, ...                              %  8
          aw, ...                              %  9
          K1, ...                              % 10
          K2, ...                              % 11
          K3, ...                              % 12
          K4, ...                              % 13
          sigma, ...                           % 14
          Δ,...                          % 15
          a23,...                              % 16
          Syn.omega(a.syn(a.ty3)), ...         % 17
          zeros(length(a.ty3),1), ...          % 18
          zeros(length(a.ty3),1)];             % 19
```

Moreover, the initial values of the state variables and function f are set here.

```
DAE.x(a.dat3(:,1)) = 0;
DAE.x(a.dat3(:,2)) = 0;
DAE.x(a.dat3(:,3)) = 0;
DAE.x(a.dat3(:,4)) = a.u(a.ty3).*a.dat3(:,12).*a.dat3(:,6);
DAE.f(a.dat3(:,1)) = 0;
DAE.f(a.dat3(:,2)) = 0;
DAE.f(a.dat3(:,3)) = 0;
DAE.f(a.dat3(:,4)) = 0;
```

10. subsasgn: Used for the assignment of the turbine and governor properties for Model 3 to 6.

11. subsref: Used for the returning turbine and governor properties for Model 3 to 6.

12. TGclass: Add Model 3 to 6 in the TGclass constructor. In addition, change *ncol* to be equal to maximum amount of columns in the matrix con and modify the corresponding format.

```
a.ncol = 20;
a.format = ['%4d %4d ',repmat('%8.4g ',1,17),'%2d'];
```

3.1.3. Power System Implementations

Device model needs to be integrated in the overall power system to evaluate its performance. PSAT prescribes the format of power systems, including *Bus.con*, *SW.con*, *PV.con*, *PQ.con*, *Line.con* and other "*Class.con*" files, all of which are chosen depending on the particular power system. Some settings and other models data, such as *Fault.con*, *Breaker.con* can also be contained in the system data file. PSAT already provides power system data files with various sample systems in the "test" folder.

As for power system simulation, compared to the PSAT GUI, command line usage could be desired when running PSAT on a remote server/host or when launching PSAT from within user defined routines. The command line usage of PSAT also allows speeding up operations. However, the PSAT GUI has a user-friendly interface and is easy to get started with. In the PSAT GUI, double click "Data File" and choose the test power system. Perturbations can be independent or integrated in the "Data File" in the form of *Fault.con*, *Breaker.con*. After selecting files, we can perform power flow by clicking the "Power Flow" button and perform other calculations as described in Section 3.1.1. What we need to pay attention to is the "Eigenvalue Analysis" routine, which will be used for small signal stability analysis in the next Chapter. The home window of the PSAT GUI is shown in Fig.3.3.

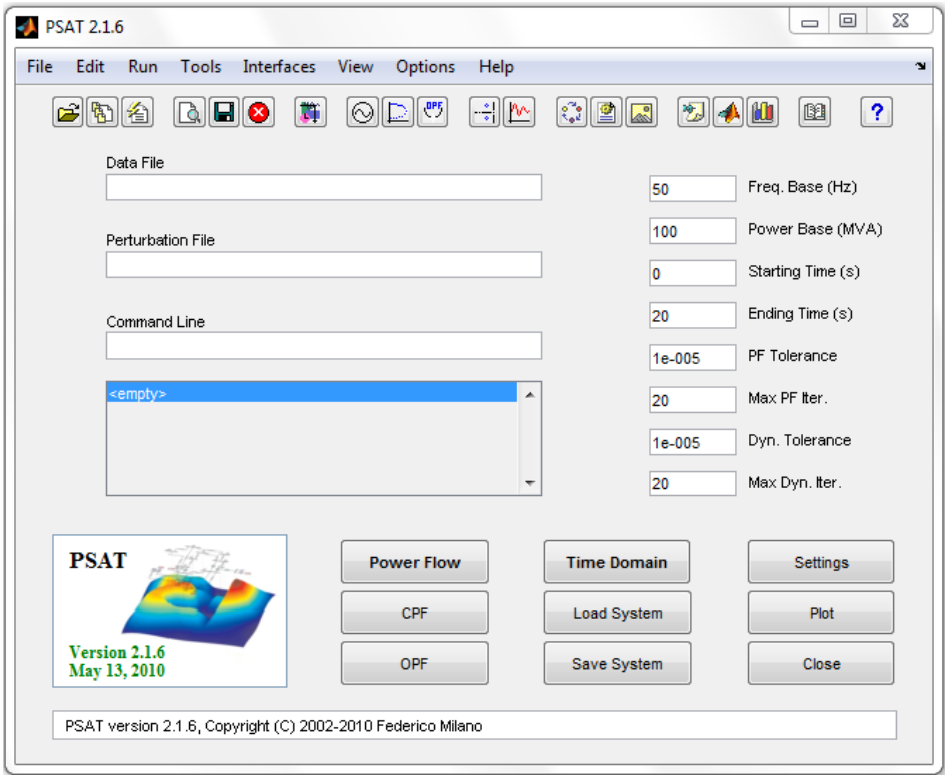


Figure 3.3.: The PSAT GUI home window

3.2. Matlab/Simulink SimPowerSystems Implementation

3.2.1. General Introduction about SimPowerSystems

SimPowerSystems (SPS) extends Simulink with tools for modeling and simulating the generation, transmission, distribution, and consumption of electrical power. It contains a library with more integrated and intelligent components and devices, specially used in power systems studies and analyses. Demos and examples can also be found in the SPS library.

3.2.2. Hydro Turbine and Governor Models Implementations

There are three steps to implement new hydro turbine and governor models in SPS.

1. Build a new model in the Simulink environment with the components in the Simulink library according to the model realization diagram, let's say 2.12 introduced in Chapter 2. As this realization diagram only contains integrators and gain components, it is easy to find all elements in realization diagram from the Simulink library. The reason why "zero-pole" blocks should not be allowed, as depicted in model block diagram in Fig.2.11, is that only simple integrators can be initialized of the state variables associated to the integrator, which significantly affects the power system performance at the beginning of the simulation. Otherwise, the system will firstly take some time to reach steady state. When double clicking gain blocks, it is better to set gain value with same letters in model realization diagram instead of fixed numbers. How to give the values for these

letters will explain in the third step. This method contributes to modifying values of parameters with ease.

2. Select all components in the model, then right click and choose “Create subsystem”. A block with input and output is created. The whole model is encapsulated in this block and the input and output are used to connect with other components in a power system as shown in Fig.3.4. “Format”, “Foreground Color” and “Background Color” can be used to set the appearance of the block.

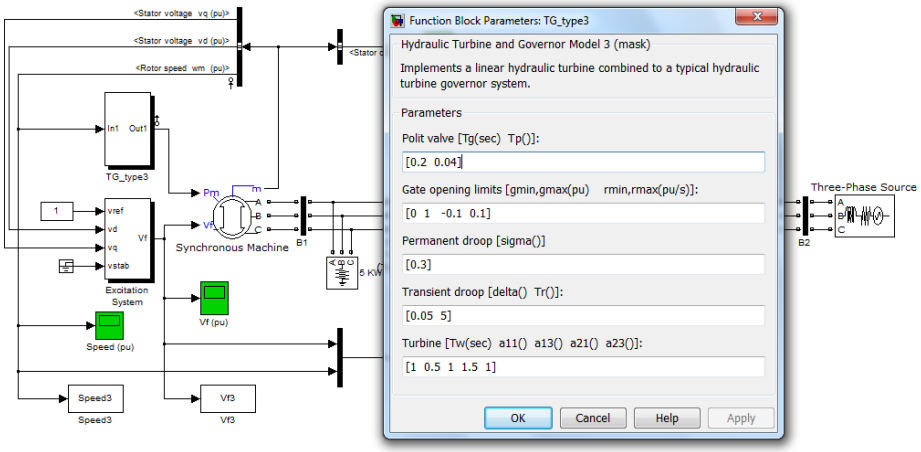


Figure 3.4.: Model 3 block diagram in a power system and its parameters mask

3. All parameters in a model are represented by letters instead of fixed numbers so that users can change them according to their own system. In this case, a settings parameter mask should be provided. Right click on the model block, choose “Mask Subsystem” then you can add or edit variables and prompt strings under “parameter”, set initialization commands under “Initialization”, set mask type and description under “Documentation”.

   Each model’s structure under the main block are shown sequentially below, followed by parameter blocks.

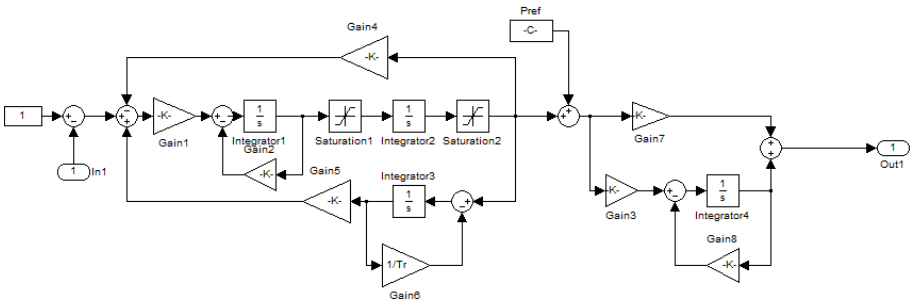


Figure 3.5.: Model 3 block diagram realization in SPS

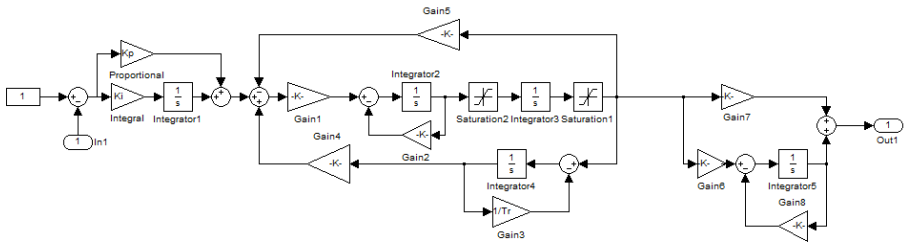


Figure 3.6.: Model 4 block diagram realization in SPS

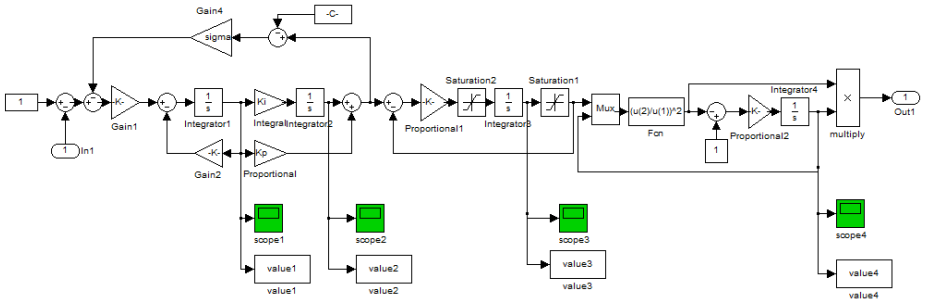


Figure 3.7.: Model 5 block diagram realization in SPS

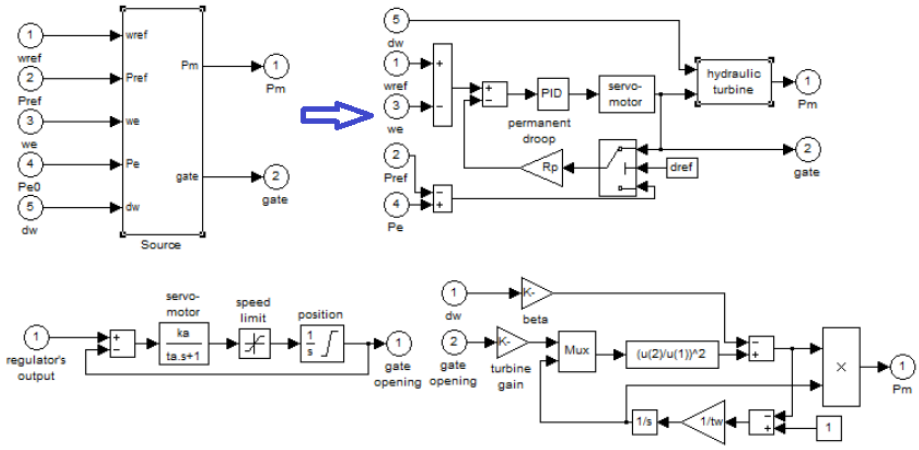
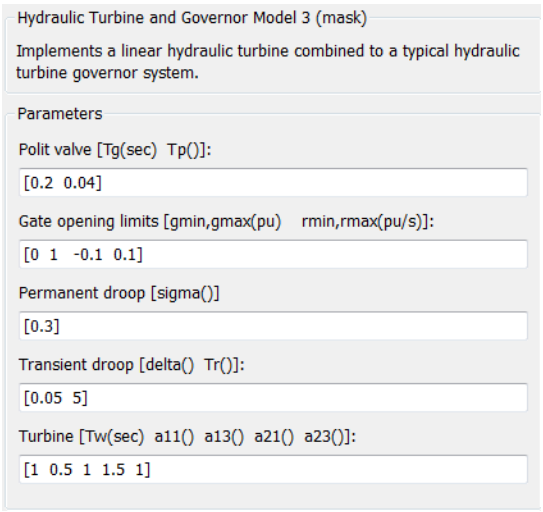


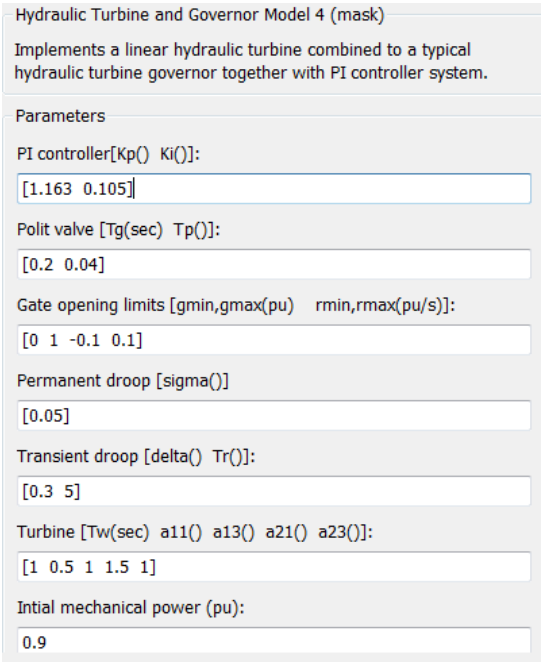
Figure 3.8.: Model 6 block diagram realization in SPS

3.2.3. Power System Implementations

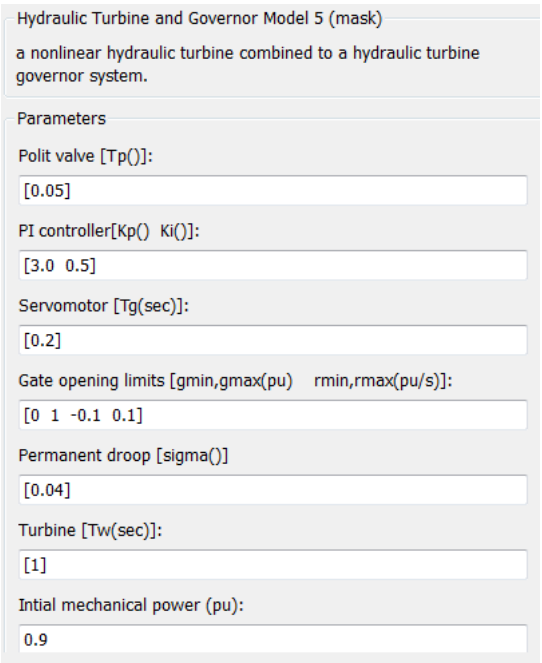
The power system in Fig.3.10 can be opened by typing "power_turbine" in the MATLAB Command Window, which is a demo power system with turbine model provided by SimPowerSystems.



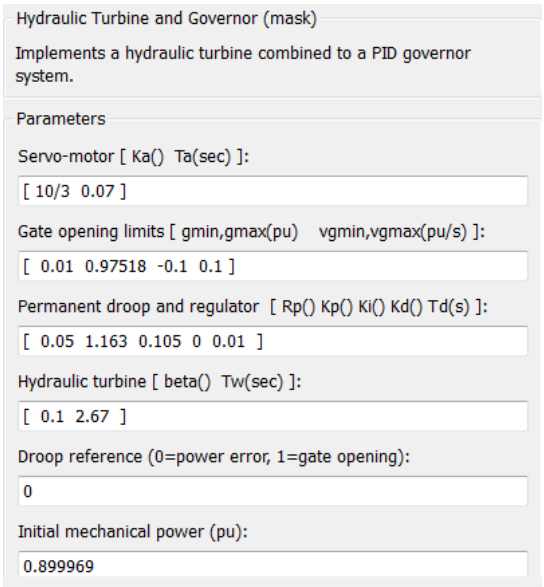
(a) Parameter block for Model 3 in SPS



(b) Parameter block for Model 4 in SPS



(c) Parameter block for Model 5 in SPS



(d) Parameter block for Model 6 in SPS

Figure 3.9.: Parameter blocks for models in SPS

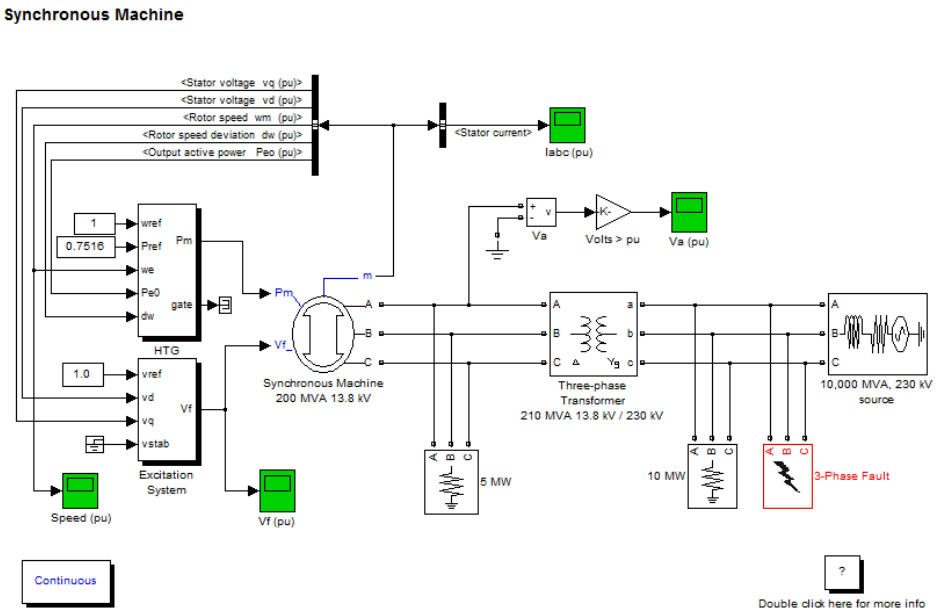


Figure 3.10.: Power turbine demo in SPS

For new power system implementations, the SPS library provides essential components such as machines, excitation systems, loads, buses, etc. After connecting all the devices, by double clicking the "powergui block", we can set simulation and configuration options and select analysis tools. SPS simulations in this thesis are carried out in "Phaser" simulation mode and Phasor frequency is 50 Hz, which is set in the "Configure Parameters" in GUI. Then in "Load Flow and Machine Initialization", the load flow can be updated with respect to how much "Active power" is set by the users.

But there is a initialization problem when new hydro turbine and governor models are applied to the power system. The machine load flow can not set automatically the initial condition for the integrators in the hydro turbine and governor models as shown in Fig.3.11. Moreover, there are tiny oscillations in the beginning of simulation. One possible solution is applying a load flow text file to assign the initialization values manually, this will be shown with an example in Chapter 4.

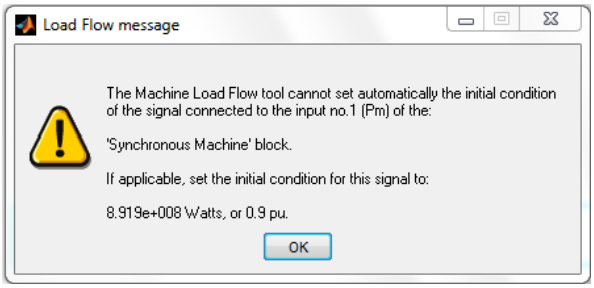


Figure 3.11.: Load Flow Message

The SPS also provides tools for linearizion analysis. First, find out the inputs and outputs of the component that needs to be linearized. Right click and choose “Linearizition Points” with a particular “Input Point”, “Output Point” or other points. Then, click on “Tools”, go on“Control Design” and choose“Linear Analysis”. There you can check “Plot linear analysis” in different forms, such as Bode response plot, Nyquist plot. Last, click on “Linear Model” button and get the analysis results of the linearized model. In the generated window, “Plot configuration” under “Edit” is helpful to obtain the desired plot configuration.

4 Model Validation

In this thesis, we examine the performances of the implemented models in two power systems—a single-machine infinite-bus (SMIB) system, and the KTH-NORDIC 32 system. The evaluation of a power system's performance is concerned with the stability of that system: ie. if it remains in an equilibrium after being subjected to a disturbance [1]. Transient (large signal) stability and small signal (small disturbance) stability analysis, as well as frequency response analysis will be performed in this chapter.

4.1. Single Machine Infinite Bus System

4.1.1. SMIB System Introduction and Parameters in PSAT and SimPowerSystems

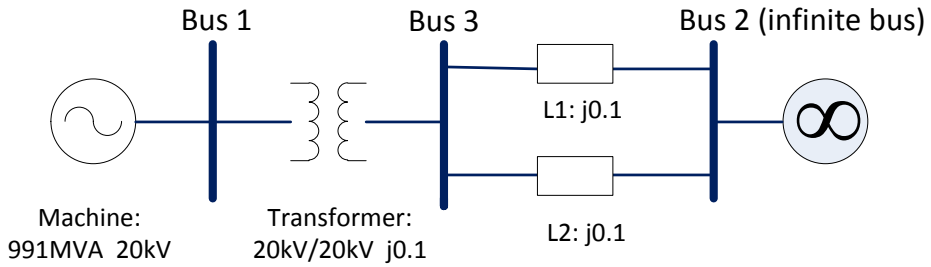


Figure 4.1.: The SMIB

Figure 4.1 shows a SMIB system consisting of one 991 MVA, 20kV, 50HZ generator, one transformer operating 20 kV on the primary and secondary, two lines with reactance of 0.1 p.u. and an infinite generator (or source) with 100000 MVA (an infinite bus).

The SMIB system data and corresponding description in PSAT is attached in Appendix A.4. The SMIB system with Model 6 in SPS is presented in Fig.4.2 as an example. Parameters for the synchronous machine, excitation system and transformer in SMIB in SPS are listed in Appendix A.5.

When simulating hydro turbine and governor models in the SMIB system using SPS, the machine load flow tool in the "powergui" can not set automatically the initial conditions of newly implemented models. In addition there are tiny oscillations in the beginning of

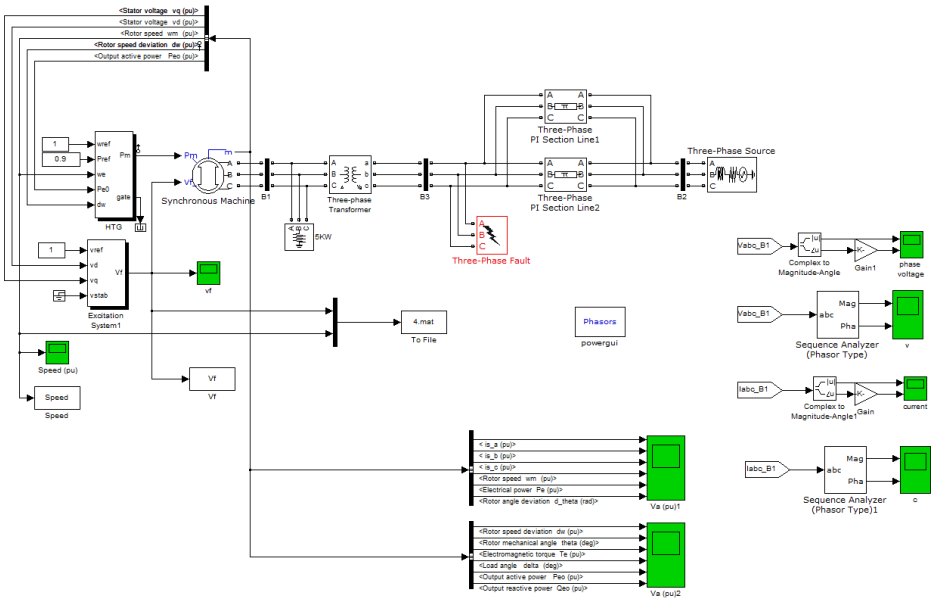


Figure 4.2.: The SMIB system with hydro turbine and governor Model 6 in SPS

the simulation in SPS as depicted in Fig.4.4. As introduced in Section 3.2.3, one possible solution is to begin the simulation with a load flow text file, where the initial values for different integrators are set. Here is an example load flow text file for Model 3. The 6th line computes the power flow, this is normally realized in the "powergui", but not for this case with new models. Lines 36th to 40th set the values of power reference and initial values of four integrators. As long as parameters have valid names in the system, they can be set to desired values by the user. Even though in this thesis we utilize a load flow text file to initialize integrators in the hydro turbine and governor models, the oscillations seen at the beginning of the simulation can not be mitigated. This reflects one shortcoming of proprietary software: it is not possible for users to access nor to modify the source code so that this error is mitigated. In this case, it is not even possible for users to fix some problems in the software by themselves.

