University of South-Eastern Norway

www.usn.no

FMH606 Master's Thesis 2019
Electrical Power Engineering

Model Fitting and State Estimation for Thermal Model of Synchronous Generator

Madhusudhan Pandey

Faculty of Technology, Natural Sciences and Maritime Sciences
Campus Porsgrunn

University of South-Eastern Norway

www.usn.no

Course: FMH606 Master's Thesis 2019
Title: *Model Fitting and State Estimation for Thermal Model of Synchronous Generator*
Pages: *119*
Keywords: *Thermal model, Unscented Kalman Filter, Ensemble Kalman Filter*

Student: *Madhusudhan Pandey*
Supervisor: *Bernt Lie, co-supervisor Thomas Øyvang*
External partner: *Skagerak Kraft*
Availability: *Open*

Summary:
The power factor of a synchronous generator, in a European hydropower generation, is constrained to [0.85, 0.95], and for Norway, it is generally below 0.86. A higher power factor means more exploited power (active power), and currents through the generator; and vice-versa. However, relaxation on the constrained power factor for higher active power generation should be balanced with thermal heating due to more winding currents through the generator which eventually influences the generator lifetime.

Thus, the measure of temperature evolution in generator metals are vital if the relaxation of power factor is allowed. In this thesis work, we are developing several thermal models for air-cooled synchronous generator based on constant, and temperature dependent resistance and specific heat capacity for metals and fluids (air, water). The developed models are then analyzed, and state estimation algorithms are applied for air and metals temperature estimation. The state estimation algorithms, Unscented Kalman Filter, and Ensemble Kalman Filter are compared. Based on computational speed Unscented Kalman Filter performs better, however, based on estimation accuracy Ensemble Kalman Filter performs better.

Preface

This report, as a Master's Thesis report, contains research works on model analysis and state estimation of air-cooled synchronous generator and submitted to the Department of Electrical, IT and Cybernetics, University of South-Eastern Norway in partial fulfillment of the requirements for Master of Science in Electrical Power Engineering. This thesis work has been started in January 2019. Out of several tasks that should be carried out as given in Appendix A, the first task explaining about the thermal model extension with electric model constraining field voltage and terminal currents has not been carried out because of availability of electrical power and voltage data for calculating terminal currents. The report mainly contains model developments, analysis, and state estimation.

The thesis work would not have been successful without continuous supervision I received from Prof. Bernt Lie. I would like to have special gratitude towards him. It was my pleasure to work with him and would love to work in future if any. I would also like to express my gratitude towards Associate Prof. Thomas Øyvang for his help and guidance. My special thanks go to my friend, Sabin who helped me with SolidWorks sketch. The report would not have been meticulously examined without working together with my classmate Prabesh. My special reverence goes to my family members who have always encouraged me for further education. A big salutation goes to Julia Community who are trying to develop such an elegant and lovely product for mankind.

Porsgrunn, 14th May 2019

Madhusudhan Pandey

Contents

List of Figures

List of Tables

Abbreviations

CFD	Computational Fluid Dynamics
DAE	Differential Algebraic Equation
EnKF	Ensemble Kalman Filter
FEM	Fintie Element Method
ODE	Ordinary Differential Equation
RMSE	Root Mean Square Error
UKF	Unscented Kalman Filter

List of symbols

The symbols are grouped together into symbols used in generator models, control systems, and Kalman filtering algorithms. Each group is separated by a horizontal line. The given units are solely based on describing mathematical models for this thesis report.

Symbol	Explanation
α_{Cu}	Temperature coefficients of resistance for copper $\left\{^{\circ}\text{C}^{-1}\right\}$
$a+bT$	Linear approximation of 7-coefficients form of temperature dependent on molar heat capacities
$\hat{c}_p, \hat{c}_p(T)$	Constant and temperature dependent specific heat capacity $\{\text{kJ/kg/K}\}$
$\tilde{c}_p, \tilde{c}_p(T)$	Constant and temperature dependent molar heat capacity $\{\text{kJ/kg/mol}\}$
$H, \dot{H}, \hat{H}$	Enthalpy, enthalpy rate, and specific enthalpy $\{\text{kJ, kJ/s, kJ/kg}\}$
M	Molar mass $\{\text{g/mol}\}$
$m, \dot{m}$	Mass of substance and mass flow rate $\{\text{kg, kg/s}\}$
p_{a}	Atmospheric pressure $\left\{\text{N/m}^2\right\}$
$\dot{Q}, \dot{Q}^{\sigma}$	Heat flow rate and power due to a source $\{\text{kW, kW}\}$
R_r, R_s	Resistance of rotor copper and stator copper $\{\text{k}\Omega, k\Omega\}$
$\mathscr{R}$	Universal gas constant $\{\text{J/K/mol}\}$
T	Temperature $\{\text{K}\}$
U	Internal energy of a system $\{\text{J}\}$
$V, \hat{V}$	Volume and specific volume of substance $\{\text{L, L/kg}\}$
x, y, z, u, θ	States, outputs, algebraic variables, inputs, and parameters
$\frac{dx}{dt}$	State derivatives
$f(.), g(.), h(.)$	Functions of state derivatives, outputs and algebraic variables
A, B, C, D	State, control, output, and feedforward matrices
$\Delta t, T_{\text{sim}}$	Simulation time step and total simulation time
λ, τ	Eigenvalues and time constant of a system
J, F, S	Jacobian in states, parameter derivatives and vector of parameter sensitivities
$x, \bar{x}, \hat{x}$	State vector, its mean, and its estimate
x_k	k^{th} component of vector x
$\hat{x}_{k\|k-1}$	*a priori* estimate of x_k on all prior measurements except at time t_k

$\hat{x}_{k\|k}$	*a posteriori* estimate of x_k on all prior measurements including at time t_k
$w \sim \mathcal{N}(\bar{w}, \mathcal{W})$	w is a normal distribution with mean $\bar{w}$ and covariance $\mathcal{W}$
X	State covariance
w, v	Process noise and measurement noise vector
$\mathcal{W}, \mathcal{V}$	Process and measurement noise covariance
ε	Innovation/error between measurement and estimate
K	Kalman gain
$\mathcal{E}, Z$	Innovation covariance, and cross-covariance between x and y

1 Introduction

This chapter deals with introductory concepts with a section including background, scope, and outline of the thesis report. In a way, this chapter deals with a big picture of the task that is carried out.

1.1 Background

In a European hydropower generation, the power factor of a synchronous generator is constrained to the range $[0.85, 0.95]$ [1]. And for a Norwegian hydropower system, the power factor is constrained below 0.86 [2]. There is always a tradeoff between choosing a higher and lower power factor normally in a hydropower system. A higher power factor means less reactive power (unexploited power) and more active power (exploited power) through the system, however, results in more currents, resulting in more heating of the hydro-generators. Thus, relaxation on constrained power factor, make possibilities on the exploitation of more active power in case of operational challenges, however; this should be balanced with thermal heating and, for the lifetime of the generator.

A brief review of modern thermal analysis of electrical machines is provided in [3]. The thermal models are based particularly on lumped-parameter thermal network (LPTN) [4, 5, 6], finite element analysis (FEM), and computational fluid dynamics (CFD) [7, 8]. A totally enclosed water-cooled thermal model of synchronous machines for an electric vehicle has been purposed in [9]. Recently, a totally enclosed thermal model of air-cooled hydro generator has been developed in [10] using closed-loop heat exchanger model for cooling heated air from the outlet of generator. The similar model has also been developed in [11]. Our[1] research study is primarily focused to further developed the model as tasks described in Appendix A.

1.2 Scope and Objectives

A mathematical model of the air-cooled synchronous generator is being implemented. The model developed in [11] is further analyzed with temperature dependent resistances

[1]Here *our* and *we* is referred to author and readers.

Table 1.1: Julia packages used

Packages	Version	Packages	Version
BoundaryValueDiffEq	2.2.3	LaTeXStrings	1.0.3
CSV	0.4.3	NLsolve	4.0.0
ControlSystems	0.5.1	OMJulia	0.0.0
DataFrames	0.17.1	Optim	0.18.1
DiffEqParamEstim	1.6.0	OrdinaryDiffEq	5.5.0
DiffEqSensitivity	3.2.0	Plots	0.24.0
DifferentialEquations	6.3.0	Polynomials	0.5.2
Distributions	0.17.0	PyPlot	2.8.1
ForwardDiff	0.10.3	Random	—
IJulia	1.18.1	Sundials	3.3.0
JLD	0.9.1	Taro	0.7.0

and specific heat capacities of metals, air, and water. The scope of this thesis lies in implementing state estimation algorithms, mainly Unscented Kalman Filter (UKF), and Ensemble Kalman Filter (EnKF) and comparing them. Before implementing these algorithms, models are simulated, linearized, checked for the stability of the linearized model, checked for controllability and observability. Similarly, local parameter sensitivity analysis is done. Furthermore, few model parameters are optimized for better fitting of model with experimental data from [12].

1.3 Software Requirements

For implementing mathematical models, plotting of results, algorithms, loading data files, and for other common tasks we are using, open-source[2] programs, OpenModelica[3], OMJulia[4], and Julia language[5] with several packages. A few important packages that we are using for accomplishing this thesis research are given in Table 1.1.

[2]https://en.wikipedia.org/wiki/Open_source

[3]It is an open-source environment for modeling and simulation. https://openmodelica.org . The version used is "OMEdit v1.14.0-dev-44-gd66d325c (64-bit)"

[4]https://www.openmodelica.org/doc/OpenModelicaUsersGuide/latest/omjulia.html

[5]https://julialang.org/

1.4 Outline of Report

In Chapter 2 we will be discussing the overview of a case study hydrogenerator at Åbjøra, Norway. The detailed design consideration is not done, however, the general layout and a basic outline on how a totally air enclosed hydro generator is set up is presented.

Chapter 3 describes the mathematical model, Differential Algebraic Equation (DAE) and Ordinary Differential Equation (ODE) of air-cooled synchronous generator. The models are developed considering constant temperature and temperature dependent resistance of rotor and stator copper of the hydrogenerator, and specific heat capacities of metals, air, and water.

We will then present the simulation of DAE and ODE models in Chapter 4. Similarly, Chapter 5 is linearization, stability, controllability, and observability of our generator models while Chapter 6 is on parameters sensitivity analysis, an overview of experimental data, simulation versus real measurements and parameters optimization. Chapter 7 will discuss state estimation algorithms, UKF and EnKF.

Chapter 8 will be for results and discussion. Future works and conclusion are presented in Chapter 9 and 10 respectively.

Similarly, Appendix A contains task descriptions that should be carried out. Appendix B list the model equations for all DAE models. Furthermore, Appendix C contains the code listing in Julia and Modelica. Modelica codes are written in OpenModelica editor while Julia codes are written in Jupyter notebook. And finally, a submitted draft paper for SIMS 2019 is given in Appendix D.

2 Overview of Air-cooled Synchronous Generator

The case study, for implementing a mathematical model of air-cooled synchronous generator is taken from a vertically mounted 103 MVA air-cooled hydro generator, Åbjøra, Norway. The machine data is given in Table 2.1.

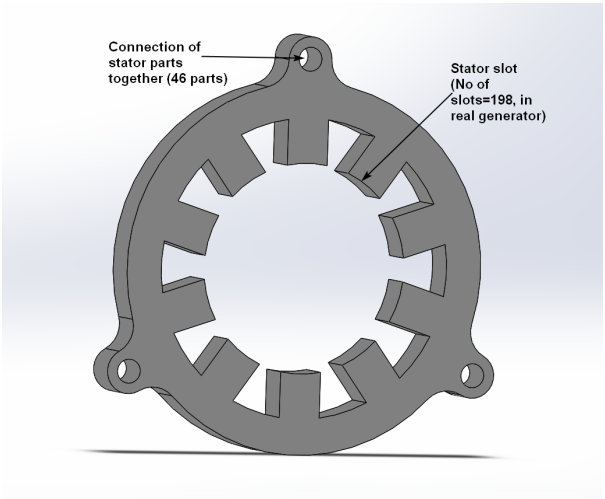
Figure 2.1 shows the stator and rotor configuration. The stator iron consists of 46 slices of iron core, where Figure 2.1(a) shows one slice among 46 slices in real stator of hydro generator at Åbjøra (We are only showing 10 slices and figures are not on scale). These slices are together connected with nuts and bolt to create a stator iron which consists of 198 slots in total for stator windings (We are only showing 10 slots). And, thus, a gap to circulate hot air coming from the air gap of rotor and stator is created as shown in the side view of stator iron in Figure 2.1(b). Figure 2.1(c) shows an isometric view of the stator and Figure 2.1(d) shows the salient pole rotor configuration[1].

Similarly, Figure 2.2 shows the overall setup consisting of the rotor and stator enclosed inside a frame. The heat exchanger is a counter-current type and mounted at the top of the frame as shown in the figure.

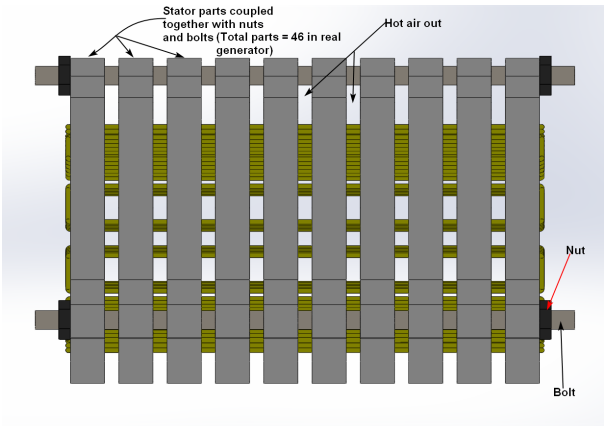
[1]It should be noted that the all design consideration is not taken here. The purpose of these diagrams is to show the basic outline for the typical hydro generator setup at the plant.

Table 2.1: Machine data from Åbjøra hydrogenerator

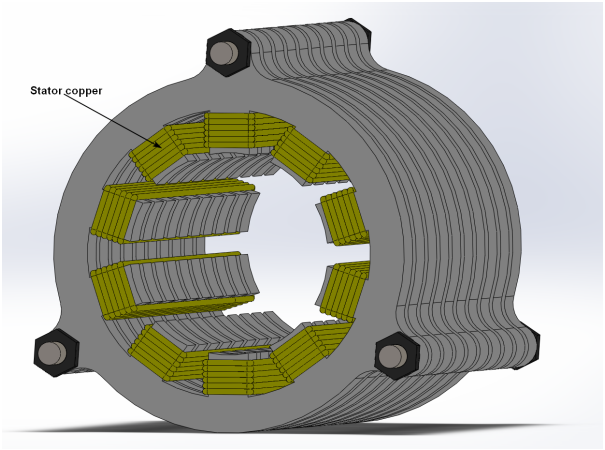
Quantity	Symbol	Value	Unit
Rated power	S_n	103	MVA
Rated power factor	$\cos\phi_\mathrm{n}$	0.9	-
Rated voltage	V_t	11	kV
Rated current	I_t	5406	A
Rated field current	I_f	1065	A
Stator bore	D	3.4	m
Stator gross iron length	l_g	2.2	m
Number of slots per pole and phase	q_s	$5\frac{1}{2}$	-
Number of parallel current paths	c_s	3	-
Number of conductors per slot	n_s	2	-
Number of field turns per pole	n_f	$40\frac{1}{2}$	-
Type of strand transposition	-	Roebel bar	-
Insulation temperature class	-	F	-
Frequency	f	50	Hz
Number of polepairs	p	6	-
Synchronous reactance	x_d	1.087	$p.u$



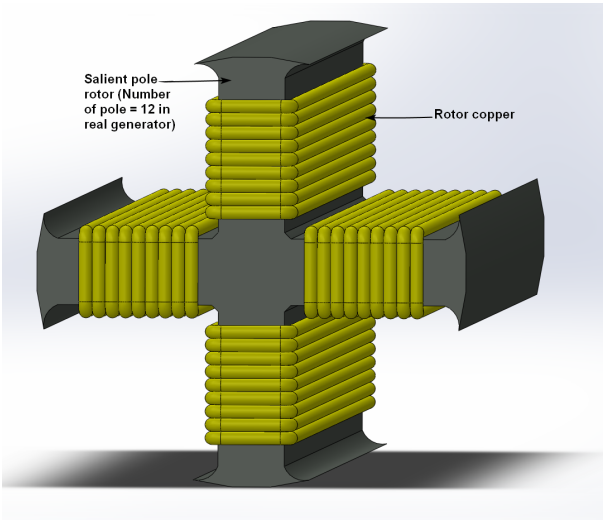
a) Part of a stator



b) Side view of stator

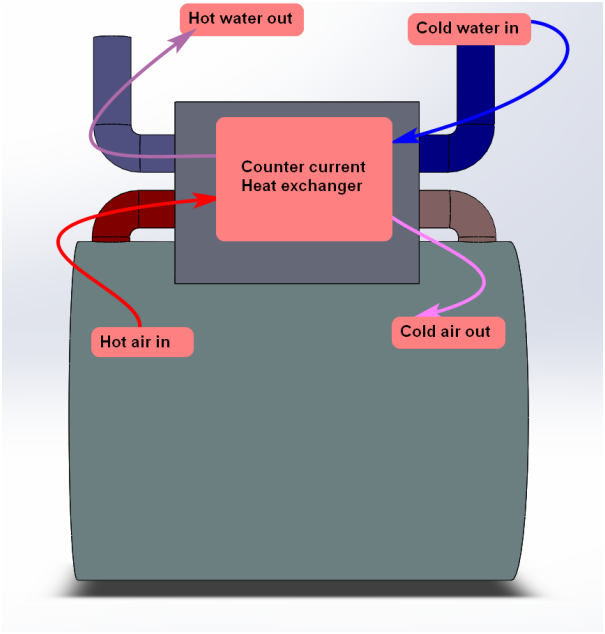


c) Isometric view of stator

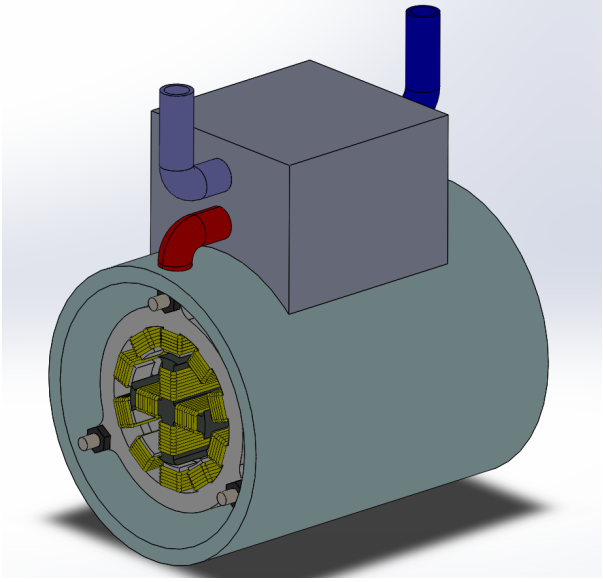


d) Salient pole rotor

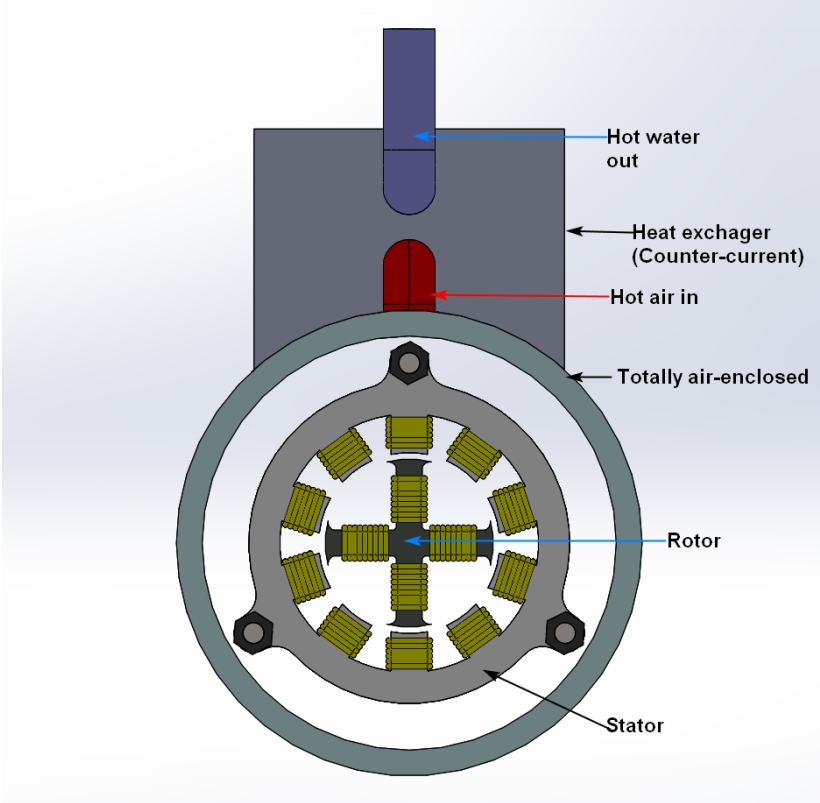
Figure 2.1: The stator and rotor configuration (Not on scale or design consideration)



a) Heat exchanger



b) Isometric view of overall setup



c) Overall setup front view

Figure 2.2: The overall enclosed configuration of air-cooled hydrogenerator (Not on scale or design consideration)

3 Thermal Model of Air-cooled Synchronous Generator

In a recent work of Øyvang et al. [10] a thermal model for a totally enclosed air-cooled synchronous generator has been developed. Similar model, with more general configuration and more efficient heat exchanger, has been developed in [11] using concepts and notations from [13]. Figure 3.1 shows the operation of the thermal model of the air-cooled synchronous generator. The cold air out of the heat exchanger is blown by a fan into the rotor/stator air gap. The air is heated by heat flow from the rotor, air gap windage and bearing friction. Furthermore, the air is forced into the iron cores which then gets heated by the heat flow from the iron cores. The heated air is now collected at the stator's outlet and passed through the heat exchanger. The heated air is then cooled at the desired temperature using continuous cold water circulation in the heat exchanger and then feed again into the air gap as a continuous process. The heat exchanger is feed with cold water, with mass flow rate $\dot{m}_{\mathrm{w}}$ at temperature $T_{\mathrm{w}}^{\mathrm{c}}$. The air mass flow rate is $\dot{m}_{\mathrm{a}}$ with temperature $T_{\mathrm{a}}^{\mathrm{h}}$ a at stator outlet and heat exchanger entry. The rotor copper heat source, $\dot{Q}_{\mathrm{r}}^{\sigma}$, is due to rotor field current, I_{f}. Similarly the stator copper heat source, $\dot{Q}_{\mathrm{s}}^{\sigma}$, is due to stator terminal current I_t. $\dot{Q}_{\mathrm{Fe}}^{\sigma}$ is stator iron heat source, and $\dot{Q}_{\mathrm{f}}^{\sigma}$ is heat generated due to friction in stator/rotor air gap. The thermal operation of air-cooled synchronous generator is mainly influenced by $\dot{m}_{\mathrm{w}}$, $\dot{m}_{\mathrm{a}}$, $T_{\mathrm{w}}^{\mathrm{c}}$, $\dot{Q}_{\mathrm{Fe}}^{\sigma}$, $\dot{Q}_{\mathrm{f}}^{\sigma}$, I_{t} and I_{f}. It is of interest to see the behavior of evolution in the rotor, stator and iron core temperatures indicated by T_{r}, T_{s} and T_{Fe}, respectively. The functional diagram for the air-cooled synchronous generator is shown in Figure 3.2 relating inputs and outputs.

The rotor copper heat source, $\dot{Q}_{\mathrm{r}}^{\sigma}$, is considered to be resistive heating with 10% magnetization loss. Similarly, the stator copper heat source is considered due to joules heating of stator resistance. The stator iron heat source, $\dot{Q}_{\mathrm{Fe}}^{\sigma}$ is considered to be constant and independent of operating conditions. The air gap heating rate, $\dot{Q}_{\mathrm{f}}^{\sigma}$is considered to be 80% of power loss due to friction at air gap, $\dot{W}_{\mathrm{f}}$.

$$\dot{Q}_{\mathrm{r}}^{\sigma} = 1.1 R_{\mathrm{r}} I_{\mathrm{f}}^{2} \tag{3.1}$$

$$\dot{Q}_{\mathrm{s}}^{\sigma} = 3 R_{\mathrm{s}} I_{\mathrm{t}}^{2} \tag{3.2}$$

$$\dot{Q}_{\mathrm{f}}^{\sigma} = 0.8 \dot{W}_{\mathrm{f}} \tag{3.3}$$

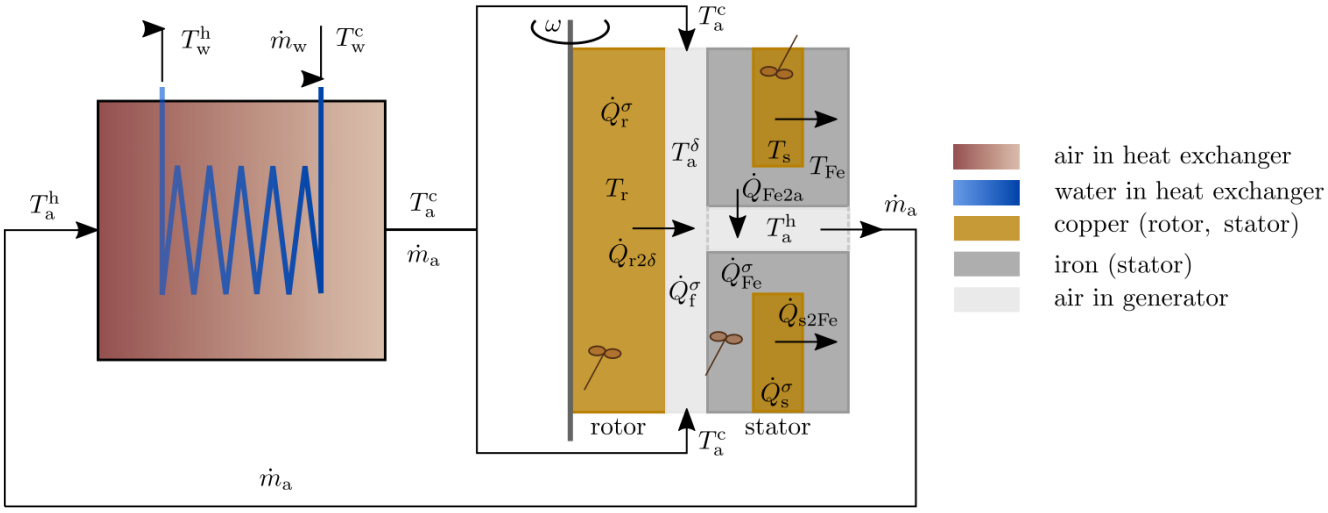


Figure 3.1: Operation of the thermal model of the air-cooled synchronous generator [11].

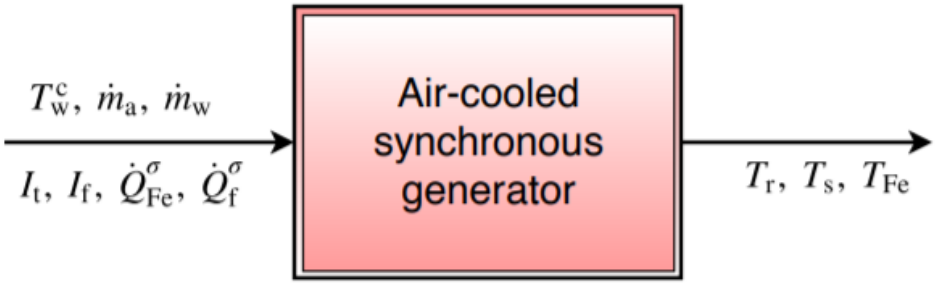


Figure 3.2: Functional diagram for air-cooled synchronous generator.

The rotor to air gap power loss $\dot{Q}_{r2\delta}$, stator copper to stator iron power loss $\dot{Q}_{s2Fe}$, and finally, power loss from stator iron to hot air out of the generator $\dot{Q}_{Fe2a}$ as shown in Figure 3.1 is given as,

$$\dot{Q}_{r2\delta} = \mathscr{U} A_{r2\delta} \left(T_r - T_a^\delta \right) \tag{3.4}$$

$$\dot{Q}_{s2Fe} = \mathscr{U} A_{s2Fe} \left(T_s - T_{Fe} \right) \tag{3.5}$$

$$\dot{Q}_{Fe2a} = \mathscr{U} A_{Fe2a} \left(T_{Fe} - T_a^h \right). \tag{3.6}$$

3.1 Model Development

From the functional diagram, Figure 3.2,

Inputs,

$$u = \left(\dot{m}_{\mathrm{w}}, \dot{m}_{\mathrm{a}}, I_{\mathrm{f}}, I_{\mathrm{t}}, \dot{Q}^{\sigma}_{\mathrm{Fe}}, \dot{Q}^{\sigma}_{\mathrm{f}}\right) \tag{3.7}$$

and Outputs,

$$y = (T_{\mathrm{r}}, T_{\mathrm{s}}, T_{\mathrm{Fe}}) \tag{3.8}$$

The objective of the model is to see how the inputs affect the outputs.

Few things are assumed while developing the model. The metal temperatures are assumed to be homogeneous as indicated with impeller symbol in Figure 3.1 and this is because heat conduction in rotor copper, stator copper and iron is assumed to be very large than the heat transported across the metal boundaries. Furthermore, we assume that air and water temperature change are faster than the metal volumes. As an implication of this assumption, we can use relevant balance laws. The model development is described in three steps.

3.1.1 Step 1: Describing relevant balance laws

Since there is no change of masses in metals (stator copper, rotor copper, and stator iron) mass balance for metals are unimportant. Similarly, we assume air having constant density throughout the operation we can neglect mass balance for air as well. However, we want to see the evolution of metals temperatures during the operation of the synchronous generator. This inferred to use thermal energy balance. The thermal energy balance equation in terms of internal energy, enthalpy rate, work rate, and heat flow is given by Eq. 3.9.

$$\frac{dU}{dt} = \dot{H}_{\mathrm{i}} - \dot{H}_{\mathrm{e}} - \dot{W}_{\mathrm{f}} + \dot{W}_{\mathrm{v}} + \dot{Q} \tag{3.9}$$

As terms like power due to friction ($\dot{W}_{\mathrm{f}}$), and added mechanical power to change the system volume given by ($\dot{W}_{\mathrm{v}}$) are non-trivial, we can neglect these quantities from the energy thermal balance equation, Eq. 3.9. The energy thermal balance equation after assumption is now given in Eq. 3.10.

$$\frac{dU}{dt} = \dot{H}_{\mathrm{i}} - \dot{H}_{\mathrm{e}} + \dot{Q}. \tag{3.10}$$

3.1.2 Step 2: Relating balance equations to output quantities

Here, internal energy, U, can be related as,

$$U = H - pV \tag{3.11}$$

where enthalpy, H, is given in terms of specific enthalpy as,

$$H = m\hat{H}.$$

The differential specific enthalpy[13, p.358] is given by 3.12,

$$d\hat{H} = \hat{c}_p dT + \hat{V}(1 - \alpha_p T)dp. \tag{3.12}$$

For an *ideal gases*[1], $\alpha_p = \frac{1}{T}$, while for solids, $\alpha_p = 0$.

Thus for ideal gases, Eq. 3.12can be written as,

$$\hat{H} = \hat{H}^o + \int_{T^o}^{T} \hat{c}_p dT.$$

Similarly for solids specific enthalpy is given as

$$\hat{H} = \hat{H}^o + \int_{T^o}^{T} \hat{c}_p dT + V(p - p^o).$$

We are assuming constant pressure in metals and this reduces specific enthalpy expression to,

$$\hat{H} = \hat{H}^o + \int_{T^o}^{T} \hat{c}_p dT \tag{3.13}$$

So the expression for specific enthalpy for both air and metals is given by Eq. 3.13.

The specific heat capacity, $\hat{c}_p$, is dependent on temperature and often represented with a function in temperature. If $\hat{c}_p$ is considered temperature independence then Eq. 3.13 can be written as,

$$\hat{H} = \hat{H}^o + \hat{c}_p(T - T^o). \tag{3.14}$$

If $\hat{c}_p$ is function of temperature than Eq. 3.13 can be written as,

$$\hat{H} = \hat{H}^o + \int_{T^o}^{T} \hat{c}_p(T) dT. \tag{3.15}$$

$\hat{c}_p(T)$ is often represented by power series in temperature, T, in *Kelvin.* The 7-coefficient[2] power series has been purposed in [14, 15] for molar heat capacity at constant pressure at temperature for the standard state, for a specified range of temperature, given by Eq. 3.16.

$$\frac{\tilde{c}_p(T)}{\mathscr{R}} = a_1 T^{-2} + a_2 T^{-1} + a_3 + a_4 T + a_5 T^2 + a_6 T^3 + a_7 T^4 \tag{3.16}$$

where $\mathscr{R}$ is *Universal gas constant.*

The 7-coefficients power series can be further realized with linear and quadratic approximations. A comparison plot for linear, quadratic and 7-coefficients form for air, water,

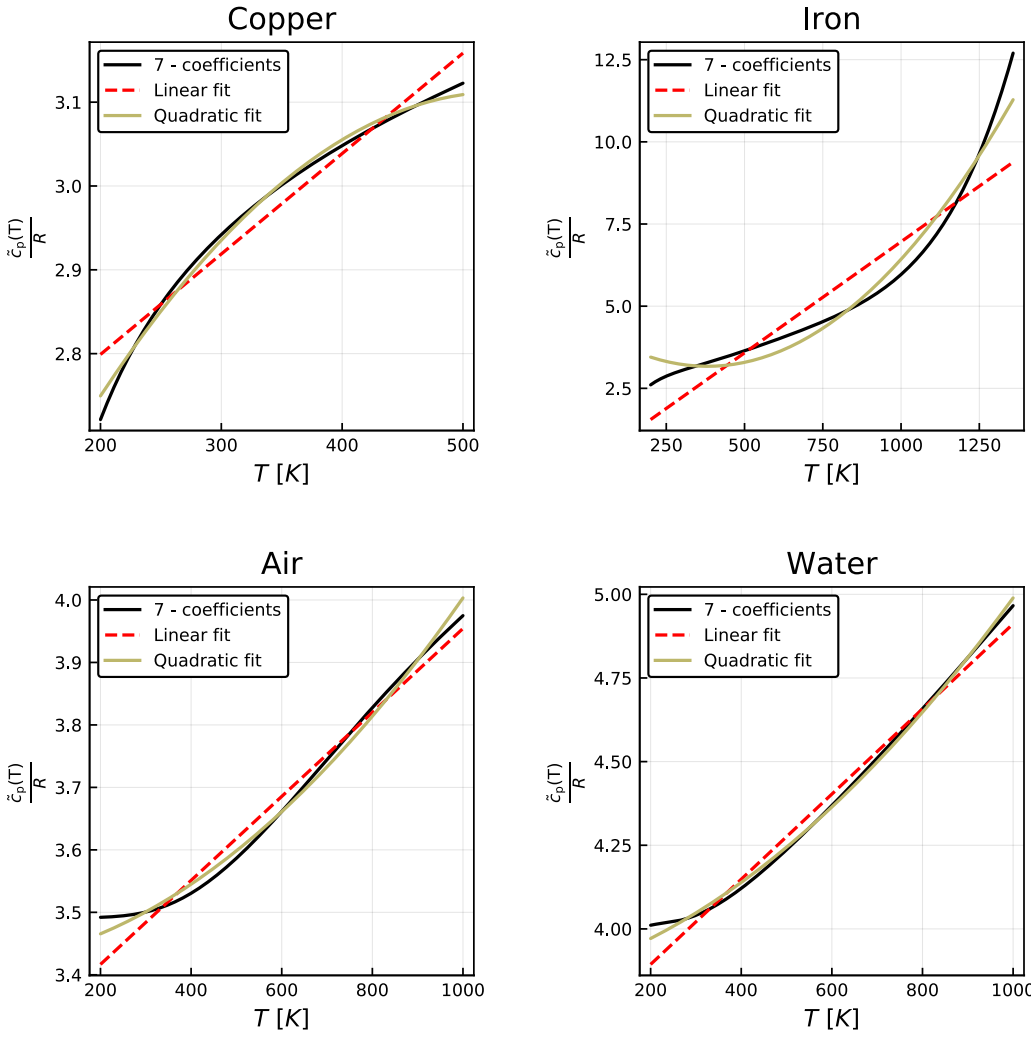


Figure 3.3: 7-coefficients, linear and quadratic approximations plots of $\frac{\tilde{c}_p(T)}{R}$ for copper, iron, air, and water.

copper, and iron is shown in Figure 3.3. The code listing to produce this plot is given in Appendix C Jupyter notebook number 10.

