BGCL: Learning Constitutive Laws for System Identification

Abhishek Patkar PATKAR2@MIT.EDU
Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA.

Kamal Youcef Toumi YOUCEF@MIT.EDU
Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA.

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Nonlinear system identification of dynamical systems is a challenging problem. Recently, learning based approaches have made attempts to embed physical priors in the learning model to improve model identification of dynamical systems. In this paper, we propose the Bond Graph based Constitutive Law learning (BGCL) framework to learn analytical expressions for constitutive laws and thus identify models for physical dynamical systems. Simulation studies conducted on a spring mass system and synchronous three phase motor are used to validate the proposed framework.

Keywords: Bond graphs, constitutive laws, electric motors, symbolic regression

1. Introduction

System identification of dynamical systems has been studied extensively. This problem becomes especially challenging when the model of the system to be identified is nonlinear in nature. Accurate models are necessary for model based control strategies. In recent years, data driven learning methods are being increasingly utilized in identification of nonlinear system dynamics. Brunton et al. (2016a) introduced a sparse regression algorithm to identify symbolic nonlinear dynamics for autonomous systems. Subsequently, Brunton et al. (2016b) applied the algorithm, SINDYc, to non-autonomous systems including a control input. Black box methods such as ResNets Lu et al. (2018) have been used to approximate ordinary differential equations (ODE). Chen et al. (2018) introduced a new family of deep neural networks for end-to-end training of ODEs. However, common learning methodologies such as neural network based approaches have been known to suffer from problems such as overfitting and low generalizability. To overcome these issues, using knowledge of physics in conjunction with data driven methods has gained prominence. Physics informed Neural Networks (PINNs) Raissi et al. (2019) have been shown to learn solutions to PDEs from data.

Bond graphs, developed by Paynter are an excellent tool for modelling dynamical systems Paynter (1961), Paynter (1992),Rosenberg (1993). A bond is used to represent transfer of energy between nodes of a bond graph. With basic building blocks consisting of energy storage elements, energy dissipative elements, sources/sinks, transformers and gyrators, one can model arbitrarily complex dynamics such as RLC circuits, Permanent Magnet Synchronous Motors (PMSM) etc Borutzky (2011). When modeling physical systems using a bond graph approach, it can be readily seen that non-linearity in a system can enter through nonlinear energy storage elements, nonlinear dissipative elements or nonlinear junction relations. These relations are referred to as the constitutive laws of those elements. Another important property encoded in bond graphs is causality. Causality determines dependent and independent energy storage elements; independent energy storage elements, referenced using preferred integral causality yield state variables indicating the minimum order required to sufficiently represent the physical system Borutzky (2011). One can readily arrive at

the familiar state space equations or the port Hamiltonian equations for control from bond graphs Rosenberg (1971), Donaire and Junco (2009).

Related Work. Several recent studies have been conducted on using physical invariants to inform the learning of models for dynamical systems. Ahmadi et al. (2018) proposed learning Lagrangians and Hamiltonians from trajectory data. These methods assume the nonlinear basis functions apriori and only tune the linear coefficients from training data, thereby limiting the expressibility of the function approximator. Greydanus et al. (2019) proposed the Hamiltonian Neural Network (HNN) for conservative systems and demonstrated their superior trajectory prediction capabilities over vanilla neural networks. Subsequent works such as Sosanya and Greydanus (2022), Zhong et al. (2020) have expanded HNN to encompass dissipative and non-autonomous systems. Hamiltonian based approaches assume knowledge of canonical co-ordinates that are not easily measurable for physical systems. For instance, in the mechanical domain, one can measure the velocity of a moving object but not its momentum. Similarly, in the electrical domain, one usually measures the electric current but not the magnetic flux. Lagrangian Neural Networks, proposed by Cranmer et al. (2020), use a neural network to learn the Lagrangian instead of the Hamiltonian to overcome the limitation of using canonical coordinates. For physical systems, having compact symbolic expressions is desirable. Seo et al. (2003) utilized genetic programming to construct bond graphs of multi-domain systems. However, the constitutive laws used therein are assumed to be linear.

Main contributions To exploit physical structure of the problem in learning the unknown dynamics, we emphasize using a bond graph based approach. For a class of physical systems, we propose a novel approach that learns the energy stored in a physical system in terms of measurable quantities. We enforce physics informed constraints derived from bond graphs while learning the function describing the stored energy, inform input excitation design and improve learning accuracy. We also demonstrate how the constitutive laws of the energy storage elements can be extracted from the stored energy function. Simulation studies on spring mass system and synchronous three-phase electric motors are used to validate the proposed approach.

2. Background

Notation: Variables in bold indicate vectors. $\dot{p}$ denotes the time derivative $\frac{dp}{dt}$ of variable p.

Bond graphs use power and energy variables to arrive at the governing equations of a dynamic system. Efforts $\boldsymbol{e}$ and flows $\boldsymbol{f}$ are called the power variables whereas as momentum $\boldsymbol{p}$ and displacement $\boldsymbol{q}$ are the energy variables. These are related to each other as $\boldsymbol{e} = \dot{\boldsymbol{p}}$ and $\boldsymbol{f} = \dot{\boldsymbol{q}}$. Across a bond, $\boldsymbol{e}$ and $\boldsymbol{f}$ are considered conjugate power variables as their inner product informs the power flowing through that bond. Tables 1 and 2 provide examples of energy and power variables respectively in some common modeling domains. Depending on the choice of state-variables, different formulations describing the dynamics of the same system are possible. Fig. 1 indicates the four different choices for state variables leading to four different formulations- Hamiltonian, Lagrangian, Co-Lagrangian and Power variable formulation. All four formulations are related to each other through Legendre transforms Duindam et al. (2009).

