JGR | Machine Learning and Computation

RESEARCH ARTICLE
10.1029/2024JH000223

A Stability Analysis of Neural Networks and Its Application to Tsunami Early Warning

Donsub Rim[1], Sanah Suri[1], Sanghyun Hong[2], Kookjin Lee[3], and Randall J. LeVeque[4]

[1]Department of Mathematics, Washington University in St. Louis, St. Louis, MO, USA, [2]School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA, [3]School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ, USA, [4]Department of Applied Mathematics, University of Washington, Seattle, WA, USA

Abstract Neural networks (NNs) enable precise modeling of complicated geophysical phenomena but can be sensitive to small input changes. In this work, we present a new method for analyzing this instability in NNs. We focus our analysis on adversarial examples, test-time inputs with carefully crafted human-imperceptible perturbations that expose the worst-case instability in a model's predictions. Our stability analysis is based on a low-rank expansion of NNs on a fixed input, and we apply our analysis to a NN model for tsunami early warning which takes geodetic measurements as the input and forecasts tsunami waveforms. The result is an improved description of local stability that explains adversarial examples generated by a standard gradient-based algorithm, and allows the generation of other comparable examples. Our analysis can predict whether noise in the geodetic input will produce an unstable output, and identifies a potential approach to filtering the input that enable more robust forecasting.

Plain Language Summary Deep learning models trained on empirical or simulated data can yield accurate predictions in real time. Consequently, their use has expanded to predicting complex physical phenomena. For example, a convolutional neural network can predict tsunami waveforms accurately using only geodetic measurements of the earthquake, thereby potentially informing early warning systems. However, indiscernible perturbations in the measurement data can incur instabilities in neural networks (NNs). Should the early warning systems rely on such models, the ramifications of unstable predictions are of great concern for safety and security. The cause of these instabilities is largely unknown, and characterizing the destabilizing perturbations and securing a NN against them are important open problems. In this work, we introduce a novel stability analysis of NNs, making progress toward understanding how these instabilities arise, by focusing on the inherent layered structure of NNs. We conduct this analysis on a previously proposed tsunami prediction model, obtaining destabilizing perturbations comparable to those found by state-of-the-art algorithms. Our methodology can potentially be applied to similar models trained on other types geophysical data for prediction, for example, in seismic forecasting models.

Key Points:

- Neural networks (NNs) suffer from input instabilities, making them difficult to deploy in critical applications like natural hazard warning
- We derive a low-rank expansion for NNs, revealing an input-dependent low-rank matrix that describes local input stability
- We illustrate that the instabilities are closely related to the singular vectors derived from the low-rank matrix

Supporting Information:

Supporting Information may be found in the online version of this article.

Correspondence to:

D. Rim,
rim@wustl.edu

Citation:

Rim, D., Suri, S., Hong, S., Lee, K., & LeVeque, R. J. (2024). A stability analysis of neural networks and its application to tsunami early warning. *Journal of Geophysical Research: Machine Learning and Computation*, *1*, e2024JH000223. https://doi.org/10.1029/2024JH000223

Received 8 APR 2024
Accepted 17 NOV 2024

1. Introduction

In recent years, neural networks (NNs) have achieved a remarkable level of performance in making predictions even when they are trained only on empirical observations (Goodfellow et al., 2016; LeCun et al., 2015). Numerous studies have confirmed that these NNs generalize strikingly well in comparison to previous regression models (Hochreiter & Schmidhuber, 1997; Krizhevsky et al., 2017; LeCun et al., 1989; Vaswani et al., 2017). Following such a strong track record, NNs have recently been proposed as tools for a variety of geoscience applications related to hazard assessment or real-time forecasting for warning systems, for example, (Liu et al., 2021; Makinoshima et al., 2021; Mulia et al., 2022; Rim et al., 2022). As one example, in (Rim et al., 2022) a NN was trained to forecast several hours of a tsunami waveform at a specified location based on only a few minutes of seismic Global Navigation Satellite System (GNSS) data acquired in real-time during an earthquake. The training data was generated by running tsunami and seismic simulations for a large set of synthetic earthquakes, and it was demonstrated that a model trained using 1,300 events could be used to generate accurate tsunami forecasts when presented with synthetic GNSS data for a new event. This particular model is used as a test case in Section 3.2, and is discussed further below and in much more detail in (Rim et al., 2022). There are two major issues that are important to address before considering the application of the model to real-world forecasting. The synthetic GNSS data was generated using a layered earth model (Zhu & Rivera, 2002), raising

questions about how well the trained model would work if presented with GNSS data from a real event. Moreover, real GNSS data contains noise and observational errors that may have small relative amplitude but could potentially excite any underlying instabilities in the model. The same concerns arise for many other applications of NNs in geophysics and beyond, where the model must be trained using synthetic data while the real data may contain observation errors or differ in other, possibly minor, ways from the training data.

In this paper we study one core aspect of this question, the stability of forecasts with respect to small changes in the input data. This is an important topic because several recent computational studies have observed that NNs are susceptible to instabilities under small input perturbations (Biggio et al., 2013; Szegedy et al., 2014). Input data that is an imperceptible perturbation of the original input but that leads to significant changes in the forecast are often called "adversarial examples" in the machine learning literature, suggesting that an adversary is attempting to fool an algorithm to gain advantage. But we note that this is just a name used in this context for data and a perturbation that exposes the underlying instability of the machine learning model, and we use it in this manner here. A famous early example from (Szegedy et al., 2014) is a NN trained to classify images that correctly classifies an image of a school bus, but for which a small perturbation in the pixel colors (imperceptible to a human) results in essentially the same image being classified as an ostrich. In this paper we are not concerned with classification problems but rather a regression problem: The forecasting of a tsunami time series based on several input GNSS time series, and an adversarial example would be one for which small changes in the inputs (perhaps at the level of observational noise) result in significant changes to the tsunami forecast (on the scale of meters). Such examples are often found by a gradient-based approach such as the projected gradient descent (PGD) algorithm (Kurakin et al., 2017; Madry et al., 2018) that search over the high dimensional space of possible perturbations in hopes of finding extreme adversarial examples.

A precise description of these instabilities will help make our discussion clear. Suppose f is a NN model that takes a vector x as an input and outputs the vector $y = f(x)$. In plain terms, we say f is *unstable* or has an *instability* at x if a small relative perturbation δx of x causes a much larger perturbation relative δy in the output y. In mathematically precise terms: We say f is unstable at x if there is a perturbation δx that is small in the sense that $\|\delta x\|_X \le \varepsilon \|x\|_X$ for some small ε, but for which the corresponding output perturbation $\delta y = f(x + \delta x) - f(x)$ is much larger in the sense that

$$\frac{\|\delta y\|_Y}{\|y\|_Y} \ge C \frac{\|\delta x\|_X}{\|x\|_X}, \quad \text{for some large constant } C. \tag{1}$$

here the choice of input and output variables, the model f, the constants C, ε, the norm for the input $\|\cdot\|_X$ and the output $\|\cdot\|_Y$, can vary depending on the context, and takes into account relevant scales or measurement units. In this notion of instability, the model f can be either a regression model or a classification model (e.g., an image classifier f that assigns to a pixelated image x an integer y corresponding to a label (Szegedy et al., 2014)). Adversarial examples are particular x and δx that reveal the instability (the name used in particular for instabilities that occur in trained deep learning models). In our tsunami forecasting context, f would be a prediction model for tsunami waveforms, taking as input geodetic signals x (surface displacements as a function of time at multiple sensors) and producing as output the tsunami waveform y (water surface elevation as a function of time at specified locations). In general the tsunami waveform is a stable function of the earthquake and the resulting seismic signal (see e.g., (LeVeque et al., 2016; Williamson et al., 2019)) and 0.5%-level perturbation is well within the signal-to-noise ratio in real measurements (Melgar et al., 2020). So in our tsunami prediction setting, we may specify the constants $\varepsilon = 0.005$, $C = 20$ above to define instabilities, that is, we say the model has an instability if a 0.5%-level perturbation in the GNSS input signal causes more than 10% relative change in the predicted tsunami waveform (since $\epsilon C = 0.1$ in this case). We use the ℓ_2-norm to define the norms of both the input and output time series.

Many analyses and mitigation strategies were proposed to understand and address this instability, to varying degrees of success (Cohen et al., 2019; Ilyas et al., 2019; Madry et al., 2018; Shafahi et al., 2019; Szegedy et al., 2014; Tramér et al., 2018; Zhang et al., 2019). But even for relatively simple models such as convolutional neural networks (CNNs), adversarial examples persist and no satisfactory stabilization method has been found. The root cause of these instabilities has not been identified. Characterizing the destabilizing perturbations and stabilizing a NN against them are important open problems. Currently proposed stabilization techniques often

come at a loss of model utility (Zhang et al., 2019) or at a high computational cost (Cohen et al., 2019). On the other hand, greater data set size and greater test accuracy correlates well with greater accuracy against adversarial examples, according to a broad survey (Miller et al., 2021). This suggests simply scaling up the data set and the model size (Kaplan et al., 2020) might stabilize the NNs eventually, but the scale needed for desired accuracy is infeasible to achieve at the moment. As a result, the development of deep learning methods that produce accurate models that are stable against adversarial examples, that is, models for which adversarial examples cannot be found, still remains an open problem, and to date there has been little mathematical understanding of how to approach this more systematically.

