2021

MACROMODELING AND ACCELERATED SIMULATIONS OF ELECTRIC MACHINES

Ajay Pratap Yadav

MACROMODELING AND ACCELERATED SIMULATIONS OF ELECTRIC MACHINES

by

AJAY PRATAP YADAV

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy at
The University of Texas at Arlington
August 2021

Arlington, Texas

Supervising Committee:
Ali Davoudi, Supervising Professor
Ramtin Madani
Rasool Kenarangui
Jonathan Bredow
David Levine

ABSTRACT

MACROMODELING AND ACCELERATED SIMULATIONS OF ELECTRIC MACHINES

AJAY PRATAP YADAV, Ph.D.

The University of Texas at Arlington, August 2021

Supervising Professors: Ali Davoudi

Electric machines are the most important element in the power grid. Given its centennial legacy and the rise of electric vehicles and distributed energy resources, it is imperative to bring new technologies into this area. This work tries to bridge the gap between electric machines and innovative research domains such as convex optimization and FPGA-based hardware acceleration. Problems of electric machine parameter identification and real-time simulation are considered. A convex optimization-based framework is designed to identify machine parameters. This tool is used to perform the macromodeling of a synchronous machine from its magnetic-equivalent circuit model. Furthermore, it is used to obtain induction machine parameters using limited and non-intrusive measurements. Given that optimization-based methods are usually offline, partial-update Kalman filter is investigated for online electric machine state and parameter estimation. Finally, the hardware acceleration of electric machine models executed on FPGA is studied.

ACKNOWLEDGEMENTS

First of all, I am grateful to Almighty Krishna for ensuring that I came out of my Ph.D. with good health.

I am thankful to Dr. Ali Davoudi for his time and patience. Thank you for the countless hours we spent together (*Honestly, I learned more life than research from him*). I am grateful to the National Science Foundation and The University of Texas at Arlington for funding my research and education. A big thanks to my collaborators for their help, especially Dr. Ramtin Madani, Dr. Mohammadreza Davoodi, Dr. Nicholas Gans, and Dr. Nuh Erdogan. Thanks to Dr. Navid Amiri and Dr. Juri Jatskevich for providing hardware measurements. I appreciate my lab members, Dr. Omar Ali Beg, Dr. Shankar Abhinav, and Dr. Susan Zuo for their support. I am grateful to Dr. Deepak Pullaguram for his invaluable help during a crunch time.

A big hug to Dr. Tuncay Altun for being with me throughout the pain & gains of this Ph.D. journey. I will miss our monthly trip to Dimassi restaurant (also, *late-night drives and Turkish music*). Special thanks to my friends Jasasswee Das, Pratik Ghate, Sayantan Chakraborty, and Erik for all the fun.

Finally, I thank my family for their love, and my wife, Dr. Megha Singh, who was there for me during my lowest moments.

July 28, 2021 Ajay Pratap Yadav

DEDICATION

To my mother's struggles for Education

TABLE OF CONTENTS

ABSTRACT ii

ACKNOWLEDGEMENTS iv

DEDICATION v

CHAPTER ONE: INTRODUCTION 1

CHAPTER TWO: MACROMODELING OF ELECTRIC MACHINES FROM AB INITIO MODELS 6

CHAPTER THREE: INDUCTION MACHINE PARAMETERIZATION FROM LIMITED TRANSIENT DATA USING CONVEX OPTIMIZATION . . . 32

CHAPTER FOUR: PMSM ESTIMATES FROM INTERMITTENT DATA . . 65

CHAPTER FIVE: HARDWARE-ASSISTED SIMULATION OF VOLTAGE-BEHIND-REACTANCE MODELS OF ELECTRIC MACHINES ON FPGA 94

CHAPTER SIX: CONCLUSION 121

APPENDIX A 122

REFERENCES 124

BIOGRAPHICAL STATEMENT 142

CHAPTER 1

INTRODUCTION

1.1 Motivation

Given that electric machines accounts for more than 65% of the industrial load, industry-wide transition to efficient machines can save billions of dollars [1, 2]. Hence, accurate models and high-performance simulation tools are too important to ignore. Electric machine models can be classified as physics-based models and lumped-parameter models [1]. Physics-based models are derived from first principles, hence, can provide high fidelity simulations that accurately mimic hardware. Popular physics-based models for electric machines are the finite-element methods (FEM) and magnetic equivalent circuits (MEC) models. FEM models are computationally expensive and mainly used for the final design verification. MEC modeling involves representing the physical machine as magnetic circuits using geometry and materials characteristics. This results in a less intensive model albeit at expense of some accuracy. Lumped-parameter models are high-level models derived using simplifying approximations, e.g., assuming sinusoidally distributed windings. Popular machine representations such as phase-domain (PD), direct–quadrature ($qd0$), and voltage-behind-reactance (VBR) models fall in this category. These models can be expressed as differential-algebraic equations and used for most system-level studies. One of the main advantages of such models is that they can be written using few parameters, e.g., rotor resistance or magnetizing inductance.

Detailed laboratory tests have been designed to identify electric machine parameters, however, it is easier said than done [3, 4]. Factors such as size or the inability

to separate the machine from attached equipment can make it impossible to run experimental tests. In the following chapters, we will present data-driven approaches for machine characterization.

Real-time simulation of different physical systems has gain traction in recent years [5, 6]. Such techniques enable testing the machine and controller performances under real-world conditions without risking the device's safety. Herein, we also present an FPGA-based simulation of VBR machine models for a family of electric machines.

1.2 Outline

This thesis is organized as follows,

1.2.1 Chapter 2: Macromodeling of electric machines from ab initio models

1.2.1.1 Description

The Lumped-parameter *qd0* model of a synchronous machine is extracted from its physics-based magnetic-equivalent circuit model. Model extraction is formulated as a weighted least square optimization that aims to minimize the mismatch between the MEC model and target *qd0* model. This problem being non-convex, cannot be solved using standard methods, therefore, is solved using cone programming.

1.2.1.2 Authors' Contribution

The author was the principal architect of this paper. Dr. Ali Davoudi supervised the work from inception to completion. Dr. Ramtin Madani and Dr. Tuncay Altun provided underlying concepts on convex optimization.

1.2.2 Chapter 3: Induction machine parameterization from limited transient data using convex optimization

1.2.2.1 Description

Induction machine parameters are identified using limited and non-intrusive observations of available input voltages, stator currents, and rotor speed. Parameter extraction is formulated as a non-convex estimation problem, which is solved using a convex optimization framework. The proposed method is experimentally verified on a squirrel-cage induction machine. This non-intrusive approach is especially useful for setups where lab testing is not feasible (e.g., large-size machines).

1.2.2.2 Authors' Contribution

The author was the principal architect of this paper. Dr. Ali Davoudi supervised the work from inception to completion. Dr. Ramtin Madani provided underlying concepts on convex optimization. Dr. Navid Amiri provided the experimental data for an induction motor. Dr. Juri Jatskevich provided valuable comments during manuscript preparation.

1.2.3 Chapter 4: PMSM Estimates from Intermittent Data

1.2.3.1 Description

Kalman filter is a standard tool in the industry for observer design. Recently, it was proposed that partially performing the measurement update in an extended Kalman filter can improve its handling of model uncertainties and nonlinearities (hence, named *partial-update Kalman Filter (PKF)*). We have designed a variant of PKF that is robust to losses in the measurement data.

1.2.3.2 Authors' Contribution

The author was the principal architect of this work. Dr. Mohammadreza Davoodi was actively involved in the write-up and provided valuable help on the derivations. Dr. Ali Davoudi supervised the work from inception to completion. Dr. Nicholas Gans provided helpful feedback on Kalman filter.

1.2.4 Chapter 5: Hardware-assisted simulation of voltage-behind-reactance models of electric machines on FPGA

1.2.4.1 Description

The acceleration of numerical simulations executed on Field-Programmable Gate Arrays (FPGAs) for electric machines represented by voltage-behind-reactance (VBR) models is studied. *In this work, the objective is to merge the domains of optimization and FPGA*. Given that FPGA deployment involves arranging hardware circuits in an FPGA board, a set of different circuits could be designed to execute the same function. Depending upon the requirements, a user could demand to minimize the utilized area of FPGA (circuit size/resources) or the circuit latency. In this work, we achieved the faster simulation of machine models on an FPGA by minimizing latency.

1.2.4.2 Authors' Contribution

The author was the principal architect of this work. Dr. Ali Davoudi supervised the work from inception to completion. Dr. Siyuan Xu performed the FPGA deployment of the machine models. Dr. Benjamin Schafer provided useful comments during manuscript preparation.

MACROMODELING OF ELECTRIC MACHINES FROM AB INITIO MODELS[1]

Authors: A. P. Yadav, T. Altun, R. Madani, and A. Davoudi.

Reprinted, with permission from all the co-authors.

Journal: IEEE Transactions on Energy Conversion [7].

DOI: 10.1109/TEC.2020.2970965

CHAPTER 2

MACROMODELING OF ELECTRIC MACHINES FROM AB INITIO MODELS

Abstract

We extract the lumped-parameter model of a wound-rotor synchronous machine from its physics-based magnetic-equivalent circuit model. Model extraction is formulated as a weighted least square optimization with nonlinear constraints in which time-domain trajectories of flux linkages, currents, and the electromagnetic torque are used as input data to obtain the parameters of the $qd0$ model of the machine. The resulting problem is non-convex and cannot be solved using standard methods. The optimization problem is, therefore, convexified using a cone programming relaxation. The solution to the relaxed problem is used as an initial point for the interior-point method, leading to a reliable framework. Accurate estimations on stator resistance, leakage and mutual inductances in stator and rotor, rotor speed, effective turns-ratio between the field and stator windings, and the number of poles are obtained. Estimated parameters are validated against measured and estimated values reported in literature, and are used to develop a behavioral $qd0$ macromodel of the machine.

2.1 Introduction

Electric machine models can be classified into lumped-parameter models, such as *abc* phase-domain models, $qd0$ models, or voltage-behind reactance models [4],

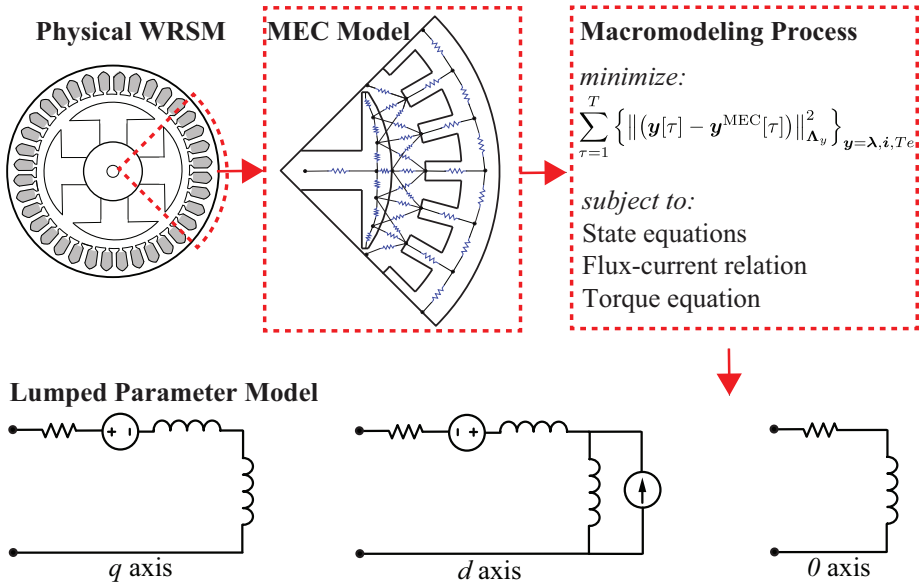


Figure 1: $qd0$ model extraction from the MEC model of a WRSM.

and those based on the first principles of physics (Maxwell equations), such as finite-element methods (FEM) or magnetic equivalent circuits (MEC). Lumped-parameter macromodels are suitable for system-level studies, drive-controller design, or hardware-in-the-loop applications[4, 8]. FEM models are highly accurate and closely mimic the hardware, but they are computationally expensive and mainly used for the final design verification. MEC modeling is an intuitive approach based on the circuit theory. Herein, we extract the lumped parameters of a dynamic *qd0* model from the data generated by the MEC model. We use a mesh-based MEC model of a wound-rotor synchronous machine (WRSM)[9, 10, 11, 12] as it exhibits better numerical properties than its nodal-based counterpart [13]. WRSMs are used in various applications, e.g., aircraft generators, ship propulsion, and wind turbines [14]. $qd0$ model extraction is analogous to replacing the spatially-distributed reluctance network of the MEC model with equivalent lumped-parameter components (see Figure 1).

IEEE Std-115 [3] documents an array of tests to identify synchronous machine parameters. Since it is not always feasible to run all the necessary tests, various techniques estimate machine parameters using transient measurements [15, 16, 17, 18, 19]. In the absence of hardware measurement, high-fidelity physics-based models, that closely mimic the experimental prototype, can be used instead. Furthermore, detailed models provide access to a host of variables that might not be available experimentally due to a limited sensory. [20] and [21] have used FEM models to determine the parameters of equivalent circuits for induction machines. [22] has used the data from a pulse test applied to the FEM model of a synchronous machine to obtain its lumped-parameter model. However, MEC models have not yet been exploited for such macromodeling purposes. Another aspect of our contribution is the extraction methodology. While optimization methods based on convex relaxation have found wide applications in power system areas [23, 24, 25], they have not yet been properly explored for model extraction and parameter estimation of electric machines [26, 27]. We formulate the parameter extraction process as a weighted least-square problem with nonlinear constraints such that it minimizes the mismatch between trajectories produced by the MEC model and those predicted by the $qd0$ model.

Due to the presence of nonlinear equality constraints, the proposed optimization problem is non-convex and cannot be solved reliably using standard interior-point method (IPM) solvers [28, 29, 30]. To eliminate the need for initialization we propose a hybrid optimization framework based on cone programming relaxation. We relax the original problem and feed the resulting solution to IPM in order to arrive at fully-feasible and globally optimal solutions [31]. The contributions of this paper are summarized below:

- We have recovered machine parameters, namely stator resistance, rotor speed, stator leakage inductance, q-axis magnetizing inductance, d-axis magnetizing inductance, equivalent turns-ratio between the field and the stator windings, and the number of poles. Extracted parameters are compared against those values reported in literature.
- The model extraction is formulated as a non-convex optimization problem using a hybrid cone programming and IPM approach.
- The extracted $qd0$ model is validated against the MEC model, and shown to capture dominant dynamical modes with more than 20 times faster model execution.

The outline of the paper is as follows. Section 2.2 is a preliminary review on the symbols and notations used throughout the paper. Section 2.3 presents the dynamic MEC model and the target $qd0$ model for WRSM. Section 2.4 elaborates a discrete $qd0$ model for WRSM, and formulates the parameter extraction process as a non-convex optimization problem. Section 2.5 presents cone programming relaxation and objective penalization. Section 2.6 studies the implementation process and results.

2.2 Notation

Matrices and vectors are presented as bold uppercase and lowercase variables, respectively (e.g., $\mathbf{X}$ or $\mathbf{x}$). $\mathbf{x}_i$ denotes the i^{th} element of vector $\mathbf{x}$. $\mathbf{X}_{ij}$ denotes the element in the i^{th} row and j^{th} column of matrix $\mathbf{X}$. $\mathbb{R}^n$ is the set of column vectors of size $n \times 1$. $\mathbb{R}^{m \times n}$ is the set of matrices of size $m \times n$. $\mathbb{S}^n$ denotes the set of symmetric matrices of size n. $\mathbf{0}_m$ and $\mathbf{1}_m$ represent vectors of size $m \times 1$ with all their elements as 0 and 1, respectively. Similarly, $\mathbf{0}_{m \times n}$ represents a matrix of size $m \times n$ with all its element equal to 0. $\boldsymbol{I}_n$ is an identity matrix of size n. $||\mathbf{x}||_2$ denotes the Euclidean norm of the vector $\mathbf{x}$. $\text{diag}(\mathbf{x})$ gives a diagonal matrix with elements of the vector $\mathbf{x}$

along its diagonal. $\text{diag}(\mathbf{X})$ gives a vector with its elements the same as the diagonal of the input matrix. $\text{diag}(\mathbf{X}, \mathbf{Y})$ produces a block diagonal matrix with input matrices along its diagonal. $(\cdot)^\top$ indicates the transpose operation. $||\mathbf{x}||^2_{\mathbf{Z}}$ represents $\mathbf{x}^\top \mathbf{Z}\mathbf{x}$. $\langle \mathbf{X}, \mathbf{Y} \rangle$ stands for the inner product of matrices $\mathbf{X}$ and $\mathbf{Y}$. $\bar{}$ denotes the square of a variable, e.g., $\overline{x}$ represents x^2 for a scalar x and $\overline{\mathbf{x}}$ is a vector with elements of $\mathbf{x}$ squared. $\sqrt{\mathbf{x}}$ denotes an element-wise square root operation on vector $\mathbf{x}$.

2.3 Ab Initio and lumped-parameter WRSM models

2.3.1 Dynamic MEC Model

This section presents a concise overview of the mesh-based MEC model of the WRSM in [9, 10, 11, 12]. We have selected the static MEC model in [9] and formed a dynamic MEC model using the procedure laid out in [12]. Therein, the field current and rotor speed are assumed constant. Damper windings are ignored in [9, 10, 11]. Different segments of stator, rotor, and airgap are modeled using flux tubes to form a magnetic circuit, as seen in Figure 1. Reluctance formulation is based on the geometry and permeability information of respective flux tubes [12]. Kirchhoff's voltage law on individual loops gives

$$\overline{\boldsymbol{\mathcal{R}}}\boldsymbol{\Phi} = \boldsymbol{\mathcal{F}}, \tag{2.1}$$

where $\overline{\boldsymbol{\mathcal{R}}} \in \mathbb{S}^{nl}$ is the matrix of reluctances, $\boldsymbol{\Phi} \in \mathbb{R}^{nl}$ is vector of flux terms in each loop, $\boldsymbol{\mathcal{F}} \in \mathbb{R}^{nl}$ is the vector of MMF sources, and nl is the number of loops. Note that due to the rotor motion, the reluctances of the flux tubes in the airgap region in matrix $\overline{\boldsymbol{\mathcal{R}}}$ changes with time and should be recalculated at every step. The elements

of $\boldsymbol{\mathcal{F}}$ can be obtained as the product of winding turns and currents for each magnetic loop. One can get the flux linkages, $\boldsymbol{\lambda}_{abcs}$, using [11]

$$\boldsymbol{\lambda}_{abcs} = P\mathbf{N}_{abcs}^{\top}\boldsymbol{\Phi}_{st}, \tag{2.2}$$

where P is the number of poles, $\mathbf{N}_{abcs}$ is the turns matrix for stator windings, and $\boldsymbol{\Phi}_{st}$ represents the flux in loops corresponding to stator segments of magnetic circuit.

[12] has reformulated (2.1) such that flux linkages, $\boldsymbol{\lambda}_{abcs}$, are the inputs and winding currents, $\boldsymbol{i}_{abcs}$, are the output.

$$\begin{bmatrix} \overline{\boldsymbol{\mathcal{R}}} & -c_{scl}\mathbf{N}_{abcs}(\boldsymbol{K_s})^{-1} \\ c_{scl}\boldsymbol{K_s}\mathbf{N}_{abcs}^{\top} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Phi} \\ \boldsymbol{i}_{qd0s}/c_{scl} \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{fld} & \mathbf{0} \\ \mathbf{0} & c_{scl}\boldsymbol{I}_3/P \end{bmatrix} \begin{bmatrix} i_{fld} \\ \boldsymbol{\lambda}_{qd0s} \end{bmatrix}. \tag{2.3}$$

The idea is to incorporate the machine dynamics into the otherwise static relation of current and flux in (2.1). $\boldsymbol{i}_{qd0s}$ and $\boldsymbol{\lambda}_{qd0s}$ are the currents and flux linkages of the stator windings in the rotor reference frame. $\boldsymbol{z}_{qd0} = \boldsymbol{K_s}\boldsymbol{z}_{abc}$, where $\boldsymbol{z}$ represents flux linkages, currents, or voltages, and $\boldsymbol{K_s}$ is the reference frame transformation matrix [4]. $\mathbf{N}_{fld}$ is the turns matrix for the field winding. i_{fld} is the field current. c_{scl} is a scaling factor to condition (2.3). Equation (2.3) can be solved to obtain currents, $\boldsymbol{i}_{qd0s}$, for given flux linkages, $\boldsymbol{\lambda}_{qd0s}$.

State-space representation of a synchronous machine in the rotor reference-frame is [4]

$$\frac{d\lambda_{qs}}{dt} = v_{qs} - r_s i_{qs} - \omega_r \lambda_{ds}, \tag{2.4a}$$

$$\frac{d\lambda_{ds}}{dt} = v_{ds} - r_s i_{ds} + \omega_r \lambda_{qs}, \tag{2.4b}$$

$$\frac{d\lambda_{0s}}{dt} = v_{0s} - r_s i_{0s}. \tag{2.4c}$$

v_{qs}, v_{ds}, and v_{0s} are the q-axis, d-axis, and 0-axis voltage terms. r_s is the stator resistance, and ω_r is the rotor speed. Dynamic MEC model solves (2.3) and (2.4) in tandem. The electromagnetic torque is calculated using [11]

$$T_e^{\text{MEC}} = \left(\frac{P}{2}\right)^2 \sum_{j=1}^{na} \left(\frac{\phi_{aj}}{P_{aj}}\right)^2 \frac{\partial P_{aj}}{\partial \theta_r}, \tag{2.5}$$

where ϕ_{aj} and P_{aj} represent flux and permeance for the j^{th} loop in the airgap region. na is the number of airgap loops (changing with the rotor position), and θ_r is the rotor position.

2.3.2 $qd0$ Model

State-space representation of a lumped-parameter model is the same as (2.4). However, the relation between flux linkages and currents is formulated as [4]

$$\lambda_{qs} = (L_{ls} + L_{mq}) i_{qs}, \tag{2.6a}$$

$$\lambda_{ds} = (L_{ls} + L_{md}) i_{ds} + \frac{2}{3} \frac{N_{fld}}{N_s} L_{md} i_{fld}, \tag{2.6b}$$

$$\lambda_{0s} = L_{ls} i_{0s}. \tag{2.6c}$$

L_{ls} is the leakage inductance of stator, L_{mq} and L_{md} are the q- and d-axis magnetizing inductances, respectively. N_{fld} and N_s are the lumped equivalency of spatially-distributed field winding and stator winding, respectively. The electromagnetic torque becomes

$$T_e = \frac{3P}{4} (\lambda_{ds} i_{qs} - \lambda_{qs} i_{ds}) \ . \tag{2.7}$$

$qd0$ parameter extraction from the MEC model is analogous to replacing the flux linkage and current relation in (2.3) with (2.6).

2.4 Setting up the Model Extraction Procedure

In order to extract the $qd0$ model from the MEC model, one can compare trajectories generated from (2.3), (2.4), and (2.5) with those predicted by (2.6), (2.4), and (2.7) and minimize their mismatch. This requires one to discretize the $qd0$ model.

2.4.1 Discretizing Machine Dynamics

While a host of methods are available, (e.g., Tustin's [32]), in this work, the forward Euler method is adopted for its simplicity. The discretized state trajectory for a general dynamic system $\frac{d\mathbf{x}}{dt} = f(\mathbf{x})$ using the forward Euler method is

$$\mathbf{x}[\tau + 1] = \mathbf{x}[\tau] + \Delta T \times f(\mathbf{x}) \ . \tag{2.8}$$

$\mathbf{x}$ is the state, ΔT is the sampling time, and τ is the time instance. The $qd0$ model in (2.4), (2.6), and (2.7) is discretized as

$$\boldsymbol{\lambda}[\tau + 1] = \boldsymbol{A}\boldsymbol{\lambda}[\tau] + \boldsymbol{R}\boldsymbol{i}[\tau] + \boldsymbol{v}[\tau], \tag{2.9a}$$

$$\boldsymbol{\lambda}[\tau] = \boldsymbol{L}\boldsymbol{i}[\tau] + \boldsymbol{\ell}\, i_{fld}, \tag{2.9b}$$

$$Q \times T_e[\tau] = \frac{3}{4}\boldsymbol{\lambda}^{\top}[\tau]\boldsymbol{M}\boldsymbol{i}[\tau]. \tag{2.9c}$$

$\boldsymbol{\lambda}[\tau] \in \mathbb{R}^3$ and $\boldsymbol{i}[\tau] \in \mathbb{R}^3$ are the flux linkages and currents, respectively, in the rotor reference frame at time instance τ. The $qd0$ subscript is dropped for brevity. $\boldsymbol{v}[\tau]$

in (2.9a) is the $qd0$ voltage terms times the sampling time, $\boldsymbol{v}[\tau] = \boldsymbol{v}_{qd0}[\tau] \times \Delta T$. Matrices $\boldsymbol{A}$, $\boldsymbol{R}$, $\boldsymbol{L}$, $\boldsymbol{\ell}$, $\boldsymbol{M}$, and Q in (2.9) are defined as

$$\boldsymbol{A} \triangleq \begin{bmatrix} 1 & -a & 0 \\ a & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{R} \triangleq \begin{bmatrix} -r & 0 & 0 \\ 0 & -r & 0 \\ 0 & 0 & -r \end{bmatrix}, \tag{2.10a}$$

$$Q \triangleq \frac{1}{P}, \qquad \boldsymbol{L} \triangleq \begin{bmatrix} l_1 & 0 & 0 \\ 0 & l_2 & 0 \\ 0 & 0 & l_3 \end{bmatrix}, \tag{2.10b}$$

$$\boldsymbol{\ell} \triangleq \begin{bmatrix} 0 \\ l_4 \\ 0 \end{bmatrix}, \quad \boldsymbol{M} \triangleq \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2.10c}$$

where $a \triangleq \omega_r \Delta T$, $r \triangleq r_s \Delta T$, $l_1 \triangleq L_{ls} + L_{mq}$, $l_2 \triangleq L_{ls} + L_{md}$, $l_3 \triangleq L_{ls}$, and $l_4 \triangleq \frac{2}{3}\frac{N_{fld}}{N_s} L_{md}$. The model extraction problem presented in following sections aims to determine the values $(a, r, l_1, l_2, l_3, l_4, Q)$ based on which machine parameters, ω_r, r_s, L_{ls}, L_{mq}, L_{md}, $\frac{N_{fld}}{N_s}$, and P can be uniquely determined.

2.4.2 Problem Formulation

Suppose that we are given the vectors $\boldsymbol{\lambda}^{\mathrm{MEC}}[\tau]$, $\boldsymbol{i}^{\mathrm{MEC}}[\tau]$, and $T_e^{\mathrm{MEC}}[\tau]$ throughout a discrete time horizon $\tau \in \{1, 2, \ldots, \mathfrak{t}\}$, representing flux linkages, currents, and torque data from the MEC model, respectively. Then, the parameter extraction for a $qd0$ model from the MEC data can be formulated as the following optimization problem:

$$\textbf{minimize} \sum_{\tau=1}^{\mathfrak{t}} \left\| \boldsymbol{\lambda}[\tau] - \boldsymbol{\lambda}^{\mathrm{MEC}}[\tau] \right\|_{\boldsymbol{\Lambda}_\alpha}^2 + \sum_{\tau=1}^{\mathfrak{t}} \left\| \boldsymbol{i}[\tau] - \boldsymbol{i}^{\mathrm{MEC}}[\tau] \right\|_{\boldsymbol{\Lambda}_\beta}^2$$

$$+\sum_{\tau=1}^{\mathfrak{t}}\gamma\left(T_e[\tau]-T_e^{\mathrm{MEC}}[\tau]\right)^2 \tag{2.11a}$$

subject to

$$\boldsymbol{\lambda}[\tau+1]=\boldsymbol{\lambda}[\tau]+a\times\Big[-\lambda_2[\tau],\lambda_1[\tau],0\Big]^\top-r\times\boldsymbol{i}[\tau]+\boldsymbol{v}[\tau],$$
$$\tau=1,2,\ldots,\mathfrak{t}-1 \tag{2.11b}$$

$$\boldsymbol{\lambda}[\tau]=\mathrm{diag}\{[l_1,l_2,l_3]\}\boldsymbol{i}[\tau]+[0,\ l_4,\ 0]^\top\times i_{fld},$$
$$\tau=1,2,\ldots,\mathfrak{t} \tag{2.11c}$$

$$Q\times T_e[\tau]=\frac{3}{4}\boldsymbol{\lambda}^\top[\tau]\Big[-i_2[\tau],i_1[\tau],0\Big]^\top,\qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.11d}$$

variables

$$\left\{\boldsymbol{\lambda}[\tau]\in\mathbb{R}^3,\boldsymbol{i}[\tau]\in\mathbb{R}^3,T_e[\tau]\in\mathbb{R}\right\}_{\tau=1}^{\mathfrak{t}},$$
$$a,l_1,l_2,l_3,l_4,r,Q\in\mathbb{R}.$$

The 3×3 matrices $\boldsymbol{\Lambda_\alpha}$ and $\boldsymbol{\Lambda_\beta}$ are defined as

$$\boldsymbol{\Lambda_\alpha}\triangleq\begin{bmatrix}\alpha_1&0&0\\0&\alpha_2&0\\0&0&\alpha_3\end{bmatrix},\ \boldsymbol{\Lambda_\beta}\triangleq\begin{bmatrix}\beta_1&0&0\\0&\beta_2&0\\0&0&\beta_3\end{bmatrix}. \tag{2.12}$$

$(\alpha_1,\alpha_2,\alpha_3)$, $(\beta_1,\beta_2,\beta_3)$ and γ are user-defined non-negative weights given to flux, current and torque data, respectively.

The objective function in (2.11a) represents the total mismatch between the MEC and $qd0$ trajectories. The equality constraint in (2.11b) represents the state equation for the $qd0$ model. Constraints in (2.11c) and (2.11d) form the flux linkage-current relationship and torque expressions, respectively. The auxiliary variable $Q=$

$1/P$ is defined to reduce the order of torque equation from three (cubic) to two (quadratic), which will help in the following convex formulation. To summarize, the optimization problem in (2.11) minimizes the sum of scaled residuals for trajectories of flux linkages, currents, and torque to uniquely obtain $\boldsymbol{\lambda}$, $\boldsymbol{i}$, T_e, $\boldsymbol{A}$, $\boldsymbol{R}$, $\boldsymbol{L}$, $\boldsymbol{\ell}$, and Q subject to the constraints in (2.11a) - (2.11d).

2.5 Convex Relaxation and Numerical Search

Without proper initialization, local search algorithms may fail to converge to a globally optimal solution or even a feasible point. To address this issue, we use cone programming relaxation followed by IPM to reliably solve the problem (2.11a) - (2.11d). The proposed convex relaxation is explained next.

2.5.1 Cone Programming Relaxation

The original optimization problem in (2.11) is non-convex due to the presence of bilinear terms:

- $a\ \lambda_2[\tau]$, $a\ \lambda_1[\tau]$ and $r\ \boldsymbol{i}[\tau]$ in (2.11b);
- $l_1\ i_1[\tau]$, $l_2\ i_2[\tau]$ and $l_3\ i_3[\tau]$ in (2.11c); as well as
- $\lambda_1[\tau]\ i_2[\tau]$, $\lambda_2[\tau]\ i_1[\tau]$ and $Q\ T_e[\tau]$ in (2.11d).

The aforementioned nonlinearity can be tackled by introducing new variable (i.e., lifting the problem). To this end, define

$$\boldsymbol{f}[\tau] \triangleq \Big[a\lambda_2[\tau],\ -a\lambda_1[\tau],\ 0\Big]^\top, \tag{2.13a}$$

$$\boldsymbol{h}[\tau] \triangleq r \times \boldsymbol{i}[\tau], \tag{2.13b}$$

$$\boldsymbol{z}[\tau] \triangleq \mathrm{diag}\{[l_1,\ l_2,\ l_3]\}\ \boldsymbol{i}[\tau], \tag{2.13c}$$

$$\boldsymbol{w}[\tau] \triangleq \Big[\lambda_1[\tau]i_2[\tau], \lambda_2[\tau]i_1[\tau],\ 0\Big]^\top, \tag{2.13d}$$

$$\theta[\tau] \triangleq Q \times T_e[\tau]. \tag{2.13e}$$

The aforementioned auxiliary terms participate in (2.11b) - (2.11d) and can be used to simplify them, as we will demonstrate later. However, in order to streamline the relaxation process, it is necessary to reformulate the definitions in (2.13) as follows:

$$\begin{aligned}
&\sqrt{(\bar{a}-a^2)\left(\bar{\lambda}_2[\tau]-\lambda_2[\tau]^2\right)} = |f_1[\tau]-a\lambda_2[\tau]|,\\
&\sqrt{(\bar{a}-a^2)\left(\bar{\lambda}_1[\tau]-\lambda_1[\tau]^2\right)} = |f_2[\tau]+a\lambda_1[\tau]|,\\
&f_3[\tau]=0 \qquad\qquad \tau=1,2,\ldots,\mathfrak{t}
\end{aligned} \tag{2.14a}$$

$$\begin{aligned}
&\sqrt{\left(\bar{\lambda}_1[\tau]-\lambda_1[\tau]^2\right)\left(\bar{i}_2[\tau]-i_2[\tau]^2\right)} = |w_1[\tau]-\lambda_1[\tau]i_2[\tau]|,\\
&\sqrt{\left(\bar{\lambda}_2[\tau]-\lambda_2[\tau]^2\right)\left(\bar{i}_1[\tau]-i_1[\tau]^2\right)} = |w_2[\tau]-\lambda_2[\tau]i_1[\tau]|,\\
&w_3[\tau]=0 \qquad\qquad \tau=1,2,\ldots,\mathfrak{t}
\end{aligned} \tag{2.14b}$$

$$\begin{aligned}
&\sqrt{\mathrm{diag}\{[\bar{l}_1-l_1^2,\ \bar{l}_2-l_2^2,\ \bar{l}_3-l_3^2]\}(\bar{\boldsymbol{i}}[\tau]-\mathrm{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau])}\\
&\qquad = |\boldsymbol{z}[\tau]-\mathrm{diag}\{[l_1,\ l_2,\ l_3]\}\ \boldsymbol{i}[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t}
\end{aligned} \tag{2.14c}$$

$$\begin{aligned}
&\sqrt{(\bar{r}-r^2)(\bar{\boldsymbol{i}}[\tau]-\mathrm{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau])}\\
&\qquad = |\boldsymbol{h}[\tau]-r\times\boldsymbol{i}[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t}
\end{aligned} \tag{2.14d}$$

$$\begin{aligned}
&\sqrt{(\bar{Q}-Q^2)(\bar{T}_e[\tau]-T_e[\tau]^2)}\\
&\qquad = |\theta[\tau]-Q\times\bar{T}_e[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t}
\end{aligned} \tag{2.14e}$$

where

$$\bar{l}_1 \triangleq l_1^2, \quad \bar{l}_2 \triangleq l_3^2, \quad \bar{l}_3 \triangleq l_3^2, \tag{2.15a}$$

$$\bar{a} \triangleq a^2, \quad \bar{r} \triangleq r^2, \quad \bar{Q} \triangleq Q^2, \tag{2.15b}$$

$$\bar{\boldsymbol{\lambda}}[\tau] \triangleq \text{diag}\{\boldsymbol{\lambda}[\tau]\}\boldsymbol{\lambda}[\tau], \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.15c}$$

$$\bar{\boldsymbol{i}}[\tau] \triangleq \text{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau], \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.15d}$$

$$\bar{T}_e[\tau] \triangleq T_e[\tau]^2. \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.15e}$$

It can be observed that the equations in (2.14) are equivalent to (2.13). In what follows, we will transform all of equalities in (2.14) and (2.15), in order to arrive to a convex relaxation:

$$\begin{aligned} \textbf{minimize } & \sum_{\tau=1}^{\mathfrak{t}} \boldsymbol{\alpha}^\top \left(\bar{\boldsymbol{\lambda}}[\tau] + \text{diag}\{\boldsymbol{\lambda}^{\text{MEC}}[\tau]\}(\boldsymbol{\lambda}^{\text{MEC}}[\tau] - 2\boldsymbol{\lambda}[\tau]) \right) \\ & + \sum_{\tau=1}^{\mathfrak{t}} \boldsymbol{\beta}^\top \left(\bar{\boldsymbol{i}}[\tau] + \text{diag}\{\boldsymbol{i}^{\text{MEC}}[\tau]\}(\boldsymbol{i}^{\text{MEC}}[\tau] - 2\boldsymbol{i}[\tau]) \right) \\ & + \sum_{\tau=1}^{\mathfrak{t}} \gamma \left(\bar{T}_e[\tau] - 2\, T_e^{\text{MEC}}[\tau] T_e[\tau] + T_e^{\text{MEC}}[\tau]^2 \right) \end{aligned} \tag{2.16a}$$

subject to

$$\boldsymbol{\lambda}[\tau+1] = \boldsymbol{\lambda}[\tau] - \boldsymbol{f}[\tau] - \boldsymbol{h}[\tau] + \boldsymbol{v}[\tau], \qquad \tau = 1, 2, \ldots, \mathfrak{t}-1 \tag{2.16b}$$

$$\boldsymbol{\lambda}[\tau] = \boldsymbol{z}[\tau] + [0,\ l_4,\ 0]^\top \times i_{fld}, \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.16c}$$

$$\theta[\tau] = \frac{3}{4}(w_2[\tau] - w_1[\tau]), \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.16d}$$

$$\bar{l}_1 \geq l_1^2, \quad \bar{l}_2 \geq l_3^2, \quad \bar{l}_3 \geq l_3^2, \tag{2.16e}$$

$$\bar{a} \geq a^2, \quad \bar{r} \geq r^2, \quad \bar{Q} \geq Q^2, \tag{2.16f}$$

$$\bar{\boldsymbol{\lambda}}[\tau] \geq \text{diag}\{\boldsymbol{\lambda}[\tau]\}\boldsymbol{\lambda}[\tau], \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.16g}$$

$$\bar{\boldsymbol{i}}[\tau] \geq \text{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau], \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.16h}$$

$$\bar{T}_e[\tau] \geq T_e[\tau]^2 \qquad \tau = 1, 2, \ldots, \mathfrak{t} \tag{2.16i}$$

$$\sqrt{(\bar{a}-a^2)\left(\bar{\lambda}_2[\tau]-\lambda_2[\tau]^2\right)} \geq |f_1[\tau]-a\lambda_2[\tau]|,$$
$$\sqrt{(\bar{a}-a^2)\left(\bar{\lambda}_1[\tau]-\lambda_1[\tau]^2\right)} \geq |f_2[\tau]+a\lambda_1[\tau]|,$$
$$f_3[\tau]=0 \qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.16j}$$

$$\sqrt{\left(\bar{\lambda}_1[\tau]-\lambda_1[\tau]^2\right)\left(\bar{i}_2[\tau]-i_2[\tau]^2\right)} \geq |w_1[\tau]-\lambda_1[\tau]i_2[\tau]|,$$
$$\sqrt{\left(\bar{\lambda}_2[\tau]-\lambda_2[\tau]^2\right)\left(\bar{i}_1[\tau]-i_1[\tau]^2\right)} \geq |w_2[\tau]-\lambda_2[\tau]i_1[\tau]|,$$
$$w_3[\tau]=0 \qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.16k}$$

$$\sqrt{\mathrm{diag}\{[\bar{l}_1-l_1^2,\ \bar{l}_2-l_2^2,\ \bar{l}_3-l_3^2]\}(\bar{\boldsymbol{i}}[\tau]-\mathrm{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau])}$$
$$\geq |\boldsymbol{z}[\tau]-\mathrm{diag}\{[l_1,\ l_2,\ l_3]\}\ \boldsymbol{i}[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.16l}$$
$$\sqrt{(\bar{r}-r^2)(\bar{\boldsymbol{i}}[\tau]-\mathrm{diag}\{\boldsymbol{i}[\tau]\}\boldsymbol{i}[\tau])}$$
$$\geq |\boldsymbol{h}[\tau]-r\times\boldsymbol{i}[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.16m}$$
$$\sqrt{(\bar{Q}-Q^2)(\bar{T}_e[\tau]-T_e[\tau]^2)}$$
$$\geq |\theta[\tau]-Q\times\bar{T}_e[\tau]| \qquad \tau=1,2,\ldots,\mathfrak{t} \tag{2.16n}$$

variables

$$\left\{\boldsymbol{\lambda}[\tau],\boldsymbol{i}[\tau],\bar{\boldsymbol{\lambda}}[\tau],\bar{\boldsymbol{i}}[\tau],\boldsymbol{f}[\tau],\boldsymbol{h}[\tau],\boldsymbol{z}[\tau],\boldsymbol{w}[\tau]\in\mathbb{R}^3\right\}_{\tau=1}^{\mathfrak{t}},$$
$$\left\{T_e[\tau],\bar{T}_e[\tau]\in\mathbb{R}\right\}_{\tau=1}^{\mathfrak{t}},$$
$$a,l_1,l_2,l_3,l_4,r,Q,\bar{a},\bar{l}_1,\bar{l}_2,\bar{l}_3,\bar{r},\bar{Q}\in\mathbb{R}.$$

The non-negative weights in the objective function in (2.16a) are $\boldsymbol{\alpha}=\mathrm{diag}(\boldsymbol{\Lambda}_{\boldsymbol{\alpha}})$ and $\boldsymbol{\beta}=\mathrm{diag}(\boldsymbol{\Lambda}_{\boldsymbol{\beta}})$. Equality constraints in (2.16b) - (2.16d) are the same as those in (2.11b) - (2.11d) which describe the WRSM model equations. The inequalities

in (2.16e) - (2.16k) implicitly impose (2.13) - (2.15) to preserve equivalency to the original problem in (2.11) and the convexified problem in (2.16). It is straightforward to observe that (2.16e) - (2.16k) are convex if formulated as linear matrix inequalities:

$$\begin{bmatrix} \bar{a} & f_k[\tau] \\ f_k[\tau] & \bar{\lambda}_{3-k}[\tau] \end{bmatrix} \succeq \begin{bmatrix} (-1)^{3-k}a \\ \lambda_{3-k}[\tau] \end{bmatrix} \begin{bmatrix} (-1)^{3-k}a \\ \lambda_{3-k}[\tau] \end{bmatrix}^\top \qquad k=1,2 \qquad (2.17a)$$

$$\begin{bmatrix} \bar{\lambda}_k[\tau] & w_k[\tau] \\ w_k[\tau] & \bar{i}_{3-k}[\tau] \end{bmatrix} \succeq \begin{bmatrix} \lambda_k[\tau] \\ i_{3-k}[\tau] \end{bmatrix} \begin{bmatrix} \lambda_k[\tau] \\ i_{3-k}[\tau] \end{bmatrix}^\top \qquad k=1,2 \qquad (2.17b)$$

$$\begin{bmatrix} \bar{l}_k & z_k[\tau] \\ z_k[\tau] & \bar{i}_k[\tau] \end{bmatrix} \succeq \begin{bmatrix} l_k \\ i_k[\tau] \end{bmatrix} \begin{bmatrix} l_k \\ i_k[\tau] \end{bmatrix}^\top \qquad k=1,2,3 \qquad (2.17c)$$

$$\begin{bmatrix} \bar{r} & h_k[\tau] \\ h_k[\tau] & \bar{i}_k[\tau] \end{bmatrix} \succeq \begin{bmatrix} r \\ i_k[\tau] \end{bmatrix} \begin{bmatrix} r \\ i_k[\tau] \end{bmatrix}^\top \qquad k=1,2,3 \qquad (2.17d)$$

$$\begin{bmatrix} \bar{Q} & \theta[\tau] \\ \theta[\tau] & \bar{T}_{e,k}[\tau] \end{bmatrix} \succeq \begin{bmatrix} Q \\ T_{e,k}[\tau] \end{bmatrix} \begin{bmatrix} Q \\ T_{e,k}[\tau] \end{bmatrix}^\top \qquad k=1,2,3 \qquad (2.17e)$$

The optimization problem in (2.16) is a relaxation and its solution may not be feasible for the original problem in (2.11). Nonetheless, the solution of convex relaxation can be used as an initial point for any general-purpose IPM solver.

