2019 American Control Conference (ACC)
Philadelphia, PA, USA, July 10-12, 2019

Learning constitutive equations of physical components with predefined feasibility conditions

Ion Matei, Johan de Kleer, Maksym Zhenirovskyy and Alexander Feldman

Abstract— **Complete models of physical systems enable a plethora of model-based methods in control, diagnosis or prognosis. Proprietary information and system complexity often hinder building such models. We address here the problem of learning physical representations of components in partially known physical systems. These representations need to be feasible: when included in the system model, at minimum the model has to simulate. We propose mathematical models for the component representations and give necessary and sufficient conditions for their feasibility. We demonstrate our approach on a illustrative example where we learn different representations of an unknown resistor component in an electrical circuit.**

I. Introduction

In our previous work [12] we showed how we can learn models of physical systems while simultaneously feasibility discovering constraints on their parameters. In this paper we follow a different avenue: *we propose a set of a-priori constraints on parameters that guarantee feasibility*. Complete physics-based models open the door to a plethora of model-based methods used for diagnosis [1], [5], prognosis [8] or control [7]. Building models is often impeded, in part, by the complexity of the physical phenomena or by the lack of complete system specifications The models considered in this paper are represented as differential algebraic equations (DAEs) of the form

$$0 = \mathbf{F}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{z}; \boldsymbol{w}) \quad (1)$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{z}; \boldsymbol{w}), \quad (2)$$

where $\mathbf{F}$, $\mathbf{h}$ are vector valued continuous maps, $\mathbf{x}$ is the state vector, $\mathbf{z}$ is the vector of algebraic variables, $\mathbf{y}$ is the vector of outputs, and $\boldsymbol{w}$ is the vector of parameters. This is a typical mathematical model for physical systems; systems whose behavior are described by variables attached to their components and relations between them. The relations are induced by parameterized *constitutive equations* and by interconnections between components. Figure 1 shows an example of an acausal electrical system with four components. The constitutive and connection equations are depicted. The parameters of the constitutive equations are usually constrained within some feasibility set. When these constraints are not satisfied, the model may become unsimulatable or unstable (a negative resistor in an RC circuit makes the model unstable). We make three basic assumptions: (i) the system topology is known; (ii) the constitutive equations of a subset of the components are available; (iii) experimental data describing the evolution of a subset of the system variables is available. For the components for which the constitutive equations are missing we propose parameterized mathematical models, and constraints on the parameters that make the models feasible. By *feasible component models* we understand models that when combined with the constitutive equations of other system components, the overall system model can be simulated, that is the solution of the DAE (1) exists over some finite time interval. From a numerical simulation perspective, feasibility translates to being able to solve the DAE (1) and determine the trajectory of $\mathbf{x}(t)$ and $\mathbf{z}(t)$ over some time interval and for some initial conditions. This process requires the Jacobian $\frac{\partial \mathbf{F}}{\partial \boldsymbol{\eta}}(\boldsymbol{\eta}(t))$, with $\boldsymbol{\eta}^T = [\dot{\mathbf{x}}^T, \mathbf{z}^T]$ to be invertible along the trajectory of the system. If this property fails at some point along the trajectory, the numerical simulation will fail.

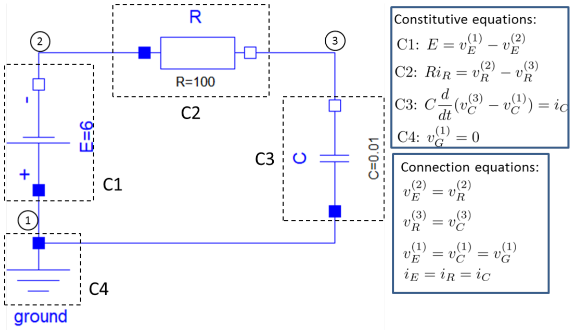


Fig. 1: Example of a component based acausal electric circuit: constitutive and connection equations

For a given (unconstrained) parameterized mathematical model, parameter model learning translates to a standard parameter estimation problem; a problem that has been extensively studied in the literature [3]. In the case where the system parameters are constrained, during the optimization process we must ensure that the parameters remain in the constraint set. If that is not the case, the cost function or gradient evaluation can fail as a result of failed system model simulations. The typical optimization-based formulation of the parameter learning process used in this paper is given by $\min_{\boldsymbol{w}} \sum_{k=0}^{N} \|\mathbf{y}_m(t_k) - \mathbf{y}(t_k; \boldsymbol{w})\|^2$, subject to $\boldsymbol{w} \in \mathcal{W}$, where $\mathcal{W}$ is the parameter feasibility set, $\mathbf{y}_m$ are measurements at a set of time samples $\{t_k\}_{k\geq 0}$, and $\mathbf{y}(t_k; \boldsymbol{w})$ are simulated measurements. Other approaches are possible, such as the Bayesian parameter estimation, where the goal is to compute the con-