In this work, we introduce a novel stability analysis for feedforward NNs derived directly from the NN architecture. Our analysis is based on a new observation that a feedforward NN f has the expansion

$$f(x) = [F_0 + F_\sigma(x)] \cdot x. \tag{2}$$

In the expansion F_0 is a constant matrix that is independent of the input and F_σ is a low-rank matrix that varies with the input x. We say the latter matrix F_σ is low-rank because its rank is less than the number of layers in the NN; The number of layers is usually much smaller than either input or parameter dimensions in practice. This expansion is new, to the best of our knowledge, and is presented in more detail in Theorem 2.4 below.

The expansion follows from simple linear algebraic tools like Householder reflection and singular value decomposition (SVD); we refer the readers to standard texts in numerical linear algebra for terminology used throughout (Groetsch, 2011; Trefethen & Bau, 1997). Moreover, the key components of the expansion, that is, the singular vectors of the low-rank matrix, can be estimated efficiently at the complexity required to evaluate the NN once. This implies that the decomposition can potentially be used during training or evaluation for stabilization. These two qualities, the simplicity of the analysis as well as the computational efficiency, are indications that the analysis can be adapted to more general situations.

Based on an analysis of this expansion, in conjunction with existing empirical evidence, we conjecture that the adversarial examples originate from the input-dependent low-rank map F_σ. To support our thesis in a high-consequence context, we apply the analysis to a NN model for tsunami early warning that was trained on synthetic geodetic signals and tsunami waveform (Rim et al., 2022). The NN model has a CNN architecture, and more specifically it uses 1D convolution and transpose convolution layers. Such a NN is already known to be susceptible to adversarial examples: for example, a 1D CNN trained on electrocardiogram data suffers from such examples (Han et al., 2020).

We show that adversarial examples found by the PGD algorithm (Kurakin et al., 2017; Madry et al., 2018) have significant components in the unstable terms in the rank-1 expansion of F_σ and that, conversely, the unstable components of F_σ serve as adversarial input perturbations. Similarly, when the unstable components are filtered, the input ceases to cause large changes in the output. This implies that the expansion in Equation 2 can potentially be used as a computationally efficient algorithm for detecting and filtering adversarial input perturbations.

2. Low-Rank Expansion of NNs and Stability Analysis

This section introduces the low-rank expansion for feedforward NNs, and formulates an explanation of how adversarial examples can appear by analyzing that expansion. We begin by defining feedforward NNs whose activation function is set as the rectified linear unit (ReLU) and deriving an expansion. Then we show that the input-dependent part of the expansion has a low-rank representation. Finally, we describe how certain singular behavior in the expansion can lead to adversarial examples.

We will introduce definitions and notations necessary to describe NNs. Given a sequence $n_\ell \in \mathbb{N}$ for index $\ell = 1, \dots, L$ signifying the layer number, let $x \in \mathbb{R}^{n_0}$ denote the input vector, and $\sigma : \mathbb{R} \to \mathbb{R}$ the nonlinear activation (called ReLU) that sets negative values to zero $\sigma(z) \equiv \max\{z, 0\}$. We will omit the range of ℓ when it is clear from the context. Let $A_\ell : \mathbb{R}^{n_\ell \times n_{\ell-1}}$ be linear maps enumerated by the index ℓ. Linear maps can be identified with a matrix in $\mathbb{R}^{n_\ell \times n_{\ell-1}}$ so we write $A_\ell(z) = A_\ell \cdot z = A_\ell z$, for matrix-vector multiplication, and will often refer to a matrix in place of the corresponding linear map. We denote by $\odot$ the entrywise application of the scalar function σ to any vector: if $z = [z_1, \dots, z_{n_\ell}]^T \in \mathbb{R}^{n_\ell}$ then $\sigma \odot z = [\sigma(z_1), \dots, \sigma(z_{n_\ell})]^T \in \mathbb{R}^{n_\ell}$.

Definition 2.1 A feedforward neural network with ReLU activation $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ *is defined as the alternating composition*

$$f(x) = A_L \cdot \sigma \odot A_{L-1} \cdot \ldots \odot A_2 \cdot \sigma \odot A_1(x). \tag{3}$$

Throughout, we will refer to such f simply as a neural network *(NN).*

This is a simplified model widely used in literature analyzing NNs (DeVore et al., 2021; Goodfellow et al., 2016; Szegedy et al., 2014). In practice it is common to have bias terms in the maps A_ℓ or to introduce max-pooling layers, but we will omit these here for the simplicity of exposition. They can also be included in our linearization in a straightforward manner. We will re-introduce these in our computational examples when we experiment with NNs used in practice.

A NN is called *fully connected* when A_ℓ's are allowed to be dense, and the NN is said to be *convolutional* when they are mostly convolutions or transpose convolutions. When there are skip connections, the NN is said to be *residual*. For detailed explanation of these or other terminology, we refer the reader to (Goodfellow et al., 2016).

2.1. Motivation

Our principle aim is to determine the local stability properties of the NNs defined by Equation 3. To motivate our approach, we start by revisiting the well-known analysis presented in (Szegedy et al., 2014). A straightforward analysis (Appendix A1) yields the bound

$$\|f(x) - f(y)\|_2 \leq \left(\prod_{\ell=1}^{L} \|A_\ell\|_2 \right) \|x - y\|_2. \tag{4}$$

Note that we will use the ℓ_2-norm denoted by $\|\cdot\|_2$ as the norm of choice throughout.

Due to this inequality, the Lipschitz constant for f is bounded above by $\prod_{\ell=1}^{L} \|A_\ell\|_2$. The bound implies that one can control the Lipschitz constant L_f of the NN by controlling $\|A_\ell\|_2$. This leads to several strategies that impose certain penalties during training (Bai et al., 2021). However, this bound neglects the effect of the nonlinear activation, and only applies to the constant matrix F_0 that is the product of the matrices A_ℓ,

$$F_0 \equiv A_L \cdot A_{L-1} \cdot \ldots \cdot A_2 \cdot A_1. \tag{5}$$

So $F_0 \cdot x$ is equal to $f(x)$ only when the output of all the intermediate hidden variables all have non-negative entries. Any instability for the linear map F_0 is straightforward to characterize: Its SVD reveals all the singular vectors with large singular values. So one finds all linear subspaces of the domain that can cause an instability through the SVD (see standard texts, for example (Groetsch, 2011)).

In contrast, adversarial examples generally depend nonlinearly on the input x and are not restricted to a fixed linear subspace. This suggests that the instability is inherently nonlinear. Moreover, the upper bound of the Lipschitz constant of f given above in Equation 5 yields an identical estimate for the Lipschitz constants for both f and F_0. That is to say, the bound cannot distinguish between two types of instabilities that have distinct dependencies on the input. As such, it would be surprising if penalizing the spectral norms of the A_ℓ's during training provided a good control of these nonlinear instabilities that are present in f but not in F_0.

The motivation for this work is to extend the completely linear stability analysis Equation 4 to the nonlinear regime. However, the fully nonlinear regime is high-dimensional, making the analysis intractable; instead, we pursue a simplifying analytic technique. The key idea is to approximate the nonlinear activation σ using a rank-1 perturbation of the identity, which dramatically simplifies the stability analysis of the entire NN yet it incorporates sufficient nonlinear effects to account for the adversarial examples. We will provide the details in the following sections.

2.2. Approximating Nonlinear Activations Using Householder Reflectors

Instead of focusing on F_0 to study the stability of f, we focus on a different representation derived from f. First, we write the ReLU activation in an alternate form. Note that ReLU can be written as

$$\sigma \odot z = \frac{1}{2}(z + |z|), \tag{6}$$

where $|z|$ is the entrywise absolute value of the vector. Since the map $z \mapsto |z|$ is a reflection, one can write it as a Householder reflection (Trefethen & Bau, 1997) in the form

$$H_z = I - 2v_z v_z^T. \tag{7}$$

The vector v_z is a scalar multiple of $|z| - z$ (see Appendix A2). Then we have that the ReLU function, when interpreted as a matrix, is also a rank-1 perturbation of the identity,

$$\sigma \odot z = \frac{1}{2}(I + H_z)z = (I - v_z v_z^T)z. \tag{8}$$

This is nonlinear since v_z depends on z, but below we will linearize about a fixed z in defining the matrix. Next, denoting the hidden units in the feedforward network by

$$z_{\ell+1} \equiv \sigma \odot A_\ell z_\ell, \qquad z_0 = x, \tag{9}$$

and enumerating the input dependent vectors v_z appearing in the rank-1 perturbation in Equation 8 corresponding to the input z_ℓ from the previous layer as in

$$v_\ell \equiv v_w \quad \text{where } w = A_\ell z_\ell, \tag{10}$$

we have that

$$\sigma \odot A_\ell z_\ell = (I - v_\ell v_\ell^T)(A_\ell z_\ell) = (A_\ell - v_\ell v_\ell^T A_\ell) z_\ell. \tag{11}$$

Now, we view the map as a rank-1 perturbation of A_ℓ. Denote the perturbation by

$$M_\ell \equiv v_\ell w_\ell^T, \qquad \text{where } w_\ell \equiv -A_\ell^T v_\ell, \quad \ell = 1, \dots, L-1. \tag{12}$$

So the ReLU σ applied to $A_\ell z_\ell$ is the sum of two linear maps A_ℓ and M_ℓ applied to z_ℓ,

$$\sigma \odot A_\ell z_\ell = (A_\ell + M_\ell) z_\ell, \qquad \ell = 1, \dots, L-1. \tag{13}$$

Note that each M_ℓ is dependent on the input, that is, $M_\ell = M_\ell(x)$. We let M_L be the zero matrix.