There are two types of energy storage elements: capacitive and inductive. An ideal capacitor is an energy storage element whose constitutive law relates effort $\boldsymbol{e}$ to displacement $\boldsymbol{q}$. Similarly, an inductor's constitutive law relates flow $\boldsymbol{f}$ to momentum $\boldsymbol{p}$. For example, in mechanical systems, an ideal spring is a capacitive energy storage element with the linear constitutive law $e = kq$. Here, the effort e is the restoring force of the spring, q is the displacement of the spring and k is the spring

Domain	Momentum ($\boldsymbol{p}$)			Displacement ($\boldsymbol{q}$)		
	Quantity	Variable	Units	Quantity	Variable	Units
Mechanical Translational	Momentum	p	$\mathrm{N\,s}$	Displacement	x	m
Mechanical Rotational	Angular momentum	L	$\mathrm{N\,m\,s}$	Angle	θ	rad
Electrical	Magnetic flux	ϕ	$\mathrm{V\,s}$	Charge	q	C

Table 1: Energy variables across different domains

Domain	Effort ($\boldsymbol{e}$)			Flow ($\boldsymbol{f}$)		
	Quantity	Variable	Units	Quantity	Variable	Units
Mechanical Translational	Force	F	N	Velocity	v	$\mathrm{m/s}$
Mechanical Rotational	Torque	τ	$\mathrm{N\,m}$	Angular velocity	ω	$\mathrm{rad/s}$
Electrical	Voltage	V	V	Current	I	A

Table 2: Power variables across different domains

constant. An ideal mass acts as an inductive element with the constitutive law $f = \frac{p}{m}$ where the flow f is the velocity of the moving mass and p is its momentum. Fig. 2, provides an illustrative relation of different constitutive laws for different elements.

3. Problem Definition and Preliminaries

In this section, we list the assumptions and formulate our problem statement.

Key Assumptions

1. A bond-graph, with energy storage elements in integral causality is available.
2. The constitutive law for resistive elements, transformers and gyrators are known whereas constitutive laws for energy storage elements are unknown.

Integral causality is the preferred causality assignment for energy storage elements in bond graphs Borutzky (2011). Algebraic loops in causality assignment lead to differential-algebraic equations (DAE) describing the dynamics which are beyond the scope of this paper. In differential causality, an energy storage element becomes dependent and thus its flow or effort can be characterized as a function of another independent energy storage element in the bond graph. Transformers and gyrators are energy transducers that transform energy from one domain to another or from one part of the system to another. Such transducers are characterized by linear, time-invariant algebraic relations between the relevant power variables. Gear systems, belt drives, electrical transformers etc. are some examples of such elements. Thus, it is reasonable to assume that their constitutive relation is known apriori. Assuming the availability of resistive constitutive laws is restrictive.

Let $\boldsymbol{u} \in \mathcal{R}^n$ denote the input power variable and $\boldsymbol{y}_u \in \mathcal{R}^n$ the corresponding conjugate power variable. Let $\boldsymbol{\phi}_i(\cdot) \in \mathcal{R}^{m_i}$ denote the constitutive law, $E_i(\cdot)$ the energy and $E_i^{'}(\cdot)$ the co-energy of the i^{th} energy storage element. Let $\boldsymbol{e}_i$ denote the effort variable and $\boldsymbol{f}_i$ the flow variable corresponding to the i^{th} energy storage element. Using the definition of co-energy and constitutive law,

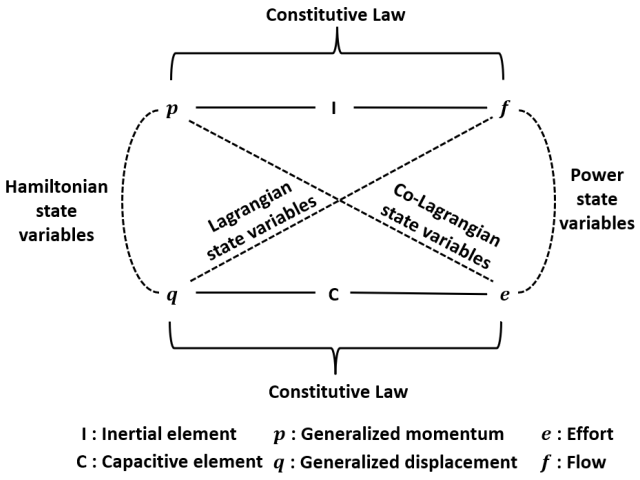
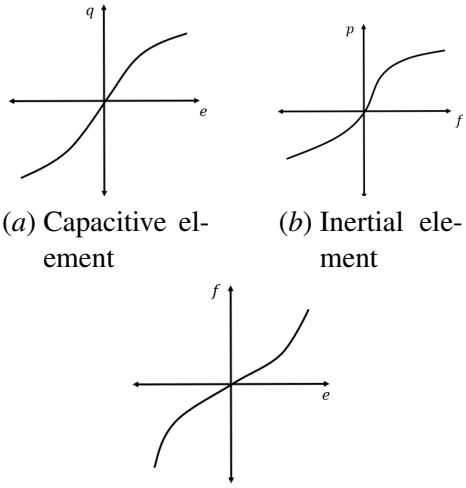


Figure 1: Interrelation of different state variable choices



(a) Capacitive element

(b) Inertial element

(c) Resistive element

Figure 2: Different examples of constitutive laws

for the i^{th} capacitive storage element, we have

$$\boldsymbol{\phi}_i(\boldsymbol{e}_i)^T \boldsymbol{e}_i = E_i(\boldsymbol{e}_i) + E_i^{'}(\boldsymbol{e}_i), \tag{1}$$