As an alternative to IPM, [33] proposes a penalization method to find feasible and near-optimal solutions to problems of the form (2.11). This approach is regarded as penalized convex relaxation. Let $\hat{\boldsymbol{x}} = \left(\{\hat{\boldsymbol{\lambda}}[\tau], \hat{\boldsymbol{i}}[\tau], \hat{T}_e[\tau]\}_{\tau=1}^{\mathfrak{t}}, \hat{a}, \hat{l}_1, \hat{l}_2, \hat{l}_3, \hat{r}, \hat{Q}\right)$ be an

optimal solution for the convex problem (2.16). If $\hat{\boldsymbol{x}}$ is not feasible for the original nonconvex problem, one can incorporate a penalty term of the form

$$\begin{aligned}
&\kappa\left(\{\hat{\boldsymbol{\lambda}}[\tau],\hat{\boldsymbol{i}}[\tau],\hat{T}_e[\tau]\}_{\tau=1}^{\mathrm{t}},\hat{a},\hat{l}_1,\hat{l}_2,\hat{l}_3,\hat{r},\hat{Q}\right)\\
&\qquad=\eta_a(\bar{a}-2\hat{a}a+\hat{a}^2)+\eta_r(\bar{r}-2\hat{r}r+\hat{r}^2)\\
&\qquad+\eta_Q(\bar{Q}-2\hat{Q}Q+\hat{Q}^2)+\eta_{l_1}(\bar{l}_1-2\hat{l_1}l_1+\hat{l}_1^2)\\
&\qquad+\eta_{l_2}(\bar{l}_2-2\hat{l}_2l_2+\hat{l}_2^2)+\eta_{l_3}(\bar{l}_3-2\hat{l}_3l_3+\hat{l}_3^2)\\
&\qquad+\eta_\lambda\sum_{\tau=1}^{\mathrm{t}}(\mathbf{1}_3^\top\bar{\boldsymbol{\lambda}}[\tau]-2\hat{\boldsymbol{\lambda}}^\top[\tau]\boldsymbol{\lambda}[\tau]+\hat{\boldsymbol{\lambda}}^\top[\tau]\hat{\boldsymbol{\lambda}}[\tau])\\
&\qquad+\eta_i\sum_{\tau=1}^{\mathrm{t}}(\mathbf{1}_3^\top\bar{\boldsymbol{i}}[\tau]-2\hat{\boldsymbol{i}}^\top[\tau]\boldsymbol{i}[\tau]+\hat{\boldsymbol{i}}^\top[\tau]\hat{\boldsymbol{i}}[\tau])\\
&\qquad+\eta_{T_e}\sum_{\tau=1}^{\mathrm{t}}(\bar{T}_e[\tau]-2\hat{T}_e[\tau]T_e[\tau]+\hat{T}_e^2),
\end{aligned}\tag{2.18}$$

in the objective function of relaxation, and solve additional rounds to obtain fully feasible points with satisfactory objective values. In (2.18), the parameters $\eta_a,\eta_r,\eta_Q,\eta_{l_1},\eta_{l_2},\eta_{l_3},\eta_\lambda,\eta_i,\eta_{T_e}\geq 0$ are user-defined.

2.6 Model Generation and Verification

2.6.1 System Setup

The MEC model of a 2 kW WRSM is adopted from [9]. The machine is constructed using M19 steel laminations, and copper conductors are used for stator and field windings. The optimization experiments are run on a workstation equipped with Intel(R) Core i7-6700 CPU (4 cores) at 3.40 GHz and 32 GB of RAM with Windows 10. The conic optimization problem in (2.16) is solved using the SDPT3 4.0 [34] and MOSEK [35] solver running on CVX [36] in the MATLAB 2017b environment. Local search is implemented using MATPOWER Interior Point Solver (MIPS) [37].

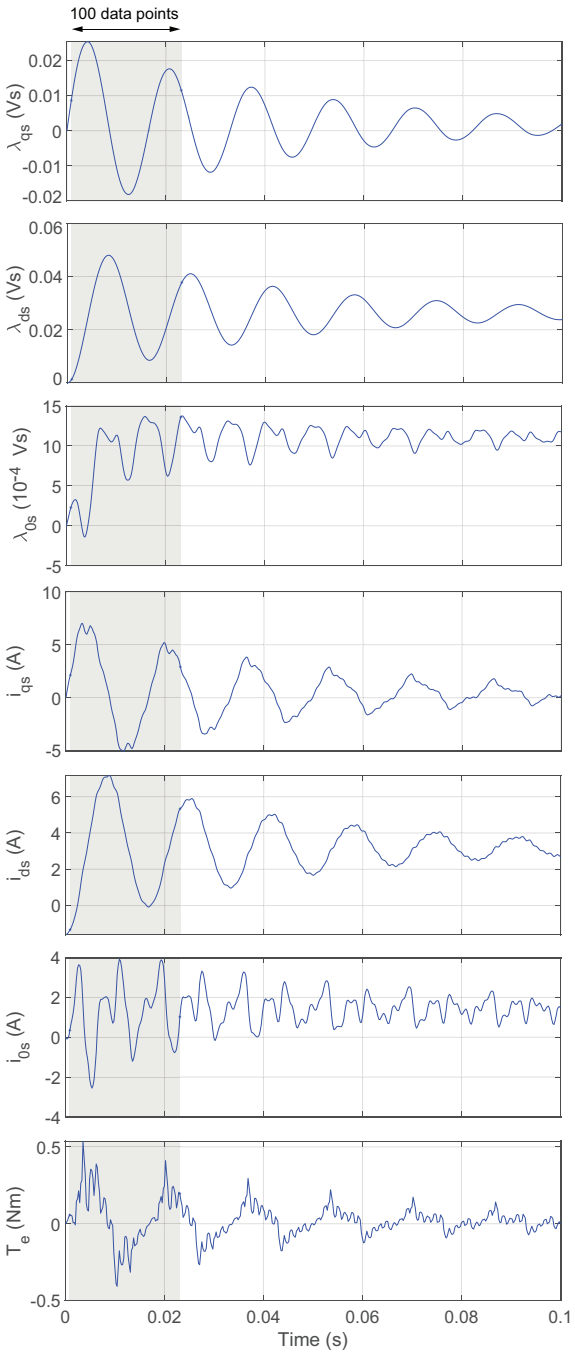


Figure 2: Time-domain transients for flux linkages, currents, and electromagnetic torque. Operating condition is $i_{fld} = 0.25$ A, $\omega_r = 3600$ RPM, $v_{qs} = 10$ V, $v_{ds} = 0$ V, and $v_{0s} = 0.25$ V. The sampling length is $\Delta T = 2.22 \times 10^{-4}$s. Only the highlighted sections of the data are used for model extraction.

2.6.2 Training Data Generated by the Dynamic MEC Model

We have considered an unbalanced operation of the WRSM to generate training data with non-zero zero-sequence flux linkage and current components. Zero initial conditions are assumed for the states. Figure 2 shows the flux linkages, currents, and electromagnetic torque waveforms generated using the MEC model. The highlighted portions of plots in Figure 2, that include 100 data points, are used in parameter extraction. More data points will improve the solution, albeit at the expense of computational time.

2.6.3 Model Extraction Procedure

The optimization problem in Section 2.5.1 is first solved using cone programming relaxation. The result is used as an initial condition for IPM. The tuning gains for the relaxed problem are $\Lambda_{\boldsymbol{\alpha}} = \Lambda_{\boldsymbol{\beta}} = \mathbf{diag}\{1, 1, 0.001\}$ and $\gamma = 1$. A flat initial condition is assumed with all unknown variables set to 1. The length of the time horizon is $\mathfrak{t} = 100$. Conic relaxation solves (2.16) over a convexified version of the feasible set of (2.11). Table 5 shows the relaxation results. The obtained set is not necessarily a feasible point (e.g., see L_{ls}), but can be a good initial point for IPM. In comparison, in all of our experiments, five rounds of penalized relaxation (see (2.18)) resulted in feasible points and parameters similar to those of IPM.

The tuning gains for IPM are $\Lambda_{\boldsymbol{\alpha}} = \mathbf{diag}\{10^4, 10^4, 10^4\}$, $\Lambda_{\boldsymbol{\beta}} = \mathbf{diag}\{0.001, 0.001, 0.001\}$, and $\gamma = 0.1$. Table 2 tabulates the parameters extracted by IPM after solving (2.11). These parameters are compared against those reported in [9] (corresponding to MEC-BH1 model). Reference value of the stator resistance, $r_s = 0.1729\ \Omega$, is directly obtained in the MEC model [9] using equivalent length and area of the windings. The reference value for the effective turns ratio between the field and stator windings, $\frac{N_{fld}}{N_s}$, is taken from that reported in [38]. As seen in Table 2, the percentage

Table 1: Initial parameters obtained using a single run of convex relaxation: MEC-BH1 ($\mathfrak{t} = 100$)

Parameter	MEC [9, 38]	Estimated Parameters	%Mismatch (w.r.t. MEC model)
ω_r*(rad/sec)	376.99	376.98	0.0005
$r_s(\Omega)$	0.1729**	0.1729	0
L_{ls} (mH)	0.83	0.27	67.47
L_{mq} (mH)	3.06	3.38	10.45
L_{md} (mH)	4.71	5.19	10.19
$\frac{N_{fld}}{N_s}$	10.94	10.23	6.50
P	4	4.44***	-

*Electrical angular speed of the rotor. **Directly obtained from the MEC model. ***The number of poles, P, takes even integers.

Table 2: Machine parameters extracted using hybrid cone programming relaxation and IPM: MEC-BH1 ($\mathfrak{t} = 100$)

Parameter	MEC [9, 38]	Hardware [9]	Estimated Parameters	%Mismatch (wrt MEC/Hardware)
ω_r*(rad/sec)	376.99	376.99	377.10	0.02 / 0.02
$r_s(\Omega)$	0.1729**	0.21	0.1738	0.52 / 17.23
L_{ls} (mH)	0.83	0.90	0.75	9.06 / 16.13
L_{mq} (mH)	3.06	3.07	2.94	3.74 / 4.05
L_{md} (mH)	4.71	4.46	4.74	0.66 / 6.30
$\frac{N_{fld}}{N_s}$	10.94	Not available	11.16	2.00 / -
P	4	4	3.90***	- / -

*Electrical angular speed of the rotor. **Directly obtained from the MEC model. ***The number of poles, P, takes even integers.

Table 3: Machine parameters extracted using hybrid cone programming relaxation and IPM: MEC-BH2 ($\mathfrak{t} = 100$)

Parameter	MEC [9, 38]	Hardware [9]	Estimated Parameters	%Mismatch (wrt MEC/Hardware)
ω_r*(rad/sec)	376.99	376.99	377.12	0.03 / 0.03
$r_s(\Omega)$	0.1729**	0.21	0.1737	0.46 / 17.28
L_{ls} (mH)	0.82	0.90	0.74	9.75 / 17.77
L_{mq} (mH)	2.94	3.07	2.84	3.40 / 7.49
L_{md} (mH)	4.33	4.46	4.34	0.23 / 2.69
$\frac{N_{fld}}{N_s}$	10.94	Not available	11.35	3.75 / -
P	4	4	3.89***	- / -

*Electrical angular speed of the rotor. **Directly obtained from the MEC model. ***The number of poles, P, takes even integers.

mismatch for the estimated parameters w.r.t. MEC reference values is highest for L_{ls} at 9.06%, whereas the rest of the parameters are estimated within 3.8% accuracy. Estimation error with respect to the hardware values is obviously higher given the inherent mismatch between the original MEC model and the hardware prototype in [9]. It should be noted that parameters reported for MEC model should be considered for comparative purpose. [9] also proposed a second MEC model of WRSM, named MEC-BH2, to address the discrepancy between the anhysteretic BH curve used in the MEC model and BH curve obtained from the material used in the machine design. For completion, results for parameter extraction on MEC-BH2 model are also provided in Table 3.

In the problem formulation in (2.11) and (2.16), transients of flux linkages, currents, and electromagnetic torque are also considered as optimization variables. The IPM solution recovers these variables along with the machine parameters. Figure 3 compares the time-domain transients of input MEC data and the trajectories obtained by the optimization process for MEC-BH1. The red curves in Figure 3 show the data from the MEC model that is given as an input to the optimization algorithm (the shaded portion of transients in Figure 2). The black curves are transients obtained from the IPM algorithm applied to the optimization problem in (2.11). As the low-order $qd0$ model cannot capture all dynamics generated by the MEC model, the optimization algorithm will minimize the mismatch between the trajectories of the input MEC model and the extracted $qd0$ model.

The leakage inductance, L_{ls}, reported in[9], are obtained using the conventional zero-sequence test as documented in[39]. Alternatively, the leakage inductance value obtained here provides the best fit between the transients produced by MEC and *qd0* models. Figure 4 compares the zero-sequence flux linkage, λ_{0s}, of MEC data against transients produced by two *qd0* models with two different sets of parameters.

Table 4: Computations and Time durations for hybrid cone programming relaxation and IPM

Test Case	**Iterations**		**Running Time**		
	Convex	IPM	Convex(s)	IPM(s)	Total(s)
MEC-BH1 ($\mathfrak{t} = 100$)	38	8	6.01	5.78	11.79
MEC-BH2 ($\mathfrak{t} = 100$)	42	8	6.40	6.02	12.42

The waveform colored green (named '*qd0* [3]') corresponds to the *qd0* model with parameters taken from [9], while the waveform colored black (named '*qd0* optimal') corresponds to the *qd0* model with parameters from Table 2. It is evident from Figure 4 that the machine parameters obtained in this chapter provide a better fit to the MEC data. Mean absolute percentage error (MAPE) [40] can be used to quantify the errors between the curves in Figure 4. MAPE is defined as

$$\text{MAPE} = \frac{100\%}{\mathfrak{t}} \sum_{t=1}^{\mathfrak{t}} \left| \frac{Y_{\text{MEC}}(t) - Y(t)}{Y_{\text{MEC}}(t)} \right|, \tag{2.19}$$

where Y_{MEC} represents the transients from the MEC model and Y is the transients produced by the *qd0* models. The MAPEs for '*qd0* [3]' and '*qd0* optimal' compared to the MEC data for Figure 4 are 40.04% and 37.42%, respectively.

Table 4 provides the iteration counts and times required for both cone programming (CVX with the SDPT3 4.0 solver) and IPM (MIPS solver) methods. Note that the longer time horizon also increases the computation time needed to obtain the optimal solution, as the number of unknown variables (and equality constraints) also increases.

2.6.4 MEC vs *qd*0 Model Comparison

The MEC model of the WRSM had been validated against a hardware prototype [9]. Herein, we reproduce Figure 10a of [9] to compare our *qd*0 model against the

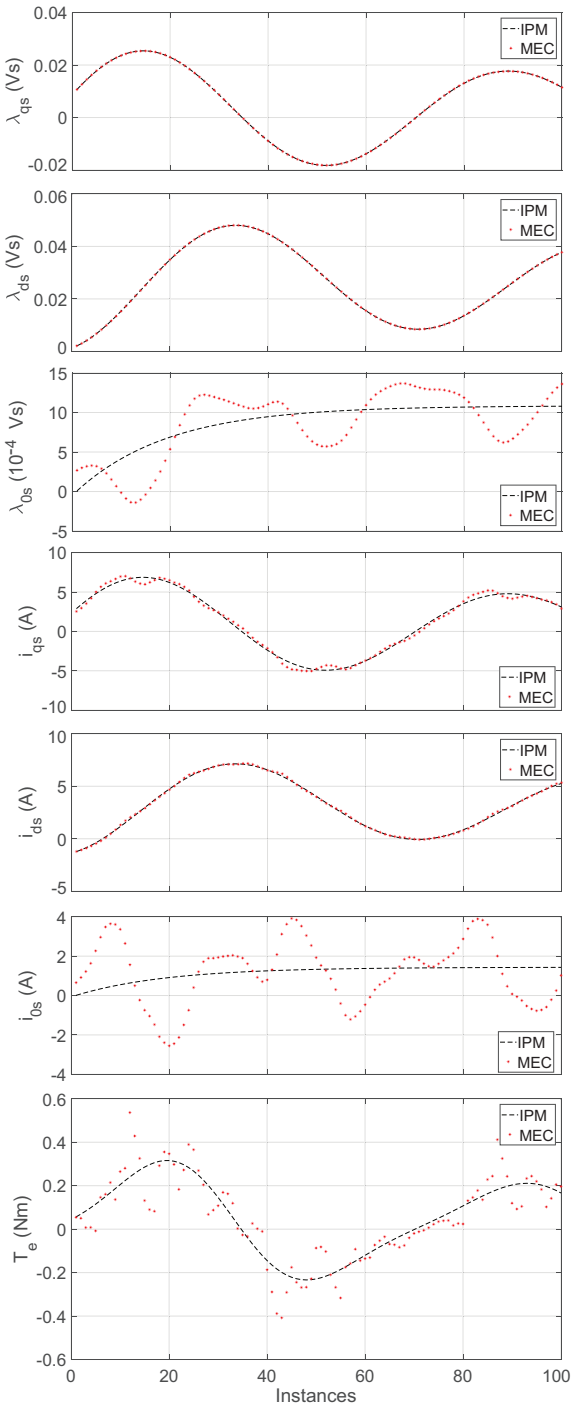


Figure 3: Trajectories obtained from IPM compared with the MEC data for MEC-BH1 using $\mathfrak{t} = 100$. Apart from the machine parameters listed in Table 2, transients of flux linkages (λ_{qs}, λ_{ds}, λ_{0s}), currents (i_{qs}, i_{ds}, i_{0s}), and electromagnetic torque (T_e) are also obtained from the solution to the optimization problem.

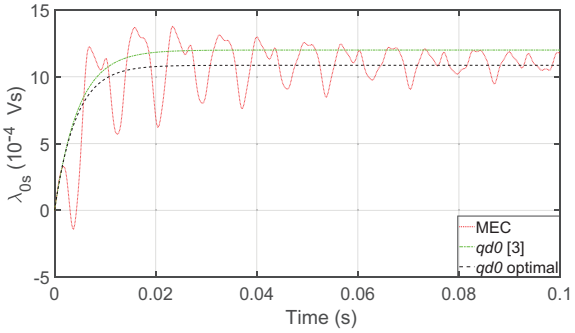


Figure 4: Zero-sequence flux linkage transients from MEC model (red), and two *qd0* models with parameters obtained from [9] and using the proposed method.

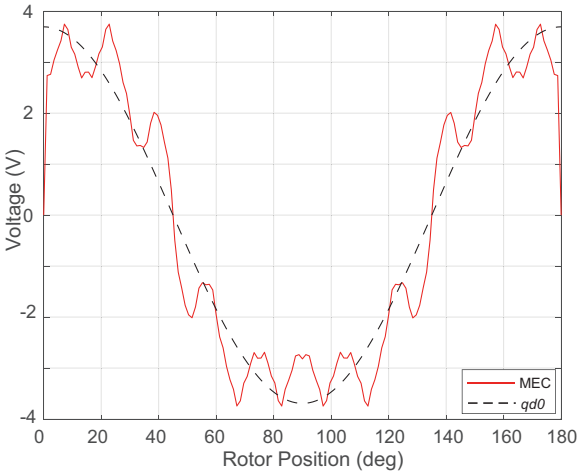


Figure 5: Open-circuit operation of the static MEC model in [9] and the resulting $qd0$ model ($i_{fld} = 1$ A, $\omega_r = 1000$ RPM).

validated MEC model. The WRSM is run under an open circuit with $i_{fld} = 1$ A and $\omega_r = 1000$ RPM. Figure 5 shows the phase-a voltage waveform for the extracted $qd0$ model, and that obtained by the static MEC model. The harmonic effects of the spatial distribution of stator slots are evident in the MEC model. The voltage waveform produced by both models are in agreement.

Next, the $qd0$ model and the dynamic MEC model are simulated with $i_{fld} = 1$ A, $\omega_r = 3600$ RPM, and a balanced load $R_{load} = 20\ \Omega$. The phase-a transients, as seen in Figure 6, show that the resulting $qd0$ model mimic the essential dynamics of

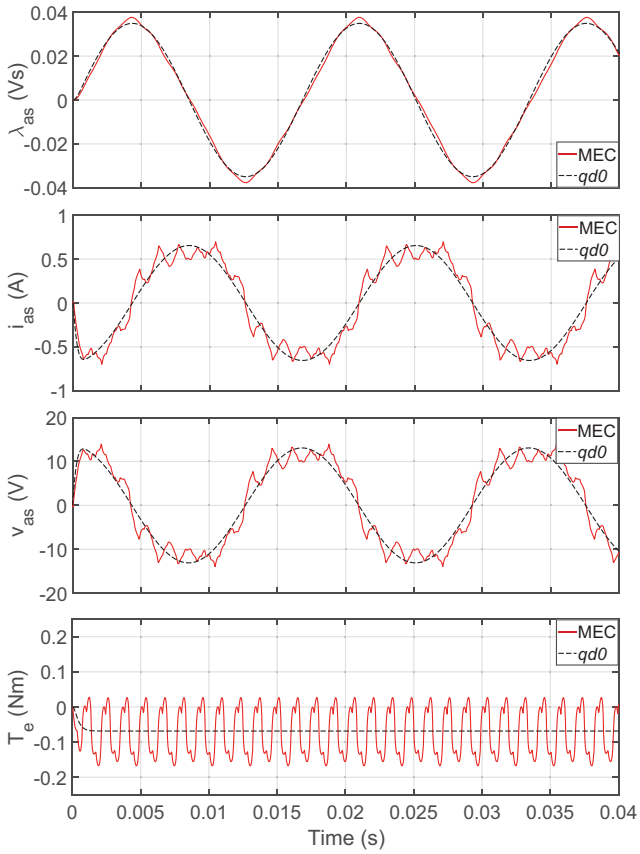


Figure 6: Flux linkage, current, voltage, and electric torque for the extracted $qd0$ model compared against the MEC model, when connected to a balanced load ($i_{fld} = 1$ A, $\omega_r = 3600$ RPM, $R_{load} = 20\ \Omega$).

the MEC model. The physical time for a 5-cycle (83.3 ms) simulation run for the MEC and $qd0$ models are 0.8362 s and 0.0377 s, respectively. The $qd0$ model is more than 20 times faster than the dynamic MEC model.

2.7 Summary

The macromodel of a 2 kW WRSM is successfully extracted from its dynamic MEC model. The parameter extraction process is formulated as an optimization problem; This problem is first covexified using a cone programming relaxation method and, then, the resulting solution is used to initialize the IPM solver. We have successfully

extracted all the machine parameters within 4% accuracy with respect to the original MEC model; The leakage inductance, L_{ls}, was estimated within 10% accuracy. The extracted $qd0$ model is compared against MEC model, and exhibits acceptable fidelity with an appreciable gain in the simulation speed. Future work could include model extraction for a more general *qd0* model that accounts for spatial harmonics. This would be a compromise between the MEC model and the classical *qd0* model with constant inductance terms. We envision that adding any general form of *qd0* model would change the equality constraints of the optimization problem. To accommodate non-sinusoidal winding distributions, the flux linkage-current relation in (2.6) will include rotor-position-dependent inductance terms. Therefore, the equality constraints in (2.11c) will be updated to include expressions that are products of time-varying inductances and winding currents (instead of time-invariant inductances and currents). Given the circumferential motion of the rotor, additional equality constraints would be needed to address the periodicity of inductances.

INDUCTION MACHINE PARAMETERIZATION FROM LIMITED TRANSIENT DATA USING CONVEX OPTIMIZATION[1]

Authors: A. P. Yadav, R. Madani, N. Amiri, J. Jatskevich, and A. Davoudi.

Reprinted, with permission from all the co-authors.

Journal: IEEE Transactions on Industrial Electronics [41].

DOI: 10.1109/TIE.2021.3060668

CHAPTER 3

INDUCTION MACHINE PARAMETERIZATION FROM LIMITED TRANSIENT DATA USING CONVEX OPTIMIZATION

Abstract

This paper identifies the parameters of an induction machine using limited and non-intrusive observations of available input voltages, stator currents, and the rotor speed. Parameter extraction is formulated as a non-convex estimation problem, which is then relaxed to a convex conic optimization problem. While the resulting relaxation could exhibit a satisfactory performance, there might be cases where the solution of convex relaxation fails to satisfy the dynamic equations of the machine. This is remedied through a local search approach initiated using the solution obtained from the relaxed problem. The proposed method is experimentally verified on a squirrel-cage induction machine with missing measured data. Using the measured signals as the benchmark, time-domain transients produced by the parameters estimated using the proposed method show almost 20% better match compared to time-domain transients produced by the parameters obtained via conventional testing.

3.1 Introduction

Accurate machine characterization is needed for drive design and control, diagnostics and condition monitoring, controller/hardware-in-the-loop applications. Given that induction machines constitute a significant portion of loads in the grid, proper

machine characterization is crucial to analysis of power system dynamics [42]. Mismatch between the actual and estimated parameter sets can deteriorate the drive performance [43]. Reliable data for most machines are not accessible, and excessive testing may not always be practical. Informative reviews on parameter identification of induction machines are presented in [44] and [27]. Conventionally, estimating parameters involves intrusive testing, e.g., IEEE Std. 112[45]. One popular approach is to excite the machine with predetermined signals and monitor its response while maintaining a standstill rotor[46, 47, 48], which is suitable for 'self-commissioning' [49, 50]. In general, intrusive testings require isolated access to the machine, additional measurement equipment, and interruption of machine operation which might not be always feasible. For example, the locked-rotor test draws in large currents and could become impractical for some industrial setups. It is desirable to extract machine parameters from (preferably a single) transients during normal operation [51]. For example, [52] utilizes different portions of current and voltage transients to approximate conventional test scenarios.

The main parameters of interest for an induction machine are stator and rotor resistances, magnetizing inductance, stator and rotor leakage inductances, and mechanical inertia. Various methodologies exist for non-intrusive parameterization of induction machines, e.g., observer-based estimators [53, 54] or least-square regression [55, 26]. Observer-based methods, such as Kalman filters, can estimate system states and a subset of machine parameters using measured signals from the machine terminals. However, Kalman filters require proper initialization and noise covariance matrices [56]. [55] reformulates the machine model in terms of K-parameters, assuming slow-varying rotor speed, resulting in a standard linear least-square regression problem. [26] and [57] further extend this work to incorporate time-varying speed into the final regression problem. However, this involves estimating first- and second-

order derivatives for certain current and flux-linkage terms, and are susceptible to noisy measurements. Usually, all these methods perform estimations using measurements of stator currents, input voltages, and rotor speed. [58] estimated machine parameters using only stator currents and voltages. This could, potentially, result in an ill-conditioned problem which would require an estimate for speed trajectory or an excellent initial guess, or could only offer a subset of parameters. Equivalent circuit model of an induction machine could be found using geometrical and electrical data [59, 60]. [61] employed finite-element model of an induction machine to extract its equivalent circuit model. [7] obtained the machine parameters from a high-fidelity magnetic-equivalent circuit model. Such methods require expert knowledge on the underlying complex models, manufacturing/fabrication errors, or material defects, and inherent the approximation present in the primary modeling effort.

One could employ nonlinear constraint optimization [62], [63] to minimize an objective function (usually, the norm of error between measured and predicted outputs) subject to machine model equations. A major challenge is the inherent non-convexity of the resulting optimization problem. Newton's method might not correctly converge without proper initialization. Various workarounds to tackle this limitation include (1) use of good initial conditions (from self-commissioning [50],[64] or conventional tests), (2) employing heuristics, e.g, enforcing box constraints on machine parameters[65], or (3) executing multiple optimization runs from different initial points [63]. Metaheuristic optimization techniques, e.g., genetic algorithms, can circumvent non-convexity albeit at a higher computational cost [66], [67, 68, 69]. In the context of power system estimation, [31] proposes a convex optimization approach to offer a good initial condition for the follow-up Newton's method. [7] has extracted parameters from magnetic equivalent circuit model of a synchronous machine using

conic relaxation, assuming availability of all inputs and states, which is not a valid assumption for a physical machine.

We leverage the convex optimization framework to parameterize an induction machine using only limited samples of measurable signals. Herein, we assume a no-load operation and use data from start-up transients. The problem of non-convexity is tackled by formulating it in a higher-dimensional space and imposing conic constraints. Unlike original equations, the relaxed formulation can be solved efficiently using off-the-shelf solvers. To properly enforce machine dynamics, we feed the outcome of convex relaxation to a local search algorithm to obtain the desired near-optimal solution. Figure 7 provides an overview of the proposed approach, with its salient features summarized as follows:

- This method is non-intrusive; Parameters are identified using only limited samples of start-up transients.
- The proposed method does not require *a priori* knowledge of most machine parameters, which makes it suitable for refurbished or re-wounded machines.
- Machine parameters, including stator and rotor resistances, stator and rotor leakage inductances, magnetizing inductance, mechanical inertia, and the friction coefficient, are simultaneously identified.
- The proposed method reformulates a non-convex optimization problem into a tractable convex approximation. Detailed treatment of this transformation is provided. A penalized improvement of this convex relaxation is also discussed and verified using a test example.
- The proposed method is experimentally verified for an induction motor prototype, and is shown to converge even with missing points in available signals. Convergence is achieved with 80% of the measurement data. Robustness of the proposed method to noisy data is discussed.

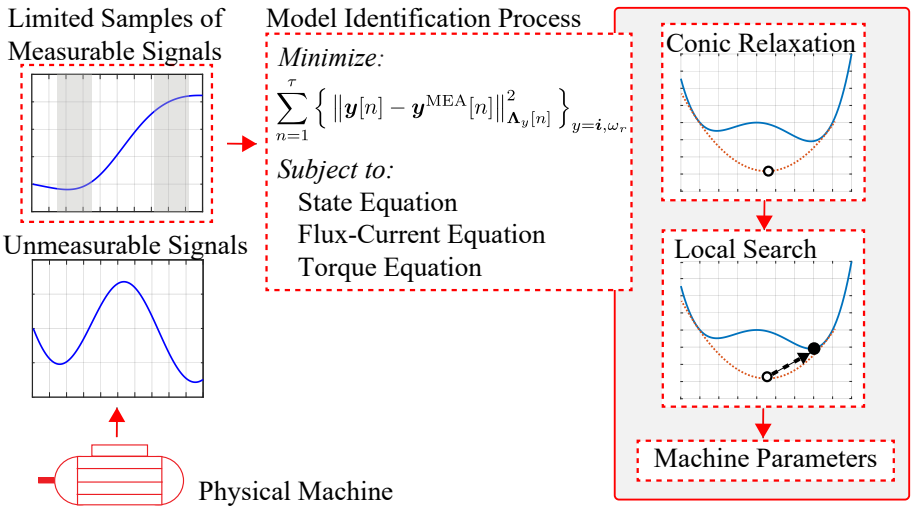


Figure 7: Overview of the proposed model identification method.

- The set of machine parameters extracted by the proposed method are shown to result in a better match with the measured transients compared to the parameter set obtained by the conventional methods. Particularly, the proposed method has resulted in 12.5%-30% reduction in error when matching the stator currents, and 8.7% improvement when matching with the rotor speed transients.

3.2 Notations

Vectors and matrices are represented using bold lowercase and uppercase variables, respectively (e.g., $\boldsymbol{y}$ and $\boldsymbol{Y}$). The j^{th} element of vector $\boldsymbol{y}$ is y_j. $\boldsymbol{I}_n$ denotes a size n identity matrix. The notation $\text{diag}\{\boldsymbol{y}\}$ represents a diagonal matrix with the vector $\boldsymbol{y}$ forming the diagonal. For an $n \times n$ symmetric positive-definite matrix $\boldsymbol{Z}$ and the vector $\boldsymbol{y} \in \mathbb{R}^n$, the norm notation $\|\boldsymbol{y}\|_{\boldsymbol{Z}}$ denotes $\sqrt{\boldsymbol{y}^\top \boldsymbol{Z}\boldsymbol{y}}$. $\otimes$ stands for the Kronecker product. Symbol $\check{\cdot}$ represents the vectorization operator, i.e., $\check{\boldsymbol{\lambda}} \triangleq \left[\boldsymbol{\lambda}[1]^\top, \boldsymbol{\lambda}[2]^\top, \boldsymbol{\lambda}[3]^\top \ldots\right]^\top$. Set $\mathcal{A}$ is convex if, for every $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{A}$ and any $\rho \in [0,1]$, $\rho\boldsymbol{y}_1 + (1-\rho)\boldsymbol{y}_2 \in \mathcal{A}$ [70].

3.3 Discrete-time Model of an Induction Machine

We present the classic dynamic model of an induction machine, and then adopt its discrete-time representation.