These derivations result in the following lemma.

Lemma 2.2 *(Matrix representation of a NNs). A NN f as in Definition 2.1 can be written as a matrix-vector product*

$$f(x) = F(x) \cdot x, \tag{14}$$

where the matrix $F : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L \times n_0}$ is given by

$$F(x) = [A_L + M_L(x)]\,[A_{L-1} + M_{L-1}(x)] \cdots [A_1 + M_1(x)], \tag{15}$$

where A_ℓ's are linear mappings in the feedforward architecture, and M_ℓ's are rank-1 matrices. We will call F the matrix representation of the NN f.

One expands the matrix product to see that F is a sum of 2^L linear transformations. Let us denote each of them by F_b where b is an integer represented in L-bits,

$$F_b = B_L B_{L-1} \dots B_1, \quad \text{where } B_\ell \equiv \begin{cases} A_\ell & \text{if } (b)_\ell = 0, \\ M_\ell & \text{if } (b)_\ell = 1. \end{cases} \tag{16}$$

With this notation we may write

$$F = \sum_{b=0}^{2^L-1} F_b. \tag{17}$$

For any $b > 0$, there is a least one ℓ such that $(b)_\ell = 1$ so $B_\ell = M_\ell$ which is rank-1, implying that F_b is also rank-1 (see Appendix A3). Writing the terms F_b with $b > 0$ as a separate sum, F is written as follows. The term with $b = 0$ is $F_0 = A_L A_{L-1} \cdots A_0$, a constant matrix which can potentially be full-rank.

Lemma 2.3 *The matrix representation F of the NN f (as in Definition 2.1) can be written*

$$F(x) = F_0 + F_\sigma(x), \quad \text{where } F_\sigma(x) \equiv \sum_{b=1}^{2^L-1} F_b(x), \quad F_b \text{ in the sum are rank-1.} \tag{18}$$

Here F_σ is dependent on the input x, but F_0 is not. Crucially, F_σ is a sum of rank-1 matrices resulting from the nonlinear activations, and its stability properties cannot be inferred solely from those of F_0. We will show in the next section that $F(x)$ can be viewed as a low-rank perturbation of F_0, and its kernel and range are critical subspaces in relation to adversarial examples. We briefly remark that Leaky ReLU can be used in place of ReLU above (see Appendix A4).

2.3. Low-Rank Householder Expansion of NNs

Recall that the matrices F_b with $b > 0$ appearing in the input dependent part of F_σ in Equation 18 and are each rank-1. But since the sum in Equation 18 includes 2^L terms, it seems that F_σ could have very high rank. On the contrary, it is at most rank $L - 1$ because of the structure of these matrices, a key fact for our analysis. We show that the domain and range for such F_b belong to fixed linear subspaces independent of b, and that these subspaces have dimensions at most $L - 1$.

Put in other terms, the input-dependent part F_σ has the rank-1 expansion,

$$F_\sigma = \sum_{\ell,\ell'=1}^{L-1} C_{\ell\ell'}\, \phi_\ell \psi_{\ell'}^T \quad \text{where } C_{\ell\ell'} = \sum_{b=1}^{2^L-1} \sum_{\substack{\ell_{\min}(b) = \ell \\ \ell_{\max}(b) = \ell'}} c_b. \tag{19}$$

It is natural to take the SVD of the matrix $(C_{\ell\ell'})$ to obtain orthonormal bases $Z = \{\zeta_\ell\}_{\ell=1}^r$ of $\Phi = \{\phi_\ell\}_{\ell=1}^r$ and $\Xi = \{\xi_\ell\}_{\ell=1}^r$ of $\Psi = \{\psi_\ell\}_{\ell=1}^r$, respectively, that have dimensions $r \le L - 1$ and transforms the double sum in Equation 19 into a single sum. The new sum with $L - 1$ terms is expressed as the SVD

$$F_\sigma = \sum_{\ell=1}^{r} d_\ell \zeta_\ell \xi_\ell^T \quad \text{with } r \le L - 1, \tag{20}$$

where $d_1 \geq d_2 \geq \cdots \geq d_r$ are singular values, whereas ζ_ℓ and ξ_ℓ are the left and right singular vectors, respectively. The functions $d_\ell, \zeta_\ell, \xi_\ell$ as well as the rank r depend on x.

The derivation of the low-rank structure in the previous section results in the following theorem. This is the main result of the paper, and the detailed derivations appear in Appendix A5.

Theorem 2.4 (Low-rank Householder expansion of NNs). *Let f be a NN as in Definition 2.1 with the matrix representation F. Then F can be expanded*

$$F(x) = F_0 + F_\sigma(x) \quad \textit{with} \quad \operatorname{rank}(F_\sigma(x)) \leq L - 1. \tag{21}$$

We refer to this expansion as the low-rank Householder expansion *or simply* low-rank expansion of NN f.

We say the expansion is low-rank in the sense that the rank of F_σ is less than the number of layers L and in many applications L is significantly smaller than the dimension of the input n_0 or the dimension of the parameters $\sum_{\ell=0}^{L-1} n_\ell n_{\ell+1}$.

Now one could linearize f about a fixed input x_0 by freezing F_σ in Equation 21. Doing so yields a constant linear map $F_\sigma(x_0)$. In this case Equation 11 becomes a rank-1 approximation of ReLU which is first order accurate, and one obtains a linearization

$$F(x_0) \cdot (x_0 + \delta x) = f(x_0) + (F_0 + F_\sigma(x_0)) \cdot \delta x. \tag{22}$$

The resulting $F(x_0) \cdot x$ agrees with $f(x)$ up to first order in the neighborhood of x_0.

We make a few remarks regarding the linearization. Observe that the ReLU linearized by freezing the input in its rank-1 perturbation Equation 8 is equal to the original ReLU output for positive entries, but is not equal otherwise. As a result, this linearization does not preserve the pattern of zeros introduced by the ReLU and it is distinct from other linearizations that do. There are straightforward extensions of Equation 11 to higher rank approximations that lead to a family of linearizations similar to that in Equation 21. We mention that there are other linearization of the nonlinear activations (e.g., see (Choromanska et al., 2015)).

2.4. Adversarial Examples From the Low-Rank Expansion

We now discuss the conditions under which an adversarial example can exist although F_0 is stable. We will assume that the induced ℓ_2-norm of F_0, and therefore its Lipschitz constant, is not large. Suppose δx is a perturbation about x_0 satisfying

$$\|\delta x\|_2 < \varepsilon \quad \text{and} \quad \frac{\|F_\sigma(x_0) \cdot \delta x\|_2}{\|F_0 \cdot \delta x\|_2} \geq 1. \tag{23}$$

Then we have (via derivations in Appendix A6)

$$\|F(x_0) \cdot \delta x\|_2 \geq \left\| \sum_{\ell=1}^{r} d_\ell \xi_\ell^T(\delta x) \zeta_\ell \right\|_2 - \|F_0\|_2 \|\delta x\|_2. \tag{24}$$

If we choose the perturbation δx satisfying Equation 23 that is also parallel to the orthonormal basis function $\xi_{\ell *}$ we have

$$\|F(x_0)\|_2 \geq \frac{\|F(x_0) \cdot \delta x\|_2}{\|\delta x\|_2} \geq d_{\ell *} - \|F_0\|_2. \tag{25}$$

Large variations in the output relative to the input can arise if $d_{\ell *} \sim 1/\varepsilon$ whereas $\|F_0\|_2 \sim 1$. In that case $\|F(x_0)\|_2 \sim 1/\varepsilon$. Since $F(x_0)$ approximates $F(x)$, the NN must also be unstable with respect to the perturbation δx.

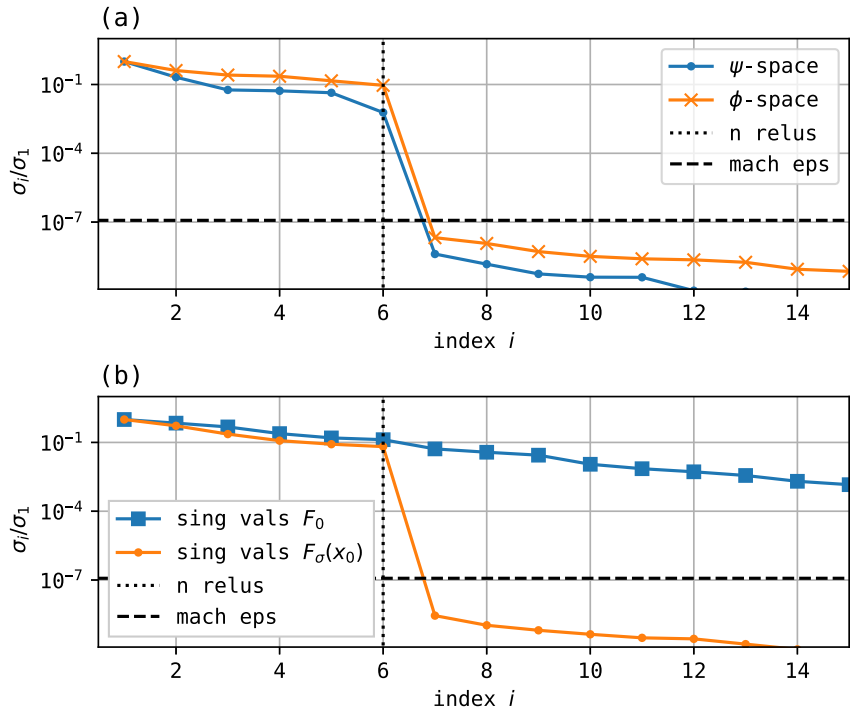

Figure 1. Singular values related to the untrained model. (a) Singular values of matrices whose columns are vectors sampled from Φ and Ψ, and (b) singular values of F_0 and $F_\sigma(x_0)$.