```
1  model = 'SMIB_test_type3';% model name
2  Pref = 0.9; % power reference
3  % Get the current load flow parameters of the model
4  % LFPARAM can be used as a template variable to define load flow
5  % parameter values to compute new machine load flow.
6  lfparam = power_loadflow(model);
7
8  % You should edit the contents of the lfparam structure
9  % and set the desired values. You could also perform this task using the
10 % load flow user interface of the powergui block
11
12 % INSERT YOUT CODE HERE AND EDIT THE LFPARAM
13 % example
14 lfparam(1).set(1).TerminalVolage = 20000;
15 lfparam(1).set(1).ActivePower = 991000000*Pref;
16
17 % Computes the machine load flow of the model
```

```
% for the load flow parameters given in LFPARAM struct.
lfparam.DisplayWarnings = 'off'; % Diable unecessary warnings regarding ...
    initialization
lf = power_loadflow(model,lfparam);

% For each result in the LF structure you should set the parameters of the
% corresponding TG and excitation blocks manually not standard SPS blocks
% In these case, there is only 1 LF element and 1 TG to set.

% Hydro Turbine and Governor Model 3 Init %
% calculate the path of all block of the TG that should be initialized ...
    with the "lf" variables
param_pref_constant  = [model '/TG_type3/Pref'];
param_ic_integrator1 = [model '/TG_type3/Integrator1'];
param_ic_integrator2 = [model '/TG_type3/Integrator2'];
param_ic_integrator3 = [model '/TG_type3/Integrator3'];
param_ic_integrator4 = [model '/TG_type3/Integrator4'];

% set the parameters of the integrator and constant block
% You could put any value or calculated based on LF here
set_param(param_pref_constant,  'Value',            mat2str(lf(1).Pmec(2)));
set_param(param_ic_integrator1, 'initialcondition', mat2str(lf(1).Pmec(2)*0));
set_param(param_ic_integrator2, 'initialcondition', mat2str(lf(1).Pmec(2)*0));
set_param(param_ic_integrator3, 'initialcondition', mat2str(lf(1).Pmec(2)*0));
set_param(param_ic_integrator4, 'initialcondition', mat2str(lf(1).Pmec(2)*3));