3.3.1 Machine Model

The induction machine model, in the arbitrary reference frame, is given by [4]

$$\frac{d\boldsymbol{\lambda}(t)}{dt} = \omega\Big[-\lambda_{ds}(t), \lambda_{qs}(t), -\lambda_{dr}(t), \lambda_{qr}(t)\Big]^\top + \omega_r(t)\Big[0, 0, \lambda_{dr}(t), -\lambda_{qr}(t)\Big]^\top - \boldsymbol{R}\boldsymbol{i}(t) + \boldsymbol{v}(t), \tag{3.1a}$$

$$\boldsymbol{\lambda}(t) = \boldsymbol{L}\boldsymbol{i}(t), \tag{3.1b}$$

where $\boldsymbol{\lambda}(t)$, $\boldsymbol{i}(t)$, and $\boldsymbol{v}(t)$ are the vectors of flux linkages, currents, and voltages, respectively, defined as

$$\boldsymbol{\lambda}(t) \triangleq [\lambda_{qs}(t), \lambda_{ds}(t), \lambda_{qr}(t), \lambda_{dr}(t)]^\top, \tag{3.2a}$$

$$\boldsymbol{i}(t) \triangleq [i_{qs}(t), i_{ds}(t), i_{qr}(t), i_{dr}(t)]^\top, \tag{3.2b}$$

$$\boldsymbol{v}(t) \triangleq [v_{qs}(t), v_{ds}(t), v_{qr}(t), v_{dr}(t)]^\top. \tag{3.2c}$$

Subscripts qs, ds, qr, and dr denote q-axis stator, d-axis stator, q-axis rotor, and d-axis rotor terms, respectively. Zero-sequence terms are neglected in this balanced representation. $\boldsymbol{R} = \text{diag}\,\{[r_s, r_s, r_r, r_r]\}$ and $\boldsymbol{L}$ is

$$\boldsymbol{L} = \begin{bmatrix} L_s & 0 & L_m & 0 \\ 0 & L_s & 0 & L_m \\ L_m & 0 & L_r & 0 \\ 0 & L_m & 0 & L_r \end{bmatrix}. \tag{3.3}$$

ω represents the speed of the chosen reference frame, $\omega_r(t)$ is the rotor speed, r_s is the stator resistance, r_r is the rotor resistance, L_s is the stator self-inductance, L_r is the rotor self-inductance, and L_m is the magnetizing inductance. Stator and rotor leakage inductances can be obtained as $L_{ls} = L_s - L_m$ and $L_{lr} = L_r - L_m$, respectively. We assume an induction machine with shorted rotor bars, i.e., $v_{qr} = v_{dr} = 0$.

The dynamics of the mechanical subsystem is [4]

$$\frac{d\omega_r(t)}{dt} = \frac{P}{2J}\left(T_e(t) - T_m(t)\right) \ , \tag{3.4}$$

where P is the number of poles, J is the lumped mechanical inertia, T_e is the electromagnetic torque, and T_m is the mechanical (load) torque. In this chapter, we consider the start-up transient of an induction machine under free acceleration, where only friction torque is present. The equivalent friction torque can be found by subtracting the machine loss from the input power at the steady state, no-load operation. For simplicity, we assume that the friction coefficient has a linear relation with the rotor mechanical speed. The load torque is

$$T_m(t) = B\omega_r(t)/(P/2). \tag{3.5}$$

B is the total effective friction coefficient [51]. Electromagnetic torque is

$$T_e(t) = \frac{3}{4}P\left(\lambda_{ds}(t)i_{qs}(t) - \lambda_{qs}(t)i_{ds}(t)\right). \tag{3.6}$$

3.3.2 Discrete-time Representation

The machine model is discretized using the forward Euler method as it results in a simple explicit equations which eases the derivation of upcoming relaxation formulations. The discrete-time representation of (3.1), (3.4), and (3.6) become

$$\begin{aligned}\boldsymbol{\lambda}[n+1] = \boldsymbol{\lambda}[n] + \Delta T\Big(\omega\Big[-\lambda_{ds}[n], \lambda_{qs}[n], -\lambda_{dr}[n], \lambda_{qr}[n]\Big]^\top \\ + \omega_r[n]\Big[0, 0, \lambda_{dr}[n], -\lambda_{qr}[n]\Big]^\top - \boldsymbol{R}\boldsymbol{i}[n] + \boldsymbol{v}[n]\Big),\end{aligned} \tag{3.7a}$$

$$Q_w\omega_r[n+1] = Q_w\omega_r[n] + \frac{\Delta T}{2}\left(T_e[n] - \frac{2B\omega_r[n]}{P}\right), \tag{3.7b}$$

$$\boldsymbol{\lambda}[n] = \boldsymbol{L}\boldsymbol{i}[n], \tag{3.7c}$$

$$T_e[n] = \frac{3}{4}P\boldsymbol{\lambda}^\top[n]\Big[-i_{ds}[n], i_{qs}[n], 0, 0\Big]^\top. \tag{3.7d}$$

$n \in \mathcal{T}$ represents a time horizon with $\mathcal{T} \triangleq \{1, 2, 3, \ldots, \tau\}$, and ΔT is the sampling time interval. The variable $Q_w \triangleq J/P$ is defined to ensure that (3.7b) remains of degree two (quadratic), which will be helpful in the upcoming convex relaxation formulations. Note that P is a known constant.

3.4 Parameter Extraction Procedure

Let $\boldsymbol{i}^{\text{MEA}}[n]$ and $\omega_r^{\text{MEA}}[n]$ denote the values of measured currents and rotor speeds, respectively, for $n \in \mathcal{T}$. Let $\mathcal{S}$ denote the set of different discrete-time horizons such as $\mathcal{T}$. The parameter identification problem is formulated as a weighted least-square optimization that minimizes the mismatch between predicted state variables in the discrete-time model (3.7) and the measured signals over $\mathcal{S}$,

minimize

$$\sum_{n\in\mathcal{S}}\Big\|\text{diag}\{\boldsymbol{\iota}[n]\}\big(\boldsymbol{i}[n] - \boldsymbol{i}^{\text{MEA}}[n]\big)\Big\|_{\boldsymbol{\Lambda}}^2 + \gamma\iota_\omega[n]\big(\omega_r[n] - \omega_r^{\text{MEA}}[n]\big)^2 \tag{3.8a}$$

subject to

$$\boldsymbol{\lambda}[n+1]=\boldsymbol{\lambda}[n]+\Delta T\Big(\omega\Big[-\lambda_{ds}[n],\lambda_{qs}[n],-\lambda_{dr}[n],\lambda_{qr}[n]\Big]^{\top}+$$

$$\omega_r[n]\Big[0,0,\lambda_{dr}[n],-\lambda_{qr}[n]\Big]^{\top}-\text{diag}\{[r_s,r_s,r_r,r_r]\}\,\boldsymbol{i}[n]+\boldsymbol{v}[n]\Big), \tag{3.8b}$$

$$Q_w\omega_r[n+1]=Q_w\omega_r[n]+\frac{\Delta T}{2}\left(T_e[n]-\frac{2B\omega_r[n]}{P}\right), \tag{3.8c}$$

$$\boldsymbol{\lambda}[n]=L_m\Big[i_{qr}[n],i_{dr}[n],i_{qs}[n],i_{ds}[n]\Big]^{\top}$$

$$+\,\text{diag}\,\{[L_s,L_s,L_r,L_r]\}\,\boldsymbol{i}[n], \tag{3.8d}$$

$$T_e[n]=\frac{3}{4}P\boldsymbol{\lambda}^{\top}[n]\Big[-i_{ds}[n],i_{qs}[n],0,0\Big]^{\top}, \tag{3.8e}$$

variables

$$\big\{\boldsymbol{\lambda}[n],\boldsymbol{i}[n]\in\mathbb{R}^4,T_e[n],\omega_r[n]\in\mathbb{R}\big\}_{n\in\mathcal{S}},$$

$$r_s,r_r,L_s,L_r,L_m,Q_w,B\in\mathbb{R}.$$

$\boldsymbol{\Lambda}=\text{diag}\{[\alpha_1,\alpha_2,\alpha_3,\alpha_4]\}$ and γ contain non-negative weights to normalize current and speed terms, respectively. $\boldsymbol{\iota}[n]\in\{0,1\}^4$ and $\iota_w[n]\in\{0,1\}$ represent binary flags indicating the availability of the n^{th} data sample. Since rotor-side currents are hard to measure, it is reasonable to assume that $\iota_3[n]=\iota_4[n]=0$ (corresponding to the rotor currents), and $\iota_1[n]=\iota_2[n]=1$ (corresponding to the stator currents). Similarly, $\iota_w[n]$ is 1 or 0 depending upon the availability of speed measurement. The objective function (3.8a) represents the mismatch between the measured signals and the transients predicted by the estimated parameter set. The objective function denotes the sum of squared residuals (same as a least-squares regression). The equality constraints (3.8b) – (3.8e) reflect the discrete-time machine model (3.7a) – (3.7d). The induction machine model presented in (3.1), (3.4), and (3.6) are reflected within the optimization formulation. The unknown variables in the optimization problem (3.8) are separately listed below the constraints for convenience. The optimization problem (3.8a) – (3.8e) solves for flux linkages $\boldsymbol{\lambda}$, currents $\boldsymbol{i}$, torque T_e, and speed

ω_r in $\mathcal{S}$, as well as the parameters $(r_s, r_r, L_s, L_r, L_m, Q_w, B)$, while minimizing the objective function, subject to machine dynamics. Measured data for stator currents (i_{qs} and i_{ds}), input voltage ($\boldsymbol{v}$), and rotor speed (ω_r), and the number of poles, P, are assumed known.

Observe that due to the the absence of rotor-side measurements, the estimation problem (3.8a) – (3.8e) suffers from solution ambiguity [71]. This is obvious from rotor-side flux linkage and current relations (see (3.7c)), where the expression $L_r i_{qr}$ (or $L_r i_{dr}$) can take identical values for different L_r and i_{qr} (or i_{dr}) in absence of rotor-side current measurements. To resolve this, we assume that the ratio L_s/L_r is known. It should be noted that if the NEMA design letter is known for the machine, the ratio L_s/L_r can be obtained from IEEE Std 112 [45, 72]. Inspired by [55, 57, 26], we assume $L_s/L_r = 1$. Note that any other known L_s/L_r ratio would would not affect the optimization process.

The optimization problem (3.8a) – (3.8e) is non-convex because of the following bilinear terms:

- $\omega_r\lambda_{qr}$, $\omega_r\lambda_{dr}$ and $r_s i_{qs}$, $r_s i_{ds}$, $r_r i_{qr}$, $r_r i_{dr}$ in (3.8b);
- $Q_w\omega_r$ and $B\omega_r$ in (3.8c);
- $L_s i_{qs}$, $L_s i_{ds}$, $L_r i_{qr}$, $L_r i_{dr}$ and $L_m i_{qr}$, $L_m i_{dr}$, $L_m i_{qs}$, $L_m i_{ds}$ in (3.8d);
- $\lambda_{qs} i_{ds}$ and $\lambda_{ds} i_{qs}$ in (3.8e).

Problem of non-convexity makes the optimization problem hard to solve, and standard tools, such as Newton's method, might not converge to the right solution without good initialization. In the following section, we transform this problem into a convex optimization formulation which could be solved in polynomial time using off-the-shelf solvers [70, 36].

3.5 Conic Relaxation and Numerical Search

To remedy the presence of non-convex bilinear terms, we introduce additional variables (lifting) and employ conic relaxation to derive a convex optimization formulation for the problem (3.8a) – (3.8e).

3.5.1 Lifting

Non-convexity due to nonlinear terms (e.g., $\omega_r \lambda_{dr}$ and $r_s i_{qs}$) can be addressed by variable substitution. As a result, the objective function (3.8a) and constraints (3.8b) – (3.8e) can be rewritten as linear (convex) functions of variables in (3.8a) – (3.8e) and newly-defined auxiliary variables. This process is known as *lifting* in which the original optimization problem is cast into a higher dimensional space and the entire non-convexity is captured in the definition of auxiliary variables.

For every $n \in \mathcal{S}$, define the following additional variables

$$\boldsymbol{f}[n] \triangleq \omega_r[n] \left[0, 0, \lambda_{dr}[n], -\lambda_{qr}[n]\right]^\top, \tag{3.9a}$$

$$\boldsymbol{g}[n] \triangleq \operatorname{diag}\left\{[r_s, r_s, r_r, r_r]\right\} \boldsymbol{i}[n], \tag{3.9b}$$

$$\boldsymbol{h}[n] \triangleq L_s \boldsymbol{i}[n], \tag{3.9c}$$

$$\boldsymbol{z}[n] \triangleq L_m \left[i_{qr}[n], i_{dr}[n], i_{qs}[n], i_{ds}[n]\right]^\top, \tag{3.9d}$$

$$\boldsymbol{y}[n] \triangleq \left[\lambda_{qs}[n] i_{ds}[n], \lambda_{ds}[n] i_{qs}[n], 0, 0\right]^\top, \tag{3.9e}$$

$$\theta[n] \triangleq Q_w \omega_r[n], \qquad \phi[n] \triangleq B \omega_r[n], \tag{3.9f}$$

$$\bar{\boldsymbol{\lambda}}[n] \triangleq \operatorname{diag}\{\boldsymbol{\lambda}[n]\}\boldsymbol{\lambda}[n], \qquad \bar{\boldsymbol{i}}[n] \triangleq \operatorname{diag}\{\boldsymbol{i}[n]\}\boldsymbol{i}[n], \tag{3.9g}$$

$$\bar{\omega}_r[n] \triangleq \omega_r^2[n], \qquad \bar{T}_e[n] \triangleq T_e^2[n]. \tag{3.9h}$$

Define

$$\bar{r}_s \triangleq r_s^2, \qquad \bar{r}_r \triangleq r_r^2, \quad \bar{Q}_w \triangleq Q_w^2, \tag{3.9i}$$

$$\bar{L}_m \triangleq L_m^2, \quad \bar{L}_s \triangleq L_s^2, \qquad \bar{B} \triangleq B^2. \tag{3.9j}$$

There are two new sets of variables in above formulation: Those like $\boldsymbol{f}[n]$ that represent the non-convex terms, and those like $\bar{r}_s$ that denote squared variable. The need for such formulations will become clear in what follows. The optimization problem (3.8a) – (3.8e) can now be reformulated in terms of the auxiliary variables (3.9a) – (3.9j). However, additional constraints need to be included in the optimization problem to account for (3.9a) – (3.9j). A standard approach in convex optimization to represent bilinear expressions is using matrix equalities [7]. For example, $g_1[n] = r_s i_{qs}[n]$ in (3.9b) can be enforced as

$$\begin{bmatrix} \bar{r}_s & g_1[n] \\ g_1[n] & \bar{i}_1[n] \end{bmatrix} = \begin{bmatrix} r_s \\ i_{qs}[n] \end{bmatrix} \begin{bmatrix} r_s \\ i_{qs}[n] \end{bmatrix}^\top. \tag{3.10}$$

Expressing (3.9a) – (3.9j) as matrix equalities is helpful as they can be easily convexified. Variables such as $\bar{r}_s$ that denote squared variable are used to enforce equality conditions. Hence, problem (3.8a) – (3.8e) can now be reformulated as

minimize

$$\sum_{n\in\mathcal{S}} \boldsymbol{\iota}^\top[n]\boldsymbol{\Lambda}\left(\bar{\boldsymbol{i}}[n] + \mathrm{diag}\{\boldsymbol{i}^{\mathrm{MEA}}[n]\}(\boldsymbol{i}^{\mathrm{MEA}}[n] - 2\boldsymbol{i}[n])\right)$$
$$+ \gamma\iota_w[n]\left(\bar{\omega}_r[n] - 2\,\omega_r^{\mathrm{MEA}}[n]\omega_r[n] + \omega_r^{\mathrm{MEA}}[n]^2\right) \tag{3.11a}$$

subject to

$$\boldsymbol{\lambda}[n+1] = \boldsymbol{\lambda}[n] + \Delta T\Big(\omega\Big[-\lambda_{ds}[n], \lambda_{qs}[n], -\lambda_{dr}[n], \lambda_{qr}[n]\Big]^\top$$

$$+ \boldsymbol{f}[n] - \boldsymbol{g}[n] + \boldsymbol{v}[n]\Big), \tag{3.11b}$$

$$\theta[n+1] = \theta[n] + \frac{\Delta T}{2}\Big(T_e[n] - \frac{2\phi[n]}{P}\Big), \tag{3.11c}$$

$$\boldsymbol{\lambda}[n] = \boldsymbol{z}[n] + \boldsymbol{h}[n], \tag{3.11d}$$

$$T_e[n] = \frac{3P}{4}(y_2[n] - y_1[n]), \tag{3.11e}$$

$$\begin{bmatrix} \bar{\omega}_r[n] & (-1)^k f_k[n] \\ (-1)^k f_k[n] & \bar{\lambda}_{7-k}[n] \end{bmatrix} = \begin{bmatrix} -\omega_r[n] \\ \lambda_{7-k}[n] \end{bmatrix} \begin{bmatrix} -\omega_r[n] \\ \lambda_{7-k}[n] \end{bmatrix}^\top,$$

$$f_{5-k}[n] = 0, \qquad k = 3, 4. \tag{3.11f}$$

$$\begin{bmatrix} \bar{r}_s & g_k[n] \\ g_k[n] & \bar{i}_k[n] \end{bmatrix} = \begin{bmatrix} r_s \\ i_k[n] \end{bmatrix} \begin{bmatrix} r_s \\ i_k[n] \end{bmatrix}^\top, \qquad k = 1, 2. \tag{3.11g}$$

$$\begin{bmatrix} \bar{r}_r & g_k[n] \\ g_k[n] & \bar{i}_k[n] \end{bmatrix} = \begin{bmatrix} r_r \\ i_k[n] \end{bmatrix} \begin{bmatrix} r_r \\ i_k[n] \end{bmatrix}^\top, \qquad k = 3, 4. \tag{3.11h}$$

$$\begin{bmatrix} \bar{L}_s & h_k[n] \\ h_k[n] & \bar{i}_k[n] \end{bmatrix} = \begin{bmatrix} L_s \\ i_k[n] \end{bmatrix} \begin{bmatrix} L_s \\ i_k[n] \end{bmatrix}^\top, \qquad k = 1, 2, 3, 4. \tag{3.11i}$$

$$\begin{bmatrix} \bar{L}_m & z_k[n] \\ z_k[n] & \bar{i}_{k+2}[n] \end{bmatrix} = \begin{bmatrix} L_m \\ i_{k+2}[n] \end{bmatrix} \begin{bmatrix} L_m \\ i_{k+2}[n] \end{bmatrix}^\top, \qquad k = 1, 2. \tag{3.11j}$$

$$\begin{bmatrix} \bar{L}_m & z_k[n] \\ z_k[n] & \bar{i}_{k-2}[n] \end{bmatrix} = \begin{bmatrix} L_m \\ i_{k-2}[n] \end{bmatrix} \begin{bmatrix} L_m \\ i_{k-2}[n] \end{bmatrix}^\top, \qquad k = 3, 4. \tag{3.11k}$$

$$\begin{bmatrix} \bar{\lambda}_k[n] & y_k[n] \\ y_k[n] & \bar{i}_{3-k}[n] \end{bmatrix} = \begin{bmatrix} \lambda_k[n] \\ i_{3-k}[n] \end{bmatrix} \begin{bmatrix} \lambda_k[n] \\ i_{3-k}[n] \end{bmatrix}^\top,$$

$$y_{2+k}[n] = 0, \qquad k = 1, 2. \tag{3.11l}$$

$$\begin{bmatrix} \bar{Q}_w & \theta[n] \\ \theta[n] & \bar{\omega}_r[n] \end{bmatrix} = \begin{bmatrix} Q_w \\ \omega_r[n] \end{bmatrix} \begin{bmatrix} Q_w \\ \omega_r[n] \end{bmatrix}^\top,$$

$$\begin{bmatrix} \bar{B} & \phi[n] \\ \phi[n] & \bar{\omega}_r[n] \end{bmatrix} = \begin{bmatrix} B \\ \omega_r[n] \end{bmatrix} \begin{bmatrix} B \\ \omega_r[n] \end{bmatrix}^\top, \tag{3.11m}$$

variables

$$\left\{\boldsymbol{\lambda}[n], \bar{\boldsymbol{\lambda}}[n], \boldsymbol{i}[n], \bar{\boldsymbol{i}}[n], \boldsymbol{f}[n], \boldsymbol{g}[n], \boldsymbol{h}[n], \boldsymbol{z}[n], \boldsymbol{y}[n] \in \mathbb{R}^4\right\}_{n \in \mathcal{S}},$$

$$\left\{T_e[n], \bar{T}_e[n], \omega_r[n], \bar{\omega}_r[n], \theta[n], \phi[n] \in \mathbb{R}\right\}_{n \in \mathcal{S}},$$

$$r_s, \bar{r}_s, r_r, \bar{r}_r, L_s, \bar{L}_s, L_m, \bar{L}_m, Q_w, \bar{Q}_w, B, \bar{B} \in \mathbb{R}.$$

The optimization problem (3.11a) – (3.11m) is equivalent to (3.8a) – (3.8e). The objective function (3.11a) and constraints (3.11b) – (3.11e) are now expressed as linear equations using auxiliary variables to achieve convexification. The updated objective function (3.11a) is formulated as an algebraic expansion of (3.8a). Considering the assumption $L_s/L_r = 1$, variable L_r is replaced by L_s in the problem formulation. Matrix equalities (3.11f) – (3.11m) enforce (3.9a) – (3.9j). However, problem (3.11a) – (3.11m) is still non-convex due to these matrix equalities (3.11f) – (3.11m). In the following subsection, these equality conditions are relaxed which make the optimization problem convex.

3.5.2 Conic Relaxation

Relaxation aims to formulate a convex approximation of a non-convex optimization problem. Such a formulation is favorable because its every minimum is a global minimum that can be readily obtained. Usually, relaxation is achieved by eliminating or modifying the constraints that lead to non-convexity [70]. For the problem (3.11a) – (3.11m), relaxed formulation can be obtained by transforming all matrix equalities in (3.11f) – (3.11m) to matrix inequalities as shown in the following.

Definition 1. *Define* $\mathcal{C}$ *as the set of vectors* $\mathbf{c} \in \mathbb{R}^5$ *that satisfy*

$$\begin{bmatrix} c_1 & c_3 \\ c_3 & c_2 \end{bmatrix} \succeq \begin{bmatrix} c_4 \\ c_5 \end{bmatrix} \begin{bmatrix} c_4 \\ c_5 \end{bmatrix}^\top . \tag{3.12}$$

It is straightforward to observe that $\mathcal{C}$ *is a convex set [70].*

minimize

$$\sum_{n\in\mathcal{S}} \boldsymbol{\iota}^\top[n]\boldsymbol{\Lambda}\left(\bar{\boldsymbol{i}}[n] + \mathrm{diag}\{\boldsymbol{i}^{\mathrm{MEA}}[n]\}(\boldsymbol{i}^{\mathrm{MEA}}[n] - 2\boldsymbol{i}[n])\right)$$
$$+ \gamma\iota_w[n]\left(\bar{\omega}_r[n] - 2\,\omega_r^{\mathrm{MEA}}[n]\omega_r[n] + \omega_r^{\mathrm{MEA}}[n]^2\right) \tag{3.13a}$$

subject to

Machine model equations: (3.11b) – (3.11e) (3.13b)

$$\left[\bar{\omega}_r[n], \bar{\lambda}_{7-k}[n], (-1)^k f_k[n], -\omega_r[n], \lambda_{7-k}[n]\right]^\top \in \mathcal{C}$$

$$f_{5-k}[n] = 0, \qquad k=3,4. \tag{3.13c}$$

$$\left[\bar{r}_s, \bar{i}_k[n], g_k[n], r_s, i_k[n]\right]^\top \in \mathcal{C}, \qquad k=1,2.$$

$$\left[\bar{r}_r, \bar{i}_k[n], g_k[n], r_r, i_k[n]\right]^\top \in \mathcal{C}, \qquad k=3,4. \tag{3.13d}$$

$$\left[\bar{L}_s, \bar{i}_k[n], h_k[n], L_s, i_k[n]\right]^\top \in \mathcal{C}, \qquad k=1,2,3,4. \tag{3.13e}$$

$$\begin{aligned}&\left[\bar{L}_m, \bar{i}_{k+2}[n], z_k[n], L_m, i_{k+2}[n]\right]^\top \in \mathcal{C}, && k=1,2.\\ &\left[\bar{L}_m, \bar{i}_{k-2}[n], z_k[n], L_m, i_{k-2}[n]\right]^\top \in \mathcal{C}, && k=3,4.\end{aligned} \tag{3.13f}$$

$$\begin{aligned}&\left[\bar{\lambda}_k[n], \bar{i}_{3-k}[n], y_k[n], \lambda_k[n], i_{3-k}[n]\right]^\top \in \mathcal{C},\\ &y_{2+k}[n] = 0, && k=1,2.\end{aligned} \tag{3.13g}$$

$$\begin{aligned}&\left[\bar{Q}_w, \bar{\omega}_r[n], \theta[n], Q_w, \omega_r[n]\right]^\top \in \mathcal{C},\\ &\left[\bar{B}, \bar{\omega}_r[n], \phi[n], B, \omega_r[n]\right]^\top \in \mathcal{C},\end{aligned} \tag{3.13h}$$

$$\bar{r}_s \geq r_s^2, \qquad \bar{r}_r \geq r_r^2, \qquad \bar{L}_s \geq L_s^2, \tag{3.13i}$$

$$\bar{L}_m \geq L_m^2, \quad \bar{Q}_w \geq Q_w^2, \quad \bar{B} \geq B^2 \tag{3.13j}$$

$$\bar{\boldsymbol{\lambda}}[n] \geq \operatorname{diag}\{\boldsymbol{\lambda}[n]\}\boldsymbol{\lambda}[n], \tag{3.13k}$$

$$\bar{\boldsymbol{i}}[n] \geq \operatorname{diag}\{\boldsymbol{i}[n]\}\boldsymbol{i}[n], \tag{3.13l}$$

$$\bar{\omega}_r[n] \geq \omega_r^2[n] \tag{3.13m}$$

$$\bar{T}_e[n] \geq T_e^2[n]. \tag{3.13n}$$

variables

$$\left\{\boldsymbol{\lambda}[n], \bar{\boldsymbol{\lambda}}[n], \boldsymbol{i}[n], \bar{\boldsymbol{i}}[n], \boldsymbol{f}[n], \boldsymbol{g}[n], \boldsymbol{h}[n], \boldsymbol{z}[n], \boldsymbol{y}[n] \in \mathbb{R}^4\right\}_{n\in\mathcal{S}},$$

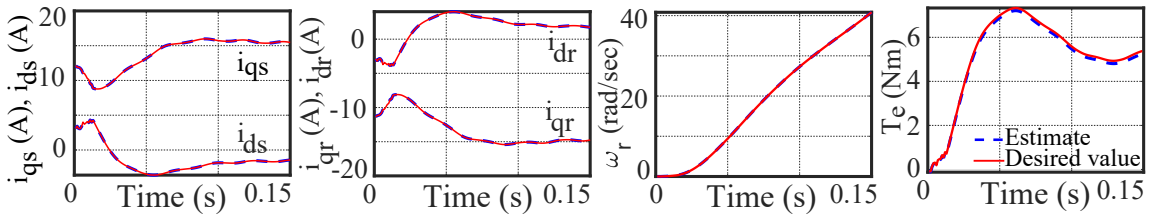
$$\left\{T_e[n], \bar{T}_e[n], \omega_r[n], \bar{\omega}_r[n], \theta[n], \phi[n] \in \mathbb{R}\right\}_{n\in\mathcal{S}},$$

$$r_s, \bar{r}_s, r_r, \bar{r}_r, L_s, \bar{L}_s, L_m, \bar{L}_m, Q_w, \bar{Q}_w, B, \bar{B} \in \mathbb{R}.$$

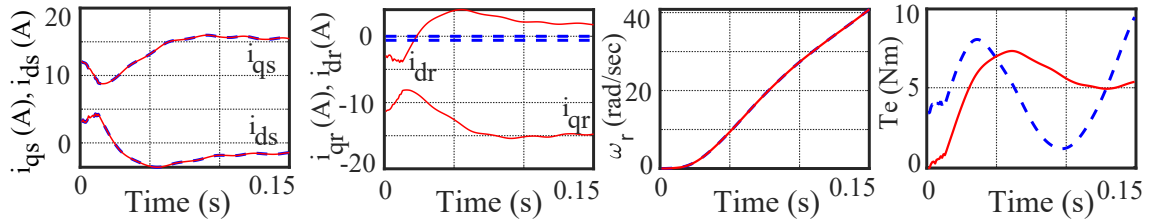
(3.13c) – (3.13n) implicitly impose the matrix equalities (3.11f) – (3.11m). (3.13c) – (3.13n) are same as (3.11f) – (3.11m) except for the equality/inequality condition. A new notation using $\mathcal{C}$ is used for compactness. If the solution to the relaxed problem (3.13a) – (3.13n) satisfies (3.11f) – (3.11m) (probably unlikely), then the relaxation is declared as exact[70]. Otherwise, the obtained solution is infeasible for the problem (3.8a) – (3.8e). To assess the solution obtained from conic relaxation, we solve (3.13a) – (3.13n) for the startup transient of an induction machine. We consider a simulated case study where the relaxed problem is solved under different scenarios. The machine model used in the numerical simulation is constructed using the parameters obtained via conventional test as seen in Table 5. The machine model is simulated with zero initial conditions and with the input voltage of 220 V (line-to-line). First, to establish a benchmark, rotor currents are *intentionally* assumed to be available. We then consider the realistic scenario that rotor currents are unavailable. Figure 8a shows the estimated currents $\boldsymbol{i}$, speed ω_r, and torque T_e (part of the optimization solution) for a scenario when the rotor currents are available with $\boldsymbol{\Lambda} = \text{diag}\{[0.1, 0.1, 0.1, 0.1]\}$ and $\gamma = 0.1$. The result obtained from conic relaxation is near optimal as evident from the estimated waveforms. However, the absence of rotor current measurements (with $\boldsymbol{\Lambda} = \text{diag}\{[0.1, 0.1, 0, 0]\}$ and $\gamma = 0.1$), leads to poor estimates for rotor currents and torque (see Figure 8b). (3.13a) – (3.13n) is a convex approximation to the original problem in (3.8a) – (3.8e), leading to an approximate solution in Figure 8b.

3.5.3 Incorporating Penalty

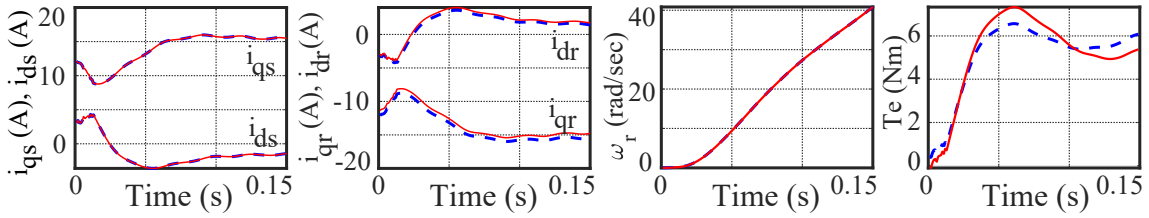
The solution from the conic relaxation can be further improved by incorporating penalty terms into the objective function [33]. Let $\hat{i}_{qr}$ and $\hat{i}_{dr}$ denote rough



(a) Estimates from conic relaxation (3.13a) – (3.13n) using rotor currents.



(b) Estimates from conic relaxation (3.13a) – (3.13n) without rotor currents.



(c) Estimates from penalized conic relaxation (3.13a) – (3.13n) without rotor currents.

Figure 8: Assessment of the estimates obtained from conic relaxation.

guesses for q-axis and d-axis rotor currents. Therefore, one can augment the objective function (3.13a) with the penalty term

$$\kappa = \sum_{n\in\mathcal{S}} \eta_{i_{qr}}(\bar{i}_{qr}[n] - 2\hat{i}_{qr}[n]i_{qr}[n] + \hat{i}^2_{qr}[n]) + \eta_{i_{dr}}(\bar{i}_{dr}[n] - 2\hat{i}_{dr}[n]i_{dr}[n] + \hat{i}^2_{dr}[n]). \tag{3.14}$$

$\eta_{i_{qr}}$ and $\eta_{i_{dr}}$ are user-defined non-negative gains with $\bar{i}_{qr} \triangleq i^2_{qr}$ and $\bar{i}_{dr} \triangleq i^2_{dr}$. The penalty term in (3.14) incentivizes the optimization solver to search around the neighborhood of guessed rotor currents. The penalty term (3.14) is a convex reformulation for $\eta_{i_{qr}}(i_{qr}[n] - \hat{i}_{qr}[n])^2 + \eta_{i_{dr}}(i_{dr}[n] - \hat{i}_{dr}[n])^2$. A decent guess for rotor currents can be

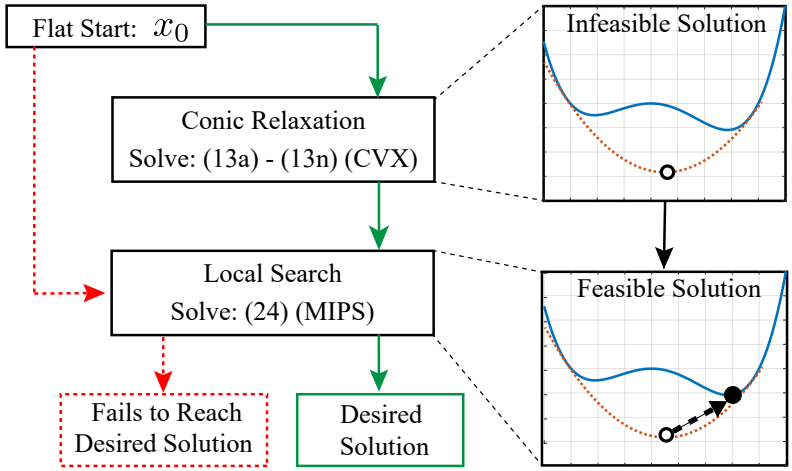


Figure 9: Overview of the two-step solution for the estimation problem in (3.8).

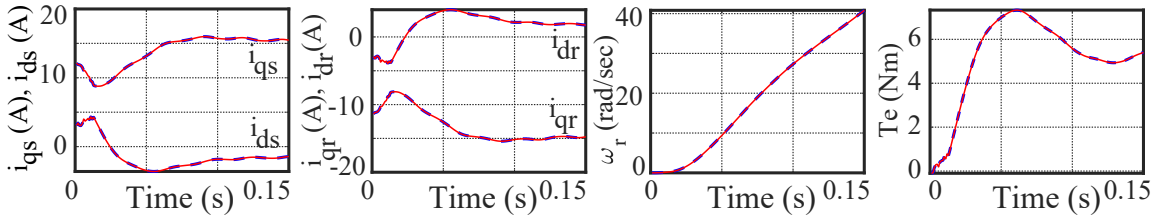


Figure 10: Local search results for the example in Fig. 8, which is initialized with the solution of the penalized conic relaxation in Fig. 8c.

$\hat{i}_{qr}[n] \approx -i_{qs}[n]$ and $\hat{i}_{dr}[n] \approx -i_{ds}[n]$ during the startup [58]. Similarly, at the steady state, $\hat{i}_{qr}[n] \approx 0$ and $\hat{i}_{dr}[n] \approx 0$. Figure 8c shows the results for the relaxed problem (3.13a) – (3.13n) when the objective function (3.13a) is augmented with the penalty term (3.14). Penalization significantly improves the results for the relaxed problem (with $\boldsymbol{\Lambda} = \text{diag}\{[0.1, 0.1, 0, 0]\}$, $\gamma = 0.1$, and $\eta_{i_{qr}} = \eta_{i_{dr}} = 1$).