NNs satisfying these conditions are readily found in computational experiments, both in untrained NNs with randomly assigned weights and in more realistic NNs with trained weights. We will demonstrate this in the following section.

3. Computational Examples

In this section, we will perform computational experiments with two NNs. The first is an untrained model with randomly assigned weights. Note that the distributions of weights often used to initialize training typically gives a stable NN, which may then become unstable after training. For this simple example we instead randomly assign weights from a Gaussian process in order to mimic a trained NN without explicitly training the network. This allows us to assess how characteristics of the weight distribution affect the degree of instability. We also infer that instabilities occur even when the NN is relatively small, magnifying as the network gets deeper.

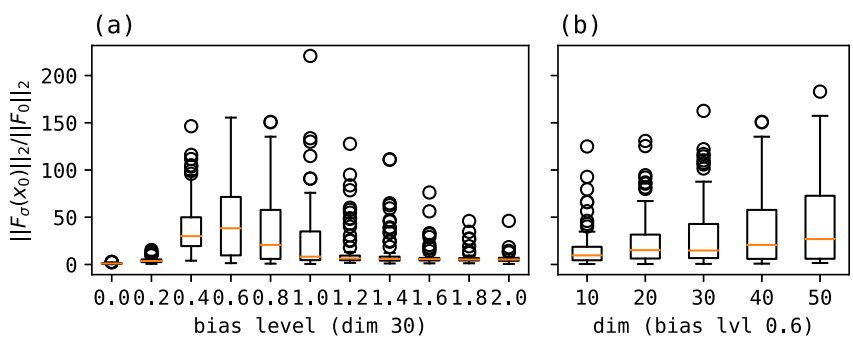

Figure 2. Ratio of spectral norms $R = ||F_\sigma(x_0)||_2/||F_0||_2$ for the untrained model with (a) varying dimensions, and (b) varying bias levels B in Equation 26.

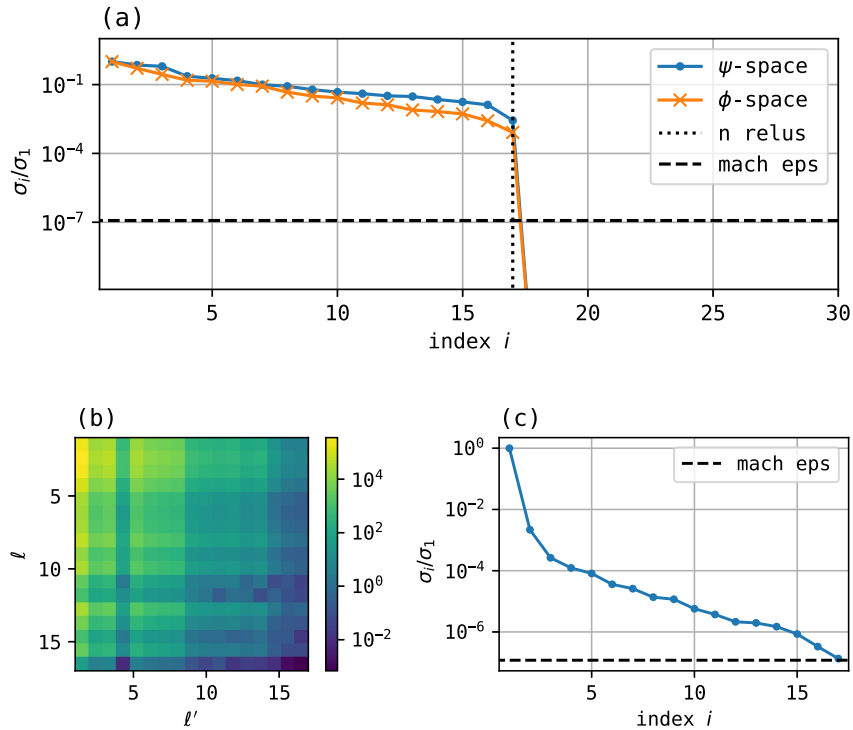

Figure 3. (a) Singular values of matrices whose columns are vectors sampled from Ψ and Φ, (b) logarithmic scale plot of absolute values of $\hat{C}_{\ell\ell'}$ estimated based on $\hat{\Psi}$ and $\hat{\Phi}$, and (c) singular values of the matrix $[\hat{C}_{\ell\ell'}]$.

The second example we study is a CNN model trained on simulated geophysical data that was studied in (Rim et al., 2022) for the purpose of tsunami early warning.

3.1. An Illustrative Example

We first consider the example involving a simple untrained NN with randomly assigned weights. This example indicates that we need neither a large nor a deep NN to see instabilities from perturbations arise. However, the instabilities become more apparent and with higher bias in larger networks. We create a fully connected NN in the form of Equation 3 with $L = 7$ layers and dimensions $n_0 = n_1 = \cdots = n_L = 30$. We choose randomly generated weights for all the linear layers A_ℓ, drawn from the standard normal distribution $\mathcal{N}(0, 1)$. We also randomly select an input and we expand the NN. The input is a vector whose entries were drawn from the uniform distribution $\mathcal{U}(-1/2,\ 1/2)$.

We compute an orthonormal basis for Φ and Ψ as defined in (Appendix A14) by sampling the vectors $\{F_b x_0 : b = 1, \ldots, 2^L - 1\}$ and $\{F_b^T f(x_0) : b = 1, \ldots, 2^L - 1\}$ then taking their SVD. The singular values are shown in Figure 1a. There is a sharp drop in the singular values to single-precision machine roundoff level after the first 6 singular values. This agrees well with our low-rank expansion in Equation 21, showing that Φ and Ψ are indeed at most $L - 1 = 6$ dimensional.

For this untrained model, it is not difficult to compute all the individual terms F_b for $b = 1, \ldots, 2^L - 1$ and explicitly form the sum $F_\sigma(x_0) = \sum_{b=1}^{2^L-1} F_b$. Then one obtains the low-rank expansion by taking the SVD of the computed $F_\sigma(x_0)$. The singular values are shown in Figure 1b. The index where there is a gap in the singular values agrees precisely with the previous plot. We have computed the correlation between the two sets of singular vectors, and found that the two orthonormal bases span the same space up to the level of numerical precision.

The size of the gap in the singular values suggests that an orthonormal basis for Φ and Ψ can be found simply by sampling $F_b x$ and $F_b^T f(x_0)$ for a number of values b then computing the SVD. Upon computing these bases, one samples the matrix $C_{\ell\ell'}$ and then computes its SVD to estimate the low-rank expansion indirectly.

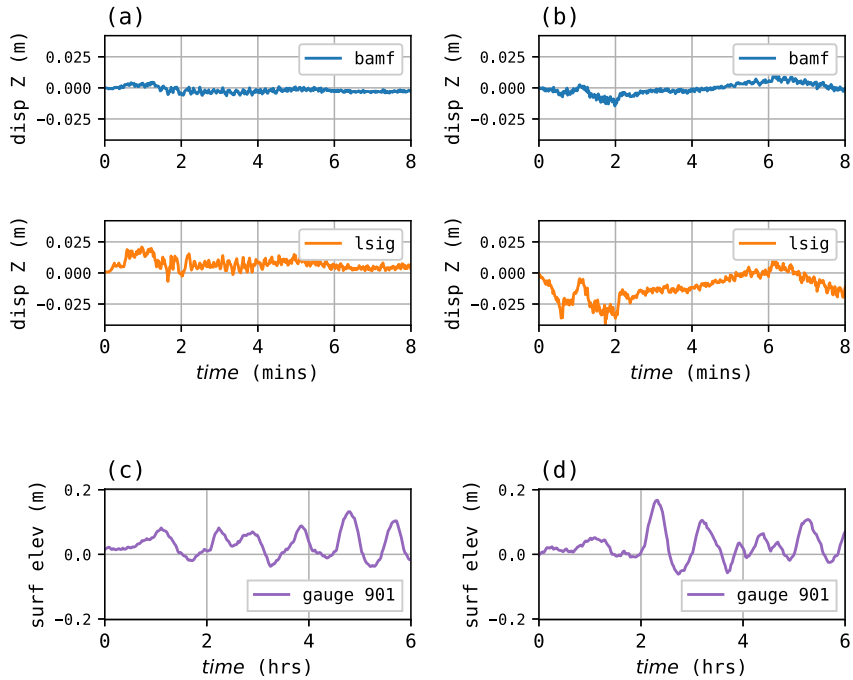

Figure 4. The singular vector pairs (ξ_ℓ, ζ_ℓ) for the domain and the range of F_σ (20) corresponding to indices $\ell = 1, 2$. (a) The right singular vector ξ_1 plotted as a Global Navigation Satellite System input at two selected stations, (c) the left singular vector ζ_1 plotted at a gauge output. Similar plot for ξ_2 (b) and ζ_2 (d).