We will be considering linear approximation for temperature dependent molar heat capacity as given in Eq. 3.17.

$$\frac{\tilde{c}_p(T)}{\mathscr{R}} = a + bT \tag{3.17}$$

Molar specific heat capacity can be converted into specific heat capacity with relation given as, $\hat{c}_p(T) = \frac{\tilde{c}_p(T)}{M}$, where M is molecular mass.

The integral in Eq. 3.15 can be calculated which results expression for Eq. 3.15 to be,

$$\hat{H} = \hat{H}^o + \frac{\mathscr{R}}{M}\left(\left(aT + \frac{b}{2}T^2\right) - \left(aT^o + \frac{b}{2}T^{o2}\right)\right) \tag{3.18}$$

Furthermore, the enthalpy rate is given as,

$\dot{H} = \dot{m}\hat{H}$

Similarly, the temperature dependency of resistance can be realized with Eq. 3.19.

$$R(T) = R(T^o)\,(1 + \alpha\,(T - T^o)) \tag{3.19}$$

where α is the temperature coefficient of resistance for material as per choice.

Finally, the heat rate flow equations are taken as described in Chapter 3.

Heat exchanger model

A generic distributed tube and shell heat exchanger configurations are described in [13, p.389]. Out of three tube-and-shell heat exchanger configuration, cross-current, co-current, and counter-current, we will be discussing the counter-current heat exchanger model since the case study has operated with the counter-current heat exchanger. The tube is considered to be flown with the water and the shell with the air. The heat exchanger model is assumed to be a steady state model. A typical tube-and-shell counter-current heat exchanger is shown in Figure 3.4.

For the counter-current configuration of heat exchanger, the model equations are given as,

$$\frac{dT_\mathrm{w}}{dx} = \frac{\mathscr{U}\wp}{\hat{c}_{p,\mathrm{w}}\dot{m}_\mathrm{w}}\,(T_\mathrm{w} - T_\mathrm{a}) \tag{3.20}$$

$$\frac{dT_\mathrm{a}}{dx} = \frac{\mathscr{U}\wp}{\hat{c}_{p,\mathrm{a}}\dot{m}_\mathrm{a}}\,(T_\mathrm{w} - T_\mathrm{a}) \tag{3.21}$$

[1] We are considering air with properties of an ideal gas.
[2] These coefficients are also known as NASA Lewis coefficients

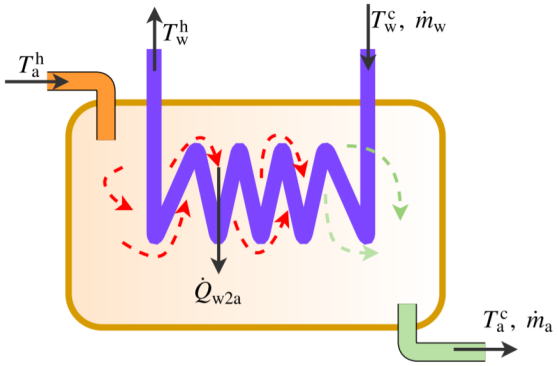


Figure 3.4: A shell-tube configuration of counter-current heat exchanger. The cold water, with mass flow rate $\dot{m}_w$, at temperature T_w^c is running inside the heat exchager to cool hot air from generator outlet, with mass flow rate $\dot{m}_a$ at temperature T_a^h. $\dot{Q}_{w2a}$ represent negative heat transfer from water to air.

with boundary conditions as, $T_w(x = L_x) = T_w^c$ and $T_a(x = 0) = T_a^h$. It should be noted that tube and shell temperatures are space derivative.

When the specific heat capacities for fluid (air, water) inside the heat exchanger are considered to be constant, the analytical solution for a two point boundary value problem is given in [13, p.400]. However, it is difficult to find an analytical solution when specific heat capacities are temperature dependent, so a numerical solution will be proposed wherever needed.

3.1.3 Step 3: Model equations

It is our interest to see the temperature dependent and temperature independent of resistance and specific heat capacity; and its effect on metal temperatures. Before developing the models we tend to develop notations to distinguish different models. First, R and $R(\mathrm{T})$ will represent models with constant and temperature dependent resistances respectively. Second, $\hat{c}_p$ and $\hat{c}_p(\mathrm{T})$ will represent constant and temperature dependent specific heat capacity. We can list four models as,

- **Model 1** $(\hat{c}_p, R)$: Resistances and specific heat capacities are both independent of temperature.
- **Model 2** $(\hat{c}_p, R(\mathrm{T}))$: Resistances are temperature dependent while specific heat capacities are temperature independent.
- **Model 3** $(\hat{c}_p(\mathrm{T}), R)$: Resistances are constant and specific heat capacities are temperature dependent.

- **Model 4** ($\hat{c}_p(\mathrm{T}), R(\mathrm{T})$): Resistances and specific heat capacities are both temperature dependent.

Model 3 and *Model 4* have further two models each *Model 3a* and *3b* and *Model 4a* and *4b*, respectively. Model 3a is considering specific heat capacities, for fluid, independent of the temperature inside the heat exchanger and Model 3b is considering specific heat capacities dependent of the temperature inside the heat exchanger, and same applies for Model 4a and 4b. This variation is taken into consideration to see the effect of specific heat capacities inside the heat exchanger. It is reasonable to have this variation as it is often difficult to find the analytical solution for heat exchanger models and numerical solution for the two point boundary value problem may have higher computational speed and cost.

3.1.3.1 Model equations for Model 1 ($\hat{c}_p, R$):

The balance equations for rotor copper, stator copper and stator iron can be written as,

$$\frac{dU_\mathrm{r}}{dt} = \dot{Q}_\mathrm{r}^{\sigma} - \dot{Q}_{\mathrm{r}2\delta} \tag{3.22}$$

$$\frac{dU_\mathrm{s}}{dt} = \dot{Q}_\mathrm{s}^{\sigma} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{3.23}$$

$$\frac{dU_\mathrm{Fe}}{dt} = \dot{Q}_\mathrm{Fe}^{\sigma} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}}. \tag{3.24}$$

Similarly for rotor/stator air gap, and forced-air inside the stator iron the balance equation are given as,

$$\frac{dU_\mathrm{a}^{\delta}}{dt} = \dot{H}_\mathrm{a}^{\mathrm{c}} - \dot{H}_\mathrm{a}^{\delta} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}_\mathrm{f}^{\sigma} \approx 0 \tag{3.25}$$

$$\frac{dU_\mathrm{a}^{\mathrm{h}}}{dt} = \dot{H}_\mathrm{a}^{\delta} - \dot{H}_\mathrm{a}^{\mathrm{h}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{3.26}$$

The internal energies for rotor copper, stator copper and stator iron are,

$$U_\mathrm{r} = H_\mathrm{r} - p_\mathrm{a} V_\mathrm{r} \tag{3.27}$$

$$U_\mathrm{s} = H_\mathrm{s} - p_\mathrm{a} V_\mathrm{s} \tag{3.28}$$

$$U_\mathrm{Fe} = H_\mathrm{Fe} - p_\mathrm{a} V_\mathrm{Fe}. \tag{3.29}$$

The total enthalpies are,

$$H_\mathrm{r} = m_\mathrm{r} \hat{H}_\mathrm{r} \tag{3.30}$$

$$H_\mathrm{s} = m_\mathrm{s} \hat{H}_\mathrm{s} \tag{3.31}$$

$$H_\mathrm{Fe} = m_\mathrm{Fe} \hat{H}_\mathrm{Fe}. \tag{3.32}$$

Similarly we can list specific enthalpies as,

$$\hat{H}_{\mathrm{r}} = \hat{H}^{o}_{\mathrm{Cu}} + \hat{c}_{p,\mathrm{Cu}}\left(T_{\mathrm{r}} - T^{o}_{\mathrm{Cu}}\right) \tag{3.33}$$
$$\hat{H}_{s} = \hat{H}^{o}_{\mathrm{Cu}} + \hat{c}_{p,\mathrm{Cu}}\left(T_{\mathrm{s}} - T^{o}_{\mathrm{Cu}}\right) \tag{3.34}$$
$$\hat{H}_{\mathrm{Fe}} = \hat{H}^{o}_{\mathrm{Fe}} + \hat{c}_{p,\mathrm{Fe}}\left(T_{\mathrm{Fe}} - T^{o}_{\mathrm{Fe}}\right) \tag{3.35}$$
$$\hat{H}^{\mathrm{c}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \hat{c}_{p,\mathrm{a}}\left(T^{\mathrm{c}}_{\mathrm{a}} - T^{o}_{\mathrm{a}}\right) \tag{3.36}$$
$$\hat{H}^{\delta}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \hat{c}_{p,\mathrm{a}}\left(T^{\delta}_{\mathrm{a}} - T^{o}_{\mathrm{a}}\right) \tag{3.37}$$
$$\hat{H}^{\mathrm{h}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \hat{c}_{p,\mathrm{a}}\left(T^{\mathrm{h}}_{\mathrm{a}} - T^{o}_{\mathrm{a}}\right). \tag{3.38}$$

Furthermore, we can list enthalpy flow rates for air as,

$$\dot{H}^{\mathrm{c}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}}\hat{H}^{\mathrm{c}}_{\mathrm{a}} \tag{3.39}$$
$$\dot{H}^{\delta}_{\mathrm{a}} = \dot{m}_{\mathrm{a}}\hat{H}^{\delta}_{\mathrm{a}} \tag{3.40}$$
$$\dot{H}^{\mathrm{h}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}}\hat{H}^{\mathrm{h}}_{\mathrm{a}}. \tag{3.41}$$

The heat rate equations are listed as,

$$\dot{Q}^{\sigma}_{\mathrm{r}} = 1.1R_{\mathrm{r}}I^{2}_{\mathrm{f}} \tag{3.42}$$
$$\dot{Q}^{\sigma}_{\mathrm{s}} = 3R_{\mathrm{s}}I^{2}_{\mathrm{t}} \tag{3.43}$$
$$\dot{Q}^{\sigma}_{\mathrm{f}} = 0.8\dot{W}_{\mathrm{f}} \tag{3.44}$$
$$\dot{Q}_{\mathrm{r2}\delta} = UA_{\mathrm{r2}\delta}\left(T_{\mathrm{r}} - T^{\delta}_{\mathrm{a}}\right) \tag{3.45}$$
$$\dot{Q}_{\mathrm{s2Fe}} = \mathscr{U}A_{\mathrm{s2Fe}}\left(T_{\mathrm{s}} - T_{\mathrm{Fe}}\right) \tag{3.46}$$
$$\dot{Q}_{\mathrm{Fe2a}} = \mathscr{U}A_{\mathrm{Fe2a}}\left(T_{\mathrm{Fe}} - T^{\mathrm{h}}_{\mathrm{a}}\right). \tag{3.47}$$

Finally the heat exchanger model equations solving the two point boundary value problem, the analytical solution is given as,

$$T^{\mathrm{h}}_{\mathrm{w}} = \frac{N^{\mathrm{w}}_{\mathrm{St}}\left(1 - e^{-N^{\Delta}_{\mathrm{St}}}\right)T^{\mathrm{h}}_{\mathrm{a}} + N^{\Delta}_{\mathrm{St}}e^{-N^{\Delta}_{\mathrm{St}}}T^{\mathrm{h}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}}e^{-N^{\Delta}_{\mathrm{St}}}} \tag{3.48}$$
$$T^{\mathrm{c}}_{\mathrm{a}} = \frac{N^{\Delta}_{\mathrm{St}}T^{\mathrm{h}}_{\mathrm{a}} + N^{\mathrm{a}}_{\mathrm{St}}\left(1 - e^{-N^{\Delta}_{\mathrm{St}}}\right)T^{\mathrm{c}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}}e^{-N^{\Delta}_{\mathrm{St}}}} \tag{3.49}$$
$$\dot{Q}_{\mathrm{w2a}} = \frac{e^{-N^{\Delta}_{\mathrm{St}}} - 1}{\frac{1}{\hat{c}_{p,\mathrm{a}}\dot{m}_{\mathrm{a}}}e^{-N^{\Delta}_{\mathrm{St}}} - \frac{1}{\hat{c}_{p,\mathrm{w}}\dot{m}_{\mathrm{w}}}}\left(T^{\mathrm{c}}_{\mathrm{w}} - T^{\mathrm{h}}_{\mathrm{a}}\right) \tag{3.50}$$

where,

$$N_{\mathrm{St}}^{\mathrm{w}} = \frac{\mathscr{U} A_{\mathrm{x}}}{\hat{c}_{p,\mathrm{w}} \dot{m}_{\mathrm{w}}} \tag{3.51}$$

$$N_{\mathrm{St}}^{\mathrm{a}} = \frac{\mathscr{U} A_{x}}{\hat{c}_{p,\mathrm{a}} \dot{m}_{\mathrm{a}}} \tag{3.52}$$

$$N_{\mathrm{St}}^{\Delta} = N_{\mathrm{St}}^{\mathrm{w}} - N_{\mathrm{St}}^{\mathrm{a}}. \tag{3.53}$$

3.1.3.2 Model equations for Model 2 $(\hat{c}_p, R(\mathrm{T}))$:

When the resistances of rotor copper and stator copper are considered to be temperature dependent, the model equations are identical as that of *Model 1*, however; Eq. 3.42 and Eq. 3.43 are replaced as,

$$\dot{Q}_{\mathrm{r}}^{\sigma} = 1.1 R_{\mathrm{r}} \left(1 + \alpha_{\mathrm{Cu}} \left(T_{\mathrm{r}} - T_{\mathrm{Cu}}^{o}\right)\right) I_{\mathrm{f}}^{2} \tag{3.54}$$

$$\dot{Q}_{\mathrm{s}}^{\sigma} = 3 R_{\mathrm{s}} \left(1 + \alpha_{\mathrm{Cu}} \left(T_{\mathrm{s}} - T_{\mathrm{Cu}}^{o}\right)\right) I_{\mathrm{t}}^{2}. \tag{3.55}$$

3.1.3.3 Model equations for Model 3 $(\hat{c}_p(\mathrm{T}), R)$:

- **Model 3a:** For this version of the model, we will be considering specific heat capacities as temperature dependent only outside of heat exchanger i.e. for metals and air inside the generator. The equations relating specific enthalpies in *Model 2* should now be changed with the temperature dependence of specific heat capacities. These are given as,

$$\hat{H}_{\mathrm{r}} = \hat{H}_{\mathrm{Cu}}^{o} + \int_{T_{\mathrm{Cu}}^{o}}^{T_{\mathrm{r}}} \hat{c}_{p,\mathrm{Cu}}(T) dT \tag{3.56}$$

$$\hat{H}_{\mathrm{s}} = \hat{H}_{\mathrm{Cu}}^{o} + \int_{T_{\mathrm{Cu}}^{o}}^{T_{\mathrm{s}}} \hat{c}_{p,\mathrm{Cu}}(T) dT \tag{3.57}$$

$$\hat{H}_{\mathrm{Fe}} = \hat{H}_{\mathrm{Fe}}^{o} + \int_{T_{Fe}^{o}}^{T_{Fe}} \hat{c}_{p,\mathrm{Fe}}(T) dT \tag{3.58}$$

$$\hat{H}_{\mathrm{a}}^{\mathrm{c}} = \hat{H}_{\mathrm{a}}^{o} + \int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\mathrm{c}}} \hat{c}_{p,\mathrm{a}}(T) dT \tag{3.59}$$

$$\hat{H}_{\mathrm{a}}^{\delta} = \hat{H}_{\mathrm{a}}^{o} + \int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\delta}} \hat{c}_{p,\mathrm{a}}(T) dT \tag{3.60}$$

$$\hat{H}_{\mathrm{a}}^{\mathrm{h}} = \hat{H}_{\mathrm{a}}^{o} + \int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\mathrm{h}}} \hat{c}_{p,\mathrm{a}}(T) dT \tag{3.61}$$

where the integral term can be expanded as,

$$\int_{T^o_{\mathrm{Cu}}}^{T_{\mathrm{r}}} \hat{c}_{p,\mathrm{Cu}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(\left(a_{\mathrm{Cu}}T_{\mathrm{r}} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{r}}^2\right) - \left(a_{\mathrm{Cu}}T^o_{\mathrm{Cu}} + \frac{b_{\mathrm{Cu}}}{2}T^{o2}_{\mathrm{Cu}}\right)\right)$$
$$\int_{T^o_{\mathrm{Cu}}}^{T_{\mathrm{s}}} \hat{c}_{p,\mathrm{Cu}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(\left(a_{\mathrm{Cu}}T_{\mathrm{s}} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{s}}^2\right) - \left(a_{\mathrm{Cu}}T^o_{\mathrm{Cu}} + \frac{b_{\mathrm{Cu}}}{2}T^{o2}_{\mathrm{Cu}}\right)\right)$$
$$\int_{T^o_{\mathrm{Fe}}}^{T_{\mathrm{Fe}}} \hat{c}_{p,\mathrm{Fe}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Fe}}}\left(\left(a_{\mathrm{Fe}}T_{\mathrm{Fe}} + \frac{b_{\mathrm{Fe}}}{2}T_{\mathrm{Fe}}^2\right) - \left(a_{\mathrm{Fe}}T^o_{\mathrm{Fe}} + \frac{b_{\mathrm{Fe}}}{2}T^{o2}_{\mathrm{Fe}}\right)\right)$$
$$\int_{T^o_{\mathrm{a}}}^{T^{\mathrm{c}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T^{\mathrm{c}}_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{\mathrm{c}2}_{\mathrm{a}}\right) - \left(a_{\mathrm{a}}T^o_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{o2}_{\mathrm{a}}\right)\right)$$
$$\int_{T^o_{\mathrm{a}}}^{T^{\delta}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T^{\delta}_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{\delta 2}_{\mathrm{a}}\right) - \left(a_{\mathrm{a}}T^o_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{o2}_{\mathrm{a}}\right)\right)$$
$$\int_{T^o_{\mathrm{a}}}^{T^{\mathrm{h}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T^{\mathrm{h}}_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{\mathrm{h}2}_{\mathrm{a}}\right) - \left(a_{\mathrm{a}}T^o_{\mathrm{a}} + \frac{b_{\mathrm{a}}}{2}T^{o2}_{\mathrm{a}}\right)\right).$$

- **Model 3b:** For this version of the model, we will be considering specific heat capacities depending on temperature for all metals and air inside the generator, and fluids (air, water) inside the heat exchanger. As specific heat capacities inside the heat exchanger are temperature dependent, the numerical solution is proposed using the two point boundary value problem. For counter-current heat exchanger, it is given as,

$$\frac{dT_{\mathrm{w}}}{dx} = \frac{\mathscr{U}\wp}{\frac{\mathscr{R}}{M_{\mathrm{w}}}(a_{\mathrm{w}} + b_{\mathrm{w}}T_{\mathrm{w}})\dot{m}_{\mathrm{w}}}(T_{\mathrm{w}} - T_{\mathrm{a}})$$
$$\frac{dT_{\mathrm{a}}}{dx} = \frac{\mathscr{U}\wp}{\frac{\mathscr{R}}{M_{\mathrm{a}}}(a_{\mathrm{a}} + b_{\mathrm{a}}T_{\mathrm{a}})\dot{m}_{\mathrm{a}}}(T_{\mathrm{w}} - T_{\mathrm{a}})$$

with boundary conditions as, $T_{\mathrm{w}}(x = L_{\mathrm{x}}) = T^{\mathrm{c}}_{\mathrm{w}}$ and $T_{\mathrm{a}}(x = 0) = T^{\mathrm{h}}_{\mathrm{a}}$.

3.1.3.4 Model equations for Model 4 $(\hat{c}_p(\mathrm{T}), R(\mathrm{T}))$:

Model 4 is similar to Model 3, however, we will take consideration of resistances depending on temperature as described in Model 2.

3.2 DAE Formulation

The standard form of DAE is given as,

$$\begin{aligned}\frac{dx}{dt} &= f(x,z,u;\theta)\\ 0 &= g(x,z,u;\theta)\\ y &= h(x,z,u;\theta).\end{aligned}$$

Out of five differential variables from balance equations, U_{a}^{δ} and $U_{\mathrm{a}}^{\mathrm{h}}$ are in steady state conditions which implies their derivatives are zero. This result with differential variable,to be,

$$x = (U_{\mathrm{r}}, U_{\mathrm{s}}, U_{\mathrm{Fe}})$$

Inputs and outputs are given by Eq. 3.7 and Eq. 3.8.

Similarly, we can list algebraic variables and model parameters for different models.

3.2.1 DAE formulation for Model 1

Algebraic variables,

$$\begin{aligned}z = \Big(&\dot{Q}_{\mathrm{r}}^{\sigma}, \dot{Q}_{\mathrm{s}}^{\sigma}, \dot{Q}_{\mathrm{f}}^{\sigma}, \dot{Q}_{\mathrm{r2}\delta}, \dot{Q}_{\mathrm{s2Fe}}, \dot{Q}_{\mathrm{Fe2a}},\\ &\hat{H}_{\mathrm{r}}, \hat{H}_{\mathrm{s}}, \hat{H}_{\mathrm{Fe}}, \hat{H}_{\mathrm{a}}^{\mathrm{c}}, \hat{H}_{\mathrm{a}}^{\delta}, \hat{H}_{\mathrm{a}}^{\mathrm{h}}, H_{\mathrm{r}}, H_{\mathrm{s}}, H_{\mathrm{Fe}}, \dot{H}_{\mathrm{a}}^{\mathrm{c}}, \dot{H}_{\mathrm{a}}^{\delta}, \dot{H}_{\mathrm{a}}^{\mathrm{h}},\\ &N_{\mathrm{St}}^{\mathrm{w}}, N_{\mathrm{St}}^{\mathrm{a}}, N_{\mathrm{St}}^{\Delta}, \dot{Q}_{\mathrm{w2a}}, T_{\mathrm{w}}^{\mathrm{h}}, T_{\mathrm{a}}^{\mathrm{c}}, T_{\mathrm{a}}^{\delta}, T_{\mathrm{a}}^{\mathrm{h}}, T_{\mathrm{r}}, T_{\mathrm{s}}, T_{\mathrm{Fe}}\Big)\end{aligned}$$

and the parameters are,

$$\begin{aligned}\theta = \Big(&p_{\mathrm{a}}, \hat{c}_{p,\mathrm{a}}, \hat{c}_{p,\mathrm{w}}, \hat{c}_{p,\mathrm{Cu}}, \hat{c}_{p,\mathrm{Fe}}, m_{\mathrm{r}}, m_{\mathrm{s}}, m_{\mathrm{Fe}},\\ &\hat{V}_{\mathrm{Cu}}, \hat{V}_{\mathrm{Fe}}, V_{\mathrm{r}}, V_{\mathrm{s}}, V_{\mathrm{Fe}}, \mathscr{U}A_{\mathrm{r2}\delta}, \mathscr{U}A_{\mathrm{s2Fe}}, \mathscr{U}A_{\mathrm{Fe2a}},\\ &hA_{\mathrm{ax}}, hA_{\mathrm{w}x}, \mathscr{U}A_{\mathrm{x}}, \hat{H}_{\mathrm{a}}^{o}, \hat{H}_{\mathrm{Cu}}^{o}, \hat{H}_{\mathrm{Fe}}^{o}, T_{\mathrm{a}}^{o}, \hat{T}_{\mathrm{Cu}}^{o}, T_{\mathrm{Fe}}^{o}, R_{\mathrm{r}}, R_{\mathrm{s}}\Big).\end{aligned}$$

3.2.2 DAE formulation for Model 2

The algebraic variables are same, however, with one extra parameter to be added is α_{Cu}.

3.2.3 DAE formulation for Model 3

For *Model 3a* algebraic variables are same as that of *Model 1* and *Model 2*. However, the parameters that need to be added are $a_{\mathrm{Cu}}, b_{\mathrm{Cu}}, a_{\mathrm{Fe}}, b_{\mathrm{Fe}}, a_{\mathrm{a}}, b_{\mathrm{a}}, \mathscr{R}, M_{\mathrm{Cu}}, M_{\mathrm{Fe}}, M_{\mathrm{a}}$.

Furthermore, for *Model 3b* as we do not have the analytical solution for the heat exchanger model, the algebraic variables get reduced and are different than other models. Algebraic variables and parameters are given as,

$$
\begin{aligned}
z = \Big(& \dot{Q}_{\mathrm{r}}^{\sigma}, \dot{Q}_{\mathrm{s}}^{\sigma}, \dot{Q}_{\mathrm{f}}^{\sigma}, \dot{Q}_{\mathrm{r}2\delta}, \dot{Q}_{\mathrm{s}2\mathrm{Fe}}, \dot{Q}_{\mathrm{Fe}2\mathrm{a}}, \\
& \hat{H}_{\mathrm{r}}, \hat{H}_{\mathrm{s}}, \hat{H}_{\mathrm{Fe}}, \hat{H}_{\mathrm{a}}^{\mathrm{c}}, \hat{H}_{\mathrm{a}}^{\delta}, \hat{H}_{\mathrm{a}}^{\mathrm{h}}, H_{\mathrm{r}}, H_{\mathrm{s}}, H_{\mathrm{Fe}}, \dot{H}_{\mathrm{a}}^{\mathrm{c}}, \dot{H}_{\mathrm{a}}^{\delta}, \dot{H}_{\mathrm{a}}^{\mathrm{h}}, \\
& T_{\mathrm{w}}^{\mathrm{h}}, T_{\mathrm{a}}^{\mathrm{c}}, T_{\mathrm{a}}^{\delta}, T_{\mathrm{a}}^{\mathrm{h}}, T_{\mathrm{r}}, T_{\mathrm{s}}, T_{\mathrm{Fe}} \Big)
\end{aligned}
$$

$$
\begin{aligned}
\theta = \Big(& p_{\mathrm{a}}, a_{\mathrm{Cu}}, b_{\mathrm{Cu}}, a_{\mathrm{Fe}}, b_{\mathrm{Fe}}, a_{\mathrm{a}}, b_{\mathrm{a}}, a_{\mathrm{w}}, b_{\mathrm{w}}, m_{\mathrm{r}}, m_{\mathrm{s}}, m_{\mathrm{Fe}}, \\
& \hat{V}_{\mathrm{Cu}}, \hat{V}_{\mathrm{Fe}}, V_{\mathrm{r}}, V_{\mathrm{s}}, V_{\mathrm{Fe}}, \mathscr{U} A_{\mathrm{r}2\delta}, \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}}, \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}}, \\
& \mathscr{U} \wp, \hat{H}_{\mathrm{a}}^{o}, \hat{H}_{\mathrm{Cu}}^{o}, \hat{H}_{\mathrm{Fe}}^{o}, T_{\mathrm{a}}^{o}, \hat{T}_{\mathrm{Cu}}^{o}, T_{\mathrm{Fe}}^{o}, R_{\mathrm{r}}, R_{\mathrm{s}}, \\
& \mathscr{R}, M_{Cu}, M_{\mathrm{Cu}}, M_{\mathrm{Fe}}, M_{\mathrm{a}} \Big).
\end{aligned}
$$

3.2.4 DAE formulation for Model 4

DAE formulation for *Model 4* is similar to *Model 3,* however with added parameter to be α_{Cu}.

Complete set of equations for DAE models are given in Appendix B.

3.3 ODE Formulation

DAE models are often regarded lucid in formulating models of physical systems since we avoid a lot of equations manipulation. Furthermore, DAE models are easily handled with a tool like OpenModelica using Modelica language. DAE models are often more complex than their corresponding ODE models because of large numbers of equations and model parameters, however; it contains a lot of information about the model.

It is often rigorous, since it is very complex for DAEs, to develop the ODE model to study the stability of the developed model. DAE models can easily be converted into ODE models with formula manipulation.

ODE models, from DAE models, are obtained by manipulating balance equations to output quantities, and in our case, they are T_r, T_s and T_Fe. By inserting expression of enthalpies to internal energy, and then to balance equation differential equations in outputs are obtained. Furthermore, we will be inserting expression for heat flow rate and enthalpy rate equations into the balance equations to get final ODE expression for our models.

3.3.1 ODE formulation for Model 1

After formula manipulation, the ODE model for *Model 1* is,

$$m_\mathrm{r}\hat{c}_{p,\mathrm{Cu}}\frac{dT_\mathrm{r}}{dt} = 1.1R_\mathrm{r}I_\mathrm{f}^2 - \mathscr{U}A_{\mathrm{r}2\delta}\left(T_\mathrm{r} - T_\mathrm{a}^{\delta}\right) \tag{3.62}$$

$$m_\mathrm{s}\hat{c}_{p,\mathrm{Cu}}\frac{dT_\mathrm{s}}{dt} = 3R_\mathrm{s}I_\mathrm{t}^2 - \mathscr{U}A_{\mathrm{s}2\mathrm{Fe}}\left(T_\mathrm{s} - T_\mathrm{Fe}\right) \tag{3.63}$$

$$m_\mathrm{Fe}\hat{c}_{p,\mathrm{Fe}}\frac{dT_\mathrm{Fe}}{dt} = \mathscr{U}A_{\mathrm{s}2\mathrm{Fe}}\left(T_\mathrm{s} - T_\mathrm{Fe}\right) - \mathscr{U}A_{\mathrm{Fe}2\mathrm{a}}\left(T_\mathrm{Fe} - T_\mathrm{a}^{h}\right) + \dot{Q}_\mathrm{Fe}^{\sigma}. \tag{3.64}$$

Similarly, for air inside the generator,

$$0 = \dot{m}_\mathrm{a}\hat{c}_{p,\mathrm{a}}\left(T_\mathrm{a}^{\mathrm{c}} - T_\mathrm{a}^{\delta}\right) + \mathscr{U}A_{\mathrm{r}2\delta}\left(T_\mathrm{r} - T_\mathrm{a}^{\delta}\right) + \dot{Q}_\mathrm{f}^{\sigma} \tag{3.65}$$

$$0 = \dot{m}_\mathrm{a}\hat{c}_{p,\mathrm{a}}\left(T_\mathrm{a}^{\delta} - T_\mathrm{a}^{\mathrm{h}}\right) + \mathscr{U}A_{\mathrm{Fe}2\mathrm{a}}\left(T_\mathrm{Fe} - T_\mathrm{a}^{\mathrm{h}}\right) \tag{3.66}$$

and the heat exchanger model equation as,

$$\left(N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}}e^{-N_\mathrm{St}^{\Delta}}\right)T_\mathrm{a}^{\mathrm{c}} = N_\mathrm{St}^{\Delta}T_\mathrm{a}^{\mathrm{h}} + N_\mathrm{St}^{\mathrm{a}}\left(1 - e^{-N_\mathrm{St}^{\Delta}}\right)T_\mathrm{w}^{\mathrm{c}}. \tag{3.67}$$

3.3.2 ODE formulation for Model 2

Applying the same reduction process from DAE to ODE as in *Model 1* formulation, we can find ODE for Model 2.

The ODE for Model 2 is similar as of Model 1, however, with a slight variation due to resistance dependence of temperature. As of same ODE formulation of Model 1, the rotor copper and stator copper temperatures differential equation is changed and given as,

$$m_\mathrm{r}\hat{c}_{p,\mathrm{Cu}}\frac{dT_\mathrm{r}}{dt} = 1.1R_\mathrm{r}\left(1 + \alpha_\mathrm{Cu}\left(T_\mathrm{r} - T_\mathrm{Cu}^{o}\right)\right)I_\mathrm{f}^2 - \mathscr{U}A_{\mathrm{r}2\delta}\left(T_\mathrm{r} - T_\mathrm{a}^{\delta}\right) \tag{3.68}$$

$$m_\mathrm{s}\hat{c}_{p,\mathrm{Cu}}\frac{dT_\mathrm{s}}{dt} = 3R_\mathrm{s}\left(1 + \alpha_\mathrm{Cu}\left(T_\mathrm{s} - T_\mathrm{Cu}^{o}\right)\right)I_\mathrm{t}^2 - \mathscr{U}A_{\mathrm{s}2\mathrm{Fe}}\left(T_\mathrm{s} - T_\mathrm{Fe}\right) \tag{3.69}$$

3.3.3 ODE formulation for Model 3

For *Model 3* it is easier to follow up with ODE formulation of *Model 4,* given in upcoming Subsection 3.3.4with the difference is only that resistances are not temperature dependent.