% Run simulation. Plot measured data and simulation results
sim('SMIB_test_type3');
plot(tout,Speed3,'b')
title('Speed')
grid on
ylim([min(Speed3) max(Speed3)])
```

4.1.2. Transient Stability Analysis

Transient (large disturbance) stability is the ability of the power system to maintain synchronism when subjected to a severe transient disturbance, such as a fault on transmission line or a load change [6].

Response to a Fault

A three-phase fault is applied at Bus 3 at t=20 s and removed at t=20.02 s. After computing the power flow, time-domain simulations show the model responses in PSAT. Figure 4.3 depicts the generator angle speed of five models in PSAT and Fig.4.4 for SPS.

Response to a Breaker Opening

The breaker is applied on Line 3, between Bus 2 and Bus 3, is set to open at 20 s and set to close at 20.02 s. Figure 4.5 depicts the generator angle speed of five models in PSAT and in Fig.4.6 for SPS.

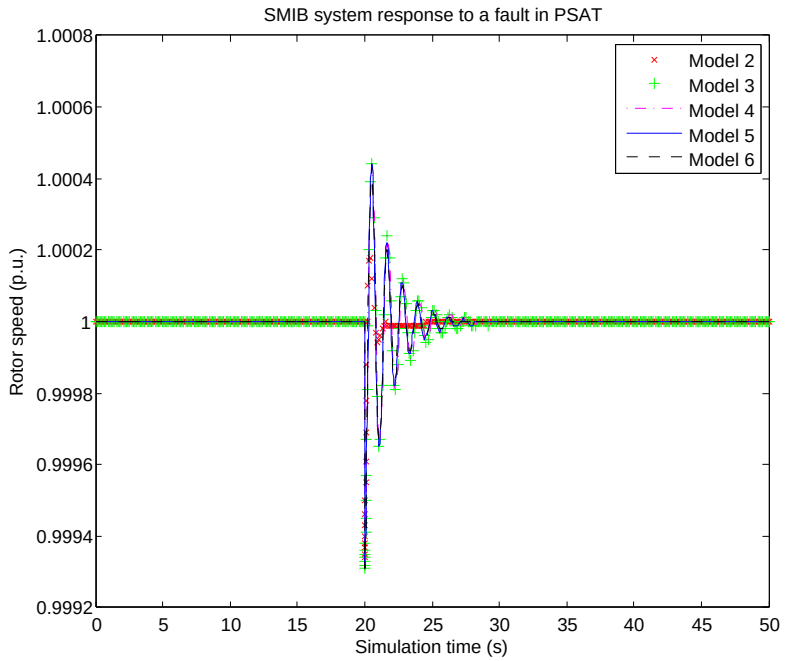


Figure 4.3.: SMIB system response to a fault in PSAT

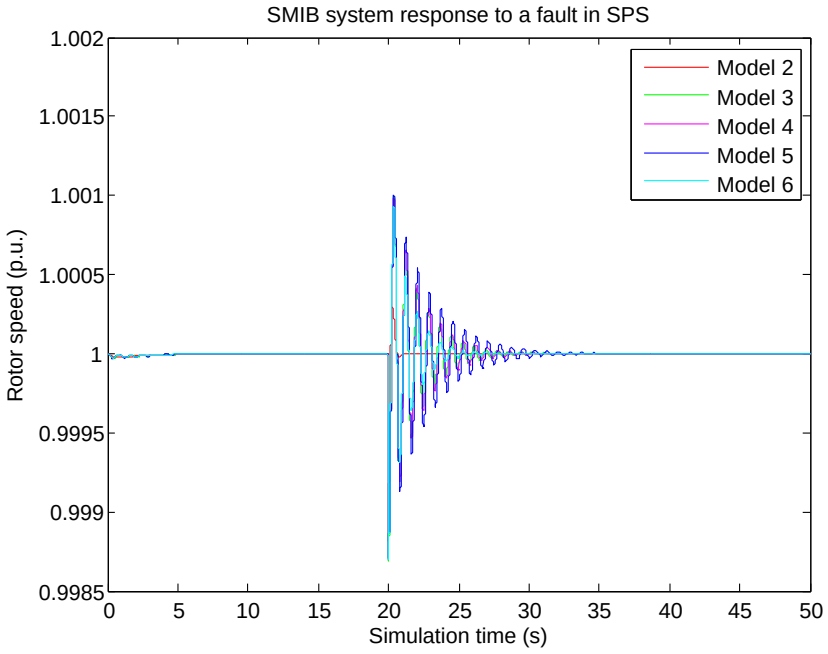


Figure 4.4.: SMIB system response to a fault in SPS

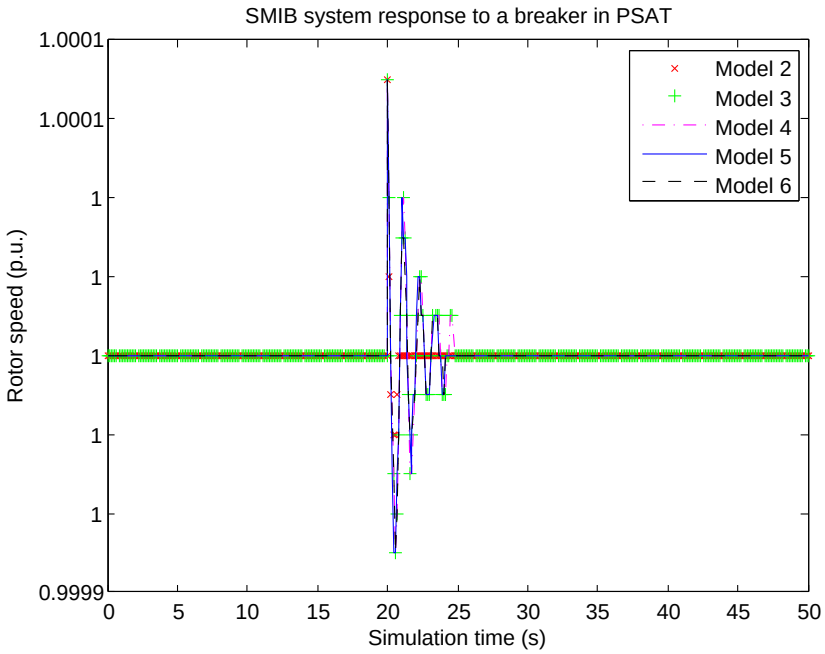


Figure 4.5.: SMIB system response to a breaker opening in PSAT

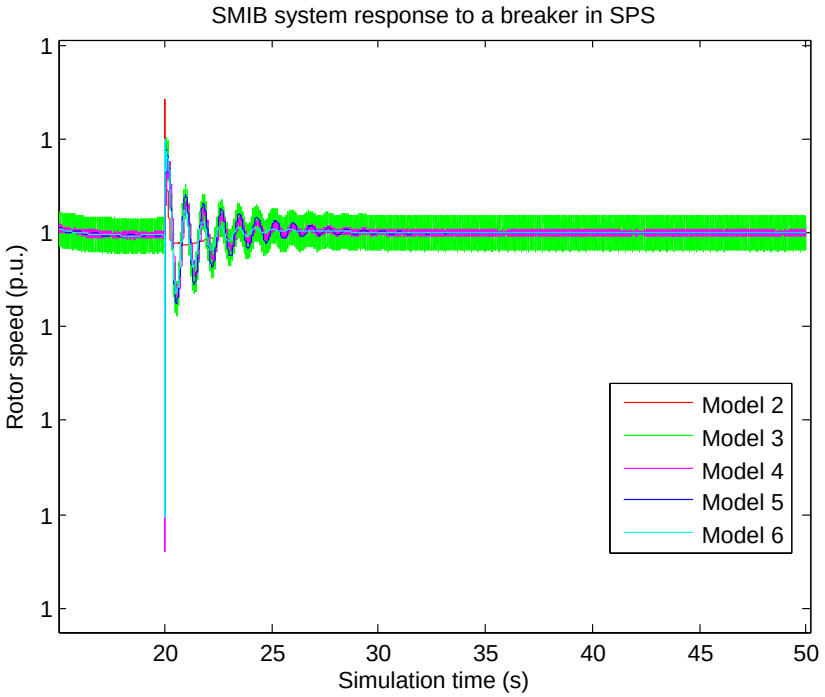


Figure 4.6.: SMIB system response to a breaker opening in SPS

Response to a Load Change

Next, we evaluate the system response to a load change by connecting a load with 0.5 p.u. active power and 0 reactive power on Bus 3 and implementing a 20% increment load change at t=2 s. This simulation is only done in PSAT.

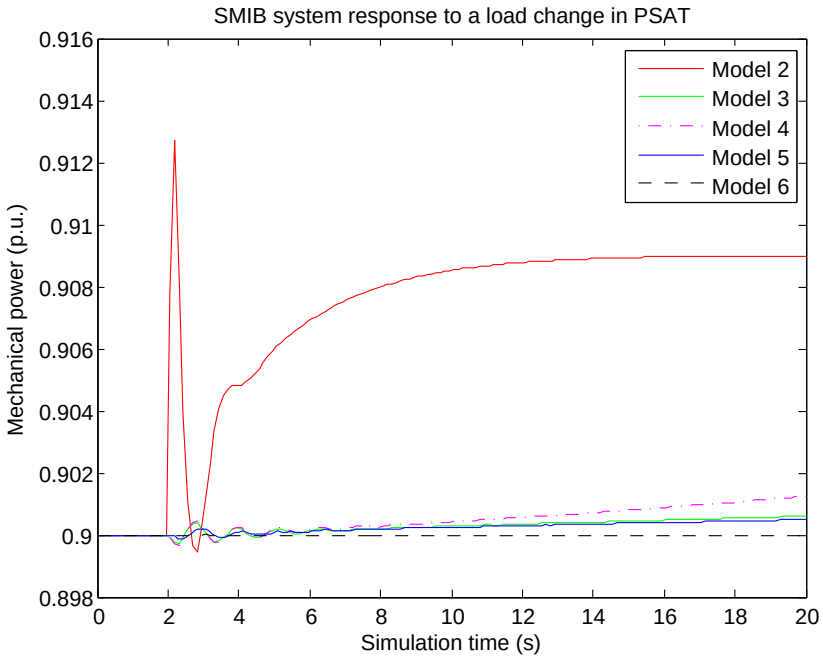


Figure 4.7.: SMIB system response to a load change in PSAT

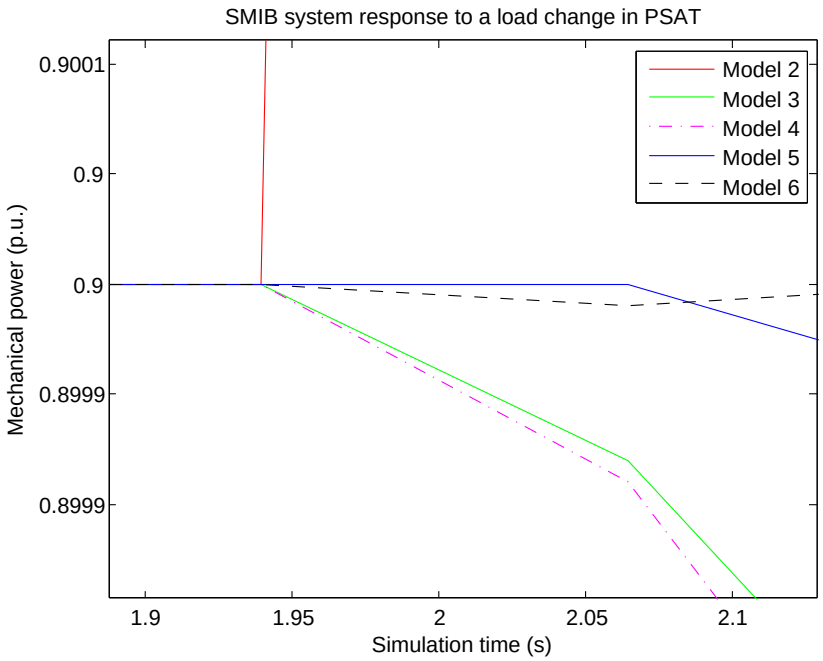


Figure 4.8.: Enlargement of Fig.4.7 during the initial steps of the simulation

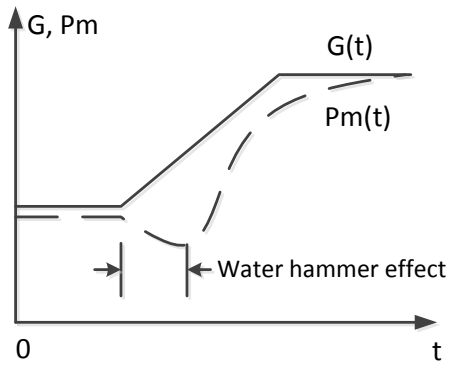


Figure 4.9.: Water hammer effect (Adapted from [27])

Analysis of Results —The time-domain simulations show the models' performances when subject to a large disturbance. For the responses to a fault and to a breaker opening, comparing SMIB system transient response in PSAT and SPS, clearly, they are not as close as we expect. All parameters and structures for every hydro turbine and governors in the two software are exactly same, but the settings for other devices, such as transformers, lines, loads, are really hard to get totally coherent. Here it can be demonstrated that the differences are in part due to model exchange barriers between two software. Another phenomenon can prove this conclusion as well. In SPS, even though Model 6 utilizes a nonlinear turbine model, it behaves faster than Model 3 and Model 4. It is probably due to Model 6 is provided by SPS with a more effective compatibility and particular design. Model 5 is supposed to be a bit slower as it is utilizing nonlinear turbine model. Comparatively speaking, Model 3 to Model 6 show relatively same performances to perturbations in PSAT.

In the responses to fault and breaker openings, Model 2 behaves with outstanding stability, which derives from its simple and robust structure. However, this model is not suitable for representing the behavior and capacity of hydro turbine and governors, which is indicated in the response to a load change.

Firstly, we should learn about the real behavior of hydro turbine and governors. When the load in a power system increases, the water flow gate has to open wider to meet the power demand. Once the water flow gate suddenly opens, the volume of the water flow will tend to increase which causes water pressure reduction, and the output mechanical power will decrease at first and then turn to increase. This effect is called "water hammer effect" as shown in Fig.4.9. The water hammer effect of the penstock greatly worsens the dynamic behavior of the hydro turbine and governors when compared to others (e.g. thermal turbine and governors). From Fig.4.7 and 4.8, we can easily find that Model 2 does not have a water hammer effect, in other words, it can not represent the performance of hydro turbine and governors.

4.1.3. Small Signal Stability Analysis

Small signal (or small disturbance) stability is the ability of a power system to maintain steady under small disturbances. The disturbances are considered sufficiently small for linearization

of system equations to be permissible for analysis purposes.

The linearization of system DAE (2.23) is as follows:

$$\Delta\dot{x} = f_x\Delta x + f_y\Delta y + f_u\Delta u \tag{4.1}$$

$$0 = g_x\Delta x + g_y\Delta y + g_u\Delta u$$

From the second equation of (4.1), we obtain:

$$\Delta y = -\frac{g_x}{g_y}\Delta x - \frac{g_u}{g_y}\Delta u \tag{4.2}$$

Combining (4.1) and (4.2):

$$\Delta\dot{x} = (f_x - \frac{f_y g_x}{g_y})\Delta x + (f_u - \frac{f_y g_u}{g_y})\Delta u \tag{4.3}$$

$$\Delta y = -\frac{g_x}{g_y}\Delta x - \frac{g_u}{g_y}\Delta u$$

These equations just correspond to the linearization form of a system with input and output matrices **A**, **B**, **C**, **D** as:

$$\Delta\dot{x} = A\Delta x + B\Delta u \tag{4.4}$$

$$\Delta y = C\Delta x + D\Delta u$$

where

$$\mathbf{A} = f_x - \frac{f_y g_x}{g_y}$$

$$\mathbf{B} = f_u - \frac{f_y g_u}{g_y}$$

$$\mathbf{C} = -\frac{g_x}{g_y}$$

$$\mathbf{D} = -\frac{g_u}{g_y}$$

The equilibrium points are calculated from

$$\dot{x} = f(x_0) = 0$$

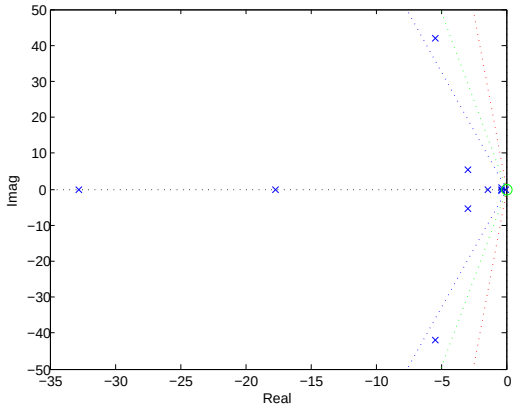
The eigenvalues λ_i can be computed from the A matrix according to

$$det(\lambda I - A) = 0 \tag{4.5}$$

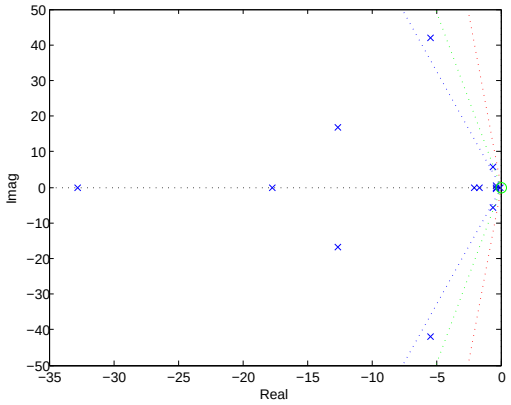
A complex eigenvalue λ is represented by

$$\lambda = \sigma \pm j\omega \tag{4.6}$$

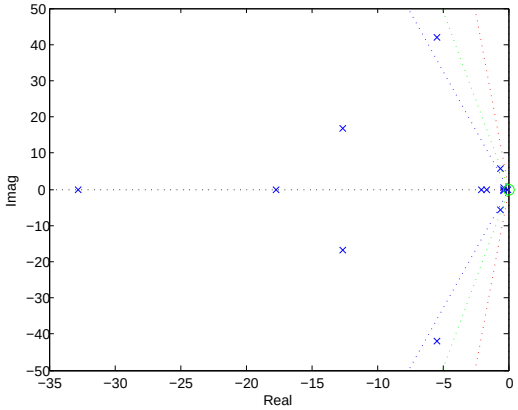
In Fig.4.10, negative real part of each eigenvalue in SMIB systems indicates they are stable. Eigenvalues for SMIB system are shown in the following figures.



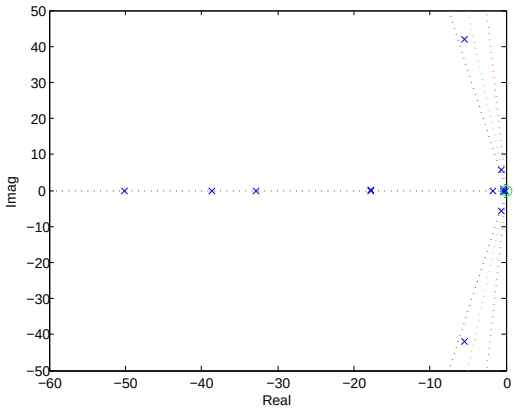
(a) Eigenvalues of SMIB system with Model 2



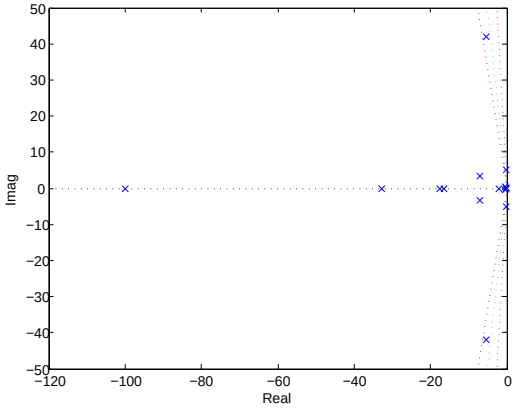
(b) Eigenvalues of SMIB system with Model 3



(c) Eigenvalues of SMIB system with Model 4



(d) Eigenvalues of SMIB system with Model 5



(e) Eigenvalues of SMIB system with Model 6

Figure 4.10.: Eigenvalues of SMIB system in PSAT

The damping ratio describes the rate of decay the system oscillations after a disturbance. It is given by

$$\zeta = \frac{-\sigma}{\sqrt{\sigma^2 + \omega^2}} \tag{4.7}$$

and the frequency (which is called "pseudo-frequency" in the PSAT manual [2], that can be observed during the transient is

$$f = \frac{\omega}{2 * \pi} \tag{4.8}$$

A lower damping ratio implies a lower decay rate, and influences the dynamic system behavior more than higher damping ratio. In this case, only two lowest damping modes are considered. Table 4.1 shows the small signal stability analysis results of the two lowest damping modes in SMIB system. Appendix A.2 provides the program to calculate the parameters in Table 4.1.

Table 4.1.: Summary of the linear analysis results of the two lowest damping modes in SMIB system

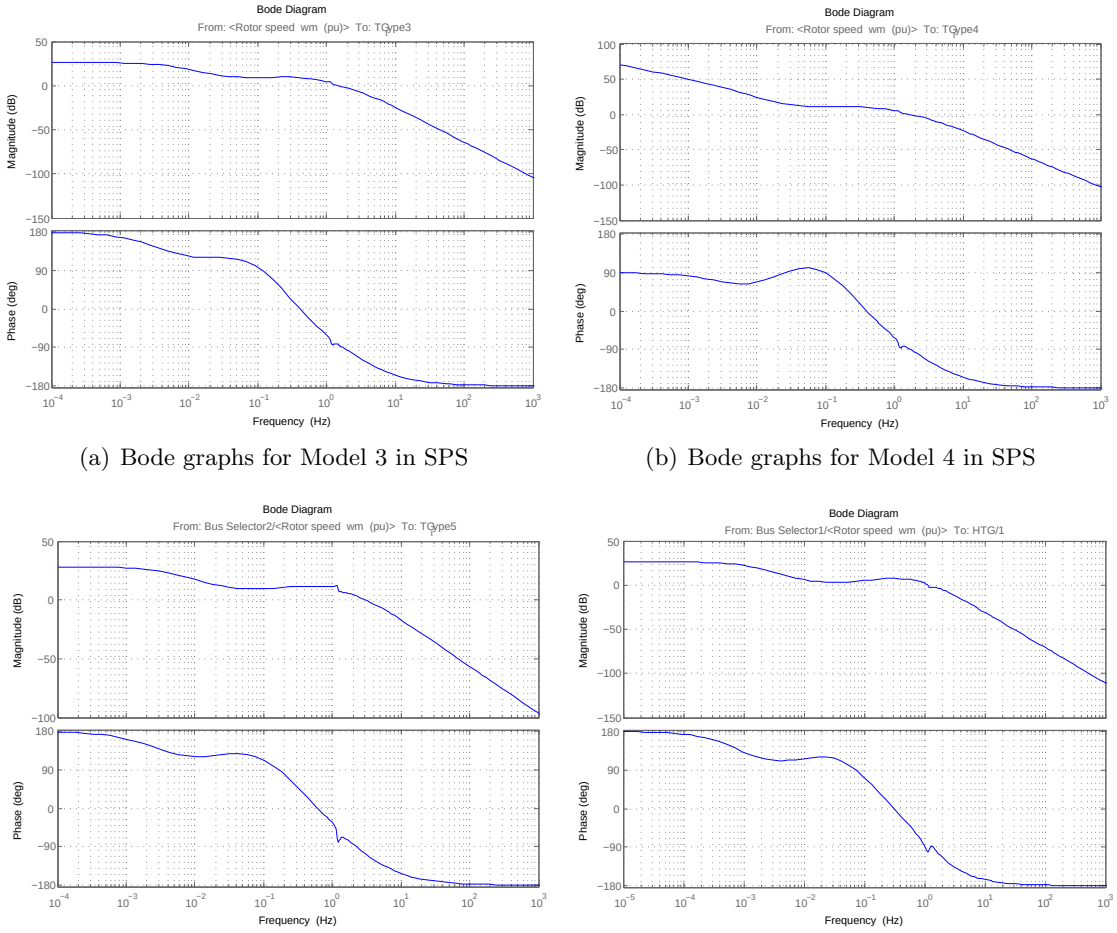
Model in system	Frequency(Hz)	Damping ratio	Eigenvalues
Model 2	0.86233	0.48048	$-2.9684 \pm 5.4182j$
	6.6960	0.12865	$-5.4581 \pm 42,072j$
Model 3	0.89782	0.10574	$-0.59983 \pm 5.6412j$
	6.6960	0.12865	$-5.4581 \pm 42,072j$
Model 4	0.89804	0.10394	$-0.58970 \pm 5.6426j$
	6.6960	0.12865	$-5.4581 \pm 42,072j$
Model 5	0.89116	0.11766	$-0.66340 \pm 5.5993j$
	6.6960	0.12865	$-5.4581 \pm 42,072j$
Model 6	0.82586	0.052253	$-0.27151 \pm 5.1890j$
	6.6960	0.12865	$-5.4581 \pm 42.072j$

As Table 4.1 illustrates, the SMIB system with Model 6 has a significantly small damping ratio for the 0.8 Hz mode, which can probably result in a high risk of instability. This may be because of its more complex nonlinear structure. Even though its low level damping does not directly affect the SMIB system, it would have a larger effect when implemented it in a more complicated system.

4.1.4. Frequency Response Analysis

The frequency response method can be less intuitive than other methods presented previously. However, it has certain advantages, especially in real-life situations such as modeling transfer functions from physical data. The frequency response is a representation of the system's response to sinusoidal inputs at varying frequencies. The output of a linear system to a sinusoidal input is a sinusoid of the same frequency but with a different amplitude and phase. The frequency response is defined as the amplitude and phase differences between the input and output sinusoids and amplitude and phase differences vary on the input sinusoids frequency. The open-loop frequency response can be used to predict the behavior of the closed-loop system [20].

