Journal of Computational Physics 318 (2016) 22–35

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Machine learning strategies for systems with invariance properties

Julia Ling *, Reese Jones, Jeremy Templeton

Sandia National Laboratories, 7011 East Avenue, Livermore, CA, 94550, United States

ARTICLE INFO

Article history:
Received 9 September 2015
Received in revised form 19 January 2016
Accepted 2 May 2016
Available online 6 May 2016

Keywords:
Machine learning
Turbulence models
Constitutive models
Tensor invariants

ABSTRACT

In many scientific fields, empirical models are employed to facilitate computational simulations of engineering systems. For example, in fluid mechanics, empirical Reynolds stress closures enable computationally-efficient Reynolds Averaged Navier Stokes simulations. Likewise, in solid mechanics, constitutive relations between the stress and strain in a material are required in deformation analysis. Traditional methods for developing and tuning empirical models usually combine physical intuition with simple regression techniques on limited data sets. The rise of high performance computing has led to a growing availability of high fidelity simulation data. These data open up the possibility of using machine learning algorithms, such as random forests or neural networks, to develop more accurate and general empirical models. A key question when using data-driven algorithms to develop these empirical models is how domain knowledge should be incorporated into the machine learning process. This paper will specifically address physical systems that possess symmetry or invariance properties. Two different methods for teaching a machine learning model an invariance property are compared. In the first method, a basis of invariant inputs is constructed, and the machine learning model is trained upon this basis, thereby embedding the invariance into the model. In the second method, the algorithm is trained on multiple transformations of the raw input data until the model learns invariance to that transformation. Results are discussed for two case studies: one in turbulence modeling and one in crystal elasticity. It is shown that in both cases embedding the invariance property into the input features yields higher performance at significantly reduced computational training costs.

1. Introduction

In many scientific domains, macroscopic phenomena are governed by microscopic interactions. In order to enable efficient simulations of such systems, empirical models are often employed to approximate the effect of the microscopic interactions. For example, in turbulence modeling, Reynolds Averaged Navier Stokes (RANS) simulations use empirical models for the Reynolds stresses to approximate the effect of turbulent transport on the mean momentum field. These simulations provide significant savings in computational cost in comparison to Direct Numerical Simulations (DNS), which directly resolve the turbulent fluctuations all the way down to the smallest scales. Empirical models are also used in solid mechanics to relate stress to deformation that is potentially complex at the atomic scale, and in thermodynamic relations for non-ideal gases,

* Corresponding author.
E-mail address: jling@sandia.gov (J. Ling).

http://dx.doi.org/10.1016/j.jcp.2016.05.003
0021-9991/

and in electromagnetism to define material properties such as conductivity. These constitutive models allow continuum methods to be used in lieu of direct simulation of the atomic interactions.

Traditionally, these models have been developed through a combination of theoretical knowledge and calibration to a limited set of data from simplified test cases. For example, the widely-used Spalart–Allmaras RANS turbulence model was initially tuned to reproduce experimental results for two-dimensional mixing layers and wakes [1]. Likewise, the wall functions commonly employed to model the effects of walls on turbulent flows were developed based on results for a flat plate boundary layer [2]. When defining the constitutive relations for hyper-elastic materials, Ogden [3] used physical arguments to determine the key terms, then fit the coefficients to experimental data. Valanis and Landel [4] used a similar methodology for forming their constitutive relation for hyper-elastic materials. These examples are typical of data-limited regimes, where simple models with only a few free parameters are desirable because they can avoid over-fitting when tuned to the limited amount of available experimental data.

More recently, the rise of high performance computing has led to the availability of large high-fidelity data sets in increasingly complex configurations. In fluid mechanics, Direct Numerical Simulations (DNS) resolve all scales of the time-dependent flow. In solid mechanics, Molecular Dynamics (MD) simulations resolve complex material deformation with atomic resolution. While it is not yet computationally feasible to run these high fidelity simulations for many full-scale applications, these simulations output huge quantities of data that can be mined to develop and improve empirical models. Furthermore, advances in experimental techniques, such as digital volume correlation [5,6], tomographic particle image velocimetry [7], and magnetic resonance velocimetry [8], are yielding experimental data sets of unprecedented size and detail.