3.3.4 ODE formulation for Model 4

The ODE model for *Model 4a* is given as,

$$m_{\mathrm{r}}\frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(a_{\mathrm{Cu}}+b_{Cu}T_{\mathrm{r}}\right)\frac{dT_{\mathrm{r}}}{dt}=1.1R_{\mathrm{r}}\left(1+\alpha_{\mathrm{Cu}}\left(T_{\mathrm{r}}-T_{\mathrm{Cu}}^{o}\right)\right)I_{\mathrm{f}}^{2}-\mathscr{U}A_{\mathrm{r2}\delta}\left(T_{\mathrm{r}}-T_{\mathrm{a}}^{\delta}\right) \tag{3.70}$$

$$m_{\mathrm{s}}\frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T_{s}\right)\frac{dT_{\mathrm{s}}}{dt}=3R_{\mathrm{s}}\left(1+\alpha_{\mathrm{Cu}}\left(T_{\mathrm{s}}-T_{\mathrm{Cu}}^{o}\right)\right)I_{\mathrm{t}}^{2}-\mathscr{U}A_{\mathrm{s2Fe}}\left(T_{\mathrm{s}}-T_{\mathrm{Fe}}\right) \tag{3.71}$$

$$m_{\mathrm{Fe}}\frac{\mathscr{R}}{M_{\mathrm{Fe}}}\left(a_{\mathrm{Fe}}+b_{\mathrm{Fe}}T_{\mathrm{Fe}}\right)\frac{dT_{\mathrm{Fe}}}{dt}=\mathscr{U}A_{\mathrm{s2Fe}}\left(T_{\mathrm{s}}-T_{\mathrm{Fe}}\right)-\mathscr{U}A_{\mathrm{Fe2a}}\left(T_{\mathrm{Fe}}-T_{\mathrm{a}}^{\mathrm{h}}\right)+\dot{Q}_{\mathrm{Fe}}^{\sigma}. \tag{3.72}$$

Similarly, for air inside the generator,

$$0=\dot{m}_{\mathrm{a}}\frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T_{\mathrm{a}}^{\mathrm{c}}\right)T_{\mathrm{a}}^{\mathrm{c}}-\left(a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T_{\mathrm{a}}^{\delta}\right)T_{\mathrm{a}}^{\delta}\right)+\mathscr{U}A_{\mathrm{r2}\delta}\left(T_{\mathrm{r}}-T_{\mathrm{a}}^{\delta}\right)+\dot{Q}_{\mathrm{f}}^{\sigma} \tag{3.73}$$

$$0=\dot{m}_{\mathrm{a}}\frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T_{\mathrm{a}}^{c}\right)T_{\mathrm{a}}^{\delta}-\left(a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T_{\mathrm{a}}^{\mathrm{h}}\right)T_{\mathrm{a}}^{\mathrm{h}}\right)+\mathscr{U}A_{\mathrm{Fe2a}}\left(T_{\mathrm{Fe}}-T_{\mathrm{a}}^{\mathrm{h}}\right). \tag{3.74}$$

And, finally, for the heat exchanger model, we will not be considering temperature dependence of specific heat capacity for *air and water inside the heat exchanger*,

$$\left(N_{\mathrm{St}}^{\mathrm{w}}-N_{\mathrm{St}}^{\mathrm{a}}e^{-N_{\mathrm{St}}^{\Delta}}\right)T_{\mathrm{a}}^{\mathrm{c}}=N_{\mathrm{St}}^{\Delta}T_{\mathrm{a}}^{\mathrm{h}}+N_{\mathrm{St}}^{\mathrm{a}}\left(1-e^{-N_{\mathrm{St}}^{\Delta}}\right)T_{\mathrm{w}}^{\mathrm{c}}. \tag{3.75}$$

However, for *Model 4b,* as we will be considering temperature dependence of specific heat capacity for *air and water inside the heat exchanger,* the expression for the heat exchanger is given by boundary value equations given as,

$$\frac{dT_{\mathrm{w}}}{dx}=\frac{\mathscr{U}\wp}{\frac{\mathscr{R}}{M_{\mathrm{w}}}\left(a_{\mathrm{w}}+b_{\mathrm{w}}T_{\mathrm{w}}\right)\dot{m}_{\mathrm{w}}}\left(T_{\mathrm{w}}-T_{\mathrm{a}}\right) \tag{3.76}$$

$$\frac{dT_{\mathrm{a}}}{dx}=\frac{\mathscr{U}\wp}{\frac{\mathscr{R}}{M_{\mathrm{a}}}\left(a_{\mathrm{a}}+b_{\mathrm{a}}T_{\mathrm{a}}\right)\dot{m}_{\mathrm{a}}}\left(T_{\mathrm{w}}-T_{\mathrm{a}}\right) \tag{3.77}$$

with boundary conditions as, $T_{\mathrm{w}}\left(x=L_{\mathrm{x}}\right)=T_{\mathrm{w}}^{\mathrm{c}}$ and $T_{\mathrm{a}}\left(x=0\right)=T_{\mathrm{a}}^{\mathrm{h}}$.

4 Simulation of DAE and ODE Models

So far we have developed model equations for several models in Chapter 2.

The DAE model equations can easily be implemented in tools like OpenModelica and then simulated. Similarly, as an alias, the model developed in OpenModelica can easily interact using advance OMJulia functionality for further analysis. Furthermore, DAE models can also be solved using DAE solver packages under Julia language.

In this report, all of the DAE models except *Model 3b* and *Model 4b* are not formulated in OpenModelica as it is difficult to obtain algebraic equations solving analytical solutions for temperature dependent heat exchanger steady-state models. Model 3b and Model 4b are solved numerically using Julia language.

The ODE models for Model 1 and 2 can be reduced in the form given as,

$$M_1 \frac{dx}{dt} = M_2 x + M_3 z + v \tag{4.1}$$

$$N_1 z = N_2 x + w \tag{4.2}$$

.

For *Model 1* we have,

$$x = (T_{\mathrm{r}}, T_{\mathrm{s}}, T_{\mathrm{Fe}})$$

$$z = (T_{\mathrm{a}}^{\mathrm{c}}, T_{\mathrm{a}}^{\delta}, T_{\mathrm{a}}^{\mathrm{h}})$$

$$M_1 = diag(m_{\mathrm{r}}\hat{c}_{p,\mathrm{Cu}}, m_{\mathrm{s}}\hat{c}_{p,\mathrm{Cu}}, m_{\mathrm{Fe}}\hat{c}_{p,\mathrm{Fe}})$$

$$M_2 = \begin{pmatrix} -\mathscr{U} A_{\mathrm{r}2\delta} & 0 & 0 \\ 0 & -\mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} & \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \\ 0 & \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} & -\mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} - \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \end{pmatrix}$$

$$M_3 = \begin{pmatrix} 0 & \mathscr{U} A_{\mathrm{r}2\delta} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \end{pmatrix}$$

and

$$v = \begin{pmatrix} 1.1R_{\mathrm{r}}I_{\mathrm{f}}^2 \\ 3R_{\mathrm{s}}I_{\mathrm{t}}^2 \\ \dot{Q}_{\mathrm{Fe}}^{\sigma} \end{pmatrix}.$$

Similarly,

$$N_1 = \begin{pmatrix} -\dot{m}_{\mathrm{a}}\hat{c}_{p,\mathrm{a}} & \dot{m}_{\mathrm{a}}\hat{c}_{p,\mathrm{a}} + \mathscr{U}A_{\mathrm{r2}\delta} & 0 \\ 0 & -\dot{m}_{\mathrm{a}}\hat{c}_{p,\mathrm{a}} & \dot{m}_{\mathrm{a}}\hat{c}_{p,\mathrm{a}} + \mathscr{U}A_{\mathrm{Fe2a}} \\ N_{\mathrm{St}}^{\mathrm{w}} - N_{\mathrm{St}}^{\mathrm{a}}e^{-N_{\mathrm{St}}^{\Delta}} & 0 & -N_{\mathrm{St}}^{\Delta} \end{pmatrix}$$

$$N_2 = \begin{pmatrix} \mathscr{U}A_{\mathrm{r2}\delta} & 0 & 0 \\ 0 & 0 & \mathscr{U}A_{\mathrm{Fe2a}} \\ 0 & 0 & 0 \end{pmatrix}$$

$$w = \begin{pmatrix} \dot{Q}_f^{\sigma} \\ 0 \\ N_{\mathrm{St}}^{\mathrm{a}}\left(1 - e^{-N_{\mathrm{St}}^{\Delta}}\right)T_{\mathrm{w}}^{\mathrm{c}} \end{pmatrix}.$$

Similar formualtion can be done for ODE Model 2.

These ODE models can be easily formulated in Julia language and then simulated and analyzed.

The parameters and operating conditions for thermal model air-cooled synchronous generator are given in Table 4.1 and 4.2.

4.1 Models Implementation

In this section, we will discuss models implementation in programming languages. Models are implemented in Modelica [16] language using OpenModelica as a tool. The implemented DAE model in Modelica can be run using OMJulia from Julia. Furthermore, we will also be using *DifferentialEquations* [17] package from Julia to implement our DAE and ODE models.

Table 4.1: Parameters for the thermal model of air-cooled synchronous generator.

Quantity	Symbol	Value
Atmoshpheric pressure	p_{a}	$1.01{*}10^5\mathrm{N/m}^2$
Specific heat capacity, air	$\hat{c}_{p,\mathrm{a}}$	1.15 kJ/kg/K
Specific heat capacity, water	$\hat{c}_{p,\mathrm{w}}$	4.2 kJ/kg/K
Specific heat capacity, copper	$\hat{c}_{p,\mathrm{Cu}}$	385 J/kg/K
Specific heat capacity, iron	$\hat{c}_{p,\mathrm{Fe}}$	465 J/kg/K
Copper mass, rotor	m_{r}	9260 kg
Copper mass, stator	m_{s}	6827 kg
Iron mass, stator	m_{Fe}	71200 kg
Specific volume, copper	$\hat{V}_{\mathrm{Cu}}$	0.112 L/kg
Specific volume, iron	$\hat{V}_{\mathrm{Fe}}$	0.127 L/kg
Heat transfer, rotor to air gap	$\mathscr{U}A_{\mathrm{r}2\delta}$	2.7 kW/K
Heat transfer, stator copper to iron	$\mathscr{U}A_{\mathrm{s}2\mathrm{Fe}}$	20 kW/K
Heat transfer, stator iron to air	$\mathscr{U}A_{\mathrm{Fe}2\mathrm{a}}$	14.3 kW/K
Heat transfer, solid to air	$h_{\mathrm{a}}A_{\mathrm{x}}$	55.6 kW/K
Heat transfer, solid to water	$h_{\mathrm{w}}A_{\mathrm{x}}$	222 kW/K
Heat transfer, air to water	$\mathscr{U}A_{\mathrm{x}}$	$1/\left(\frac{1}{h_{\mathrm{a}}A_{\mathrm{x}}}+\frac{1}{h_{\mathrm{w}}A_{\mathrm{x}}}\right)$
Overall heat transfer coefficient*perimeter	$\mathscr{U}\wp$	0.88 kW/K/m
Reference specific enthalpies, $j \in \{a,Cu,Fe\}$	$\hat{H}_j^\circ$	0 kJ/kg
Reference temperatures, $j \in \{a,Cu,Fe\}$	T_j°	25°C
Rotor copper ohmic resistance	R_{r}	0.127Ω
Stator copper ohmic resistance	R_{s}	$1.95\ \mathrm{m}\Omega$
Universal gas constant	$\mathscr{R}$	8.314 J/K/mol
Molar mass, air	M_{a}	28.97 g/mol
Molar mass, water	M_{w}	18.01 g/mol
Molar mass, copper	M_{Cu}	63.54 g/mol
Molar mass, iron	M_{Fe}	55.84 g/mol
NASA Lewis coefficient-linear approx., air	$a_{\mathrm{a}}+b_{\mathrm{a}}T$	$3.28{+}0.000672T$
NASA Lewis coefficient-linear approx., water	$a_{\mathrm{w}}+b_{\mathrm{w}}T$	$3.63{+}0.001272T$
NASA Lewis coefficient-linear approx., copper	$a_{\mathrm{Cu}}+b_{\mathrm{Cu}}T$	$2.56{+}0.001200T$
NASA Lewis coefficient-linear approx., iron	$a_{\mathrm{Fe}}+b_{\mathrm{Fe}}T$	$0.19{+}0.00676T$

Table 4.2: Operating conditions for the thermal model of air-cooled synchronous generator.

Quantity	Symbol	Value
Initial value, rotor temperature	$T_\mathrm{r}(t=0)$	28°C
Initial value, stator copper temperature	$T_\mathrm{s}(t=0)$	28°C
Initial value, stator iron temperature	$T_\mathrm{Fe}(t=0)$	28°C
Initial specific enthalpy of rotor	$\hat{H}_\mathrm{r}(0)$	$\hat{H}^\circ_\mathrm{Cu} + \hat{c}_{p,\mathrm{Cu}}(T_\mathrm{r}(0) - T^\circ_\mathrm{Cu})$ in J/kg
Initial specific enthalpy of stator copper	$\hat{H}_s(0)$	$\hat{H}^\circ_\mathrm{Cu} + \hat{c}_{p,\mathrm{Cu}}(T_\mathrm{s}(0) - T^\circ_\mathrm{Cu})$ in J/kg
Initial specific enthalpy of stator iron	$\hat{H}_\mathrm{Fe}(0)$	$\hat{H}^\circ_\mathrm{Fe} + \hat{c}_{p,\mathrm{Fe}}(T_\mathrm{r}(0) - T^\circ_\mathrm{Fe})$ in J/kg
Initial enthalpy of rotor	$H_\mathrm{r}(0)$	$m_\mathrm{r}\hat{H}_\mathrm{r}$ in J
Initial enthalpy stator copper	$H_\mathrm{s}(0)$	$m_\mathrm{s}\hat{H}_\mathrm{s}$ in J
Initial enthalpy of stator iron	$H_\mathrm{Fe}(0)$	$m_\mathrm{Fe}\hat{H}_\mathrm{Fe}$ in J
Initial internal energy of rotor	$U_\mathrm{r}(0)$	$H_\mathrm{r}(0) - p_\mathrm{a}V_\mathrm{r}$ in J
Initial internal energy of stator copper	$U_\mathrm{s}(0)$	$H_\mathrm{s}(0) - p_\mathrm{a}V_\mathrm{s}$ in J
Initial internal energy of stator iron	$U_\mathrm{Fe}(0)$	$H_\mathrm{Fe}(0) - p_\mathrm{a}V_\mathrm{Fe}$ in J
Influent water temperature	T^c_w	3.8°C
Water mass flow rate	$\dot{m}_\mathrm{w}$	53.9 kg/s
Air mass flow rate	$\dot{m}_\mathrm{a}$	49.2 kg/s
Rated rotor field current	I_f	1055 A
Rated Stator terminal current	I_t	5360 A
Stator iron generated heat	$\dot{Q}^\sigma_\mathrm{Fe}$	212 kW
Friction work	$\dot{W}_\mathrm{f}$	528 kW
Power loss due to friction at air gap	$\dot{Q}^\sigma_\mathrm{f}$	$0.8\dot{W}_\mathrm{f}$

4.1.1 Models implementation in Modelica

A model class is created. This model class contains model parameters, variables, initial conditions, inputs, outputs, and equations. For Model 1, a model class *ModGenerator* is created shown as below[1],

```
model ModGenerator
  //Parameters
  parameter Real pa=1.01e5 "Atmospheric pressure, Pa";
  //
  parameter Real chpa=1.15 "Specific heat capacity air,kJ.kg-1.K-1";
  parameter Real chpw=4.2 "Specific heat capacity water,kJ.kg-1.K-1";
    .
    .
    .
  / Declaring variables
    // -- states
    Real Ur(start=Ur0,fixed=true) "Initializing internal energy of rotor copper,kJ";
    Real Us(start=Us0,fixed=true) "Initializing internal energy of stator copper,kJ";
    Real UFe(start=UFe0,fixed=true) "Initializing internal energy of stator iron,kJ";
    // -- auxiliary variables
    Real Qdrs "Heat flow source in rotor copper,kW";
    Real Qdss "Heat flow source in stator copper,kW";
    Real Qdfs "Heat flow source due to friction loss,kW";
    .
    .
    .
    //-- output variables
    output Real Tr "Temperature of rotor copper,C";
    .
    .
    // -- input variables
    input Real Twc "Cold water feed,C";
    input Real Ifd "Field current in rotor,A";
    .
    .
    .
    equation
    der(Ur)=Qdrs-Qdr2d;    der(Us)=Qdss-Qds2Fe;
    .
    .
    .
end ModGenerator
```

Now model *instantiation* is done defining *inputs* in another class named as *SimGenerator* as shown below,

```
model SimGenerator
    //Instantiate model of Air cooled Synchronous Generator
    ModGenerator G;
    // Declaring variables
    // -- inputs
    Real _Twc "Cold water feed,C";
    Real _mdw "Heat exchanger water mass flow rate,kg/s";
    Real _mda "Circulating air mass flow rate,kg/s";
    Real _Ifd "Field current in rotor,A";
```

[1]We are using Sublime Text, a text editor like Notepad++, for writing scripts for Modelica language.

```
10      Real _It "Terminal current in each stator phase,A";
11      Real _QdFes "Heat flow source in stator iron,kW";
12      Real _Wdf "Friction work rate in air gap,kW";
13      // Equations
14      equation
15      // -- input values
16      _Twc=3.8;
17      _mdw=53.9;
18      _mda=49.2;
19      _Ifd=1055;
20      _It=5360;
21      _QdFes=212;
22      _Wdf=528;
23      // -- injecting input functions to model inputs
24      G.Twc=_Twc;
25      G.mdw=_mdw;
26      G.mda=_mda;
27      G.Ifd=_Ifd;
28      G.It=_It;
29      G.QdFes=_QdFes;
30      G.Wdf=_Wdf;
31  end SimGenerator;
```

Here, *G is an instance of model class ModGenerator.* Now both model classes are wrapped inside a package with the same filename. For Model 1 we can choose a package name as *Model1* and file name should be *Model1.mo* . The program listing is given in Appendix C.

The file *Model1.mo* can now be open and analyze using OpenModelica. The model can also be loaded, as an alias, using OMJulia advance functionality.

To simulate this model first we need to add package *OMJulia.jl* in Julia[2] which will issue us for using OMJulia functionality. Next, we create OMJulia session object to define a *ModelicaSystem* environment inside Julia to access OMJulia advance functionality. For instances, simulating the model, acquiring *get/set*[3] functionalities, linearization of models, sensitivity analysis, etc are few of the things from OMJulia API.

For simulating *Model1.mo* we can do it as[4],

```
1  # Creating OMJulia session object
2  # gen is considered to be an object
3  gen = OMJulia.OMCSession()
4  # Creating a modelica system using session object
5  gen.ModelicaSystem("/pathtofile/Model1.mo","Model1.SimGenerator");
6  # Setting up simulation
```

[2]It should be noted that the installation of OpenModelica should be prior as of OMJulia. Julia creates required files for running OpenModelica in the backend as soon as it encounters OpenModelica has been installed.

[3]These methods allows us to get all quantities, inputs, outputs, etc and for setting models parameters, simulation options, etc.

[4]All of the Julia scripts are written in Jupyter notebook (https://jupyter.org/), which is accessed by

```
julia>using IJulia
julia>notebook();
```

```
gen.setSimulationOptions(["stopTime=$(500*60)","stepSize=60"])
# Simulting model
gen.simulate()
```

The simulation step size is considered to be 1 min with 5 hrs of total simulation time.

Similarly, we can simulate Model 2, Model 3a and Model 4a in both OpenModelica and in Julia using OMJulia advance functionality.

4.1.2 Models implementation in Julia

DifferentialEquations.jl is a feature-rich and highly performant package for solving differential equations [17] in Julia. It has got a rich functionality for simulation and analysis of DAE[5] and ODE[6] equations.

For defining a DAE problem, it is straight forward using a *DAEProblem* method. It requires a function that needs to be a DAE function containing differential and algebraic equations, initial conditions of differential variables, initial values for all variables and specifying the differential variable.

The function can be created for example, for Model 1,

```
#creating a model
function model1_DAE(err,dxdt,x,parameters,t)
    #
   # Naming derivatives of differential variables
    dUrdt = dxdt[1]
    dUsdt = dxdt[2]
    dUFedt = dxdt[3]
    # Naming differential variables
    Ur = x[1]
    Us = x[2]
    UFe = x[3]
    # Naming algebraic variables
    Qdrs= x[4] # " Heat flow source in rotor copper , kW "
    Qdss = x[5] # " Heat flow source in stator copper , kW "
  .
  .
  .
    # Equations
    #----------
    # - Algebraic equations
    err[1] = Hdac - Hdad + Qdr2d + Qdfs
    err[2] = Hdad - Hdah + QdFe2a
  .
  .
  .
  # - differential equations
    err[26] = -dUrdt + Qdrs - Qdr2d
    err[27] = -dUsdt + Qdss - Qds2Fe
```

[5]http://docs.juliadiffeq.org/latest/tutorials/dae_example.html
[6]http://docs.juliadiffeq.org/latest/tutorials/ode_example.html

```
  err[28] = -dUFedt + QdFes + Qds2Fe - QdFe2a
    #
    return err
end
```

Now all the variables initial values with initial derivatives of differential variables are given as,

```
#Initial values for algebraic variables
x0 = zeros(28,1)
x0[1] = Ur0
x0[2] = Us0
x0[3] = UFe0
x0[4] = 1.1* Rr * u_Ifd(0)^2
x0[5] = 3.0* Rs * u_It(0)^2
.
.
.
# Initial derivatives
dxdt0 = zeros(28,1)
dxdt0[1]= x0[4]-x0[7]
dxdt0[2]= x0[5]-x0[8]
dxdt0[3]= u_QdFes(0)+x0[8]-x0[9]
# Specifying differential variables
diff_vars = fill(false,28)
diff_vars[1],diff_vars[2],diff_vars[3]=true,true,true
# Time span
tspan = (0.0,500*60)
```

We then define a problem and solve it using a suitable solver[7] using `solver(...)` method. For our case, we are using `IDA()` from Sundails.jl[8] package.

```
# Issuing packages
using DifferentialEquations
using Sundials
# Defining a problem
problem=DAEProblem(model1_DAE,dxdt0,x0,tspan,parameters,differential_vars=diff_vars)
# Solving using suitable solver
solution=solve(problem,IDA())
```

The solution contains two arrays element of algebraic variables and time instant, u and t, respectively, containing solutions of DAE problem for each time span which can be stored in suitable variables and then use it later for further manipulation, for example using it for plotting temperature of rotor copper, stator copper or stator iron.

```
# 26,27,28 refers that Tr, Ts and TFe are considered to be 26th, 27th and 28th algebraic
    variable
Tr_model1_DAE=[solution.u[i][26]fori in 1:length(solution.u)]
Ts_model1_DAE=[solution.u[i][27] for i in 1:length(solution.u)]
```

[7] https://docs.juliadiffeq.org/latest/solvers/dae_solve.html
[8] https://github.com/JuliaDiffEq/Sundials.jl

```
TFe_model1_DAE=[solution.u[i][28] for i in 1:length(solution.u)]
t_model1_DAE = [solution.t[i] for i in 1:length(solution.u)];
```

In a similar way using `ODEProblem(...)` method from DifferentialEquations.jl we can solve ODE problems.

For example, we can set problem and find it solutions as,

```
# Initial values
x0 = [28.,28.,28.]
# Time span
tspan = (0, 500*60.)
# Defining ODE problem using ODE function
# for each models containing ODE equations, eg. model1_ODE
prob=ODEProblem(model1_ODE,x0,tspan,p)
sol = solve(prob, ABM54(), dt=60)
```

For solving DAE and ODE problems for Model 3b and 4b a two point boundary value problem should be solved. The analytical solution is difficult to find, however, can be posed with a numerical solution, and we will be using `BVProblem(...)` method[9].

First, we need to define the two boundary value equation creating a function and a *residual*[10] function that calculates residue of boundary equations. This two function then can be solved using the `BVProblem(...)` method. And using suitable boundary value problem solver we will be solving its solution.

For example the boundary value problem describe by Eq. 3.20 and Eq. 3.21 can be pose in Julia as,

```
heat_exchanger_length=(0.0,1.0)
# Defining heat exchanger boundary problem
function heat_exchanger!(dT,T,parameters,t)
    #T[1]=Ta, T[2]=Tw
    dT[1]=Up/(chpa*mda)*(T[2]-T[1])
    dT[2]=Up/(chpw*mdw)*(T[2]-T[1])
end
##initailGuess=[Tah,Twh]=for counter-current
#initialGuess=[42.27,8.36]
initial_guess=[Ta_o, u_Twc(0)]
function boundary_condition!(residual,T,par_is,t)
    residual[1]=T[1][1]-Tah  #Tah
    residual[2]=T[end][2]-Twc #Twc
end
#Posing boundary value problem with BVProblem() method
bvp=BVProblem(heat_exchanger!,boundary_condition!,initial_guess,heat_exchanger_length,
    parameters)
#Solving problem with Shooting Vern method
sol=solve(bvp, Shooting(Vern7()))
```

[9] https://docs.juliadiffeq.org/latest/tutorials/bvp_example.html

[10] This will calculate a residual in each iteration while solving numerically. While a suitable solver, in Julia for eg. IDA() or Vern(), will try to minimize this residual error to zero as possible within a tolerance limit and finds the solution.

The simulation of DAE models and ODE models are given in Appendix C in Jupyter notebook number 3 and 15 respectively.

4.2 Simulation with Nominal Inputs and Operating Conditions

The simulated outputs for different models can be compared together with supplied nominal inputs and operating conditions given in Table 4.2. Figure 4.1 and 4.2 shows simulated outputs with nominal inputs supplied to models.

4.3 Heat exchanger profiles

It is interesting to see the temperature variation inside the heat exchanger when resistance and specific heat capacities are constant and temperature dependent as shown in Figure 4.3.

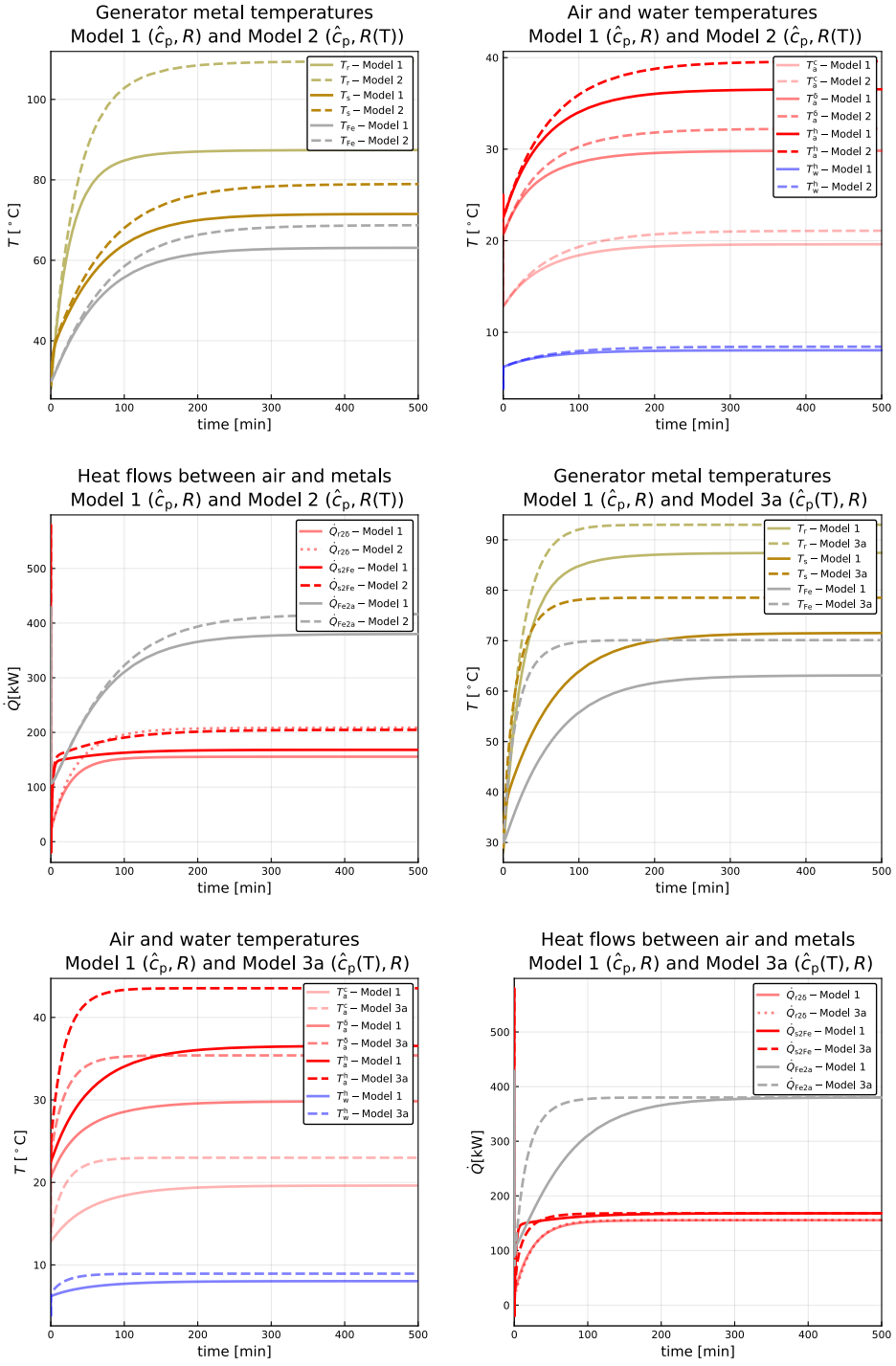


Figure 4.1: Simulated outputs with nominal inputs for Model 1 and Model 2.

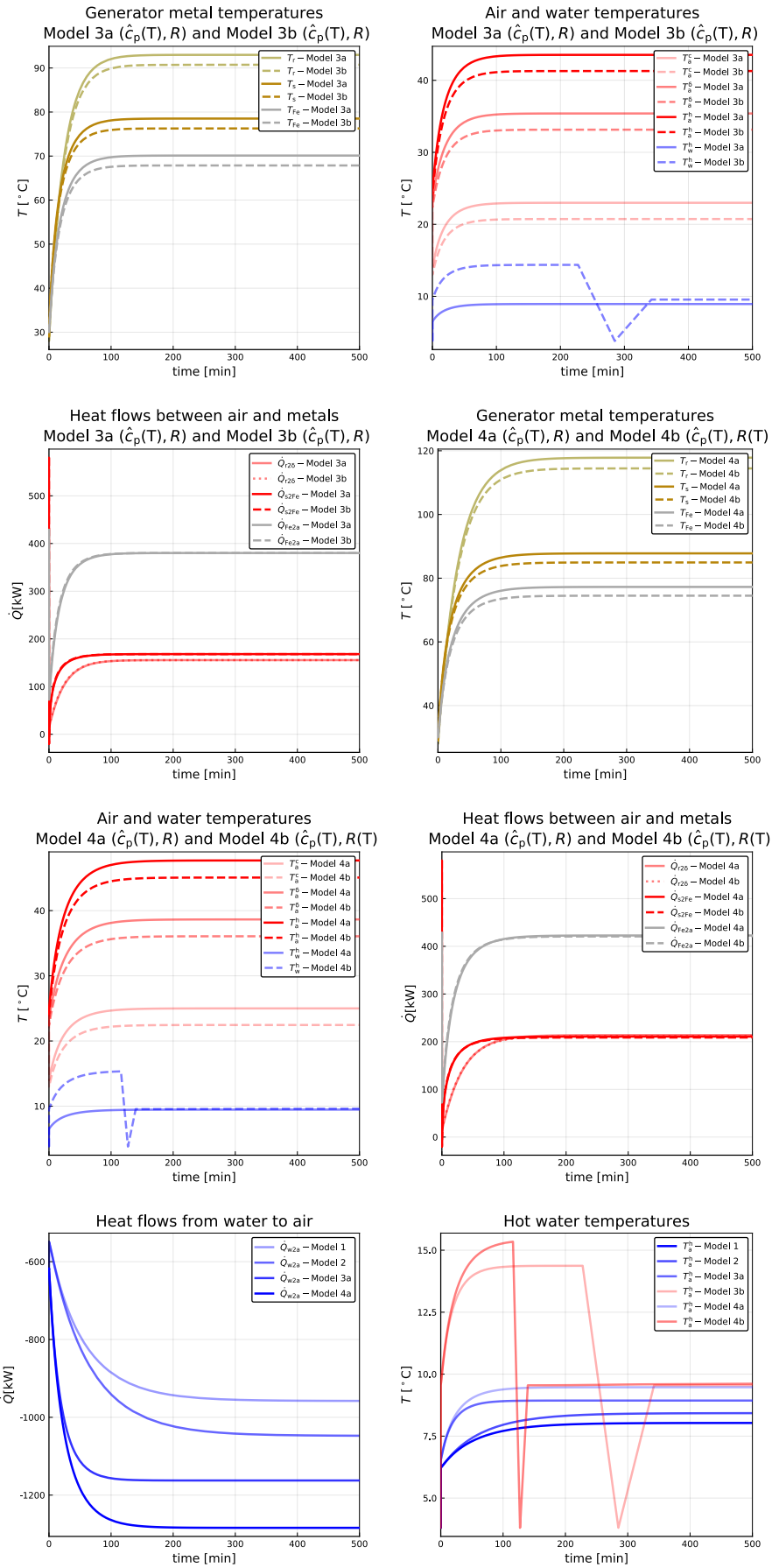


Figure 4.2: Simulated outputs with nominal inputs for Model 3a, 3b, 4a and 4b. Heat flows from water to air and the temperature of hot water are compared for all the models.

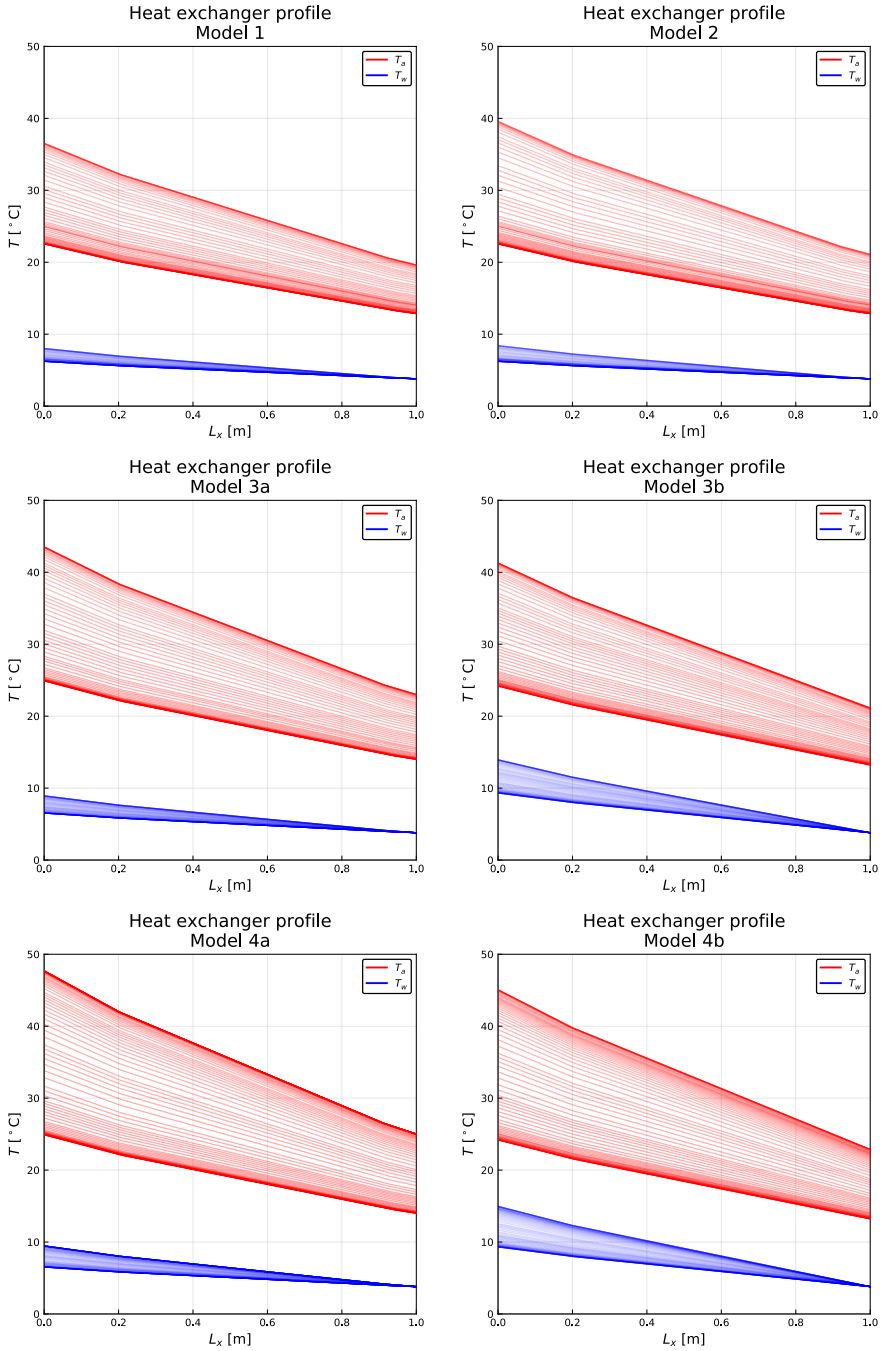


Figure 4.3: Heat exchanger profiles for all models. The figure contains 50 number of lines for both T_w and T_a where temperature is plotted for every 10 min for 500 min of simulation with nominal inputs and operating conditions.