Rewriting the co-energy in terms of the constitutive law we get,

$$\boldsymbol{\phi}_i(\boldsymbol{e}_i)^T \boldsymbol{e}_i - \int_0^{\boldsymbol{e}_i} \boldsymbol{\phi}_i(\tilde{\boldsymbol{e}}_i)^T d\tilde{\boldsymbol{e}}_i = E_i(\boldsymbol{e}_i) \tag{2}$$

Similarly, we can write the relation for the j^{th} inductive element,

$$\boldsymbol{\phi}_j(\boldsymbol{f}_j)^T \boldsymbol{f}_j - \int_0^{\boldsymbol{f}_j} \boldsymbol{\phi}_j(\tilde{\boldsymbol{f}}_j) d\tilde{\boldsymbol{f}}_j = E_j(\boldsymbol{f}_j) \tag{3}$$

The total energy stored in the system is then given by eq. (4)

$$E(\boldsymbol{f}, \boldsymbol{e}) = \sum_{i=1}^{k} \left(\boldsymbol{\phi}_i(\boldsymbol{f}_i)^T \boldsymbol{f}_i - \int_0^{f_i} \boldsymbol{\phi}_i(\tilde{\boldsymbol{f}}_i)^T d\tilde{\boldsymbol{f}}_i \right) + \sum_{j=1}^{l} \left(\boldsymbol{\phi}_j(\boldsymbol{e}_j)^T \boldsymbol{e}_j - \int_0^{\boldsymbol{e}_j} \boldsymbol{\phi}_j(\tilde{\boldsymbol{e}}_j)^T d\tilde{\boldsymbol{e}}_j \right) \tag{4}$$

Unlike the Hamiltoninan $H(\boldsymbol{q}, \boldsymbol{p})$, the stored energy function in eq. (4) is a function of $\boldsymbol{e}, \boldsymbol{f}$ which can be measured in physical systems. To extract the constitutive laws, we take partial derivatives w.r.t to the efforts and flows of a particular storage element. It can be shown that the following relation exists between the stored energy and the Jacobian matrix $\boldsymbol{J}(\cdot)$ of the i^{th} constitutive law.

$$\frac{\partial E}{\partial \boldsymbol{f}_i} = \boldsymbol{J}(\boldsymbol{f}_i)^T \boldsymbol{f}_i \tag{5}$$

Eq. (18) gives the general condition for each entry of a constitutive law in the general multiport nonlinear case. The derivation has been provided in the appendix. Thus, by learning a functional form of E, one may extract the Jacobian and subsequently the constitutive laws of the energy storage elements. In this paper, we train a PySR model Cranmer (2023) to learn the stored energy function.
Stored Energy Computation: To learn E, we need to first have access to the stored energy. In this subsection, the stored energy computation is described. At any time t, the power into the system P_{in}, the power dissipated P_{dis} and the power stored P_{st} are related as:

$$\begin{aligned} P_{st}(t) &= P_{in}(t) - P_{dis}(t) \\ &= \boldsymbol{u}^T \boldsymbol{y}_u - P_{dis}(t) \end{aligned}$$

Since the input signals are measured and the resistive field is assumed to be known, the stored power $P_{st}(t)$ can be computed. Now the stored energy at any time instant t can be computed by numerically integrating the stored power expression:

$$E(t) = \int_0^t P_{st}(\tilde{t}) d\tilde{t} \tag{6}$$

4. Proposed framework

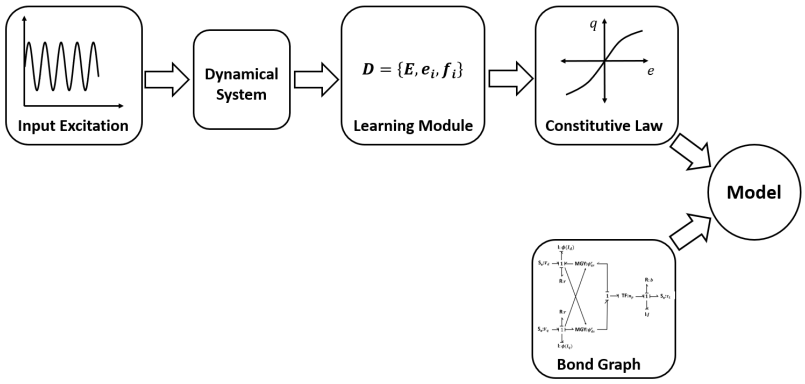


Figure 3: Proposed Bond graph Constitutive law (BGCL) based model identification framework

Fig. 3 illustrates the proposed framework for identifying the system model. The key idea of the proposed framework is to enforce the constraint in eq. (18) coupled with an appropriately designed control input to learn the system dynamics. The dynamical system under consideration is excited by a designed input signal $\boldsymbol{u}_{in}$. The relevant efforts $\boldsymbol{e}$ and flows $\boldsymbol{f}$ are measured and used to compute the energy stored in the system $\boldsymbol{E}$ using (6). These signals are then used as training data for the learning module. To learn interpretable models for physical systems, we use PySR to learn the stored energy $E(t)$ as a function of $\boldsymbol{e}, \boldsymbol{f}$. Subsequently, eq. (18) can be solved to recover the constitutive law. The learnt constitutive laws along with the bond-graph are used to infer the system dynamics.

We now consider different cases to simplify eq. (18). Without loss of generality, the subsequent analysis is performed on inductive elements but it can be repeated for capacitive elements.

