Blind Deconvolution of Turbulent Flows Using Neural Networks

Cory Mosiman, Rahul Chowdhury, Olga Doronina, Umakant Padhy, Rajarshi Basak

Our Proposal

Our work is based on Maulik and San's paper published in 2017, which applies a 'single-layer feed-forward artificial neural network architecture trained through a supervised learning approach for the deconvolution of flow variables from their coarse-grained computations.'[3] The application of deconvolution to fluid flows is similar to image deblurring: given a blurry (low resolution photo), how can you increase the sharpness (remove blurring artifacts)? Deconvolution has been very successful in its application to images, however, the focus of the aforementioned work is applying blind deconvolution to turbulent flows. For computational feasibility turbulent flow simulations often resolve large scales on a coarse grid while modelling sub-grid stresses (Large-Eddy Simulation). The aim of this ANN is to recover subfilter-scale structure of the flow (deconvolution) without knowing the filter function applied (blind). Sample results from the study are below. We aim to use the John Hopkins Turbulence Database (JHTDB) for our studies.

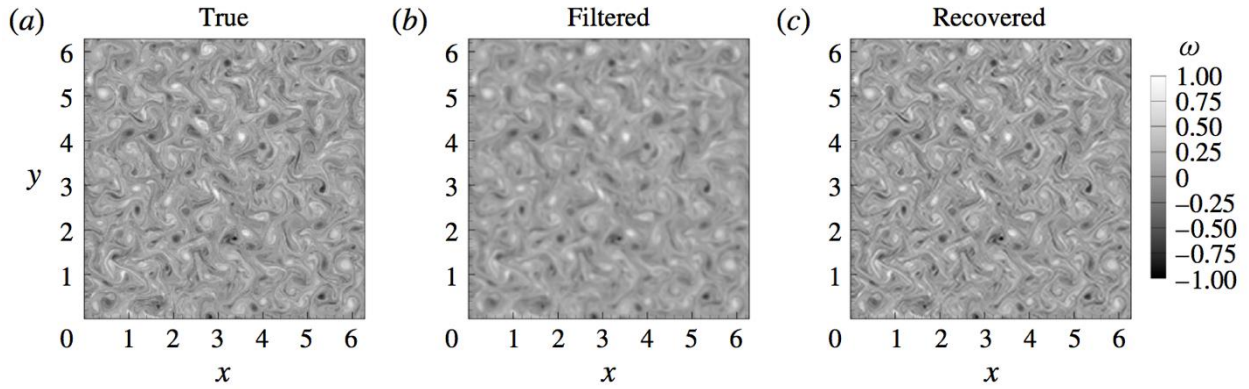

Figure 1 Sample results of applying deconvolution to a turbulent fluid flow.

Motivation

Our motivation for studying this problem is to:

a) Learn how to apply neural networks to a real-world dataset
b) Understand both the *convolution* and *deconvolution* operations and how they are used in neural network approaches
c) Understand *why* convolution and deconvolution operators are used and to what applications they are appropriate
d) Expand upon the current solution to compare at least one additional machine learning method to the existing solution
e) Get smarter

Process

Date	**Milestone**
March 11th	Dataset exploration, preprocessing, detailed outline of tasks
March 18th	1. Exploration of deconvolution techniques and applications in Python 2. Complete review of ANN, convolution, deconvolution 3. Discussion of team with course instructor for check of understanding
April 11th	Replication of existing methods, ANN and blind deconvolution
	Other subtasks milestones TBD on March 11th

Introduction

Artificial neural networks attempt to replicate the functionality of the brain's neural system in an abstract manner for decision making. More formally, "The goal of a feedforward network is to approximate some function … [such that] a classifier, $\mathrm{y} = \mathrm{f}^*(x)$ maps an input $\boldsymbol{x}$ to a category y" [2] by learning the weighted parameters θ. In general, this is the overall goal of any classification algorithm, however, artificial neural networks accomplish this by using a 'directed acyclic graph'[2] where the output of each node is a function of multiple input values. *Feedforward* simply implies that inputs provided to the neural network propagate through and produce a single output – there are no feedback connections.

Convolution is "an integral that expresses the amount of overlap of one function (g) as it is shifted over another function (f)."[1] In essence, the convolution operation is often used in order to achieve three concepts: a) sparse interaction, b) parameter sharing, c) equivariant representations.[2] Inputs to the convolution are matrices or tensors. An example of the convolution operation is illustrated below, where x is the input matrix, w is the kernel (parameter values), $s(t)$ is the output or feature map. A noise term, ε, is also included since most physical systems are often affected by some random degradation.

$$s(t) = (x * w)(t) + \varepsilon$$

The goal of *convolution* is to resolve the output, $s(t)$, based on the inputs and transfer function. The goal of *deconvolution* is to recover initial input values knowing the output and kernel function. *Blind deconvolution* refers to the 'estimation of the underlying blur kernel without any knowledge or closed form estimate of its true nature.'[3] For our approach, we will recreate the blind deconvolution approach as our main focus, with the potential of expanding to alternative simulation approaches.

Main Approach

The figure below illustrates the main technique of the paper – calculate the point w* using a set of 'neighboring' points, $\overline{w}$. The study used 100 neurons – we will similarly attempt with 100 neurons initially and then test other neuron counts to evaluate performance.

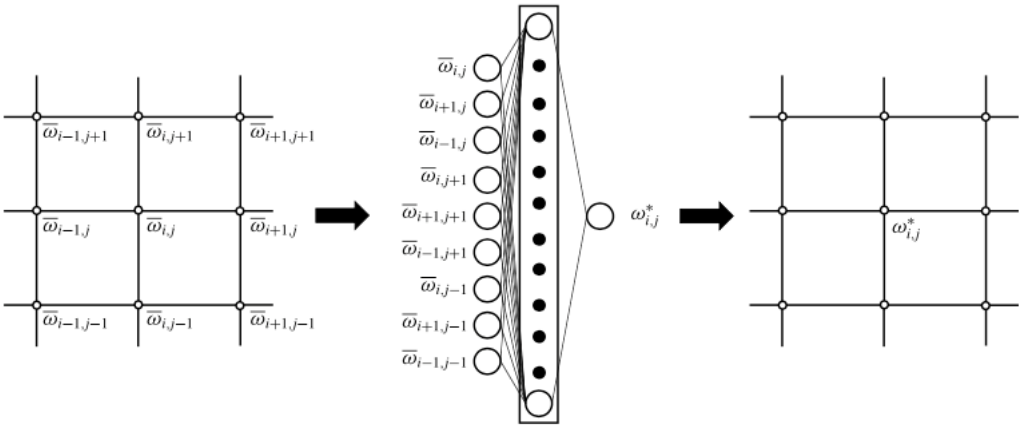

Figure 2 Proposed deconvolution strategy with the aim of recovering w from $\overline{w}$.*

[1] http://mathworld.wolfram.com/Convolution.html

[2] http://www.deeplearningbook.org/

[3] R. Maulik and O. San, "A neural network approach for the blind deconvolution of turbulent flows," *J. Fluid Mech.*, vol. 831, pp. 151–181, 2017