Ion Matei, Johan de Kleer, Maksym Zhenirovskyy and Alexander Feldman are with the Palo Alto Research Center (PARC), Palo Alto, CA (emails: `ion.matei@parc.com`, `dekleer@parc.com`, `maksym.zhenirovskyy@parc.com`, `afeldman@parc.com`)

This material is based upon work supported by in part by the Defense Advanced Re-search Projects Agency (DARPA) under Agreements No. HR00111990027 and

978-1-5386-7926-5/$31.00 ©2019 AACC

ditional probability $P(\mathbf{w}|\mathbf{y}(t_0),\ldots,\mathbf{y}(t_N))$, where the feasibility constrained is expressed as $\int_{\mathcal{W}} dP(\mathbf{w}|\mathbf{y}(t_0),\ldots,\mathbf{y}(t_N)) = 1$.

This paper brings two main contributions: (i) *we propose parameterized mathematical models for physical components*, and (ii) *we propose feasibility constraints on the model parameters which translates into characterizing the set $\mathcal{W}$.*

Paper structure: We first introduce representations of physical components (Section II). We continue with the introduction of necessary and sufficient conditions for their feasibility (Section III) that are further specialized in conditions on the component parameters (Section IV). The proposed representations are tested when learning the model for an unknown resistor component in an electrical circuit (Section V).

II. Models for physical components

Models of physical components have connectors (interfaces) through which energy is exchanged with other system components. Connectors are characterized by flow variables (e.g., current, force, torque) and potential like variables (e.g., electric potential, displacement, angle). The flow variables are conserved at connection points, while the potential like variables are equal (generalization of Kirchhoff laws). The constitutive equations define constraints between the connector variables. This formalism is an instance of the more general port-Hamiltonian formalism described in [16].

In what follows we limit ourselves to two connectors components, where the flow variables are denoted by f, and the potential variables are denoted by x. The constitutive equations of the model are a set of equations in terms of the pairs of variables (f_a, x_a) and (f_b, x_b). The same idea can be applied to components with more connectors. In the case of a memoryless component we have $\boldsymbol{G}(f_a, f_b, x_a, x_b; w) = 0$, where $\boldsymbol{G} : \mathbb{R}^4 \to \mathbb{R}^2$ is a vector valued differentiable map that constrains the connector variables, and w is a vector of parameters. Although $\boldsymbol{G}$ defines only 2 equations, it contains 4 variables. The remaining 2 variables are computed from additional equations that are generated when connected to other components. We can model components with memory as well by including derivatives of the component's variables: $\boldsymbol{G}(\boldsymbol{\xi}; w) = 0$, where $\boldsymbol{\xi} = [f_a, f_a^{(1)}, \ldots, f_a^{(m)}, f_b, f_b^{(1)}, \ldots, f_b^{(m)}, x_a, x_a^{(1)}, \ldots, x_a^{(n)}, x_b, \ldots, x_b^{(n)}]$, with $x^{(n)}$ the n^{th} derivative of x. It may appear that we have $2 \times (n+1) + 2 \times (m+1)$ variables and only two equations ($\boldsymbol{G}$ remains a two dimensional vector valued function). For this type of components however, only the largest derivatives are unknown variables, while the lower order derivatives are assumed known from the previous integration step. In what follows we will investigate various choices for $\boldsymbol{G}$ and come up with feasibility conditions for their parameters.

A. Special cases

Typically, in physics-based models there are three types of templates for physical component models:

Type 1: One of the most common behavioral template corresponds to the case where no flow is lost through the components. The constitutive equations are given by $G_1(f_a, f_b) = f_a + f_b = 0$ and $G_2(f_a, \ldots, f_a^{(m)}, x_a - x_b, \ldots, x_a^{(n)} - x_b^{(n)}; w) = 0$. This template covers linear or nonlinear components from multiple domains such as resistors, capacitors, inductors, springs, or dampers.

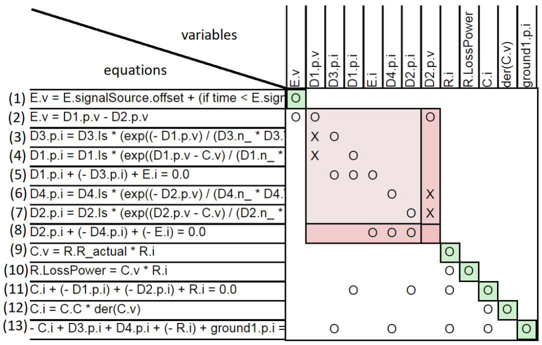
Type 2: There are physical components that suffer flow losses, but do not change the potential variables. Hence the flow at the two connectors will be different. The template for this case can be expressed as $G_1(x_a, x_b) = x_a - x_b = 0$ and $G_2(f_a, \ldots, f_a^{(m)}, f_b, \ldots, f_b^{(m)}, x_a, \ldots, x_a^{(n)}; w) = 0$.