4.1. Linear Models

For a linear system, the constitutive law $\boldsymbol{\phi(f_i)}$ is a linear function of f_i and hence $\boldsymbol{J(f_i)}$ is a constant. Thus Eq. (18) simplifies to

$$\frac{\partial E}{\partial \boldsymbol{f}_i} = \boldsymbol{L}_{\boldsymbol{f}_i}^T \boldsymbol{f}_i \tag{7}$$

where $\boldsymbol{L}_{\boldsymbol{f}_i}$ is a constant matrix. Thus, the equation search can be constrained to look for functions of the form:

$$E(\boldsymbol{f}_1, \cdots, \boldsymbol{f}_n, \boldsymbol{e}_1, \cdots, \boldsymbol{e}_m) = \sum_{i=0}^{n} \frac{1}{2} \boldsymbol{f}_i^T \boldsymbol{L}_{f_i} \boldsymbol{f}_i + \sum_{j=0}^{m} \frac{1}{2} \boldsymbol{e}_j^T \boldsymbol{L}_{e_j} \boldsymbol{e}_j \tag{8}$$

This is the familiar quadratic form of stored energy in a linear system. In line with linear system identification theory, a step input u can be used to excite the dynamic system and collect the dataset E, e, f. Once the analytical form for E has been identified, for each m dimensional multi-port element, we can use m distinct data-points to identify the constant matrix $\boldsymbol{L}_{\boldsymbol{f}_i}$.

4.2. Nonlinear Models

While a general closed form expression like eq. (8) is unavailable for nonlinear models, (18) implies searching for separable energy functions of the form: $E(\boldsymbol{f}_1, \cdots, \boldsymbol{f}_n, \boldsymbol{e}_1, \cdots, \boldsymbol{e}_m) = \sum_{i=0}^{n} g_i(\boldsymbol{f}_i) + \sum_{j=0}^{m} h_j(\boldsymbol{e}_j)$.

Input Excitation Design: To leverage separability, the input u can be designed such that all flow variables except one are regulated to a constant. This can be accomplished using a PI control design since all flow variables are independent. Thus, each function g_i, h_j can be learnt individually, further narrowing the equation search. In case of nonlinear single port elements, (18) reduces to the scalar equation

$$\frac{\partial E}{\partial f_i} = J(f_i) f_i \tag{9}$$

Since the input excitation regulated all other flow variables to a constant, the L.H.S of (9) can be approximated using the first order method of finite differences as $\frac{\partial E}{\partial f_i} \approx \frac{\Delta E}{\Delta f_i}$, thereby allowing us to directly learn the Jacobian $J(\cdot)$.

5. Simulation Studies

In this section, we apply the proposed approach to learn the dynamics of the classical spring-mass system with a nonlinear spring and a three-phase electric motor with nonlinear flux relations. The signal measurements in the simulation studies are assumed to be noiseless. Since the stored energy is computed using (6), any noise in measurements will be integrated and distort the stored energy value. The proposed approach attempts to learn the energy as a function of state; since measurement noise would be state independent, the proposed approach should work in the presence of noisy measurements. Nevertheless, the effect of noisy measurements needs to be extensively studied and will be part of future work.

Baseline: To evaluate the performance of the BGCL framework, we compare the state prediction performance of the model learnt using the BGCL framework to that of an appropriate baseline. Considering the bond graph informed learning approach, other physics informed approaches such as Hamiltonian Neural Networks (HNNs) may be considered appropriate; however, such approaches

rely on measurement of canonical co-ordinates, whereas the BGCL approach utilizes power variables. SINDYc, a data-driven method, attempts to learn a sparse symbolic representation of unknown dynamics of non-autonomous systems.

5.1. Spring mass system

The familiar state space equation of motion for a mass tied to a wall through a spring is given by:

$$\dot{x}_1(t) = x_2(t), \tag{10}$$

$$\dot{x}_2(t) = -\frac{g(x_1(t))}{m} + \frac{1}{m}u(t). \tag{11}$$

where $x_1(t)$ is the displacement, $x_2(t)$ is the velocity. $g(x_1)$ is the unknown spring force and m is the unknown mass. The unknown nonlinear constitutive law is the power law for springs $e_1(t) = k|x|^{0.5}$, with $k = 0.5$. The mass m was taken to be 1. A sinusoidal input $u = sin(2\pi t)$ was used to excite the system for time $t = [0, 2.5]$ s. The details of the PySR model are provided in the appendix 7.2.2. The energy function learnt by PySR is

$$\hat{E}(e_1, f_1) = 2.67e_1^3 + 0.5f_1^2 \tag{12}$$

Using (18), the jacobian for the spring constitutive law is $J(e_1) = 8e_1$. Thus the constitutive law for the spring is found by integrating the Jacobian giving $x = 4e_1^2$ or $e_1 = 0.5|x|^{0.5}$. Thus the true constitutive laws for both the spring and mass are recovered through the BGCL approach.

5.2. Three phase synchronous motors

Motors such as Synchronous motors (SyR), Interior-permanent magnet motors (IPM) exhibit nonlinear relations between the d, q axis currents Guglielmi et al. (2006); Stumberger et al. (2003); Bianchi and Bolognani (1998). Significant research efforts have been directed towards identifying symbolic d, q axis flux models to enable advanced model based control such as Maximum Torque per Ampere (MTPA), Maximum Torque per Volts (MTPV), and Model Predictive Control (MPC) Ortombina et al. (2018),Bedetti et al. (2016),Qu et al. (2012),Patkar et al. (2024). Fig. 4 shows a