Frequency responses can be described by Bode graphs. A Bode graph consists of two sub graphs, representing the magnitude and phase of $G(j\omega)$, respectively. Here $G(s)$ is the open loop transfer function of a system and $G(j\omega)$ is obtained by substituting $j\omega$ for s. The magnitude of $|G(j\omega)|$ for ω is plotted on a logarithmic scale in decibels, where $db = 20log_{10}|G(j\omega)|$ and the phase shift is in degrees. Consequently, a system's response to sinusoids can be predicted by a Bode graph. Moreover, as Fourier transforms indicate, any continuous signal can be represented by a sum of simpler trigonometric functions with different frequencies in the frequency domain. Hence, we can predict any system response to any control actions.



(a) Bode graphs for Model 3 in SPS

(b) Bode graphs for Model 4 in SPS

(c) Bode graphs for Model 5 in SPS

(d) Bode graphs for Model 6 in SPS

Figure 4.11.: Bode graphs for models in SMIB using SPS for the transfer function from ω to P_m (input as the rotor speed and output as the mechanical power)

The Bode graph function belongs to "linearize analysis" under "control design" in the Simulink environment. Compared to PSAT, one apparent advantage in SPS is that each component can be individually engaged in a Bode graph by choosing corresponding input and output ports. This method has been described in Section 3.2.3, Chapter 3 and is applied to hydro turbine and governor models in SMIB systems in SPS. Figures in 4.11 depict the frequency response analysis between the input of rotor speed and output mechanical power.

Figure 4.11 shows the Bode graphs for Model 3 to Model 6. In the Bode plots there exists a small switchback along the smooth curve. This switchback corresponds to the frequency whose damping ratio is the lowest one. Figure 4.12 to 4.14 point out the frequencies of switchback in the Bode plot, along with the corresponding eigenvalues in pole-zero map. We now investigate the possible components that cause this eigenvalue and switchback.

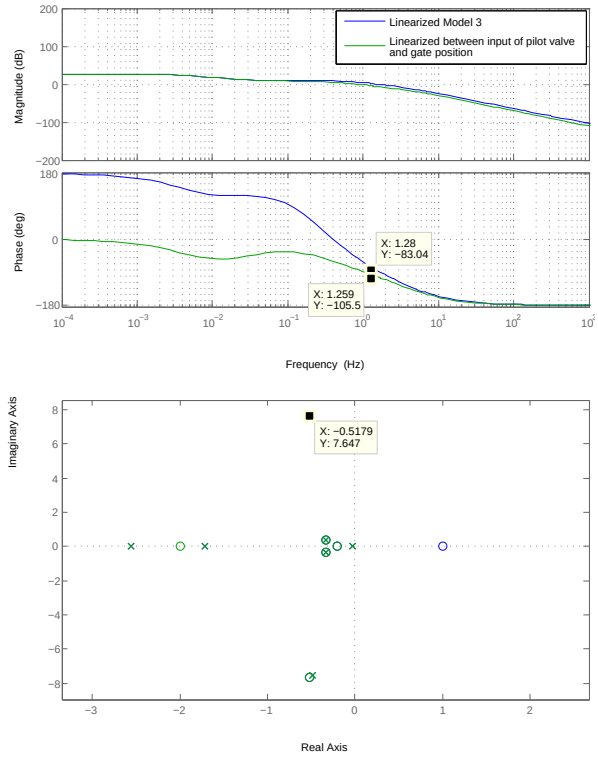


Figure 4.12.: Bode graphs and pole-zero map of Model 3 in SPS for the transfer function from ω to P_m (input as the rotor speed and output as the mechanical power)

In the Bode plot of Fig.4.12, as indicated in the legend, the blue curve is the frequency corresponding to the linearized Model 3 with its input as the rotor speed and the output as the mechanical power. While the green one is for one linearized component of Model 3 whose input is the pilot valve input and output is the gate position difference. Both of them have the same switchback frequency, which can indicate that the component in the green curve influences the switchback and corresponding eigenvalue. This frequency can also be obtained from a pole-zero map, where 'Y' represents the ω in equation (4.8), so the calculated frequency is $Y/(2*\pi) = 1.22$Hz, that is very close to the switchback frequency in the Bode plot and the deviation probably is caused by measurement error.

Similar case happens with Model 4 as shown in Fig.4.13. However, as the turbine part in both Model 3 and Model 4 is linearized, it only acts as a gain in the whole model, not creating any explicit low-damping eigenvalue. As for Model 5 (same as the case with Model 6) in Fig.4.14, the nonlinear turbine part produces the same switchback frequency that linearized Model 5 does. Therefore, nonlinear turbines contribute to the eigenvalue owning lowest damping ratio.

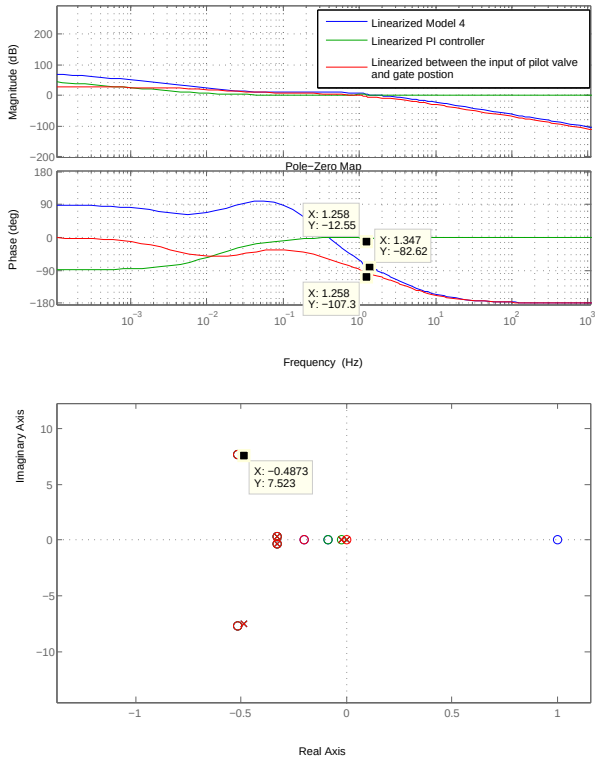


Figure 4.13.: Bode graphs and pole-zero map of Model 4 in SPS for the transfer function from ω to P_m (input as the rotor speed and output as the mechanical power)

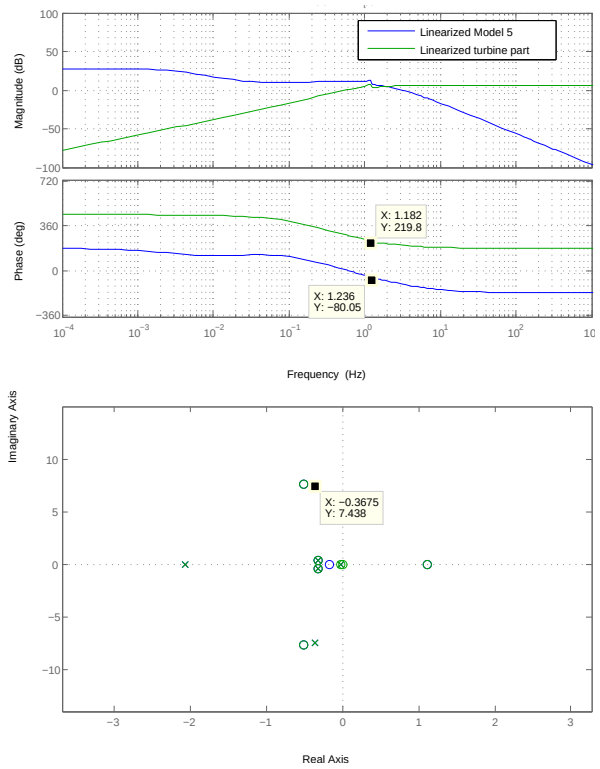


Figure 4.14.: Bode graphs and pole-zero map of Model 5 in SPS for the transfer function from ω to P_m (input as the rotor speed and output as the mechanical power)

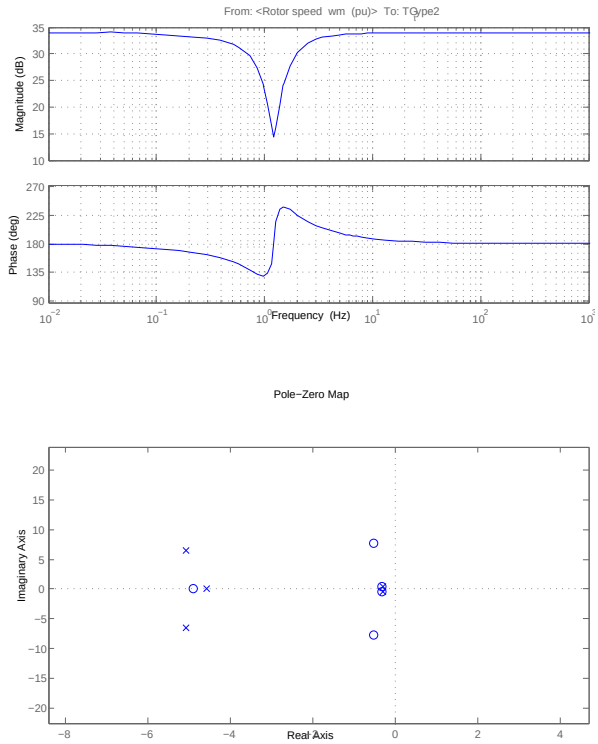


Figure 4.15.: Bode graphs and pole-zero map of Model 2 in SPS for the transfer function from ω to P_m (input as the rotor speed and output as the mechanical power)

Another point we can highlight from the zero-pole map for Model 2 is that this system is a minimum phase system as its poles and zeros are all in the left half plane. On the other hand, Model 3 to Model 6 are non minimum phase systems with one right half plane zero. This is the essential reason which highlights that Model 2 has no water hammer effect. Non minimum phase systems are slower in response because of their behavior at the start of the response. It means non minimum phase system behaves phase lag compared to minimum phase system. However this delay is just the property of water flow like Model 3 to Model 6 are showing. But Model 2 has no capacity to represent this property of hydro turbine and governor in reality, and should only be used for thermal or steam units.

4.1.5. Linear and Nonlinear Model Validation Comparison

Linear models can be used to simplify power system analysis and allow control design. This section verifies the the linear model response to a step change at hydro turbine and governor's speed reference, and finds out how well the linearized model represents the behavior of nonlinear model in the linear operating region where the nonlinear model has been linearized. Both time-domain simulations run in PSAT and Model 3 serves as hydro turbine and governor in SMIB system.

As shown in Fig.4.16, the behavior of the linearized model is considerably close to the nonlinear model. It is demonstrated that the linear model does capture the same dominant modes of the nonlinear model in the time-domain simulation. Consequently, we can conclude that linearized model is verified to represent nonlinear models. Moreover, linear analysis, including

small signal analysis, frequency response analysis are therefore accurate enough to provide the stability information for nonlinear model which can be used for control design.

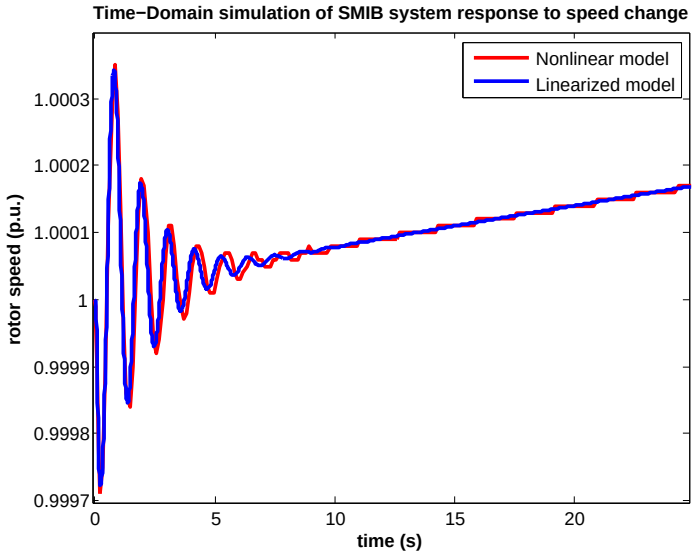


Figure 4.16.: Time-Domain simulation of SMIB system response to speed change

4.2. KTH-NORDIC32 System

4.2.1. KTH-NORDIC32 System Introduction and Parameters in PSAT

The KTH-NORDIC32 system is depicted in Fig.4.17. The overall topology is longitudinal; two large regions are connected through considerably weak transmission lines. The first region is formed by the North and the Equivalent areas located in the upper part, while the second region is formed by the Central and the South areas located in the bottom part. The system has 20 generators, 12 of which are hydro generators located in the North and the Equivalent areas, whereas the rest are thermal generators located in the Central and the South areas. In this thesis the 12 hydro turbine and governors will utilize Model 2 or Model 3, while 8 thermal turbine and governors only use Model 1. There is more generation in the upper areas while more loads congregate in the bottom areas resulting in a heavy power transfer from the northern area to the southern area, which easily leads to system oscillations [22].

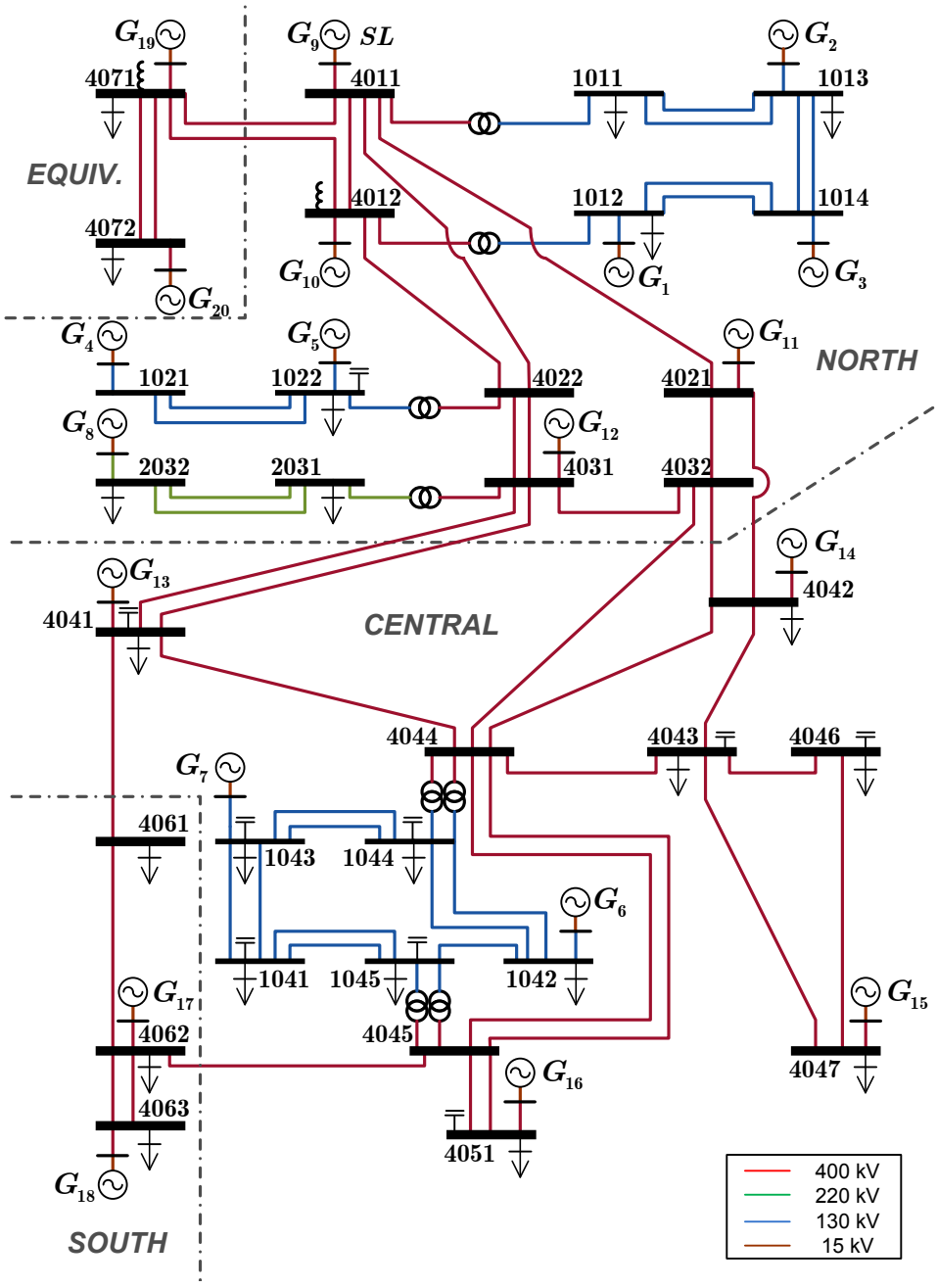


Figure 4.17.: KTH-NORDIC32 Test System (taken from [22])

The turbine and governor data (Tg.con) for the KTH-NORDIC32 System either with Model 1&2 or Model 1&3 is attached in the Appendix A.6.

4.2.2. Transient Stability Analysis

A three-phase fault is applied at "BUS1011" at t=5 s and removed at t=5.02 s. The reason why the fault is set on this bus is to decrease the fault influence on critical generators, and

at the same time so that the dominant power flow is not disturbed. This allows for a good comparison of the performance of the turbine and governor models implemented in this thesis. In the KTH-NORDIC32 System, Model 1 is utilized as the thermal turbine and governors for the 8 thermal generators, while hydro turbine and governors will take use of Model 2 or Model 3, respectively. As discussed in the previous section, Model 2 is unsuitable for representing hydro turbine and governors, and is used here for comparison purpose only. Figures 4.18 and 4.19 depict the response of the generators rotor speeds in the KTH-NORDIC32 System with Model 2 and Model 3, respectively.