A typical example that corresponds to this template is the mechanical brake whose equations are given by $x_a = x_b$ and $f_a + f_b = g(x_a, \ldots, x_a^{(n)}; w)$, where g is a map that determines the flow loss as a function of x_a and its derivatives, and possibly an external signal. For example, for viscous loss we have $g(x_a^{(1)}, u; w) = u \cdot w \cdot x_a^{(1)}$. Variable u is an exogenous signal that activates/deactivates the brake.

Type 3: There are cases where the flow and potential like equations are completely separate, with corresponding template $G_1(f_a, \ldots, f_a^{(m)}, f_b, \ldots, f_b^{(m)}; w) = 0$ and $G_2(x_a, \ldots, x_a^{(n)}, x_b, \ldots, x_b^{(n)}; w) = 0$. This is the case of ideal transformers, or in particular, ideal gears where $G_1(f_a, f_b) = f_a + w f_b$ and $G_2(x_a, x_b; w) = w x_a - x_b$.

III. Feasibility of component models

In this section we give necessary and sufficient conditions for the feasibility of component models. We consider memoryless type 1-3 component models, although the conditions can be easily generalized for other component types The existence of a DAE solution depends on the invertibility of the system Jacobian along the system trajectory. All modern DAE solvers, before simulating a DAE, transform the DAE into a block lower triangular form (BLT). Figure 2 depicts the BLT form for a rectifier electrical circuit. Each row corresponds to an equation. Each column corresponds to a variable. Variables that belong to diagonal blocks of dimension greater than one (equations 2 through 8) are computed by solving a system of equations. Variables that belong to blocks of dimension one (equation 1, equations 9 through 13) are computed by solving one equation only. If the system of equations is nonlinear, the Newton-Raphson algorithm is used to compute the variables. The BLT form



	variables equations	E.v	D1.p.v	D3.p.i	D1.p.i	E.i	D4.p.i	D2.p.i	D2.p.v	R.i	R.LossPower	C.i	der(C.v)	ground1.p.i
(1)	E.v = E.signalSource.offset + (if time < E.sign	O												
(2)	E.v = D1.p.v - D2.p.v	O	O						O					
(3)	D3.p.i = D3.Is * (exp((- D1.p.v) / (D3.n_ * D3.		X	O										
(4)	D1.p.i = D1.Is * (exp((D1.p.v - C.v) / (D1.n_ *		X		O									
(5)	D1.p.i + (- D3.p.i) + E.i = 0.0			O	O	O								
(6)	D4.p.i = D4.Is * (exp((- D2.p.v) / (D4.n_ * D4.						O		X					
(7)	D2.p.i = D2.Is * (exp((D2.p.v - C.v) / (D2.n_ *							O	X					
(8)	D2.p.i + (- D4.p.i) + (- E.i) = 0.0					O	O	O						
(9)	C.v = R.R_actual * R.i									O				
(10)	R.LossPower = C.v * R.i									O	O			
(11)	C.i + (- D1.p.i) + (- D2.p.i) + R.i = 0.0				O			O		O		O		
(12)	C.i = C.C * der(C.v)											O	O	
(13)	- C.i + D3.p.i + D4.p.i + (- R.i) + ground1.p.i =			O			O			O		O		O

Fig. 2: BLT form for a rectifier circuit

defines a causal relation for computing variables: what variables are needed to computed other variables. For example, to compute the variable on column 10, we first need to compute the variable on column 9, which is computed from equation 9. It also shows that we in fact need to invert lower dimensional matrices to solve for the system variables. Ordinary differential equations (ODEs) can be brought to a diagonal form, that is, each variable is computed from one equation only.

A. *Necessary feasibility conditions*

The simplest scenario for computing the variables that belong to the equations of a component is the case where we compute one single variable from an equation while all other variables that appear in the respective equation have already been computed. This is the case when a variable belongs to a one dimensional block of the BLT form. Since depending on the configuration of the system this is a plausible scenario, conditions that enable such computations are necessary. They are not sufficient though since the variables in the equations can belong to higher dimensional diagonal blocks, when the component is used in a different system configuration. We summarize these conditions in the following proposition.

Proposition 3.1: If the type 1-3 component model is feasible then the following partial derivatives must be invertible along all possible system trajectories: (**type 1**) $\frac{\partial G_2}{\partial f_a}$, $\frac{\partial G_2}{\partial x_a}$, $\frac{\partial G_2}{\partial x_b}$, (**type 2**) $\frac{\partial G_2}{\partial f_a}$, $\frac{\partial G_2}{\partial f_b}$, $\frac{\partial G_2}{\partial x_a}$, (**type 3**) $\frac{\partial G_1}{\partial f_a}$, $\frac{\partial G_1}{\partial f_b}$, $\frac{\partial G_2}{\partial x_a}$, $\frac{\partial G_2}{\partial x_b}$. □
The above proposition states that no matter what variable we need to compute, the Newton-Raphson algorithm will not fail due to a singular Jacobian, as long as the algorithm is used to compute a single variable only. This is not always the case, since the Newton-Raphson algorithm can be applied to solve a system of equations, depending on the system configuration.