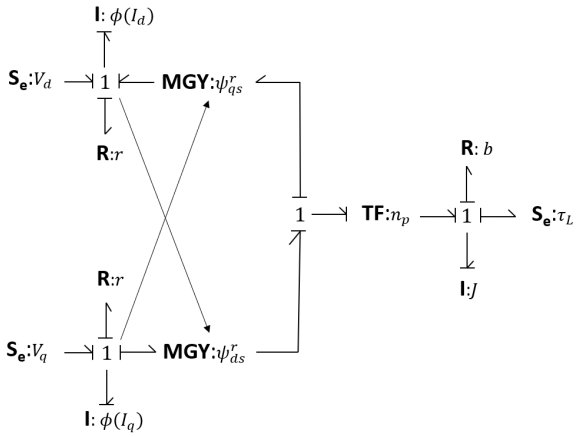


Figure 4: Bond graph of a Permanent Magnet Synchronous Machine (PMSM)

bond graph representation for a three-phase motor in the d, q axis. The constitutive laws for the

magnetic flux in the d, q axis are unknown. All the remaining constitutive relations are assumed to be known. The governing equations in the d, q axis are Pillay and Krishnan (1989):

$$\begin{cases} \dot{\phi}_q &= -R_s I_q - \omega_e \phi_d + V_q, \\ \dot{\phi}_d &= -R_s I_d + \omega_e \phi_q + V_d, \\ \tau_e &= \frac{3}{2} n_p (\phi_d I_q - \phi_q I_d) \\ \dot{\omega}_r &= \frac{\tau_e}{J} \end{cases} \tag{13}$$

where ϕ_q, ϕ_d denote the q, d axis flux linkages, R_s is the stator resistance, I_q, I_d are the q, d axis currents, V_q, V_d are the q, d axis applied voltages, J is the moment of inertia, ω_r is the mechanical angular frequency and ω_e is the electrical angular frequency. We also have $\omega_e = n_p \omega_r$, where n_p is the number of magnet pole pairs. For current control, (13) is rewritten as:

$$\dot{\boldsymbol{I}} = \boldsymbol{L}^{-1} \left(-\boldsymbol{R}\boldsymbol{I} + \boldsymbol{\omega_e}\boldsymbol{\phi} + \boldsymbol{V} \right) \tag{14}$$

Here $\boldsymbol{I} = [I_d, I_q]^T$, $\boldsymbol{R} = \begin{bmatrix} R_s & 0 \\ 0 & R_s \end{bmatrix}$, $\boldsymbol{\omega_e} = \begin{bmatrix} 0 & \omega_e \\ -\omega_e & 0 \end{bmatrix}$, $\boldsymbol{\phi} = [\phi_d, \phi_q]^T$, $\boldsymbol{L}$ is the Jacobian matrix given by $\boldsymbol{L} = \begin{bmatrix} \frac{\partial \phi_q}{\partial I_d} & \frac{\partial \phi_q}{\partial I_q} \\ \frac{\partial \phi_d}{\partial I_d} & \frac{\partial \phi_d}{\partial I_q} \end{bmatrix}$ and $\boldsymbol{V} = [V_d, V_q]^T$.

Nonlinear Flux Model: Model 3 in table 3, models the nonlinear saturation behavior observed in electric motor flux. To generate dataset $D_d = (E, I_d, I_q)$, I_q is regulated at a constant value of 1A using a PI controller applied to V_q while a chirp signal with a starting frequency of 0.1Hz and an ending frequency of 50Hz is used as input excitation in the d axis. The system is excited for 40s. Similarly, dataset $D_q = (E, I_d, I_q)$ is generated by regulating $I_d = 1\text{A}$ and applying a chirp signal in q axis. The details of PySR model are provided in 7.2.4.

SINDYc Setup: To facilitate symbolic learning of non-autonomous dynamical systems, SINDYc allows for a state library and a control library. To draw a fair comparison between the proposed framework and SINDYc, the control library degree is chosen to be 1, as from (13), it is known apriori that the control input will only have degree 1 terms. Since we are attempting to learn (14), we also specify a tensor product of the state and control libraries. The polynomial library with degree 2 is used as the state library and a polynomial library with degree 1 is used as the control library. Higher polynomial degrees in the state library were also tested, but resulted in poorer prediction performance compared to that of the degree 2 state library. The SR3 optimizer with L_0 regularizer was used in training the SINDYc model.

5.3. Results and Discussion

The BGCL framework was successful in recovering the exact expressions for nonlinear spring mass system. For comparing the performance of BGCL against SINDYc, we focus our discussion on the nonlinear single port case.

Table 5 compares the true constitutive laws with those learnt through the BGCL framework for model 3. It is evident that while $\hat{\phi}_q$ has the same functional form as ϕ_q, the same is not true for $\hat{\phi}_d$. From figs. 5(*a*), 5(*b*), it is evident that the learnt function is able to capture the shape and approximate the true constitutive law closely. Using (14) and $\hat{\phi}_d, \hat{\phi}_d$, we arrive at the BGCL learnt model for the motor.

Model	Type	ϕ_q	ϕ_d
1	Linear Single-port	$L_q I_q$	$L_d I_d$
2	Linear Multi-port	$L_q I_q + L_{dq} I_d$	$L_d I_d + L_{dq} I_q$
3	Nonlinear Single-port	$\frac{I_q}{a_1\lvert I_q\rvert + b_1}$	$\frac{I_d}{a_2\lvert I_d\rvert + b_2}$

Table 3: Simulated constitutive laws

Param.	Value	Param.	Value
R_s	1.2 Ω	a_1	8
L_d	20 mH	b_1	65
L_q	7 mH	a_2	4
L_{dq}	1 mH	b_2	30

Table 4: Simulation parameters

ϕ_q	$\hat{\phi}_q$	ϕ_d	$\hat{\phi}_d$
$\frac{I_q}{8\lvert I_q\rvert + 65}$	$\frac{I_q}{8\lvert I_q\rvert + 100}$	$\frac{I_d}{4\lvert I_d\rvert + 30}$	$0.13\tan^{-1}(0.15 I_d)$