Next, we attempt to verify whether we can generate adversarial examples from a random untrained model. Above we discussed the conditions in which adversarial examples can be precisely identified by the low-rank expansion Equation 21. We will show that we can meet these conditions with our untrained model, if we make use of a customized Gaussian process noise model for the weights in the linear layers. We assign randomized weights entrywise by forcing that A_ℓ is lower Hessenberg, that its entries below the first subdiagonal are zero, then drawing the nonzero entries from a normal distribution with a negative bias,

$$(A_\ell)_{ij} \quad \sim \quad \mathcal{N}(-B, 1) \quad \text{if } i - j > 1 \tag{26}$$

where B is a bias in the range $[0, 2]$. We explicitly compute spectral norms of F_0 and $F_\sigma(x_0)$ to compare their ratios

$$R = \frac{||F_\sigma(x_0)||_2}{||F_0||_2}. \tag{27}$$

Note that if $R \gg 1$ it implies that δx satisfying Equation 25 with $d_{\ell_*} = R||F_0||_2$ exists ($\ell_* = 1$), resulting in

$$\frac{||F(x_0) \cdot \delta x||_2}{||\delta x||_2} \geq (R - 1)||F_0||_2. \tag{28}$$

To observe the dependence on the dimension $m = n_0 = \cdots = n_L$ and the bias level B, we calculate the ratio R for 100 randomly drawn NNs for each bias level $B = 0.0, 0.2, 0.4, \ldots, 2.0$ for fixed dimension $m = 30$, then for each dimension $m = 10, 20, \ldots, 50$ for fixed bias level $B = 0.6$.

The statistics of the ratio is shown in two sets of box plots in Figure 2. The ratio easily reaches 200 and above, meaning that a small perturbation in the input in the direction of the first singular vector (ξ_1 in Equation 20) can cause a much larger response in $F_\sigma(x_0)$ compared to that in F_0. One observes that the bias level at around 0.8 maximizes the ratio, and that in the unbiased or highly biased cases the ratio stays modest, although outliers exist in the latter case. When the dimension m is increased, the quartiles tend to increase with the dimension. This

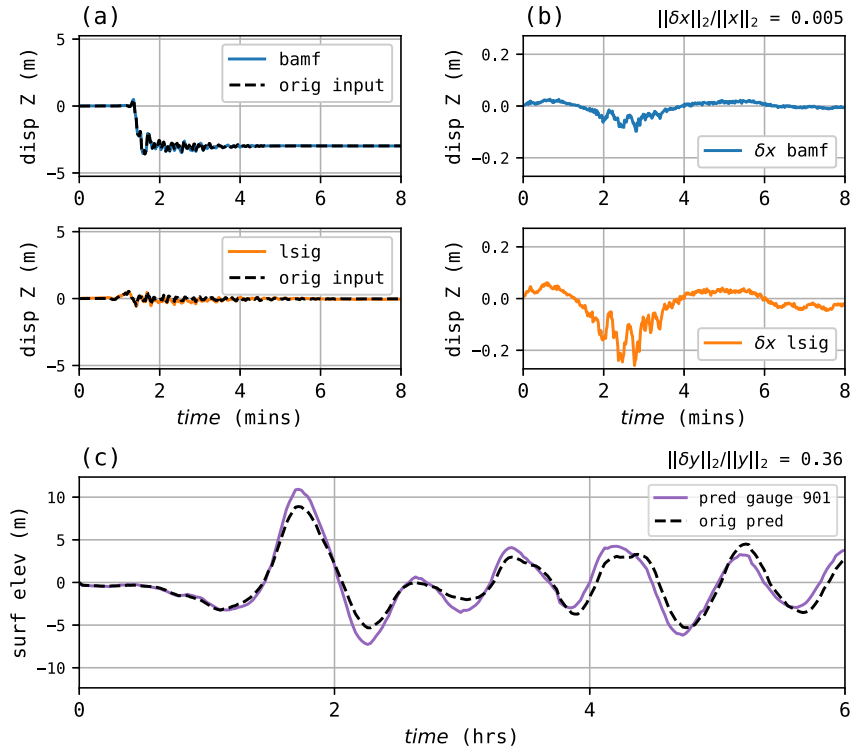

Figure 5. An adversarial example found by projected gradient descent. (a) The perturbed input $x_0 + \delta x$ at two selected stations, (b) the perturbation δx, and (c) the resulting perturbed output $f(x_0 + \delta x)$ at gauge 901. An imperceptible 0.5% change in the input causes a large 36% change in the output.

indicates that among NNs with the same random model for the weights those with larger dimensions are more likely have larger values of R.

3.2. Tsunami Early Warning Model

We apply our analysis to a tsunami early warning model (Rim et al., 2022). This model is summarized very briefly below; full details and many figures illustrating the data and quality of forecasts are presented in the original paper. We compute the low-rank expansion in Equation 21, estimate the input space Ξ in Equation 20, then compare the adversarial perturbations found by PGD with perturbations to the singular vectors $\{\xi_\ell\}$ in the expansion. We repeat the experiments for standard noise perturbations.

3.2.1. Model Architecture and Its Expansion

The model is a standard CNN that maps the geodetic time-series measurements coming from 60 GNSS stations, each with east (E), north (N), vertical (Z) components. Accordingly, the input dimension is $n_0 = 60 \cdot 3 \cdot 512$ because there are 60 stations and 3 components and the length of the time-series is 512. The output of the NN is the full tsunami waveform at 3 different gauge locations. The output dimension is $n_L = 3 \cdot 256$, as there are 3 gauges and the length of the time-series is 256. For training data we used 1,300 synthetic earthquakes originally generated by Melgar et al. (2016) using the MudPy software (Melgar, 2020), which also generates synthetic GNSS data. The seafloor deformation for each event was then used to generate the corresponding tsunami waveform using the GeoClaw software (Clawpack Development Team, 2020).

Throughout this section, we will plot inputs and outputs at only two GNSS stations named bamf and lsig and only show the Z component of the signal, in order to illustrate typical results. We also show the tsunami wave gauge forecasts as only a single gauge location, Gauge 901, which is located in Discovery Bay, WA. The plots for the other components, and at the other GNSS stations or gauge locations, have similar characteristics (see Figures S1–S6 in Supporting Information S1).

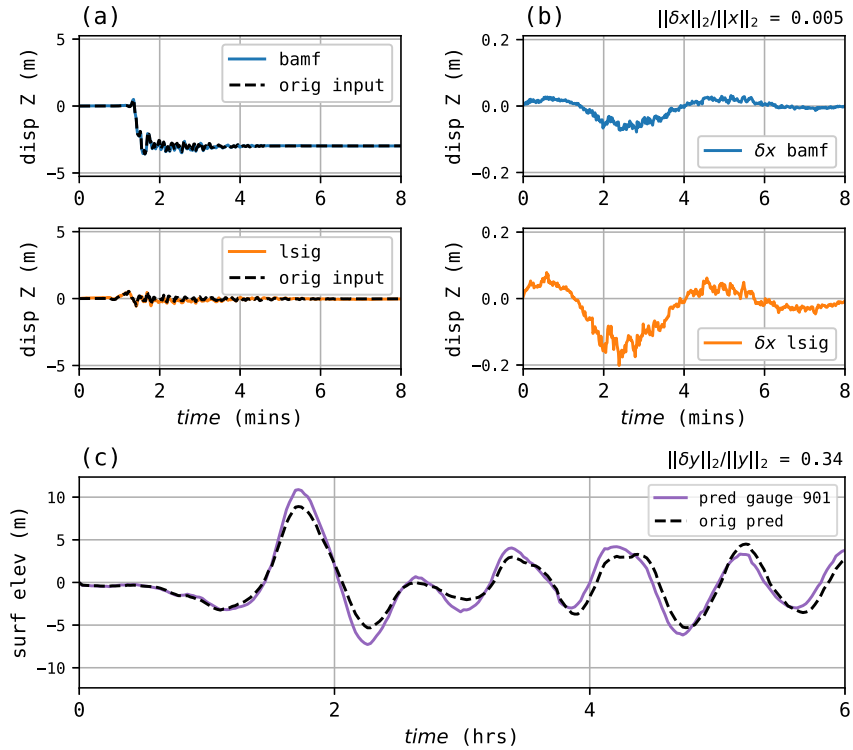

Figure 6. Input and output perturbations for the projected adversarial example. (a) The perturbed input $x_0 + (\delta x)_{\text{proj}}$ at two selected stations, (b) the projected perturbation $(\delta x)_{\text{proj}}$, and (c) the resulting perturbed output $f(x_0 + (\delta x)_{\text{proj}})$. An imperceptible 0.5% change in input causes a large 34% change in the output.

Given an input, the model first applies a sequence of 9 pairs of convolutional and max-pool layers, then applies the 8 transpose convolutional layers. Between each pairs, we apply the Leaky ReLU activation function with negative slope 1/2. The channel output sizes for each convolutional and transpose convolutional layer are 2^6-2^6-2^7-2^7-2^7-2^8-2^8-2^9-2^9 and 2^9-2^9-2^8-2^8-2^7-2^7-2^6-2^6, respectively. In the study, an ensemble of 25 NNs were trained. We will analyze the stability of one NN in the ensemble.

Here we will omit the details regarding this NN model and focus on studying its stability properties. For full details regarding the geographical locations of these stations and gauges, as well as the NN model and its training, we refer to the original reference (Rim et al., 2022); also, see Data and Resources.