B. *Sufficient feasibility conditions*

There is a modeling "artifice" that can transform the above necessary conditions into sufficient conditions as well. We demonstrate this artifice through an example in the electrical domain. Consider the electric circuit with two nonlinear resistors shown in Figure 3. The behavior of the circuit is described by the equations $g_1(i, E-v; w_1) = 0$ and $g_2(i, v; w_2) = 0$, where g_1 and g_2 represent the nonlinear maps relating the current and voltage for each of the two resistors, and w_1 and w_2 are the model parameters that we are trying to learn. Finding the unknowns i and v requires the invertibility of the system Jacobian, which translates to satisfying $\frac{\partial g_1}{\partial i}\frac{\partial g_2}{\partial v} - \frac{\partial g_1}{\partial v}\frac{\partial g_2}{\partial i} \neq 0$. The necessary feasibility conditions introduced above for each component do not guarantee that the determinant is non-zero. For complex systems, the Jacobian becomes a matrix of large dimensions and checking for invertibility becomes even more complicated. We make a "small" change in the model by introducing components with memory at the connectors of the resistor R2 that have a minimal impact on the system behavior. Figure 4 shows the same electrical circuit in which a small capacitor has been added between the two resistors. As the capacitance C_ϵ converges to zero, the behavior of the second circuit converges to the behavior of the original one. The behavior of the circuit is described now by $g_1(i_1, E-v; w_1) = 0$, $C_\epsilon \dot{v} = i_\epsilon$, $i_1 + i_\epsilon = i_2$ and $g_2(i_2, v; w_2) = 0$. Since the state of the capacitor is the potential at the non-grounded connector of the resistor R2, its values at each time instant of the simulation is known, being computed at the previous time instant. This means that v is known and we can compute the currents i_1 and i_2 by solving independently $g_1(i_1, E-v; w_1) = 0$ and $g_2(i_2, v; w_2) = 0$ for the unknowns i_1 and i_2, respectively. Given that the necessary feasibility conditions are satisfied, solution for i_1 and i_2 can be found. Hence, we have shown how the necessary conditions can become sufficient conditions as well, when components are modeled this way.

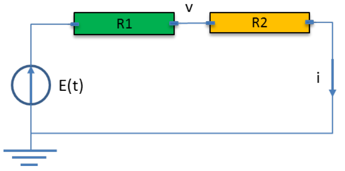


Fig. 3: Electric circuit with two nonlinear resistor

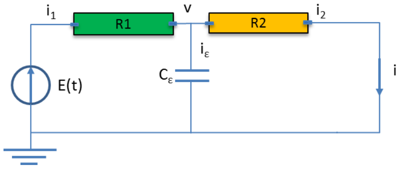


Fig. 4: Electric circuit with two nonlinear resistor and leakage capacitor

An alternative avenue to ensure the feasibility of a component model is to make sure that the component variables satisfy some property that make them "well behaved". Such a property is *dissipativity*. For dissipative components, their internal energy cannot surpass the supplied energy. The dissipative property is significant for two reasons: 1) it is preserved under composition (i.e., composing two dissipative components generates a dissipative component), and 2) systems composed of dissipative components are stable [11]. Formally, this can be expressed as $E(t_1) - E(t_0) = -\int_{t_0}^{t_1} p(\tau)d\tau \leq 0$, for any t_1, where $E(t)$ is the component energy and $p(t)$ represents the component's instantaneous power. This property holds if $p(t) = f_a x_a + f_b x_b \geq 0$ for all $t \geq 0$, for example. The following statement summarizes the above idea.

Proposition 3.2: If the component model is such that the component is dissipative, then the model is feasible. □
The intuition of the previous proposition is straightforward: adding a dissipative component to a model that contains physical models of components does not affect the stability of the overall model. In case the real component is not actually dissipative, adding such model may not make sense though.

IV. Feasibility constraints in terms of model parameters

In the previous section we discussed necessary and sufficient conditions for the component feasibility. In this section we show how these conditions translate to conditions on parameters for particular choices of mathematical models.

As in the previous section we specialize the parameter conditions for Type 1-3 component models. In the case of Type 1-2 models, we need to find a map $g : \mathbb{R}^3 \to \mathbb{R}$ such that $g(x,y,z;\boldsymbol{w}) = 0$, where x, y, z are three variables. In the case of Type 3 components, we have two separate maps, each map depending on two variables only. The necessary conditions for feasibility are: $\frac{\partial g}{\partial x}$, $\frac{\partial g}{\partial y}$ and $\frac{\partial g}{\partial z}$ need to be invertible for all feasible values of x, y and z, and the choice of parameter $\boldsymbol{w}$.