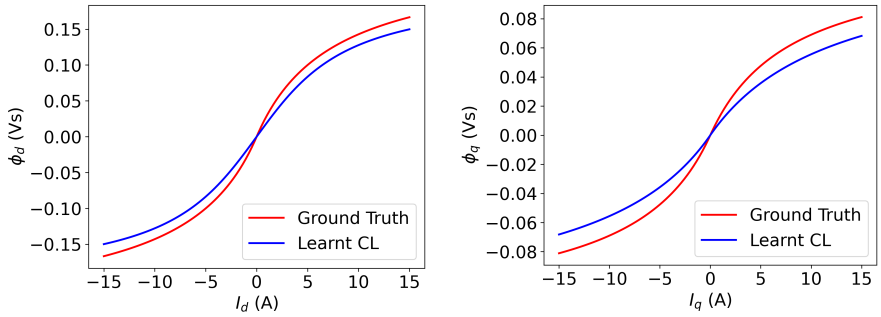
Table 5: Analytical expression comparison of the true and learnt constitutive laws

We compare the step response of the BGCL model and the SINDYc model with the step response of the true model to evaluate their relative performance. A step input excitation of $V_d = 10\text{V}$ and $V_q = 5\text{V}$ is used and the response of the true model, BGCL model and SINDYc model are recorded. It can be seen from figs. 6(*a*) and 6(*b*) that the BGCL prediction emulates the true I_d, I_q current response and is seen to outperform the SINDYc model prediction. The BGCL prediction for the angular speed ω_r is seen to have a steady state error whereas the SINDYc prediction is seen to be more accurate in steady state. However, the BGCL prediction emulates the shape of the true response and thus may be stated to have a better transient response as compared to that of SINDYc.

In addition to the step response comparison, we further test the BGCL and SINDYc model predictions to a sinusoidal excitation sequence given by $V_d = 10\sin(2\pi t)$, $V_q = 5\cos(2\pi t)$ and a sawtooth excitation sequence given by $V_d = 10s(2\pi t + \frac{\pi}{2})$, $V_q = 5s(2\pi t)$, where $s(\cdot)$ is the sawtooth function. From figs. 7(*a*),7(*b*), 7(*c*) we can readily see that while both the BGCL and SINDYc model predictions closely track the true response, the BGCL model prediction is marginal better than that of the SINDYc model. Figs. 8(*a*), 8(*b*) and 8(*c*) draw a comparison between the prediction performance of the BGCL and SINDYc model to sawtooth excitation. From these figs. it is clear that the BGCL model prediction closely tracks the true response and is significantly better than the SINDYc model prediction. By enforcing (18) and learning the constitutive law, the proposed approach outperforms a purely data-driven approach in SINDYc.

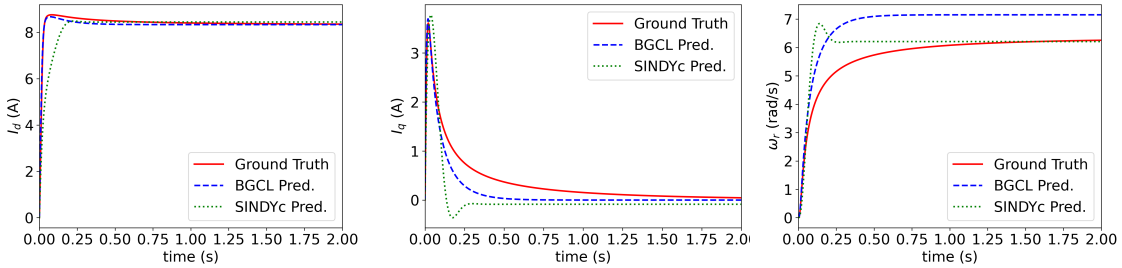
6. Conclusions

In this paper, we proposed the BGCL framework for learning constitutive laws and the governing equations of a dynamical system. The proposed approach was validated through simulation studies performed on a spring mass system and a three phase synchronous motor. Comparison studies with SINDYc showed that by learning expressions for constitutive laws, a better model prediction performance can be achieved. The main limitation of the BGCL approach is the assumption that the constitutive relations of the dissipative elements and transformer elements are assumed to be known. Future work will aim to relax this assumption.



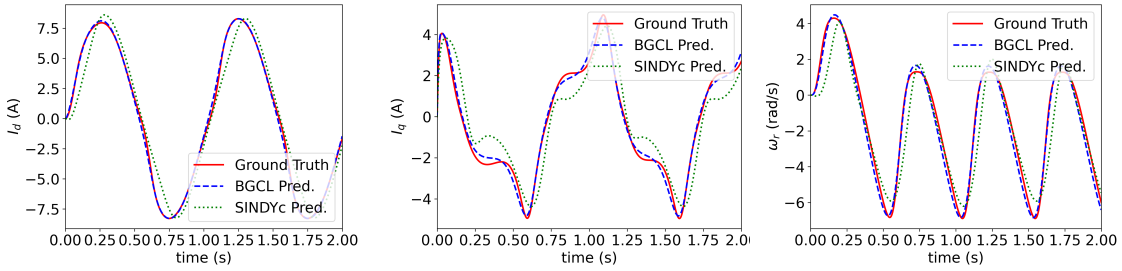
(*a*) d axis flux (*b*) q axis flux

Figure 5: BGCL d, q axis learnt constitutive laws compared to the ground truth.



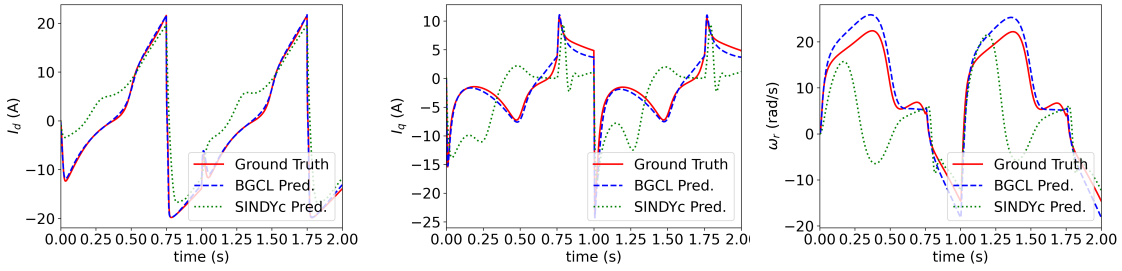
(*a*) I_d prediction (*b*) I_q prediction (*c*) ω_r prediction

Figure 6: Step response of BGCL, SINDYc compared to the true response.