This NN architecture includes bias terms and max-pool layers that were absent in our analysis of Equation 3. Introducing the bias terms result in additional input-dependent terms in our expansion in Equation 20 that we can include into $F_\sigma(x_0)$ in a straightforward manner, and this does not increase its rank. We also freeze the max-pool layers (Appendix A7) so that they effectively become permutations, resulting in a linearization. We note here that the technique linearizing the ReLU above can also be applied to max-pool layers, and doing so would result in a different linearization. We will not pursue this here, however.

3.2.2. Basis Estimation

We sample the input and output spaces $\Psi \equiv \text{span}\{\psi_\ell\}_\ell^r$ and $\Phi \equiv \text{span}\{\phi_\ell\}_\ell^r$ (recall Equation 19) by computing the outer product F_b for a small subsample of b of size $3L$ (out of $2^L - 1$ possible values, see (Appendix A12)). Then collecting these sampled vectors and taking the SVD, we see a gap in the singular values and obtain the bases $\hat{\Phi} = \{\hat{\phi}_\ell\}$ and $\hat{\Psi} = \{\hat{\psi}_\ell\}$. See Figure 3 for a plot of singular values of the sampled basis. Next, we use them as surrogates for the sum in Equation 19

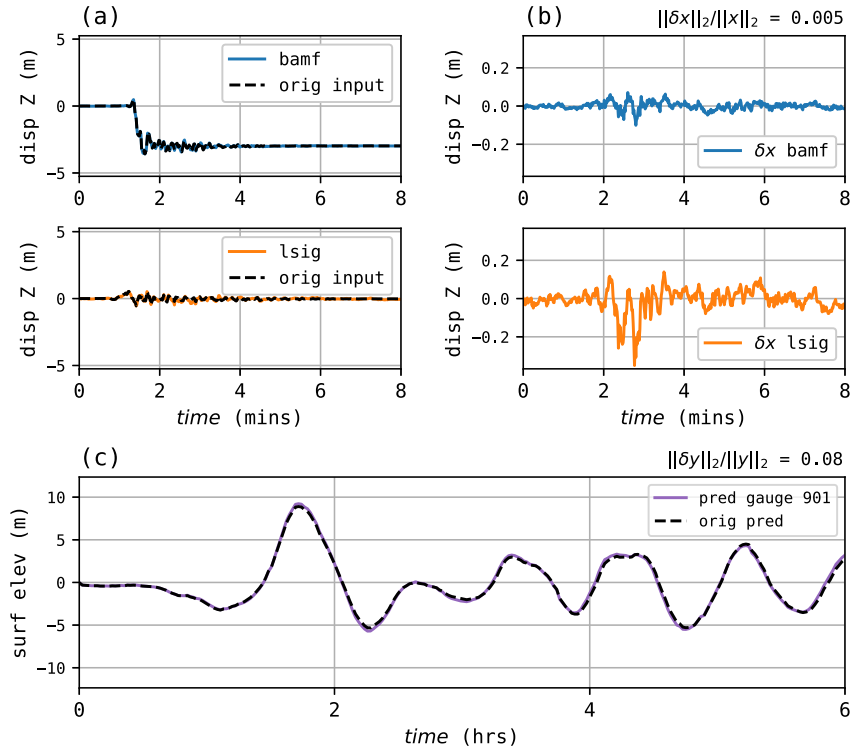

Figure 7. Filtering of an adversarial example found by projected gradient descent. (a) The perturbed input $x_0 + (\delta x)_{\text{filter}}$ at two selected stations, (b) the filtered perturbation $(\delta x)_{\text{filter}}$, and (c) the resulting perturbed output $f\left(x_0 + (\delta x)_{\text{filter}}\right)$. The amount of output perturbation is at 8%, close to that of the input perturbation.

$$\hat{F}_\sigma(x_0) = \sum_{\ell,\ell'} \hat{C}_{\ell\ell'} \hat{\phi}_\ell \hat{\psi}_{\ell'}. \tag{29}$$

Next, we calculate $\hat{C}_{\ell\ell'}$ by computing all of F_b for $b = 1, \ldots, 2^L - 1$ and projecting them into the bases $\hat{\Phi}$ and $\hat{\Psi}$. Taking the SVD of $\hat{C}_{\ell\ell'}$, we obtain our approximation $\hat{Z} = \{\hat{\zeta}_\ell\}$ and $\hat{\Xi} = \{\hat{\xi}_\ell\}$ of Z and Ξ. A plot of absolute values of $\hat{C}_{\ell\ell'}$, together with the singular values of $\hat{C}_{\ell\ell'}$ are shown in Figure 3. First two basis in $\hat{Z}$ and $\hat{\Xi}$ are shown in Figure 4.

We remark here that the estimation of $\hat{C}_{\ell\ell'}$ is a computationally expensive task, whereas the computation of the orthonormal bases $\hat{\Phi}$ and $\hat{\Psi}$ only required evaluations of the NN for $3L$ sample values of b, yet the plot of absolute values of $\hat{C}_{\ell\ell'}$ shows that the ordering of the bases remains largely unchanged. Therefore the estimated bases $\hat{\Phi}$ and $\hat{\Psi}$ can potentially serve as computationally cheap substitutes for $\hat{Z}$ and $\hat{\Xi}$. As a consequence, the basis in the low-rank expansion can be computed at a small cost at any input x_0, that is, at the cost equivalent to one evaluation of the NN.

3.2.3. Adversarial Examples From the Low-Rank Expansion

Now that we have an estimate of the low-rank expansion in Equation 20 for the tsunami model, we perform tests to verify if adversarial examples discovered via optimization lie in the linear subspace Ψ.

We compute an adversarial example $x_0 + \delta x$ about an input x_0 using PGD. In PGD, we attempt to maximize the test loss, a strategy that is sometimes called an untargeted attack. Within the PGD algorithm, we have used both ℓ_∞ and ℓ_2-projections and found that ℓ_2-projection yields worse adversarial examples. We will use ℓ_2 throughout but using ℓ_∞-projections instead lead to similar conclusions. To measure the size of the perturbations in input and output, we will use the relative 2-norm (e.g., for input change we use the ratio $\|\delta x\|_2/\|x\|_2$).

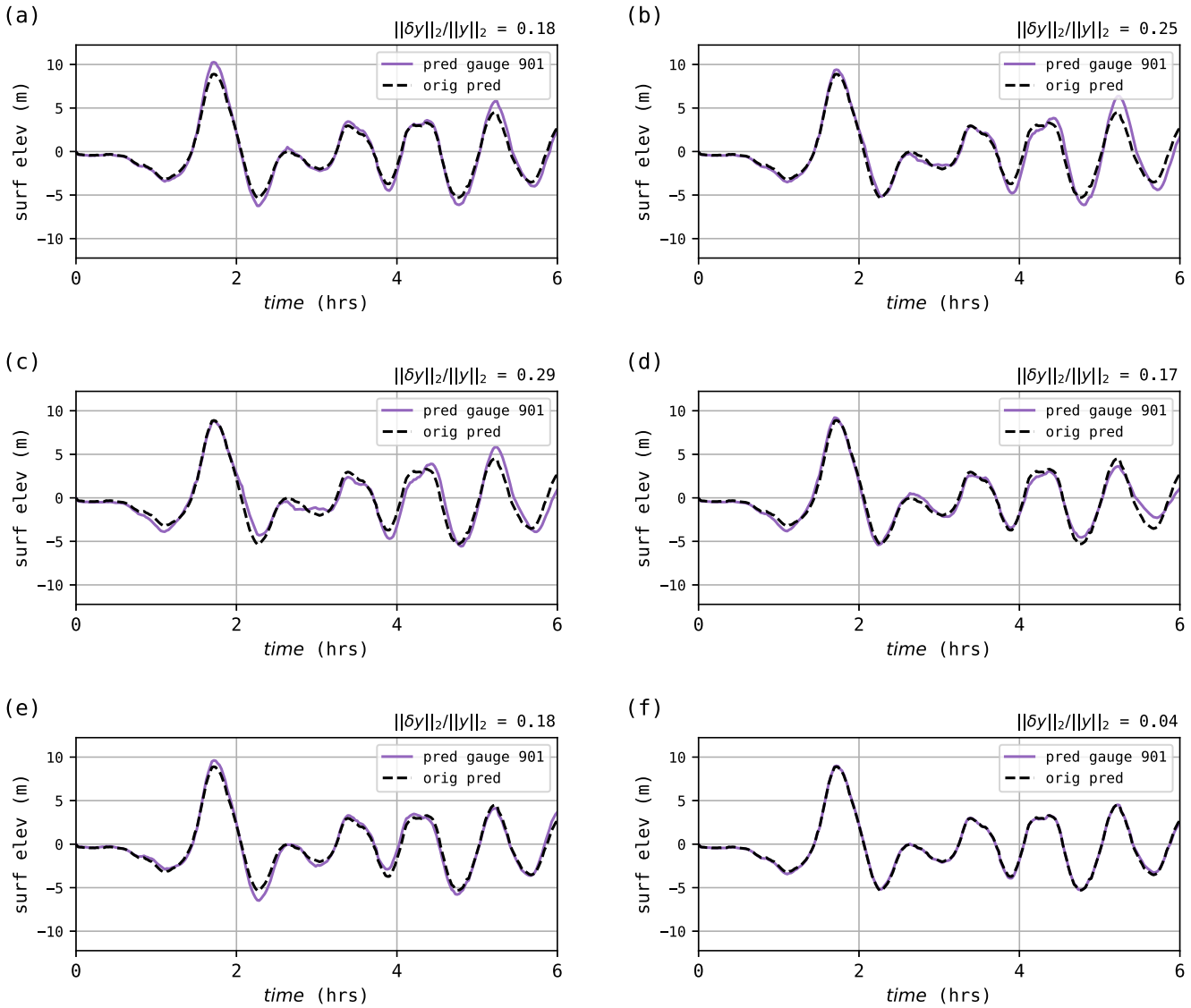

Figure 8. Perturbation of output caused by input perturbation along the basis functions of $\hat{\Xi}$. (a) to (f) Show output perturbation when δx is chosen to be parallel to ξ_ℓ and rescaled to be 0.5% of the $||x_0||_2$, for index ℓ from 1 to 6. Some of these input perturbations cause large changes in the output, comparable to the adversarial perturbations found by projected gradient descent in Figure 5.