We can narrow down the choices for g, by making it separable:

$$g(x,y,z;\boldsymbol{w}) = g_x(x;w^x) + g_y(y;w^y) + g_z(z;w^z), \qquad (3)$$

which makes the partial derivatives depend only on the variables on which the derivative is taken and on the associated parameters. For example $\frac{\partial g}{\partial x}(x,y,z;\boldsymbol{w}) = \frac{\partial g_x}{\partial x}(x;w^x)$. Even with this simplification, there are many choices for the maps g_x, g_y and g_z. We discuss two such choices and analyze how the necessary feasibility conditions are specialized for them. Let

$$g_l(l;w^l) = \sum_{i=1}^{N} w_i^l l^{2i-1}, \; l \in \{x,y,z\} \qquad (4)$$

resulting in $\frac{\partial g_l}{\partial l}(l;w^l) = \sum_{i=1}^N (2i-1) w_i^l l^{2(i-1)}$. Since all the terms depending on l are positive for any value of l, the invertibility of $\frac{\partial g_l}{\partial l}$ is decided by the parameters w^l. It suffice to impose the condition that all entries of w^l have the same sign to ensure that $\frac{\partial g_l}{\partial l}$ is invertible for all values of l. A sufficient condition that ensures $\left[\frac{\partial g_l}{\partial l}\right]^2$, for $l \in \{x,y,z\}$ to be strictly positive is that the entries of each w^x, w^y and w^z have the same sign, with the first entry being non-zero. We summarize this result in the following proposition.

Proposition 4.1: Let the map g be as in (3)-(4). If the entries of each of the parameter vectors w^x, w^y and w^z have the same sign, with the first entry of each nonzero:

$$w_i^l w_{i+1}^l \geq 0, \; w_1^l \neq 0, \; i \in \{1,\ldots,N-1\}, \; l \in \{x,y,z\}. \qquad (5)$$

then the necessary feasibility conditions are satisfied. □

We can constrain even more the parameters of the polynomial model if we impose a disipativity constraint for the component. By additionally specializing the form of g we can obtain easy to verify feasibility conditions. In the case of Type 1 models let $g(x,y,z;\boldsymbol{w}) = \sum_{i=1}^N w_i^x x^{2i-1} + \sum_{i=1}^N w_i^y (y-z)^{2i-1}$. As before we impose the constraint that all entries of each of w^x, w^y have the same sign. If additionally we impose w^x and w^y to have opposite signs then x and $y-z$ must have the same sign for $g(x,y,z;\boldsymbol{w}) = 0$ to have a solution. We summarize this result in the following statement.

Proposition 4.2: Let $g(x,y,z;\boldsymbol{w}) = \sum_{i=1}^N w_i^x x^{2i-1} + \sum_{i=1}^N w_i^y (y-z)^{2i-1}$ be the second equation for the Type 1 model, with $x = f_a$, $y = x_a$ and $z = x_b$. If $w_i^x \geq 0$, $w_1^x > 0$, $w_i^y \leq 0$ and $w_1^y < 0$, for $i = 2,\ldots,N$ then the component is dissipative. □

We can readily extend this to Type 2 models.

Proposition 4.3: Let $g(x,y,z;\boldsymbol{w}) = \sum_{i=1}^N w_i^x (x+y)^{2i-1} + \sum_{i=1}^N w_i^z z^{2i-1}$ be the second equation for the Type 2 model, with $x = f_a$, $y = f_b$ and $z = x_a$. If $w_i^x \geq 0$, $w_1^x > 0$, $w_i^z \leq 0$ and $w_1^z < 0$, for $i = 2,\ldots,N$ then the component is dissipative. □

Both propositions can be proven by a contradiction argument. They are stronger results since they provide *sufficient* conditions for the dissipativity property to be satisfied. That is, as long as the parameters satisfy the given conditions we are guaranteed that the component is dissipative. Similar results as in Propositions 4.2 and 4.3 can be determined for the original version of the map g introduced in (3) and (4), if we impose $w^y = -w^z$ in the case of Type 1 model, and $w^x = w^y$ in the case of Type 2 model. Our choice of representation for g comes from legacy reasons, since $y-z$ has the meaning of a potential difference. In the case of Type 3 model, a simple dissipative condition is not immediate, except for the linear case. If we assume $G_1(f_a,f_b) = w_1 f_a + f_b$ and $G_2(x_a,x_b) = w_2 x_a - x_b$, then the dissipative conditions becomes $f_a x_a + f_b x_b = f_a x_a(1 - w_1 w_2) \geq 0$ which is true irrespective of the values of f_a and x_a provided $w_1 = \frac{1}{w_2}$. But this is the case of the ideal transformer. For more general cases, we can build a set of constraints $f_a(t_k)x_a(t_k) + f_b(t_k)x_b(t_k) \geq 0$ where $\{t_k\}_{k\geq 0}$ are samples of the simulation time, and add them to the optimization problem for learning the component parameters.