5 Linearization, Stability, Controllability, and Observability

It is prime to see whether the system is stable at operating conditions or not. This chapter gives a brief review on linearization, stability, controllability, and observability for our generator models.

5.1 Linearization

Consider we have a nonlinear ODE model given as,

$$\frac{dx}{dt} = f(x,u;\theta)$$
$$y = g(x,u;\theta)$$

Let us assume that (x^*,u^*) is an operating point where the model is valid. Then Taylor series expansion of $f(x,u;\theta)$ at operating point (x^*,u^*) is given by[13],

$$f(x,u;\theta) \approx f(x^*,u^*;\theta) + \left.\frac{\partial f(x,u;\theta)}{\partial x}\right|_* (x-x^*) + \left.\frac{\partial f(x,u;\theta)}{\partial u}\right|_* (u-u^*)$$

.

The model can now be realized in the form of,

$$\frac{dx^\delta}{dt} = f(x^*,u^*;\theta) + Ax^\delta + Bu^\delta \tag{5.1}$$

where $x^\delta = x - x^*$, $u^\delta = u - u^*$, $A = \left.\frac{\partial f(x,u;\theta)}{\partial x}\right|_*$,$B = \left.\frac{\partial f(x,u;\theta)}{\partial u}\right|_*$,and $f(x^*,u^*;\theta) = 0$ since the model is valid at (x^*,u^*).

Thus, the linearized model is written as,

$$\frac{dx^\delta}{dt} = Ax^\delta + Bu^\delta$$

Similarly,

$$y^{\delta} = Cx^{\delta} + Du^{\delta}$$

where A and B are Jacobian matrices in states and inputs, also called as *system or state* and *control or input* matrices, respectively. Similarly, C and D are called as *output* and *feedforward* matrices respectively.

5.2 Stability

There are several stability analysis approaches, however, we will be analyzing our system stability on the basis of eigenvalues of matrix A.

Let $\lambda = \{\lambda_1, \lambda_2, ...\lambda_n\}$ be eigenvalues of matrix A, where n is then number of states in the system, then it is defined that system is stable only if $\forall j : \Re(\lambda_i) < 0$.

5.3 Controllability

It is often defined as the ability of a system to change its states while system input(s) are changed. The controllability of the system helps to determine a system engineer whether a system can be controlled (in a more specific/desired way) or not.

A system is controllable if the *rank* of *controllability matrix* is equal to the rank of the system matrix A.

For a system with system matrix A and control matrix B, the controllability matrix is given as,

$$\mathscr{C} = \begin{pmatrix} B \\ AB \\ A^2B \\ . \\ . \\ . \\ A^{n-1}B \end{pmatrix}.$$

For a controllability of a system, $\text{rank}(A) = \text{rank}(\mathscr{C})$.

5.4 Observability

A system is said to be observable only if and only if information about the states of a system can be determined from the measurements. For a system to be observable,

$$\text{rank}(A) = \text{rank}(\mathscr{O})$$

where, $\mathscr{O}$ is an observability matrix given as,

$$\mathscr{O} = \begin{pmatrix} C \\ CA \\ CA \\ . \\ . \\ . \\ CA^{n-1} \end{pmatrix}.$$

.

5.5 Linearization, Stability, Controllability, and Observability of Model 1 and Model 2

Model 1 and Model 2 can be linearized around operating points with steady-state solutions. There are several methods available in Julia's DifferentailEquations.jl package for finding steady-state solutions. Similarly, we will be using ForwardDiff.jl package for calculating jacobians for computing system and input/control matrix. Once A and B are calculated we can linearize the system.

Furthermore, for stability analysis, we will be finding *eigenvalues* of A matrix and see whether the system is stable around operating conditions. The time constant for the system can be found from eigenvalues.

The ODE for Model 1 can be formulated in Julia as,

```
#Author: Bernt Lie
#Edited on purpose and use by: Madhusudhan Pandey
# Necessary packages
using DifferentialEquations
using LinearAlgebra
using Plots
using Plots.PlotMeasures
using LaTeXStrings
pyplot();
# ODE model of Model1
```

```
function model1(x,u,t)
    # unpacking states
    Tr = x[1]
    Ts = x[2]
    TFe = x[3]
    # upacking inputs
    Twc = u[1]
    Ifd = u[2]
    It = u[3]
    QdFes = u[4]
    Wdf = u[5]
    mdw = u[6]
    mda = u[7]
    # parameters
    chpa = 1.15
    chpw = 4.2
    chpCu = 0.385
    chpFe = 0.465
    # Masses
    mr = 9260.
    ms = 6827.
    mFe = 71200.
    # Heat transfer coefficients
    UAr2d = 2.7
    UAs2Fe = 20.
    UAFe2a = 14.3
    hAax = 55.6
    hAwx = 222.
    UAx = 1/(1/hAax+1/hAwx)
    # Resistances
    Rr = 0.127e-3
    Rs = 1.95e-6
    #
    Qdfs = 0.8*Wdf
    # Stanton numbers
    NSta = UAx/chpa/mda
    NStw = UAx/chpw/mdw
    NStd = NStw - NSta
    # Matrices
    M1 = diagm(0=>[mr*chpCu , ms*chpCu , mFe*chpFe])
    M2 = [-UAr2d 0. 0.; 0. -UAs2Fe UAs2Fe; 0. UAs2Fe -UAs2Fe-UAFe2a]
    M3 = [0. UAr2d 0.; 0. 0. 0.; 0. 0. UAFe2a]
    #
    N1 = [-mda*chpa mda*chpa+UAr2d 0.; 0. -mda*chpa mda*chpa+UAFe2a; NStw-NSta*
        exp(-NStd) 0. -NStd]
    N2 = [UAr2d 0. 0.; 0. 0. UAFe2a; 0. 0. 0.]
    #
    v = [1.1*Rr*Ifd^2, 3*Rs*It^2, QdFes]
    w = [Qdfs, 0., NSta*(1-exp(-NStd))*Twc]
    #
    z = N1\(N2*x + w)
    dxdt = M1\(M2*x+M3*z + v)
    return dxdt
end
```

Further analysis of this ODE can be done as,

```
# Nominal Inputs
u = [3.8,1055,5360,212,528,53.9,49.2]
# Initial states
x0 = [28.,28., 28.]
# Time span
tspan = (0., 300*60.)
```

```
# findind steady state solution
prob_steady_state = ODEProblem(model1,x0,(0.,Inf),u)
sol_steady_state = solve(prob_steady_state,DynamicSS(Tsit5()))
# steady state state values
xs = sol_steady_state.u[end,1]
# nominal inputs
us = u
using ForwardDiff
# functions in states and inputs
fx(x) = gen(x,us,0)
fu(u) = gen(xs,u,0)
# Jacobians at operating points
A = ForwardDiff.jacobian(fx,xs)
B = ForwardDiff.jacobian(fu,us)
# Calculating time constants
# Eigen value of system matrix, A
using LinearAlgebra
# Eigenvector
R = eigvecs(A)
# Eigenvalues
lambda = eigvals(A)
# Time Constants
tau = -@. inv(lambda)/60
```

Thus, Model 1 can be linearized with,

$$A = \begin{pmatrix} -7.02*10^{-4} & 0 & 1.11*10^{-4} \\ 0 & -7.60*10^{-3} & 7.60*10^{-3} \\ 2.48*10^{-5} & 6.04*10^{-4} & -8.98*10^{-4} \end{pmatrix}$$

$$B = \begin{pmatrix} 5.91*10^{-4} & 1.04*10^{-4} & 0 & 0 & 1.617*10^{-5} & -2.21*10^{-5} & -1.39*^{-5} \\ 0 & 0 & 4.28*10^{-5} & 0 & 0 & 0 & 0 \\ 2.69*10^{-4} & 0 & 0 & 3.02*10^{-5} & 7.36*10^{-6} & -10^{-5} & -6.99*10^{-5} \end{pmatrix}$$

, and since we will only have measurements of two of our states, stator copper and stator iron temperatures, T_s and T_{Fe}, respectively we have,

$$C = \begin{pmatrix} 0 & 1 & 1 \end{pmatrix}.$$

The eigenvalues are given as,

$$\lambda = \left\{ -7.08*10^{-4}, -2.65*10^{-4}, -8.23*10^{-3} \right\}$$

and since we have all the eigenvalues in the *left half-plane* we can say that our generator Model 1 is stable at operating conditions.

Similarly, the time constants, given by $\tau_{j\varepsilon\{1,2,3\}} = \frac{1}{|\lambda_{j\varepsilon\{1,2,3\}}|}$, are,

$$\tau = \left\{ \begin{array}{lll} 23.52, & 62.69, & 2.02 \end{array} \right\} \text{min}.$$

For checking controllability and observability, *ControlSystems.jl* [1] package under Julia has got enriched functionality. When A, B, and C matrix are known then, controllability and observability can be found as,

```
# Controllability and observability
using ControlSystems

# Rank of matrix, A
# System order
rank_of_A = rank(A)
# Controllability matrix
ctrb_C = ctrb(A,B)
# Rank of controlability matrix
rank_of_ctrb_C=rank(ctrb_C)


# Observability
C = [0. 1. 1.]
# Observability matrix
obsv_O = obsv(A,C)
rank_of_obsv_O=rank(obsv_O)
```

. This produces,

$$\text{rank}(A) = \text{rank}(\mathscr{C}) = \text{rank}(\mathscr{O}) = 3$$

; therefore, our Model 1 is both controllable and observable.

In a similar way we can linerize Model 2 and check for stability, observaility, and controllability. The time constant for Model 2 is found to be $\tau = \{\ 31.12,\ \ 68.92,\ \ 2.05\ \}\,\text{min}$.

The code listing for linearizing, finding stability, controllability, and observability for Model 1 and 2 is given in Appendix C with Jupyter notebook number 11.

[1] *http://juliacontrol.github.io/ControlSystems.jl/latest/*

6 Parameters Sensitivity Analysis and Model Fitting

This chapter focuses mainly on determining the sensitivity in states due to parameters. Furthermore, we will have an overview of the experimental data. We will then simulate models and plot simulated outputs with real measurements. At the end of the chapter, we will optimize a few parameters for better fitting of outputs from simulation to measurements.

6.1 Parameters sensitivity analysis

It is often prime to analyze sensitivity in states due to parameters changes. This effect can be analyzed by *parameter sensitivity analysis.* Several methods are available [18, 19]; however, we will be discussing *local sensitivity analysis* formulating *a sensitivity ODE* in this report.

For a system, $\frac{dx}{dt} = f(x,u;\theta)$, having k number of system states and n number of parameters, local sensitivity for $j\varepsilon\{1,2,...,n\}$ can be computed using,

$$\frac{d}{dt}\frac{\partial x}{\partial \theta_j} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial \theta_j} + \frac{\partial f}{\partial \theta_j} = J * S_j + F_j$$

where,

$$J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_k} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_k} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_k}{\partial x_1} & \frac{\partial f_k}{\partial x_2} & \cdots & \frac{\partial f_k}{\partial x_k} \end{pmatrix}$$

is Jacobian in states,

$$F_j = \begin{pmatrix} \frac{\partial f_1}{\partial \theta_j} \\ \frac{\partial f_2}{\partial \theta_j} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial f_k}{\partial \theta_j} \end{pmatrix}$$

is parameter derivatives, and

$$S_j = \begin{pmatrix} \frac{\partial x_1}{\partial \theta_j} \\ \frac{\partial x_2}{\partial \theta_j} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial x_k}{\partial \theta_j} \end{pmatrix}$$

is a vector of sensitivities. Both $F_j$and S_j are calculated simultaneously for each iteration while computing numerically. This problem can easily be formulated in Julia using package *DiffEqSensitivty.jl* [1].

For defining local sensitivity problem a method `ODELocalSensitivityProblem(..)`is used. After posing a local sensitivity problem we can solve this using any `solve(....)` method using relevant solver.

For our Model 1 ($\hat{c}_p$,R), the parameter sensitivity analysis is done in *ParametersSensitivityAnalysis.ipynb* Jupyter notebook given in Appendix C. Figure 6.1, 6.2, 6.3, and 6.4 the result from sensitivity analysis. It shows that states are sensitive to resistance than any other parameters.

6.2 Overview of Experiemental Data

The *heat-run test* of the synchronous machine was performed for 600 min [10]. For each minute, for a supplied filed current, starting from *a cold-start* the data consists of different measurements. The *cold-run* was up to 53 min where the terminal voltage is build-up due to residual flux in rotor windings. After cold-run field current is increased which increases the temperature of stator copper and stator iron.

The measurements quantities can be summarized in Table 6.1.

[1] http://docs.juliadiffeq.org/latest/analysis/sensitivity.html

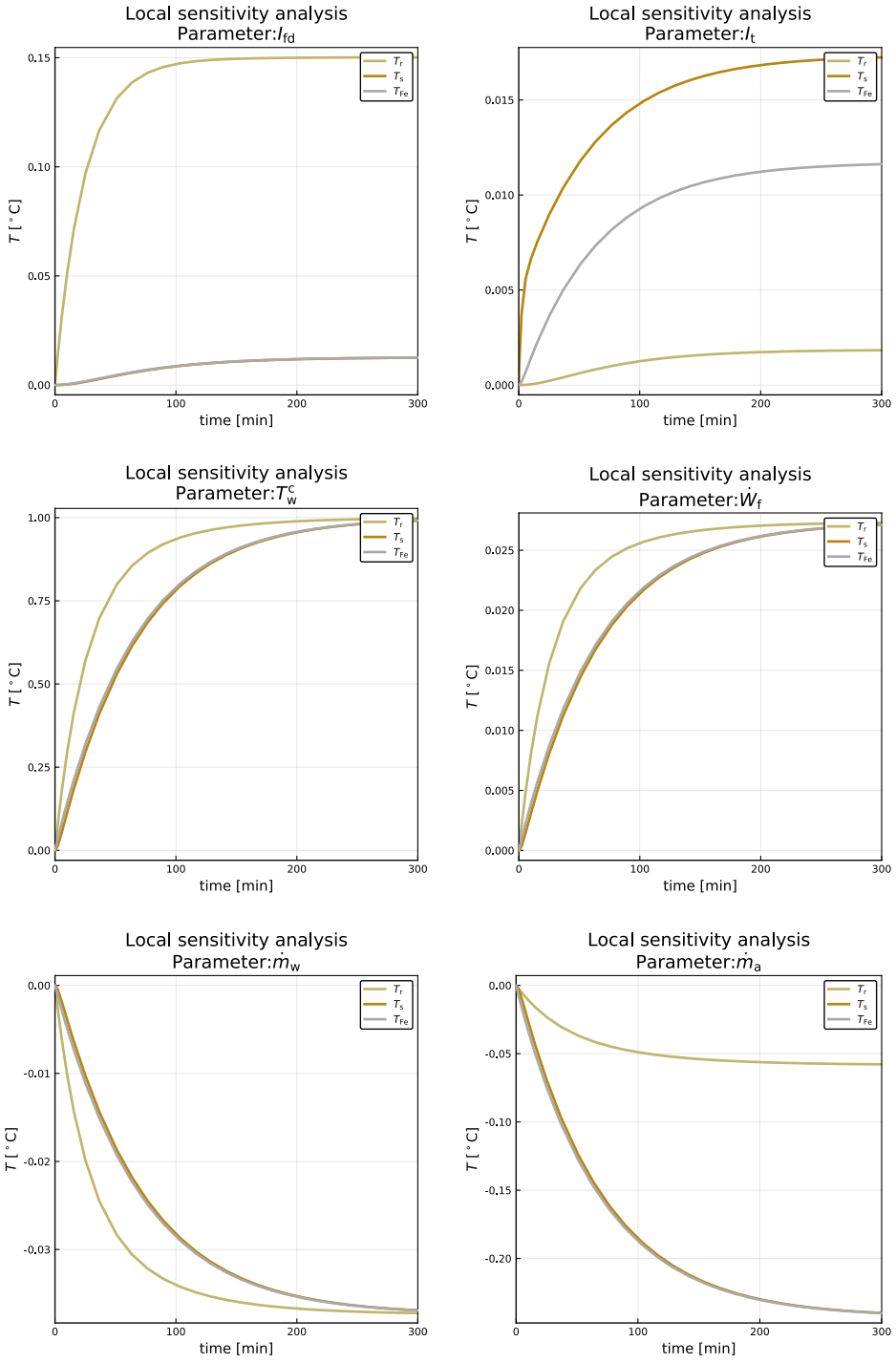


Figure 6.1: Sensitivity in states due to nominal inputs.

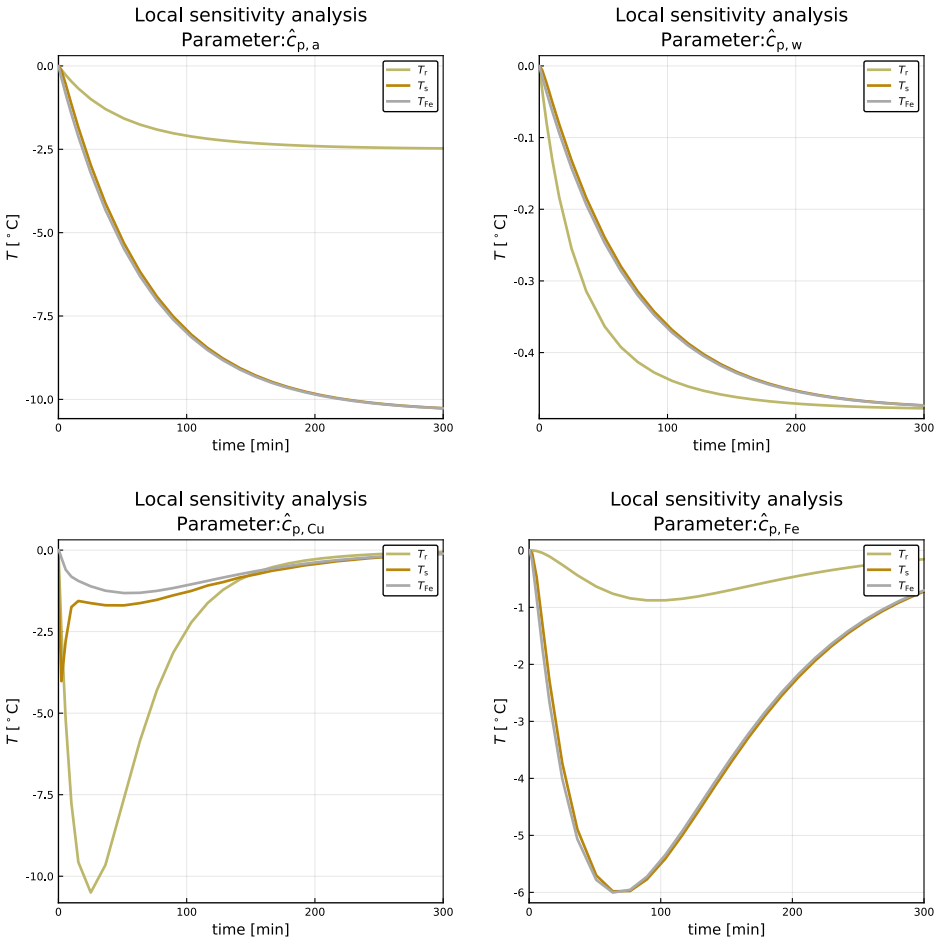


Figure 6.2: Sensitivity in states due to specific heat capacities ($\hat{c}_p$).

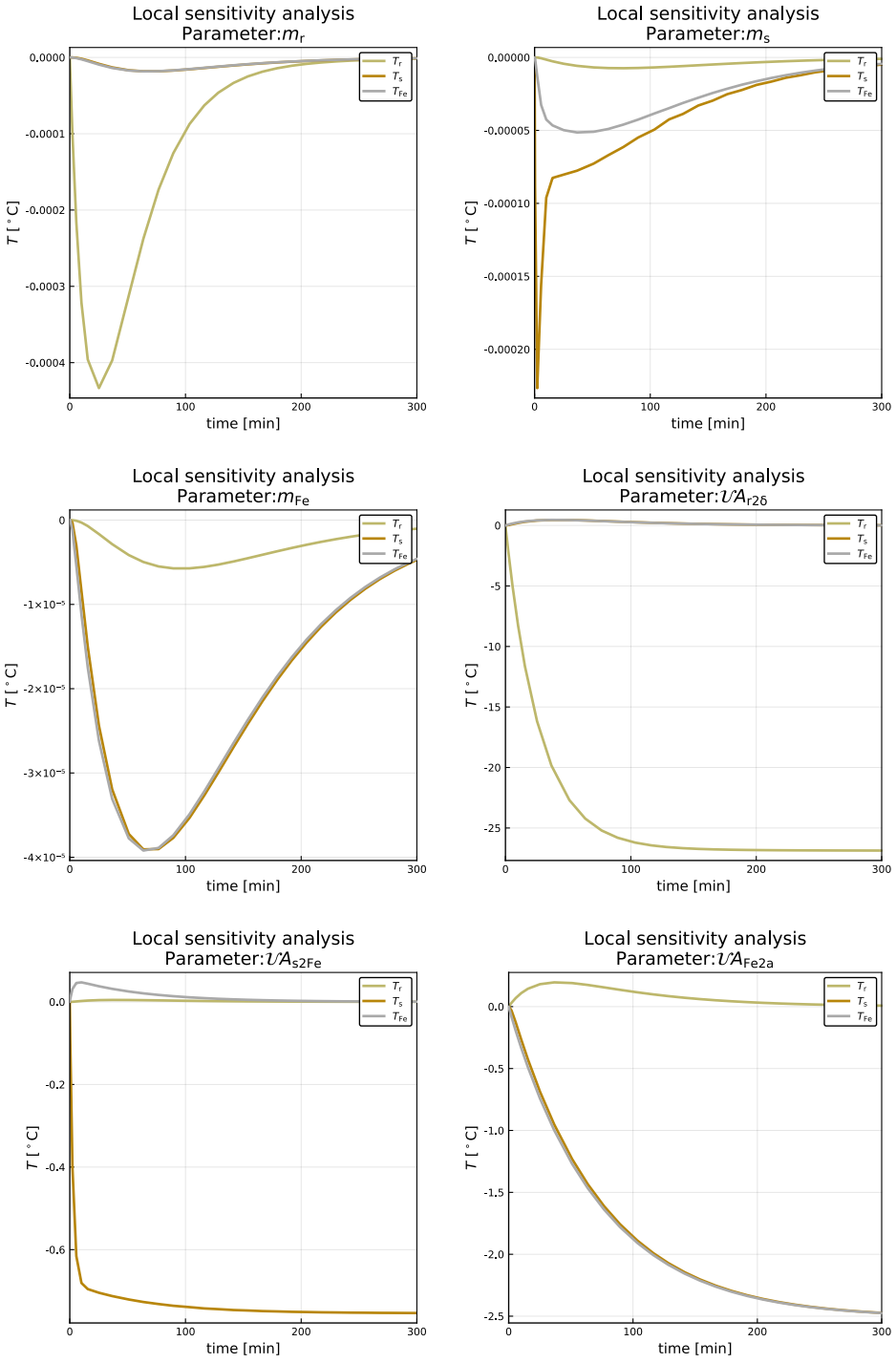


Figure 6.3: Sensitivity in states due to metal masses and heat transfer.

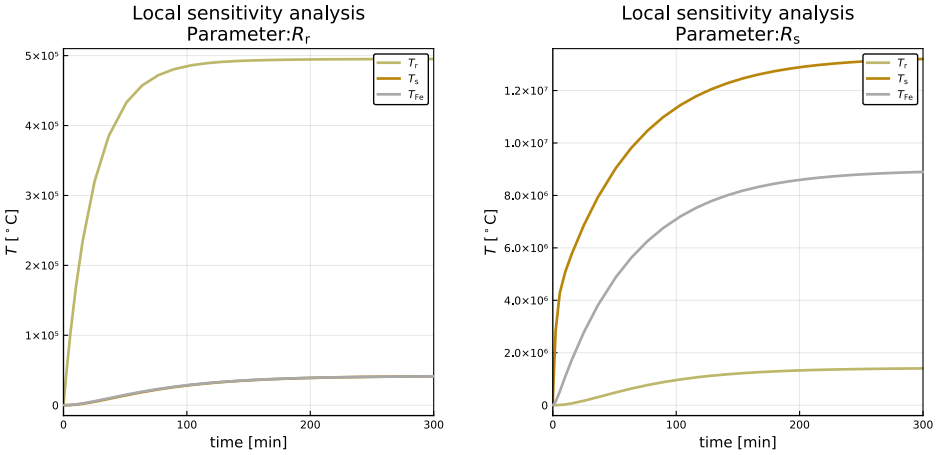


Figure 6.4: Sensitivity in states due to rotor and stator copper resistances.

Table 6.1: Measured quantities from heat-run test. The expression for terminal current is shown at end row, 2nd column of table to indicate that I_t is not measured using sensor, however, calculated from mathematical expression shown inline. All other quantities are measured using sensors.

Quantity	Symbol	Units	Sensor	No. of sensors
Generator terminal voltage	V_t	kV	-	-
Active power of generator	P_g	MW	-	-
Reactive power of generator	Q_g	MVar.	-	-
Rotor field current	I_f	A	-	-
Temperature of stator copper	T_s	°C	PT 100	15
Temperature of stator iron	T_{Fe}	°C	PT 100	4
Hot air temperature	T_a^h	°C	Pt 100/CTD	2/2
Cold air temperature	T_a^c	°C	Pt 100/CTD	2/2
Cold water temperature	T_w^c	°C	Analog	-
Hot water temperature	T_w^h	°C	Analog	-
Terminal current	$I_t = \frac{P_g^2+Q_g^2}{\sqrt{3}*V_t}$	A	-	-

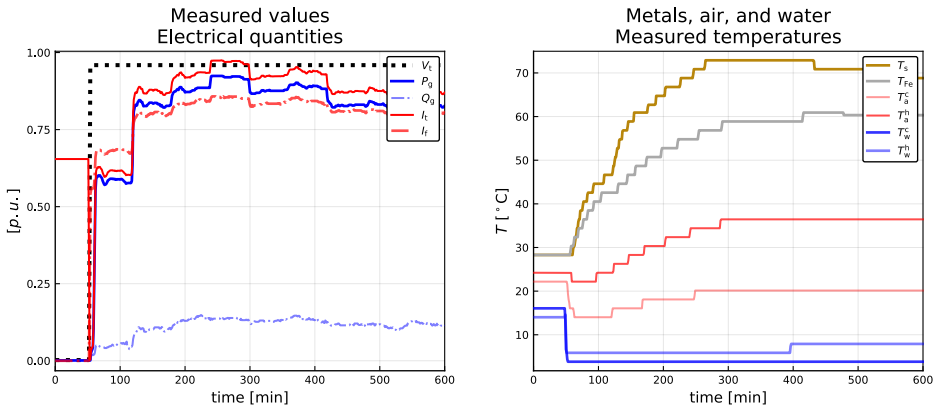


Figure 6.5: Experimental data for generator model from 600 min heat-run test.

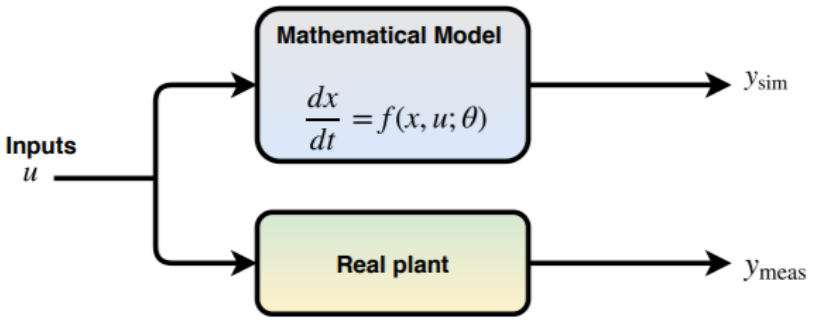


Figure 6.6: Mathematical model and plant are run together with same the inputs. y_{sim} and y_{meas} represents simulated and measured outputs respectively.

The experimental data is plotted in Figure 6.5.

6.3 Simulation versus Real Measurements

The mathematical model and real system often termed as *plant,* are run parallel with the same supplied inputs as shown in Figure 6.6. The real measurements and simulated outputs for different models are given in Figure 6.7. The plotting is shown after the cold-run.

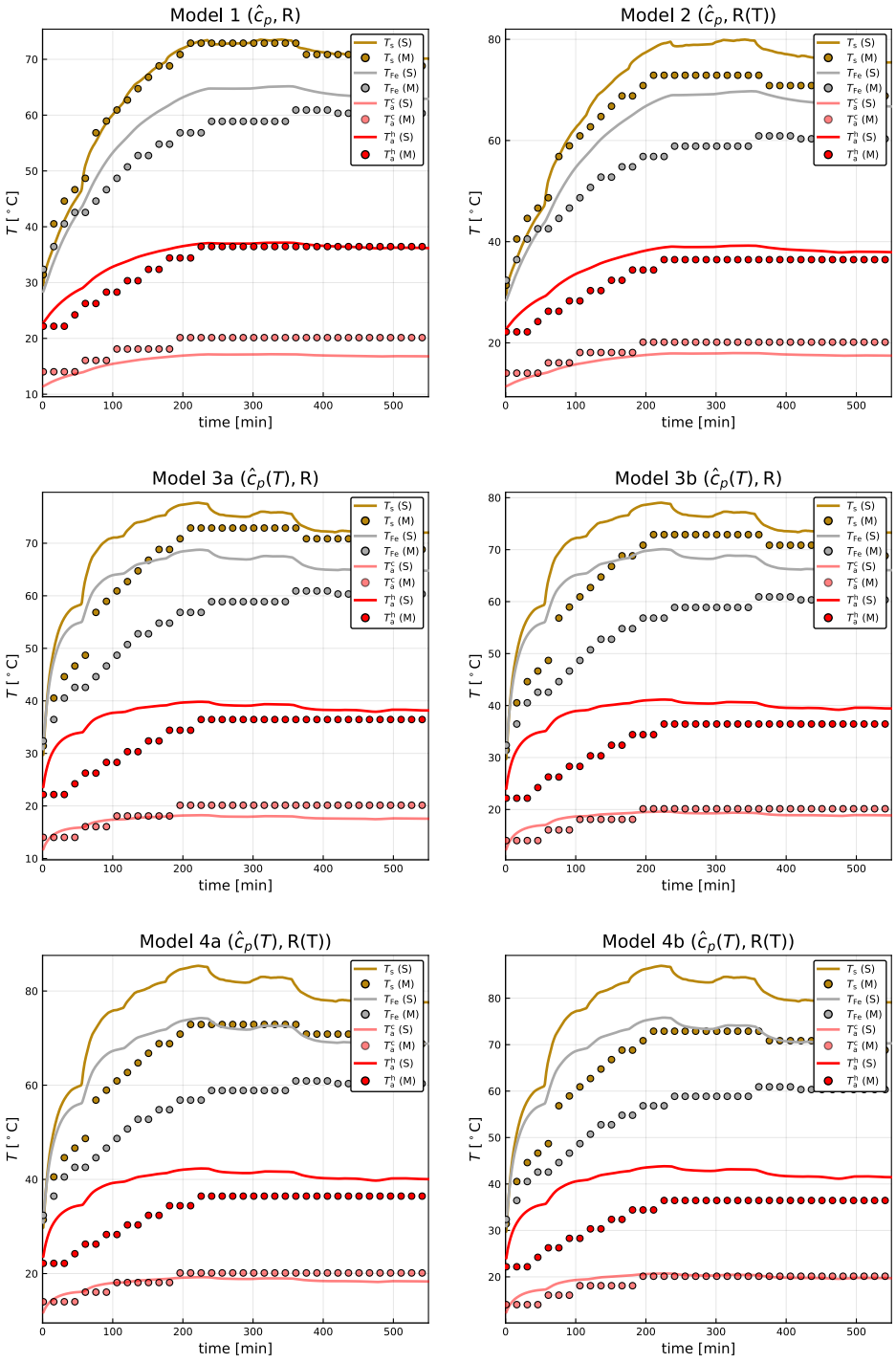


Figure 6.7: Simulation versus real measurements plotted together. T_s(M) represents stator copper temperature measurement and T_s(S) represents simulated output.

6.4 Parameters Optimization

From Figure 6.7 we see, for all of our four models, our simulation does not fit better with real measurements. There might be several possible reasons for that. First, since states are very sensitive to resistance the correct choice of resistance value with errors up to three or four decimal places might be one reason for that. Similarly, while measuring other quantities using sensors, measurement errors due to measurement noises might not be considered. Besides, the temperature distribution inside metals may not be homogenous and the measured temperatures are not true measurements. Few quantities are guessed work and others may be tuned as per hit and trial.

We will be considering Model 1 as our generator model for parameter optimization.

To better fit the model with the real measurements, we can use parameters tuning with hit and trial selection for parameters. However, since we have many parameters it is very tedious to tune parameters with hit and trial method. Another method can be using parameter optimization algorithms for minimizing the *mean-square-error*.

Julia has got several packages for solving optimization problem [20, 21]. However, we choose, *Optim.jl*, explained in [22]. Several optimization algorithms are available, however; we choose to use *box-constrained-optimization*. This gives possibilities for choosing lower and upper bounds for parameters and solve optimization under that bounds. The parameter optimization script using package Optim in Julia for the differential equation is given below,

Code listing for parameter optimization is given as,

```
# Using packages for parameter estimation for differential equations
using DifferentialEquations
using DiffEqParamEstim
# Using Optim.jl package for optimization
using Optim
# Building a loss objective function for differential equation and defining a L2Loss function
    containing data
# Since we have data for Ts and TFe our problem is 3-states problem 2-states measurement data
    optimization problem
# We need to save Ts and TFe only from ODE
t = collect(range(0,stop=583,length=584))
cost_function=build_loss_objective(prob,ABM54(),dt=60.,save_idxs=2:3,L2Loss(t,data),
                                   maxiters=10000,verbose=false)
# Setting 10% lower bounds and upper bounds for parameters
lower = 0.9*[53.9,49.2,2.7,14.3,20.]
upper = 1.1*[53.9,49.2,2.7,14.3,20.]
# Optimizing using Fminbox() method
# Optimized parameter are: mda,mdw,UAr2d,UAs2Fe,UAFe2a
result = optimize(cost_function,lower, upper,[53.9,49.2,2.7,14.3,20.], Fminbox())
# minimized parameters
result.minimizer
```

Table 6.2 shows the chosen parameters for optimization for better model fitting. We are choosing five parameters in which states are having similar sensitivity as shown in Section 6.1. The lower bounds and upper bounds are set to 10% each.

Table 6.2: Parameters optimization for model fitting for Model 1.