Next, we compute the projection of δx onto $\hat{\Psi}$

$$(\delta x)_{\text{proj}} \equiv \hat{\Psi}\hat{\Psi}^T(\delta x). \tag{30}$$

then compare the resulting output perturbation. In order to show a fair comparison of the output perturbations from two perturbations δx and $(\delta x)_{\text{proj}}$, we will scale them so that their 2-norm is 0.5% of the 2-norm of the input x_0.

We show in Figures 5 and 6 the adversarial example $x_0 + \delta x$, the input perturbation δx, the resulting output $y_0 + \delta y$ where $y_0 = f(x_0)$. We observe that $(\delta x)_{\text{proj}}$ causes a large perturbation in the output comparable to δx, 34% change versus 36%. Note also that $(\delta x)_{\text{proj}}$ has a profile similar to that of δx, showing that much of the perturbation is unchanged through the filter.

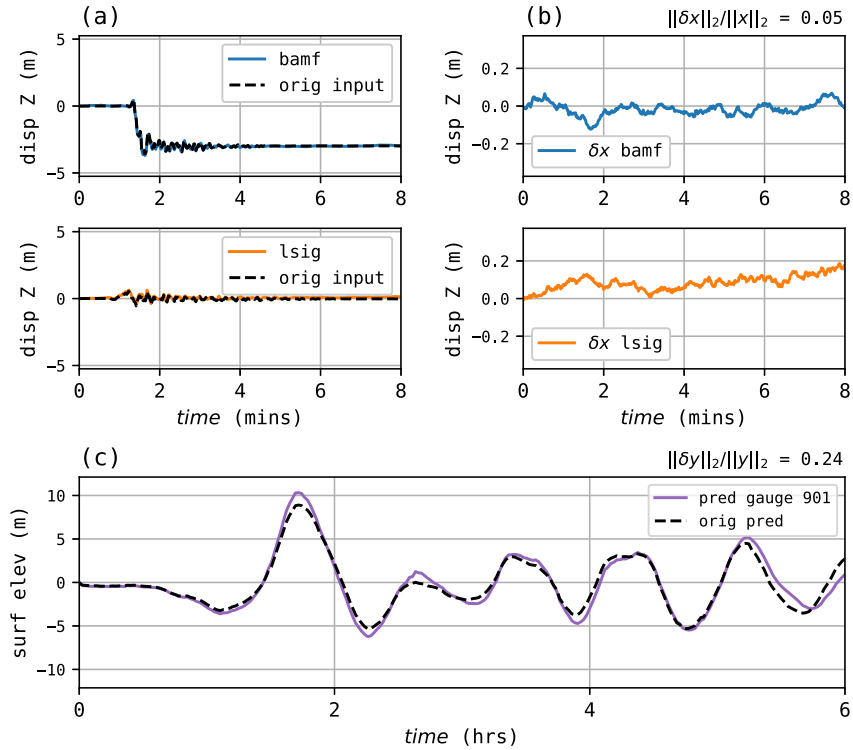

Figure 9. An adversarial example drawn from Brownian noise. (a) The perturbed input $x_0 + \delta x$ at two selected stations, (b) the perturbation δx, and (c) resulting perturbed output $f(x_0 + \delta x)$. Noise level of 5% in the input causes 24% change in the output.

This suggests the possibility that removing from δx its projection to Ψ can lead to a more stable output. So we define the filtered perturbation as the orthogonal complement

$$(\delta x)_{\text{filter}} \equiv \left(I - \hat{\Psi}\hat{\Psi}^T\right)(\delta x). \tag{31}$$

We again scale the filtered input to be 0.5% of the ℓ_2-norm of the input, as we have for δx and $(\delta x)_{\text{proj}}$, and compute the output of the perturbed input. The results are plot in Figure 7. Now the output perturbation is at a similar level as that of the input perturbation, 3% versus 2%.

Next, following the analysis above, we measure the amount of output perturbation caused by perturbing the input using the basis $\hat{\Xi}$. The results for the first six vectors are shown in Figure 8. Observe that perturbation by ξ_3 causes 29% change in the output, a severe change comparable to that of the adversarial example found by PGD. Plots for all GNSS stations and gauges are shown in Figures S1–S6 in Supporting Information S1.

3.2.4. Noise in GNSS Data

We now consider the case that the input perturbation δx is physical noise rather than a specific perturbation in an adversarial example computed by PGD. Since this is the type of noise that is most likely to affect real measurements, it is imperative to check if such innocuous and expected perturbations could excite these instabilities. We generate Brownian noise for each GNSS time-series, then use it as the perturbation δx. The resulting perturbation is shown in Figure 9. A noise level of 5% causes a 24% change in the output. After filtering, however, the output change is reduced to 3% as shown in Figure 10.

To study the effect of input noise more closely, we experiment with three different types of noise perturbations: white noise, Brownian noise, and power-law noise. For the latter, the spectral density is chosen as the Gaussian $\exp\left[-k^2/\sigma^2\right]$ with $\sigma = \sqrt{10}$. We draw 1,000 sample perturbations from each noise and compute the input and

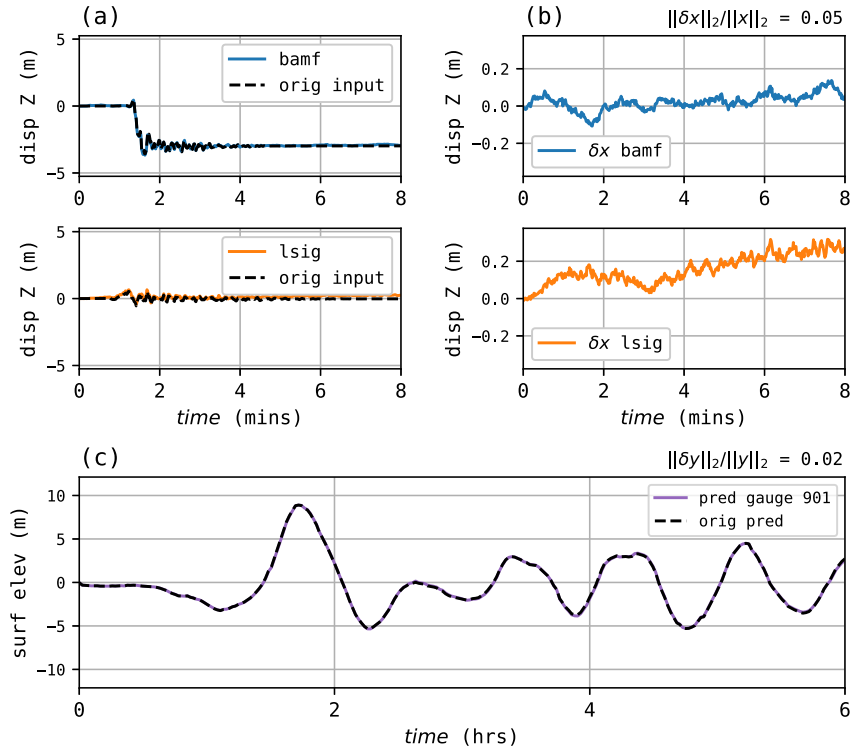

Figure 10. Filtering of Brownian noise. (a) The perturbed input $x_0 + (\delta x)_{\text{filter}}$ at two selected stations, (b) the filtered perturbation $(\delta x)_{\text{filter}}$, and (c) resulting perturbed output $f\left(x_0 + (\delta x)_{\text{filter}}\right)$. Filtering the input reduces the output perturbation by a factor of 10.

output change ratio $||\delta y||_2/||\delta x||_2$. We scale the noise uniformly across samples so that $||\delta x||_2/||x_0||_2$ is around 5%. In addition, we compute the same ratio for the filtered noise $(\delta x)_{\text{filter}}$ as in Equation 31.

The results are plotted in Figure 11. The NN is not significantly affected by white noise (a similar observation can be made in Figure 7 where filtered perturbation resembles white noise) whereas Brownian noise and power law noise do cause large changes in the output, up to a factor of Equations 5–6. Since tsunami prediction is a risk intolerant application and a fair amount of noise is expected in the measurement, large changes in the output have critical consequences. However, filtering out the noise tends to yield significantly smaller output changes and could reduce the sensitivity of the network to adversarial examples.