(*a*) I_d prediction (*b*) I_q prediction (*c*) ω_r prediction

Figure 7: Prediction comparison of BGCL, SINDYc to ground truth with a sinusoidal excitation input.



(*a*) I_d prediction (*b*) I_q prediction (*c*) ω_r prediction

Figure 8: Prediction comparison of BGCL, SINDYc to ground truth with a sawtooth excitation input.

7. Appendix

7.1. Constitutive Law derivation

To extract the constitutive laws, we take partial derivatives w.r.t to the efforts and flows of a particular storage element. Without loss of generality, consider the i^{th} element to be a m-port inductive element. Then $\boldsymbol{f}_i = [f_{i1}, f_{i2}, f_{i3}, ...f_{im}]$. Since the energy storage elements are independent, we have

$$\frac{\partial E}{\partial \boldsymbol{f}_i} = \frac{\partial \left(\phi_i(\boldsymbol{f}_i)^T \boldsymbol{f}_i - \int_0^{\boldsymbol{f}_i} \phi_i(\tilde{\boldsymbol{f}}_i)^T d\tilde{\boldsymbol{f}}_i \right)}{\partial \boldsymbol{f}_i} \tag{15}$$

To simplify eq. (15),

$$\frac{\partial \phi_i(\boldsymbol{f}_i)^T \boldsymbol{f}_i}{\partial \boldsymbol{f}_i} = \boldsymbol{J}(\boldsymbol{f}_i)^T \boldsymbol{f}_i + \boldsymbol{\phi}_i(\boldsymbol{f}_i) \tag{16}$$

where $\boldsymbol{J}(\boldsymbol{f}_i)$ is the Jacobian matrix of $\boldsymbol{\phi}_i(\boldsymbol{f}_i)$.

Since the co-energy is path independent, we have

$$\frac{\partial \int_0^{\boldsymbol{f}_i} \phi_i(\tilde{\boldsymbol{f}}_i)^T d\tilde{\boldsymbol{f}}_i}{\partial \boldsymbol{f}_i} = \phi_i(\boldsymbol{f}_i) \tag{17}$$

Using eq. (16) and eq. (17), we can simplify eq. (15) to

$$\frac{\partial E}{\partial \boldsymbol{f}_i} = \boldsymbol{J}(\boldsymbol{f}_i)^T \boldsymbol{f}_i \tag{18}$$

Eq. (18) gives the general condition for each entry of a constitutive law in the general multiport nonlinear case. Thus if a functional form of E is learnt, one may extract the Jacobian and subsequently the constitutive laws of the energy storage elements.

7.2. Simulation Results

For completeness, the cases for ideal spring mass and linear single and multi-port models for 3-phase motors were also simulated and the results are included here.

7.2.1. IDEAL SPRING MASS

For the ideal spring mass system, the constitutive law $e_1(t) = g(x_1(t)) = kx_1(t)$, where k is the spring constant and $e_1(t)$ represents the effort or restoring force of the spring. The mass constitutive law gives $f_1(t) = x_2(t) = \frac{p(t)}{m}$, where m is the mass and $f_1(t)$ represents the flow or velocity of the mass. k, m are taken to be unknown. As described in 4.1, the system is excited by a step input and the time evolution of the spring force $e_1(t) = g(x_1(t))$, the velocity $f_1(t) = x_2(t)$ are recorded. Using eq. (6), the stored energy $E(t)$ is computed. Using eq. (8), we search for $\hat{a}, \hat{b}$ such that $E(e_1, f_1) = \frac{1}{2}\hat{a}e_1^2 + \frac{1}{2}\hat{b}f_1^2$ is satisfied. Solving (18), we get $k = \frac{1}{\hat{a}}$ and $\hat{b} = m$. In the absence of noise, a variety of simulated $\hat{a}, \hat{b}$ values could be correctly recovered.

7.2.2. NON-LINEAR SPRING MASS SYSTEM

The PySR model setup used for the case of the non-linear spring mass system is described in this appendix subsection. The complexity of expressions is limited to 20. The number of iterations for the model was set to 200. Binary operators $[+, *, -, /]$ are available for the model. Unary operators functions containing [square,cube,abs] are included in the model. Nested constraints penalizing more than one unary operator inside another are applied. Eq. (18) is enforced by requiring the searched expressions to be $E(e_1, f_1) = g_1(e_1) + h_1(f_1)$. This is implemented by defining a custom objective function in the model. Constants are penalized by setting their complexity to 5.

7.2.3. LINEAR FLUX MODELS

Following the procedure outlined in 4.1, models $1, 2$ in table 3 with unknown constants L_d, L_q, L_{dq} as listed in table 4 are excited with a voltage step input $V_d = 10\text{V}$ and $V_q = 5\text{V}$. The training dataset $D = (E, I_d, I_q)$ is then collected. Using (8), the values for L_d, L_q, L_{dq} are correctly learnt.

7.2.4. NON-LINEAR FLUX MODEL

The PySR model setup used for the case of the non-linear flux model for motors is described in this appendix subsection. The complexity of expressions is limited to 13. The number of iterations for the model was set to 500. Binary operators $[+, *, -, /]$ are available for the model. Unary operators functions containing [square,cube,abs] are included in the model. Nested constraints penalizing more than one unary operator inside another are applied. Eq. (18) is enforced by requiring the searched expressions to be factorized. This is implemented by defining a custom objective function in the model.