Parameters	Nominal values	Lower bounds	Upper bounds	Optimized values
$\dot{m}_{\mathrm{a}}$	53.9 kg/s	10%	10%	48.51 kg/s
$\dot{m}_{\mathrm{w}}$	49.2 kg/s	10%	10%	44.28 kg/s
$\mathscr{U}A_{\mathrm{r2}\delta}$	2.70 kW/K	10%	10%	2.96 kW/K
$\mathscr{U}A_{\mathrm{s2Fe}}$	20.0 kW/K	10%	10%	18.0 kW/K
$\mathscr{U}A_{\mathrm{Fe2a}}$	14.3 kW/K	10%	10%	12.87 kW/K

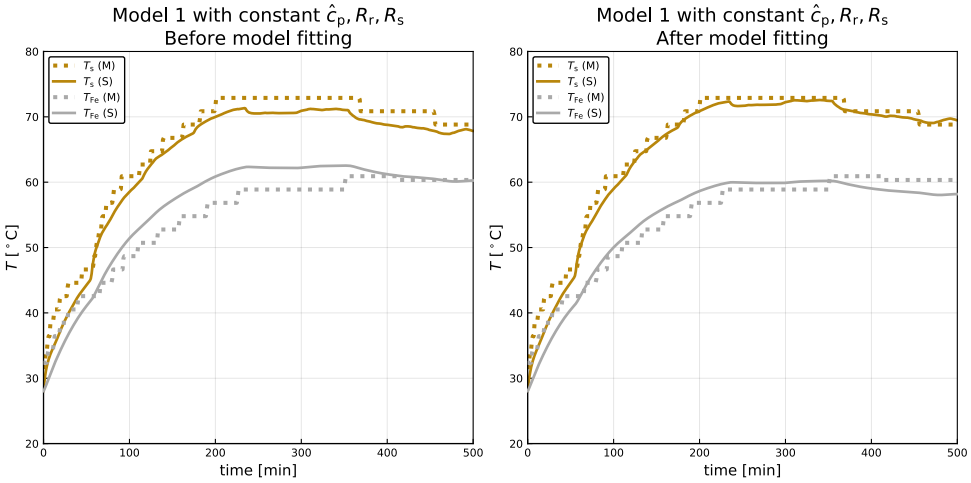


Figure 6.8: Model 1 Fitting using optimized parameters

The code listing written in Jupyter notebook for parameters optimization is given in Appendix C with Jupyter notebook file named as *ParameterOptimization.ipynb*.

The model fitting with optimized parameters for Model 1 is shown in Figure 6.8.

7 State Estimation

State estimation is a process of estimating state variables, which can not be measured directly or not measured because of any reasons, in a control system. For instance, in our generator model we have measurements for two of the states which are T_s and T_{Fe}, while the third state T_r is to be estimated with the help of others.

Several state estimation methods are available; however, we stick to the Kalman filter for our state estimation [23]. There are several derivatives of Kalman filter for different control process [24]. In this thesis report, we only present two of the derivatives, Unscented Kalman Filter (UKF) and Ensemble Kalman Filter (EnKF). UKF and EnKF algorithms, we will be using similar notations, are well compared in [25].

7.1 Introduction on Notations Used

The notations develop in describing an (original) Kalman filter algorithm, and its derivatives are often not superficial to comprehend and use; but varies from text to text, articles to articles. Readers while reading this report are requested to follow notations and their meanings from this Table 7.1.

A nonlinear dynamic system is often represented as,

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1})$$
$$y_k = h(x_k, u_k, v_k)$$

with $w_k \sim \mathcal{N}(\bar{w}_k, \mathcal{W}_k)$ and $v_k \sim \mathcal{N}(\bar{v}_k, \mathcal{V}_k)$.

7.2 Unscented Kalman filter

The original Kalman filter [23], developed for linear systems, was later extended for nonlinear systems with Extended Kalman Filter (EKF)[1]. For a linear system, the mean and

[1]It is a derivate of Kalman filter used for the state estimation of a nonlinear problem linearized around operating points using Jacobians

Table 7.1: Notations and their explanations for UKF and EnKF algorithms

Symbols	Explanations
$x, \bar{x}, \hat{x}$	State vector, its mean, its estimate
x_k	k^{th} component of vector x
$\hat{x}_{k\|k-1}$	*a priori* estimate of x_k on all prior measurements except at time t_k
$\hat{x}_{k\|k}$	*a posteriori* estimate of x_k on all prior measurements including at time t_k
X	State covariance
w	Process noise vector
v	Measurement noise vector
$\mathscr{W}$	Process noise covariance
$\mathscr{V}$	Measurement noise covariance
K	Kalman gain
$\mathscr{E}$	Innovation covariance
Z	Cross covariace between x and y
ε	Innovation, error between measurement and estimate

covariance are exactly propagated using linear (original) Kalman filter while for non-linear systems linearized around operating points using *Jacobians* the mean and covariance are propagated approximately [24]. For systems having extremely nonlinear dynamics, EKF fails for proper estimation; however for propagating mean and covariances in such systems *unscented transformation*[2] is used as given in [26]. The UKF algorithm is given in Table 7.2.

7.3 Ensemble Kalman Filter

EnKF is a Monte Carlo implementation for updating statistical moments for propagating through the control process. Unlike EKF and UKF the statistical moments, such as mean and covariances, are propagated using a random cloud of mean in state space [27]. Therefore, we tend to avoid the formulation of Jacobians and/or unscented transformation in an EnKF. The EnKF is implemented in Table 7.3.

[2]Consider a system with state(s) vector x having mean $\bar{x}$ and covariance X. Then, in unscented transformation $2n$, where n is the number of states, *sigma points*, are propagated through the nonlinear process and measurement equations. Sigma points are *a set of individual points in state space* whose ensemble mean and covariance are equal to $\bar{x}$ and X.

Table 7.2: Algorithm: UKF

Initialization, $k = 1$:

$\hat{x}_{1|1} = \mathbb{E}(x_1) = \bar{x}_1$
$X_{1|1} = X_1$

for $k = 2,\ 3, ...$

Propagation step:

1. Generate *sigma points* using *unscented transformation*
$x^{(i)}_{k-1|k-1} = \hat{x}_{k-1|k-1} + \tilde{x}^{(i)}\ \ i \in \{1, 2, ...2n\}$
where,
$\tilde{x}^{(i)} = \left(\sqrt{nX_{k-1|k-1}}\right)^T_i\ \ i \in \{1, 2, ...n\}$
$\tilde{x}^{(n+i)} = -\left(\sqrt{nX_{k-1|k-1}}\right)^T_i\ \ i \in \{1, 2, ...n\}$

2. Propagate sigma points through process model
$x^{(i)}_{k|k-1} = f\left(x^{(i)}_{k-1|k-1}, u_{k-1}, \bar{w}_k\right)\ \ i \in \{1, 2, ...2n\}$

3. *a priori* state and covariance estimate
$\hat{x}_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} x^{(i)}_{k|k-1}$
$X_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)^T + \mathscr{W}_k$

Information update:

(Note: We are using the same sigma points from propagation step; however, we can generate new sets of sigma points same as in propagation step and proceed further. It is a trade-off between computational cost and performance of the filter.)

1. Propagate sigma points through measurement equation
$y^{(i)}_{k|k-1} = h\left(x^{(i)}_{k-1|k-1}, u_{k-1}, \bar{v}_k\right)\ \ i \in \{1, 2, ...2n\}$

2. Predicted measurements
$\hat{y}_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} y^{(i)}_{k|k-1}$

3. Innovation and cross covariance
$\mathscr{E}_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n}\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T + \mathscr{V}_k$
$Z_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$

4. Kalman gain
$K_k = Z_{k|k-1}\mathscr{E}^{-1}_{k|k-1}$

5. *a posteriori* update
$\varepsilon_{k|k-1} = y_k - \hat{y}_{k|k-1}$
$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\varepsilon_{k|k-1}$
$X_{k|k} = X_{k|k-1} - K_k\mathscr{E}_{k|k-1}K^T_k$

Table 7.3: Algorithm: EnKF

Initialization, $k = 1$:

$x^i_{1|1} \sim \mathcal{N}(\bar{x}_1, X_1),\ i \in \{1,2,...,n_p\}$
$w^i_k \sim \mathcal{N}(\bar{w}_1, \mathcal{W}_k),\ i \in \{1,2,...,n_p\}$
$v^i_k \sim \mathcal{N}(\bar{v}_1, \mathcal{V}_k),\ i \in \{1,2,...,n_p\}$

$\hat{x}_{1|1} = \frac{1}{n_p} \sum_{i=1}^{n_p} x^{(i)}_{1|1}$
$X_{1|1} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} \left(x^{(i)}_{1|1} - \hat{x}_{1|1}\right) \left(x^{(i)}_{1|1} - \hat{x}_{1|1}\right)^T$

for $k = 2,\ 3, ...$

Propagation step:

1. Propagate particles through process model
$x^{(i)}_{k|k-1} = f\left(x^{(i)}_{k-1|k-1}, u_{k-1}, w^{(i)}_{k-1}\right)\ i \in \{1,2,...,n_p\}$

2. *a priori* state and covariance estimates
$\hat{x}_{k|k-1} = \frac{1}{n_p} \sum_{i=1}^{n_p} x^{(i)}_{k|k-1}$
$X_{k|k-1} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} \left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right) \left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)^T$

Information update:

1. Propagate particles through measurement equation
$y^{(i)}_{k|k-1} = h\left(x^{(i)}_{k-1|k-1}, u_{k-1}, v^{(i)}_{k-1}\right)\ i \in \{1,2,...,n_p\}$

2. Predicted measurements
$\hat{y}_{k|k-1} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} y^{(i)}_{k|k-1}$

3. Innovation and cross covariance
$\mathcal{E}_{k|k-1} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} \left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right) \left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$
$Z_{k|k-1} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} \left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right) \left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$

4. Kalman gain
$K_k = Z_{k|k-1} \mathcal{E}^{-1}_{k|k-1}$

5. *a posteriori* update of state and covariance
$\varepsilon^{(i)}_{k|k-1} = y_k - y^{(i)}_{k|k-1}$
$x^{(i)}_{k|k} = x^{(i)}_{k|k-1} + K_k \varepsilon^{(i)}_{k|k-1}$
$\hat{x}_{k|k} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} x^{(i)}_{k|k}$
$X_{k|k} = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} \left(x^{(i)}_{k|k} - \hat{x}_{k|k}\right) \left(x^{(i)}_{k|k} - \hat{x}_{k|k}\right)^T$

8 Results and Discussion

This chapter provides important results after implementation of several tasks carried out in earlier Chapters. First, results from model implementation are discussed. Second, results from the model analysis are presented and finally, we will present results from state estimation.

8.1 Model Implementation

The computational speed for the model implemented[1] in OpenModelica, OMJulia, and Julia for solving DAE models can be compared in Table 8.1. The solver used for OpenModelica and OMJulia is DASSL and IDA for Julia.

The computational time for solving models in Julia is faster as compared to OpenModelica and OMJulia. The computational time required for solving models using OMJulia is the highest among Julia and OpenModelica and this is because models are solved using OpenModlica API in Julia. In OMJulia models are first solved using OpenModelica in the backend and render to Julia with few time lags, and thus computational time for solving models increases.

The computational time for solving Model 3b and Model 4b are higher even in Julia. It is because of the fact that the two point boundary value problem should be solved when specific heat capacities for fluids (air, water) depends on the temperature inside the heat exchanger. The computational cost for the numerical solution of models with boundary value problem is higher. When model complexity increases the computational time for solving models increases. For instance, computational time for solving models with constant $\hat{c}_p$ and R is less than for model with $\hat{c}_p(\mathrm{T})$ and $R(\mathrm{T})$.

Although, solving DAE models in Julia have less computational time the preliminary model implementation is often tedious in Julia. The model implementation in Modelica does not require initial values for algebraic variables, and are implicitly solved by the

[1] The simulation environment is set up with "`Processor: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2901 Mhz, 2 Core(s), 4 Logical Processor(s)`". The OpenModelica is `Official Release Version 1.13.2 (64bit)`, OMJulia with `Version 0.0.0` and Julia programming language `Version 1.0.3 (2018-12-18)`.

Table 8.1: Comptational speed for solving DAE models using OpenModelica, OMJulia and Julia. The mean time is taken from 1000 runs for Julia, 100 runs for OMJulia, and for OpenModelica the sample is taken from 10 runs. The simulation time is for 500 min.

Model	OpenModelica	OMJulia	Julia
Model 1	40.9 ms	9.39 s	3.36 ms
Model 2	50.72 ms	9.18 s	3.98 ms
Model 3a	52.30 ms	9.85 s	4.86 ms
Model 3b	-	-	306 ms
Model 4a	51.68 ms	9.48 s	4.98 ms
Model 4b	-	-	297 ms

compiler, however in Julia initial values for each algebraic variables should be explicitly defined.

Figure 8.1 shows a subjective comparison for solving DAE models in Julia, OMJulia and OpenModelica.

8.2 Model Analysis

1. Figure 4.1 compares simulated outputs for Model 1 and 2, and Model 1 and 3a. It shows that generator metal temperatures are more sensitive for models with $R(\mathrm{T})$ than models with $\hat{c}_p(\mathrm{T})$, however; opposite for air and water temperatures. For models with $R(\mathrm{T})$, there is an increase in heat transfer between air and metals, however; models with $\hat{c}_p(T)$ heat transfer between air and metals have few or almost no difference at steady state.

2. Figure 4.2 shows comparison of Model 3a and 3b, and Model 4a and 4b. Model 3a and 4a are models considering constant specific heat capacity inside heat exchanger while Model 3b and 4b are models considering temperature dependent specific heat capacity. The generator metals and air temperatures have very few difference in temperature between Model 3a and 3b, and Model 4a and 4b, around $2-4\,^{\circ}\mathrm{C}$; however, almost no difference for heat transfer and hot water temperature. The hot water temperature for Model 3b has an abrupt change from $14\,^{\circ}\mathrm{C}$ to $1\,^{\circ}\mathrm{C}$ at around 250 min but regain temperature of $9\,^{\circ}\mathrm{C}$ and becomes steady after 350 min. Similarly, hot water temperature for Model 4b has an abrupt change from $14\,^{\circ}\mathrm{C}$ to $1\,^{\circ}\mathrm{C}$ at around 120 min but regain temperature of $9\,^{\circ}\mathrm{C}$ and becomes steady after 140 min. The cost of solving the numerical solution for Model 3b and 4b is higher than Model 3a and 4a because of numerical solution for solving boundary value problem for heat

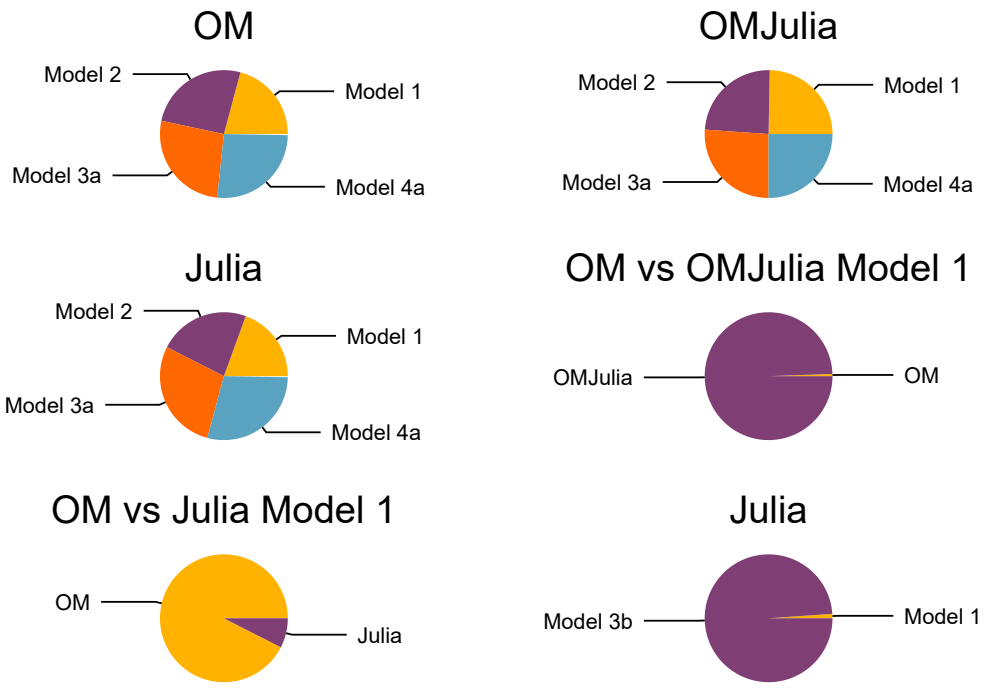


Figure 8.1: Subjective comparison for computational time for solving DAE models in OpenModelica, OMJulia and Julia. *OM* in the figure refers to OpenModelica. The computational time in Julia is faster than OpenModelica and OMJulia. Solving Model 3b in Julia has higher computational time than solving Model 1 because of a two point boundary value numerical solution for heat exchanger for Model 3b.

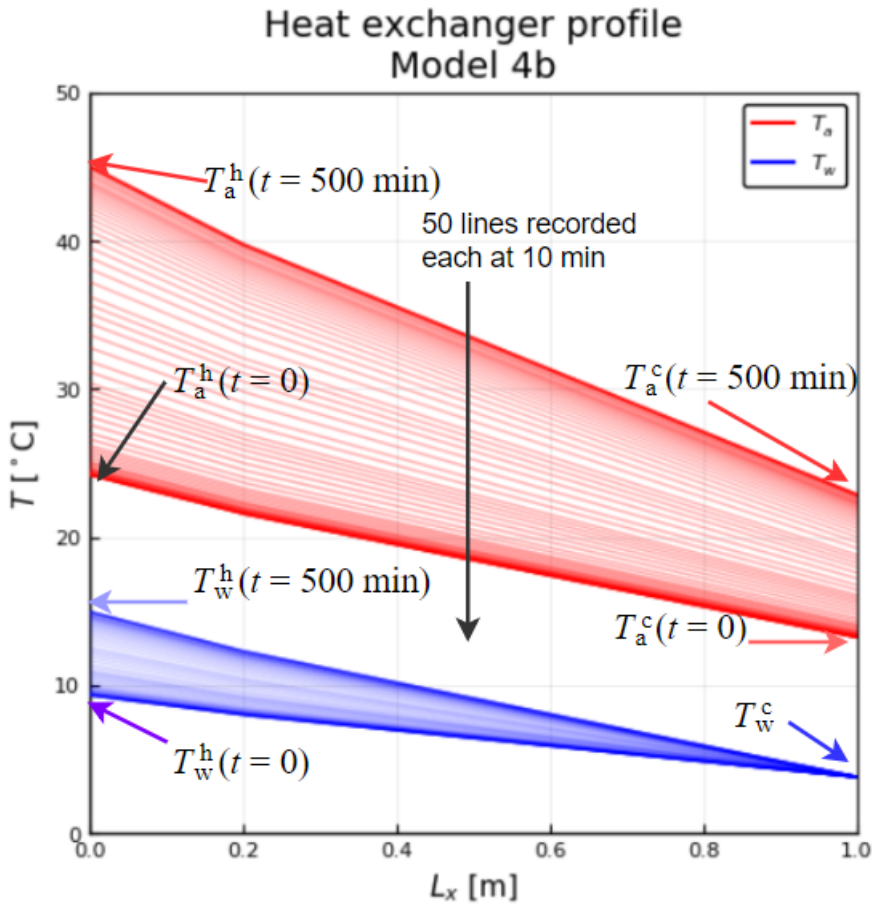


Figure 8.2: Heat exchanger profile at $t = 0$ to $t = 500\,\text{min}$ for Model 4b

exchanger. There is a tradeoff between model accuracy, and computational speed. The heat transfer rate from air to water increases as the model complexity increases from Model 1 to 2 to 3a to 4a.

3. Figure 4.3 and 8.2 shows heat exchanger profiles for all the models. As the model complexity increases from constant R and $\hat{c}_p$ to temperature dependent, the temperature range for air and water inside heat exchanger changes for nominal inputs. A careful observation from the heat exchanger profiles from Model 4a and 4b show that Model 4b considering temperature dependent $\hat{c}_p$ inside the heat exchanger has substantial decrease in T_a^h around $2-3\,^\circ\text{C}$ than Model 4a with constant $\hat{c}_p$ inside the heat exchanger, however, T_w^h increases for Model 4b than to Model 4a. Similar results can be seen for Model 3a and 3b. This indicates that the model with temperature dependent $\hat{c}_p$ has better heat transfer from air to water representing a more realistic model in terms of heat transfer.

4. Figure 6.1, 6.2, 6.3, and 6.4 shows sensitivity in states due to various parameters.

It shows that states are more sensitive to resistances than any other parameters. Figure 6.4 shows that T_r and T_s are more sensitive to R_r and R_s than T_{Fe}. States are least sensitive to the generator metal masses as shown in Figure 6.3.

8.3 State Estimation

Before starting state estimation we set initial values for process noise covariance ($\mathscr{W}$) and measurement noise covariance ($\mathscr{V}$). UKF and EnKF are initialized with $\mathscr{W} = K_T * diag(4,4,4)$ and $\mathscr{V} = diag(1,1)$ and state covariance as $X = 10 * \mathscr{W}$, where K_T is tuning parameter.

It is often hard to find $\mathscr{W}$. After possible tuning the filter by hit and trial with K_T, the closer value was found to be 1.5. Both the process noise w and measurement noise v are considered to be *white Gaussian noise* with zero-mean. The simulation step Δt is set to 1 min and the total time of simulation is 584 min.

The simulation environment is setup with "`Processor: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2901 Mhz, 2 Core(s), 4 Logical Processor(s)`" in Julia programming language `Version 1.0.3 (2018-12-18)`.

For comparison of UKF and EnKF, air and metals temperatures estimates, root mean square error (RMSE) of innovation residuals, $\varepsilon = y_k - \hat{y}_{k|k-1}$, its covariance $\mathscr{E}$, and computational time of filters are taken into consideration.

Several important results can be obtained from metals and air temperature estimation using UKF and EnKF with different n_p.

8.3.1 Comparison of UKF and EnKF based on temperatures estimation

It can be seen from Figure 8.3, for estimated rotor copper and air gap temperatures for Model 1; for EnKF as the number of particles n_p increases EnKF converge approximately as the UKF estimates. For instance, EnKF estimation when increased from $n_p = 50$ to $n_p = 1000$ gives similar estimation with UKF. Similarly, from Figure 8.6 and 8.4, for Model 2, UKF and EnKF with $n_p = 1000$ gives similar rotor temperature estimates.

However; from Figure 8.8, for rotor temperature estimates, for Model 3a and 3b, and Model 4a and 4b, UKF and EnKF with $n_p = 1000$ gives very different results with an average difference of 10 °C.

Figure 8.5 and 8.7 shows that air temperatures estimates from UKF and EnKF with $n_p = 1000$ give similar results. Similarly, stator copper and stator iron temperatures estimates for both UKF and EnKF with $n_p = 1000$ give approximately the same results.

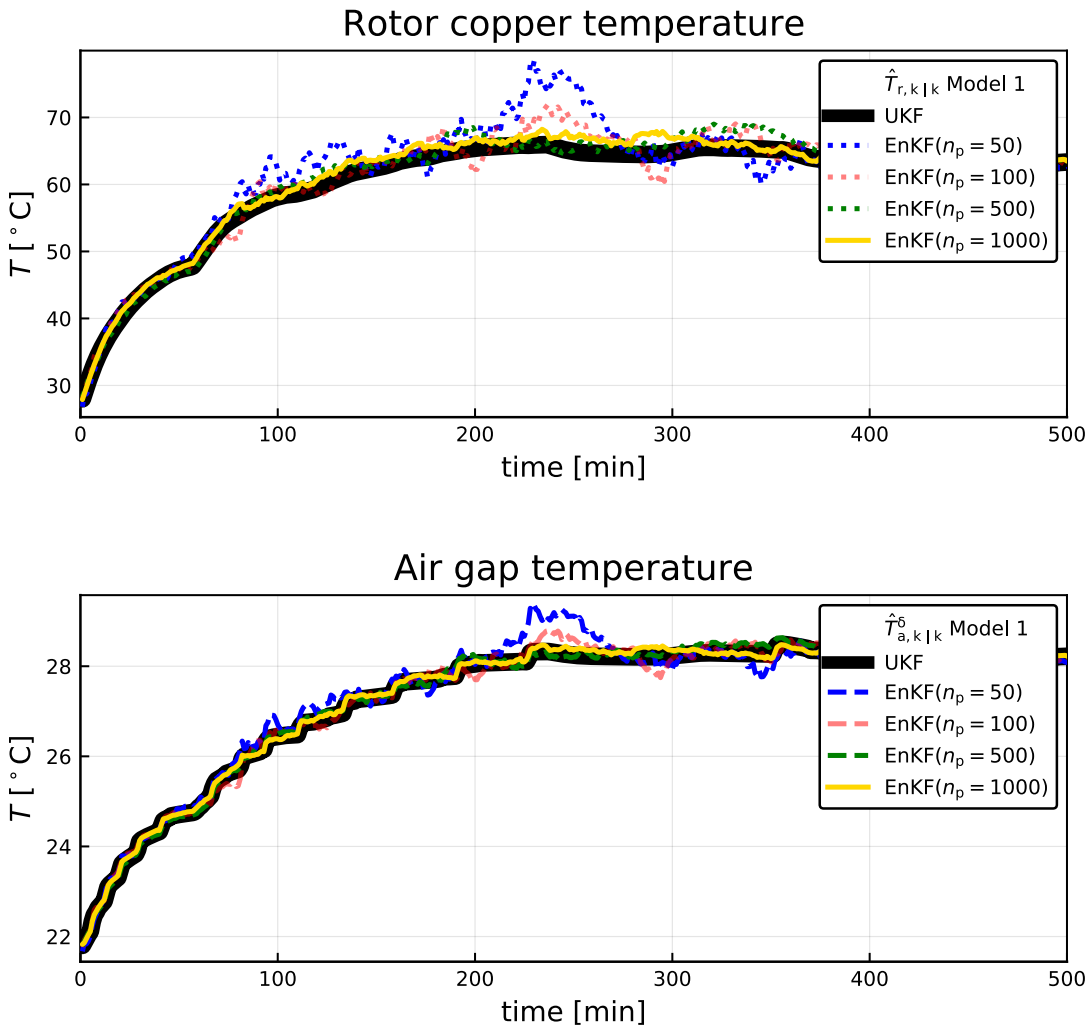


Figure 8.3: Rotor and air gap temperature estimation using UKF and EnKF with different n_p.

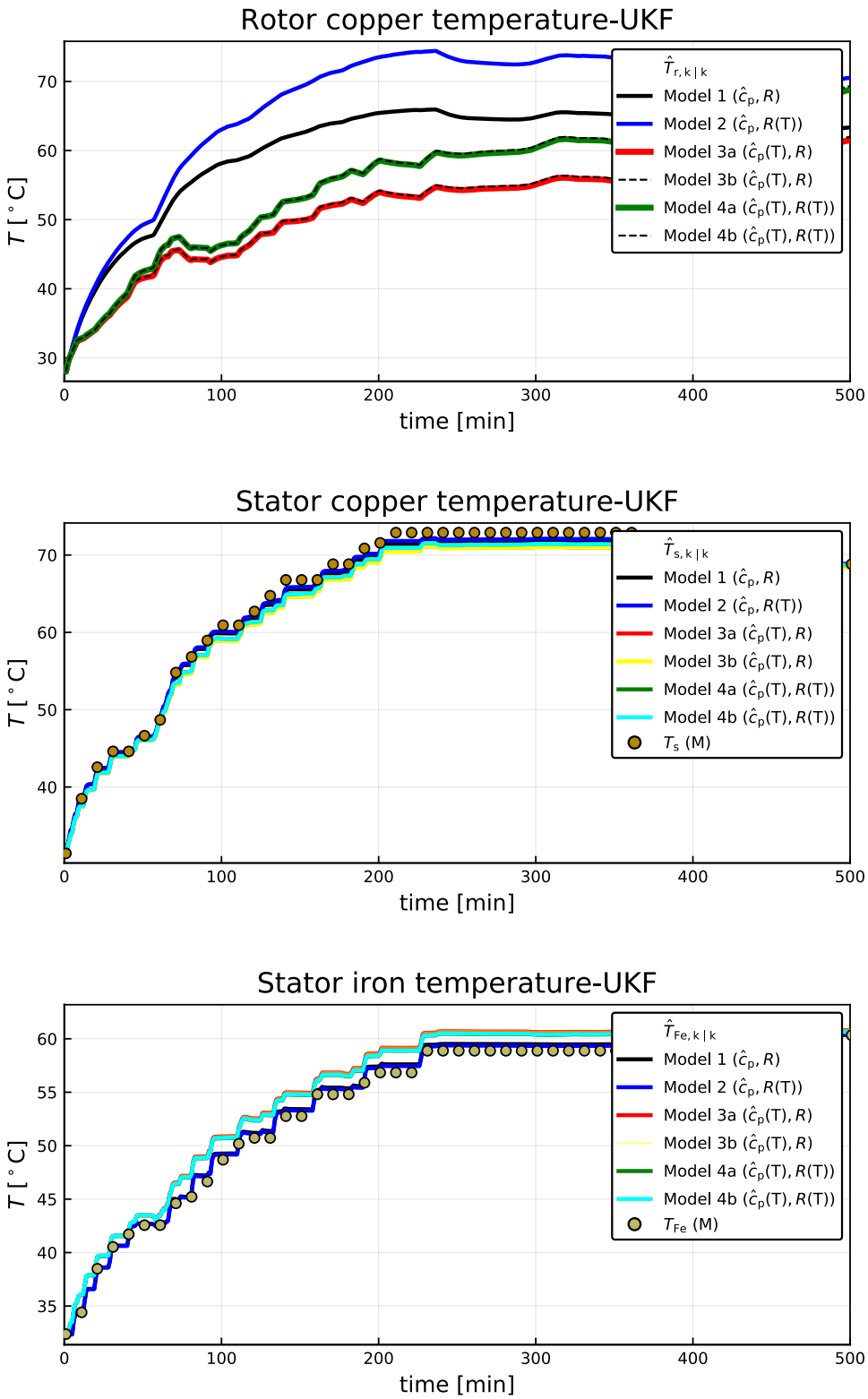


Figure 8.4: Metals temperatures estimation using UKF for different models.

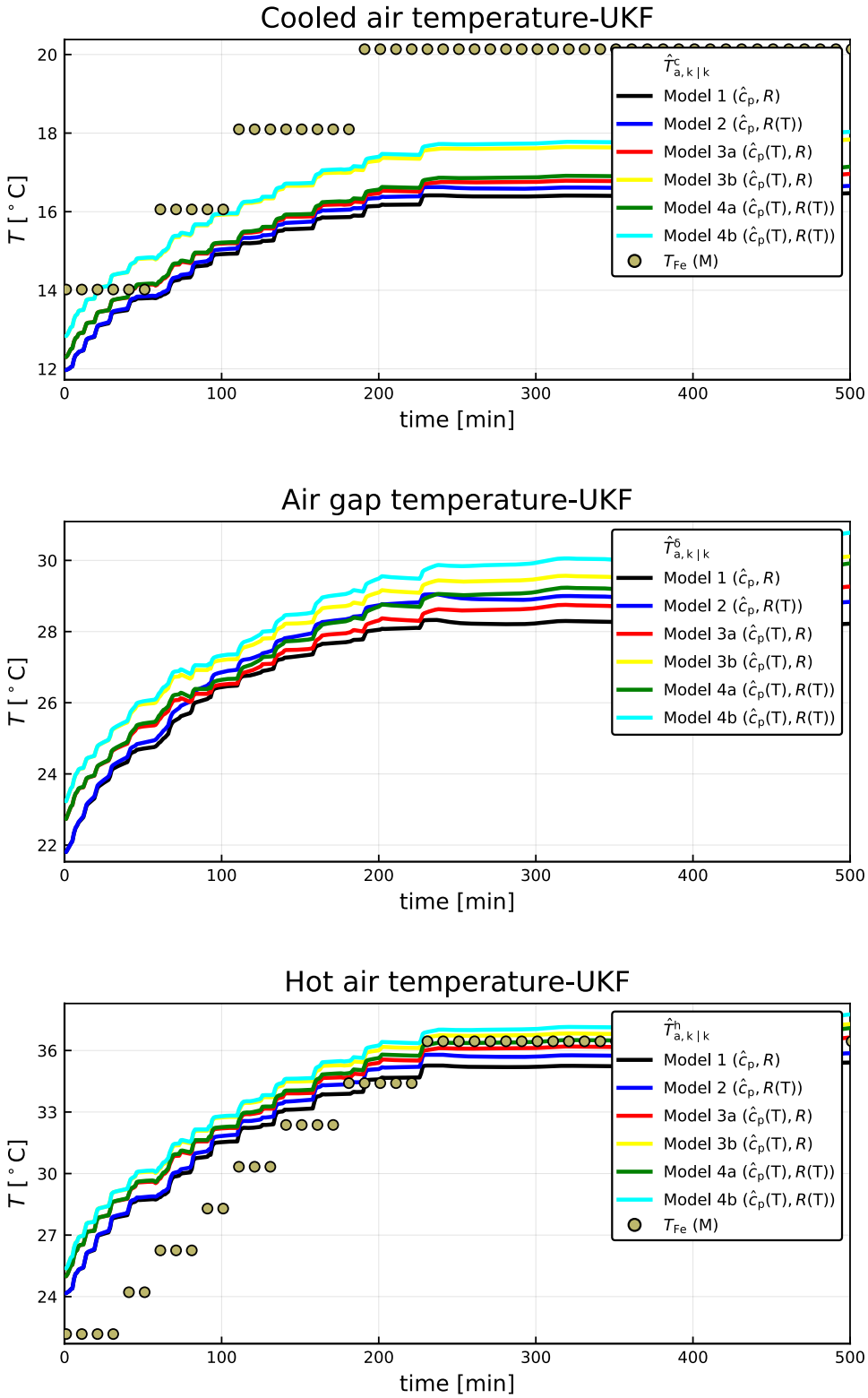


Figure 8.5: Air temperatures estimation using UKF for different models.

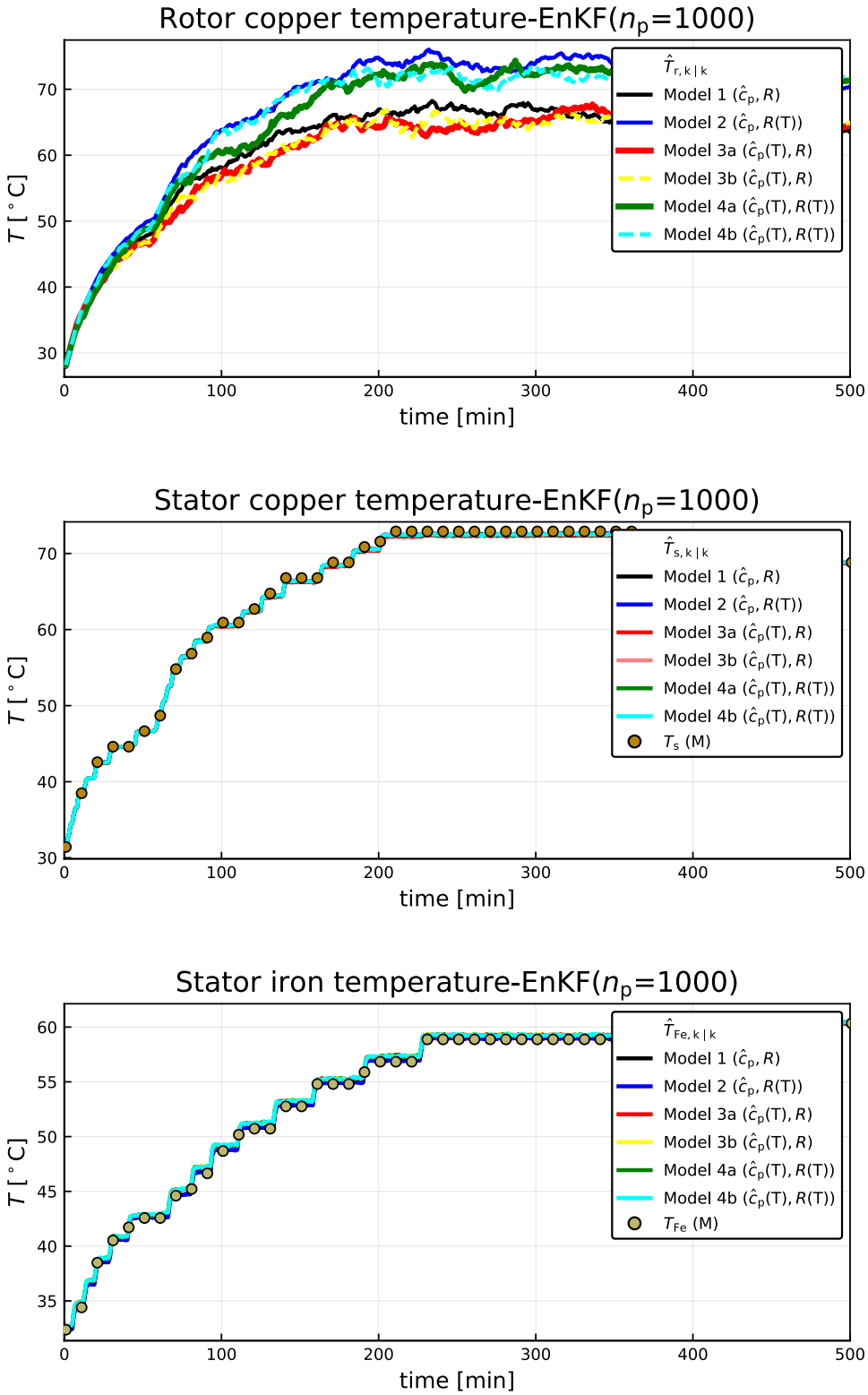


Figure 8.6: Metals temperatures estimation using EnKF ($n_p = 1000$) for different models.