The polynomial form for g is one choice among many. Recalling that a neural network (NN) is a universal approximator [4], we can choose g to be modeled as a NN. In the standard case, training a NN is equivalent to learning a set of parameters for an input-output map. In our case, we do not learn a map, but rather a constraint equation, and we need to impose additional conditions on the parameters of the NN to ensure feasibility. Consider the m layers NN described by

$$\mathbf{z}^{[i]} = \mathbf{W}^{[i]}\mathbf{x}^{[i]} + \mathbf{b}^{[i]} \qquad (6)$$

$$\mathbf{x}^{[i+1]} = \sigma\left(\mathbf{z}^{[i]}\right), \qquad (7)$$

where $\mathbf{W}^{[1]} = [w_x^{[1]}, w_y^{[1]}, w_z^{[1]}]$, $\mathbf{x}^{[1]} = [x,y,z]^T$ and $\mathbf{z}^{[m]} = g(x,y,x) = 0$, with a sigmoid function implementing the nonlinearity, and having a linear output layer. The partial derivative $\frac{\partial g}{\partial x}$ is given by $\frac{\partial g}{\partial x} = \mathbf{W}^{[m]}\mathbf{D}^{[m-1]}\mathbf{W}^{[m-1]}\ldots, \mathbf{D}^{[1]}w_x^{[1]}$, where $\mathbf{D}^{[i]} = \sigma\left(\mathbf{z}^{[i]}\right)\left(1 - \sigma\left(\mathbf{z}^{[i]}\right)\right)$ are diagonal matrices. The following result introduces a sufficient condition that guarantees that $\frac{\partial g}{\partial x}$ is non-zero for all values of its argument.

Proposition 4.4: Let the map introduced in (6)-(7) define the behavior of the second equation of Type 1-2 models. If all product terms in the sums representing the entries of $\mathbf{W}^{[m]}\ldots\mathbf{W}^{[2]}w_l^{[1]}$, $\forall l \in \{x,y,z\}$ have the same sign, and at least one of them is non zero for each $l \in \{x,y,z\}$, then the necessary feasibility conditions are satisfied. □

We give the intuition of the above proposition through an example. Consider a one layer NN described by $g(x,y,z;\boldsymbol{w}) = w_2^T\sigma(\mathbf{z}) + b_2$, $\mathbf{z} = [w_1^x, w_1^y, w_1^z][x,y,z]^T + b_1$. In this setup, $\mathbf{z} \in \mathbb{R}^L$, w_1^x, w_1^y, w_1^z, w_2, b_1 are vectors of size L while b_2 is a scalar, and $\boldsymbol{w} = [w_1^{x\,T}, w_1^{y\,T}, w_1^{z\,T}, b_1^T, w_2^T, b_2]$. The partial derivative of g_x with respect to x is $\frac{\partial g}{\partial x}(x;\boldsymbol{w}) = w_2^T \mathrm{diag}\{\sigma(\mathbf{z}_i)(1-\sigma(\mathbf{z}_i))\}w_1^x$, where $\mathrm{diag}\{\sigma(\mathbf{z}_i)(1-\sigma(\mathbf{z}_i))\}$ is a diagonal matrix where the i^{th} diagonal entry is $\sigma(\mathbf{z}_i)(1-\sigma(\mathbf{z}_i))$, which is the derivative of the sigmoid function. The partial derivative can be explicitly written as $\frac{\partial g}{\partial x}(x;\boldsymbol{w}) = \sum_{i=1}^{L} \sigma(\mathbf{z}_i)(1-\sigma(\mathbf{z}_i))w_{1,i}^x w_{2,i}$, which is non-zero provided all terms $w_{1,i}^x w_{2,i}$ have the same sign and at least one of them is non-zero. The same idea can be applied to the multi-layer case. The same idea can be repeated for Type 2 and 3 models to come up with conditions on model parameters that enforce the necessary conditions. Coming up with parameter constraints that enforce dissipativity for NN models is not straightforward. One option is to impose explicit constraints on the variable trajectories that ensure dissipativity, that is, $f_a(t_k)x_a(t_k) + f_b(t_k)x_b(t_k) \geq 0$, for $k \geq 0$.

V. Learning feasible components: illustrative example

In the previous section we discussed mathematical models for two connectors physical components and their feasibility conditions. In this section we discuss strategies for learning the component's parameters and give an illustrative example. We recall that the problem we are trying to solve is learning the parameters of unknown physical components under feasibility constraints. Unlike standard ML problems, we do not necessarily measure the input and output of a map. Rather we have indirect information about the behavior of a component through a set of measurements; measurements that may be taken at other components. We assume the measurements do contain information about the behavior of the component. In other words, the behavior of the component is inferrable from the available measurements. In effect we are dealing with a parameter estimation problem with feasibility constraints. Parameter estimation problems can be addressed by filtering methods by extending the state vector to include the parameter vector (Kalman filter [9], particle filter [2]). The feasibility constraint must also be integrated into the filter, which may add complications especially in the case of the Kalman filter. Alternatively, we can use an optimization-based approach that allow for an easier integration of the feasibility constraints.