3.5.4 Local Search

As discussed in Section 3.5.2, the solution to the relaxed optimization problem (3.13a) – (3.13n) could be infeasible for the original problem (3.8a) – (3.8e). It can, however, serve as an excellent initial condition to a local search algorithm[31]. To solve

the estimation problem using an iterative Newton's method, we need to formulate the Karush-Kuhn-Tucker (KKT) conditions [70]. By defining the optimization variable as

$$\boldsymbol{x} \triangleq \left[\check{\boldsymbol{\lambda}}^\top, \check{\boldsymbol{i}}^\top, \check{\boldsymbol{T}}_e^\top, \breve{\boldsymbol{\omega}}_r^\top, L_s, L_m, r_s, r_r, Q_w, B\right]^\top, \tag{3.15}$$

the equality constraint (3.8b) – (3.8e) can then be cast as

$$\boldsymbol{\mathcal{E}}(\boldsymbol{x}) \triangleq [\boldsymbol{\mathcal{E}_1}(\boldsymbol{x})^\top\ \boldsymbol{\mathcal{E}_2}(\boldsymbol{x})^\top\ \boldsymbol{\mathcal{E}_3}(\boldsymbol{x})^\top\ \boldsymbol{\mathcal{E}_4}(\boldsymbol{x})^\top]^\top = \boldsymbol{0}, \tag{3.16}$$

where

$$\begin{aligned}
\boldsymbol{\mathcal{E}_1}(\boldsymbol{x}) &\triangleq (\boldsymbol{K}_1 - \boldsymbol{K}_2)\check{\boldsymbol{\lambda}} - \Delta T\Big((\mathrm{diag}\,\{\boldsymbol{K}_4\breve{\boldsymbol{\omega}}_r\}\otimes \boldsymbol{I}_4)\boldsymbol{N}_1\boldsymbol{K}_2\check{\boldsymbol{\lambda}} \\
&\quad + \omega \boldsymbol{N}_2\boldsymbol{K}_2\check{\boldsymbol{\lambda}} - (\boldsymbol{I}_{\tau-1}\otimes \boldsymbol{R})\boldsymbol{K}_2\check{\boldsymbol{i}} + \boldsymbol{K}_2\check{\boldsymbol{v}}\Big), && \text{(3.17a)}\\
\boldsymbol{\mathcal{E}_2}(\boldsymbol{x}) &\triangleq Q_w\boldsymbol{K}_3\breve{\boldsymbol{\omega}}_r - \boldsymbol{K}_4\big(Q_w\breve{\boldsymbol{\omega}}_r + \frac{\Delta T}{2}(\check{\boldsymbol{T}}_e - \frac{2B}{P}\breve{\boldsymbol{\omega}}_r)\big), && \text{(3.17b)}\\
\boldsymbol{\mathcal{E}_3}(\boldsymbol{x}) &\triangleq \check{\boldsymbol{\lambda}} - (\boldsymbol{I}_\tau \otimes \mathrm{diag}\,\{[L_s, L_s, L_s, L_s]\})\check{\boldsymbol{i}} - L_m\boldsymbol{N}_3\check{\boldsymbol{i}}, && \text{(3.17c)}\\
\boldsymbol{\mathcal{E}_4}(\boldsymbol{x}) &\triangleq \check{\boldsymbol{T}}_e - \frac{3P}{4}\mathrm{diag}\{(\boldsymbol{I}_\tau \otimes \boldsymbol{d}_2^\top)\check{\boldsymbol{\lambda}}\}(\boldsymbol{I}_\tau \otimes \boldsymbol{d}_1^\top)\check{\boldsymbol{i}} + \frac{3P}{4}\mathrm{diag}\{(\boldsymbol{I}_\tau \otimes \boldsymbol{d}_1^\top)\check{\boldsymbol{\lambda}}\}(\boldsymbol{I}_\tau \otimes \boldsymbol{d}_2^\top)\check{\boldsymbol{i}}. && \text{(3.17d)}
\end{aligned}$$

$\boldsymbol{K}_1 \triangleq [\boldsymbol{0}_{4(\tau-1)\times 4}, \boldsymbol{I}_{4(\tau-1)}]$, $\boldsymbol{K}_2 \triangleq [\boldsymbol{I}_{4(\tau-1)}, \boldsymbol{0}_{4(\tau-1)\times 4}]$, $\boldsymbol{K}_3 \triangleq [\boldsymbol{0}_{\tau-1}, \boldsymbol{I}_{\tau-1}]$, and $\boldsymbol{K}_4 \triangleq [\boldsymbol{I}_{\tau-1}, \boldsymbol{0}_{\tau-1}]$ and

$$\mathbf{N}_1 \triangleq \boldsymbol{I}_{\tau-1} \otimes (\boldsymbol{d}_3\boldsymbol{d}_4^\top - \boldsymbol{d}_4\boldsymbol{d}_3^\top), \tag{3.18a}$$

$$\mathbf{N}_2 \triangleq \boldsymbol{I}_{\tau-1} \otimes (-\boldsymbol{d}_1\boldsymbol{d}_2^\top + \boldsymbol{d}_2\boldsymbol{d}_1^\top - \boldsymbol{d}_3\boldsymbol{d}_4^\top + \boldsymbol{d}_4\boldsymbol{d}_3^\top), \tag{3.18b}$$

$$\mathbf{N}_3 \triangleq \boldsymbol{I}_{\tau} \otimes (\boldsymbol{d}_1\boldsymbol{d}_3^\top + \boldsymbol{d}_2\boldsymbol{d}_4^\top + \boldsymbol{d}_3\boldsymbol{d}_1^\top + \boldsymbol{d}_4\boldsymbol{d}_2^\top). \tag{3.18c}$$

$\boldsymbol{x}$ denotes the concatenated form of the optimization variable. Symbol $\breve{\cdot}$ represents the vectorization operator (see notations). $\boldsymbol{\mathcal{E}_1}(\boldsymbol{x})$, $\boldsymbol{\mathcal{E}_2}(\boldsymbol{x})$, $\boldsymbol{\mathcal{E}_3}(\boldsymbol{x})$, and $\boldsymbol{\mathcal{E}_4}(\boldsymbol{x})$ are the

$$\boldsymbol{J}\big(\boldsymbol{x}\big) = \begin{bmatrix} \nabla_{\check{\boldsymbol{\lambda}}}\mathcal{E}_1 & \nabla_{\check{\boldsymbol{i}}}\mathcal{E}_1 & \mathbf{0} & \nabla_{\check{\omega}_r}\mathcal{E}_1 & \mathbf{0} & \mathbf{0} & \nabla_{r_s}\mathcal{E}_1 & \nabla_{r_r}\mathcal{E}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \nabla_{\check{T}_e}\mathcal{E}_2 & \nabla_{\check{\omega}_r}\mathcal{E}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \nabla_{Q_w}\mathcal{E}_2 & \nabla_{B}\mathcal{E}_2 \\ \nabla_{\check{\boldsymbol{\lambda}}}\mathcal{E}_3 & \nabla_{\check{\boldsymbol{i}}}\mathcal{E}_3 & \mathbf{0} & \mathbf{0} & \nabla_{L_s}\mathcal{E}_3 & \nabla_{L_m}\mathcal{E}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \nabla_{\check{\boldsymbol{\lambda}}}\mathcal{E}_4 & \nabla_{\check{\boldsymbol{i}}}\mathcal{E}_4 & \nabla_{\check{T}_e}\mathcal{E}_4 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.19}$$

vectorized form of machine model (3.7). Variables $\mathbf{K}_1-\mathbf{K}_4$ and $\mathbf{N}_1-\mathbf{N}_3$ are defined to achieve vectorization. $(\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4)$ are the standard basis of $\mathbb{R}^4$. One can formulate the Jacobian matrix $\boldsymbol{J}\big(\boldsymbol{x}\big)$ for the constraints (3.17a) – (3.17d) in the form of (3.19). In (3.19), $\nabla_{\{.\}}$ denotes the derivative with respect to the subscript variable, e.g., $\nabla_{\check{\boldsymbol{\lambda}}}\mathcal{E}_1$ is the derivative of $\mathcal{E}_1$ with respect to $\check{\boldsymbol{\lambda}}$. Additionally, the Lagrangian function $\mathcal{L}$ of the optimization problem (3.8a) – (3.8e) can be cast as

$$\begin{aligned} \mathcal{L}\big(\boldsymbol{x};\boldsymbol{\nu}\big) \triangleq & (\check{\boldsymbol{i}} - \check{\boldsymbol{i}}^{\mathrm{MEA}})^{\top}\big(\mathrm{diag}\{\check{\boldsymbol{\iota}}\}(\boldsymbol{I}_{\tau}\otimes\boldsymbol{\Lambda})\big)(\check{\boldsymbol{i}} - \check{\boldsymbol{i}}^{\mathrm{MEA}}) \\ & + \gamma(\check{\boldsymbol{\omega}}_r - \check{\boldsymbol{\omega}}_r^{\mathrm{MEA}})^{\top}\mathrm{diag}\{\check{\boldsymbol{\iota}}_{\omega}\}(\check{\boldsymbol{\omega}}_r - \check{\boldsymbol{\omega}}_r^{\mathrm{MEA}}) + \boldsymbol{\nu}^{\top}\mathcal{E}(\boldsymbol{x}), \end{aligned} \tag{3.20}$$

where $\boldsymbol{\nu}$ is the vector of Lagrange multipliers, $\check{\boldsymbol{i}}^{\mathrm{MEA}}$ and $\check{\boldsymbol{\omega}}_r^{\mathrm{MEA}}$ denote vectorized current and speed measurements. The gradient of $\mathcal{L}$ with respect to $\boldsymbol{x}$ can be formulated as

$$\begin{aligned} \boldsymbol{G}\big(\boldsymbol{x};\,\boldsymbol{\nu}\big) \triangleq 2\times\Big[\mathbf{0}, \Big(\big(\mathrm{diag}\{\check{\boldsymbol{\iota}}\}(\boldsymbol{I}_{\tau}\otimes\boldsymbol{\Lambda})\big)(\check{\boldsymbol{i}} - \check{\boldsymbol{i}}^{\mathrm{MEA}})\Big)^{\top}, \mathbf{0}, \\ \gamma\Big(\mathrm{diag}\{\check{\boldsymbol{\iota}}_{\omega}\}(\check{\boldsymbol{\omega}}_r - \check{\boldsymbol{\omega}}_r^{\mathrm{MEA}})\Big)^{\top}, \mathbf{0}\Big] + \boldsymbol{\nu}^{\top}\boldsymbol{J}\big(\boldsymbol{x}\big). \end{aligned} \tag{3.21}$$

Finally, the Hessian of $\mathcal{L}$ is

$$\boldsymbol{H}\big(\boldsymbol{x};\,\boldsymbol{\nu}\big) \triangleq \begin{bmatrix} \boldsymbol{H}_{11} & \boldsymbol{H}_{12} \\ \boldsymbol{H}_{12}^{\top} & \mathbf{0} \end{bmatrix}, \tag{3.22}$$

where

$$\boldsymbol{H}_{11} \triangleq \begin{bmatrix} \mathbf{0} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{\lambda}} \partial \check{\boldsymbol{i}}} & \mathbf{0} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{\lambda}} \partial \breve{\boldsymbol{\omega}}_r} \\ \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}} \partial \check{\boldsymbol{\lambda}}} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}}^2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \frac{\partial^2 \mathcal{L}}{\partial \breve{\boldsymbol{\omega}}_r \partial \check{\boldsymbol{\lambda}}} & \mathbf{0} & \mathbf{0} & \frac{\partial^2 \mathcal{L}}{\partial \breve{\boldsymbol{\omega}}_r^2} \end{bmatrix}, \tag{3.23a}$$

$$\boldsymbol{H}_{12} \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}} \partial L_s} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}} \partial L_m} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}} \partial r_s} & \frac{\partial^2 \mathcal{L}}{\partial \check{\boldsymbol{i}} \partial r_r} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial^2 \mathcal{L}}{\partial \breve{\boldsymbol{\omega}}_r \partial Q_w} & \frac{\partial^2 \mathcal{L}}{\partial \breve{\boldsymbol{\omega}}_r \partial B} \end{bmatrix}. \tag{3.23b}$$

Newton steps of the form

$$\begin{bmatrix} \varDelta \boldsymbol{x} \\ \\ \varDelta \boldsymbol{\nu} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{H}\big(\boldsymbol{x};\, \boldsymbol{\nu}\big) & \boldsymbol{J}\big(\boldsymbol{x}\big)^\top \\ \\ \boldsymbol{J}\big(\boldsymbol{x}\big) & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{G}\big(\boldsymbol{x};\, \boldsymbol{\nu}\big)^\top \\ \\ \boldsymbol{\mathcal{E}}(\boldsymbol{x}) \end{bmatrix}, \tag{3.24}$$

converge to a solution that meets the KKT optimality conditions. Figure 10 shows the results for the example presented in Figure 8, when the solution of the penalized conic relaxation (Fig. 8c) initializes a local search procedure. Figure 9 shows the steps needed to solve the estimation problem (3.8a) – (3.8e).

3.6 Experimental Studies

3.6.1 Numerical and Experimental Setups

The numerical studies are performed on a workstation using Windows 10 equipped with quad-core Intel® Core™ i7-6700 with 32 GB RAM. The relaxed optimization problem (3.13a) – (3.13n) is solved using the SDPT3 4.0 [34] solver in the CVX [36]

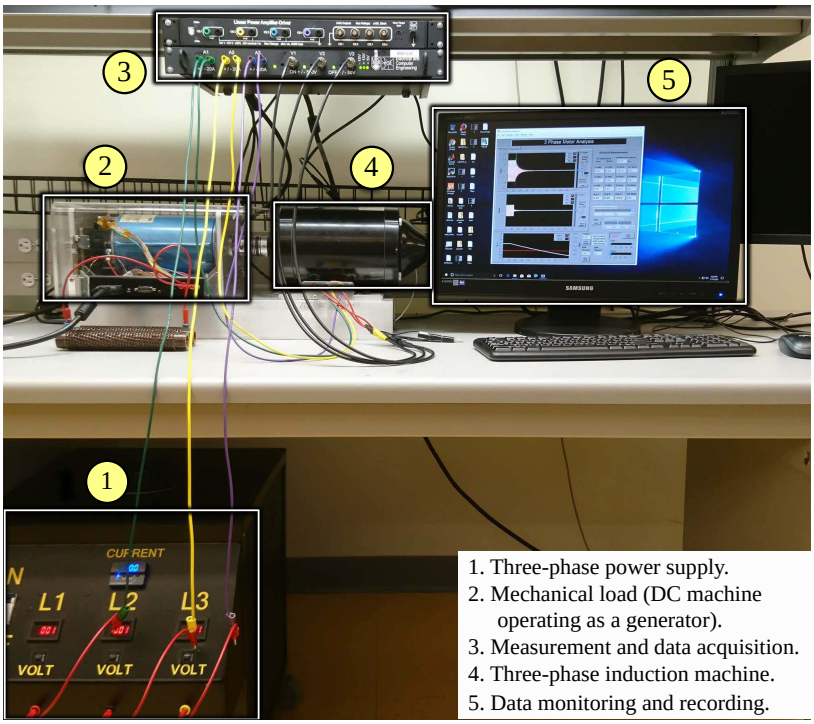


Figure 11: Hardware setup used to measure machine transients and characteristics.

environment on MATLAB 2019a. MATPOWER Interior Point Solver (MIPS) [73] version 1.3.1 performs the local search. MIPS solves (3.24) using the formulations of objective function and equality constraints (3.8a) – (3.8e), Jacobian matrix (3.19), Hessian matrix (3.22), and the solution of the relaxed optimization problem (3.13a) – (3.13n) as an initial condition. The termination tolerances for the MIPS solver, namely, `gradtol`, `feastol`, `comptol`, and `costtol` are selected as 10^{-8}, 10^{-8}, 10^{-6}, and 10^{-6}, respectively. The maximum iteration count is set to 50.

Figure 11 shows the experimental setup used for measurement acquisition. The four-pole motor is excited with a voltage of 220 V (line-to-line). The measured data is demonstrated in Fig. 12. For comparison, machine parameters have also been identified through standard intrusive characterization tests[45]. These conventional tests include dc stator resistance measurements, locked rotor test (used for identifying

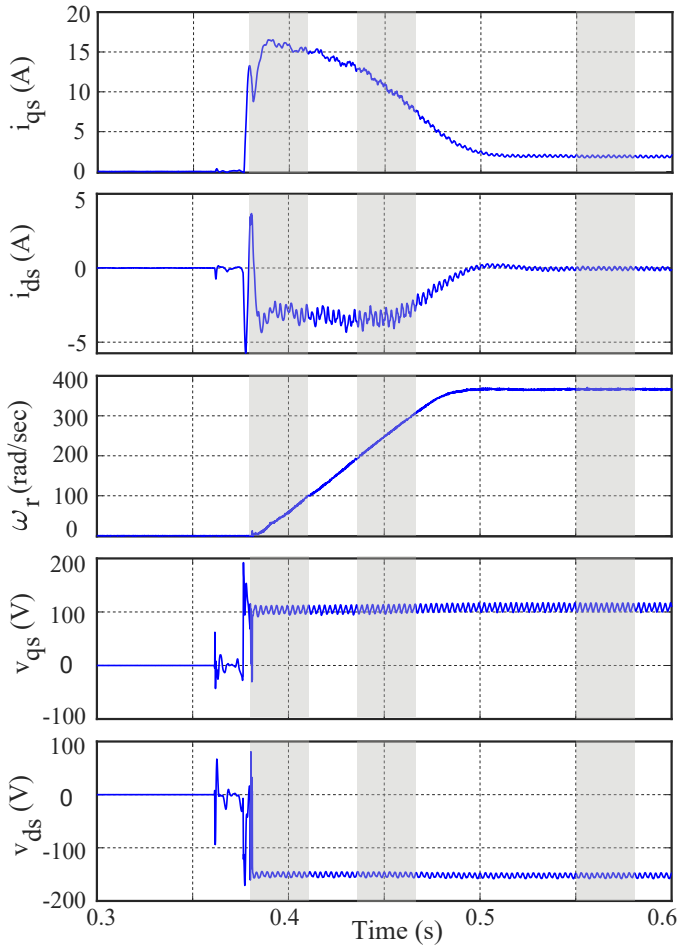


Figure 12: Measured startup transients for the underlying induction machine. The highlighted portion of the transients are used for parameter identification.

rotor resistance along with stator and rotor leakage inductances), no-load test (used for identifying magnetizing inductance), no-load deceleration test (used for identifying total lumped inertia of the machine and dynamometer), and no-load startup test (used for identifying start-up impedance of the machine).

3.6.2 Parameter Extraction from Measurement

Parameters such as leakage inductance and inertia are more dominant during transients, whereas magnetizing inductance has a prominent effect at the steady state.

Hence, limited data from both acceleration and steady-state phases of the measured waveforms are used for the model identification procedure. Sampling time of 100 μs is used in the measured data shown in Fig. 12. The highlighted portions of the data is used for parameter extraction. Synchronous reference frame is chosen for the machine model with $\omega = 120\pi$. The non-negative weights for the relaxed problem (3.13a) – (3.13n) and the local search (3.8a) – (3.8e) are $\boldsymbol{\Lambda} = \text{diag}\{[0.1, 0.1, 0, 0]\}$ and $\gamma = 0.1$, with binary flags as $\boldsymbol{\iota} = [1, 1, 0, 0]^\top$ and $\iota_w = 1$. While choosing $\boldsymbol{\Lambda}$ and γ, one should note that (1) larger gain implies higher priority for the optimization solver, and (2) gains can be used to normalize the order of terms in the objective function. For example, $\boldsymbol{\Lambda} = \text{diag}\{[1, 1, 0, 0]\}$ and $\gamma = 1$ or $\boldsymbol{\Lambda} = \text{diag}\{[0.01, 0.01, 0, 0]\}$ and $\gamma = 0.01$ are also viable options. The gains for penalty terms in (3.14) are set as $\eta_{i_{qr}} = \eta_{i_{dr}} = 1$. The optimization problem (3.13a) – (3.13n), with the penalty term (3.14), is first solved. The outcome of this relaxation is then used as an initial point for the local search (3.24) (see Fig. 9). A single run of convex relaxation takes approximately 185 seconds (including all overheads) on average. Local search concludes within 60 seconds. Table 5 compares the machine parameters estimated by the proposed method against parameters extracted using conventional tests. In addition to resistances, inductances, and inertia terms, we could estimate the friction coefficient as well. Table 5 also lists the values of objective function (3.8a) that, for the proposed method, is almost half of that predicted by parameters obtained from conventional tests. Since rotor current measurements are not available, (3.8a) takes the following form

$$\sum_{n\in\mathcal{S}} \alpha_1\iota_1[n](i_{qs}[n]-i_{qs}^{\text{MEA}}[n])^2+\alpha_2\iota_2[n](i_{ds}[n]-i_{ds}^{\text{MEA}}[n])^2 \qquad (3.25)$$
$$+\gamma\iota_\omega[n]\big(\omega_r[n]-\omega_r^{\text{MEA}}[n]\big)^2.$$

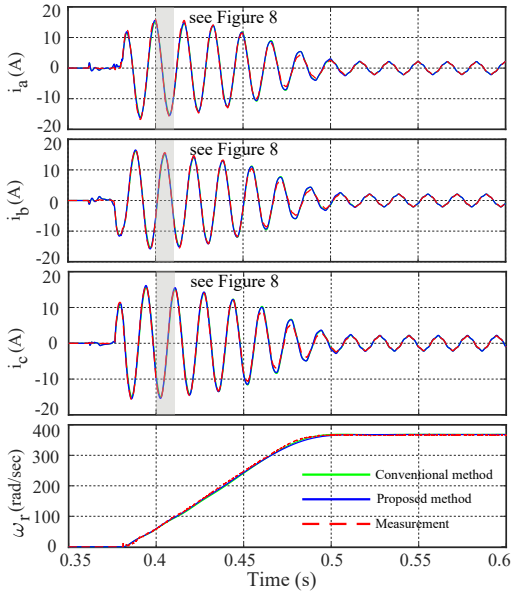


Figure 13: Estimated parameters listed in Table 5 are used to simulate the induction machine model from zero initial conditions. The resulting current and speed waveforms are compared against measured data.

The aggregate value of (3.25) is attributed to individual mismatch expressions for stator currents (i_{qs} and i_{ds}) and rotor speed (ω_r). For the proposed method, the contribution due to the first expression in (3.25) is about 10.5, the second expression is 14.4, and the last expression is 623.4. For parameters obtained from conventional tests, respective expressions contribute about 27.3 (mismatch in i_{qs}), 34.0 (mismatch in i_{ds}), and 1173.9 (mismatch in ω_r). The absolute value of (3.25) depends on user-defined coefficients α_1, α_2, and γ. Therefore, one should consider the relative improvement in the value of the objective function obtained by the proposed method, as shown in the last column of Table 5. Moreover, as opposed to multiple interruptive and intrusive tests needed in the conventional approach, our method extracts machine parameters from measured data obtained during the machine's normal operation.

Two dynamic models built using two sets of machine parameters, one extracted using the proposed method and another obtained via conventional methods, are con-

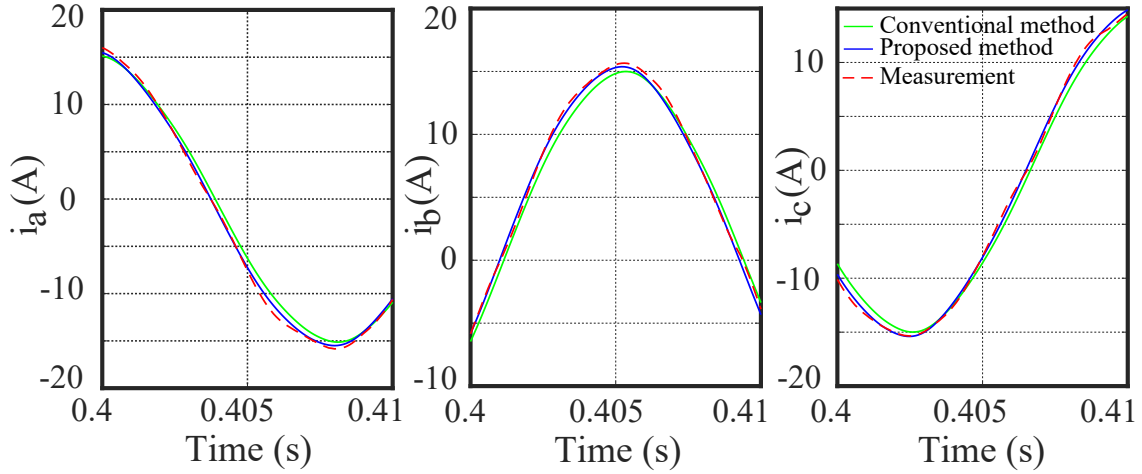


Figure 14: Zoomed-in version of the highlighted portions of Fig. 13. We can observe a better match between the waveforms predicted by the estimated parameters with measurements.

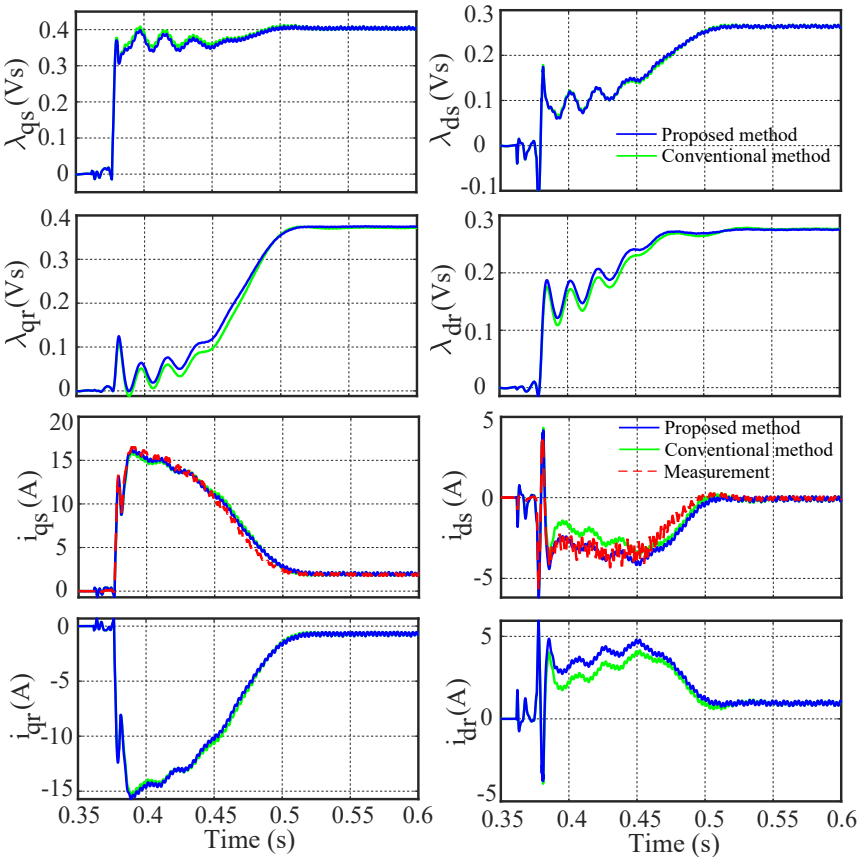


Figure 15: Trajectories of *qd* flux linkages and currents obtained using the parameters listed in Table 5. Stator currents are compared against measurements.

Table 5: Machine parameters extracted using the conventional tests and the proposed method

Methodology	L_s (H)	L_m (H)	L_r (H)	r_s (Ω)	r_r (Ω)	J (kg.m^2)	B (N.m.s/rad)	Objective fn (3.8a)
Proposed Method	0.3149	0.3040	0.3149	4.50	3.45	0.0041	0.0089	648.3
Conventional Tests	0.3207	0.3087	0.3207	4.52	3.23	0.0037	-	1235.2

sidered. Figures 13 and 14 compare the stator currents and speed waveforms obtained from simulating the machine models using the two sets of parameters listed in Table 5. Figure 15 shows the trajectories of both stator and rotor flux linkages and currents in the *qd*-axis. The machine model is simulated with zero initial conditions. Parameters obtained from the proposed method result in an excellent fit to the measured data as evident from Fig. 13. Figure 14 shows a zoomed-in view of a portion of stator current in Figure 13. It is evident that the time-domain transients predicted by the parameters obtained by the proposed method match better with the measured signal compared to transients predicted by the parameters obtained using the conventional methods. This fit can be quantified using metrics like the root-mean-square error (RMSE) or the 2-norm error defined as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i^{\text{MEA}} - x_i^{\text{EST}})^2}, \tag{3.26a}$$

$$\text{2-norm error} = \frac{\sqrt{\sum_{i=1}^{N}(x_i^{\text{MEA}} - x_i^{\text{EST}})^2}}{\sqrt{\sum_{i=1}^{N}(x_i^{\text{MEA}})^2}} \times 100. \tag{3.26b}$$

x_i^{EST} and x_i^{MEA} are the i^{th} samples of the estimated and measured signals, respectively. N denotes the number of samples considered. Table 6 lists RMSE and 2-norm error values while comparing the two sets of machine waveforms against real measurements. Parameters extracted by the proposed method result in lower RMSE and 2-norm error as compared to the parameter obtained from conventional tests. Table

Table 6: Quantification of the mismatch against measurement

Signal	RMSE		2-norm		Percentage Improved
	Proposed method	**Conventional tests**	**Proposed method**	**Conventional tests**	
i_a	0.433	0.625	6.847	9.882	30.7
i_b	0.455	0.520	7.188	8.216	12.5
i_c	0.441	0.593	7.076	9.514	25.6
ω_r	3.70	4.05	1.227	1.343	8.7
Average					19.3

6 lists the percentage reduction in error values against parameters from conventional tests. RMSE reduces from 0.625 to 0.433 for the phase-a current, a 30% percent improvement. The RMSE for rotor speed has improved by more than 8%. On average, we see a 19.3% improvement in error. Percentage improvements in Table 6 are the same for both RMSE and 2-norm metrics.

3.6.3 Estimation with Missing Data Points

We now test the resilience of the proposed method against loss in measured data. We assume that indicators $\boldsymbol{\iota}[n]$ and $\iota_w[n]$ take the value 1 with a probability of 0.8, such that random data points for stator currents and rotor speed are flagged as unavailable in (3.8a) and (3.13a). This is implemented using the rand function (uniformly distributed pseudorandom numbers) in MATLAB. Figure 16 compares the estimated current and speed trajectories against the input measurements. Signal loss is shown by zeros along the time axis for illustration purposes (lost data are not necessarily zero in value). The optimization algorithm successfully reproduces the entire current trajectory along with machine parameters.

3.6.4 Impact of Noisy Input Data

The machine model is first simulated assuming the parameters listed in Table 5 (conventional tests values) and, then, the resulting state transients are polluted

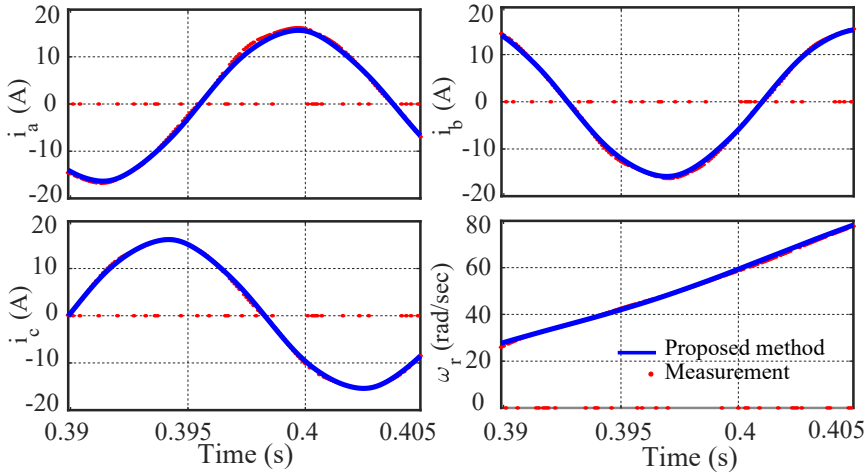


Figure 16: Stator currents and rotor speed transients, considering lossy measurement, predicted by the estimated parameters versus input measurement. The red dots along the time-axis represent the instances of data loss.

Table 7: Parameter sets obtained from noisy data

Machine Parameters	**0% noise**	**2% noise**	**5% noise**
L_s	0.3207	0.3216	0.3193
L_m	0.3087	0.3096	0.3074
L_r	0.3207	0.3216	0.3193
r_s	4.52	4.57	4.38
r_r	3.23	3.18	3.27
J	0.0037	0.0036	0.0038
B	0.0089	0.0087	0.0085

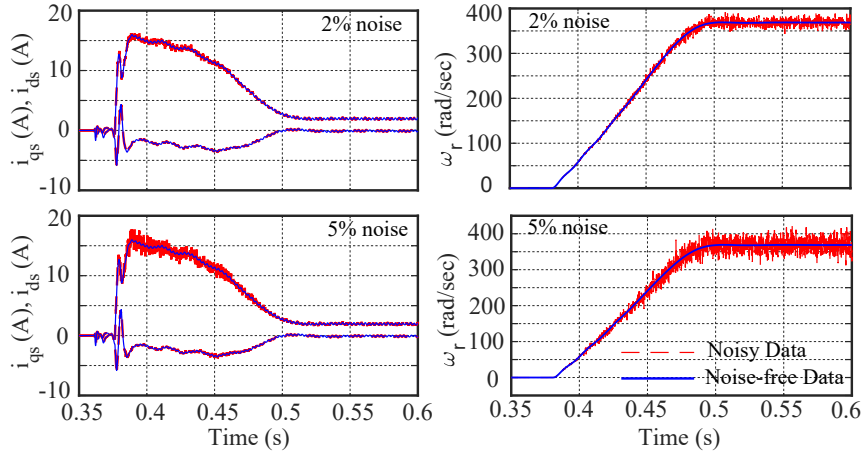


Figure 17: Simulated machine transients polluted with different noise levels.

Table 8: Estimation results under various sampling times

Machine Parameters	$\Delta T = 50$ μs	$\Delta T = 100$ μs	$\Delta T = 200$ μs	$\Delta T = 300$ μs
L_s	0.3153	0.3149	0.3068	0.3412
L_m	0.3043	0.3040	0.2958	0.3300
L_r	0.3153	0.3149	0.3068	0.3412
r_s	4.51	4.50	4.44	4.68
r_r	3.44	3.45	3.52	3.26
J	0.0041	0.0041	0.0042	0.0040
B	0.0089	0.0089	0.0077	0.0091

with noise signals with zero mean and 2%/5% standard deviations. Figure 17 shows both the noise-free and distorted transients. Table 7 lists the parameters obtained under different noise scenarios. As seen, estimated parameters, in presence of noisy data, are in good agreement with those obtained from noise-free data. In practice, low-pass filtering effects of acquisition devices eliminate severe noisy data. Interested readers can refer to [74] for a detailed theoretical analysis on the impact of noise on the performance of conic relaxation equipped a weighted least squared estimator, and to [75, 76] for the treatment of process or measurement noise in estimation process.

3.6.5 Impact of Sampling Time

For a given time horizon of data, increasing sampling time interval ΔT reduces the number of data points used in the estimation process. On the other hand, ΔT should be sufficiently small to have fidelity with the original machine equations. Using available data from Figure 12, Table 8 lists parameters extracted under different sampling time intervals. As ΔT is increased, estimated parameters start to deviate due to errors induced by the discretization process. Figure 18 compares the measured data against machine transients simulated using the parameters sets in the last two columns of Table 8 (and their respective sampling times). Notice that, particularly, at ΔT=300μs, simulated transients clearly deviate from their measured counterparts.

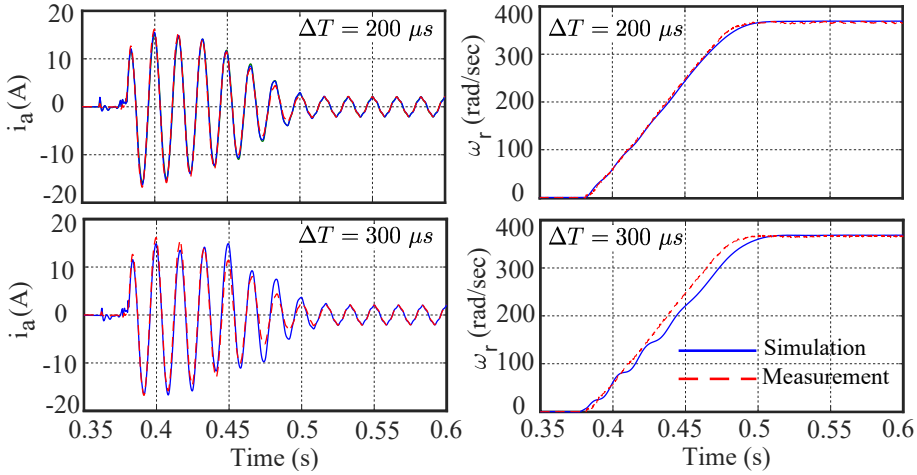


Figure 18: Machine transients simulated under different ΔTs, with parameter sets obtained under those ΔTs, compared to hardware measurement.

3.7 Summary

We extract the parameters of an induction machine in a non-intrusive manner using startup transients. The non-convex parameter identification problem is convexified using conic relaxation, whose output is transformed into an accurate solution for the machine dynamical equations using a local search. The proposed method is experimentally shown to identify machine parameters, even with intermittent losses in measured data. These parameters are shown to result in a better match with the measured signals compared to those obtained using conventional tests (nearly a 20% improvement in matching transient waveforms). Future research direction includes expanding this approach to more comprehensive machine models (e.g., with variable parameters).

PMSM ESTIMATES FROM INTERMITTENT DATA

Authors: A. P. Yadav, M. Davoodi, N. Gans, and A. Davoudi.

Reprinted, with permission from all the co-authors.

(`under preparation`).

CHAPTER 4

PMSM ESTIMATES FROM INTERMITTENT DATA

Abstract

The partial-update Kalman filter (PKF) is an extension of the Schmidt Kalman filter, which can improve capabilities of the conventional extended Kalman filter for handling the model uncertainties and nonlinearities. Herein, we adapt the PKF to estimate the states and parameters of electric machines, particularly in cases with intermittent observations. To account for missing data within the filter, arrival of new measurements is treated as a Bernoulli process. We show that the estimation error of the proposed filter remains bounded if the system satisfies mild assumptions. Moreover, we show that the prediction error covariance matrix is guaranteed to be bounded if the observation arrival rate has a lower bound. Hardware experiments validate this technique for a permanent magnet synchronous motor.

4.1 Introduction

Accurate and robust estimation of states and parameters of electric machines are needed for drive design and control, system-level studies, or condition monitoring [77]. State estimation usually involves observer design for hard-to-measure signals such as flux linkages or rotor currents. Parameter estimation obtains a set of parameters, such as resistances and inductances, in the machine model. Online estimation approaches, e.g., Kalman filter, are indispensable as machine parameters can change during the

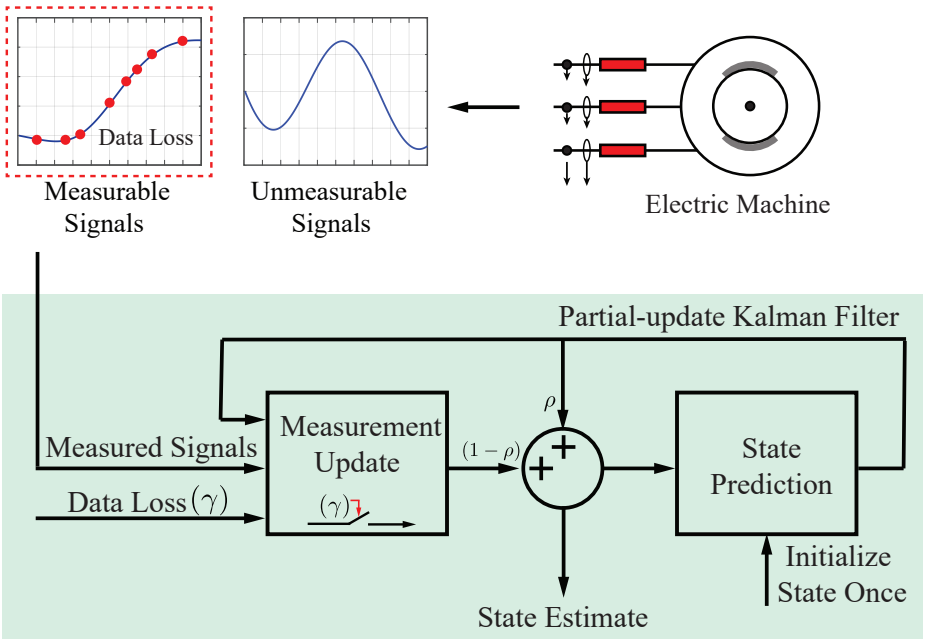


Figure 19: Overview of the partial-update Kalman filter. Estimates for states/parameters are obtained as weighted sums of the time update and measurement update terms. Loss of data is modeled using a binary random variable γ_k.

machine operation. Kalman filter-based estimators are popular due to their simple yet stochastic structure, which incorporates both process and measurement noises. Popular Kalman filter applications in electric machines include (1) state estimation, e.g., flux or speed[78, 79], (2) parameter estimation [53], and (3) condition monitoring during machine operation [80]. While Kalman filters are intended for linear systems, an extended Kalman filter (EKF) can be used for nonlinear systems such as electric machines [81]. Limitations of EKF include (1) error due to linearizing nonlinear system equations, (2) filter tuning, i.e., finding correct noise covariance matrices, and (3) ensuring a stable operation. In this chapter, we address these issues through a partial-update Kalman filter and validate our findings on a permanent magnet synchronous motor (PMSM).