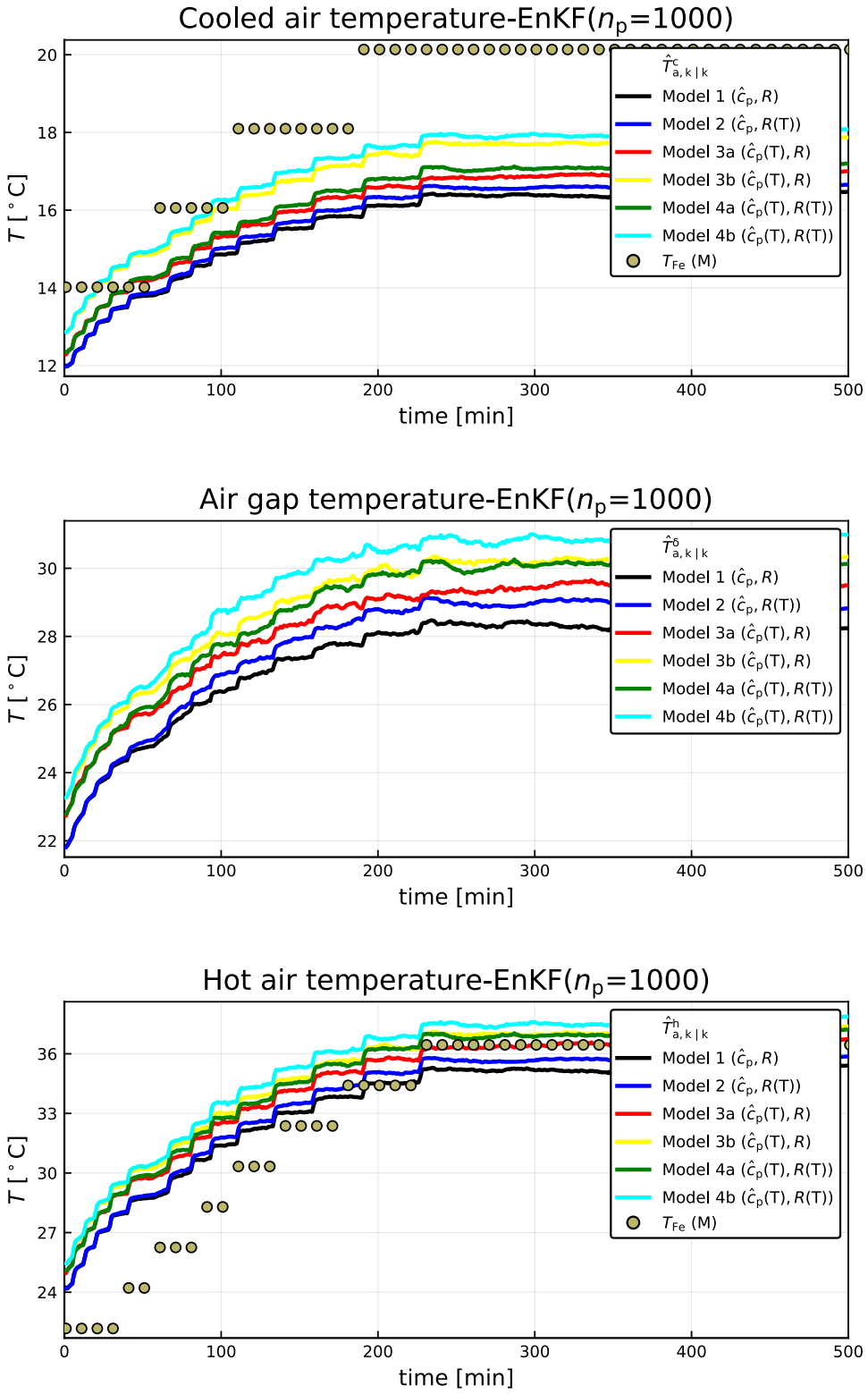


Figure 8.7: Air temperatures estimation using EnKF ($n_p = 1000$) for different models.

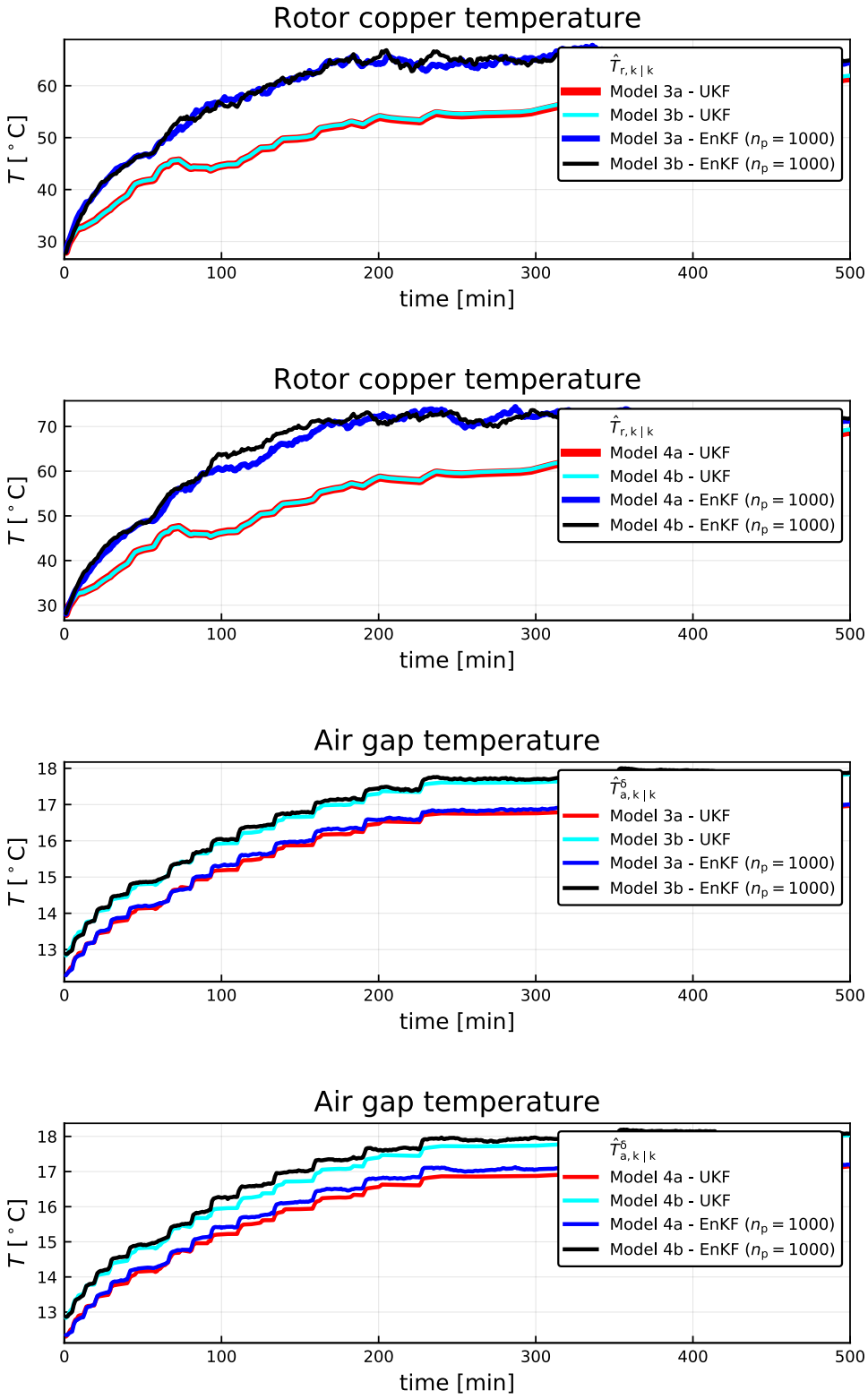


Figure 8.8: Air temperatures estimation using EnKF ($n_p = 1000$) for different models.

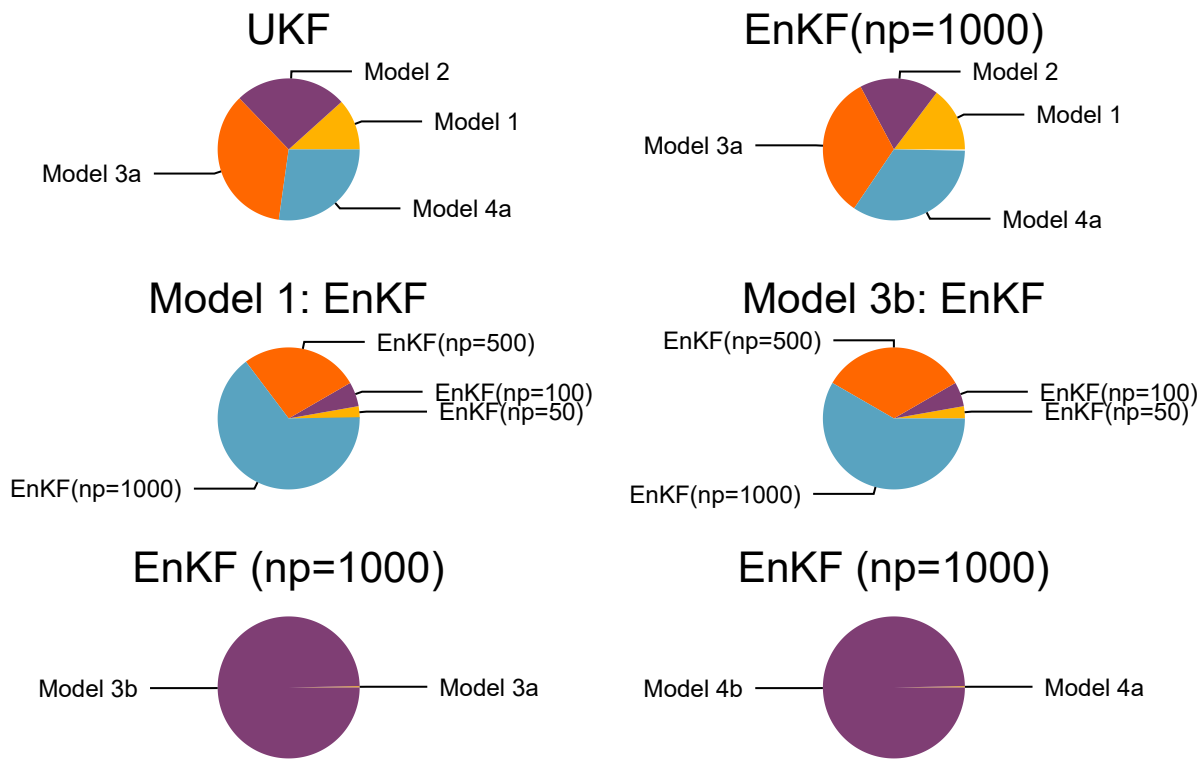


Figure 8.9: Comparison of computational time for UKF and EnKF with different models.

Figure 8.4 shows that rotor temperature estimates decreases from models with constant $\hat{c}_p$and $\boldsymbol{R}$ to temperature dependent $\hat{c}_p$ and $\boldsymbol{R}$. UKF estimation of rotor temperature is exactly same for Model 3a and 3b which shows that there is no difference in estimation either constant or temperature dependent $\hat{c}_p$ is taken inside heat exchanger is considered or not. And, a similar result is obtained from Model 4a and 4b.

As the model complexity increases from Model 1 to Model 4b, Figure 8.5 and 8.7 shows that estimation in air temperatures values increases from simple to complex.

8.3.2 Comparison of UKF and EnKF based on estimation accuracy and computational time

We can compare UKF and EnKF based on estimation accuracy measuring RMSE of innovation residuals $\boldsymbol{\varepsilon}$ and its covariance $\mathscr{E}$, and the computational time for each filter.

8.2

Table 8.2: Comparison of UKF and EnKF based on RMSE of residual ε and its covariance $\mathscr{E}$, and computational time. T_{sim} is the total simulation time.

Model	KF	RMSE (ε)	RMSE ($\mathscr{E}$)	Elapsed ($\Delta t = 1\,\text{min}$)	Elapsed ($T_{\text{sim}} = 584\,\text{min}$)
1	UKF	2.214	7.203	0.578 ms	0.338 s
	EnKF $(n_{\text{p}} = 50)$	2.066	8.662	1.863 ms	1.088 s
	EnKF $(n_{\text{p}} = 100)$	2.039	8.705	3.785 ms	2.211 s
	EnKF $(n_{\text{p}} = 500)$	2.010	8.636	18.595 ms	10.860 s
	EnKF $(n_{\text{p}} = 1000)$	2.012	8.551	45.102 ms	26.343 s
2	UKF	1.652	7.241	1.274 ms	0.744 s
	EnKF $(n_{\text{p}} = 50)$	1.573	8.634	3.037 ms	1.774 s
	EnKF $(n_{\text{p}} = 100)$	1.524	8.778	5.845 ms	3.414 s
	EnKF $(n_{\text{p}} = 500)$	1.500	8.518	28.645 ms	16.729 s
	EnKF $(n_{\text{p}} = 1000)$	1.492	8.523	55.179 ms	32.225 s
3a	UKF	3.137	5.970	1.782 ms	1.041 s
	EnKF $(n_{\text{p}} = 50)$	2.725	8.386	5.545 ms	3.238 s
	EnKF $(n_{\text{p}} = 100)$	2.700	8.143	13.087 ms	7.643 s
	EnKF $(n_{\text{p}} = 500)$	2.706	8.178	62.780 ms	36.663 s
	EnKF $(n_{\text{p}} = 1000)$	2.699	8.243	100.33 ms	58.595 s
3b	UKF	3.200	5.971	205.582 ms	2.001 min
	EnKF $(n_{\text{p}} = 50)$	2.774	8.360	1.370 s	13.335 min
	EnKF $(n_{\text{p}} = 100)$	2.783	8.267	2.662 s	25.916 min
	EnKF $(n_{\text{p}} = 500)$	2.747	8.255	0.270 min	2.63 hr
	EnKF $(n_{\text{p}} = 1000)$	2.741	8.259	0.471 min	4.59 hr
4a	UKF	2.730	6.010	1.366 ms	0.798 s
	EnKF $(n_{\text{p}} = 50)$	2.394	8.650	5.402 ms	3.154 s
	EnKF $(n_{\text{p}} = 100)$	2.375	8.386	9.053 ms	5.287 s
	EnKF $(n_{\text{p}} = 500)$	2.337	8.251	61.433 ms	35.877 s
	EnKF $(n_{\text{p}} = 1000)$	2.330	8.319	104.44 ms	60.993 s
4b	UKF	2.802	6.012	131.506 ms	1.280 min
	EnKF $(n_{\text{p}} = 50)$	2.470	8.539	974.075 ms	9.481 min
	EnKF $(n_{\text{p}} = 100)$	2.423	8.531	2.65 s	25.878 min
	EnKF $(n_{\text{p}} = 500)$	2.401	8.223	0.259 min	2.528 hr
	EnKF $(n_{\text{p}} = 1000)$	2.400	8.257	0.495 min	4.820 hr

8.3.2.1 Estimation accuracy

It can be seen from Table 8.2 that RMSE of innovation residuals ($\boldsymbol{\varepsilon}$) is least for Model 2 while for other models it is almost the same. Also, RMSE of $\mathscr{E}$ is not that much different from one model to another. However, residuals of $\boldsymbol{\varepsilon}$ are higher for UKF than EnKF and residuals of $\mathscr{E}$ is lower for EnKF than UKF. Furthermore, residuals decrease as we increase the number of particles in EnKF from $n_\mathrm{p} = 50$ to 100 to 500 to 1000. Thus, EnKF estimation has less estimation error while we increase the number of particles.

8.3.2.2 Computational time

From Table 8.2 it can be seen that computational time increases when model complexity increases from Model 1 to Model 4, that is, models with constant $\hat{c}_p$ and $\boldsymbol{R}$ have less computational time than the model with temperature dependent $\hat{c}_p$ and $\boldsymbol{R}$.

Similarly, UKF has less computational time than EnKF. Furthermore, computational time increases as the number of particles increases for EnKF.

Model 3b and 4b have the highest computational time as compared to other models and it is because considering $\hat{c}_p(\boldsymbol{T})$ inside heat exchanger for air and water a numerical solution needs to be solved for the tow point boundary value problem. For a total simulation time of $584\,\mathrm{min}$ state estimation for Model 3b using EnKF $\left(n_\mathrm{p} = 1000\right)$ is $4.59\,\mathrm{hr}$ while for Model 3a is only $58.595\,\mathrm{s}$. Similar results can be seen for Model 4a and 4b.

For a single iteration of $\Delta t = 1\,\mathrm{min}$, it can be seen that UKF is faster, than EnKF with different particles for all models, and has less computational time.

Figure 8.9 shows a subjective comparison of computational time for UKF and EnKF. Also, it shows that computational time for Model 3a and 3b, and Model 4a and 4b have a vast difference because Model 3b and 4b are based on the numerical solution of boundary value problem for the heat exchanger which requires higher computational time.

9 Future Works

For further studies, to be very specific,

- the computational time for solving boundary value problem for heat exchanger, for Model 3b and 4b, can be improved,
- analysis of linearization, stability, observability, and controllability of models other than Model 1 and 2 can be performed,
- a few of the parameters are only used for parameter optimization for fitting Model 1, all parameters can be considered for fitting Model 1, parameter optimization for other models can also be tested and verified,
- local parameter sensitivity analysis performed in this thesis can be compared with global parameter sensitivity analysis,
- few other state estimation algorithms like Particle filter, different versions of EnKF and UKF can be further developed and compared,
- models can be solved using other possible platforms other than Julia, OMJulia and Modelica, like MATLAB, *modia.jl* [1],etc., it is often interesting to compare computational time for solving complex models.

[1] https://github.com/ModiaSim/Modia.jl

10 Conclusion

The central focus of this thesis is to develop and have state estimation for the temperature of the metals of the thermal model of the air-cooled synchronous generator. The models are developed and analyzed using constant and temperatures dependent resistances of rotor and stator copper windings, and specific heat capacities of fluids (air, water). Six DAE and ODE models have been purposed, simulated, and out of six, the computational time for four models have been compared in Julia, OMJulia, and OpenModelica. For steady state, the two point boundary value solution of the heat exchanger when fluids' specific heat capacities depending on temperature, a numerical solution is purposed using Julia language. UKF and EnKF based state estimation algorithms are used for generator metals and air temperature estimation. Both algorithms are compared based on estimation accuracy and computational time. UKF performs better than EnKF based on computational speed while EnKF performs better based on estimation accuracy. Model 2 with temperature dependent resistance has better estimation accuracy based on RMSE of innovation residuals of estimators than any other models.

Bibliography

[1] B. Lie, "Project, fm1015 modelling of dynamic systems," University of South-Eastern Norway, Sep. 2018, group project task.

[2] ENTSO-E, "Commission regulation (eu) 2016/631 of 14 april 2016 establishing a network code on requirements for grid connection of generators. technical report, european net- work of transmission system operators for electricity, entso-e avenue de cortenbergh 100 1000 brussels bel- gium," Tech. Rep., 2016.

[3] Statnett, "Fiks funksjonskrav i kraftsystemet [functional require- ments in the power system]. technical report," Tech. Rep., 2012.

[4] A. Boglietti, A. Cavagnino, D. Staton, M. Shanel, M. Mueller, and C. Mejuto, "Evolution and modern approaches for thermal analysis of electrical machines," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 3, pp. 871–882, March 2009.

[5] A. M. EL-Refaie, N. C. Harris, T. M. Jahns, and K. M. Rahman, "Thermal analysis of multibarrier interior pm synchronous machine using lumped parameter model," *IEEE Transactions on Energy Conversion*, vol. 19, no. 2, pp. 303–309, June 2004.

[6] S. Nategh, O. Wallmark, M. Leksell, and S. Zhao, "Thermal analysis of a pmasrm using partial fea and lumped parameter modeling," *IEEE Transactions on Energy Conversion*, vol. 27, no. 2, pp. 477–488, June 2012.

[7] N. Rostami, M. R. Feyzi, J. Pyrhonen, A. Parviainen, and M. Niemela, "Lumped-parameter thermal model for axial flux permanent magnet machines," *IEEE Transactions on magnetics*, vol. 49, no. 3, pp. 1178–1184, 2013.

[8] F. Marignetti, V. D. Colli, and Y. Coia, "Design of axial flux pm synchronous machines through 3-d coupled electromagnetic thermal and fluid-dynamical finite-element analysis," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 10, pp. 3591–3601, 2008.

[9] Y.-K. R. Chin, E. Nordlund, and A. Staton, "Thermal analysis-lumped-circuit model and finite element analysis," in *IPEC 2003: 6th International Power Engineering Conference*. NANYANG TECHNOLOGICAL UNIV, 2003, pp. 952–957.

[10] B. Zhang, R. Qu, W. Xu, J. Wang, and Y. Chen, "Thermal model of totally enclosed water-cooled permanent magnet synchronous machines for electric vehicle applications," in *2014 International Conference on Electrical Machines (ICEM)*. IEEE, 2014, pp. 2205–2211.

[11] T. Øyvang, J. K. Nøland, G. J. Hegglid, and B. Lie, "Online model-based thermal prediction for flexible control of an air-cooled hydrogenerator," *IEEE Transactions on Industrial Electronics*, 2018.

[12] T. Øyvang, "Enhanced power capability of generator units for increased operational security," Porsgrunn, 2013.

[13] B. Lie, "Modeling of dynamic systems," Aug. 2017a, lecture notes, Version of August 8.

[14] B. J. McBride, M. J. Zehe, and S. Gordon, "Nasa glenn coefficients for calculating thermodynamic properties of individual species," 2002.

[15] M. J. Zehe, S. Gordon, and B. J. McBride, "Cap: A computer code for generating tabular thermodynamic functions from nasa lewis coefficients," 2002.

[16] P. Fritzson, *Principles of object-oriented modeling and simulation with Modelica 3.3: a cyber-physical approach*. John Wiley & Sons, 2014.

[17] C. Rackauckas and Q. Nie, "Differentialequations. jl–a performant and feature-rich ecosystem for solving differential equations in julia," *Journal of Open Research Software*, vol. 5, no. 1, 2017.

[18] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto, *Sensitivity analysis in practice: a guide to assessing scientific models*. Wiley Online Library, 2004.

[19] D. G. Cacuci, *Sensitivity & uncertainty analysis, volume 1: Theory*. Chapman and Hall/CRC, 2003.

[20] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[21] JuliaNLSolvers, "Julianlsolvers/lsqfit.jl," Apr 2019. [Online]. Available: https://github.com/JuliaNLSolvers/LsqFit.jl

[22] P. K. Mogensen and A. N. Riseth, "Optim: A mathematical optimization package for Julia," *Journal of Open Source Software*, vol. 3, no. 24, p. 615, 2018.

[23] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[24] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[25] O. Brastein, B. Lie, R. Sharma, and N.-O. Skeie, "Parameter estimation for externally simulated thermal network models," *Energy and Buildings*, 2019.

[26] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068. International Society for Optics and Photonics, 1997, pp. 182–194.

[27] G. Evensen, "The ensemble kalman filter: Theoretical formulation and practical implementation," *Ocean dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

Appendix A

Task Descriptions

University of South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences, Campus Porsgrunn

FMH606 Master's Thesis

Title: Model Fitting and State Estimation for Thermal Model of Synchronous Generator

USN supervisor: Bernt Lie, co-supervisor Thomas Øyvang

External partner: Skagerak Kraft

Task background:
Currently, strict constraints are imposed on the operation of synchronous generators for hydropower production: the Power Factor ("cos(ϕ)"; lies in the range [-1,1]) is typically restricted to be less than 0.86. The Power Factor is the ratio of *active* power to *apparent*/total power. Active power is the power delivered to the grid, while the total (apparent) power is the sum of active power and reactive power. Reactive power is typically stored in the system due to induction and capacitance. A higher Power Factor thus implies:

- Less power stored in the system, i.e., less reserve power.
- More power through the system, which implies higher currents in the generator, and heating of the generator.

It is of interest to consider the possibility of relaxing on the constraint of the Power Factor. The main advantage of relaxing the Power Factor constraint is that it makes it possible to take out more of the stored (reactive) power in case of operational challenges, and thus may make it possible to avoid shut-down of the generator during short-term problems. This must, however, be balanced against the added strain on the generator due to thermal heating, with possible shortened life-time of the generator.

In a recent Ph.D. study (Øyvang, 2018), the possibility of modelling and controlling the thermal development in a synchronous generator was developed. That study includes a discussion on how to find relevant model parameters for generators in combination with detailed 3D simulations, for the combined thermal model with a simple electric generator model. A reformulation of the thermal model has been studied in a group project in course FM1015 Modelling of Dynamic Systems (Lie, 2018).

It is of interest to further develop the thermal model with the electric model. Specifically, the thermal model should use the approach in Lie (2018). The thermal model should be extended with a simple electric model of the generator. It is of particular interest to study how model parameters can be adjusted to make the model better fit experimental data. Furthermore, it is of interest to study state estimation for the model, and how model fitting (parameter estimation) and state estimation can be carried out with a model implemented in Modelica and/or Julia. *Even without background in state estimation, it is relatively straightforward to understand the basics of state estimation*.

Modelica is a suitable language for implementing models of the resulting system. Julia is a free language which is easy to install, and with syntax similar that of MATLAB and Python. OMJulia is a tool for integrating Julia with OpenModelica. Together, OpenModelica and Julia with OMJulia makes it trivial to operate the model, linearize the model, do state estimation, do model fitting, etc. It is also possible to implement the model in Julia; this may lead to a faster computations.

References:

Øyvang, Thomas (2018). *Enhanced power capability of generator units for increased operational security*. Ph.d.-thesis, University of South-Eastern Norway.

Lie, Bernt (2018). *Group project task, course FM1015 Modelling of Dynamic Systems*. University of South-Eastern Norway.

Task description:

The following tasks are relevant:

1. Give an overview of a thermal model of a synchronous generator, and how it can be extended with an electric model constraining field voltage and terminal currents. Discuss required model parameters in the model, and how to find these.
2. Give an overview of experimental data to test the model. Data from Øyvang (2018) should be used. Implement the model in Modelica, run it from Julia, and test the model against experimental data of Øyvang (2018).
3. Develop a state estimation algorithm in Julia, e.g., an Unscented Kalman Filter (as used in Øyvang, 2018). Compare this to a similar Ensemble Kalman Filter. [Both of these algorithms are relatively simple to understand and use.]
4. Discuss how the model can be tuned to improved model fitting by adjusting model parameters, and test out your algorithms.
5. Report the work in the Master's Thesis, and possibly in a suitable conference/journal paper.

Student category: EPE, IIA, PT, EET (most suitable for EPE and IIA students, but possible for all with some understanding of electric machines).

Practical arrangements:

The candidate will receive information from Skagerak Kraft, but will sit elsewhere. The candidate is expected to meet with the supervisor once per week from January until mid April (up to 1 hour each week), and will work independently after that. The candidate is expected to hand in partial reports every 3 weeks, and is offered feedback on these reports.

Supervision:

As a general rule, the student is entitled to 15-20 hours of supervision. This includes necessary time for the supervisor to prepare for supervision meetings (reading material to be discussed, etc).

Signatures:

Supervisor (date and signature):

Student (write clearly in all capitalized letters):

Student (date and signature):

Appendix B

Model equations for DAE models

Model 1

$$\frac{dU_\mathrm{r}}{dt} = \dot{Q}_\mathrm{r}^{\sigma} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.1}$$

$$\frac{dU_\mathrm{s}}{dt} = \dot{Q}_\mathrm{s}^{\sigma} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.2}$$

$$\frac{dU_\mathrm{Fe}}{dt} = \dot{Q}_\mathrm{Fe}^{\sigma} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.3}$$

$$\frac{dU_\mathrm{a}^{\delta}}{dt} = \dot{H}_\mathrm{a}^{\mathrm{c}} - \dot{H}_\mathrm{a}^{\delta} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}_\mathrm{f}^{\sigma} \approx 0 \tag{B.4}$$

$$\frac{dU_\mathrm{a}^{\mathrm{h}}}{dt} = \dot{H}_\mathrm{a}^{\delta} - \dot{H}_\mathrm{a}^{\mathrm{h}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.5}$$

$$U_\mathrm{r} = H_\mathrm{r} - p_\mathrm{a} V_\mathrm{r} \tag{B.6}$$

$$U_\mathrm{s} = H_\mathrm{s} - p_\mathrm{a} V_\mathrm{s} \tag{B.7}$$

$$U_\mathrm{Fe} = H_\mathrm{Fe} - p_\mathrm{a} V_\mathrm{Fe} \tag{B.8}$$

$$H_\mathrm{r} = m_\mathrm{r} \hat{H}_\mathrm{r} \tag{B.9}$$

$$H_\mathrm{s} = m_\mathrm{s} \hat{H}_\mathrm{s} \tag{B.10}$$

$$H_\mathrm{Fe} = m_\mathrm{Fe} \hat{H}_\mathrm{Fe} \tag{B.11}$$

$$\hat{H}_\mathrm{r} = \hat{H}_\mathrm{Cu}^{o} + \hat{c}_{p,\mathrm{Cu}} \left(T_\mathrm{r} - T_\mathrm{Cu}^{o}\right) \tag{B.12}$$

$$\hat{H}_s = \hat{H}_\mathrm{Cu}^{o} + \hat{c}_{p,\mathrm{Cu}} \left(T_\mathrm{s} - T_\mathrm{Cu}^{o}\right) \tag{B.13}$$

$$\hat{H}_\mathrm{Fe} = \hat{H}_\mathrm{Fe}^{o} + \hat{c}_{p,\mathrm{Fe}} \left(T_{Fe} - T_\mathrm{Fe}^{o}\right) \tag{B.14}$$

$$\hat{H}_\mathrm{a}^{\mathrm{c}} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\mathrm{c}} - T_\mathrm{a}^{o}\right) \tag{B.15}$$

$$\hat{H}_\mathrm{a}^{\delta} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\delta} - T_\mathrm{a}^{o}\right) \tag{B.16}$$

$$\hat{H}_\mathrm{a}^{\mathrm{h}} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\mathrm{h}} - T_a^{o}\right) \tag{B.17}$$

$$\dot{H}_\mathrm{a}^{\mathrm{c}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{c}} \tag{B.18}$$

$$\dot{H}_\mathrm{a}^{\delta} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\delta} \tag{B.19}$$

$$\dot{H}_\mathrm{a}^{\mathrm{h}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{h}}. \tag{B.20}$$

$$\dot{Q}_\mathrm{r}^{\sigma} = 1.1 R_\mathrm{r} I_\mathrm{f}^2 \tag{B.21}$$

$$\dot{Q}_\mathrm{s}^{\sigma} = 3 R_\mathrm{s} I_\mathrm{t}^2 \tag{B.22}$$

$$\dot{Q}_\mathrm{f}^{\sigma} = 0.8 \dot{W}_\mathrm{f} \tag{B.23}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_\mathrm{r} - T_\mathrm{a}^{\delta}\right) \tag{B.24}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_\mathrm{s} - T_\mathrm{Fe}\right) \tag{B.25}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_\mathrm{Fe} - T_\mathrm{a}^{\mathrm{h}}\right) \tag{B.26}$$

$$T_\mathrm{w}^{\mathrm{h}} = \frac{N_\mathrm{St}^{\mathrm{w}} \left(1 - e^{-N_\mathrm{St}^{\Delta}}\right) T_\mathrm{a}^{\mathrm{h}} + N_\mathrm{St}^{\Delta} e^{-N_\mathrm{St}^{\Delta}} T_\mathrm{w}^{\mathrm{h}}}{N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}}} \tag{B.27}$$

$$T_\mathrm{a}^{\mathrm{c}} = \frac{N_\mathrm{St}^{\Delta} T_\mathrm{a}^{\mathrm{h}} + N_\mathrm{St}^{\mathrm{a}} \left(1 - e^{-N_\mathrm{St}^{\Delta}}\right) T_\mathrm{w}^{\mathrm{c}}}{N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}}} \tag{B.28}$$

$$\dot{Q}_{\mathrm{w}2\mathrm{a}} = \frac{e^{-N_\mathrm{St}^{\Delta}} - 1}{\frac{1}{\hat{c}_{p,\mathrm{a}} \dot{m}_\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}} - \frac{1}{\hat{c}_{p,\mathrm{w}} \dot{m}_\mathrm{w}}} \left(T_\mathrm{w}^{\mathrm{c}} - T_\mathrm{a}^{\mathrm{h}}\right) \tag{B.29}$$

$$N_\mathrm{St}^{\mathrm{w}} = \frac{\mathscr{U} A_\mathrm{x}}{\hat{c}_{p,\mathrm{w}} \dot{m}_\mathrm{w}} \tag{B.30}$$

$$N_\mathrm{St}^{\mathrm{a}} = \frac{\mathscr{U} A_x}{\hat{c}_{p,\mathrm{a}} \dot{m}_\mathrm{a}} \tag{B.31}$$

$$N_\mathrm{St}^{\Delta} = N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}}. \tag{B.32}$$