There have already been several efforts to use machine learning algorithms to leverage these large high-fidelity data sets to construct more accurate model closures. Milano and Koumoutsakos [9] used DNS results for a turbulent channel flow to train a neural network to reconstruct the near wall flow. Tracey et al. [10,11] used machine learning algorithms to model the Reynolds stress anisotropy, and also to mimic the source terms from the Spalart–Allmaras turbulence model. Duraisamy et al. [12] used neural networks and Gaussian processes to model intermittency in transitional turbulence, and Zhang et al. [13] used these algorithms to model turbulence production in channel flow. Data driven models have also been used to calibrate models for sub-surface flows [14]. Ling and Templeton [15] used Random Forests to predict regions of high model form uncertainty in RANS results.

Data-driven methods have also been adopted to develop advanced constitutive models in solid mechanics. Ghaboussi [16,17] proposed the use of neural networks for constitutive modeling of trusses and fiber-reinforced composites. Subsequently, neural networks have been used to model the stress–strain relation in a variety of materials, including sand [18], viscoplastic materials [19], metal alloys [20], and concrete [21]. Feng and Yang [22] also used data-driven genetic evolution methods to derive a constitutive relation for composite materials. Shen et al. [23] and Liang et al. [24] implemented neural networks to model the strain energy function for hyper-elastic materials and elastomeric foams, respectively. Clearly, data-driven methods are gaining popularity in the context of solid mechanics as well.

One of the key questions when applying data-driven techniques to physical systems is to what extent domain knowledge can and should be built into the algorithms. One school of thought in the machine learning community is that very little domain knowledge needs to be injected into the algorithm, and that given sufficient data, the algorithm should be capable of discerning patterns and structure on its own [25,26]. One drawback of embedding domain knowledge into the machine learning model is that if that domain knowledge proves faulty, it can lead to diminished performance in the machine-learned model. For example, Piatetsky [25] points out that an algorithm determining potential truck routes could use domain knowledge to rule out any routes that involve driving over water; while this rule seems reasonable and could reduce computational cost, it might be inapplicable in the winter when frozen lakes would permit truck passage. Grzes and Kudenko [27] showed that building erroneous domain knowledge into their reinforcement learning model led to significant deterioration in model performance. In many physical systems, however, definitive and exact domain knowledge does exist in the form of known invariances and symmetries.

For example, the classical laws of motion are known to obey Galilean invariance, which states that these laws do not change in different inertial frames of reference. In fact, the laws of motion are intricately linked to invariance properties. Noether's theorem [28,29] describes how symmetry properties represent fundamental constraints that can be used to derive the laws of motion, underlining how central these properties are in physical systems. Given the importance of these symmetry properties, it would be desirable, both for model accuracy and physical realizability, for any machine learning model of a system with an invariance property to respect those invariances. In the context of fluid mechanics, Galilean invariance dictates that any scalar flow variable, such as pressure or velocity magnitude, will be invariant to rotations, reflections, or translations of the frame of reference. Likewise, the same principle dictates that solid deformation response will depend only on stretches, not on local rotations. Similarly, many crystalline materials exhibit material symmetry properties, such that scalar functions like the internal strain energy will be invariant to rotations within the symmetry group associated with the crystal. A function is an invariant under a given transformation if its value does not change when its inputs are subjected to that transformation. A more thorough discussion of invariants is given in Sec. 2.1.

While there has been increasing interest in applying data-driven methods to closure models for physical systems, there is no real consensus in the literature on how the input feature sets for these models should be chosen, and whether these input features should respect the invariance properties of the system. Whereas in traditional closure modeling, invariance properties are typically enforced explicitly, machine learning closure models have the option of using non-invariant inputs