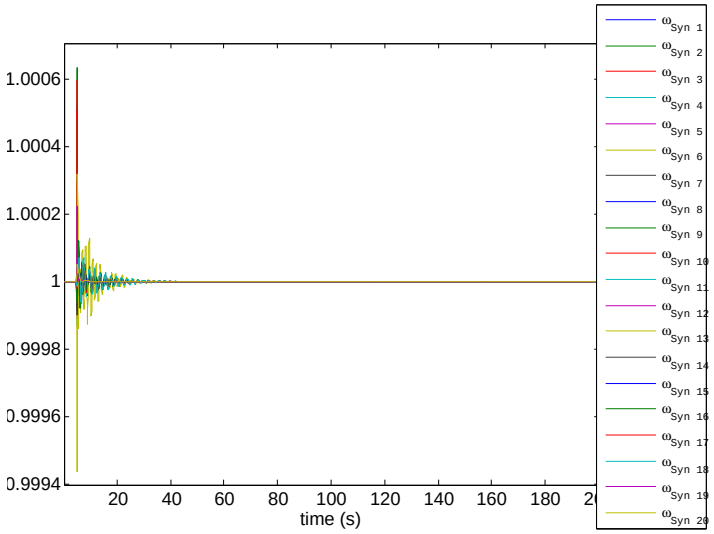


Figure 4.18.: KTH-NORDIC32 test system with Model 1 and Model 2 response to a fault

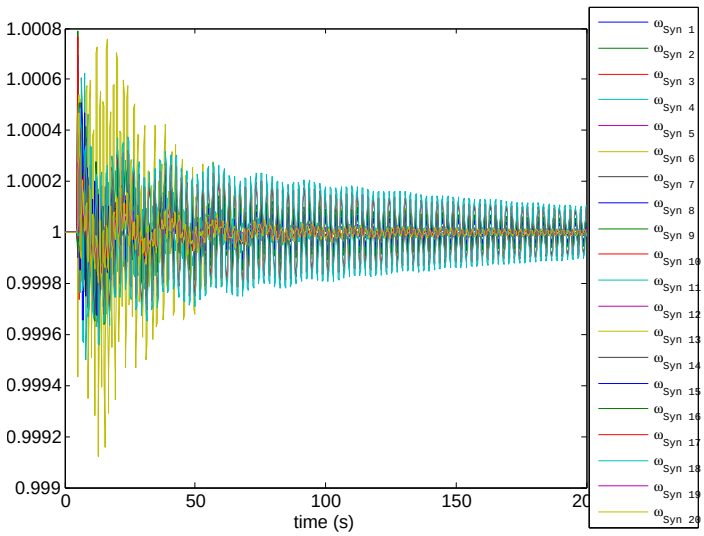
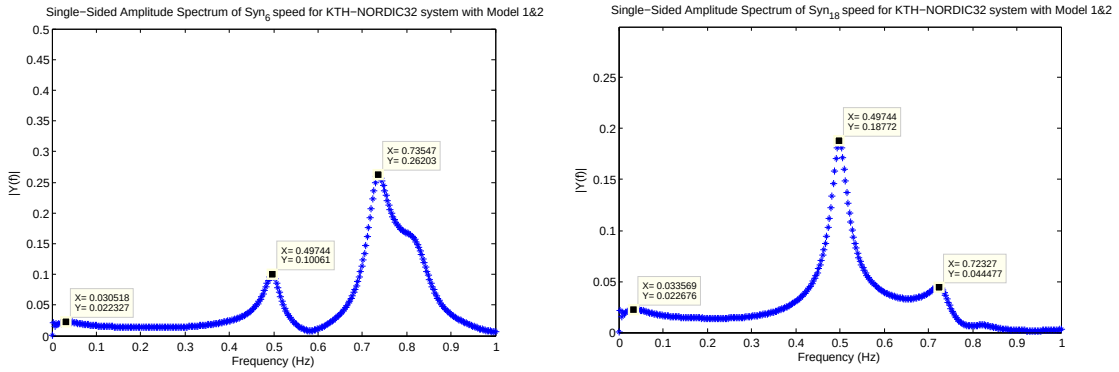


Figure 4.19.: KTH-NORDIC32 test system with Model 1 and Model 3 response to a fault

Comparing these two simulations, we can easily find out that the system with Model 2 recovers to steady state faster than that with Model 3. Moreover, in the system with Model 3,



(a) FFT on ω_6 (b) FFT on ω_{18}

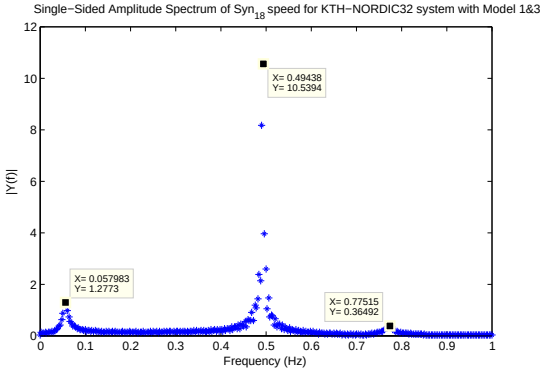
Figure 4.20.: FFT on rotor speed signals in KTH-NORDIC32 including Model 2

there are particular oscillation frequencies around a steady value "1", even when the system already gets back to steady state. In fact this is a normal and common phenomenon in large power system called system oscillations. It is system internal swing resulting from electric power flowing from one area to another in order to keep the balance of consumption and generation [23], [24], [25].

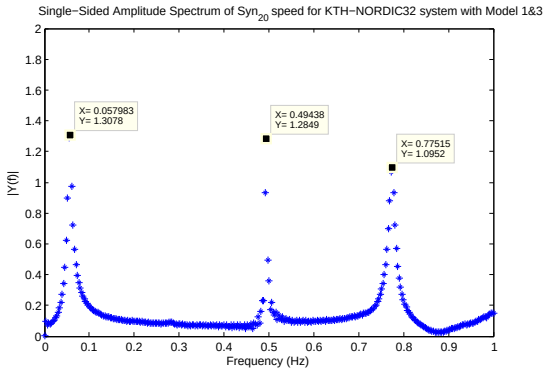
It is difficult to identify the frequencies of system oscillations by looking at the time-domain simulation signals. The fast Fourier transform (FFT) should be used to convert the time-domain signals into frequency domain, so we can determine the particular frequencies in the frequency domain. Making sure that the signal for FFT is from a fixed point time domain simulation, and that the smaller time step is, the higher resolution it can be obtained. Here we select the 5 to 200 seconds of the ω_6, ω_{18} signals from the system which includes Model 2 and 5 to 200 seconds of the ω_{18}, ω_{20} and ω_6 signals from the system which includes Model 3. FFT operates with 0.01 second time step and the associated MATLAB script can be found in Appendix A.7.

Figure 4.20 depicts the frequencies of rotor speed signals. Obviously, there are two primary frequencies around 0.5 Hz and 0.7 Hz and an inconspicuous frequency about 0.03 Hz for both ω_6 and ω_{18}. As the frequency of oscillations were caused by the fault and controlled by hydro turbine and governor (here for short we can call it "turbine/governor dynamics") normally keeps below 0.1 Hz, we can assert that the frequency 0.03 Hz in figures belongs to the "turbine/governor dynamics". The other frequencies are due to system oscillations, which are extremely weak and disappear quickly, as shown in Fig.4.18. Similarly, Fig.4.21 indicates the frequencies of system oscillations and "turbine/governor dynamics". When hydro turbine and governor models are used, the related "turbine/governor dynamics" as shown in Fig.4.21 become more prominent, indicated by an increase energy of the related mode at about 0.05 Hz.

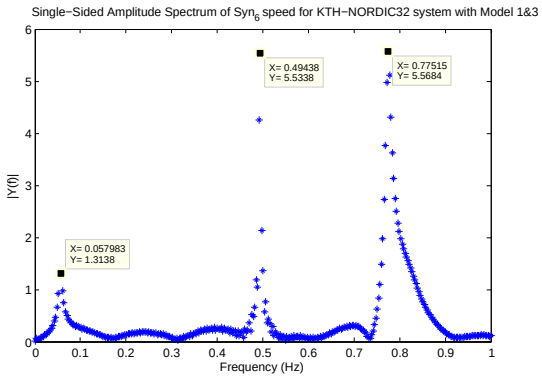
Even though the system with Model 2 behaves faster in response to the fault, we can not assert Model 3 is bad, because Model 3 represent the real behavior and properties of hydro turbine and governors, as we discussed in Sections 4.1.2 and 4.1.4.



(a) FFT on ω_{18}



(b) FFT on ω_{20}

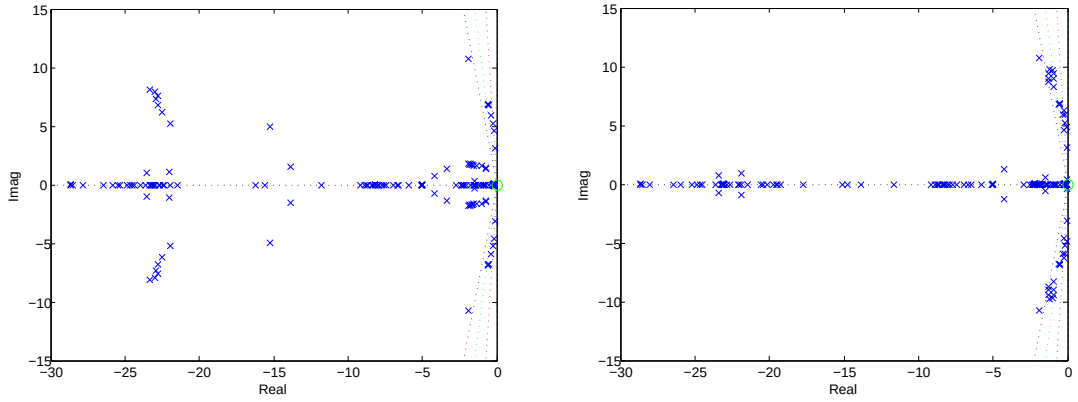


(c) FFT on ω_6

Figure 4.21.: FFT on rotor speed signals in KTH-NORDIC32 including Model 3

4.2.3. Small Signal Stability Analysis

All eigenvalues for the KTH-NORDIC32 test system, either with Model 2 or Model 3, are located in the left half plane, which indicates systems are stable. Making use of the program in Appendix A.2, we can get the two lowest damping modes in the KTH-NORDIC32 system like we did for the SMIB system. Moreover, which states are most associated with these two eigenvalues can be directly figured out from the "Eigenvalue Report" in the "eigenvalue analysis" routine from PSAT. Looking at these frequencies, they are just the system oscillation frequencies as computed in the FFT operation. This means the two lowest decay rate are just at the frequencies of system oscillations.



(a) KTH-NORDIC32 test system with Model 2 (b) KTH-NORDIC32 test system with Model 3

Figure 4.22.: Eigenvalues for KTH-NORDIC32 test system either with Model 2 or Model 3

Table 4.2.: Linear analysis results of the two lowest damping modes in KTH-NORDIC32

Model	Frequency	Damping ratio	Eigenvalues	Most associated states
Model 2	0.49866	0.035223	$-0.11043 \pm 3.1331j$	ω_{18}, δ_{18}
	0.73218	0.031801	$-0.14637 \pm 4.6004j$	ω_{6}, δ_{6}
Model 3	0.49362	0.0019950	$-0.0061875 \pm 3.1015j$	ω_{18}, δ_{18}
	0.77442	0.0082036	$-0.039918 \pm 4.8658j$	ω_{20}, δ_{20}

5 Discussion

Derived from all hydro turbine and governor models' performances in transient stability analysis in the SMIB system, newly developed Model 3 to Model 6 can represent the behavior and capacity of hydro turbine and governor. However, Model 2 can not represent the performance of hydro turbine and governors, such as "water hammer effect". The relevant and essential principle is highlighted by the pole-zero map. Model 3 to Model 6 are non minimum phase systems with one right half plane zero. And non minimum phase system behaves phase lag compared to minimum phase system, like Model 2. Moreover, this delay is just one important property of water flow.

On the other hand, normally, nonlinear devices' continuously changeable derivatives of internal states result in vast calculation and slower response to disturbances, such as the nonlinear hydro turbines in Model 5 and Model 6. However, they give a higher capacity to represent the hydro turbine and governors in reality. Furthermore, models' response to a disturbance also depends on the simulation environment. The most apparent example is Model 6, which behaves best among Model 2 to Model 6 in SPS, while acts worser in PSAT. Such great different performances in two softwares are due to model and data exchange barriers in different simulation softwares.

SPS is a Simulink-based toolbox for electromagnetic transient studies, while PSAT is MATLAB-based and aimed for power flow, optimal power flow, continuation power flow and electromechanical transients [2]. It is unfair and also unrealistic to compare these two software packages. However, PSAT is an open source software while SPS is a commercial product. Therefore there is no limitation in PSAT for carrying out our work: we can investigate, modify or improve the source code, doing whatever as we want. While SPS is not fully visible and depends on the numerical solver implementation, as a consequence, we can not change anything we want. This brings a number of difficulties and inaccuracies to exchange Model 6 and its corresponding parameters from SPS to PSAT. At the same time, we find that the simulations in SPS can not exactly start from steady state, even when initial state values are set correctly. To fix this problem, a load flow file was utilized to set initial values, but there was no improvement. For a not fully visible software, like SPS, it is too hard to figure out the internal reason behind these inconsistencies and fix them.

In a large scale power system, such as the KTH-NORDIC32, normally there are system oscillations, which results from electric power flowing from one area to another in order to keep the balance of consumption and generation. That's why the time-domain simulation of the

KTH-NORDIC32 system with Model 1 and Model 3 exists small oscillation with particular frequency around steady value. The system oscillations can not be totally fixed by hydro turbine and governor control, but can be improved in some degree through parameter tuning.

We can obtain the frequencies of system oscillations through two ways: the fast Fourier transform (FFT) and small signal stability analysis. Figures 4.20 and 4.21 are transformed from fixed time step time-domain simulation by FFT. The frequency below 0.1 Hz is product of hydro turbine and governor model, while the other two primary frequencies are due to system oscillations. As shown in Table 4.2, two lowest damping modes caused by system oscillations are provided in small signal stability analysis. Apart from frequencies, we can also get to know the corresponding eigenvalues, damping ratios and most associated states.

6 Conclusion and Future Work

6.1. Conclusion