Acknowledgments

This work was sponsored by Weichai Power Research Institute.

References

Mohamadreza Ahmadi, Ufuk Topcu, and Clarence Rowley. Control-oriented learning of lagrangian and hamiltonian systems. In *2018 Annual American Control Conference (ACC)*, pages 520–525, 2018. doi: 10.23919/ACC.2018.8431726.

Nicola Bedetti, Sandro Calligaro, and Roberto Petrella. Stand-still self-identification of flux characteristics for synchronous reluctance machines using novel saturation approximating function and multiple linear regression. *IEEE Transactions on Industry Applications*, 52(4):3083–3092, 2016. doi: 10.1109/TIA.2016.2535413.

N. Bianchi and S. Bolognani. Magnetic models of saturated interior permanent magnet motors based on finite element analysis. In *Conference Record of 1998 IEEE Industry Applications Conference. Thirty-Third IAS Annual Meeting (Cat. No.98CH36242)*, volume 1, pages 27–34 vol.1, 1998. doi: 10.1109/IAS.1998.732255.

Wolfgang Borutzky. *Bond graph modelling of engineering systems*, volume 103. Springer, 2011.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016a. doi: 10.1073/pnas.1517384113. URL https://www.pnas.org/doi/abs/10.1073/pnas.1517384113.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Sparse identification of nonlinear dynamics with control (sindyc). *IFAC-PapersOnLine*, 49(18):710–715, 2016b. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2016.10.249. URL https://www.sciencedirect.com/science/article/pii/S2405896316318298. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

Miles Cranmer. Interpretable machine learning for science with pysr and symbolicregression.jl, 2023. URL https://arxiv.org/abs/2305.01582.

Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks, 2020.

Alejandro Donaire and Sergio Junco. Derivation of input-state-output port-hamiltonian systems from bond graphs. *Simulation Modelling Practice and Theory*, 17(1):137–151, 2009. ISSN 1569-190X. doi: https://doi.org/10.1016/j.simpat.2008.02.007. URL https://www.sciencedirect.com/science/article/pii/S1569190X08000312. Bond Graph Modelling.

Vincent Duindam, Alessandro Macchelli, Stefano Stramigioli, and Herman Bruyninckx. *Modeling and control of complex physical systems: the port-Hamiltonian approach*. Springer Science & Business Media, 2009.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/26cd8ecadce0d4efd6cc8a8725cbd1f8-Paper.pdf.

Paolo Guglielmi, Michele Pastorelli, and Alfredo Vagati. Cross-saturation effects in IPM motors and related impact on sensorless control. *IEEE Transactions on Industry Applications*, 42(6): 1516–1522, 2006. doi: 10.1109/TIA.2006.882646.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285. PMLR, 2018.

Ludovico Ortombina, Fabio Tinazzi, and Mauro Zigliotto. Magnetic modeling of synchronous reluctance and internal permanent magnet motors using radial basis function networks. *IEEE Transactions on Industrial Electronics*, 65(2):1140–1148, 2018. doi: 10.1109/TIE.2017.2733502.

Abhishek Patkar, Dehong Liu, Yebin Wang, and Kamal Youcef Toumi. Magnetic flux map acquisition using a compressed sensing method. In *2024 IEEE 19th Conference on Industrial Electronics and Applications (ICIEA)*, pages 1–6, 2024. doi: 10.1109/ICIEA61579.2024.10665180.

Henry M Paynter. Analysis and design of engineering systems. *MIT press*, 1961.

Henry M Paynter. An epistemic prehistory of bond graphs. *Bond graphs for engineers*, pages 3–17, 1992.

P. Pillay and R. Krishnan. Modeling, simulation, and analysis of permanent-magnet motor drives. i. the permanent-magnet synchronous motor drive. *IEEE Transactions on Industry Applications*, 25(2):265–273, 1989. doi: 10.1109/28.25541.

Zengcai Qu, Toni Tuovinen, and Marko Hinkkanen. Inclusion of magnetic saturation in dynamic models of synchronous reluctance motors. In *2012 XXth International Conference on Electrical Machines*, pages 994–1000, 2012. doi: 10.1109/ICElMach.2012.6349997.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

R. C. Rosenberg. State-Space Formulation for Bond Graph Models of Multiport Systems. *Journal of Dynamic Systems, Measurement, and Control*, 93(1):35–40, 03 1971. ISSN 0022-0434. doi: 10.1115/1.3426458. URL https://doi.org/10.1115/1.3426458.

RC Rosenberg. Reflections on engineering systems and bond graphs. 1993.

Kisung Seo, Zhun Fan, Jianjun Hu, Erik D. Goodman, and Ronald C. Rosenberg. Toward a unified and automated design methodology for multi-domain dynamic systems using bond graphs

and genetic programming. *Mechatronics*, 13(8):851–885, 2003. ISSN 0957-4158. doi: https://doi.org/10.1016/S0957-4158(03)00006-0. URL https://www.sciencedirect.com/science/article/pii/S0957415803000060. Computational Intelligence in Mechatronic Systems.

Andrew Sosanya and Sam Greydanus. Dissipative hamiltonian neural networks: Learning dissipative and conservative dynamics separately, 2022.

B. Stumberger, G. Stumberger, D. Dolinar, A. Hamler, and M. Trlep. Evaluation of saturation and cross-magnetization effects in interior permanent-magnet synchronous motor. *IEEE Transactions on Industry Applications*, 39(5):1264–1271, 2003. doi: 10.1109/TIA.2003.816538.

Yaofeng Desmond Zhong, Biswadip Dey, and Amit Chakraborty. Symplectic ode-net: Learning hamiltonian dynamics with control, 2020.