Inequality constraints on component parameters can be eliminated through variable transformations. This enable the use of optimization algorithms for unconstrained problems. We can use both gradient-based and gradient-free optimization algorithms. Gradient based-algorithms have the additional challenge of gradient evaluations. Approximations are an option but they incur the risk of error propagation. Alternatively, we can use automatic differentiation, a feature commonly present in deep learning platforms such TensorFlow [6] or Pytorch [13]. The feature can accommodate ODE representations of the system dynamics only. Due to the quadratic nature of the cost function, quasi-Newton methods (Levenberg-Marquardt algorithm [10] or trust-region-reflective algorithm [17]) are appropriate. For components with a relatively small number of parameters, gradient-free algorithms (e.g., Powell, Nelder-Mead [14], [15]) can be used. In both type of algorithms, the bulk of the numerical effort is generated by the numerical simulations of the system dynamics needed to evaluate the cost function or the gradients of the cost function.

To demonstrate our approach, we consider the Cauer circuit shown in Figure 5, where the objective is to model the unknown resistor `resistor`. The approach can be used for multiple unknown components. The circuit is powered by a 6V source (`E1`) and the output measurements is the voltage across the resistor `R4`. The parameter values for the circuit components are shown on the circuit diagram. For example, we assume that the unknown `resistor` is in fact a linear 2Ω resistor. We evaluate different models for the unknown resistor and learn their parameters. We start with a dissipative model for the resistor: $w_1 i + w_2 i^3 + w_3 i^5 = w_4 v + w_5 v^3 + w_6 v^5$, where i is the current through the resistor, and v is the potential difference across the resistor, with $w_j > 0$ for $j = 1 \ldots 6$. Without loss of generality we can set $w_6 = 1$. Since we have a small number of parameters, we use a gradient-free optimization algorithm (Powell), avoiding the need to approximate gradients. The inequality constraints are eliminated through variable transformations: $w_j = |\tilde{w}_j|$, $j = 1 \ldots, 5$. The optimization result is shown in Table I, and the voltage-current map is depicted in Figure 6. Although, not perfectly a linear map (in particular around zero), is a reasonable approximation for a linear resistor.

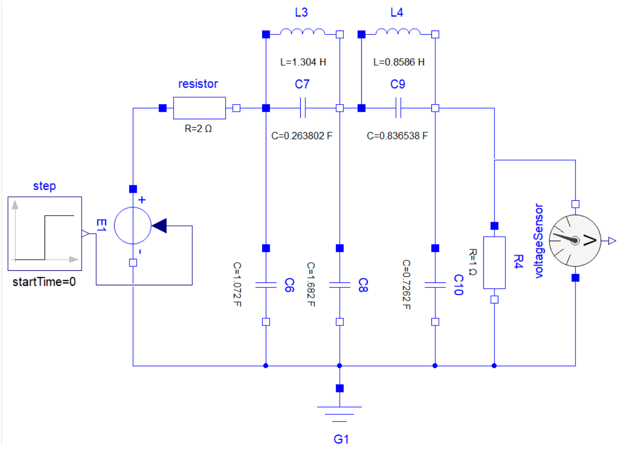


Fig. 5: Cauer circuit

w_1	w_2	w_3	w_4	w_5
541.292	69.263	24.775	342.897	0.5148

TABLE I: Polynomial model parameter values

We consider next a NN like model for the unknown resistor, given by the equations $z_1 = w_1 v + w_2 i + b_1$, $z_2 = w_3 v + w_4 i + b_2$ and $0 = w_5 \tanh(z_1) + w_6 \tanh(z_2) + b_3$, where $\tanh(z) = (e^z - e^{-z})/(e^z + e^{-z})$. The model shown above is a NN with one hidden layer, with the activation function given by the "tanh" function. The resistor model can be compactly represented as $g(i,v) = 0$, where $g(i,v) = w_5 \tanh(w_1 v + w_2 i + b_1) + w_6 \tanh(w_3 v + w_4 i + b_2) + b_3$. The