This thesis develops four hydro turbine and governor models according to the structures of hydro turbines and mechanical turbine governors. All models' structures, realization diagrams and DAEs are provided for further implementation in Chapter 2. At the same time, it also focuses on how to implement models in two software: PSAT—an open source software—and SPS—a proprietary software. Moreover, detailed implementation steps are provided in Chapter 3.

To evaluate models features, we examine the performances of the implemented models in two power systems—a single-machine infinite-bus (SMIB) system, and the KTH-NORDIC 32 system. The evaluation of a power system's performance is concerned with the stability of that system: ie. if it remains in an equilibrium after being subjected to a disturbance [1]. Transient (large signal) stability and small signal (small disturbance) stability analysis, as well as frequency response analysis are performed in Chapter 4.

Single-machine infinite-bus system

In transient stability analysis, we present time-domain simulations of system responses to a fault and to a breaker opening both in PSAT and SPS, and also a response to a load change in PSAT. All of the three simulations demonstrate models performances response to large disturbances. For the first two, transient responses in PSAT and SPS are not as close as we expect due to model exchange barriers between the two software. The different responses to a load change between Model 2 and the other models indicate that Model 2 can not represent the behavior and capacity of hydro turbine and governor in reality, while Model 3 to Model 6 can.

In small signal stability analysis, the disturbances are considered sufficiently small for linearizion of system equations to be permissible for analysis purposes. In this case, we can compute the eigenvalues and their corresponding frequencies and damping ratios. As a lower damping ratio implies a lower decay rate and influences the dynamic system behavior more than a higher damping ratio, only two lowest damping modes need to be considered. On the other hand, the essential principle that Model 2 is unsuitable for representing hydro turbine and governor can be understood from its pole-zero map: Model 2 is minimum phase system, consequently, it can not behave phase lag and has no time delay in the response to a load change compared to other models which are non-minimum phase system.

In frequency response analysis, Bode graphs are drawn both in PSAT and SPS. In PSAT, Bode graphs are plot for the whole SMIB system with as the input and as the output. In SPS, Bode graphs are plotted for the hydro turbine and governor models between the input as rotor speed difference and the output as mechanical power. We can clearly recognize the switchbacks in the Bode graph for models in SPS, which indicate the frequencies of the lowest damping modes in different cases. The same frequencies also can be calculated from the corresponding eigenvalues shown in the pole-zero maps.

As for the linear and nonlinear model validation comparison, we can conclude that linearized model is verified to represent nonlinear model. Moreover, linear analysis, including small signal analysis, frequency response analysis are therefore accurate enough to provide the stability information for nonlinear model, which can be used for control design.

KTH-NORDIC32 system

In transient stability analysis, a three-phase fault is applied as a disturbance to the system. Even though Model 2 is unsuitable for representing hydro turbine and governor, it is used in the KTH-NORDIC32 system for comparison purposes only. In order to determine the frequencies of system oscillations, the FFT is applied to transform time domain simulation into frequency domain. In Fig.4.20 and Fig.4.21, the frequencies below 0.1 Hz are product of hydro turbine and governor models, while the other two primary frequencies are due to two lowest damping modes of system oscillations.

In small signal stability analysis, the frequencies of system oscillations can be computed together with corresponding eigenvalues and damping ratios. The calculation results are almost same with the one obtained from FFT.

6.2. Future Work

The values of parameters in hydro turbine and governor models can impact system oscillations [24]. In this case, one future work can be tuning parameters of hydro turbine and governors in order to reduce the system oscillations.

Hydro turbine and governor models may be implemented in other power systems and used for real-time simulation. What's more, we can further explore models' performance in real time simulation environment [28].

References

[1] Jan Machowski, Janusz Bialek, and Jim Bumby, *Power System Dynamics: Stability and Control*, second edition, John Wiley & Sons, 1997.

[2] F. Milano, *Power System Analysis Toolbox Documentation for PSAT version 2.1.6*, May 13, 2010.

[3] Yuwa Chompoobutrgool, and Luigi Vanfretti. "Master 's Thesis Proposal: Turbine Governor Modeling and Scripting in an Open Source Simulation Software", Sep.22, 2010.

[4] IEEE Committee Report."Dynamic Models for Steam and Hydro Turbines in Power System Studies", *IEEE Trans.*, Vol. PAS-92. pp. 1904-1905, November/ December 1973.

[5] IEEE Working Group Report. "Hydraulic Turbine and Turbine Control Models for System Dynamic Studies ", *IEEE Trans.*, Vol. PWRS-7, No.1, pp.167-179, February 1992.

[6] P. Kundur, *Power System Stability and Control*, McGraw-Hill, 1993.

[7] F. Milano, "An Open Source Power System Analysis Toolbox", *IEEE Trans.*, Vol. 20, Issue 3, pp.1199-1206, August 2005.

[8] L. Vanfretti, and F. Milano, "Application of the PSAT, an Open Source Software for Educational and Research Purposes", *IEEE*, 2007.

[9] D.G. Ramey and J.W. Skooglund, "Detailed Hydrogovernor Representation for System Stability Studies," *IEEE Trans.*, Vol. PAS-89, pp. 106-112, January 1970.

[10] Thierry Van Cutsem, IEEE Working Group on Test Systems for Voltage stability analysis, "Description, Modeling and Simulation Results of a Test System for Voltage Stability Analysis", Version2, November 2010.

[11] The MathWorks, Inc. *"http://www.mathworks.com/help/toolbox/physmod/powersys/ref/hydraulicturbineandgovernor.html"*, June,2011.

[12] F. Milano, "An Open Source Power System Analysis Toolbox", *IEEE Trans.*, November 2004.

[13] F. Milano, M. Zhou, GuanJi Hou, "Open Model For Exchanging Power System Data", IEEE PES General Meeting, 2009.

[14] GNU Operating System. *"http://www.gnu.org/"*

[15] Slackware Linux Essentials. *"http://www.slackbook.org/html/introduction-opensource .html"*

[16] Open Source Initiative. *"http://opensource.org/docs/osd"*

[17] J.M. Undrill and J.L. Woodward, "Nonlinear Hydro Governing Model and Improved Calculation for Determining Temporary Droop," *IEEE Trans.*, Vol. PAS-86, pp.443-453, April 1967.

[18] The MathWorks, Inc. "http://www.mathworks.com/products/simpower/index.html?ref= productlink"

[19] Louis N. Hannett, James W. Feltes, B. Fardanesh, "Field Tests to Validate Hydro Turbine-Governor Model Structure and Parameters", *IEEE Trans.*, Vol. 9, No.4, November 1994.

[20] RoyMech. *"http://www.roymech.co.uk/Related/Control/Frequency_Response.html"*

[21] Wikipedia. *"http://en.wikipedia.org/wiki/Bode_plot"*

[22] Y.Chompoobutrgool, and L.Vanfretti, "Linear Analysis of the KTH-NORDIC32 System", Internal Report, *http://idisk.me.com/vanfretti/Public/publications/reports/ 2011_YCLV_kthnordic32.pdf*, EPS, KTH, March 2011.

[23] F. R. Schleif and G. E. Martin, "Damping of System Oscillations with a Hydrogenerating Unit", *IEEE Trans.*, Vol. PAS-86, No. 4 April 1967.

[24] F. R. Schleif and A. B. Wilbor, "The Coordination of Hydraulic Turbine Governors for Power System Operation", *IEEE Trans.*, vol. Pas-85, no.7, pp. 750-758, July 1966.

[25] Dmitry N. Kosterev, and Carson W. Taylor, "Model Validation for the August 10,1996 WSCC System Outage", *IEEE Trans.*, Vol. 14, No. 3, August 1999.

[26] PSAT Download. *"http://www.uclm.es/area/gsee/Web/Federico//psat.htm"*

[27] Power System Element Models. *"http://wendang.baidu.com/view/29778b1d59eef8c75 fbfb313.html"*

[28] Opal-RT Simulator. *"http://www.opal-rt.com/success-story/smart-grids-simulation-kth -uses-opal-rt-simulators-to-conduct-groundbreaking-research-and-education"*

A Appendix

A.1. Matlab file for drawing Bode graphs in PSAT