Nuisance parameters denote those variables (such as static biases) that are not required to be estimated, but whose uncertainties need to be accounted for better accuracy of estimate variables of interest. In such cases, the Schmidt Kalman filter (SKF) accounts for uncertainties in system parameters and provide better estimates compared to standard Kalman filter [82]. Therein, nuisance parameters are added as a part of the state vector to incorporate model uncertainties into the estimates. However, these parameters are only considered, and not estimated, i.e., even though they are not estimated directly, the state estimates and covariance are updated based on an estimated parameter covariance [82]. In this way, the parameter uncertainty is propagated through the dynamic system while avoiding issues regarding simultaneous estimation of states and nuisance parameters (such as increased computational cost and possible instabilities) [83]. SKFs can be especially helpful for parameters with low observability. Such scenarios can also occur in machine applications, e.g., the load torque can often act as a nuisance parameter in an estimation problem. However, the treatment on nuisance parameters is fairly limited in the context of electric machines.

Partial-update Kalman filter (PKF) is a hybrid approach between SKF and EKF that can improve upon the performances of both by tuning gains [84]. A partial update (a percentage of the measurement update) of a certain state/parameter is considered, which corresponds to the evolution and observation processes and the filter application. Herein, we adopt the PKF to estimate PMSM states and parameters. Filter equations are obtained by a slight modification of the measurement update stage of the EKF equations. This enables handling a wider range of uncertainties and nonlinearities as compared to the standard EKF, while maintaining its simplicity and low computational costs [85].

In practice, not all machine states are observable from its terminal measurements. Moreover, measured signals can be noisy, or lost due to faulty sensory devices.

This intermittency in observed data can significantly degrade the estimation performance [86]. Stability and convergence of the estimation process for linear, extended, unscented, and Cubature Kalman filters in the presence of intermittent observations have been studied in [87, 88, 89, 90]. To the best of our knowledge, there are no results reported in the literature on stability and convergence properties of PKF subject to intermittent measurements. The salient contributions of this paper are summarized as follows

- The PKF is implemented for state and parameter estimation problems in a PMSM. The estimation algorithm is further extended to accommodate missing measured data.
- The error dynamics of the proposed filter are analyzed. Expression on the critical arrival rate of measurement data is derived under which the filter is stable and accurate.
- This method is verified through simulated and experimental studies for PMSM under varying operating conditions.

This paper is arranged as follows. Section 4.2 provides the notations and preliminaries. Section 4.3 summarizes machine models, their augmentation, and their discretization. In Section 4.4, the implementation of EKF and PKF, as well as incorporating measurement losses, are studied. In Section 4.5, the feasibility of PKF to perform machine estimates despite measurement loss is investigated by bounding the estimation and covariance matrix errors. Section 4.6 investigates the theoretical results for a PMSM using both numerical simulation and hardware measurement. Section 4.7 provides summary remarks.

4.2 Notation and Preliminaries

$\mathbb{R}$ denotes the field of real numbers, and $\mathbb{R}^n$ represents the set of n-dimensional real vectors. Given a matrix $\mathbf{A}$, let $\mathbf{A}^\top$ and $\mathbf{A}^{-1}$ denote, respectively, its transpose and its inverse. The identity matrix with dimension n is denoted by $\mathbf{I}_n$. $\|.\|$ stands for the euclidean norm of vectors and the spectral norm of matrices, and $\|.\|_2$ denotes the L^2-norm of vectors. $\mathbb{E}[x]$ represents the expectation (expected value) of x. $\lambda_{\min}(.)$ and $\lambda_{\max}(.)$, respectively, denote the smallest and the largest eigenvalues of a real matrix.

The following lemmas and theorems will be needed later.

Lemma 1: [89] For any two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, and the scalar $\vartheta > 0$, the following inequality holds

$$\mathbf{x}\mathbf{y}^\top + \mathbf{y}\mathbf{x}^\top \leq \vartheta \mathbf{x}\mathbf{x}^\top + \vartheta^{-1}\mathbf{y}\mathbf{y}^\top. \tag{4.1}$$

Lemma 2: [91] Assume that matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{B} \in \mathbb{R}^{m\times n}$, and $\mathbf{C} \in \mathbb{R}^{n\times n}$, if $\mathbf{A} > 0$ and $\mathbf{C} > 0$, then

$$\mathbf{A}^{-1} > \mathbf{B}(\mathbf{B}^\top \mathbf{A}\mathbf{B} + \mathbf{C})^{-1}\mathbf{B}^\top. \tag{4.2}$$

Theorem 1. *If a stochastic process $V_k(\boldsymbol{\zeta}_k)$ satisfies the following conditions:*

$$\underline{v}||\boldsymbol{\zeta}_k||^2 \leq V_k(\boldsymbol{\zeta}_k) \leq \bar{v}||\boldsymbol{\zeta}_k||^2, \tag{4.3}$$

$$\mathbb{E}[V_k(\boldsymbol{\zeta}_k)] - V_{k-1}(\boldsymbol{\zeta}_{k-1}) \leq \mu - \alpha V_{k-1}(\boldsymbol{\zeta}_{k-1}), \tag{4.4}$$

where scalars $\underline{v}, \bar{v}, \mu > 0$ and $0 < \alpha \leq 1$, then the stochastic process is exponentially bounded in a mean-square sense, i.e. $\mathbb{E}[\|\boldsymbol{\zeta}_k\|^2] \leq \frac{\bar{v}}{\underline{v}}\mathbb{E}[\|\boldsymbol{\zeta}_0\|^2](1-\alpha)^k + \frac{\mu}{\underline{v}}\sum_{i=1}^{k-1}(1-\alpha)^i$.

Proof: Please see [92, 90].

4.3 Augmented Models of Electric Machines

Herein, we discuss the dynamic model of a PMSM for use in the estimation algorithm, its augmentation, and discretization.

4.3.1 Permanent-magnet Synchronous Motor Model

PMSM model in q/d-reference frame is as follows [93]

$$L_d \frac{di_{ds}}{dt} = v_{ds} - R_s i_{ds} + P_p \omega_m L_q i_{qs}, \tag{4.5a}$$

$$L_q \frac{di_{qs}}{dt} = v_{qs} - R_s i_{qs} - P_p \omega_m L_d i_{ds} - P_p \omega_m \lambda_m, \tag{4.5b}$$

$$J_m \frac{d\omega_m}{dt} = T_e - d_m \omega_m - T_L, \tag{4.5c}$$

$$\frac{d\theta_m}{dt} = \omega_m, \tag{4.5d}$$

$$T_e = \frac{3P_p}{2}(\lambda_m i_{qs} + (L_d - L_q) i_{qs} i_{ds}). \tag{4.5e}$$

i_{qs}, i_{ds}, v_{qs}, and v_{ds} are the currents and voltages in the q/d-reference frame. R_s is the stator resistance, L_d and L_q are the q/d-axes inductances, λ_m is the flux linkage of the permanent magnet, and P_p represents the pole-pairs. The mechanical subsystem states are the rotor speed ω_m and rotor position θ_m. Lastly, J_m, d_m, T_e, and T_L are the inertia, friction coefficient, electromagnetic torque, and load torque, respectively.

The PMSM model in stationary axes (α/β-reference frame) is favorable in speed sensorless applications, as the stator currents and voltages measurements (in *abc*-

reference) are easily transformed to stationary axes representation. In such a case, the PMSM states are $[i_{s\alpha}, i_{s\beta}, \omega_m, \theta_m]^\top$, with

$$\begin{bmatrix} f_d \\ f_q \end{bmatrix} = \begin{bmatrix} \cos(P_p\theta_m) & \sin(P_p\theta_m) \\ -\sin(P_p\theta_m) & \cos(P_p\theta_m) \end{bmatrix} \begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix}. \tag{4.6}$$

$[f_d, f_q]^\top$ and $[f_\alpha, f_\beta]^\top$ denote currents and voltages in q/d- and α/β-reference frames, respectively. As per (4.6), the voltage equations for PMSM in α/β-reference frame becomes

$$L_s \frac{di_{s\alpha}}{dt} = v_{s\alpha} - R_s i_{s\alpha} + \lambda_m P_p \omega_m \sin(P_p\theta_m), \tag{4.7a}$$

$$L_s \frac{di_{s\beta}}{dt} = v_{s\beta} - R_s i_{s\beta} - \lambda_m P_p \omega_m \cos(P_p\theta_m). \tag{4.7b}$$

In this chapter, we consider a surface-mounted PMSM where inductances $L_s = L_q = L_d$. The state equations for rotor speed ω_m and position θ_m take the same form as (4.5c) and (4.5d), with a modified formulation for T_e. The complete PMSM model in stationary axes can be found in [93].

4.3.2 Model Augmentation for Parameter Estimation

Machine dynamics in (4.5) can be compactly written as $\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}(\mathbf{x}(t), \mathbf{u}(t))$, where $\mathbf{x}(t)$ and $\mathbf{u}(t)$ denote the system states and inputs, respectively. For the PMSM model in (4.5), it is straightforward to see that $\mathbf{x} \triangleq [i_{qs}, i_{ds}, \omega_m, \theta_m]^\top$ and $\mathbf{u} \triangleq [v_{qs}, v_{ds}]^\top$. To facilitate parameter estimation via Kalman filters, machine models are usually augmented with additional state equations representing the parameters of interest [53]. Let $\mathbf{p}$ denote the set of parameters that need to be estimated. Assuming

time-invariant or slowly-varying parameter values, their dynamics can be written as $\frac{d\mathbf{p}}{dt} \approx 0$. Hence, the overall system considered for estimation becomes

$$\frac{d}{dt}\begin{bmatrix}\mathbf{x}(t)\\ \mathbf{p}\end{bmatrix} = \begin{bmatrix}\mathbf{F}(\mathbf{x}(t),\mathbf{u}(t))\\ \mathbf{0}\end{bmatrix}. \tag{4.8}$$

For consistency, we have slightly abused notations by using $\mathbf{F}(\mathbf{x}(t),\mathbf{u}(t))$ instead of the more accurate $\mathbf{F}(\mathbf{x}(t),\mathbf{p},\mathbf{u}(t))$.

4.3.3 Discretization of Augmented Machine Equations

The augmented system (4.8) needs to be discretized to obtain a formulation suited for the Kalman filter. Herein, the forward Euler method is used for its simplicity. In this chapter, we intend to present estimator formulations that can be easily extended to any machine type. Hence, the Kalman filter equations are written for the following general discrete-time system

$$\begin{cases}\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k,\mathbf{w}_k),\\ \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k,\mathbf{v}_k).\end{cases} \tag{4.9}$$

Interested readers can refer to [94] for details on formulating (4.8) into (4.9) after discretization. Variables $\mathbf{x}_k \in \mathbb{R}^n$, $\mathbf{u}_k \in \mathbb{R}^p$, and $\mathbf{y}_k \in \mathbb{R}^m$ are the state, input, and measurement vectors, respectively, at time instance k. Continuously differentiable functions $\mathbf{f}$ and $\mathbf{h}$ represent the system and output models, respectively. $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0},\mathbf{Q}_k)$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0},\mathbf{R}_k)$ denote process and measurement noise, which are assumed uncorrelated. Noise is inherently present in actual physical systems, hence, it is incorporated within the model (4.9). Process noise $\mathbf{w}_k$ accounts for the modeling

inaccuracies as well as external disturbance inputs, while the measurement noise $\mathbf{v}_k$ represents measurement/sensing error in the output signals.

4.4 Partial-update Kalman Filter with Intermittent Observations

In this section, we first describe a summary of the extended Kalman filtering framework in the presence of intermittent observations. Then, we establish the filter equations for the PKF without and with intermittent measurements.

4.4.1 Extended Kalman Filter

In a Kalman filter and its nonlinear variant, EKF, the state estimates are performed in a recursive fashion using two stages, *time update* (also known as *a priori*) and *measurement update* (also known as *a posteriori*). In the time-update stage, an estimate of the state $\mathbf{x}_k$ is denoted as $\hat{\mathbf{x}}_{k|k-1}$, which represents the estimate at the k^{th} time instance based on measurements $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k-1}\}$. In the measurement update stage, the estimate of the state $\mathbf{x}_k$ is denoted as $\hat{\mathbf{x}}_{k|k}$ which represents the estimate based on measurements $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k-1}, \mathbf{y}_k\}$. Given that Kalman filter is designed for linear systems, formulations for EKF require linearization of the system model at the operating condition. Since the functions $\mathbf{f}$ at $(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0})$ and $\mathbf{h}$ at $(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0})$ are continuously differentiable, they can be written using Taylor expansion as

$$\begin{aligned}\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) &= \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0}) + \mathbf{A}_k(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}) + \mathbf{G}_k\mathbf{w}_k \\ &\quad + \boldsymbol{\phi}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{w}_k), \qquad (4.10)\\ \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) &= \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0}) + \mathbf{C}_k(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}) + \mathbf{D}_k\mathbf{v}_k \\ &\quad + \boldsymbol{\mathcal{X}}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k-1}, \mathbf{v}_k), \qquad (4.11)\end{aligned}$$

where $\mathbf{A}_k = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0})$, $\mathbf{C}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0})$, $\mathbf{G}_k = \frac{\partial \mathbf{f}}{\partial \mathbf{w}}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0})$, and $\mathbf{D}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{v}}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0})$. $\boldsymbol{\phi}$ and $\boldsymbol{\mathcal{X}}$ denote functions representing higher order terms. Thereafter, the time update and measurement update equations for the EKF can be written as [95]

4.4.1.1 Time Update

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0}), \tag{4.12}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}_k \mathbf{P}_{k|k} \mathbf{A}_k^\top + \mathbf{Q}_k. \tag{4.13}$$

4.4.1.2 Measurement Update

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1|k} \mathbf{C}_{k+1}^\top \left[\mathbf{C}_{k+1} \mathbf{P}_{k+1|k} \mathbf{C}_{k+1}^\top + \mathbf{R}_{k+1} \right]^{-1}, \tag{4.14}$$

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_{k+1} \big(\mathbf{y}_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}, \mathbf{0}) \big), \tag{4.15}$$

$$\mathbf{P}_{k+1|k+1} = (\mathbf{I}_n - \mathbf{K}_{k+1} \mathbf{C}_{k+1}) \mathbf{P}_{k+1|k}. \tag{4.16}$$

Matrices $\mathbf{P}_{k+1|k} = \mathbb{E}[(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k})(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k})^\top]$ and $\mathbf{P}_{k+1|k+1} = \mathbb{E}[(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k+1})(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k+1})^\top]$ denote the estimation error covariance matrix for the time update and measurement update, respectively. $\mathbf{K}_k$ denotes the Kalman gain. $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the covariances for the process and measurement noises, as discussed in Section 4.3.3. The results from (4.15) and (4.16) are taken as the outputs from the EKF.

4.4.2 EKF with Intermittent Loss in Measurements

Herein, we consider a case in which estimation needs to be performed under lossy measurements or intermittent observations [87]. In such cases, the arrival of the observation at time k is defined as a Bernoulli process γ_k with the parameter κ, where probability $Pr\{\gamma_k = 1\} = \kappa$, hence, $Pr\{\gamma_k = 0\} = 1 - \kappa$. In order to incorporate

intermittent observations, the measurement update equations in EKF are modified as [87]

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \gamma_{k+1}\mathbf{K}_{k+1}\big(\mathbf{y}_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}, \mathbf{0})\big), \tag{4.17}$$

$$\mathbf{P}_{k+1|k+1} = (\mathbf{I}_n - \gamma_{k+1}\mathbf{K}_{k+1}\mathbf{C}_{k+1})\mathbf{P}_{k+1|k}. \tag{4.18}$$

Note that to obtain the above equations, it has been assumed that the actual value of γ_{k+1} along with the measurement y_{k+1} are being transmitted to the filter [87, 88].

4.4.3 Partial-update Kalman Filter

The PKF finds the estimates for states and the estimation error covariance matrix as a convex combination of results from the *time update* and the *measurement update* of the EKF. For every time instance k, the element-wise representation for the PKF is [84, 85]

$$\hat{\mathbf{x}}^i_{\text{pkf}} = \rho_i \hat{\mathbf{x}}^i_{k+1|k} + (1 - \rho_i)\hat{\mathbf{x}}^i_{k+1|k+1}, \tag{4.19}$$

$$\mathbf{P}^{ij}_{\text{pkf}} = \rho_i \rho_j \mathbf{P}^{ij}_{k+1|k} + (1 - \rho_i\rho_j)\mathbf{P}^{ij}_{k+1|k+1}. \tag{4.20}$$

$\hat{\mathbf{x}}^i_{\text{pkf}}$ is the state estimate via partial-update Kalman filter, where superscript i denotes the i^{th} element of the state vector. $\mathbf{P}^{ij}_{\text{pkf}}$ is the estimation error covariance matrix with superscript ij representing the matrix element in the i^{th} row and j^{th} column. $\rho_i = 1 - \beta_i$, where $\beta_i \in [0, 1]$ is the weight percentage of the measurement update used. For $\beta_i = 0$ and $\beta_i = 1$, the PKF reduces to the Schmidt (with no measurement update) and the EKF (with full measurement update), respectively.

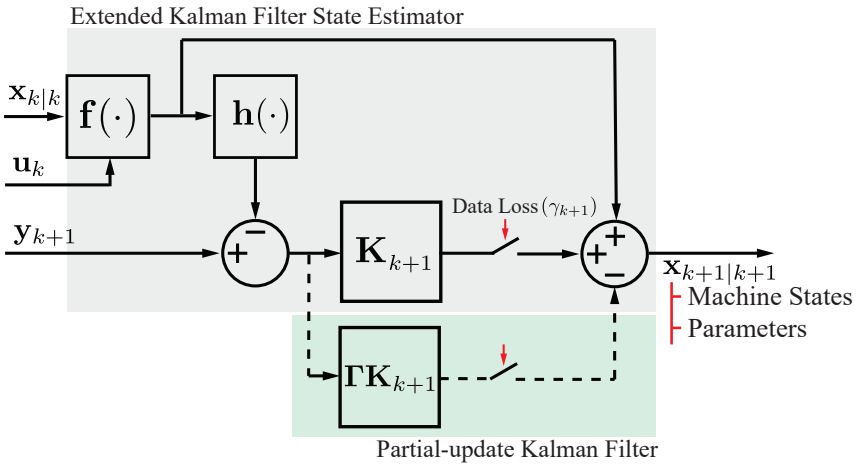


Figure 20: Overview of the PKF. Weighted average of the time and measurement updates in (4.19) is the same as an additional term (highlighted green) in the conventional extended Kalman filter (see (4.25)).

Note that the PKF formulations has a three-stage form compared to the EKF which involves only two stages. Equations (4.19) and (4.20) can be reorganized in order to give the PKF a two-stage form. Vectorizing (4.19) while using (4.15) results in

$$\begin{aligned}\hat{\mathbf{x}}_{\text{pkf}} &= \mathbf{\Gamma}\hat{\mathbf{x}}_{k+1|k} + (\mathbf{I}_n - \mathbf{\Gamma})\hat{\mathbf{x}}_{k+1|k+1},\\ &= \mathbf{\Gamma}\hat{\mathbf{x}}_{k+1|k}+(\mathbf{I}_n-\mathbf{\Gamma})\Big(\hat{\mathbf{x}}_{k+1|k}+\mathbf{K}_{k+1}\big(\mathbf{y}_{k+1}-\mathbf{h}(\hat{\mathbf{x}}_{k+1|k},\mathbf{0})\big)\Big),\\ &= \hat{\mathbf{x}}_{k+1|k}+\tilde{\mathbf{K}}_{k+1}\big(\mathbf{y}_{k+1}-\mathbf{h}(\hat{\mathbf{x}}_{k+1|k},\mathbf{0})\big),\end{aligned} \tag{4.21}$$

where $\tilde{\mathbf{K}}_k = (\mathbf{I}_n - \mathbf{\Gamma})\mathbf{K}_k$ and $\mathbf{\Gamma} = \text{diag}\{\rho_1, \rho_2, \ldots, \rho_n\}$. Similarly, (4.20) can be vectorized as (see Section III in [85])

$$\mathbf{P}_{\text{pkf}} = (\mathbf{I}_n - \mathbf{K}_{k+1}\mathbf{C}_{k+1})\mathbf{P}_{k+1|k} + \mathbf{\Gamma}\mathbf{K}_{k+1}\mathbf{C}_{k+1}\mathbf{P}_{k+1|k}\mathbf{\Gamma}. \tag{4.22}$$

Note that (4.21) and (4.22) resemble the *measurement update* form of the EKF in (4.15) and (4.16), respectively, and are functions of *time update* terms. In

the following sections, we will replace the subscript 'pkf' with $k+1|k+1$ in (4.21) and (4.22) to have a formulation similar to EKF.

4.4.4 PKF with Intermittent Measurement

In this chapter, we extend the PKF in [84, 85] to incorporate losses within measurement signals. Taking the EKF with measurement loss (4.17), (4.18), and combining them with the single update form of the PKF (4.21), (4.22), the PKF for measurement loss can be formulated as

4.4.4.1 Time Update

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{0}), \tag{4.23}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}_k \mathbf{P}_{k|k} \mathbf{A}_k^\top + \mathbf{Q}_k. \tag{4.24}$$

4.4.4.2 Measurement Update

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \gamma_{k+1} \tilde{\mathbf{K}}_{k+1} \big(\mathbf{y}_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}, \mathbf{0}) \big), \tag{4.25}$$

$$\begin{aligned} \mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} &- \gamma_{k+1} \mathbf{K}_{k+1} \mathbf{C}_{k+1} \mathbf{P}_{k+1|k} \\ &+ \gamma_{k+1} \mathbf{\Gamma} \mathbf{K}_{k+1} \mathbf{C}_{k+1} \mathbf{P}_{k+1|k} \mathbf{\Gamma}. \end{aligned} \tag{4.26}$$

Fig. 20 shows the block diagram for the PKF. Given $\tilde{\mathbf{K}}_k = (\mathbf{I}_n - \mathbf{\Gamma})\mathbf{K}_k$, the PKF can be obtained by adding an additional block (shaded green in Fig. 20) within the conventional EKF.

4.5 Bounds on the Estimation Error and the Error Covariance Matrix

In this section, we investigate the feasibility of the PKF (4.23) – (4.26) for estimation despite measurement loss.

4.5.1 Error Dynamics of the PKF

The *a priori* and *a posteriori* estimation errors can be defined as $\mathbf{e}_{k+1|k} = (\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k})$ and $\mathbf{e}_{k+1|k+1} = (\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k+1})$, respectively. Per the formulations of the PKF in (4.23) and (4.25), and Taylor expansions in (4.10) and (4.11), the estimation error dynamics can be formulated as

$$\begin{aligned}\mathbf{e}_{k|k} = (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k)\mathbf{e}_{k|k-1} - \gamma_k \tilde{\mathbf{K}}_k \mathbf{D}_k \mathbf{v}_k \\ - \gamma_k \tilde{\mathbf{K}}_k \boldsymbol{\mathcal{X}}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k-1}, \mathbf{v}_k),\end{aligned} \tag{4.27}$$

$$\mathbf{e}_{k+1|k} = \mathbf{A}_k \mathbf{e}_{k|k} + \mathbf{G}_k \mathbf{w}_k + \boldsymbol{\phi}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{w}_k). \tag{4.28}$$

By combining (4.27) and (4.28), it can be obtained that

$$\mathbf{e}_{k+1|k} = \mathbf{A}_k(\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k)\mathbf{e}_{k|k-1} + r_k + s_k, \tag{4.29}$$

where

$$\begin{cases} r_k \triangleq \boldsymbol{\phi}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{w}_k) - \gamma_k \mathbf{A}_k \tilde{\mathbf{K}}_k \boldsymbol{\mathcal{X}}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k-1}, \mathbf{v}_k), \\ s_k \triangleq \mathbf{G}_k \mathbf{w}_k - \gamma_k \mathbf{A}_k \tilde{\mathbf{K}}_k \mathbf{D}_k \mathbf{v}_k. \end{cases} \tag{4.30}$$

We now present the conditions that ensure that the estimation error and the prediction error covariance matrix from the PKF remain bounded, under the following two assumptions.

Assumption 1. There exist positive scalars $\bar{a}, \bar{c}, \bar{d}, \bar{g}, \bar{\Gamma}, \bar{q}, \bar{r}, \underline{q}$, and $\underline{r}$, such that $\|\mathbf{A}_k\| \leq \bar{a}, \|\mathbf{C}_k\| \leq \bar{c}, \|\mathbf{D}_k\| \leq \bar{d}, \|\mathbf{G}_k\| \leq \bar{g}, \|\mathbf{\Gamma}_k\| \leq \bar{\Gamma}$ and $\underline{q}\mathbf{I}_n \leq \mathbf{Q}_k \leq \bar{q}\mathbf{I}_n, \underline{r}\mathbf{I}_m \leq \mathbf{R}_k \leq \bar{r}\mathbf{I}_m$.

Assumption 2. There exist positive constant scalars $\bar{p}$ and $\underline{p}$ such that $\underline{p}\mathbf{I}_n \leq \mathbf{P}_{k+1|k+1} \leq \mathbf{P}_{k+1|k} \leq \bar{p}\mathbf{I}_n$.

Moreover, we assume that the residuals of the Taylor expansion in (4.10) and (4.11) are bounded in a neighborhood of the point of expansion. Indeed, for all $\|\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}\|_2 \leq \delta_\phi$ and $\|\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\|_2 \leq \delta_{\mathcal{X}}$, there holds

$$\begin{cases} \|\boldsymbol{\phi}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k}, \mathbf{u}_k, \mathbf{w}_k)\|_2 \leq \epsilon_\phi \|\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}\|_2^2, \\ \|\boldsymbol{\mathcal{X}}(\mathbf{x}_k, \hat{\mathbf{x}}_{k|k-1}, \mathbf{v}_k)\|_2 \leq \epsilon_{\mathcal{X}} \|\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\|_2^2. \end{cases} \tag{4.31}$$

where δ_ϕ, $\delta_{\mathcal{X}}$, ϵ_ϕ, $\epsilon_{\mathcal{X}}$ are bounded positive real numbers.

4.5.2 Boundedness of the Estimation Error

The main goal of this section is to show that the estimation error is stocastically bounded.

Theorem 2. *Consider the proposed PKF (4.23) – (4.26). If assumptions 1 and 2 are satisfied, there exists a real scalar* $0 < \alpha < 1$ *such that*

$$(\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k) \leq (1-\alpha)\mathbf{P}_{k|k-1}^{-1}. \tag{4.32}$$

Proof: From (4.24), we have [88]

$$\mathbf{P}_{k+1|k} = \mathbf{A}_k \mathbf{P}_{k|k} \mathbf{A}_k^\top + \mathbf{Q}_k > \Big(1 + \frac{\underline{q}}{2\bar{a}^2 + \bar{p}}\Big) \mathbf{A}_k \mathbf{P}_{k|k} \mathbf{A}_k^\top. \tag{4.33}$$

By adding and subtracting $\gamma_{k+1}\mathbf{P}_{k+1|k}\mathbf{C}_{k+1}^{\top}\mathbf{K}_{k+1}^{\top}$ to the right side of (4.26), and since $\gamma_k = \gamma_k^2$, we can derive

$$\begin{aligned}\mathbf{P}_{k|k} = &(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)\mathbf{P}_{k|k-1}(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)^{\top} \\ &+ \gamma_k\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\top} + \gamma_k\boldsymbol{\Gamma}\mathbf{K}_k\mathbf{C}_k\mathbf{P}_{k|k-1}\boldsymbol{\Gamma}.\end{aligned} \tag{4.34}$$

Multiplying $\left[\mathbf{C}_{k+1}\mathbf{P}_{k+1|k}\mathbf{C}_{k+1}^{\top} + \mathbf{R}_{k+1}\right]\mathbf{K}_k^{\top}$ from right on both sides of (4.14), we can show

$$\mathbf{K}_k\mathbf{C}_k\mathbf{P}_{k|k-1} = \mathbf{K}_k\left[\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^{\top} + \mathbf{R}_k\right]\mathbf{K}_k^{\top}. \tag{4.35}$$

By combining (4.33), (4.34), and (4.35), we have

$$\begin{aligned}\mathbf{P}_{k+1|k} > &\Big(1 + \frac{\underline{q}}{2\bar{a}^2 + \bar{p}}\Big)\mathbf{A}_k\Big((\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)\mathbf{P}_{k|k-1} \\ &(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)^{\top} + \gamma_k\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\top} + \\ &\gamma_k\boldsymbol{\Gamma}\mathbf{K}_k(\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^{\top} + \mathbf{R}_k)\mathbf{K}_k^{\top}\boldsymbol{\Gamma}\Big)\mathbf{A}_k^{\top}.\end{aligned} \tag{4.36}$$

From the fact that $\mathbf{P}_{k+1|k}, \mathbf{R}_{k+1} > 0$, we have

$$\begin{aligned}\mathbf{P}_{k+1|k} > &\Big(1 + \frac{\underline{q}}{2\bar{a}^2 + \bar{p}}\Big)\mathbf{A}_k(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k) \\ &\times \mathbf{P}_{k|k-1}(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)^{\top}\mathbf{A}_k^{\top}.\end{aligned} \tag{4.37}$$

The rest of proof is similar to *Lemma* 6.3 in [88], and is omitted here for brevity. □

Now, we are able to state the main results of this section.

Theorem 3. *Consider the nonlinear system (4.9), and the PKF equations (4.23) – (4.26). If assumptions 1 and 2 are satisfied, and for a bound $\delta > 0$ on the initial*

estimation error, i.e. $\mathbb{E}[\|\mathbf{e}_{1|0}\|^2] \leq \delta$*, the estimation error* $\mathbf{e}_{k+1|k}$ *is exponentially bounded in a mean-square sense.*

Proof: We define a Lyapunov function $V_k(\mathbf{e}_{k|k-1}) = \mathbf{e}_{k|k-1}^\top \mathbf{P}_{k|k-1}^{-1} \mathbf{e}_{k|k-1}$. As per (4.29), this function becomes

$$\begin{aligned}
&V_{k+1}(\mathbf{e}_{k+1|k}) = \\
&\mathbf{e}_{k|k-1}^\top (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k) \mathbf{e}_{k|k-1} \\
&+ r_k^\top \mathbf{P}_{k+1|k}^{-1} [2\mathbf{A}_k (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k) \mathbf{e}_{k|k-1} + r_k] \\
&+ s_k^\top \mathbf{P}_{k+1|k}^{-1} [2\mathbf{A}_k (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k) \mathbf{e}_{k|k-1} + 2r_k + s_k]. \qquad (4.38)
\end{aligned}$$

The Lyapunov function in (4.38) can be rewritten as

$$\begin{aligned}
&V_{k+1}(\mathbf{e}_{k+1|k}) = \mu_{1k} + \mu_{2k} + \mu_{3k} + \qquad (4.39) \\
&\mathbf{e}_{k|k-1}^\top (\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k) \mathbf{e}_{k|k-1},
\end{aligned}$$

where

$$\begin{aligned}
\mu_{1k} &= \mathbf{e}_{k|k-1}^\top [(\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\gamma_k \boldsymbol{\Gamma} \mathbf{K}_k \mathbf{C}_k)] \mathbf{e}_{k|k-1} \\
&\quad + \mathbf{e}_{k|k-1}^\top [(\gamma_k \boldsymbol{\Gamma} \mathbf{K}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\mathbf{I}_n - \gamma_k \mathbf{K}_k \mathbf{C}_k)] \mathbf{e}_{k|k-1} \\
&\quad + \mathbf{e}_{k|k-1}^\top [(\gamma_k \boldsymbol{\Gamma} \mathbf{K}_k \mathbf{C}_k)^\top \mathbf{A}_k^\top \mathbf{P}_{k+1|k}^{-1} \mathbf{A}_k (\gamma_k \boldsymbol{\Gamma} \mathbf{K}_k \mathbf{C}_k)] \mathbf{e}_{k|k-1}, \\
\mu_{2k} &= r_k^\top \mathbf{P}_{k+1|k}^{-1} [2\mathbf{A}_k (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k) \mathbf{e}_{k|k-1} + r_k], \\
\mu_{3k} &= s_k^\top \mathbf{P}_{k+1|k}^{-1} [2\mathbf{A}_k (\mathbf{I}_n - \gamma_k \tilde{\mathbf{K}}_k \mathbf{C}_k) \mathbf{e}_{k|k-1} + 2r_k + s_k].
\end{aligned}$$

The upper bound of the last expression in (4.39) can be obtained using Theorem 2. Moreover, from Lemma 3.2 and Lemma 3.3 in [92], one can show that μ_{2k} and

μ_{3k} have upper bounds. Substituting (4.32) into $V_{k+1}(\mathbf{e}_{k+1|k})$, taking the conditional expectation, and following the results of [92, 88], we have

$$\mathbb{E}[V_{k+1}(\mathbf{e}_{k+1|k})] \leq (1-\alpha)V_k(\mathbf{e}_{k|k-1}) + \mathbb{E}[\mu_{1k}] + \bar{\mu}_2 + \bar{\mu}_3. \tag{4.40}$$

$\bar{\mu}_2 = \kappa_1\varepsilon_1$, $\bar{\mu}_3 = \kappa_2\varepsilon_2$, κ_1, κ_2 are positive bounds depending on positive real numbers in Assumption 1, Assumption 2, and (4.31), ε_1 is a function of ϵ_ϕ, $\epsilon_\mathcal{X}$, and ε_2 is a positive bound on stochastic noise covariance matrices, i.e., $\mathbb{E}[\mathbf{v}_k\mathbf{v}_k^\top] \leq \varepsilon_2^2\mathbf{I}_m$, $\mathbb{E}[\mathbf{w}_k\mathbf{w}_k^\top] \leq \varepsilon_2^2\mathbf{I}_n$.

It remains to show that $\mathbb{E}[\mu_{1k}]$ is also bounded by a positive constant. Both sides of $\mathbb{E}[\mu_{1k}]$ are scalars. Using Lemma 1 and computing the trace of $\mathbb{E}[\mu_{1k}]$, and by considering $\mathbb{E}[\mathbf{e}_{k|k-1}\mathbf{e}_{k|k-1}^\top] = \mathbf{P}_{k|k-1}$, we have

$$\begin{aligned}\mathbb{E}[\mu_{1k}] \leq tr[&\vartheta\mathbf{M}_k(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)\mathbf{P}_{k|k-1}(\mathbf{I}_n - \gamma_k\mathbf{K}_k\mathbf{C}_k)^\top\mathbf{M}_k \\ &+ \vartheta^{-1}(\gamma_k\mathbf{\Gamma}\mathbf{K}_k\mathbf{C}_k)^\top\mathbf{P}_{k|k-1}(\gamma_k\mathbf{\Gamma}\mathbf{K}_k\mathbf{C}_k) \\ &+ (\gamma_k\mathbf{\Gamma}\mathbf{K}_k\mathbf{C}_k)^\top\mathbf{M}_k(\gamma_k\mathbf{\Gamma}\mathbf{K}_k\mathbf{C}_k)\mathbf{P}_{k|k-1}],\end{aligned} \tag{4.41}$$

where $\mathbf{M}_k = \mathbf{A}_k^\top\mathbf{P}_{k+1|k}^{-1}\mathbf{A}_k$. It is clear that $\mathbb{E}[\mu_{1k}]$ has a positive upper bound $\bar{\mu}_1$. Then, (4.40) can be extended as

$$\mathbb{E}[V_{k+1}(\mathbf{e}_{k+1|k})] - V_k(\mathbf{e}_{k|k-1}) \leq \bar{\mu} - \alpha V_k(\mathbf{e}_{k|k-1}), \tag{4.42}$$

where $\bar{\mu} = \bar{\mu}_1 + \bar{\mu}_2 + \bar{\mu}_3$.

Equation (4.42) takes the form similar to a bounded stochastic process in Theorem 1. Hence, the proposed PKF for measurement loss will be bounded in the mean-square sense. □

The proof of the Theorem 3 is based on the boundedness assumption of the estimation error covariance matrix. However, this assumption can be violated when

$\gamma_k = 0$ for infinitely many k [88]. Consequently, the behavior of the error covariance matrix will be analyzed, and an upper bound for its expectation in presence of measurement losses will be derived.

4.5.3 Boundedness of the Error Covariance Matrix

In the following theorem, the boundedness of the error covariance matrix will be shown.

Theorem 4. *Suppose that Assumption 1 holds, and the linearized form of the nonlinear system (4.9) satisfies a modified uniform observability condition as in [88]. Then, there exists a constant $\bar{p} > 0$, such that*

$$\mathbb{E}[\mathbf{P}_{k+1|k}] \leq \bar{p}\mathbf{I}_n, \tag{4.43}$$

provided that

- $\mathbf{C}_k^{-1}$ *exists and satisfies* $\|\mathbf{C}_k^{-1}\| \leq \underline{c}^{-1}$, *and the measurement data arrival probability satisfies* $\gamma > 1 - \frac{\bar{a}^{-2} - \bar{\Gamma}^2}{1 - \bar{\Gamma}^2}$,

or

- *the data arrival probability satisfies* $\gamma > \frac{1 - \bar{a}^{-2}}{\lambda - \bar{\Gamma}^2}$, *where* $0 < \lambda < \lambda_0 \leq 1$, *and* λ_0 *will be later defined in (4.52).*

Proof: Two different cases are considered.