Model 2

$$\frac{dU_\mathrm{r}}{dt} = \dot{Q}_\mathrm{r}^{\sigma} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.33}$$

$$\frac{dU_\mathrm{s}}{dt} = \dot{Q}_\mathrm{s}^{\sigma} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.34}$$

$$\frac{dU_\mathrm{Fe}}{dt} = \dot{Q}_\mathrm{Fe}^{\sigma} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.35}$$

$$\frac{dU_\mathrm{a}^{\delta}}{dt} = \dot{H}_\mathrm{a}^{\mathrm{c}} - \dot{H}_\mathrm{a}^{\delta} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}_\mathrm{f}^{\sigma} \approx 0 \tag{B.36}$$

$$\frac{dU_\mathrm{a}^{\mathrm{h}}}{dt} = \dot{H}_\mathrm{a}^{\delta} - \dot{H}_\mathrm{a}^{\mathrm{h}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.37}$$

$$U_\mathrm{r} = H_\mathrm{r} - p_\mathrm{a} V_\mathrm{r} \tag{B.38}$$

$$U_\mathrm{s} = H_\mathrm{s} - p_\mathrm{a} V_\mathrm{s} \tag{B.39}$$

$$U_\mathrm{Fe} = H_\mathrm{Fe} - p_\mathrm{a} V_\mathrm{Fe} \tag{B.40}$$

$$H_\mathrm{r} = m_\mathrm{r} \hat{H}_\mathrm{r} \tag{B.41}$$

$$H_\mathrm{s} = m_\mathrm{s} \hat{H}_\mathrm{s} \tag{B.42}$$

$$H_\mathrm{Fe} = m_\mathrm{Fe} \hat{H}_\mathrm{Fe} \tag{B.43}$$

$$\hat{H}_\mathrm{r} = \hat{H}_\mathrm{Cu}^{o} + \hat{c}_{p,\mathrm{Cu}} \left(T_\mathrm{r} - T_\mathrm{Cu}^{o}\right) \tag{B.44}$$

$$\hat{H}_s = \hat{H}_\mathrm{Cu}^{o} + \hat{c}_{p,\mathrm{Cu}} \left(T_\mathrm{s} - T_\mathrm{Cu}^{o}\right) \tag{B.45}$$

$$\hat{H}_\mathrm{Fe} = \hat{H}_\mathrm{Fe}^{o} + \hat{c}_{p,\mathrm{Fe}} \left(T_{Fe} - T_\mathrm{Fe}^{o}\right) \tag{B.46}$$

$$\hat{H}_\mathrm{a}^{\mathrm{c}} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\mathrm{c}} - T_\mathrm{a}^{o}\right) \tag{B.47}$$

$$\hat{H}_\mathrm{a}^{\delta} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\delta} - T_\mathrm{a}^{o}\right) \tag{B.48}$$

$$\hat{H}_\mathrm{a}^{\mathrm{h}} = \hat{H}_\mathrm{a}^{o} + \hat{c}_{p,\mathrm{a}} \left(T_\mathrm{a}^{\mathrm{h}} - T_a^{o}\right) \tag{B.49}$$

$$\dot{H}_\mathrm{a}^{\mathrm{c}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{c}} \tag{B.50}$$

$$\dot{H}_\mathrm{a}^{\delta} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\delta} \tag{B.51}$$

$$\dot{H}_\mathrm{a}^{\mathrm{h}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{h}}. \tag{B.52}$$

$$\dot{Q}_\mathrm{r}^{\sigma} = 1.1 R_\mathrm{r} \left(1 + \alpha_{\mathrm{Cu}} \left(T_\mathrm{r} - T_\mathrm{Cu}^{o}\right)\right) I_\mathrm{f}^2 \tag{B.53}$$

$$\dot{Q}_\mathrm{s}^{\sigma} = 3 R_\mathrm{s} \left(1 + \alpha_{\mathrm{Cu}} \left(T_\mathrm{s} - T_\mathrm{Cu}^{o}\right)\right) I_\mathrm{t}^2 \tag{B.54}$$

$$\dot{Q}_\mathrm{f}^{\sigma} = 0.8 \dot{W}_\mathrm{f} \tag{B.55}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_\mathrm{r} - T_\mathrm{a}^{\delta}\right) \tag{B.56}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_\mathrm{s} - T_\mathrm{Fe}\right) \tag{B.57}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_\mathrm{Fe} - T_\mathrm{a}^{\mathrm{h}}\right) \tag{B.58}$$

$$T_\mathrm{w}^{\mathrm{h}} = \frac{N_\mathrm{St}^{\mathrm{w}} \left(1 - e^{-N_\mathrm{St}^{\Delta}}\right) T_\mathrm{a}^{\mathrm{h}} + N_\mathrm{St}^{\Delta} e^{-N_\mathrm{St}^{\Delta}} T_\mathrm{w}^{\mathrm{h}}}{N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}}} \tag{B.59}$$

$$T_\mathrm{a}^{\mathrm{c}} = \frac{N_\mathrm{St}^{\Delta} T_\mathrm{a}^{\mathrm{h}} + N_\mathrm{St}^{\mathrm{a}} \left(1 - e^{-N_\mathrm{St}^{\Delta}}\right) T_\mathrm{w}^{\mathrm{c}}}{N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}}} \tag{B.60}$$

$$\dot{Q}_{\mathrm{w}2\mathrm{a}} = \frac{e^{-N_\mathrm{St}^{\Delta}} - 1}{\frac{1}{\hat{c}_{p,\mathrm{a}} \dot{m}_\mathrm{a}} e^{-N_\mathrm{St}^{\Delta}} - \frac{1}{\hat{c}_{p,\mathrm{w}} \dot{m}_\mathrm{w}}} \left(T_\mathrm{w}^{\mathrm{c}} - T_\mathrm{a}^{\mathrm{h}}\right) \tag{B.61}$$

$$N_\mathrm{St}^{\mathrm{w}} = \frac{\mathscr{U} A_\mathrm{x}}{\hat{c}_{p,\mathrm{w}} \dot{m}_\mathrm{w}} \tag{B.62}$$

$$N_\mathrm{St}^{\mathrm{a}} = \frac{\mathscr{U} A_x}{\hat{c}_{p,\mathrm{a}} \dot{m}_\mathrm{a}} \tag{B.63}$$

$$N_\mathrm{St}^{\Delta} = N_\mathrm{St}^{\mathrm{w}} - N_\mathrm{St}^{\mathrm{a}}. \tag{B.64}$$

Model 3a

$$\frac{dU_{\mathrm{r}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{r}} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.65}$$

$$\frac{dU_{\mathrm{s}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{s}} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.66}$$

$$\frac{dU_{\mathrm{Fe}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{Fe}} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.67}$$

$$\frac{dU^{\delta}_{\mathrm{a}}}{dt} = \dot{H}^{\mathrm{c}}_{\mathrm{a}} - \dot{H}^{\delta}_{\mathrm{a}} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}^{\sigma}_{\mathrm{f}} \approx 0 \tag{B.68}$$

$$\frac{dU^{\mathrm{h}}_{\mathrm{a}}}{dt} = \dot{H}^{\delta}_{\mathrm{a}} - \dot{H}^{\mathrm{h}}_{\mathrm{a}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.69}$$

$$U_{\mathrm{r}} = H_{\mathrm{r}} - p_{\mathrm{a}} V_{\mathrm{r}} \tag{B.70}$$

$$U_{\mathrm{s}} = H_{\mathrm{s}} - p_{\mathrm{a}} V_{\mathrm{s}} \tag{B.71}$$

$$U_{\mathrm{Fe}} = H_{\mathrm{Fe}} - p_{\mathrm{a}} V_{\mathrm{Fe}} \tag{B.72}$$

$$H_{\mathrm{r}} = m_{\mathrm{r}} \hat{H}_{\mathrm{r}} \tag{B.73}$$

$$H_{\mathrm{s}} = m_{\mathrm{s}} \hat{H}_{\mathrm{s}} \tag{B.74}$$

$$H_{\mathrm{Fe}} = m_{\mathrm{Fe}} \hat{H}_{\mathrm{Fe}} \tag{B.75}$$

$$\hat{H}_{\mathrm{r}} = \hat{H}^{o}_{\mathrm{Cu}} + \int_{T^{o}_{\mathrm{Cu}}}^{T_{\mathrm{r}}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.76}$$

$$\hat{H}_{\mathrm{s}} = \hat{H}^{o}_{\mathrm{Cu}} + \int_{T^{o}_{\mathrm{Cu}}}^{T_{\mathrm{s}}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.77}$$

$$\hat{H}_{\mathrm{Fe}} = \hat{H}^{o}_{\mathrm{Fe}} + \int_{T^{o}_{Fe}}^{T_{Fe}} \hat{c}_{p,\mathrm{Fe}}(T)dT \tag{B.78}$$

$$\hat{H}^{\mathrm{c}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\mathrm{c}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.79}$$

$$\hat{H}^{\delta}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\delta}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.80}$$

$$\hat{H}^{\mathrm{h}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\mathrm{h}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.81}$$

$$\dot{H}^{\mathrm{c}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\mathrm{c}}_{\mathrm{a}} \tag{B.82}$$

$$\dot{H}^{\delta}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\delta}_{\mathrm{a}} \tag{B.83}$$

$$\dot{H}^{\mathrm{h}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\mathrm{h}}_{\mathrm{a}} \tag{B.84}$$

$$\dot{Q}^{\sigma}_{\mathrm{r}} = 1.1 R_{\mathrm{r}} I^{2}_{\mathrm{f}} \tag{B.85}$$

$$\dot{Q}^{\sigma}_{\mathrm{s}} = 3 R_{\mathrm{s}} I^{2}_{\mathrm{t}} \tag{B.86}$$

$$\dot{Q}^{\sigma}_{\mathrm{f}} = 0.8 \dot{W}_{\mathrm{f}} \tag{B.87}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_{\mathrm{r}} - T^{\delta}_{\mathrm{a}} \right) \tag{B.88}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_{\mathrm{s}} - T_{\mathrm{Fe}} \right) \tag{B.89}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_{\mathrm{Fe}} - T^{\mathrm{h}}_{\mathrm{a}} \right) \tag{B.90}$$

$$T^{\mathrm{h}}_{\mathrm{w}} = \frac{N^{\mathrm{w}}_{\mathrm{St}} \left(1 - e^{-N^{\Delta}_{\mathrm{St}}} \right) T^{\mathrm{h}}_{\mathrm{a}} + N^{\Delta}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}} T^{\mathrm{h}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}}} \tag{B.91}$$

$$T^{\mathrm{c}}_{\mathrm{a}} = \frac{N^{\Delta}_{\mathrm{St}} T^{\mathrm{h}}_{\mathrm{a}} + N^{\mathrm{a}}_{\mathrm{St}} \left(1 - e^{-N^{\Delta}_{\mathrm{St}}} \right) T^{\mathrm{c}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}}} \tag{B.92}$$

$$\dot{Q}_{\mathrm{w}2\mathrm{a}} = \frac{e^{-N^{\Delta}_{\mathrm{St}}} - 1}{\frac{1}{\hat{c}_{p,\mathrm{a}} \dot{m}_{\mathrm{a}}} e^{-N^{\Delta}_{\mathrm{St}}} - \frac{1}{\hat{c}_{p,\mathrm{w}} \dot{m}_{\mathrm{w}}}} \left(T^{\mathrm{c}}_{\mathrm{w}} - T^{\mathrm{h}}_{\mathrm{a}} \right) \tag{B.93}$$

$$N^{\mathrm{w}}_{\mathrm{St}} = \frac{\mathscr{U} A_{\mathrm{x}}}{\hat{c}_{p,\mathrm{w}} \dot{m}_{\mathrm{w}}} \tag{B.94}$$

$$N^{\mathrm{a}}_{\mathrm{St}} = \frac{\mathscr{U} A_{x}}{\hat{c}_{p,\mathrm{a}} \dot{m}_{\mathrm{a}}} \tag{B.95}$$

$$N^{\Delta}_{\mathrm{St}} = N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}}. \tag{B.96}$$

Model 4a

$$\frac{dU_{\mathrm{r}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{r}} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.97}$$

$$\frac{dU_{\mathrm{s}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{s}} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.98}$$

$$\frac{dU_{\mathrm{Fe}}}{dt} = \dot{Q}^{\sigma}_{\mathrm{Fe}} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.99}$$

$$\frac{dU^{\delta}_{\mathrm{a}}}{dt} = \dot{H}^{\mathrm{c}}_{\mathrm{a}} - \dot{H}^{\delta}_{\mathrm{a}} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}^{\sigma}_{\mathrm{f}} \approx 0 \tag{B.100}$$

$$\frac{dU^{\mathrm{h}}_{\mathrm{a}}}{dt} = \dot{H}^{\delta}_{\mathrm{a}} - \dot{H}^{\mathrm{h}}_{\mathrm{a}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.101}$$

$$U_{\mathrm{r}} = H_{\mathrm{r}} - p_{\mathrm{a}} V_{\mathrm{r}} \tag{B.102}$$

$$U_{\mathrm{s}} = H_{\mathrm{s}} - p_{\mathrm{a}} V_{\mathrm{s}} \tag{B.103}$$

$$U_{\mathrm{Fe}} = H_{\mathrm{Fe}} - p_{\mathrm{a}} V_{\mathrm{Fe}} \tag{B.104}$$

$$H_{\mathrm{r}} = m_{\mathrm{r}} \hat{H}_{\mathrm{r}} \tag{B.105}$$

$$H_{\mathrm{s}} = m_{\mathrm{s}} \hat{H}_{\mathrm{s}} \tag{B.106}$$

$$H_{\mathrm{Fe}} = m_{\mathrm{Fe}} \hat{H}_{\mathrm{Fe}} \tag{B.107}$$

$$\hat{H}_{\mathrm{r}} = \hat{H}^{o}_{\mathrm{Cu}} + \int_{T^{o}_{\mathrm{Cu}}}^{T_{\mathrm{r}}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.108}$$

$$\hat{H}_{\mathrm{s}} = \hat{H}^{o}_{\mathrm{Cu}} + \int_{T^{o}_{\mathrm{Cu}}}^{T_{\mathrm{s}}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.109}$$

$$\hat{H}_{\mathrm{Fe}} = \hat{H}^{o}_{\mathrm{Fe}} + \int_{T^{o}_{Fe}}^{T_{Fe}} \hat{c}_{p,\mathrm{Fe}}(T)dT \tag{B.110}$$

$$\hat{H}^{\mathrm{c}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\mathrm{c}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.111}$$

$$\hat{H}^{\delta}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\delta}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.112}$$

$$\hat{H}^{\mathrm{h}}_{\mathrm{a}} = \hat{H}^{o}_{\mathrm{a}} + \int_{T^{o}_{\mathrm{a}}}^{T^{\mathrm{h}}_{\mathrm{a}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.113}$$

$$\dot{H}^{\mathrm{c}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\mathrm{c}}_{\mathrm{a}} \tag{B.114}$$

$$\dot{H}^{\delta}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\delta}_{\mathrm{a}} \tag{B.115}$$

$$\dot{H}^{\mathrm{h}}_{\mathrm{a}} = \dot{m}_{\mathrm{a}} \hat{H}^{\mathrm{h}}_{\mathrm{a}}. \tag{B.116}$$

$$\dot{Q}^{\sigma}_{\mathrm{r}} = 1.1 R_{\mathrm{r}} \left(1 + \alpha_{\mathrm{Cu}} \left(T_{\mathrm{r}} - T^{o}_{\mathrm{Cu}} \right) \right) I^{2}_{\mathrm{f}} \tag{B.117}$$

$$\dot{Q}^{\sigma}_{\mathrm{s}} = 3 R_{\mathrm{s}} \left(1 + \alpha_{\mathrm{Cu}} \left(T_{\mathrm{s}} - T^{o}_{\mathrm{Cu}} \right) \right) I^{2}_{\mathrm{t}} \tag{B.118}$$

$$\dot{Q}^{\sigma}_{\mathrm{f}} = 0.8 \dot{W}_{\mathrm{f}} \tag{B.119}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_{\mathrm{r}} - T^{\delta}_{\mathrm{a}} \right) \tag{B.120}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_{\mathrm{s}} - T_{\mathrm{Fe}} \right) \tag{B.121}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_{\mathrm{Fe}} - T^{\mathrm{h}}_{\mathrm{a}} \right) \tag{B.122}$$

$$T^{\mathrm{h}}_{\mathrm{w}} = \frac{N^{\mathrm{w}}_{\mathrm{St}} \left(1 - e^{-N^{\Delta}_{\mathrm{St}}} \right) T^{\mathrm{h}}_{\mathrm{a}} + N^{\Delta}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}} T^{\mathrm{h}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}}} \tag{B.123}$$

$$T^{\mathrm{c}}_{\mathrm{a}} = \frac{N^{\Delta}_{\mathrm{St}} T^{\mathrm{h}}_{\mathrm{a}} + N^{\mathrm{a}}_{\mathrm{St}} \left(1 - e^{-N^{\Delta}_{\mathrm{St}}} \right) T^{\mathrm{c}}_{\mathrm{w}}}{N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}} e^{-N^{\Delta}_{\mathrm{St}}}} \tag{B.124}$$

$$\dot{Q}_{\mathrm{w}2\mathrm{a}} = \frac{e^{-N^{\Delta}_{\mathrm{St}}} - 1}{\frac{1}{\hat{c}_{p,\mathrm{a}} \dot{m}_{\mathrm{a}}} e^{-N^{\Delta}_{\mathrm{St}}} - \frac{1}{\hat{c}_{p,\mathrm{w}} \dot{m}_{\mathrm{w}}}} \left(T^{\mathrm{c}}_{\mathrm{w}} - T^{\mathrm{h}}_{\mathrm{a}} \right) \tag{B.125}$$

$$N^{\mathrm{w}}_{\mathrm{St}} = \frac{\mathscr{U} A_{\mathrm{x}}}{\hat{c}_{p,\mathrm{w}} \dot{m}_{\mathrm{w}}} \tag{B.126}$$

$$N^{\mathrm{a}}_{\mathrm{St}} = \frac{\mathscr{U} A_{x}}{\hat{c}_{p,\mathrm{a}} \dot{m}_{\mathrm{a}}} \tag{B.127}$$

$$N^{\Delta}_{\mathrm{St}} = N^{\mathrm{w}}_{\mathrm{St}} - N^{\mathrm{a}}_{\mathrm{St}}. \tag{B.128}$$

Model 3b

$$\frac{dU_\mathrm{r}}{dt} = \dot{Q}_\mathrm{r}^{\sigma} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.129}$$

$$\frac{dU_\mathrm{s}}{dt} = \dot{Q}_\mathrm{s}^{\sigma} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.130}$$

$$\frac{dU_\mathrm{Fe}}{dt} = \dot{Q}_\mathrm{Fe}^{\sigma} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.131}$$

$$\frac{dU_\mathrm{a}^{\delta}}{dt} = \dot{H}_\mathrm{a}^{\mathrm{c}} - \dot{H}_\mathrm{a}^{\delta} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}_\mathrm{f}^{\sigma} \approx 0 \tag{B.132}$$

$$\frac{dU_\mathrm{a}^{\mathrm{h}}}{dt} = \dot{H}_\mathrm{a}^{\delta} - \dot{H}_\mathrm{a}^{\mathrm{h}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.133}$$

$$U_\mathrm{r} = H_\mathrm{r} - p_\mathrm{a} V_\mathrm{r} \tag{B.134}$$

$$U_\mathrm{s} = H_\mathrm{s} - p_\mathrm{a} V_\mathrm{s} \tag{B.135}$$

$$U_\mathrm{Fe} = H_\mathrm{Fe} - p_\mathrm{a} V_\mathrm{Fe} \tag{B.136}$$

$$H_\mathrm{r} = m_\mathrm{r} \hat{H}_\mathrm{r} \tag{B.137}$$

$$H_\mathrm{s} = m_\mathrm{s} \hat{H}_\mathrm{s} \tag{B.138}$$

$$H_\mathrm{Fe} = m_\mathrm{Fe} \hat{H}_\mathrm{Fe} \tag{B.139}$$

$$\hat{H}_\mathrm{r} = \hat{H}_\mathrm{Cu}^{o} + \int_{T_\mathrm{Cu}^{o}}^{T_\mathrm{r}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.140}$$

$$\hat{H}_\mathrm{s} = \hat{H}_\mathrm{Cu}^{o} + \int_{T_\mathrm{Cu}^{o}}^{T_\mathrm{s}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.141}$$

$$\hat{H}_\mathrm{Fe} = \hat{H}_\mathrm{Fe}^{o} + \int_{T_{Fe}^{o}}^{T_{Fe}} \hat{c}_{p,\mathrm{Fe}}(T)dT \tag{B.142}$$

$$\hat{H}_\mathrm{a}^{\mathrm{c}} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\mathrm{c}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.143}$$

$$\hat{H}_\mathrm{a}^{\delta} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\delta}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.144}$$

$$\hat{H}_\mathrm{a}^{\mathrm{h}} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\mathrm{h}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.145}$$

$$\dot{H}_\mathrm{a}^{\mathrm{c}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{c}} \tag{B.146}$$

$$\dot{H}_\mathrm{a}^{\delta} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\delta} \tag{B.147}$$

$$\dot{H}_\mathrm{a}^{\mathrm{h}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{h}} \tag{B.148}$$

$$\dot{Q}_\mathrm{r}^{\sigma} = 1.1 R_\mathrm{r} I_\mathrm{f}^2 \tag{B.149}$$

$$\dot{Q}_\mathrm{s}^{\sigma} = 3 R_\mathrm{s} I_\mathrm{t}^2 \tag{B.150}$$

$$\dot{Q}_\mathrm{f}^{\sigma} = 0.8 \dot{W}_\mathrm{f} \tag{B.151}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_\mathrm{r} - T_\mathrm{a}^{\delta} \right) \tag{B.152}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_\mathrm{s} - T_\mathrm{Fe} \right) \tag{B.153}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_\mathrm{Fe} - T_\mathrm{a}^{\mathrm{h}} \right) \tag{B.154}$$

$$\frac{dT_\mathrm{w}}{dx} = \frac{\mathscr{U} \wp}{\frac{\mathscr{R}}{M_\mathrm{w}} \left(a_\mathrm{w} + b_\mathrm{w} T_\mathrm{w} \right) \dot{m}_\mathrm{w}} \left(T_\mathrm{w} - T_\mathrm{a} \right) \tag{B.155}$$

$$\frac{dT_\mathrm{a}}{dx} = \frac{\mathscr{U} \wp}{\frac{\mathscr{R}}{M_\mathrm{a}} \left(a_\mathrm{a} + b_\mathrm{a} T_\mathrm{a} \right) \dot{m}_\mathrm{a}} \left(T_\mathrm{w} - T_\mathrm{a} \right) \tag{B.156}$$

$$T_\mathrm{w}(x = L_\mathrm{x}) = T_\mathrm{w}^{\mathrm{c}} \tag{B.157}$$

$$T_\mathrm{a}(x = 0) = T_\mathrm{a}^{\mathrm{h}}. \tag{B.158}$$

Model 4b

$$\frac{dU_\mathrm{r}}{dt} = \dot{Q}_\mathrm{r}^{\sigma} - \dot{Q}_{\mathrm{r}2\delta} \tag{B.159}$$

$$\frac{dU_\mathrm{s}}{dt} = \dot{Q}_\mathrm{s}^{\sigma} - \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.160}$$

$$\frac{dU_\mathrm{Fe}}{dt} = \dot{Q}_\mathrm{Fe}^{\sigma} - \dot{Q}_{\mathrm{Fe}2\mathrm{a}} + \dot{Q}_{\mathrm{s}2\mathrm{Fe}} \tag{B.161}$$

$$\frac{dU_\mathrm{a}^{\delta}}{dt} = \dot{H}_\mathrm{a}^{\mathrm{c}} - \dot{H}_\mathrm{a}^{\delta} + \dot{Q}_{\mathrm{r}2\delta} + \dot{Q}_\mathrm{f}^{\sigma} \approx 0 \tag{B.162}$$

$$\frac{dU_\mathrm{a}^{\mathrm{h}}}{dt} = \dot{H}_\mathrm{a}^{\delta} - \dot{H}_\mathrm{a}^{\mathrm{h}} + \dot{Q}_{\mathrm{Fe}2\mathrm{a}} \approx 0. \tag{B.163}$$

$$U_\mathrm{r} = H_\mathrm{r} - p_\mathrm{a} V_\mathrm{r} \tag{B.164}$$

$$U_\mathrm{s} = H_\mathrm{s} - p_\mathrm{a} V_\mathrm{s} \tag{B.165}$$

$$U_\mathrm{Fe} = H_\mathrm{Fe} - p_\mathrm{a} V_\mathrm{Fe} \tag{B.166}$$

$$H_\mathrm{r} = m_\mathrm{r} \hat{H}_\mathrm{r} \tag{B.167}$$

$$H_\mathrm{s} = m_\mathrm{s} \hat{H}_\mathrm{s} \tag{B.168}$$

$$H_\mathrm{Fe} = m_\mathrm{Fe} \hat{H}_\mathrm{Fe} \tag{B.169}$$

$$\hat{H}_\mathrm{r} = \hat{H}_\mathrm{Cu}^{o} + \int_{T_\mathrm{Cu}^{o}}^{T_\mathrm{r}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.170}$$

$$\hat{H}_\mathrm{s} = \hat{H}_\mathrm{Cu}^{o} + \int_{T_\mathrm{Cu}^{o}}^{T_\mathrm{s}} \hat{c}_{p,\mathrm{Cu}}(T)dT \tag{B.171}$$

$$\hat{H}_\mathrm{Fe} = \hat{H}_\mathrm{Fe}^{o} + \int_{T_{Fe}^{o}}^{T_{Fe}} \hat{c}_{p,\mathrm{Fe}}(T)dT \tag{B.172}$$

$$\hat{H}_\mathrm{a}^{\mathrm{c}} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\mathrm{c}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.173}$$

$$\hat{H}_\mathrm{a}^{\delta} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\delta}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.174}$$

$$\hat{H}_\mathrm{a}^{\mathrm{h}} = \hat{H}_\mathrm{a}^{o} + \int_{T_\mathrm{a}^{o}}^{T_\mathrm{a}^{\mathrm{h}}} \hat{c}_{p,\mathrm{a}}(T)dT \tag{B.175}$$

$$\dot{H}_\mathrm{a}^{\mathrm{c}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{c}} \tag{B.176}$$

$$\dot{H}_\mathrm{a}^{\delta} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\delta} \tag{B.177}$$

$$\dot{H}_\mathrm{a}^{\mathrm{h}} = \dot{m}_\mathrm{a} \hat{H}_\mathrm{a}^{\mathrm{h}}. \tag{B.178}$$

$$\dot{Q}_\mathrm{r}^{\sigma} = 1.1 R_\mathrm{r} \left(1 + \alpha_\mathrm{Cu} \left(T_\mathrm{r} - T_\mathrm{Cu}^{o} \right) \right) I_\mathrm{f}^2 \tag{B.179}$$

$$\dot{Q}_\mathrm{s}^{\sigma} = 3 R_\mathrm{s} \left(1 + \alpha_\mathrm{Cu} \left(T_\mathrm{s} - T_\mathrm{Cu}^{o} \right) \right) I_\mathrm{t}^2 \tag{B.180}$$

$$\dot{Q}_\mathrm{f}^{\sigma} = 0.8 \dot{W}_\mathrm{f} \tag{B.181}$$

$$\dot{Q}_{\mathrm{r}2\delta} = U A_{\mathrm{r}2\delta} \left(T_\mathrm{r} - T_\mathrm{a}^{\delta} \right) \tag{B.182}$$

$$\dot{Q}_{\mathrm{s}2\mathrm{Fe}} = \mathscr{U} A_{\mathrm{s}2\mathrm{Fe}} \left(T_\mathrm{s} - T_\mathrm{Fe} \right) \tag{B.183}$$

$$\dot{Q}_{\mathrm{Fe}2\mathrm{a}} = \mathscr{U} A_{\mathrm{Fe}2\mathrm{a}} \left(T_\mathrm{Fe} - T_\mathrm{a}^{\mathrm{h}} \right) \tag{B.184}$$

$$\frac{dT_\mathrm{w}}{dx} = \frac{\mathscr{U} \wp}{\frac{\mathscr{R}}{M_\mathrm{w}} \left(a_\mathrm{w} + b_\mathrm{w} T_\mathrm{w} \right) \dot{m}_\mathrm{w}} \left(T_\mathrm{w} - T_\mathrm{a} \right) \tag{B.185}$$

$$\frac{dT_\mathrm{a}}{dx} = \frac{\mathscr{U} \wp}{\frac{\mathscr{R}}{M_\mathrm{a}} \left(a_\mathrm{a} + b_\mathrm{a} T_\mathrm{a} \right) \dot{m}_\mathrm{a}} \left(T_\mathrm{w} - T_\mathrm{a} \right) \tag{B.186}$$

$$T_\mathrm{w}(x = L_\mathrm{x}) = T_\mathrm{w}^{\mathrm{c}} \tag{B.187}$$

$$T_\mathrm{a}(x = 0) = T_\mathrm{a}^{\mathrm{h}}. \tag{B.188}$$

The integral terms in Model 3 and Model 4 should be replaced as,

$$\int_{T_{\mathrm{Cu}}^{o}}^{T_{\mathrm{r}}} \hat{c}_{p,\mathrm{Cu}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(\left(a_{\mathrm{Cu}}T_{\mathrm{r}} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{r}}^{2}\right) - \left(a_{\mathrm{Cu}}T_{\mathrm{Cu}}^{o} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{Cu}}^{o2}\right)\right)$$

$$\int_{T_{\mathrm{Cu}}^{o}}^{T_{\mathrm{s}}} \hat{c}_{p,\mathrm{Cu}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Cu}}}\left(\left(a_{\mathrm{Cu}}T_{\mathrm{s}} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{s}}^{2}\right) - \left(a_{\mathrm{Cu}}T_{\mathrm{Cu}}^{o} + \frac{b_{\mathrm{Cu}}}{2}T_{\mathrm{Cu}}^{o2}\right)\right)$$

$$\int_{T_{\mathrm{Fe}}^{o}}^{T_{\mathrm{Fe}}} \hat{c}_{p,\mathrm{Fe}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{Fe}}}\left(\left(a_{\mathrm{Fe}}T_{\mathrm{Fe}} + \frac{b_{\mathrm{Fe}}}{2}T_{\mathrm{Fe}}^{2}\right) - \left(a_{\mathrm{Fe}}T_{\mathrm{Fe}}^{o} + \frac{b_{\mathrm{Fe}}}{2}T_{\mathrm{Fe}}^{o2}\right)\right)$$

$$\int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\mathrm{c}}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T_{\mathrm{a}}^{\mathrm{c}} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{\mathrm{c}2}\right) - \left(a_{\mathrm{a}}T_{\mathrm{a}}^{o} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{o2}\right)\right)$$

$$\int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\delta}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T_{\mathrm{a}}^{\delta} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{\delta 2}\right) - \left(a_{\mathrm{a}}T_{\mathrm{a}}^{o} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{o2}\right)\right)$$

$$\int_{T_{\mathrm{a}}^{o}}^{T_{\mathrm{a}}^{\mathrm{h}}} \hat{c}_{p,\mathrm{a}}(T)dT = \frac{\mathscr{R}}{M_{\mathrm{a}}}\left(\left(a_{\mathrm{a}}T_{\mathrm{a}}^{\mathrm{h}} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{\mathrm{h}2}\right) - \left(a_{\mathrm{a}}T_{\mathrm{a}}^{o} + \frac{b_{\mathrm{a}}}{2}T_{\mathrm{a}}^{o2}\right)\right).$$

Appendix C

Code listing

All the code listing written in Jupyter notebook can be found at

`https://github.com/pandeysudan27/MasterThesis2019`.

The link cosists of notebooks named as,

1. `ComparisionOfKalmanFilter.ipynb`
2. `ComputingTimeForDAEModelsOMJulia.ipynb`
3. `DAEModelsSimulationJuliaHeatExchangerProfiles.ipynb`
4. `KalmanFilterModel1.ipynb`
5. `KalmanFilterModel2.ipynb`
6. `KalmanFilterModel3a.ipynb`
7. `KalmanFilterModel3b.ipynb`
8. `KalmanFilterModel4a.ipynb`
9. `KalmanFilterModel4b.ipynb`
10. `LewisCoefficientsLinearAndQuadApproximation.ipynb`
11. `LinearizationStabilityControllabilityObservabilityModel1and2.ipynb`
12. `OverViewOfExperimentalData.ipynb`
13. `ParameterOptimization.ipynb`
14. `ParametersSensitivityAnalysis.ipynb`
15. `SimulatorVersusRealMeasurements.ipynb`

It also contains Modelica code for Model 1, 2, 3a, and 4b with file name as `Model1.mo`, `Model2.mo`, `Model3a.mo`, and `Model4a.mo`.

Appendix D

Submitted draft paper for SIMS 2019

State Estimation of a Thermal Model of Air-cooled Synchronous Generator

Madhusudhan Pandey, Thomas Øyvang, Bernt Lie

University of South-Eastern Norway, Porsgrunn, Norway, Bernt.Lie@usn.no

Abstract

In this paper, we extend a previous study on a totally enclosed thermal model of a synchronous generator, with temperature state estimation using experimental data. The extension includes a new formulation of the system model, with 4 different model variations with and without temperature dependence in the metal, air, and water heat capacities and the copper resistances, where temperature variation in water and/or air requires a non-standard heat exchanger model. In the former study, the Unscented Kalman Filter (UKF) was used for state estimation. Here, we include both the UKF as well as the Ensemble Kalman Filter (EnKF) in the comparison. UKF and EnKF are compared based on estimation accuracy and computational speed. Results show that EnKF exhibits lower RMSE for the innovation process and thus is more accurate than the UKF even with a "minimum" of 50 particles, but the UKF with 6 sigma points (3 states) is faster. It is too early to conclude which of 4 models is more accurate, as they need to be tuned individually wrt. parameter fitting. *Keywords*: Air-cooled synchronous generator, dynamic model, state estimation, Unscented Kalman filter, Ensemble Kalman filter

1 Introduction

1.1 Background

Due to the increase in intermittent renewable energy resources, hydropower plants will become a key component to provide higher operational flexibility in the future power system. In European hydropower generation, the synchronous generator power factor is restricted to the range $[0.85, 0.95]$, (ENTSO-E, 2016); for Norway, the power factor should be less than 0.86, (Statnett, 2012).

The power factor is the ratio of active power to apparent (complex) power. A small power factor implies a reduced active power production compared to a higher power factor. High production of active power is desired by the plant owners, but an increased power factor may cause problems due to the thermal design limitation of the machine. An important question is: would it be acceptable to relax on the constraint on the power factor for a limited time period in order to take out unexploited power in critical situations? To allow for such a relaxation in the power factor, it is important to have a measure of the temperature evolution, and how this influences the lifetime of the generator.

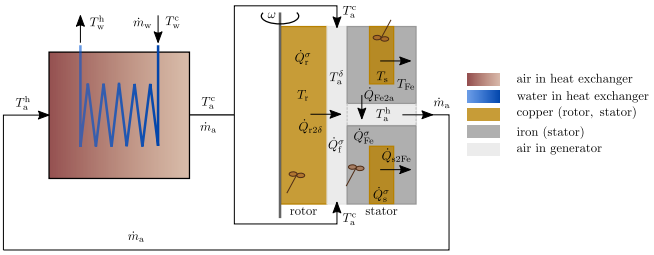


Figure 1. Thermal model of air-cooled synchronous generator, from (Lie, 2018).

In this paper, we consider how to obtain information about the temperature evolution.

A thermal model of a totally enclosed air-cooled hydro generator was developed in (Øyvang, 2018), using a closed-loop, water cooled heat exchanger for cooling heated air from the outlet of generator, and applied to a case study of a vertically mounted 103 MVA air-cooled hydro generator at Åbjøra, Norway. A similar model with more general structure and more efficient heat exchanger description was developed in (Lie, 2018).

It is of interest to extend the description in (Lie, 2018) with temperature dependent heat capacities (metals, air) and temperature dependent copper resistances. Furthermore, it is of interest to carry out a more extensive study on state estimation compared to (Øyvang, 2018), using several variations of the Unscented Kalman Filter (UKF) as well as introducing the Ensemble Kalman Filter (EnKF).

1.2 Organization of paper

The paper is organized as follows. The mathematical model is presented in Section 2. State estimation algorithms UKF and EnKF are presented in Section 3. Results are presented and discussed in Section 4. Finally, conclusions are drawn in Section 5, together with possible future work.