```
clear all
% initialize psat
initpsat
% do not reload data file
clpsat.readfile = 0;
% runpsat('d_nordic_heavy_c3_fault.m','data')
runpsat('d_001.m','data')
%% hange the Settings
Settings.freq   = 50; % (default)
Settings.lftol  = 1e-012;
Settings.dyntol = 1e-012;
Settings.pv2pq = 1;
Settings.tstep = 0.01;
Settings.tf = 20;
Settings.fixt = 1; % fixed time step
%% sets up matrices A, B, C and D for linear analysis
runpsat('pf')
fm_abcd;
A=LA.a;
B=LA.b_tg;
C=LA.c_y(8,:);
D=LA.d_tg(8);
sys=ss(A,B,C,D);
%%figure
figure
bode(sys,{0.01,100}),grid on
```

A.2. Calculation of eigenvalues and corresponding frequencies and damping ratios

```
clear all
% initialize psat
initpsat
% do not reload data file
clpsat.readfile = 0;
```

```
runpsat('d_nordic_heavy_c3_fault.m','data')
% runpsat('d_001.m','data')
%% hange the Settings
Settings.freq   = 50; % (default)
Settings.lftol  = 1e-012;
Settings.dyntol = 1e-012;
Settings.pv2pq = 1;
Settings.tstep = 0.01;
Settings.tf = 20;
Settings.fixt = 1; % fixed time step
%% sets up matrices A, B, C and D for linear analysis
runpsat('pf')
fm_abcd;
[V, D] = eig(LA.a);
eigen = diag(D);
% damping ratio
damping = -real(D)./sqrt(real(D)^2+imag(D)^2);
damping_ratio = diag(damping);
freq = diag(imag(D)/(2*pi));
format short e
[damping_ratio  eigen    freq]
```

A.3. Time-domain simulation for linearized models in PSAT

```
clear all, clc
cd 'C:\psat_Matlab'
initpsat
clpsat.readfile = 0;
cd 'C:\psat_Matlab\tests'
runpsat('d_001.m','data')
% Change the Settings
Settings.freq   = 50; % (default)
Settings.lftol  = 1e-012;
Settings.dyntol = 1e-012;
Settings.pv2pq = 1;
Settings.lfmit = 50;
runpsat('pf')
% compute in and out variables
fm_abcd;
%%
t = 0:0.01:50;  % initialize the time vector
u = zeros(1,length(t)); %set up a disturbance
for id=1:length(t);
    u(id) = 0.02;
end
% set up the initial state variables
x0 = zeros(DAE.n,1);
w0 = DAE.x(Syn.omega(1));
% run linear time domain simulation
[dy,dt,dx] = lsim(ss(LA.a,LA.b_tg,LA.c_y,LA.d_tg),u,t,x0);
wmat = ones(length(dt),1);
dw = dx(:,Syn.omega(1));
w_lsim = dw + wmat*diag(w0);

```