<u>Case 1:</u> **The $\mathbf{C}_k$ matrix is invertible.** By combining (4.14), (4.24) and (4.26), the prediction error covariance matrix becomes

$$\begin{aligned}\mathbf{P}_{k+1|k} = &\mathbf{A}_k\big(\mathbf{P}_{k|k-1} - \gamma_k \mathbf{P}_{k|k-1}\mathbf{C}_k^\top \left[\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \\ &\mathbf{C}_k\mathbf{P}_{k|k-1} + \gamma_k \mathbf{\Gamma}\mathbf{P}_{k|k-1}\mathbf{C}_k^\top \left[\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \\ &\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{\Gamma}\big)\mathbf{A}_k^\top + \mathbf{Q}_k.\end{aligned} \tag{4.44}$$

Consider $\gamma_k \mathbf{\Gamma} \mathbf{P}_{k|k-1} \mathbf{C}_k^\top \left[\mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{\Gamma}$ in (4.44). Using the matrix inversion lemma $(\mathbf{A}+\mathbf{B})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}\mathbf{A}^{-1}$, and defining $\mathbf{A} \triangleq \mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top$ and $\mathbf{B} \triangleq \mathbf{R}_k$, we derive

$$\begin{aligned}\mathbf{P}_{k+1|k} = \mathbf{A}_k \big(&\mathbf{P}_{k|k-1} - \gamma_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top \left[\mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \\ &\mathbf{C}_k \mathbf{P}_{k|k-1} + \gamma_k \mathbf{\Gamma} \big(\mathbf{P}_{k|k-1} - \mathbf{C}_k^{-1} \left[\mathbf{C}_k^{-\top} \mathbf{P}_{k|k-1}^{-1} \mathbf{C}_k^{-1} + \mathbf{R}_k^{-1}\right]^{-1} \\ &\mathbf{C}_k^{-\top}\big) \mathbf{\Gamma}\big) \mathbf{A}_k^\top + \mathbf{Q}_k. \end{aligned} \tag{4.45}$$

Since $-\mathbf{\Gamma} \mathbf{C}_k^{-1} \left[\mathbf{C}_k^{-\top} \mathbf{P}_{k|k-1}^{-1} \mathbf{C}_k^{-1} + \mathbf{R}_k^{-1}\right]^{-1} \mathbf{C}_k^{-\top} \mathbf{\Gamma}$ is a symmetric negative term, there exist an $\epsilon > 0$ such that

$$\begin{aligned}\mathbf{P}_{k+1|k} \leq \mathbf{A}_k \big(&\mathbf{P}_{k|k-1} - \gamma_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top \left[\mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \\ &\mathbf{C}_k \mathbf{P}_{k|k-1} + \gamma_k \mathbf{\Gamma} \mathbf{P}_{k|k-1} \mathbf{\Gamma} + \gamma_k \epsilon \mathbf{I}_n \big) \mathbf{A}_k^\top + \mathbf{Q}_k. \end{aligned} \tag{4.46}$$

Using the inequality $(\mathbf{A}+\mathbf{B})^{-1} > \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1}$, and defining $\mathbf{A} \triangleq \mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top$ and $\mathbf{B} \triangleq \mathbf{R}_k$, we have [90]

$$\begin{aligned}\mathbf{P}_{k+1|k} \leq (1-\gamma_k) \mathbf{A}_k \mathbf{P}_{k|k-1} \mathbf{A}_k^\top + \gamma_k \mathbf{A}_k \mathbf{C}_k^{-1} \mathbf{R}_k \mathbf{C}_k^{-\top} \mathbf{A}_k^\top \\ + \gamma_k \mathbf{A}_k \mathbf{\Gamma} \mathbf{P}_{k|k-1} \mathbf{\Gamma} \mathbf{A}_k^\top + \gamma_k \epsilon \mathbf{A}_k \mathbf{A}_k^\top + \mathbf{Q}_k. \end{aligned} \tag{4.47}$$

Considering the bounds of matrices in Assumption 1,

$$\begin{aligned}\mathbf{P}_{k+1|k} \leq (1 - \gamma_k + \gamma_k \bar{\Gamma}^2) \mathbf{A}_k \mathbf{P}_{k|k-1} \mathbf{A}_k^\top \\ + \gamma_k (\bar{r}/\bar{c}^2 + \epsilon) \mathbf{A}_k \mathbf{A}_k^\top + \bar{q} \mathbf{I}_n. \end{aligned} \tag{4.48}$$

Define $\gamma = \mathbb{E}[\gamma_k]$ and assume that $\mathbb{E}[\mathbf{P}_{1|0}] > 0$. Then,

$$\begin{aligned}\mathbb{E}[\mathbf{P}_{2|1}] &\leq \mathbb{E}[(1-\gamma+\gamma\bar{\Gamma}^2)\bar{a}^2\mathbf{P}_{1|0} + \gamma\bar{a}^2(\bar{r}/\bar{c}^2+\epsilon)\mathbf{I}_n + \bar{q}\mathbf{I}_n] \\ &\leq (1-\gamma+\gamma\bar{\Gamma}^2)\bar{a}^2 p\mathbf{I}_n + p\mathbf{I}_n,\end{aligned} \tag{4.49}$$

where $p = \max(\|\mathbf{P}_{1|0}\|, \gamma\bar{a}^2(\bar{r}/\bar{c}^2+\epsilon)+\bar{q})$. Iterating, and by using an induction algorithm, we can show that:

$$\mathbb{E}[\mathbf{P}_{k+1|k}] < p\sum_{j=0}^{k-1}[(1-\gamma+\gamma\bar{\Gamma}^2)\bar{a}^2]^j\mathbf{I}_n. \tag{4.50}$$

The upper bound in (4.43) is concluded from (4.50). Moreover, the sum in (4.50) converges when $\gamma > \frac{1-\bar{a}^{-2}}{1-\bar{\Gamma}^2} = 1 - \frac{\bar{a}^{-2}-\bar{\Gamma}^2}{1-\bar{\Gamma}^2}$.

Case 2: The $\mathbf{C}_k$ matrix is not invertible. Applying Lemma 2 to $\gamma_k\boldsymbol{\Gamma}\mathbf{P}_{k|k-1}\mathbf{C}_k^\top\left[\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1}\mathbf{C}_k\mathbf{P}_{k|k-1}\boldsymbol{\Gamma}$ in (4.44) gives

$$\begin{aligned}\mathbf{P}_{k+1|k} \leq \mathbf{A}_k\big(&\mathbf{P}_{k|k-1} - \gamma_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top\left[\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right]^{-1} \\ &\mathbf{C}_k\mathbf{P}_{k|k-1} + \gamma_k\boldsymbol{\Gamma}\mathbf{P}_{k|k-1}\boldsymbol{\Gamma}\big)\mathbf{A}_k^\top + \mathbf{Q}_k.\end{aligned} \tag{4.51}$$

Based on Lemma 2, we can define $0 < \lambda < \lambda_0 \leq 1$, where

$$\begin{aligned}\lambda_0 = \min\{&\lambda_{\min}[\mathbf{C}_k^\top\left(\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{C}_k]/\lambda_{\max}[\mathbf{P}_{k|k-1}^{-1}], \\ &k = \{1,...,N\}\},\end{aligned} \tag{4.52}$$

such that

$$\lambda\mathbf{P}_{k|k-1}^{-1} < \mathbf{C}_k^\top\left(\mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{C}_k. \tag{4.53}$$

By combining (4.51) and (4.53), we have

$$\mathbf{P}_{k+1|k} \leq \mathbf{A}_k\big(\mathbf{P}_{k|k-1} - \lambda\gamma_k\mathbf{P}_{k|k-1} + \gamma_k\mathbf{\Gamma}\mathbf{P}_{k|k-1}\mathbf{\Gamma}\big)\mathbf{A}_k^\top + \mathbf{Q}_k. \tag{4.54}$$

Consequently, similar to the Case 1, it can be concluded that

$$\mathbb{E}[\mathbf{P}_{k+1|k}] < p\sum_{j=0}^{k-1}[(1-\lambda\gamma+\gamma\bar{\Gamma}^2)\bar{a}^2]^j\mathbf{I}_n, \tag{4.55}$$

where $p = \max(\|\mathbf{P}_{1|0}\|, \bar{q})$. Moreover, (4.55) converges when $\gamma > \frac{1-\bar{a}^{-2}}{\lambda-\bar{\Gamma}^2}$. This completes the proof. □

Remark 1. *The assumption of invertible $\mathbf{C}_k$ could be restrictive. However, in most existing work, e.g., [88, 96, 89, 90], this is a necessary assumption for finding the bound of $\mathbf{P}_{k+1|k}$. Compared to [88, 96, 89, 90], we have shown that, even for the case of singular $\mathbf{C}_k$, the $\mathbf{P}_{k+1|k}$ matrix is bounded and a minimum critical value for data loss exists.*

Remark 2. *For the case of* EKF *($\mathbf{\Gamma} = 0$), and considering the assumption on the invertible $\mathbf{C}_k$ matrix, the critical value γ is $\gamma > 1 - \bar{a}^{-2}$ which corroborates with the results of [88].*

4.6 Validation and Verification

In this section, we show that PKF can outperform the conventional EKF in the estimation process, and validate the PKF using the PMSM hardware. Nominal parameters of the PMSM are provided in [97]. The machine models and Kalman filters are simulated in the Simulink 10.1 environment.

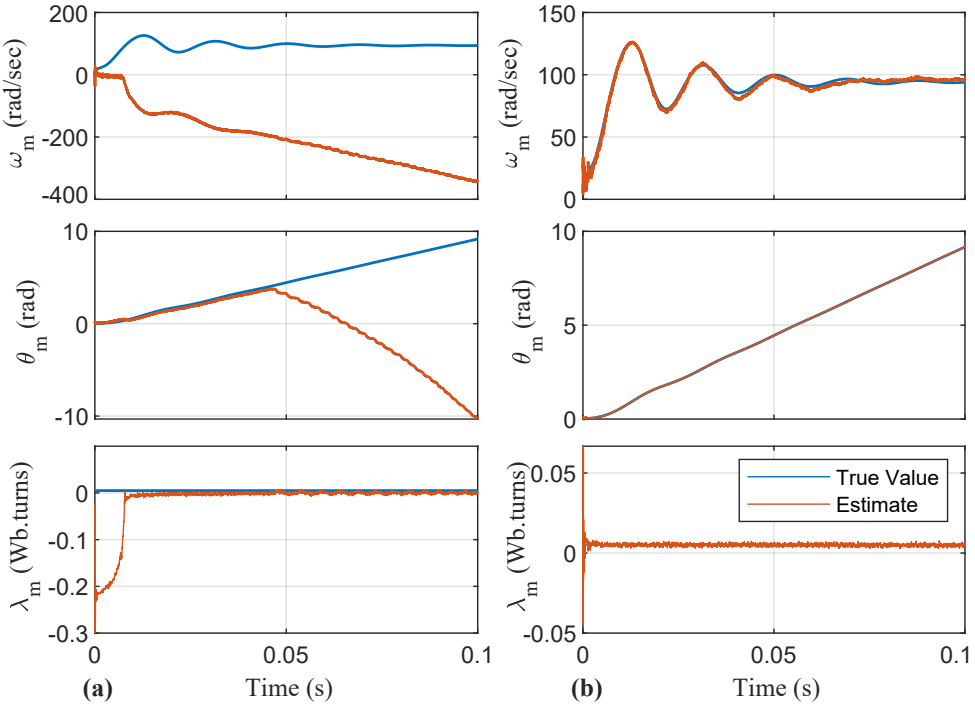


Figure 21: Comparison between estimates from (a) EKF and (b) PKF. EKF stability issues can be addressed in PKF through partial update using $\mathbf{\Gamma}$.

4.6.1 Advantages of PKF over EKF

4.6.1.1 Stability

One of the main drawbacks for EKF is stability issues, e.g., linearization in EKF can induce error in the estimation which could cause instability. We present a simulated case study on the PMSM model, with a speed-sensorless application where an observer needs to estimate rotor speed/position and flux linkage using stator current and voltage measurements. Given that position information is unavailable, the PMSM model in α/β-reference frame is chosen. The PMSM model is simulated with an initial state $[4, 2, 50, 0]^\top$ assuming process noise and measurement noise covariances as $\text{diag}\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$ and $\text{diag}\{10^{-1}, 10^{-1}\}$, respectively. The augmented state vector for both filters is $[i_{s\alpha}, i_{s\beta}, \omega_m, \theta_m, \lambda_m]^\top$ with stator currents $[i_{s\alpha}, i_{s\beta}]^\top$ as out-

put measurements, and stator voltages $[v_{s\alpha}, v_{s\beta}]^\top$ as inputs. The voltage and current signals generated from this simulation are sampled at 10 μs and fed to both EKF and PKF for estimations. To highlight the effectiveness of PKF, we consider an unlikely scenario where both EKF and PKF are designed assuming accurate knowledge of the covariance matrices and initial conditions. Hence, the filters are initialized with an initial augmented state $\mathbf{x}_0 = [4, 2, 50, 0, 0]^\top$, error covariance $\mathbf{P}_0 = \mathbf{I}_5$, measurement noise covariance $\mathbf{R} = \text{diag}\{10^{-1}, 10^{-1}\}$, and the process noise covariance of $\mathbf{Q} = \text{diag}\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}, 10^{-6}\}$. Note that due to model augmentation in the Kalman filter, the sizes of $\mathbf{x}_0$ and $\mathbf{Q}$ have increased (see Section 4.3). Fig. 21 compares the results from both EKF and PKF showing the estimates for speed, position, and magnet flux linkage. The PKF can provide better convergence in such cases due to its ability to control the measurement update with variable $\mathbf{\Gamma}$. We chose $\mathbf{\Gamma} = \text{diag}\{0.2, 0.2, 0.2, 0.2, 0.75\}$, denoting 80% measurement update for system states $[i_{s\alpha}, i_{s\beta}, \omega_m, \theta_m]^\top$ and only 25% measurement update for the slow-varying parameter λ_m [85]. EKF becomes unstable while the PKF maintains its stability due to the partial nature of measurement update.

4.6.1.2 Uncertain Nuisance Parameter

Nuisance parameters are machine parameters that are imprecisely known, and their true values are not necessarily needed to be estimated. However, the uncertainties in these parameters need to be accounted for an overall accurate estimation. A straightforward way to handle this would be to include these parameters in the system state (model augmentation) and estimate them along with the states, although, this could lead to a divergence similar to Fig. 21a. Another approach is to employ the Schmidt Kalman filter, wherein the nuisance parameters are only 'considered' during

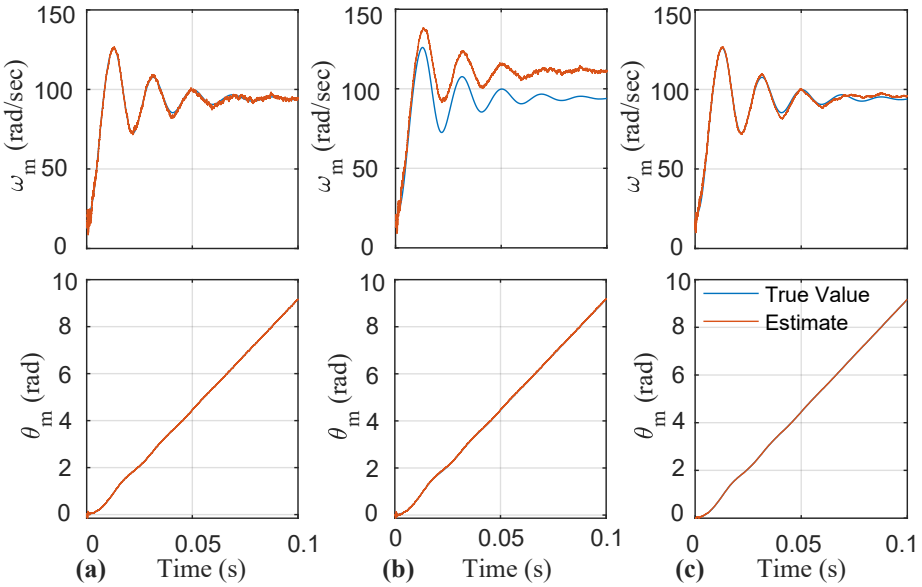


Figure 22: Comparison between estimates from (a) EKF when there is no nuisance parameter in the system (ideal case), (b) EKF with λ_m as nuisance parameter in system, (c) PKF with nuisance parameter λ_m only being 'considered' but not estimated, similar to the Schmidt Kalman filter.

the time update stage of the filter [84]. The PKF can be designed to mimic such properties of the SKF through the variable $\mathbf{\Gamma}$. We present a case to make this point.

Consider a similar scenario where the PMSM model is simulated with initial state $[4, 2, 50, 0]^\top$ with process noise and measurement noise covariances $\text{diag}\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$ and $\text{diag}\{10^{-1}, 10^{-1}\}$, respectively. The objective is to only estimate rotor speed and position from stator currents and voltages. However, there is uncertainty in the parameter λ_m with its value being around 0.004 (mean) with a standard deviation of 0.001. Fig. 22 compares the results from three different filters, assuming accurate knowledge of noise covariance matrices. Fig. 22a shows the estimates from the EKF when there is no nuisance parameter, and λ_m is known accurately. Fig. 22b shows the EKF estimates assum-

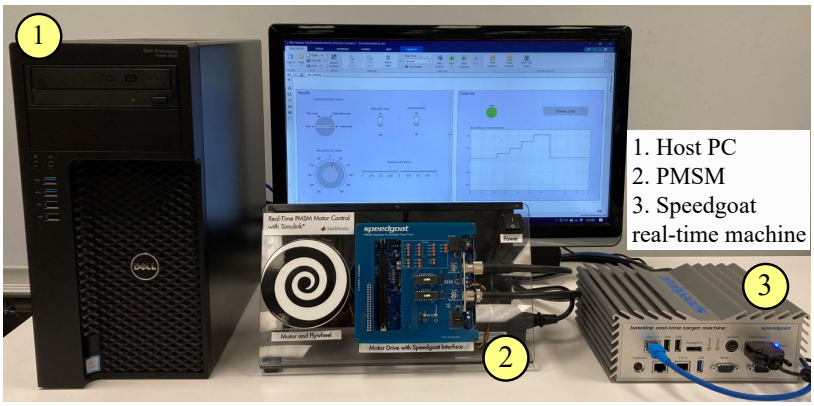


Figure 23: PMSM hardware setup. PMSM is controlled using the FOC speed control implemented on the Speedgoat real-time target machine.

ing $\lambda_m = 0.004$ (ignoring the uncertainty). It is evident from speed estimates in Fig. 22b that the presence of nuisance parameters can deteriorate the Kalman filter's accuracy. In such a scenario, the PKF could be designed that will only 'consider' the uncertainty in λ_m during time update stage. The PKF is initialized with an augmented state $\mathbf{x}_0 = [4, 2, 50, 0, 0.004]^\top$, error covariance $\mathbf{P}_0 = \mathbf{I}_5$, measurement noise covariance $\mathbf{R} = \text{diag}\{10^{-1}, 10^{-1}\}$, and process noise covariance of $\mathbf{Q} = \text{diag}\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}, 10^{-6}\}$, and $\mathbf{\Gamma} = \text{diag}\{0.8, 0.8, 0.8, 0.8, 1\}$. Note that with this choice of $\mathbf{\Gamma}$, the measurement update stage for the parameter λ_m is not performed, and the filter tries to mimic an SKF filter known to tackle the uncertainties due to a nuisance parameter.

In the above cases, with the further tuning of the EKF, some initial conditions and covariance matrices could be found that results in an acceptable estimation. Nevertheless, the purpose was to show how partial updates in the Kalman filter can be helpful if tuning of EKF is not improving final results.

4.6.2 Hardware Validation of PMSM Estimations

A PMSM, with the given specifications and controlled using field-oriented control (FOC) deployed on a Speedgoat real-time target machine [98], is considered in Fig. 23. Herein, we run the PKF under two scenarios: 1) sensorless speed estimation (assuming speed measurements are unavailable); 2) Online PMSM parmeterization. The machine states, outputs, and inputs for the first case are chosen as $\mathbf{x} = [i_{s\alpha}, i_{s\beta}, \omega_m, \theta_m]^\top$, $\mathbf{y} = [i_{s\alpha}, i_{s\beta}]^\top$, and $\mathbf{u} = [v_{s\alpha}, v_{s\beta}]^\top$, respectively. Note that the machine equations are assumed in the α/β stationary reference frame. Furthermore, we assume that the mechanical load is unknown (similar to [99]), hence, the machine model is augmented with $\mathbf{p} = T_L$, with $\frac{dT_L}{dt} \approx 0$. PMSM measurements are sampled every 10 μs. The noise covariance matrices are $\mathbf{Q} = \text{diag}\{10^{-2}, 10^{-2}, 10^{-1}, \pi/24, 0.045\}$ and $\mathbf{R} = \text{diag}\{10^{-4}, 10^{-4}\}$ with
$\mathbf{\Gamma} = \text{diag}\{0.1, 0.1, 0.1, 0.1, 0.1\}$. Fig. 24a shows the estimated results from PKF operating under measurements with 20% loss in data. This loss is modeled using the Bernoulli Binary Generator block in Simulink. The rotor speed is step-changed at times 1.6 s, 2.7 s, and 4.3 s, approximately. From Fig. 24a, it is clear that the PKF continues to provide accurate speed estimations during multiple changes in speed along with other machine states.

Since machines' conditions can vary, an online parameter estimator could be incorporated within the drive. Herein, we consider a case similar to [81] for an online estimation of machine state and permanent magnet flux linkage. Note that the PKF is designed assuming available speed measurements. In this case, the PMSM model is considered in the q/d-reference frame. The augmented state can be formed as $\mathbf{x} \triangleq [i_{ds}, i_{qs}, \omega_m, \theta_m, \lambda_m]^\top$, output as $\mathbf{y} \triangleq [i_{ds}, i_{qs}, \omega_m]^\top$, and input as $\mathbf{u} \triangleq [v_{ds}, v_{qs}]^\top$. PMSM measurements are sampled every 10 μs with $\mathbf{Q} = \text{diag}\{10^{-1}, 10^{-1}, 5, 10^{-1}, 10^{-6}\}$,

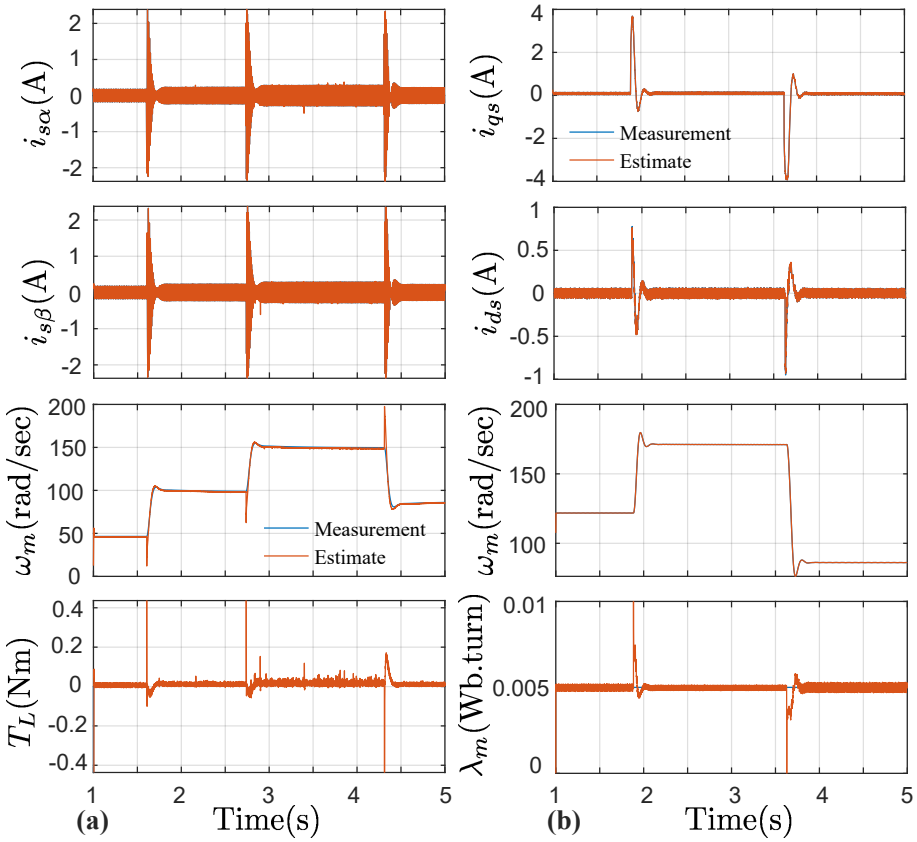


Figure 24: (a) Sensorless speed and load torque estimation. (b) Online parameter estimation. Estimations are performed with 20% loss in measurement data. PKF continues to provide accurate estimations during step changes in speed.

$\mathbf{R} = \text{diag}\{10^{-3}, 10^{-2}, 10^{-1}\}$, $\mathbf{\Gamma} = \text{diag}\{0, 0, 0.1, 0.1, 0.75\}$, and 20% data loss. Figure 24b shows successful estimation results for machine states and parameter λ_m.

4.7 Summary

We have extended the PKF to estimate the states and parameters of PMSMs. We have proposed a variant of PKF to incorporate measurement loss in the incoming data. Through case studies, we have highlighted the advantages of PKF over EKF, while handling system instabilities and nuisance parameters, and validated PKF in estimating parameters and states using experimental PMSM studies.

HARDWARE-ASSISTED SIMULATION OF VOLTAGE-BEHIND-REACTANCE MODELS OF ELECTRIC MACHINES ON FPGA[1]

Authors: A. P. Yadav, S. Xu, B.C. Schafer, and A. Davoudi.

Reprinted, with permission from all the co-authors.

Journal: IEEE Transactions on Energy Conversion [100].

DOI: 10.1109/TEC.2020.2976618.

CHAPTER 5

HARDWARE-ASSISTED SIMULATION OF VOLTAGE-BEHIND-REACTANCE MODELS OF ELECTRIC MACHINES ON FPGA

Abstract

This paper studies the acceleration of numerical simulations executed on Field-Programmable Gate Arrays (FPGAs) for electric machines presented by voltage-behind-reactance (VBR) models. In VBR models, the stator dynamics are modeled in *abc* coordinates, while the rotor dynamics are formulated in *qd* reference frame. Both induction motors and synchronous generators, operating without and with magnetic saturation, are considered. Once VBR models of these machine types are reviewed, their dynamic models are discretized using Runge-Kutta numerical routines. The detailed mapping of such discrete models to FPGA is provided using High-Level Synthesis, which directly converts untimed descriptions into VHDL or Verilog. An automated method finds the fastest FPGA architecture by finding the best set of synthesis options. Experimental results show that our FPGA-based acceleration flow leads to about 92-168 times average simulation speed-up for various machine types compared to the MATLAB simulation.

5.1 Introduction

Lumped-parameter models of electric machines could be broadly classified into *abc* phase-domain models (PD), *qd* reference-frame models (QD), and the voltage-behind-reactance (VBR) models. In the latter form, the stator dynamics are modeled in *abc* coordinates, while the rotor dynamics are formulated in *qd* reference frame [101]. This facilitates an easy interface with the external electrical network, such as electric drives or the power grid, easy integration in nodal-variable-based simulation packages, and partitioned stator and rotor dynamics resulting in an improved eigenstructure with better numerical stability. Early VBR-type modeling of electric machines can be traced back to 1927 [102], with various formulations becoming available since (e.g., for unsaturated [103] and saturated [104] synchronous machines models). Discretization of state equations provides equivalent difference equations for electromagnetic transient simulation studies of unsaturated [105, 101, 106] or saturated machine models [107, 108]. While PD, QD, and VBR models are essentially interchangeable in the continuous-time domain, when discretized using a numerical integration routine, the VBR model exhibits better numerical accuracy and simulation efficiency versus PD models [109]. However, the presence of dynamic saliency results in the need to update the state equations (during transient simulation) making VBR models computationally expensive. This has led to the development of constant-parameter VBR models [110].

Numerical operation of integration algorithms can be discretized into simple arithmetic units (e.g., addition and multiplication), and implemented on a reconfigurable digital logic platform like a Field-Programmable Gate Array (FPGA). This approach exploits the inherent massive parallelism of many applications which, in turn, translates into significant speed-ups compared to the sequential execution of the same application on an off-the-shelf processor. Modern FPGAs have embedded

hard macros such as digital signal processor (DSP)-blocks and embedded memory, or even embedded ARM processors like in Xilinx's Zynq FPGA and Intel's Cyclone V system-on-a-chip (SoC) FPGA. Electric machine models can be mapped onto these devices with a limited need for off-chip resources, making them suitable candidates for hardware-assisted simulation acceleration. Moreover, the flexible connectivity interface to FPGAs, and their very low I/O latency, allow them to be reconfigured with a variety of controller hardware, and make them a suitable choice for hardware- or software-in-the-loop studies [111, 112, 113]. One main challenge with FPGAs is that they are typically programmed using a low-level hardware description language (HDL) such as Verilog or VHDL. These languages require the programmer to have expert knowledge of the hardware in order to efficiently map the target application on the FPGA. This impediment makes them less attractive compared to existing high-level software solutions, more time consuming, and prone to unintended mistakes.

To enhance the FPGA user base, their vendors have resorted to programming them with High-Level Synthesis (HLS), that accepts an untimed behavioral description (such as C or C++) as an input, and automatically generates efficient Verilog or VHDL codes. Off-the-shelf solutions such as MATLAB's HDL Coder [114] can automatically generate a synthesizable VHDL/Verilog code directly from MATLAB and map it onto an FPGA. Unfortunately, this approach only works with a handful of dedicated FPGAs, as the I/O interfaces are fixed, and not with modern programmable SoC FPGAs with embedded ARM processors. Furthermore, MATLAB's HDL coder synthesizes the code into a specific architectural template. In contrast, HLS tools, that take as an input C or C++, allow to automatically generate a set of optimal design variants with unique area versus performance trade-offs, a.k.a. Design Space Exploration (DSE). This enables an optimal FPGA design with the least model latency.

An overview of FPGA-based simulation of electric machines can be found in [1]. Comprehensive treatment on implementing models of DC machine and QD models of AC machines on an FPGA is provided in [115]. [116] has presented FPGA-based simulation of nonlinear MEC model of a faulted electric machine. [6] has designed a model decomposition technique to perform real-time simulation of large-scale energy systems, such as microgrids, on multiple FPGA devices. [117] proposed a robust numerical interface for power hardware-in-the-loop applications using FPGAs. [118] implements the QD model of induction machine on FPGA with 54 times gain in simulation speed when compared to an offline simulation in MATLAB. Following the seminal work in [103], there has been consistent progress in VBR modeling techniques. Recently, [119, 120] have developed VBR models for multi-phase electric machine systems often used in AC-DC energy conversion systems. However, adaptation of VBR models for FPGA implementation is very limited e.g., [121] has designed an FPGA-based grid simulator, that includes an approximate constant-parameter VBR model of an unsaturated machine. The salient contributions of this paper are given below:

- Provide an overview of exact VBR models of induction motors and synchronous generators operating in unsaturated or saturated regions, and discretize resulting models using a fourth-order Runge-Kutta approach.
- Use HLS to quickly map these models onto FPGAs with little requirement for hardware background of target FPGAs.
- Present an automatic search method based on the gradient descent and genetic algorithm to find the fastest FPGA implementation. A detailed comparison is provided between the default HLS FPGA implementation and the fastest HLS FPGA implementation of the VBR models.

- Present extensive studies highlighting the speedup of FPGA-based versus software-based simulations.

5.2 VBR Models of Electric Machines

This section provides a brief overview of dynamic VBR models for synchronous machines [101, 103] and induction machines [106], and their treatment when dealing with magnetic saturation (synchronous [108], induction [107]).

5.2.1 Synchronous Machines

We consider a synchronous machine with two damper windings along the q-axis, *kq1* and *kq2*, and one field winding, *fd*, and one damper winding, *kd*, along the d-axis. Damper windings are shorted, and v_{fd} is applied to the field winding. Motor convention is assumed with currents treated as positive when flowing into the machine. Stator currents and rotor flux linkages are adopted as the state variables[101], leading to

$$\begin{cases} \frac{d}{dt}[\mathbf{L}''_{abcs}(\theta_r)\mathbf{i}_{abcs}] = \mathbf{v}_{abcs} - \mathbf{R}_s\mathbf{i}_{abcs} - \mathbf{e}''_{abcs} \\ \frac{d}{dt}\lambda_j = -\frac{r_j}{L_{lj}}(\lambda_j - \lambda_{mq}); \quad j = kq1, kq2. \\ \frac{d}{dt}\lambda_j = -\frac{r_j}{L_{lj}}(\lambda_j - \lambda_{md}) + v_j; \quad j = fd, kd. \end{cases} \tag{5.1}$$

$\mathbf{i}_{abcs}$ and $\mathbf{v}_{abcs}$ are the stator currents and voltages, respectively. $\mathbf{R}_s$ is a diagonal matrix with the stator resistance value, r_s. λ_j is the flux linkage of the rotor winding (damper or field winding). r_j and L_{lj} are the resistor and leakage inductance of the corresponding rotor winding, respectively. λ_{mq} and λ_{md} are the magnetizing flux linkages in corresponding axes,

$$\begin{cases} \lambda_{mq} = L''_{mq}\left(\frac{\lambda_{kq1}}{L_{lkq1}} + \frac{\lambda_{kq2}}{L_{lkq2}} + i_{qs}\right) \\ \lambda_{md} = L''_{md}\left(\frac{\lambda_{kd}}{L_{lkd}} + \frac{\lambda_{fd}}{L_{lfd}} + i_{ds}\right) \end{cases} . \tag{5.2}$$

The sub-transient inductance matrix, $\mathbf{L}''_{abcs}$, is defined as

$$\mathbf{L}''_{abcs}(\theta_r) = \begin{bmatrix} L_S(2\theta_r) & L_M(2\theta_r - \frac{2\pi}{3}) & L_M(2\theta_r + \frac{2\pi}{3}) \\ L_M(2\theta_r - \frac{2\pi}{3}) & L_S(2\theta_r - \frac{4\pi}{3}) & L_M(2\theta_r) \\ L_M(2\theta_r + \frac{2\pi}{3}) & L_M(2\theta_r) & L_S(2\theta_r + \frac{4\pi}{3}) \end{bmatrix} . \tag{5.3}$$

The sub-transient voltage is $\mathbf{e}''_{abcs} = [\mathbf{K}^r_s]^{-1}[e''_q \;\; e''_d \;\; 0]^T$, where $\mathbf{K}^r_s$ is the transformation matrix from stator to rotor reference frames. e''_q, e''_d, L''_{md}, L''_{mq}, L_S and L_M are detailed in [101].

5.2.2 Induction Machines

There are several VBR formulations of an induction machine with different numerical properties[106]. We adopt the VBR-III formulation, that offers diagonal resistance and inductance matrices for a three-phase, wye-connected machine. A squirrel-cage machine with shorted rotor windings is considered. The state equations for the electric subsystem are

$$\begin{cases} \mathbf{L}_D \frac{d}{dt}\mathbf{i}_{abcs} = \mathbf{v}_{abcs} - \mathbf{R}_D \mathbf{i}_{abcs} - \mathbf{e}''_{abcs} \\ \frac{d}{dt}\lambda_{qr} = -\frac{r_r}{L_{lr}}(\lambda_{qr} - \lambda_{mq}) \\ \frac{d}{dt}\lambda_{dr} = -\frac{r_r}{L_{lr}}(\lambda_{dr} - \lambda_{md}) \end{cases} . \tag{5.4}$$

λ_{qr} and λ_{dr} are the q/d-axis rotor flux linkages. The magnetizing flux linkage terms in corresponding axes are

$$\begin{cases} \lambda_{mq} = L_m^{''}\left(i_{qs} + \frac{\lambda_{qr}}{L_{lr}}\right) \\ \lambda_{md} = L_m^{''}\left(i_{ds} + \frac{\lambda_{dr}}{L_{lr}}\right) \end{cases}, \tag{5.5}$$

where $L_m^{''} = (1/L_m + 1/L_{lr})^{-1}$. L_m and L_{lr} are the mutual and the rotor leakage inductances, respectively. The diagonal resistance, $\mathbf{R}_D$, and inductance matrices, $\mathbf{L}_D$, have $r_s + (L_m^{''2}/L_{lr}^2)r_r$ and $L_{ls} + L_m^{''}$ as respective elements. r_s and r_r are the respective resistances of stator and rotor windings. The sub-transient voltage is $\mathbf{e}_{abcs}^{''} = [\mathbf{K}_s^r]^{-1}\begin{bmatrix} e_q^{''} & e_d^{''} & 0 \end{bmatrix}^T$, with $e_q^{''}$ and $e_d^{''}$ detailed in [106].

5.2.3 Treatment of Magnetic Saturation

The two key issues when incorporating magnetic saturation are (i) dynamic saliency due to rotor structure, and (ii) formulation of saturation characteristic. Assuming a constant saliency factor, S_F, an anisotropic salient-pole machine can be represented as an isotropic one with the following main flux linkage, λ_m, and magnetizing current, i_m, [108]

$$\begin{cases} \lambda_m = \sqrt{\lambda_{md}^2 + (\lambda_{mq}/S_F)^2} \\ i_m = \sqrt{i_{md}^2 + (S_F i_{mq})^2} \end{cases}. \tag{5.6}$$

The relationship between λ_m and i_m is modeled using an arctangent function [122], and approximated as

$$\lambda_m = L_D i_m + \lambda_{res}, \tag{5.7}$$

where $L_D = {}^{\partial\lambda_m}/_{\partial i_m}$ and λ_{res} are dynamic inductance and residual flux, respectively [108]. This linear expression for λ_m can be projected to qd-axes as λ_{mq}, λ_{md}, λ_{resq}, and λ_{resd}.

5.2.3.1 Incorporating saturation in synchronous machine models

The state equations for a saturated synchronous machine have the same structure as those of the unsaturated machine in (5.1), but the expressions for inductance terms, magnetizing flux linkages, and sub-transient variables change due to the effect of magnetic saturation [108] to include the saliency factor, S_F, dynamic inductance, L_D, or residual flux, λ_{resq}.

5.2.3.2 Incorporating saturation in induction machine models

The state-space model for a saturated induction machine is given in [107]

$$\begin{cases} \mathbf{L}''_{abcsat}\frac{d}{dt}\mathbf{i}_{abcs} = \mathbf{v}_{abcs} - \mathbf{R}_s\mathbf{i}_{abcs} - \mathbf{e}''_{abcsat} \\ \frac{d}{dt}\lambda_{qr} = -\frac{r_r}{L_{lr}}(\lambda_{qr} - \lambda_{mq}) \\ \frac{d}{dt}\lambda_{dr} = -\frac{r_r}{L_{lr}}(\lambda_{dr} - \lambda_{md}) \end{cases}, \tag{5.8}$$

where

$$\begin{cases} \lambda_{mq} = L''_D\left(i_{qs} + \frac{\lambda_{qr}}{L_{lr}}\right) + \frac{L_D{}''}{L_D}\lambda_{resq} \\ \lambda_{md} = L''_D\left(i_{ds} + \frac{\lambda_{dr}}{L_{lr}}\right) + \frac{L_D{}''}{L_D}\lambda_{resd} \end{cases}, \tag{5.9}$$

and $L''_D = ({}^1/_{L_D} + {}^1/_{L_{lr}})^{-1}$. $\mathbf{R}_s$ is a diagonal matrix with stator resistances, r_s, as elements. $\mathbf{e}''_{abcsat}$ is the sub-transient voltage term. The sub-transient inductance, $\mathbf{L}''_{abcsat}$, is

$$\mathbf{L}''_{abcsat} = \begin{bmatrix} L_{ls} + \frac{2}{3}L''_D & -\frac{L''_D}{3} & -\frac{L''_D}{3} \\ -\frac{L''_D}{3} & L_{ls} + \frac{2}{3}L''_D & -\frac{L''_D}{3} \\ -\frac{L''_D}{3} & -\frac{L''_D}{3} & L_{ls} + \frac{2}{3}L''_D \end{bmatrix}. \tag{5.10}$$

5.2.4 Mechanical Subsystem

In all the above VBR models, the mechanical subsystem is

$$\begin{cases} \frac{d}{dt}\theta_r = \omega_r \\ \frac{d}{dt}\omega_r = \frac{p}{2J}(T_e - T_m) \\ T_e = \frac{3p}{4}(\lambda_{md} i_{qs} - \lambda_{mq} i_{ds}) \end{cases}. \tag{5.11}$$

θ_r is its rotor position and ω_r is the angular speed. T_e and T_m are the electromagnetic and mechanical torques, respectively. p is the number of poles and J is the inertia. i_{qs} and i_{ds} are the q- and d-axis stator currents transformed from $\mathbf{i}_{abcs}$.