2 Mathematical model

Figure 1 shows the thermal operation of an air-cooled synchronous generator.

The cold air out of the heat exchanger is blown by a fan into the rotor/stator air gap. The air is heated by heat flow from rotor, air gap windage, and bearing friction. Next, air is forced into ducts through the stator iron core where it gets heated by heat flow from the iron. At the outlet

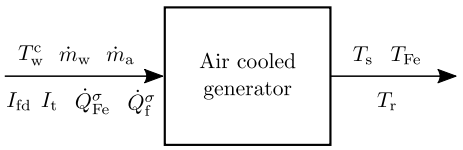


Figure 2. Functional diagram for air-cooled synchronous generator, from (Lie, 2018).

from the stator ducts, the heated air is collected and passed through a counter-current heat exchanger. The heated air is cooled down through the heat exchanger using continuous cold water circulation, before it is re-injected into the air gap in a continuous, closed loop process.

The water mass flow rate through the heat exchanger is $\dot{m}_w$, and it enters at temperature T_w^c and leaves the heat exchanger at temperature T_w^h. The air mass flow rate is $\dot{m}_a$ with temperature T_a^h at stator outlet and heat exchanger entry; through the heat exchanger, the air is cooled down to temperature T_a^c. The metal volumes are assumed to be homogeneous in temperature, with rotor copper at temperature T_r, stator copper at temperature T_s, and stator iron at temperature T_{Fe}. Rotor copper is heated by heat rate $\dot{Q}_r^{\sigma}$ due to resistive electric loss from the field current I_f. Similarly, the stator copper is heated by heat rate $\dot{Q}_s^{\sigma}$ due to stator terminal current I_t. The stator iron is heated by heat rate $\dot{Q}_{Fe}^{\sigma}$ due to eddy current losses and hysteresis losses, (Hargreaves et al., 2011). The air gap between rotor and stator is heated at heat rate $\dot{Q}_f^{\sigma}$ due to bearing and windage losses, (Øyvang, 2018). In addition, heat conduction/convection between the various volumes take place. It is of interest to consider how the inputs $\dot{m}_w$, $\dot{m}_a$, T_w^c, $\dot{Q}_{Fe}^{\sigma}$, $\dot{Q}_f^{\sigma}$, I_t and I_f influence the temperatures in the generator metals, T_r, T_s, and T_{Fe}. A functional diagram for the air-cooled synchronous generator is shown in Figure 2 relating inputs and outputs.

The mathematical model governing generator metal temperatures is taken from (Lie, 2018),

$$m_r \hat{c}_{p,Cu} \frac{dT_r}{dt} = 1.1 R_r I_f^2 - \mathscr{U} A_{r2\delta} \left(T_r - T_a^{\delta} \right) \quad (1)$$

$$m_s \hat{c}_{p,Cu} \frac{dT_s}{dt} = 3 R_s I_t^2 - \mathscr{U} A_{s2Fe} \left(T_s - T_{Fe} \right) \quad (2)$$

$$m_{Fe} \hat{c}_{p,Fe} \frac{dT_{Fe}}{dt} = \mathscr{U} A_{s2Fe} \left(T_s - T_{Fe} \right) - \mathscr{U} A_{Fe2a} \left(T_{Fe} - T_a^h \right) + \dot{Q}_{Fe}^{\sigma}. \quad (3)$$

Here, m_r, m_s, and m_{Fe} are the masses of the respective metal volumes. $\hat{c}_{p,Cu}$ and $\hat{c}_{p,Fe}$ are specific heat capacities of copper and iron, respectively. R_r and R_s are resistances of copper in the rotor and stator, respectively, $\mathscr{U} A_{r2\delta}$, $\mathscr{U} A_{s2Fe}$, and $\mathscr{U} A_{Fe2a}$ are heat transfer factors between rotor metal and rotor-stator air-gap, stator copper and stator iron, and stator iron and stator duct air gaps, respectively. T_a^{δ} and T_a^h are air temperatures in the rotor-stator air-gap and in the stator duct, respectively.

Similarly, for air inside the generator,

$$0 = \dot{m}_a \hat{c}_{p,a} \left(T_a^c - T_a^{\delta} \right) + \mathscr{U} A_{r2\delta} \left(T_r - T_a^{\delta} \right) + \dot{Q}_f^{\sigma} \quad (4)$$

$$0 = \dot{m}_a \hat{c}_{p,a} \left(T_a^{\delta} - T_a^h \right) + \mathscr{U} A_{Fe2a} \left(T_{Fe} - T_a^h \right). \quad (5)$$

Here, $\hat{c}_{p,a}$ is the specific heat capacity of air.

For the heat exchanger, we introduce *Stanton numbers* N_{St}^w and N_{St}^a,

$$N_{St}^w = \frac{\mathscr{U} A_x}{\hat{c}_{p,w} \dot{m}_w}$$
$$N_{St}^a = \frac{\mathscr{U} A_x}{\hat{c}_{p,a} \dot{m}_a}$$
$$N_{St}^{\Delta} = N_{St}^w - N_{St}^a.$$

Here, $\hat{c}_{p,w}$ is the specific heat capacity of water, and $\mathscr{U} A_x$ is the heat transfer factor between water and air in the heat exchanger. Provided that the Stanton numbers are constant and independent of (i) position, and (ii) temperatures, the counter-current heat exchanger model is

$$\left(N_{St}^w - N_{St}^a \exp\left(-N_{St}^{\Delta} \right) \right) T_a^c = N_{St}^{\Delta} T_a^h + N_{St}^a \left(1 - \exp\left(-N_{St}^{\Delta} \right) \right) T_w^c. \quad (6)$$

The heat exchanger model in Eq. 6 is the result of analytically solving a linear two point boundary value problem.

This model can be extended in several directions, by (a) introducing temperature dependence in the specific heat capacities $\hat{c}_{p,j}$, (b) introducing temperature dependence in the copper resistances R_r and R_s, and (c) in principle also in the heat transfer factors $\mathscr{U} A_j$. The only substantial change in the model is that if any of the Stanton numbers become temperature dependent, this will invalidate Eq. 6, and the involved two point boundary value problem must be solved numerically instead of analytically. Here, we assume constant Stanton numbers, even when the specific heat capacity of air is allowed to vary in Eqs. 4–5.

To this end, four different models will be considered here:

- Model 1: constant values, $\hat{c}_p$, R
- Model 2: constant specific heat capacity, temperature dependent resistance, $\hat{c}_p$, $R(T)$
- Model 3: temperature dependent specific heat capacity, constant resistance, $\hat{c}_p(T)$, R
- Model 4: temperature dependence specific heat capacity and resistance, $\hat{c}_p(T)$, $R(T)$.

To simplify the discussion and avoid invalidating the heat exchanger model in Eq. 6, we will assume that specific heat capacity of air is constant in the heat exchanger but varies with temperature in the air gap/air duct, while we will introduce temperature dependence in copper and iron. To this end, for $\hat{c}_{p,j}(T)$, $j \in \{\text{a,Cu,Fe}\}$, we will use a linear approximation given as,[1]

$$\hat{c}_{p,j}(T) = \frac{\mathscr{R}}{M_j}(a_j + b_j T), \tag{7}$$

where $\mathscr{R}$ is *universal gas constant* and M_j is the molecular mass. For the copper resistance,

$$R_j(T_j) = R_j^\circ(1 + \alpha_{\text{Cu}}(T_j - T_{\text{Cu}}^\circ)), \quad j \in \{\text{r,s}\} \tag{8}$$

where α_{Cu} is temperature coefficient of resistance for copper.

The parameters for the model of (Øyvang, 2018) are given in Table 1.

Operating conditions for the model are given in Table 2.

2.1 Overview of experimental data

A *heat-run test* of the synchronous hydro generator machine was performed for 600 min, (Øyvang, 2018). Table 3 lists measured quantities in the test.

Measurements were logged every minute for a supplied field current (I_{f}) from cold-start. The cold-run lasted 53 min, where the terminal voltage was built-up by residual flux in rotor windings. After the cold-run period, the supplied field current was increased leading to an increase in the measured stator copper and iron temperatures. The experimental results are displayed in Figure 3.

3 State Estimation

Notation used in the state estimation algorithms are given in Table 4.

A relatively general nonlinear system model can be represented as

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) + w_k \\ y_k &= h(x_k) + v_k \end{aligned} \tag{9}$$

with $w_k \sim \mathcal{N}(\bar{w}_k, \mathscr{W}_k)$ and $v_k \sim \mathcal{N}(\bar{v}_k, \mathscr{V}_k)$.

For our model, the state is $x = \begin{pmatrix} T_{\text{r}} & T_{\text{s}} & T_{\text{Fe}} \end{pmatrix}$, while the measurements are $y = \begin{pmatrix} T_{\text{s}} & T_{\text{Fe}} \end{pmatrix}$. We wish to combine the measurements (y) with the state space model to estimate the unmeasured rotor copper temperature T_{r} and air gap temperature T_{a}^{δ}. To do that, we use two different Kalman Filter algorithms: the Unscented Kalman Filter (UKF) is presented in (Simon, 2006), while the Ensemble Kalman Filter (EnKF) is succinctly described in (Brastein et al., 2019). A summary of the UKF and EnKF algorithms are given in Tables 5 and 6, respectively.

[1] We will be considering linear approximation for temperature dependent specific heat capacity. The 7-coefficients, often called as NASA Lewis coefficients, power series form is given in (McBride et al., 2002; Zehe et al., 2002) which is converted to linear approximation for simplifying mathematical models.

Table 1. Parameters for air-cooled synchronous generator model. For the NASA Lewis coefficients, see Eq. 7.

Quantity	Symbol	Value
Atmospheric pressure	p_{a}	$1.01 \cdot 10^5\,\text{N/m}^2$
Specific heat capacity, air	$\hat{c}_{p,\text{a}}$	$1.15\,\text{kJ/kg/K}$
Specific heat capacity, water	$\hat{c}_{p,\text{w}}$	$4.2\,\text{kJ/kg/K}$
Specific heat capacity, copper	$\hat{c}_{p,\text{Cu}}$	$385\,\text{J/kg/K}$
Specific heat capacity, iron	$\hat{c}_{p,\text{Fe}}$	$465\,\text{J/kg/K}$
Universal gas constant	$\mathscr{R}$	$8.314\,\text{J/K/mol}$
Molar mass, air	M_{a}	$28.97\,\text{g/mol}$
Molar mass, water	M_{w}	$18.01\,\text{g/mol}$
Molar mass, copper	M_{Cu}	$63.54\,\text{g/mol}$
Molar mass, iron	M_{Fe}	$55.84\,\text{g/mol}$
NASA Lewis coefficient-linear approx., air	a_{a}, b_{a}	$3.28,\ 6.72 \cdot 10^{-4}$
NASA Lewis coefficient-linear approx., copper	a_{Cu}, b_{Cu}	$2.56,\ 1.2 \cdot 10^{-3}$
NASA Lewis coefficient-linear approx., iron	a_{Fe}, b_{Fe}	$0.19,\ 6.76 \cdot 10^{-3}$
Copper mass, rotor	m_{r}	$9260\,\text{kg}$
Copper mass, stator	m_{s}	$6827\,\text{kg}$
Iron mass, stator	m_{Fe}	$71.2 \cdot 10^3\,\text{kg}$
Heat transfer, rotor to air gap	$\mathscr{U}A_{\text{r2}\delta}$	$2.7\,\text{kW/K}$
Heat transfer, stator copper to iron	$\mathscr{U}A_{\text{s2Fe}}$	$20\,\text{kW/K}$
Heat transfer, stator iron to air	$\mathscr{U}A_{\text{Fe2a}}$	$14.3\,\text{kW/K}$
Heat transfer, solid to air	$h_{\text{a}}A_{\text{x}}$	$55.6\,\text{kW/K}$
Heat transfer, solid to water	$h_{\text{w}}A_{\text{x}}$	$222\,\text{kW/K}$
Heat transfer, air to water	$\mathscr{U}A_{\text{x}}$	$1/\left(\frac{1}{h_{\text{a}}A_{\text{x}}} + \frac{1}{h_{\text{w}}A_{\text{x}}}\right)$
Reference temperature air	T_{a}°	$25\,^\circ\text{C}$
Rotor copper ohmic resistance, $T_{\text{r}}^\circ = 15.7\,^\circ\text{C}$	R_{r}°	$0.127\,\Omega$
Stator copper ohmic resistance, $T_{\text{s}}^\circ = 20\,^\circ\text{C}$	R_{s}°	$1.95\,\text{m}\Omega$
Resistance nominal temperature	T_{Cu}°	$25\,^\circ\text{C}$
Resistance temperature coeff.	α_{Cu}	$4.04 \cdot 10^{-3}\,^\circ\text{C}^{-1}$

Table 2. Operating conditions for air cooled synchronous generator model.

Quantity	Symbol	Value
Initial value, rotor temperature	$T_\mathrm{r}\,(t=0)$	28 °C
Initial value, stator copper temperature	$T_\mathrm{s}\,(t=0)$	28 °C
Initial value, stator iron temperature	$T_\mathrm{Fe}\,(t=0)$	28 °C
Influent water temperature	T_w^c	3.8 °C
Water mass flow rate	$\dot{m}_\mathrm{w}$	53.9 kg/s
Air mass flow rate	$\dot{m}_\mathrm{a}$	49.2 kg/s
Rated rotor field current	I_f	1055 A
Rated stator terminal current, rated	I_t	5360 A
Stator iron generated heat	$\dot{Q}_\mathrm{Fe}^\sigma$	212 kW
Friction work	$\dot{W}_\mathrm{f}$	528 kW
Friction heating	$\dot{Q}_\mathrm{f}^\sigma$	$0.8\cdot\dot{W}_\mathrm{f}$

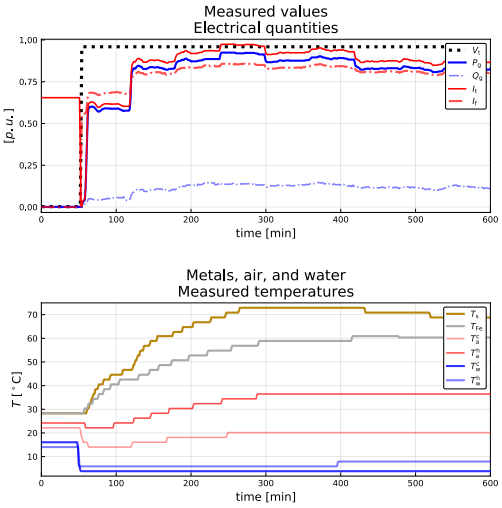


Figure 3. Experimental data for generator model from a 600 min heat-run test.

Table 3. Measured quantities.

Quantity	Symbol	Units	Sensor	#
Generator terminal voltage	V_t	kV	–	–
Active power of generator	P_g	MW	–	–
Reactive power of generator	Q_g	MVar	–	–
Rotor field current	I_f	A	–	–
Temperature of stator copper	T_s	°C	PT100	15
Temperature of stator iron	T_Fe	°C	PT100	4
Hot air temperature	T_a^h	°C	PT100 /CTD	2/2
Cold air temperature	T_a^c	°C	PT100 /CTD	2/2
Cold water temperature	T_w^c	°C	Analog	–
Hot water temperature	T_w^h	°C	Analog	–
Terminal current	$I_\mathrm{t}=\frac{P_g^2+Q_g^2}{\sqrt{3}\cdot V_\mathrm{t}}$	A	–	–

Table 4. Notations for the UKF and EnKF algorithms.

Symbol	Description
$x, \bar{x}, \hat{x}$	State vector, its mean, its estimate
x_k	Vector x at time instance k
$\hat{x}_{k\|k-1}$	*a priori* estimate of x_k based on measurements up to time t_{k-1}
$\hat{x}_{k\|k}$	*a posteriori* estimate of x_k based on measurements up to time t_k
X	State co-variance
w	Process noise
v	Measurement noise
$\mathscr{W}$	Process noise co-variance
$\mathscr{V}$	Measurement noise co-variance
K	Kalman gain
$\mathscr{E}$	Innovation co-variance
Z	Cross co-variance
ε	Error between measurement and estimate

Table 5. Algorithm: UKF.

Initialization, $k = 1$:

$\hat{x}_{1|1} = \mathbb{E}(x_1) = \bar{x}_1$

$X_{1|1} = X_1$

for $k = 2,\ 3, ...$

Propagation step:

1. Generate *sigma points* using *unscented transformation*

$x^{(i)}_{k-1|k-1} = \hat{x}_{k-1|k-1} + \tilde{x}^{(i)}, \quad i \in \{1,2,...2n\}$

where, with Cholesky root R: $R^T R = n \cdot X_{k-1|k-1}$,

$\tilde{x}^{(i)} = R_{:,i}, \quad i \in \{1,2,...n\}$

$\tilde{x}^{(n+i)} = -R_{:,i}, \quad i \in \{1,2,...n\}$

2. Propagate sigma points through process model

$x^{(i)}_{k|k-1} = f\left(x^{(i)}_{k-1|k-1}, u_{k-1}, \bar{w}_k\right), \quad i \in \{1,2,...2n\}$

3. *a priori* state and co-variance estimate

$\hat{x}_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} x^{(i)}_{k|k-1}$

$X_{k|k-1} =$

$\frac{1}{2n}\sum_{i=1}^{2n}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)^T + \mathscr{W}_k$

Information update:

1. Propagate sigma points through measurement equation

$y^{(i)}_{k|k-1} = h\left(x^{(i)}_{k-1|k-1}, u_{k-1}, \bar{v}_k\right), \quad i \in \{1,2,...2n\}$

2. Predicted measurements

$\hat{y}_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} y^{(i)}_{k|k-1}$

3. Innovation and cross co-variance

$\mathscr{E}_{k|k-1} =$

$\frac{1}{2n}\sum_{i=1}^{2n}\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T + \mathscr{V}_k$

$Z_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$

4. Kalman gain

$K_k = Z_{k|k-1}\mathscr{E}^{-1}_{k|k-1}$

5. *a posteriori* update

$\varepsilon_{k|k-1} = y_k - \hat{y}_{k|k-1}$

$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\varepsilon_{k|k-1}$

$X_{k|k} = X_{k|k-1} - K_k\mathscr{E}_{k|k-1}K_k^T$

Table 6. Algorithm: EnKF

Initialization, $k = 1$:

$x^i_{1|1} \sim \mathscr{N}(\bar{x}_1, X_1),\ i \in \{1,2,...,n_p\}$

$w^i_k \sim \mathscr{N}(\bar{w}_1, \mathscr{W}_k),\ i \in \{1,2,...,n_p\}$

$v^i_k \sim \mathscr{N}(\bar{v}_1, \mathscr{V}_k),\ i \in \{1,2,...,n_p\}$

$\hat{x}_{1|1} = \frac{1}{n_p}\sum_{i=1}^{n_p} x^{(i)}_{1|1}$

$X_{1|1} = \frac{1}{n_p-1}\sum_{i=1}^{n_p}\left(x^{(i)}_{1|1} - \hat{x}_{1|1}\right)\left(x^{(i)}_{1|1} - \hat{x}_{1|1}\right)^T$

for $k = 2,\ 3, ...$

Propagation step:

1. Propagate particles through process model

$x^{(i)}_{k|k-1} = f\left(x^{(i)}_{k-1|k-1}, u_{k-1}, w^{(i)}_{k-1}\right)\ i \in \{1,2,...,n_p\}$

2. *a priori* state and co-variance estimates

$\hat{x}_{k|k-1} = \frac{1}{n_p}\sum_{i=1}^{n_p} x^{(i)}_{k|k-1}$

$X_{k|k-1} =$

$\frac{1}{n_p-1}\sum_{i=1}^{n_p}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)^T$

Information update:

1. Propagate particles through measurement equation

$y^{(i)}_{k|k-1} = h\left(x^{(i)}_{k-1|k-1}, u_{k-1}, v^{(i)}_{k-1}\right)\ i \in \{1,2,...,n_p\}$

2. Predicted measurements

$\hat{y}_{k|k-1} = \frac{1}{n_p-1}\sum_{i=1}^{n_p} y^{(i)}_{k|k-1}$

3. Innovation and cross co-variance

$\mathscr{E}_{k|k-1} =$

$\frac{1}{n_p-1}\sum_{i=1}^{n_p}\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$

$Z_{k|k-1} =$

$\frac{1}{n_p-1}\sum_{i=1}^{n_p}\left(x^{(i)}_{k|k-1} - \hat{x}_{k|k-1}\right)\left(y^{(i)}_{k|k-1} - \hat{y}_{k|k-1}\right)^T$

4. Kalman gain

$K_k = Z_{k|k-1}\mathscr{E}^{-1}_{k|k-1}$

5. *a posteriori* update of state and co-variance

$\varepsilon^{(i)}_{k|k-1} = y_k - y^{(i)}_{k|k-1}$

$x^{(i)}_{k|k} = x^{(i)}_{k|k-1} + K_k\varepsilon^{(i)}_{k|k-1}$

$\hat{x}_{k|k} = \frac{1}{n_p-1}\sum_{i=1}^{n_p} x^{(i)}_{k|k}$

$X_{k|k} = \frac{1}{n_p-1}\sum_{i=1}^{n_p}\left(x^{(i)}_{k|k} - \hat{x}_{k|k}\right)\left(x^{(i)}_{k|k} - \hat{x}_{k|k}\right)^T$

The UKF and EnKF are initialized with $\mathscr{W} = \text{diag}(4,4,4)$, $\mathscr{V} = \text{diag}(1,1)$ and $X = 10 \cdot \mathscr{W}$. Both the process noise w and measurement noise v are considered to be *white Gaussian noise* with zero-mean. The simulation time step Δt is set to 1 min and the total time of simulation is 584 min.

The simulation environment is the Julia programming language[2]. UKF and EnKF are compared based on root mean square error (RMSE) of innovation residuals, $\varepsilon = y_k - \hat{y}_{k|k-1}$, and computational speed[3].

4 Results and Discussion

The result for air and metals temperature estimation for Model 1 ($\hat{c}_p$, R) using UKF and EnKF for different particles is given in Figure 4.

Similarly, for four different models the estimates using UKF is given in Figure 5 and using EnKF with n_p=1000 is given in Figure 6.

The rotor copper temperature and air gap temperature estimates using EnKF, for Model 1, with different particles is given in Figure 7.

Figure 5 and 6 show a substantial difference in rotor copper and air gap temperature estimates for Model 3 and Model 4: models with temperature dependence in $\hat{c}_p$ tend to decrease the temperature of metals, but increase the air temperatures. In opposition to this, models with temperature dependence in R show an increase in both metal and air temperatures.

Figure 7 shows a comparison of EnKF depending on particle number n_p: with increased n_p, the estimates converge better and give a result similar to that of the UKF.

A comparison of UKF and EnKF with different number of particles, based on RMSE of innovation residuals and computational speed, is given in Table 7.

The results show that the RMSE of the UKF is larger than that of the EnKF. Furthermore, for EnKF the residuals decrease with increased number of particles n_p. The RMSE of residuals were lowest for Model 2 as compared to the other models. The computational time increases from UKF to EnKF and with n_p. The computational time also increases when the model complexity increases from Model 1 to 2 to 3 to 4 for EnKF with $n_p = 1000$.

5 Conclusions and future work

State estimation using UKF, and EnKF with different number of particles, have been studied for four different models. Results indicate that temperature dependent heat capacities increase air temperatures and reduce metal temperatures, while temperature dependent resistances increase all temperatures. EnKF shows better estimation accuracy than UKF, but with a penalty in computational speed. In the comparison, we have re-used the constant

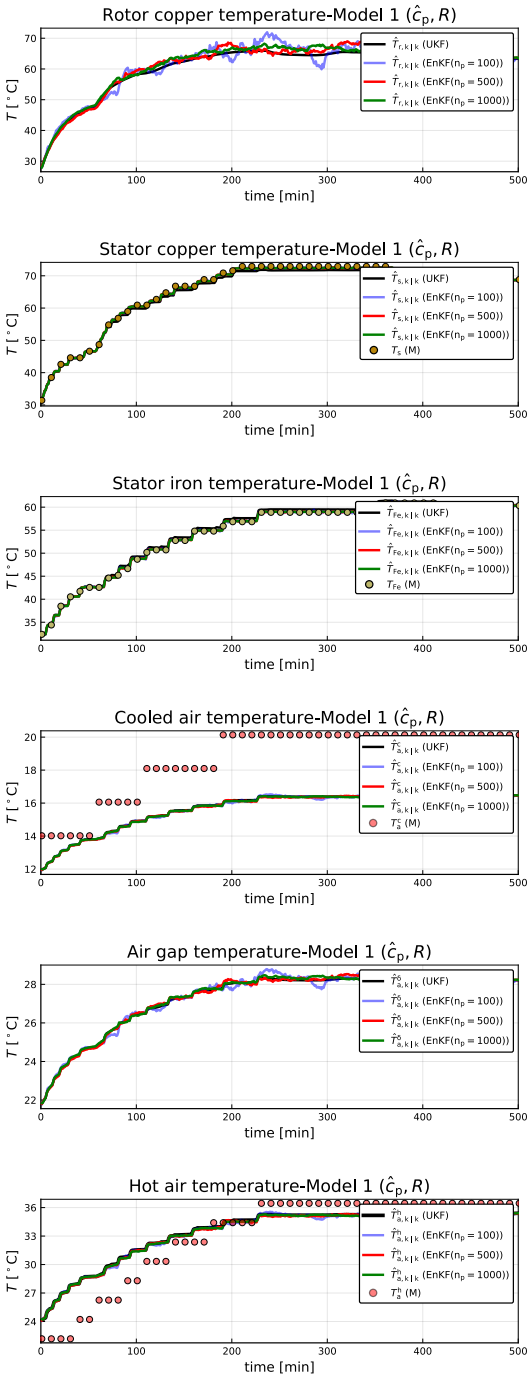
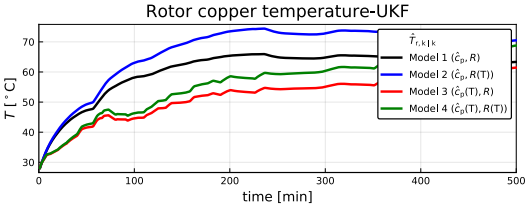
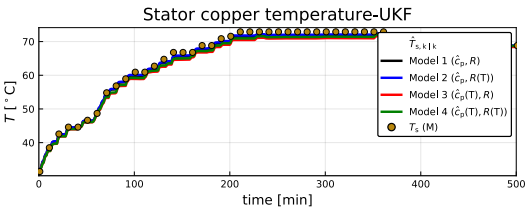


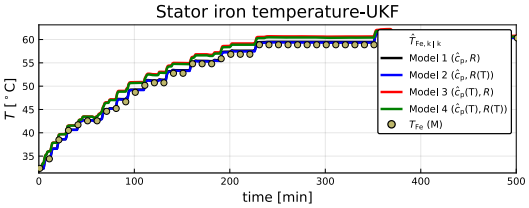
Figure 4. Air and metal temperature estimates using UKF and EnKF for Model 1 ($\hat{c}_p$, R). Subscript $k \mid k$ represents *a posteriori* estimate.

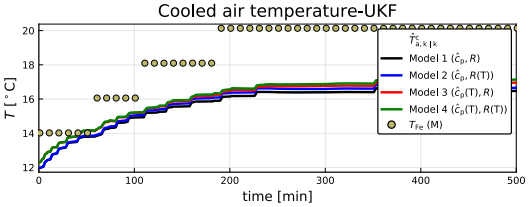
[2] Version 1.0.3 (2018-12-18)

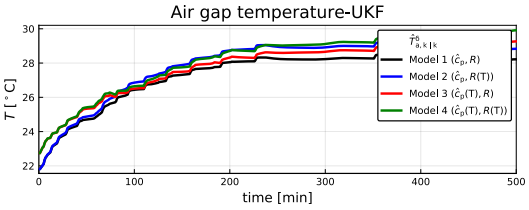
[3] Processor: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2901 Mhz, 2 Core(s), 4 Logical Processor(s)











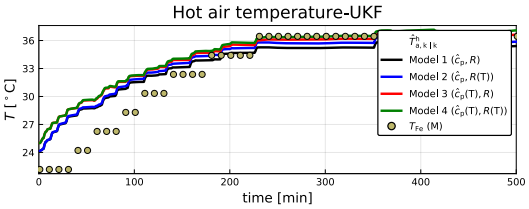
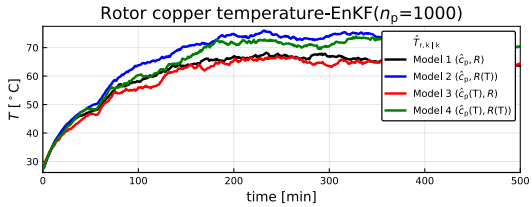
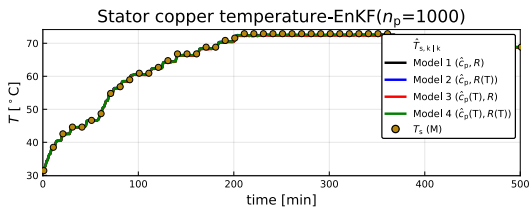
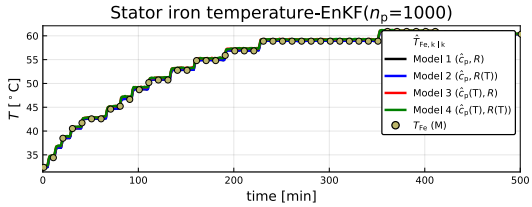
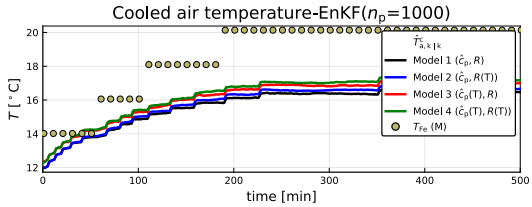


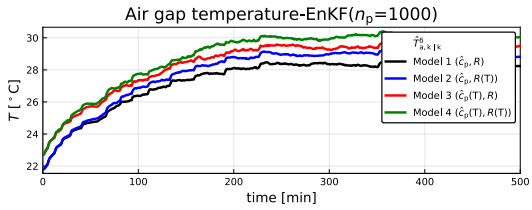
Figure 5. Air and metal temperature estimates using UKF for different models.











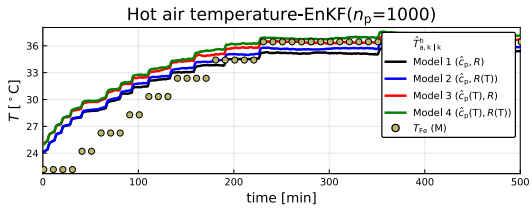


Figure 6. Air and metal temperature estimates using EnKF ($n_p = 1000$) for different models.

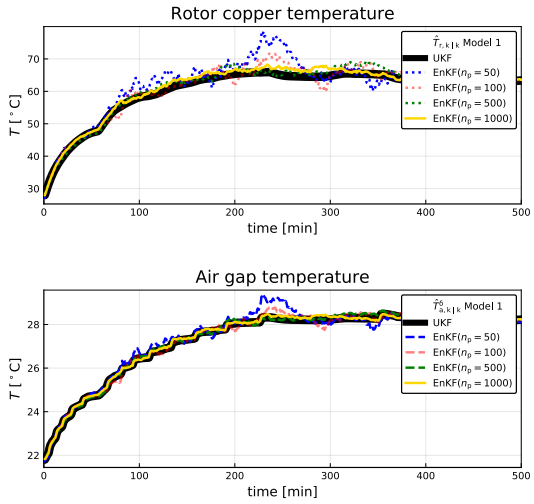


Figure 7. Rotor copper temperature and air gap temperature estimates using different number of particles for EnKF.

Table 7. Comparing Kalman filters with different models.

Model	KF	RMSE(ε)	Elapsed[s]
1	UKF	2.215	0.338
	EnKF($n_p = 50$)	2.066	1.088
	EnKF($n_p = 100$)	2.039	2.211
	EnKF($n_p = 500$)	2.010	10.860
	EnKF($n_p = 1000$)	2.012	26.343
2	UKF	1.652	0.744
	EnKF($n_p = 50$)	1.573	1.774
	EnKF($n_p = 100$)	1.524	3.414
	EnKF($n_p = 500$)	1.500	16.729
	EnKF($n_p = 1000$)	1.492	32.225
3	UKF	3.137	1.041
	EnKF($n_p = 50$)	2.735	3.238
	EnKF($n_p = 100$)	2.729	7.643
	EnKF($n_p = 500$)	2.705	36.663
	EnKF($n_p = 1000$)	2.701	58.595
4	UKF	2.730	0.798
	EnKF($n_p = 50$)	2.407	3.154
	EnKF($n_p = 100$)	2.342	5.287
	EnKF($n_p = 500$)	2.331	35.877
	EnKF($n_p = 1000$)	2.327	60.993

model parameters in all the models. Because these parameters essentially have been tuned for Model 1, it is difficult to draw strong conclusions on which model is best at this moment.

Future work will involve studies of (i) temperature dependent specific heat capacity for air and water with numeric solution of the resulting two point boundary value problem, (ii) extending the number of outputs from two (T_s, T_{Fe}) to four (T_s, T_{Fe}, T_a^c, and T_a^h), (iii) and a more formal model fitting for the various models.

References

Ole Magnus Brastein, Bernt Lie, Roshan Sharma, and Nils-Olav Skeie. Parameter estimation for externally simulated thermal network models. *Energy and Buildings*, 191:200–210, 2019. doi:10.1016/j.enbuild.2019.03.018.

ENTSO-E. Commission regulation (eu) 2016/631 of 14 april 2016 establishing a network code on requirements for grid connection of generators. Technical report, European Network of Transmission System Operators for Electricity, ENTSO-E Avenue de Cortenbergh 100 1000 Brussels Belgium, 2016.

Philip A. Hargreaves, B.C. Mecrow, and Ross Hall. Calculation of Iron Loss in Electrical Generators Using Finite-Element Analysis. *Industry Applications, IEEE Transactions on*, 48(5):1368–1373, May 2011. doi:10.1109/IEMDC.2011.5994805.

Bernt Lie. Solution, Project, FM1015 Modelling of Dynamic Systems. University of South-Eastern Norway, November 2018.

Bonnie J McBride, Michael J Zehe, and Sanford Gordon. Nasa glenn coefficients for calculating thermodynamic properties of individual species. Technical Report NASA/TP–2002-21155, NASA, NASA Center for Aerospace Information 7121 Standard Drive Hanover, MD 21076, 2002. URL `http://gltrs.grc.nasa.gov/GLTRS`.

Thomas Øyvang. *Enhanced power capability of generator unites for increased operational security*. PhD thesis, University of South-Eastern Norway, Faculty of Technology, Natural Sciences and Maritime Sciences University of South-Eastern Norway N-2018 Porsgrunn Norway, December 2018. ISBN: 978-82-7206-503-3 (print) ISBN: 978-82-7206-504-0 (online).

Dan Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience, Hoboken, New Jersey, 2006.

Statnett. Fiks funksjonskrav i kraftsystemet [functional requirements in the power system]. Technical report, Statnett, 2012.

Michael J. Zehe, Sanford Gordon, and Bonnie J. McBride. CAP: A Computer Code for Generating Tabular Thermodynamic Functions from NASA Lewis Coefficients. Technical Report NASA/TP–2001-210959/REV1, NASA, NASA Center for Aerospace Information 7121 Standard Drive Hanover, MD 21076, 2002. URL `http://gltrs.grc.nasa.gov/GLTRS`.