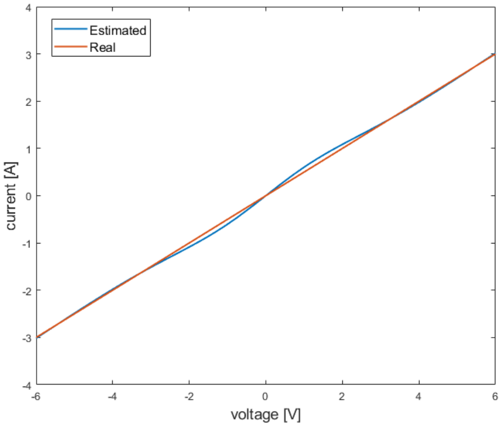


Fig. 6: Polynomial model: voltage-current map

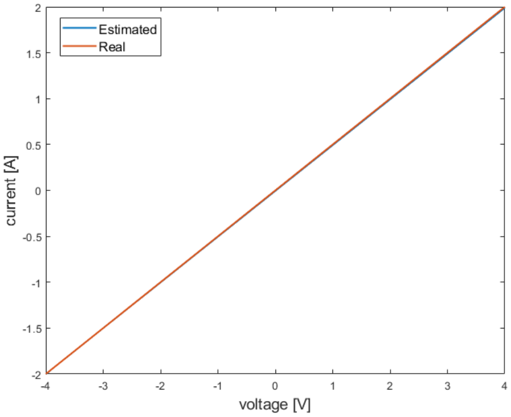


Fig. 7: NN model: voltage-current map

partial derivatives of g with respect to i and v are $\frac{\partial g}{\partial i} = w_5 w_2 \left[1 - \tanh(z_1)^2\right] + w_6 w_4 \left[1 - \tanh(z_2)^2\right]$ and $\frac{\partial g}{\partial v} = w_5 w_1 \left[1 - \tanh(z_1)\right]^2 + w_6 w_3 \left[1 - \tanh(z_2)^2\right]$. If we impose the conditions $w_1 > 0$, $w_3 > 0$, $w_2 < 0$, $w_4 < 0$ and $w_5 > 0$, $w_6 > 0$, and recalling that $1 - \tanh(z)^2 > 0$ for all z, we can conclude that $\frac{\partial g}{\partial i} \neq 0$ and $\frac{\partial g}{\partial v} \neq 0$ for all possible values of i and v. In other words, if we use a Newton-Raphson algorithm to compute i (when v is known), or v (when i is known), the algorithm will not fail due to a singular Jacobian. We used the Powell gradient-free optimization algorithm, where the constraints were eliminated through variable transformations: $w_1 = |\tilde{w}_1|$, $w_2 = -|\tilde{w}_2|$, $w_3 = |\tilde{w}_3|$, $w_4 = -|\tilde{w}_4|$, $w_5 = |\tilde{w}_5|$, and $w_6 = |\tilde{w}_6|$. The optimization results are shown in Tables II. Figure 7 shows that the estimated voltage-current map using the NN model follows closely the "real" voltage-current map of the linear resistor.

w_1	w_2	w_3	w_4	w_5	w_6
7.73e-08	-1.6712	1.324	-0.9843	0.999	1.00
b_1	b_2	b_3			
0.0232	-0.0362	1.444e-06			

TABLE II: NN model parameter values

VI. Conclusions

We addressed the problem of learning representations of physical components for partially known models of physical systems. We proposed different types of representations and mathematical models for them. We introduced necessary and sufficient conditions for their feasibility; conditions that were further specialized in conditions in terms of model parameters. Satisfying these conditions ensure successful model simulations during the parameter search process. We demonstrated our approach in the case of learning the model for a resistor in an electrical circuit. As future steps, we will develop training algorithms that use automatic differentiation to compute the gradients of the cost function, and are able to deal with DAE representations of the system dynamics.

References

[1] H. Alwi, C. Edwards, and C.P. Tan. *Fault Detection and Fault-Tolerant Control Using Sliding Modes*. Springer, London, 1974 (ISBN: 0-12-078250-2).

[2] M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, 50:174–188, 2002.

[3] Y. Bard. *Nonlinear Parameter Estimation*. Academic, New York, 2011 (ISBN: 978-0-85729-649-8).

[4] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

[5] Johan de Kleer and Brian C Williams. Diagnosing multiple faults. *Artificial intelligence*, 32(1):97–130, 1987.

[6] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[7] Carlos E. Garcia, David M. Prett, and Manfred Morari. Model predictive control: Theory and practice - A survey. *Automatica*, 25(3):335 – 348, 1989.

[8] Andrew K.S. Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483 – 1510, 2006.

[9] R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[10] Christian Kanzow, Nobuo Yamashita, and Masao Fukushima. Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *Journal of Computational and Applied Mathematics*, 172(2):375 – 397, 2004.

[11] H.K. Khalil. *Nonlinear Systems*. Pearson Education. Prentice Hall, 2002.

[12] I. Matei, J. de Kleer, and R. Minhas. Learning constitutive equations of physical components with constraints discovery. In *2018 Annual American Control Conference (ACC)*, pages 4819–4824, June 2018.

[13] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[14] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[15] M.J.D. Powell. A view of algorithms for optimization without derivatives. Technical report, University of Cambridge, UK, May 2007.

[16] Arjan van der Schaft. Port-hamiltonian systems: an introductory survey. In *Proceedings of the International Congress of Mathematicians Vol. III*, number suppl 2, pages 1339–1365. European Mathematical Society Publishing House (EMS Ph), 2006.

[17] Ya-xiang Yuan. Recent advances in trust region algorithms. *Mathematical Programming*, 151(1):249–281, Jun 2015.