5.3 Discretization and Numerical Integration

Dynamic VBR models discussed in the previous section can be simulated using various numerical routines. Consider a dynamic system,

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(t, \mathbf{x}); \quad \mathbf{x}(t_0) = \mathbf{x}_0. \tag{5.12}$$

$\mathbf{x}$, t_0, and $\mathbf{x}_0$ represent the state, initial time, and initial state, respectively. For the induction machine in (5.4), this becomes

$$\mathbf{x} = \begin{bmatrix} \mathbf{i}_{abcs} & \lambda_{qr} & \lambda_{dr} & \omega_r & \theta_r \end{bmatrix}^T, \tag{5.13}$$

$$\mathbf{f} = \begin{bmatrix} \mathbf{L}_D^{-1}(\mathbf{v}_{abcs} - \mathbf{R}_D\mathbf{i}_{abcs} - \mathbf{e}''_{abcs}) \\ -\frac{r_r}{L_{lr}}(\lambda_{qr} - \lambda_{mq}) \\ -\frac{r_r}{L_{lr}}(\lambda_{dr} - \lambda_{md}) \\ \frac{P}{2J}(\frac{3P}{4}(\lambda_{md}i_{qs} - \lambda_{mq}i_{ds}) - T_m) \\ \omega_r \end{bmatrix}. \tag{5.14}$$

The 4^{th}-order Runge-Kutta (RK4) numerical integration routine [123], with a step size of h, can be formulated for (5.12) as

$$\begin{cases} \mathbf{x}_n = \mathbf{x}_{n-1} + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4) \\ t_n = t_{n-1} + h \end{cases}, \tag{5.15}$$

where $\mathbf{x}_n$ and t_n are state and time at the n^{th} instant. The definitions of coefficients $\mathbf{k}_1 - \mathbf{k}_4$ are standard

$$\begin{cases} \mathbf{k}_1 = \mathbf{f}(t_{n-1}, \mathbf{x}_{n-1}) \\ \mathbf{k}_2 = \mathbf{f}(t_{n-1} + 0.5h, \mathbf{x}_{n-1} + 0.5h\mathbf{k}_1) \\ \mathbf{k}_3 = \mathbf{f}(t_{n-1} + 0.5h, \mathbf{x}_{n-1} + 0.5h\mathbf{k}_2) \\ \mathbf{k}_4 = \mathbf{f}(t_{n-1} + h, \mathbf{x}_{n-1} + h\mathbf{k}_3) \end{cases}. \tag{5.16}$$

This 4^{th}-order approximation provides an acceptable trade-off between accuracy and computational cost. Other integration routines, such as Gears methods, can be similarly adopted [124]. Appendix 6.1 discusses the choice of step size that ensures the numerical stability of RK4.

5.4 Field Programmable Gate Arrays

FPGAs are arrays of logic blocks, with different numbers of lookup tables (LUTs) and flip-flops, connected by programmable wires (switch boxes) [125]. Such architectural flexibility allows FPGA to implement any logic function. The maximum achievable clock frequency depends on the depth of the logic circuit (logic delay) and the wiring needed to connect the logic blocks (wire delay). FPGAs can have up to 35 billion transistors [126] which allows designers to deploy large-scale microelectronic systems. FPGAs are increasingly used as low-cost hardware accelerators to rival parallel computers handling heavy computations [127]. FPGA-assisted simulation accelerations has found application in particle assembly [128], image processing [129], and financial portfolio simulation [130].

Figure 25 shows a block diagram of a traditional FPGA using Xilinx's notation for logic blocks, called Configurable Logic Blocks (CLBs). These logic blocks are alternatively referred to, by Intel®, as Adaptive Logic Modules (ALMs). Modern FPGAs have columns of embedded memory and Digital Signal Processing (DSP) blocks. Figure 25 shows a traditional flow to configurate an FPGA [127]. Traditionally, the user writes Verilog or VHDL codes to describe the circuit to be mapped. The logic synthesis tool then converts this code into a gate netlist which is, in turn, passed to the mapper and place and router. Finally, a bitstream that configures the internal resources of the FPGA is generated. This flow has recently been extended to synthesize untimed software descriptions through High-Level Synthesis.

5.5 High-Level Synthesis

Expert knowledge on hardware needed to program FPGAs is the main impediment to their large-scale adaption. The US Bureau of Labor Statistics reported in

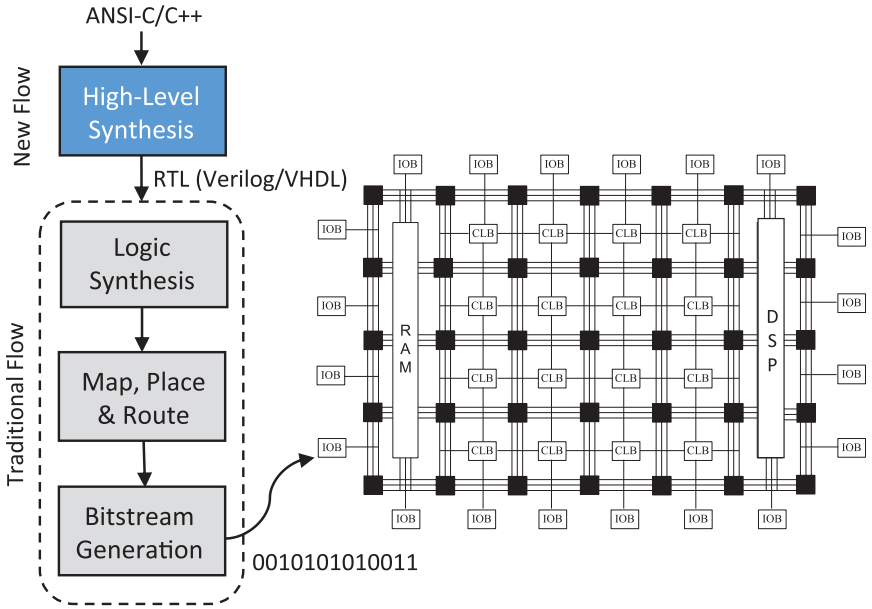


Figure 25: Overview of FPGA's configuration flow and internal structure.

2017 that the ratio of software engineers to hardware engineers almost doubled since 2010 [131]. Since models to be accelerated on FPGAs are almost always developed initially using a high-level programming language like ANSI-C, C++, or MATLAB, it makes sense to have a direct flow between these high-level programming languages and the hardware implementation without resorting to low-level hardware description languages. FPGA vendors have released tools that enable this path through HLS – a synthesis process that accepts an untimed behavioral description such as ANSI-C, and produces hardware description in Verilog or VHDL [132]. This approach widens the user base while significantly reducing the time to map an algorithmic description to hardware. Another distinct advantage of HLS is that once the algorithmic description is fixed, HLS can automatically generate design variants with different resource utilization (e.g., area) versus performance trade-offs. This is possible because certain operations, referred to as *explorable* operations, such as loops, functions, and arrays can be synthesized on an FPGA in different ways leading to different areas and per-

formance. For example, arrays can be mapped to registers or memory, and loops can be unrolled or pipelined[133]. These options can be selected using synthesis directives as pragmas that are added to the behavioral description as comments[132].

Figure 26 shows a sample code along with pragmas. The code snippet shows a behavioral description that computes the moving average of eight numbers [134]. The synthesis of the array that holds the eight values can be controlled through *pragma1*. This allows the array to be synthesized as RAM, registers, or fully expanded. The addition loop can be fully or partially unrolled, not unrolled, or pipelined (*pragma2*). Different combinations of these pragma settings lead to multiple implementations of this behavioral description as shown by the red curve in Figure 2. The main challenge, for a non-expert, is to determine the right combination of pragmas to generate the desired circuit. This is a non-trivial task, especially for complex descriptions, as the synthesis option combinations increase exponentially with the number of explorable operations [135]. To mitigate that, we investigate an automatic HLS design space explorer to find the synthesis options that lead to the fastest implementation of VBR models of electric machines on an FPGA.

HLS design space exploration can be cast as a multi-objective optimization problem that aims to minimize conflicting design parameters. Typically, these parameters are the design area (A), which in our case is the FPGA's resource utilized, and the performance parameter such as latency (L) defined as the number of clock cycles required to generate a new output [135]. The ultimate goal of HLS DSE is to find a set of Pareto-optimal designs [132]. These designs form the Pareto front ($\bar{PF}$), where $\bar{PF}$ is composed of the dominant designs $\bar{P}F = \{d_1, d_2, \ldots, d_n\}$ such that, for any design, $d_i \in \bar{PF}$,

$$A(d_i) \leq A(d_q) \quad \text{and} \quad L(d_i) \leq L(d_q), \tag{5.17}$$

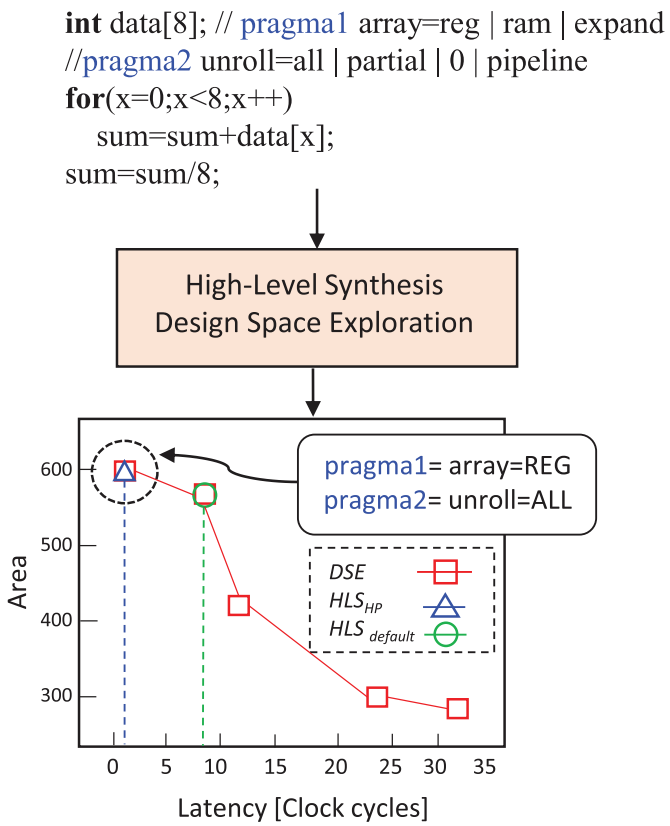


Figure 26: Design space exploration to find the arithmetic average of 8 numbers with pragmas that lead to the fastest implementation (HLS_{HP}).

where $d_q \notin \bar{PF}$. In other words, a design is Pareto-optimal if there is no other candidate in the search space that exhibits less latency or smaller area than d_i. Since the search space grows superlinearly with the number of explorable synthesis options, it is often not possible to claim Pareto-optimality for larger designs[132]. The designs obtained are often called non-dominated or dominating designs. Therefore, heuristics are needed for HLS DSE. An overview of various heuristics is provided in [132].

In this work, we are not interested in finding the Pareto frontier, rather the fastest implementation that can accelerate the simulation of a given VBR model. When synthesizing the behavioral description with no pragmas, the HLS tool will use a default set of options. For example, for the sample code in Figure 26, the default

ALGORITHM 1: Complete MATLAB-to-FPGA hardware acceleration flow

input : $MATLAB_{input}, f_{target}, Techlib$
$MATLAB_{input}$: Behavioral description written in MATLAB
f_{target}: Target synthesis frequency
$TechLib$: Technology library
output: $Design_{.bit/.sof}$
$Design_{.bit/.sof}$: configuration bitstream for FPGA

1 /* **Phase I:** Manual Code Refinement */
2 $C_{synth} \leftarrow MATLAB\_to\_C(MATLAB_{input})$;
3 $C_{taps} \leftarrow C\_insert\_taps(C_{synth})$;
4 $sim\_report \leftarrow sim\_C\_code(C_{taps})$;
5 $C_{opt\_hw} \leftarrow data\_type\_refinement(C_{synth}, sim\_report)$;
6 /* **Phase II:** Fastest Architecture Search*/
7 $HLS_{HP} \leftarrow HLS\_DSE(C_{opt\_hw}, f_{target}, Techlib)$;
8 $RTL_{HP} \leftarrow extract\_fastest\_design(HLS_{HP})$;
9 $Design_{.bit/.sof} \leftarrow logic\_synth(RTL_{HP})$;
10 return($Design_{.bit/.sof}$);

HLS setting leads to the design $HLS_{default} = \{A = 580, L = 8\}$ (green circle in Figure 26), with an area of 580 LUTs and the latency of 8 clock cycles. The goal in this work is to find the set of pragmas that lead to the fastest possible implementation, HLS_{HP}. This is achieved if pragma1 is set to register and pragma 2 is set to fully unroll the loop. In this case, $HLS_{HP} = \{A = 610, L = 1\}$, which implies that the area has increased from 580 to 610, while the latency has decreased from 8 to 1, i.e., the new circuit has become 8 times faster.

5.6 Model Deployment on FPGAs

In this section, we detail how the dynamic VBR models are accelerated on an FPGA using HLS. Once the dynamic VBR models in Section 5.2 are discretized using the Runge-Kutta routine in Section 5.3, they can be deployed on FPGA boards for an accelerated execution of numerical simulation.

Figure 27 and Algorithm 1 highlight the two distinct phases of the hardware implementation flow. The first phase manually re-implements the MATLAB code into

ANSI-C, as available HLS tools [136] mainly supports ANSI-C and C++. MATLAB also offers a direct MATLAB-to-HDL synthesis tool called HDL coder [137]. However, one shortcoming of HDL coder is that it has a limited number of synthesis options that do not allow tailoring the implementation to different designers' preferences (e.g., low power or high performance). The second phase automatically generates the FPGA configuration file by, first, finding the fastest hardware implementation using HLS design-space exploration and, then, performing logic synthesis, place and route, and finally, generating the bitstream to configure the FPGA (.bit or .sof depending on the FPGA vendor). These two phases of hardware implementation flow are detailed in what follows.

Phase I: Manual Code Refinement: First, the original MATLAB code is manually translated into a synthesizable C code (ANSI-C) (line 2 in Algorithm 1). This step is shown in Algorithm 1 to be achieved using the function $MATLAB\_to\_C$ for a concise representation. The result is an ANSI-C description that is synthesizable, but has not been fully optimized (C_{synth}). Problems that can benefit from FPGA-based hardware acceleration are commonly those with fixed-point data type that can be performed in parallel [138]. As the floating-point data types used in the MATLAB environment could require too many logic resources, they are refined using fixed-point data types supported by the HLS tool. To help further optimize the resulting hardware, different trigonometric functions used in VBR models of Section 5.2 (e.g., to treat saturation) are also refined. HLS tool vendors typically provide a library with synthesizable mathematical functions. The precision of these trigonometric functions is automatically set based on an iterative process that guarantees matching their outputs with reference outputs obtained from the original MATLAB code. This ensures that no numerical stability issue appears when running the accelerated algorithms on

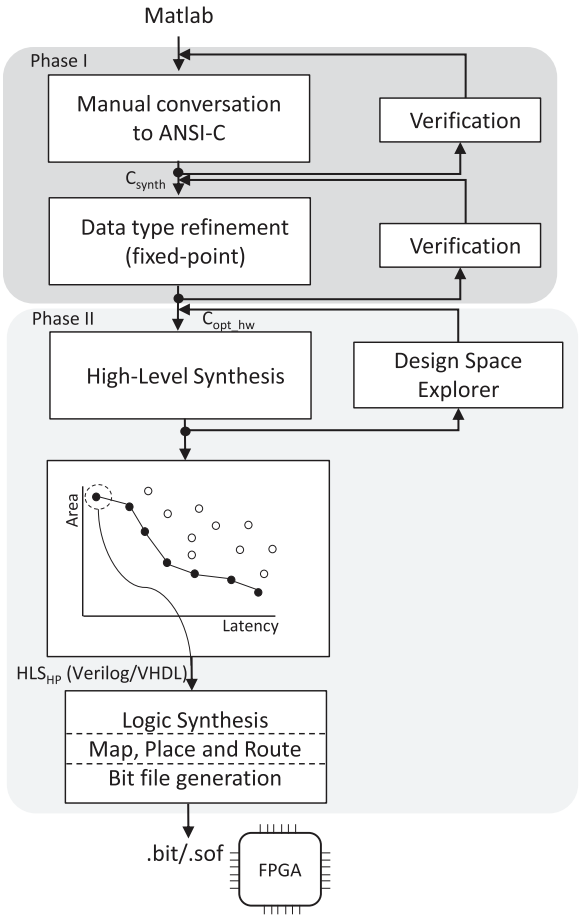


Figure 27: MATLAB-to-FPGA hardware acceleration flow.

the FPGA. For this purpose, the bitwidth of the signals are reduced to the smallest allowable bitwidth that does not introduce any error, by automatically inserting *taps* into the converted C code through a simple Python script (line 3 in Algorithm 1, using function *C_insert_taps*). These taps record all values of the tapped signals in a simulation report (*sim_report*) when the C descriptions are compiled and executed with the given test data (line 4 in Algorithm 1, using function *sim_C_code*). The maximum range and precision of each signal is computed, and every internal signal is adjusted accordingly by specifying it as a fixed-point data type (line 5 in Algorithm 1, using function *data_type_refinement*). The result is a refined behavioral description optimized for HLS (C_{opt_hw}).

Phase II: Fastest Architecture Search: The resulting behavioral description is, in turn, passed to an automated HLS DSE to find the fastest implementation (micro-architecture) for a given VBR model.

Problem Formulation: Given a behavioral description in ANSI-C (C) to be accelerated on an FPGA, a library of synthesis directives as pragmas P that form a search space $\mathbb{R}_C^P$, and a commercial HLS tool (HLS) that can estimate the performance in terms of latency $L(HLS, C, P)$ and resource utilization $R_i(HLS, C, P)$, find the configuration $c \in \mathbb{R}_C^P$, in a given time limit, such that c fits in the FPGA and the latency is minimized. This problem can be formulated as

$$\begin{aligned} &\min \ \ L \\ &\text{subject to} \ \ R_i \leq R_{FPGA} \ , \end{aligned} \tag{5.18}$$

where L is the latency to be minimized and R_i is the FPGA resource that has to be smaller than available FPGA resources (R_{FPGA}). In other words, the optimal design fits in a given FPGA and minimizes the clock cycles to generate a new output. This step is represented by function *HLS_DSE* in line 7 of Algorithm 1.

Because the combination of synthesis options increases exponentially with the number of explorable operations, an exhaustive enumerations of all the combinations is not possible. Thus, we tailor two meta-heuristics to find the fastest implementations (line 7 in Algorithm 1), namely, gradient descent and genetic algorithm, and compare their results.

Gradient Descent Search: We start from a random set of pragma combinations $c_1 = \{P_1, P_2, \ldots, P_n\}$, and synthesize it using HLS to obtain the Latency, L_1, and the

resource utilization, R_1. This initial configuration c_1 is updated by following the steepest descent (largest gradient). One limitation of this approach is that it requires the objective function to be differentiable to find the configuration that leads to the largest gradient. We leverage the finite difference method to approximate the gradient value by using the HLS tool as a black box. Given a candidate configuration c_j perturbed from the current configuration c_i, we approximate the gradient as

$$Gradient(c_j, c_i) = \frac{L(c_j) - L(c_i)}{R(c_j) - R(c_i)}. \tag{5.19}$$

Given the current configuration, c_i, the next candidate configuration, c_j, (as per the steepest descent) can be obtained by perturbing a fixed percentage of the pragmas in c_i, and computing their latency and resource utilization for (5.19). The configuration with the largest gradient is chosen next. Experimentally, we observed that a perturbation rate of 20% led to satisfactory results. To ensure (near) optimal solution, we have repeated this process for different initial seeds.

Genetic Algorithm Search: Each explorable operation (array, function, or loops) represents a gene to which a synthesis directive, in the form of pragma, is assigned. A set of genes builds a chromosome (an HLS design). The genetic algorithm starts by generating a random population of unique design combinations of chromosomes. This initial population then generates new chromosomes using evolution-based techniques of selection, crossover, and mutation. The crossover and mutation rates for the genetic algorithm are 0.8 and 0.1, respectively. Interested readers can refer to [133] for a detailed treatment on genetic algorithm-based design space exploration.

The result of these explorations is the fastest micro-architecture, HLS_{HP}, that is passed on to the FPGA's back-end tools that synthesizes this Verilog code, places and routes it, and generates the configuration bitstream for the FPGA. This stage is

fully automated (lines 8-9 in Algorithm 1, using functions *extract_fastest_design* and *logic_synth* [139]). RTL_{HP} and $Design_{.bit/.sof}$ represent the Register-transfer-level design and bitstream file, respectively, for HLS_{HP}.

5.7 Comparative Studies

FPGA prototypes of various machine types, for models reviewed in Section 5.2 and discretized using numerical routines in Section 5.3, are developed using the design protocols offered in Section 5.6. The following studies are inspired by and the machine parameters are adopted from [4]. These machine parameters are provided in Appendix 6.2. The machine models are simulated in the MATLAB environment via scripts (.m file format) using the fourth-order Runge-Kutta routine with a fixed step size of 0.5 ms. The Simulink environment, and the inbuilt ordinary differential equation (*ode*) solver routines in the MATLAB environment, are avoided (due to their overhead) to ensure that both MATLAB and FPGA environments numerically execute the same set of equations, which allows for a fair comparison. Simulation runs on FPGAs are compared against their original MATLAB simulation runs to deduce the comparative gain in the simulation speed. This gain is defined as the average acceleration observed over several runs. Simulations are executed on a 1.6 GHz machine, with an Intel(R) Xeon(R) CPU E5-2603 v3 processor and 32 GB of RAM, running Linux Fedora Core 20. The version of MATLAB used is 2018a. The HLS tool, the logic synthesis tool, and the target FPGA board are CyberWorkBench v.6.1 [136], Quartus II v.17.0 [140], and Terasic DE1-SoC board with an Intel Cyclone V FPGA 5CSEMA5F31C6, respectively.

Each model refined in ANSI-C is explored using the gradient descent and genetic algorithm explorers. In all scenarios, both explorers led to the same fastest implementation (HLS_{HP}). We compare the results of the fastest implementation

reported by the explorers versus the default implementation when the ANSI-C code is directly synthesized ($HLS_{default}$). Tables 9 and 10 summarize the experimental results. Table 9 provides the FPGA resource utilization, for each machine model, as well as the maximum clock frequency and circuit latency. Table 10 shows dynamic performances predicted using MATLAB and FPGA. It can be observed that, in all four cases, the transients produced from both MATLAB and FPGA are identical (see Figures 28, 29, 30, and 31).

First, the VBR model of a synchronous machine, connected to an infinite bus, is simulated with a sudden change in the input torque (see Figure 28). The FPGA execution is 69× and 101× faster than those of MATLAB for the $HLS_{default}$ and HLS_{HP}, respectively. Table 9(a) shows the FPGA resources required for these two FPGA implementations. Next, free acceleration (no load) operation of an induction machine under rated line-to-line voltage is considered. The speedups obtained in this case are 136× and 271× for the $HLS_{default}$ and HLS_{HP}, respectively, while an excellent match between the traces resulting from two implementations are reported in Figure 29. Table 9(b) shows the resources and timing information of the FPGA implementation.

The next set of experiments consider VBR models of electric machines operating with magnetic saturation. The VBR model of a low-speed synchronous machine is simulated under a three-phase fault condition. Table 9(c) details the resource utilization for the VBR model deployment on the target FPGA. Comparison of numerical model executions in MATLAB and FPGA, reflected in Figure 30, shows a speedup of 74× and 114× for the $HLS_{default}$ and HLS_{HP}, respectively. Finally, an induction machine model operating under saturated conditions is considered for a single-phase voltage sag scenario. Phase a is set to 50% of the rated voltage for 0.1 s while, to ensure saturation, the input voltages are 10% higher than the rated voltage. Ta-

Table 9: FPGA resource utilization to implement different VBR models

VBR model of a synchronous machine (a)		
Resources	$HLS_{default}$	HLS_{HP}
Number of Registers	12,731	13,887
Number of ALMs	20,658/32,070 (64%)	22,144/32,070 (69%)
BlockRAM bits	202,752/4,065,280 (5%)	202,752/4,065,280 (5%)
RAM Blocks	30/397 (8%)	30/397 (8%)
Number of DSPs	50/87 (57%)	50/87 (57%)
Max frequency (MHz)	49.05	50.2
Latency (clk cycles)	173	155
VBR model of an induction machine (b)		
Number of Registers	6,504	10,252
Number of ALMs	11,216/32,070 (35%)	18,321/32,070 (57%)
BLockRAM bits	168,960/4,065,280 (4%)	105,336/4,065,280 (3%)
RAM Blocks	20/397 (5%)	15/397 (4%)
Number of DSPs	28/87 (32%)	38/87 (44%)
Max frequency (MHz)	48.62	49.2
Latency (clk cycles)	108	75
VBR model of a saturated synchronous machine (c)		
Number of Registers	66,010	75,363
Number of ALMs	28,847/32,070 (90%)	34,488/30,488 (95%)
BlockRAM bits	2,862,592/4,065,280(70%)	2,099,640 (52%)
RAM Blocks	230/397 (58%)	180/397 (45%)
Number of DSPs	87/87 (100%)	87/87 (100%)
Max frequency (MHz)	84.75	82.11
Latency (clk cycles)	276	201
VBR model of a saturated induction machine (d)		
Number of Registers	39,021	55,387
Number of ALMs	22,280/32,070 (69%)	26,852
BLockRAM bits	1,455,872/4,065,280 (36%)	1,234,499 (30%)
RAM Blocks	110/397 (28%)	100 (25%)
Number of DSPs	85/87 (98%)	85/87 (98%)
Max frequency (MHz)	81.75	83.4
Latency (clk cycles)	227	199

ble 9(d) summarizes the FPGA resources needed to deploy the corresponding VBR model. While an excellent match between MATLAB and FPGA transients are reported in Figure 31, 88× and 186× simulation speedups are achieved for $HLS_{default}$ and HLS_{HP}, respectively.

From Table 9, it can be observed that the HLS_{HP} implementation requires more registers, ALMs (LUTs), and DSP macros than the default implementation ($HLS_{default}$), while needing less BlockRAM. This is because, while minimizing latency, the explorer generates a set of synthesis attributes (pragmas) that prefer reg-

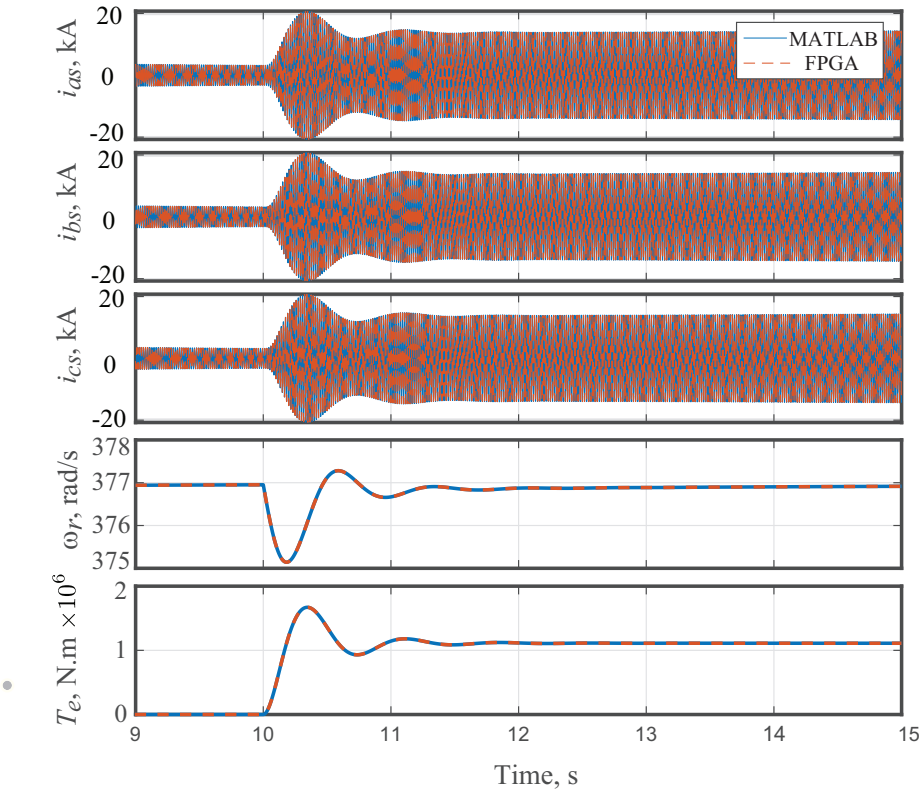


Figure 28: Dynamic performance of the VBR model of a synchronous machine, deployed in MATLAB and FPGA environments, under a sudden change in the input torque from 0 to 1.11×10^6 N.m at $t = 10$ s.

isters instead of memories (as registers are faster and can be accessed in parallel). Moreover, these (near) optimal sets of pragmas exploit parallelism by aggressively unrolling loops and inlining functions. As shown, this leads to hardware designs that require a larger amount of resources, while leading to shorter latencies and, hence, shorter simulation runtime.

Table 10 summarizes the speedups achieved by the two hardware implementations compared to the original MATLAB simulation. On average, the default HLS implementation ($HLS_{default}$) leads to a speedup of 92×, while the implementation found by the explorer (HLS DSE) leads to an average speedup of 168×. This is 1.83× gain over the performance that default HLS would have provided.

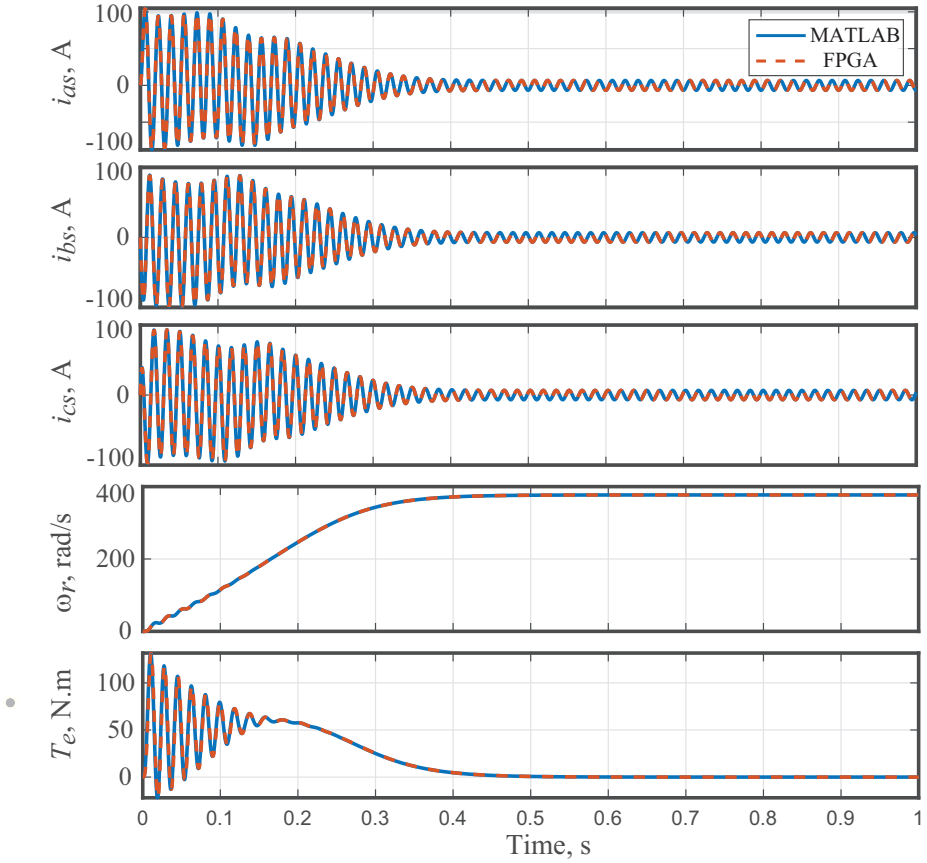


Figure 29: Simulated induction machine dynamics, based on VBR model in both MATLAB and FPGA environments, under free-acceleration conditions.

5.8 Summary

Voltage-behind-reactance models of electric machines are becoming popular due to the ease of network interfacing offered by their phase-domain (abc) representation of the stator subsystem [141, 107, 105, 142]. FPGA deployment of dynamic models of electric machines is very useful to accelerate the simulation and hardware-in-the-loop applications. By eliminating overheads in simulation software environments (e.g., MATLAB), a noticeable acceleration of the numerical simulation is expected. VBR models of synchronous machines and induction motors, operating in the linear or saturated regions, are automatically deployed on an FPGA platform.

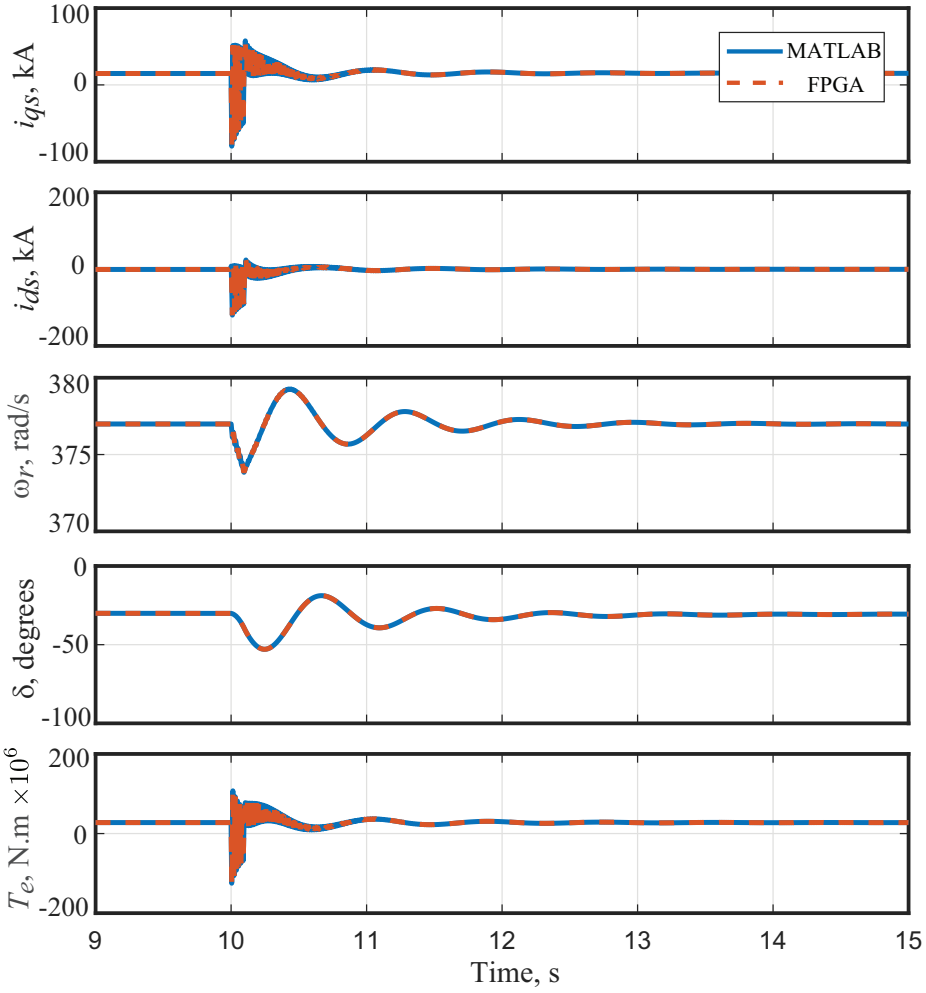


Figure 30: Three-phase fault (at $t = 10$ s) for VBR model of a saturated synchronous machine implemented in both MATLAB and FPGA environments.

Another aspect of this work is the design space explorer that obtains the fastest architecture for mapping machine models on FPGA. This is achieved by formulating the DSE search as a multi-objective optimization problem, and solved using both gradient descent and genetic algorithm methods. The performance of the fastest HLS FPGA implementation (obtained from DSE) is compared against the default HLS FPGA implementation for the considered VBR models. Future directions include achieving multi-resolution simulation on FPGA that would take advantage of the

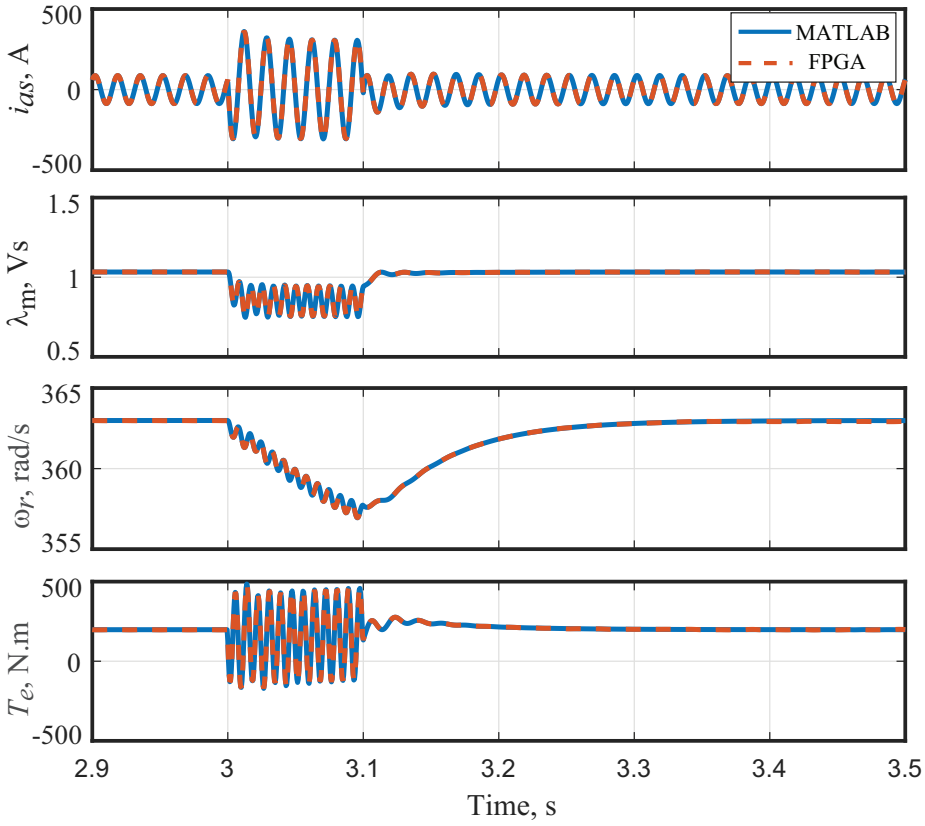


Figure 31: Time-domain transients obtained from both MATLAB and FPGA implementations of the VBR model of a saturated induction machine under phase a voltage sag ($v_{as} = 0.5$ p.u.) during $t = 3$ s to $t = 3.1$ s.

decoupled rotor and stator dynamics to be solved with different solvers and step sizes, and employing approximate constant-parameter VBR models that lead to a time-invariant inductance terms and a faster execution [142, 109, 110].