```
plot(t, w_lsim,'LineWidth',2)
title('Time—Domain simulation of SMIB system response to speed ...
    change','fontsize',10,'fontweight','b')
xlabel('time (s)','fontsize',10,'fontweight','b')
ylabel('rotor speed (p.u.)','fontsize',10,'fontweight','b')
legend('Linearized model')
```

A.4. SMIB system data file in PSAT

```
Bus.con = [ ...
    1   20     1.05    0;
    2   20    1.081    0;
    3   20        1    0];
% Column Variable Description Unit
% 1  Bus number int
% 2  Voltage base kV
% 3\dagger Voltage amplitude initial guess p.u.
% 4\dagger Voltage phase initial guess rad
% 5\dagger Area number (not used yet...) int
% 6\dagger Region number (not used yet...) int

SW.con = [ ...
    2  100   20  1.081  0];
% Column Variable Description Unit
% 1  Bus number int
% 2  Power rating MVA
% 3  Voltage rating kV
% 4  Voltage magnitude p.u.
% 5  Reference Angle p.u.
% 6\dagger Maximum reactive power p.u.
% 7\dagger Minimum reactive power p.u.
% 8\dagger Maximum voltage p.u.
% 9\dagger Minimum voltage p.u.
%10\dagger Active power guess p.u.
%11\dagger Loss participation coefficient
%12\dagger Reference bus {0, 1}
%13\dagger Connection status {0, 1}

PV.con = [ ...
    1   100   20  0.9  1.05 ];
% Column Variable Description Unit
% 1  Bus number int
% 2  Power rating MVA
% 3  Voltage rating kV
% 4  Active Power p.u.
% 5  Voltage Magnitude p.u.
% 6\dagger Maximum reactive power p.u.
% 7\dagger Minimum reactive power p.u.
% 8\dagger Maximum voltage p.u.
% 9\dagger Minimum voltage p.u.
%10\dagger Loss participation coefficient —
%11\dagger Connection status {0, 1}

Line.con = [ ...
```

```
46     1   3  100   20   50   0   1   0   0.1   0;
47     2   3  100   20   50   0   0   0   0.1   0;
48     2   3  100   20   50   0   0   0   0.1   0];
49  % Column Variable Description Unit
50  % 1  From Bus int
51  % 2  To Bus int
52  % 3  Power rating MVA
53  % 4  Voltage rating kV
54  % 5  Frequency rating Hz
55  % 6  Line length km
56  % 7  - not used -
57  % 8  Resistance p.u.(omega/km)
58  % 9  Reactance p.u. (H/km)
59  %10  Susceptance p.u. (F/km)
60  %11\dagger - not used -
61  %12\dagger - not used -
62  %13\dagger Current limit p.u.
63  %14\dagger Active power limit p.u.
64  %15\dagger Apparent power limit p.u.
65  %16\dagger Connection status {0, 1}
66
67  % Column Variable Description Unit
68  % 1  From Bus int
69  % 2  To Bus int
70  % 3  Power rating MVA
71  % 4  Voltage rating of primary winding kV
72  % 5  Frequency rating Hz
73  % 6  - not used -
74  % 7  kT = Vn1/Vn2 Nominal voltage ratio kV/kV
75  % 8  Resistance p.u.
76  % 9  Reactance p.u.
77  %10  - not used -
78  %11\dagger Fixed tap ratio p.u./p.u.
79  %12\dagger Fixed phase shift deg
80  %13\dagger Current limit p.u.
81  %14\dagger Active power limit p.u.
82  %15\dagger Apparent power limit p.u.
83  %16\dagger Connection status {0, 1}
84
85  Syn.con = [ ...
86     1  991     20  50  6     0.15 0  2    0.245  0.2   5   0.031  1.91  ...
          0.42 0.2   0.66 0.061 2.8755*2 0  ;
87     2  1e+005  20  50  2     0    0  0    0.01   0     0   0      0     0 ...
             0     0    0     6   2  ];
88
89  % Column Variable Description Unit Model
90  % 1  Bus number int all
91  % 2  Power rating MVA all
92  % 3  Voltage rating kV all
93  % 4  Frequency rating Hz all
94  % 5  - Machine model - all
95  % 6  Leakage reactance (not used) p.u. all
96  % 7  Armature resistance p.u. all
97  % 8  d-axis synchronous reactance p.u. all but II
98  % 9  d-axis transient reactance p.u. all
99  %10  d-axis sub-transient reactance p.u. V.2, VI, VIII
100 %11  d-axis open circuit transient time constant s all but II
```

```
%12  d-axis open circuit sub-transient time constant s V.2, VI, VIII
%13  q-axis synchronous reactance p.u. all but II
%14  q-axis transient reactance p.u. IV, V.1, VI, VIII
%15  q-axis sub-transient reactance p.u. V.2, VI, VIII
%16  q-axis open circuit transient time constant s IV, V.1, VI, VIII
%17  q-axis open circuit sub-transient time constant s V.1, V.2, VI, VIII
%18  M = 2H Mechanical starting time (2 * inertia constant) kWs/kVA all
%19  Damping coefficient
%20\dagger Speed feedback gain gain all but V.3 and VIII
%21\dagger Active power feedback gain gain all but V.3 and VIII
%22V Active power ratio at node [0,1] all
%23\dagger Reactive power ratio at node [0,1] all
%24\dagger d-axis additional leakage time constant s V.2, VI, VIII
%25\dagger S(1.0) First saturation factor - all but II and V.3
%26\dagger S(1.2) Second saturation factor - all but II and V.3
%27\dagger nCOI Center of inertia number int all
%28\dagger Connection status {0, 1} all

Exc.con = [ ...
   1    2    11.5  -11.5  400  0.1  0.45  1  0.01  1  0.001  0.0006  0.9  1];
% Column Variable Description Unit
% 1  Generator number int
% 2  Exciter type int
% 3  Maximum regulator voltage p.u.
% 4  Minimum regulator voltage p.u.
% 5  Ka Amplifier gain p.u./p.u.
% 6  Ta Amplifier time constant s
% 7  Kf Stabilizer gain p.u./p.u.
% 8  Tf Stabilizer time constant s
% 9  Ke Field circuit integral deviation p.u./p.u.
%10  Te Field circuit time constant s
%11  Tr Measurement time constant s
%12  Ae 1st ceiling coefficient -
%13  Be 2nd ceiling coefficient -
%14\dagger Connection status {0, 1}

Tg.con = [ ...
%   1  1  1  0.04   1  0  5    0.2   5   0.01   6     1   0] ;
%   1  2  1  0.04   1  0  5    5     0     0    0     1   0] ;
1  3  1  0.008  1  0  0.1  -0.1  0.04  5.0  0.04  0.3  1  0.5  1  1.5  1  1];
%   1  4  1  0.008  1  0  0.1  -0.1  0.04  5.0  0.04  0.3  1  0.5  1  1.5  ...
     1  1.163 0.105 1];
%   1  5  1  0.2    1  0  0.1  -0.1  0.05  1    0.04  3   0.5  1];
%   1  6  1  10/3  1  0  0.1  -0.1  0.07  2.67  0.1  1.163   0.105  0  ...
     0.01  0.04  1];

Fault.con = [ 3  100  20  50  20  20.02  0.15  0];
% Column Variable Description Unit
% 1 Bus number int
% 2 Sn Power rating MVA
% 3 Vn Voltage rating kV
% 4 fn Frequency rating Hz
% 5 tf Fault time s
% 6 tc Clearance time s
% 7 rf Fault resistance p.u.
% 8 xf Fault reactance p.u.

```

```
% Breaker.con = [ 3  3  100   20   50   1   20   20.02];
% Column Variable Description Unit
% 1  Line number int
% 2  Bus number int
% 3  Power rating MVA
% 4  Voltage rating kV
% 5  Frequency rating Hz
% 6  Connection status {0, 1}
% 7  First intervention time s
% 8  Second intervention time s
% 9\dagger Apply first intervention {0, 1}
%10\dagger Apply second intervention {0, 1}

Settings.t0 = 0;
Settings.tf = 50;
```

A.5. Parameters of components in SMIB system in SPS

Table A.1.: Parameters of Synchronous Machine in SMIB in SPS

Parameters		Value
Nominal power	$P_n(VA)$	991e6
Line-to-line voltage	$V_n(Vrms)$	20000
Frequency	$f_n(Hz)$	50
Reactances	$X_d(s)$	2.0
	$X'_d(s)$	0.245
	$X''_d(s)$	0.2
	$X_q(s)$	1.91
	$X'_q(s)$	0.42
	$X''_q(s)$	0.2
	$X_l(s)$	0.15
d axis time constants		Open-circuit
q axis time constants		Open-circuit
Time constants	$T'_{do}(s)$	5
	$T''_{do}(s)$	0.031
	$T'_{qo}(s)$	0.66
	$T''_{qo}(s)$	0.061
Stator resistance	$Rs(pu)$	0
Inertia coefficient	$H(s)$	2.8755
Friction factor	$F(pu)$	0
Pole pairs	$p()$	2

Table A.2.: Parameters of Excitation System in SMIB in SPS

Parameters		Value
Low-pass filter time constant	$T_r(s)$	0.001
Regulator gain	$K_a()$	400
Regulator time constant	$T_a(s)$	0.1
Exciter	K_e	0.01
	T_e	1
Transient gain reduction	$T_b(s)$	0
	$T_c(s)$	0
Damping filter gain	$K_f()$	0.45
Damping filter time constant	$T_f(s)$	1
Regulator output limits	$Ef_{min}, Ef_{max}(pu)$	-11.5, 11.5
Regulator gain	$K_p()$	0
Initial values of terminal voltage	$V_t0(pu)$	1
Initial values of field voltage	$V_f0(pu)$	3.36098

Table A.3.: Parameters of Three-Phase Transformer in SMIB in SPS

Parameters		Value
Nominal power	$P_n(VA)$	991e6
Frequency	$f_n(Hz)$	50
Winding 1 parameters	$V1Ph-Ph(Vrms)$	20000
	$R_1(pu))$	0
	$L_1(pu)$	0.05
Winding 2 parameters	$V2Ph-Ph(Vrms)$	20000
	$R_2(pu))$	0
	$L_2(pu)$	0.05
Magnetization resistance	$R_m(pu)$	1
Magnetization inductance	$R_m(pu)$	1

A.6. Tg.con data for KTH-NORDIC32 system

```
% KTH-NORDIC32 system with Turbine and Governor Model 1&2.
Tg.con = [ ...
% 1   2   3   4       5       6     7    8     9   10     11  12
     1   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     2   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     3   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     4   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     5   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     6   1   1   0.04    0.95    0     5    0.2   5   0.01   6   1 ;
     7   1   1   0.04    0.95    0     5    0.2   5   0.01   6   1 ;
     8   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     9   2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     10  2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
     11  2   1   0.04    0.95    0     0.5  3     0   0      0   1 ;
```

```
15      12  2   1   0.04    0.95    0       0.5  3     0   0       0   1 ;
16      13  1   1   0.04    0.95    -0.5    5    0.2   5   0.01    6   1 ;
17      14  1   1   0.04    0.95    0       5    0.2   5   0.01    6   1 ;
18      15  1   1   0.04    0.95    0       5    0.2   5   0.01    6   1 ;
19      16  1   1   0.04    0.95    0       5    0.2   5   0.01    6   1 ;
20      17  1   1   0.04    0.95    0       5    0.2   5   0.01    6   1 ;
21      18  1   1   0.04    0.95    0       5    0.2   5   0.01    6   1 ;
22      19  2   1   0.08    0.95    0       0.5  3     0   0       0   1 ;
23      20  2   1   0.08    0.95    0       0.5  3     0   0       0   1 ;
24      ];
```

```
1  % KTH-NORDIC32 system with Turbine and Governor Model 1&3.
2  Tg.con = [ ...
3   % 1   2   3   4     5      6     7      8     9      10   11    12   13  ...
       14  15  16 17 18
4       1   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
5       2   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
6       3   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
7       4   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
8       5   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
9       6   1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
10      7   1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
11      8   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
12      9   3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
13      10  3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
14      11  3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
15      12  3   1  0.2    1      0     0.1  -0.1 0.04  5   0.04  0.3  1  0.5  ...
           1  1.5  1  1;
16      13  1   1  0.04   0.95   -0.5  5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
17      14  1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
18      15  1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
19      16  1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
20      17  1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
21      18  1   1  0.04   0.95   0     5    0.2   5   0.01   6   1   0    0   ...
           0   0   0  0;
22      19  3   1  0.2    1      0     0.1  -0.1 0.04  5   0.08  0.3  1  0.5  ...
           1  1.5  1  1;
23      20  3   1  0.2    1      0     0.1  -0.1 0.04  5   0.08  0.3  1  0.5  ...
           1  1.5  1  1;
24      ];
```

A.7. FFT algorithm for analyzing rotor speed signals

```
Fs = 1/0.01;                    % Sampling frequency
T = 1/Fs;                      % Sample time
L = length(omega6_13(:,2));                    % Length of signal
t = (0:L-1)*T;                 % Time vector
y0 = omega6_13(:,2)-mean(omega6_13(:,2));
y = y0*(180/pi)*1000;
NFFT = 2^nextpow2(L); % Next power of 2 from length of y
Y = fft(y,NFFT)/L;
f = Fs/2*linspace(0,1,NFFT/2+1);
% Plot single-sided amplitude spectrum.
figure
stem(f,2*abs(Y(1:NFFT/2+1)))
title('Single-Sided Amplitude Spectrum of Syn_{6} speed for KTH-NORDIC32 ...
    system with Model 1&3')
xlabel('Frequency (Hz)')
ylabel('|Y(f)|')
axis([0 1 0 5])
```