Table 10: Running time comparison between MATLAB and FPGA implementations of VBR models of different machines types

Models	**Running Time**			**Speedup**	**Speedup**
	MATLAB [s]	$HLS_{default}$ [ms]	HLS_{HP} [ms]	MATLAB vs. $HLS_{default}$	MATLAB vs. HLS_{HP}
Synchronous machine	10.91	157.39	108.45	69	101
Induction machine	1.57	11.52	5.80	136	271
Saturated synchronous machine	7.19	97.69	63.28	74	114
Saturated induction machine	2.44	27.77	13.09	88	186
				Average = 92	**Average = 168**

CHAPTER 6

CONCLUSION

Electric machine models are identified using techniques based on convex optimization and Kalman filter. These data-driven methods extract machine parameters using transient measurements during normal operation, without resorting to laboratory tests. First, a *qd0* model of a synchronous machine is extracted from its MEC model using cone programming. The developed approach, which uses both convex optimization and local search, is further extended for induction machine parameter identification. Herein, robustness to lossy data is also considered. Second, an online partial-update Kalman filter is designed for PMSM estimation problems for lossy incoming measurements. Finally, faster FPGA-based simulation of machine models is achieved by minimizing latency.

Future extension of this work could explore a more general *qd0* models for model extraction. One could also consider scenarios with time-varying parameters. Multi-resolution simulation on FPGA that would take advantage of the decoupled rotor and stator dynamics in VBR models is another interesting research direction. Finally, we hope to bring in tools from machine learning, such as neural networks, for the macromodeling of electric machines.

APPENDIX A

6.1 Appendix A: Discussion on The Choice of Step size

The stability characteristics of a multi-step integration routine can be determined using its stability function [143]. For the classical RK4 method, the stability function, $q(z)$, becomes

$$q(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}, \tag{6.1}$$

where $z = h\sigma$. h is the step size and σ is eigenvalue of the linearized machine model. To ensure numerical stability, $|q(z)| \leq 1$. Step size, $h = 0.5$ ms, is chosen as a good trade-off between accuracy and performance. Interested readers can refer to Figure 9 in [106] and Figure 10 in [101] that report the relative error against the step size for the VBR models of induction machine and synchronous machine, respectively.

6.2 Appendix A: Machine Parameters

Parameters for different machines are adopted from [4]

1. Unsaturated synchronous machine: 26 kV, 835 MVA, $p = 2, r_s = 0.00243\ \Omega, X_{ls} = 0.1538\ \Omega, X_d = 1.457\ \Omega, r'_{kq1} = 0.00144\Omega, X'_{lkq1} = 0.6578\ \Omega, r'_{kq2} = 0.00681\ \Omega, X'_{lkq2} = 0.07602\ \Omega, r'_{fd} = 0.00075\ \Omega, X'_{lfd} = 0.1145\ \Omega, r'_{kd} = 0.01080\ \Omega, X'_{lkd} = 0.06577\ \Omega, X_q = 1.457\ \Omega, J = 0.0658 \times 10^6$ kgm^2.
2. Unsaturated induction machine: 3 hp, 220 V, $p = 4,\ r_s = 0.435\ \Omega,\ X_{ls} = 0.754\ \Omega,\ X_M = 26.13\ \Omega,\ X'_{lr} = 0.754\ \Omega,\ r'_r = 0.816\ \Omega,\ J = 0.089$ kgm^2.

3. Saturated synchronous machine: 20 kV, 325 MVA, $p = 64, r_s = 0.00243\ \Omega, X_{ls} = 0.1478\ \Omega, X_q = 0.5911\ \Omega, r'_{kq2} = 0.01675\ \Omega, X'_{lkq2} = 0.1267\ \Omega, X_d = 1.0467\ \Omega, r'_{fd} = 0.00050\ \Omega, X'_{lfd} = 0.2523\ \Omega, r'_{kd} = 0.01736\ \Omega, X'_{lkd} = 0.1970\ \Omega, J = 35.1 \times 10^6$ kgm^2. *Saturation characteristics parameters:* $\tau_T = 0.3,\ \lambda_T = 43.3962,\ M_d = 331.1652,\ M_a = 730.5857$.
4. Saturated induction machine: 50 hp, 460 V, $p = 4,\ r_s = 0.087\ \Omega,\ X_{ls} = 0.302\ \Omega,\ X_M = 13.08\ \Omega,\ X'_{lr} = 0.302\ \Omega,\ r'_r = 0.228\ \Omega,\ J = 1.662$ kgm^2. *Saturation characteristics parameters:* $\tau_T = 20,\ \lambda_T = 0.82,\ M_d = 62.75,\ M_a = 88.95$.

REFERENCES

[1] S. Mojlish, N. Erdogan, D. Levine, and A. Davoudi, "Review of hardware platforms for real-time simulation of electric machines," *IEEE Transactions on Transportation Electrification*, vol. 3, no. 1, pp. 130–146, March 2017.

[2] F. Gao, X. Zheng, S. Bozhko, C. I. Hill, and G. Asher, "Modal analysis of a pmsg-based dc electrical power system in the more electric aircraft using eigenvalues sensitivity," *IEEE Transactions on Transportation Electrification*, vol. 1, no. 1, pp. 65–76, June 2015.

[3] "Ieee guide for test procedures for synchronous machines part iacceptance and performance testing part ii-test procedures and parameter determination for dynamic analysis," *IEEE Std 115-2009*, 2010.

[4] P. Krause, O. Wasynczuk, S. D. Sudhoff, and S. Pekarek, *Analysis of electric machinery and drive systems*, 3rd ed. Piscataway, NJ, USA: IEEE Press, 2013.

[5] B. Jandaghi and V. Dinavahi, "Prototyping of nonlinear time-stepped finite element simulation for linear induction machines on parallel reconfigurable hardware," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 10, pp. 7711–7720, Oct 2017.

[6] M. Milton, A. Benigni, and A. Monti, "Real-time multi-fpga simulation of energy conversion systems," *IEEE Transactions on Energy Conversion*, pp. 1–1, 2019.

[7] A. P. Yadav, T. Altun, R. Madani, and A. Davoudi, "Macromodeling of electric machines from ab initio models," *IEEE Trans. Energy Convers.*, vol. 35, no. 2, pp. 908–916, Jun. 2020.

[8] N. Roshandel Tavana and V. Dinavahi, "A general framework for fpga-based real-time emulation of electrical machines for hil applications," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2041–2053, April 2015.

[9] M. L. Bash and S. Pekarek, "Analysis and validation of a population-based design of a wound-rotor synchronous machine," *IEEE Transactions on Energy Conversion*, vol. 27, no. 3, pp. 603–614, Sep. 2012.

[10] M. L. Bash, J. M. Williams, and S. D. Pekarek, "Incorporating motion in mesh-based magnetic equivalent circuits," *IEEE Transactions on Energy Conversion*, vol. 25, no. 2, pp. 329–338, June 2010.

[11] M. L. Bash and S. D. Pekarek, "Modeling of salient-pole wound-rotor synchronous machines for population-based design," *IEEE Transactions on Energy Conversion*, vol. 26, no. 2, pp. 381–392, June 2011.

[12] R. Wang, S. Pekarek, M. L. Bash, A. Larson, and R. V. Maaren, "Incorporating dynamics in a mesh-based magnetic equivalent circuit model of synchronous machines," *IEEE Transactions on Energy Conversion*, vol. 30, no. 3, pp. 821–832, Sep. 2015.

[13] H. W. Derbas, J. M. Williams, A. C. Koenig, and S. D. Pekarek, "A comparison of nodal-and mesh-based magnetic equivalent circuit models," *IEEE Transactions on Energy Conversion*, vol. 24, no. 2, pp. 388–396, May 2009.

[14] G. Friedrich and A. Girardin, "Integrated starter generator," *IEEE Industry Applications Magazine*, vol. 15, no. 4, pp. 26–34, July 2009.

[15] M. Karrari and O. P. Malik, "Identification of physical parameters of a synchronous generator from online measurements," *IEEE Transactions on Energy Conversion*, vol. 19, no. 2, pp. 407–415, June 2004.

[16] E. Kyriakides, G. T. Heydt, and V. Vittal, "Online parameter estimation of round rotor synchronous generators including magnetic saturation," *IEEE Transactions on Energy Conversion*, vol. 20, no. 3, pp. 529–537, Sep. 2005.

[17] J. J. Sanchez-Gasca, C. J. Bridenbaugh, C. E. J. Bowler, and J. S. Edmonds, "Trajectory sensitivity based identification of synchronous generator and excitation system parameters," *IEEE Transactions on Power Systems*, vol. 3, no. 4, pp. 1814–1822, Nov 1988.

[18] G. Valverde, E. Kyriakides, G. T. Heydt, and V. Terzija, "Nonlinear estimation of synchronous machine parameters using operating data," *IEEE Transactions on Energy Conversion*, vol. 26, no. 3, pp. 831–839, Sep. 2011.

[19] T. Boileau, N. Leboeuf, B. Nahid-Mobarakeh, and F. Meibody-Tabar, "Online identification of pmsm parameters: Parameter identifiability and estimator comparative study," *IEEE Transactions on Industry Applications*, vol. 47, no. 4, pp. 1944–1957, July 2011.

[20] Ping Zhou, J. Gilmore, Z. Badics, and Z. J. Cendes, "Finite element analysis of induction motors based on computing detailed equivalent circuit parameters," *IEEE Transactions on Magnetics*, vol. 34, no. 5, pp. 3499–3502, Sep. 1998.

[21] A. Repo, P. Rasilo, A. Niemenmaa, and A. Arkkio, "Identification of electromagnetic torque model for induction machines with numerical magnetic field solution," *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 1586–1589, June 2008.

[22] O. Makela, A. Repo, and A. Arkkio, "Parameter estimation for synchronous machines using numerical pulse test within finite element analysis," in *18th International Conference on Electrical Machines*, 2008, pp. 1–5.

[23] S. Huang, Q. Wu, J. Wang, and H. Zhao, "A sufficient condition on convex relaxation of ac optimal power flow in distribution networks," *IEEE Transactions on Power Systems*, vol. 32, no. 2, pp. 1359–1368, March 2017.

[24] J. Li, F. Liu, Z. Wang, S. H. Low, and S. Mei, "Optimal power flow in standalone dc microgrids," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5496–5506, Sep. 2018.

[25] S. Bahrami, F. Therrien, V. W. S. Wong, and J. Jatskevich, "Semidefinite relaxation of optimal power flow for ac–dc grids," *IEEE Transactions on Power Systems*, vol. 32, no. 1, pp. 289–304, Jan 2017.

[26] H. Kojooyan-Jafari, L. Monjo, F. Corcoles, and J. Pedra, "Parameter estimation of wound-rotor induction motors from transient measurements," *IEEE Trans. Energy Conv.*, vol. 29, no. 2, pp. 300–308, Jun. 2014.

[27] S. A. Odhano, P. Pescetto, H. A. A. Awan, M. Hinkkanen, G. Pellegrino, and R. Bojoi, "Parameter identification and self-commissioning in AC motor drives: A technology status review," *IEEE Trans. Power Electron.*, vol. 34, no. 4, pp. 3603–3614, Apr. 2019.

[28] V. H. Quintana, G. L. Torres, and J. Medina-Palomo, "Interior-point methods and their applications to power systems: a classification of publications and software codes," *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 170–176, Feb 2000.

[29] P. Beyhaghi, D. Cavaglieri, and T. Bewley, "Delaunay-based derivative-free optimization via global surrogates, part i: linear constraints," *Journal of Global Optimization*, vol. 66, no. 3, pp. 331–382, 2016.

[30] M. Kojima, S. Mizuno, and A. Yoshise, *A Primal-Dual Interior Point Algorithm for Linear Programming*. New York, NY: Springer New York, 1989, pp. 29–47. [Online]. Available: https://doi.org/10.1007/978-1-4613-9617-8_2

[31] Y. Weng, Q. Li, R. Negi, and M. Ilic, "Semidefinite programming for power system state estimation," in *Proc. IEEE Power & Energy Soc. Gen. Meeting*, San Diego, CA, USA, 2012, pp. 1–8.

[32] C. A. Rojas, J. I. Yuz, M. Aguirre, and J. Rodriguez, "A comparison of discrete-time models for model predictive control of induction motor drives," in *IEEE International Conference on Industrial Technology (ICIT)*, 2015, pp. 568–573.

[33] R. Madani, M. Kheirandishfard, J. Lavaei, and A. Atamturk, "Penalized semidefinite programming for quadratically-constrained quadratic optimization," *arXiv preprint arXiv:2004.14328*, Apr. 2020.

[34] R. H. Tütüncü, K. C. Toh, and M. J. Todd, "Solving semidefinite-quadratic-linear programs using SDPT3," *Mathematical Programming*, vol. 95, no. 2, pp. 189–217, Feb. 2003.

[35] MOSEK-Aps, *The MOSEK optimization toolbox for MATLAB manual. Version 8.1.*, 2017. [Online]. Available: http://docs.mosek.com/8.1/toolbox/index.html

[36] M. Grant and S. Boyd. (2014, Mar.) CVX: Matlab software for disciplined convex programming, version 2.1. [Online]. Available: http://cvxr.com/cvx

[37] R. D. Zimmerman and H. Wang, "Matpower interior point solver mips 1.3 user's manual," 2016.

[38] R. Wang, S. Pekarek, and M. Bash, "Alternative excitation strategies for a wound rotor synchronous machine drive," in *IEEE Energy Conversion Congress and Exposition (ECCE)*, 2012, pp. 2300–2307.

[39] Hua Bai, S. D. Pekarek, J. Tichenor, W. Eversman, D. J. Buening, G. R. Holbrook, M. L. Hull, R. J. Krefta, and S. J. Shields, "Analytical derivation of a coupled-circuit model of a claw-pole alternator with concentrated stator windings," *IEEE Trans. Energy Conv.*, vol. 17, no. 1, pp. 32–38, March 2002.

[40] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, October–December 2006.

[41] A. P. Yadav, R. Madani, N. Amiri, J. Jatskevich, and A. Davoudi, "Induction machine parameterization from limited transient data using convex optimization," *IEEE Transactions on Industrial Electronics*, pp. 1–1, Feb. 2021.

[42] P. Kundur, *Power system stability and control*. New York, USA: McGraw-Hill, 1994.

[43] H. A. Maksoud, S. M. Shaaban, M. S. Zaky, and H. Z. Azazi, "Performance and stability improvement of afo for sensorless im drives in low speeds regenerating mode," *IEEE Trans. Power Electron.*, vol. 34, no. 8, pp. 7812–7825, Aug. 2019.

[44] H. A. Toliyat, E. Levi, and M. Raina, "A review of RFO induction motor parameter estimation techniques," *IEEE Trans. Energy Convers.*, vol. 18, no. 2, pp. 271–283, Jun. 2003.

[45] *IEEE Standard Test Procedure for Polyphase Induction Motors and Generators*, IEEE Std. 112-2017, 2018.

[46] L. Monjo, H. Kojooyan-Jafari, F. Corcoles, and J. Pedra, "Squirrel-cage induction motor parameter estimation using a variable frequency test," *IEEE Trans. Energy Conv.*, vol. 30, no. 2, pp. 550–557, Jun. 2015.

[47] J. Ruan and S. Wang, "A prediction error method-based self-commissioning scheme for parameter identification of induction motors in sensorless drives," *IEEE Trans. Energy Conv.*, vol. 30, no. 1, pp. 384–393, Mar. 2015.

[48] S. R. P. Reddy and U. Loganathan, "Offline recursive identification of electrical parameters of vsi-fed induction motor drives," *IEEE Trans. Power Electron.*, vol. 35, no. 10, pp. 10 711–10 719, Oct. 2020.

[49] S. Lee, A. Yoo, H. Lee, Y. Yoon, and B. Han, "Identification of induction motor parameters at standstill based on integral calculation," *IEEE Trans. Ind. Appl.*, vol. 53, no. 3, pp. 2130–2139, May 2017.

[50] M. Carraro and M. Zigliotto, "Automatic parameter identification of inverter-fed induction motors at standstill," *IEEE Trans. Ind. Electron.*, vol. 61, no. 9, pp. 4605–4613, Sep. 2014.

[51] J. Benzaquen, J. Rengifo, E. Albanez, and J. M. Aller, "Parameter estimation for deep-bar induction machines using instantaneous stator measurements from a direct startup," *IEEE Trans. Energy Convers.*, vol. 32, no. 2, pp. 516–524, Jun. 2017.

[52] S. Chiniforoosh, L. M. Vargas, L. Wang, and J. Jatskevich, "Online characterization procedure for induction machines using start-up and loading transients," in *Proc. IEEE Canada Elect. Power Conf.*, Vancouver, BC, Canada, 2008, pp. 1–5.

[53] P. Huynh, H. Zhu, and D. Aliprantis, "Non-intrusive parameter estimation for single-phase induction motors using transient data," in *Proc. IEEE Power & Energy Conf.*, Champaign, IL, USA, 2015, pp. 1–8.

[54] D. J. Atkinson, P. P. Acarnley, and J. W. Finch, "Observers for induction motor state and parameter estimation," *IEEE Trans. Ind. Appl.*, vol. 27, no. 6, pp. 1119–1127, Nov. 1991.

[55] J. Stephan, M. Bodson, and J. Chiasson, "Real-time estimation of the parameters and fluxes of induction motors," *IEEE Trans. Ind. Appl.*, vol. 30, no. 3, pp. 746–759, May 1994.

[56] F. Auger, M. Hilairet, J. M. Guerrero, E. Monmasson, T. Orlowska-Kowalska, and S. Katsura, "Industrial applications of the kalman filter: A review," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5458–5471, Dec. 2013.

[57] Kaiyu Wang, J. Chiasson, M. Bodson, and L. M. Tolbert, "A nonlinear least-squares approach for identification of the induction motor parameters," *IEEE Trans. Autom. Control*, vol. 50, no. 10, pp. 1622–1628, Oct. 2005.

[58] S. R. Shaw and S. B. Leeb, "Identification of induction motor parameters from transient stator current measurements," *IEEE Trans. Ind. Electron.*, vol. 46, no. 1, pp. 139–149, Feb. 1999.

[59] A. Boglietti, A. Cavagnino, and M. Lazzari, "Computational algorithms for induction-motor equivalent circuit parameter determination - part i: Resistances and leakage reactances," *IEEE Trans. Ind. Electron.*, vol. 58, no. 9, pp. 3723–3733, Sep. 2011.

[60] ——, "Computational algorithms for induction motor equivalent circuit parameter determination - part ii: Skin effect and magnetizing characteristics," *IEEE Trans. Ind. Electron.*, vol. 58, no. 9, pp. 3734–3740, Sep. 2011.

[61] Z. Ling, L. Zhou, S. Guo, and Y. Zhang, "Equivalent circuit parameters calculation of induction motor by finite element analysis," *IEEE Trans. Magn.*, vol. 50, no. 2, pp. 833–836, Feb. 2014.

[62] M. Cirrincione, M. Pucci, G. Cirrincione, and G. Capolino, "Constrained minimization for parameter estimation of induction motors in saturated and unsaturated conditions," *IEEE Trans. Ind. Electron.*, vol. 52, no. 5, pp. 1391–1402, Oct. 2005.

[63] L. Fagiano, M. Lauricella, D. Angelosante, and E. Ragaini, "Identification of induction motors using smart circuit breakers," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 6, pp. 2638–2646, Nov. 2019.

[64] Y. He, Y. Wang, Y. Feng, and Z. Wang, "Parameter identification of an induction machine at standstill using the vector constructing method," *IEEE Trans. Power Electron.*, vol. 27, no. 2, pp. 905–915, Feb. 2012.

[65] F. Duan, R. Zivanovic, S. Al-Sarawi, and D. Mba, "Induction motor parameter estimation using sparse grid optimization algorithm," *IEEE Trans. Ind. Informat.*, vol. 12, no. 4, pp. 1453–1461, Aug. 2016.

[66] K. S. Huang, Q. H. Wu, and D. R. Turner, "Effective identification of induction motor parameters based on fewer measurements," *IEEE Trans. Energy Convers.*, vol. 17, no. 1, pp. 55–60, Mar. 2002.

[67] Jong-Wook Kim and Sang Woo Kim, "Parameter identification of induction motors using dynamic encoding algorithm for searches (deas)," *IEEE Trans. Energy Conv.*, vol. 20, no. 1, pp. 16–24, Mar. 2005.

[68] D. Bhowmick, M. Manna, and S. K. Chowdhury, "Estimation of equivalent circuit parameters of transformer and induction motor from load data," *IEEE Trans. Ind. Appl.*, vol. 54, no. 3, pp. 2784–2791, May 2018.

[69] Z. Liu, H. Wei, X. Li, K. Liu, and Q. Zhong, "Global identification of electrical and mechanical parameters in pmsm drive based on dynamic self-learning pso," *IEEE Trans. Power Electron.*, vol. 33, no. 12, pp. 10 858–10 871, Dec. 2018.

[70] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, U.K.: Cambridge University Press, 2004.

[71] A. M. Alturas, S. M. Gadoue, B. Zahawi, and M. A. Elgendy, "On the identifiability of steady-state induction machine models using external measurements," *IEEE Trans. Energy Conv.*, vol. 31, no. 1, pp. 251–259, Mar. 2016.

[72] D. M. Reed, H. F. Hofmann, and J. Sun, "Offline identification of induction machine parameters with core loss estimation using the stator current locus," *IEEE Trans. Energy Convers.*, vol. 31, no. 4, pp. 1549–1558, Dec. 2016.

[73] R. D. Zimmerman and H. Wang. (2019, Jun.) MATPOWER interior point solver (MIPS) user's manual, version 1.3.1. [Online]. Available: https://matpower.org/docs/MIPS-manual-1.3.1.pdf

[74] R. Madani, J. Lavaei, and R. Baldick, "Convexification of power flow equations in the presence of noisy measurements," *IEEE Trans. Autom. Control*, vol. 64, no. 8, pp. 3101–3116, Aug. 2019.

[75] M. Iqbal, A. I. Bhatti, S. I. Ayubi, and Q. Khan, "Robust parameter estimation of nonlinear systems using sliding-mode differentiator observer," *IEEE Trans. Indust. Electron.*, vol. 58, no. 2, pp. 680–689, Feb. 2011.

[76] F. Alonge, F. D'Ippolito, and A. Sferlazza, "Sensorless control of induction-motor drive based on robust kalman filter and adaptive speed estimation," *IEEE Trans. Indust. Electron.*, vol. 61, no. 3, pp. 1444–1453, Mar. 2014.

[77] M. S. Rafaq and J. Jung, "A comprehensive review of state-of-the-art parameter estimation techniques for permanent magnet synchronous motors in wide speed range," *IEEE Trans. Indust. Informat.*, vol. 16, no. 7, pp. 4747–4758, July 2020.

[78] M. Barut, S. Bogosyan, and M. Gokasan, "Speed-sensorless estimation for induction motors using extended kalman filters," *IEEE Trans. Indust. Electron.*, vol. 54, no. 1, pp. 272–280, Feb. 2007.

[79] Y. Shi, K. Sun, L. Huang, and Y. Li, "Online identification of permanent magnet flux based on extended kalman filter for ipmsm drive with position sensorless control," *IEEE Trans. Indust. Electron.*, vol. 59, no. 11, pp. 4169–4178, Nov. 2012.

[80] B. Aubert, J. Régnier, S. Caux, and D. Alejo, "Kalman-filter-based indicator for online interturn short circuits detection in permanent-magnet synchronous generators," *IEEE Trans. Indust. Electron.*, vol. 62, no. 3, pp. 1921–1930, Mar. 2015.

[81] X. Li and R. Kennel, "General formulation of kalman-filter-based online parameter identification methods for vsi-fed pmsm," *IEEE Trans. Indust. Electron.*, vol. 68, no. 4, pp. 2856–2864, Apr. 2021.

[82] D. Woodbury and J. Junkins, "On the consider kalman filter," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, ON, Canada, 2010, pp. 2010–7752.

[83] P. J. Hadwin, T. A. Sipkens, K. A. Thomson, F. Liu, and K. J. Daun, "Kalman filter approach for uncertainty quantification in time-resolved laser-induced incandescence," *OSA*, vol. 35, no. 3, pp. 386–396, Mar 2018.

[84] K. M. Brink, "Partial-update schmidt–kalman filter," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 9, pp. 2214–2228, May 2017.

[85] J. H. Ramos, K. M. Brink, and J. E. Hurtado, "Square Root Partial-Update Kalman Filter," in *22th International Conference on Information Fusion*, Ottawa, ON, Canada, 2019, pp. 1–8.

[86] C. Yang, J. Zheng, X. Ren, W. Yang, H. Shi, and L. Shi, "Multi-sensor kalman filtering with intermittent measurements," *IEEE Trans. Autom. Control*, vol. 63, no. 3, pp. 797–804, Mar. 2017.

[87] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman filtering with intermittent observations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1453–1464, Sep. 2004.

[88] S. Kluge, K. Reif, and M. Brokate, "Stochastic stability of the extended kalman filter with intermittent observations," *IEEE Trans. Autom. Control*, vol. 55, no. 2, pp. 514–518, Feb. 2010.

[89] L. Li, D. Yu, Y. Xia, and H. Yang, "Event-triggered ukf for nonlinear dynamic systems with packet dropout," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 18, pp. 4208–4226, Mar. 2017.

[90] M. Kooshkbaghi and H. J. Marquez, "Event-triggered discrete-time cubature kalman filter for nonlinear dynamical systems with packet dropout," *IEEE Trans. Autom. Control*, vol. 65, no. 5, pp. 2278–2285, May 2020.

[91] K. Xiong, H. Zhang, and C. Chan, "Performance evaluation of UKF-based nonlinear filtering," *Automatica*, vol. 42, no. 2, pp. 261–270, Dec. 2006.

[92] K. Reif, S. Gunther, E. Yaz, and R. Unbehauen, "Stochastic stability of the discrete-time extended kalman filter," *IEEE Trans. Autom. Control*, vol. 44, no. 4, pp. 714–728, Apr. 1999.

[93] L. Salvatore and S. Stasi, "Application of ekf to parameter and state estimation of pmsm drive," *IEE Proceedings B (Electric Power Applications)*, vol. 139, no. 3, pp. 155–164, May 1992.

[94] Z. Yin, G. Li, Y. Zhang, and J. Liu, "Symmetric-strong-tracking-extended-kalman-filter-based sensorless control of induction motor drives for modeling error reduction," *IEEE Trans. Indust. Informat.*, vol. 15, no. 2, pp. 650–662, Feb. 2019.

[95] F. L. Lewis, L. Xie, and D. Popa, *Optimal and robust estimation: with an introduction to stochastic control theory*, 2nd ed. Boca Raton: CRC Press, 2008.

[96] L. Li and Y. Xia, "Stochastic stability of the unscented kalman filter with intermittent observations," *Automatica*, vol. 48, no. 5, pp. 978–981, Mar. 2012.

[97] Anaheim Automation. Bly17 series brushless dc motors. [Online]. Available: https://www.anaheimautomation.com/manuals/brushless/L010228%20-%20BLY17%20Series%20Spec%20Sheet.pdf

[98] S. Carpiuc and C. Villegas, "FPGA-based rapid control prototyping of permanent magnet synchronous motor servo drives," in *International Exhibition and Conference for Power Electronics, Intelligent Motion, Renewable Energy and Energy Management*, 2019, pp. 1–6.

[99] C. De Angelo, G. Bossio, J. Solsona, G. O. Garcia, and M. I. Valla, "Mechanical sensorless speed control of permanent-magnet ac motors driving an unknown load," *IEEE Trans. Indust. Electron.*, vol. 53, no. 2, pp. 406–414, Apr. 2006.

[100] A. P. Yadav, S. Xu, B. C. Schafer, and A. Davoudi, "Hardware-assisted simulation of voltage-behind-reactance models of electric machines on fpga," *IEEE Transactions on Energy Conversion*, vol. 35, no. 3, pp. 1247–1257, Sep. 2020.

[101] L. Wang, J. Jatskevich, and H. W. Dommel, "Re-examination of synchronous machine modeling techniques for electromagnetic transient simulations," *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1221–1230, Aug 2007.

[102] R. E. Doherty and C. A. Nickle, "Synchronous machines-iii torque-angle characteristics under transient conditions," *Transactions of the American Institute of Electrical Engineers*, vol. XLVI, pp. 1–18, Jan 1927.

[103] S. D. Pekarek, O. Wasynczuk, and H. J. Hegner, "An efficient and accurate model for the simulation and analysis of synchronous machine/converter systems," *IEEE Transactions on Energy Conversion*, vol. 13, no. 1, pp. 42–48, March 1998.

[104] S. D. Pekarek, E. A. Walters, and B. T. Kuhn, "An efficient and accurate method of representing magnetic saturation in physical-variable models of synchronous machines," *IEEE Transactions on Energy Conversion*, vol. 14, no. 1, pp. 72–79, March 1999.

[105] L. Wang and J. Jatskevich, "A voltage-behind-reactance synchronous machine model for the emtp-type solution," *IEEE Transactions on Power Systems*, vol. 21, no. 4, pp. 1539–1549, Nov 2006.

[106] L. Wang, J. Jatskevich, and S. D. Pekarek, "Modeling of induction machines using a voltage-behind-reactance formulation," *IEEE Transactions on Energy Conversion*, vol. 23, no. 2, pp. 382–392, June 2008.

[107] L. Wang and J. Jatskevich, "Including magnetic saturation in voltage-behind-reactance induction machine model for emtp-type solution," *IEEE Transactions on Power Systems*, vol. 25, no. 2, pp. 975–987, May 2010.

[108] L. Wang and J. Jatskevich, "Magnetically-saturable voltage-behind-reactance synchronous machine model for emtp-type solution," *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 2355–2363, Nov 2011.

[109] M. Chapariha, F. Therrien, J. Jatskevich, and H. W. Dommel, "Explicit formulations for constant-parameter voltage-behind-reactance interfacing of synchronous machine models," *IEEE Transactions on Energy Conversion*, vol. 28, no. 4, pp. 1053–1063, Dec 2013.

[110] F. Therrien, M. Chapariha, and J. Jatskevich, "Constant-parameter voltage-behind-reactance induction machine model including main flux saturation," *IEEE Transactions on Energy Conversion*, vol. 30, no. 1, pp. 90–102, March 2015.

[111] B. Lu, X. Wu, H. Figueroa, and A. Monti, "A low-cost real-time hardware-in-the-loop testing approach of power electronics controls," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 2, pp. 919–931, April 2007.

[112] G. G. Parma and V. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," *IEEE Transactions on Power Delivery*, vol. 22, no. 2, pp. 1235–1246, April 2007.

[113] L. Herrera, C. Li, X. Yao, and J. Wang, "Fpga-based detailed real-time simulation of power converters and electric machines for ev hil applications," *IEEE Transactions on Industry Applications*, vol. 51, no. 2, pp. 1702–1712, March 2015.

[114] MathWorks, "Hdl coder, getting started guide," 2019. [Online]. Available: https://www.mathworks.com/help/pdf_doc/hdlcoder/hdlcoder_gs.pdf

[115] N. Roshandel Tavana and V. Dinavahi, "A general framework for fpga-based real-time emulation of electrical machines for hil applications," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2041–2053, April 2015.

[116] B. Jandaghi and V. Dinavahi, "Real-time hil emulation of faulted electric machines based on nonlinear mec model," *IEEE Transactions on Energy Conversion*, vol. 34, no. 3, pp. 1190–1199, Sep. 2019.

[117] O. Tremblay, D. Rimorov, R. Gagnon, and H. Fortim-Blanchette, "A multi-time-step transmission line interface for power hardware-in-the-loop simulators," *IEEE Transactions on Energy Conversion*, pp. 1–1, 2019.

[118] H. Chen, S. Sun, D. C. Aliprantis, and J. Zambreno, "Dynamic simulation of electric machines on fpga boards," in *2009 IEEE International Electric Machines and Drives Conference*, May 2009, pp. 1523–1528.

[119] S. Ebrahimi, N. Amiri, L. Wang, and J. Jatskevich, "Efficient modeling of six-phase pm synchronous machine-rectifier systems in state-variable-based simulation programs," *IEEE Transactions on Energy Conversion*, vol. 33, no. 3, pp. 1557–1570, Sep. 2018.

[120] N. Amiri, S. Ebrahimi, Y. Huang, J. Jatskevich, and S. D. Pekarek, "Constant-parameter voltage-behind-reactance modeling of five-phase synchronous machines with air-gap flux harmonics," *IEEE Transactions on Energy Conversion*, pp. 1–1, 2019.

[121] R. Mirzahosseini, "Fpga-based real-time simulation platform for power grids including multiple converters," Ph.D. dissertation, University of Toronto, 2017. [Online]. Available: https://tspace.library.utoronto.ca/handle/1807/93029

[122] K. A. Corzine, B. T. Kuhn, S. D. Sudhoff, and H. J. Hegner, "An improved method for incorporating magnetic saturation in the q-d synchronous machine

model," *IEEE Transactions on Energy Conversion*, vol. 13, no. 3, pp. 270–275, Sep. 1998.

[123] M. Tenenbaum and H. Pollard, *Ordinary differential equations*. Dover Publications Inc., New York, 1986.

[124] Y. X. Wang and J. M. Wen, "Gear method for solving differential equations of gear systems," *Journal of Physics: Conference Series*, vol. 48, pp. 143–148, oct 2006.

[125] I. Kuon, R. Tessier, and J. Rose, *FPGA Architecture: Survey and Challenges*. now, 2008. [Online]. Available: https://ieeexplore.ieee.org/document/8187326

[126] Xilinx, "Xilinx announces the world's largest fpga featuring 9 million system logic cells," Tech. Rep., 2019. [Online]. Available: www.xilinx.com/news/press/2019/xilinx-announces-the-world-s-largest-fpga-featuring-9-million-system-logic-cells.html

[127] B.Carrion Schafer, "Acceleration of the discrete element method on a reconfigurable co-processor," Ph.D. dissertation, The University of Birmingham, UK, 2003.

[128] B. Carrion Schafer, S. F. Quigley, and A. H. C. Chan, "Analysis and implementation of the discrete element method using a dedicated highly parallel architecture in reconfigurable computing," in *Proceedings. 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, April 2002, pp. 173–181.

[129] B. A. Draper, J. R. Beveridge, A. P. W. Bohm, C. Ross, and M. Chawathe, "Accelerated image processing on fpgas," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1543–1551, Dec 2003.

[130] D. B. Thomas and W. Luk, "Credit risk modelling using hardware accelerated monte-carlo simulation," in *2008 16th International Symposium on Field-Programmable Custom Computing Machines*, April 2008, pp. 229–238.

[131] U. B. of labor statistics, "Engineering statistics," 2017. [Online]. Available: https://www.bls.gov/ooh/architecture-and-engineering/home.htm

[132] B. Carrion Schafer and Z. Wang, "High-level synthesis design space exploration: Past, present and future," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–12, 2019.

[133] B. C. Schafer and K. Wakabayashi, "Machine learning predictive modelling high-level synthesis design space exploration," *IET Computers Digital Techniques*, vol. 6, no. 3, pp. 153–159, May 2012.

[134] S. Xu and B. C. Schafer, "Exposing approximate computing optimizations at different levels: From behavioral to gate-level," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 11, pp. 3077–3088, Nov 2017.

[135] Dong Liu and B. C. Schafer, "Efficient and reliable high-level synthesis design space explorer for fpgas," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, Aug 2016, pp. 1–8.

[136] NEC CyberWorkBench, 2019. [Online]. Available: www.cyberworkbench.com

[137] Matlab, "Hdl coder," 2019. [Online]. Available: https://www.mathworks.com/products/hdl-coder.html

[138] B. Carrion Schafer, S. F. Quigley, and A. H. C. Chan, "Analysis and implementation of the discrete element method using a dedicated highly parallel architecture in reconfigurable computing," in *Proceedings. 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, April 2002, pp. 173–181.

[139] B. C. Schafer and K. Wakabayashi, "Divide and conquer high-level synthesis design space exploration," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 17, no. 3, pp. 29:1–29:19, Jul. 2012.

[140] Intel Quartus II, 2019. [Online]. Available: www.altera.com

[141] L. Wang, J. Jatskevich, C. Wang, and P. Li, "A voltage-behind-reactance induction machine model for the emtp-type solution," *IEEE Transactions on Power Systems*, vol. 23, no. 3, pp. 1226–1238, Aug 2008.

[142] F. Therrien, L. Wang, M. Chapariha, and J. Jatskevich, "Constant-parameter interfacing of induction machine models considering main flux saturation in emtp-type programs," *IEEE Transactions on Energy Conversion*, vol. 31, no. 1, pp. 12–26, March 2016.

[143] J. Andrus, "Stability of a multi-rate method for numerical integration of ode's," *Computers & Mathematics with applications*, vol. 25, no. 2, pp. 3–14, 1993.

BIOGRAPHICAL STATEMENT

Ajay Pratap Yadav received Bachelor's and Master's degrees in Electrical engineering from the Indian Institute of Technology Roorkee and the Indian Institute of Technology Kanpur, in 2010 and 2014, respectively. He is currently pursuing his Ph.D. at the University of Texas at Arlington. His research interests include electric machine modeling, system identification, optimization, and microgrids.