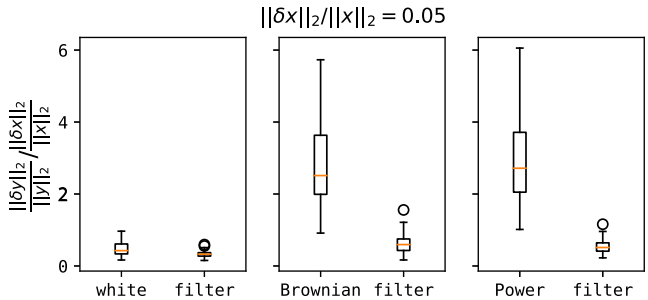

Figure 11. Ratio of input and output changes for different types of noise, white noise, Brownian noise, and power law noise.

4. Conclusion

This work proposes a novel stability analysis, derived from a new low-rank expansion of feedforward NNs that uses ReLU-type activations. Our analysis suggests a mechanism by which adversarial examples can occur, and establishes a close relation between these examples and the SVD of the input dependent low-rank matrix that appear in the expansion. The computational examples demonstrate that the analysis applies to both untrained models initialized with random weights and a tsunami warning model trained on empirical data. These results reveal that the models trained on simulated geophysical data also suffer from the same instabilities as those trained on image classification data, and urge caution in deploying NN models trained for hazard prediction and assessment. The analysis will be potentially useful in devising new approaches for developing defenses against adversarial examples, which is important for security- or safety-critical applications. The analysis is widely applicable to NN models trained on other geophysical data

sets, such as larger seismic data sets that include real measurements, making it a computationally efficient tool for identifying and analyzing the instabilities in various applications.

Appendix A: Mathematical Details

A1. Naïve Estimate for Lipschitz Constant of NNs

One first observes that σ has a Lipschitz constant $L_\sigma = 1$ since

$$\|\sigma(x) - \sigma(y)\|_2 \leq \|\max\{x,0\} - \max\{y,0\}\|_2 \leq \|x - y\|_2 \tag{A1}$$

for all $x, y \in \mathbb{R}^{n_\ell}$. This allows one to derive the bound

$$\begin{aligned}
&\|f(x) - f(y)\|_2 \\
&\leq \|A_L \cdot \sigma \odot \ldots \odot A_1(x) \; - \; A_L \cdot \sigma \odot \ldots \odot A_1(y)\|_2 \\
&\leq \|A_L\|_2 \|\sigma \odot A_{L-1} \cdot \ldots \odot A_1(x) \; - \; \sigma \odot A_{L-1} \cdot \ldots \odot A_1(y)\|_2 \\
&\leq \|A_L\|_2 \|A_{L-1} \cdot \ldots \odot A_1(x) \; - \; A_{L-1} \cdot \ldots \odot A_1(y)\|_2 \\
&\quad \vdots \\
&\leq \left(\prod_{\ell=1}^{L} \|A_\ell\|_2 \right) \|x - y\|_2.
\end{aligned} \tag{A2}$$

A2. The Householder Reflector

The formula for the vector v_z appearing in Equation 7 is given by

$$v_z = \begin{cases} \dfrac{|z| - z}{[2z^T(z - |z|)]^{\frac{1}{2}}} & \text{if } |z| \neq z, \\ 0 & \text{if } |z| = z. \end{cases} \tag{A3}$$

A3. Rank of Products

The rank of the matrix F_b for $b > 0$ is upper bounded as follows,

$$\operatorname{rank}(F_b) = \operatorname{rank}(B_L \ldots B_1) \leq \min_\ell (\operatorname{rank}(B_\ell)) \leq 1. \tag{A4}$$

A4. Householder Reflector for Leaky ReLUs

Leaky ReLU $\hat{\sigma}$ (Goodfellow et al., 2016) can be written as

$$\hat{\sigma} \odot z = (1 - \beta)z + \beta|z| \tag{A5}$$

for some hyper-parameter $\beta \in \left[\frac{1}{4}, \frac{1}{2}\right]$. We write

$$\begin{aligned}
\hat{\sigma} \odot z &= \left((1 - \beta)I + \beta H_z\right) z \\
&= \left((1 - \beta)I + \beta\left(I - v_z v_z^T\right)\right) z = \left(I - \beta v_z v_z^T\right) z,
\end{aligned} \tag{A6}$$

and the expansion in Equation 18 follows with minor changes.

A5. Derivation of the Low-Rank Structure in the Householder Expansion

We show that there are linear subspaces $\Psi \subset \mathbb{R}^{n_L}$ and $\Phi \subset \mathbb{R}^{n_0}$ satisfying

$$\begin{aligned} &\text{(i)} \quad \text{Range}(F_b) \subset \Phi,\ [\text{Ker}(F_b)]^{\perp} \subset \Psi \text{ for all } b > 0, \\ &\text{(ii)} \quad \dim(\Phi), \dim(\Psi) \leq L - 1. \end{aligned} \tag{A7}$$

We will proceed by explicitly finding such Φ and Ψ. For any F_b with $b > 0$, let us denote

$$\begin{aligned} \ell_{\min}(b) &\equiv \min\{\ell = 1, \ldots, L : (b)_\ell = 1\}, \\ \ell_{\max}(b) &\equiv \max\{\ell = 1, \ldots, L : (b)_\ell = 1\}. \end{aligned} \tag{A8}$$

For example, given a 9-bit number $b = 001010100$, we would have $\ell_{\min}(b) = 3$ and $\ell_{\max}(b) = 7$. We will sometimes drop the dependence on b and write $\ell_{\min}, \ell_{\max}$ for $\ell_{\min}(b), \ell_{\max}(b)$ for brevity.

Recalling that each term F_b appearing in F_σ was defined as the matrix product

$$F_b = A_L \cdots A_{\ell_{\max}+1} M_{\ell_{\max}} \cdots M_{\ell_{\min}} A_{\ell_{\min}-1} \cdots A_1, \tag{A9}$$

and inserting the outer-product form for $M_{\ell_{\min}}$ and $M_{\ell_{\max}}$ as in Equation 12,

$$F_b = A_L \cdots A_{\ell_{\max}+1} \left(v_{\ell_{\max}} w_{\ell_{\max}}^T\right) \cdots \left(v_{\ell_{\min}} w_{\ell_{\min}}^T\right) A_{\ell_{\min}-1} \cdots A_1. \tag{A10}$$

Grouping the matrix-vector products we may rearrange,

$$\begin{aligned} F_b &= \left(A_L \cdots A_{\ell_{\max}+1} v_{\ell_{\max}}\right)\left(w_{\ell_{\max}}^T \cdots v_{\ell_{\min}}\right)\left(w_{\ell_{\min}}^T A_{\ell_{\min}-1} \cdots A_1\right) \\ &= \left(w_{\ell_{\max}}^T \cdots v_{\ell_{\min}}\right)\left(A_L \cdots A_{\ell_{\max}+1} v_{\ell_{\max}}\right)\left(w_{\ell_{\min}}^T A_{\ell_{\min}-1} \cdots A_1\right), \end{aligned} \tag{A11}$$

Since $w_{\ell_{\max}}^T \cdots v_{\ell_{\min}}$ is scalar. As a result F_b is an outer product of two vectors,

$$F_b = c_b \, \phi_{\ell_{\min}} \psi_{\ell_{\max}}^T \quad \text{for } b > 0, \tag{A12}$$

in which the vectors ϕ_ℓ, ψ_ℓ and scalar c_b are defined as follows.

- The vectors ϕ_ℓ and ψ_ℓ are

$$\phi_\ell \equiv A_L \cdots A_{\ell+1} v_\ell, \quad \psi_\ell \equiv -A_1^T \cdots A_{\ell-1}^T A_\ell^T v_\ell. \tag{A13}$$

Corresponding linear spaces spanned by $\{\phi_\ell\}$ and $\{\psi_\ell\}$ are the linear subspaces from Appendix A7 we are seeking, so we let

$$\begin{aligned} \Phi &\equiv \text{span}\{\phi_\ell : \ell = 1, \ldots, L-1\}, \\ \Psi &\equiv \text{span}\{\psi_\ell : \ell = 1, \ldots, L-1\}. \end{aligned} \tag{A14}$$

- The scalar coefficients c_b are

$$c_b \equiv w_{\ell_{\max}}^T B_{\ell_{\max}-1} \cdots B_{\ell_{\min}+1} v_{\ell_{\min}} \tag{A15}$$

where B_ℓ are as in Equation 16. If this is an empty product, we set $c_b = 1$.

As a result of Appendix 12 the matrices F_b $(b = 1, \ldots, 2^L - 1)$ have their range in Φ and the orthogonal complement of its kernel in Ψ as required in Appendix A7. We note here that while the number of terms in the sum for F_σ in Equation 18 is exponential in L, the domain and range of F_σ belongs to a linear space whose dimension is linear in L.

A6. Perturbation Lower Bounds

The lower bound in Equation 25 follows by the triangle inequality

$$\begin{aligned} \|F(x_0)\cdot\delta x\|_2 &= \|[F_0 + F_\sigma(x_0)]\cdot\delta x\|_2 \\ &\geq \|F_\sigma(x_0)\cdot\delta x\|_2 - \|F_0\cdot\delta x\|_2 \\ &\geq \|F_\sigma(x_0)\cdot\delta x\|_2 - \|F_0\|_2\|\delta x\|_2, \end{aligned} \tag{A16}$$

then using the rank-1 expansion in Equation 20.

A7. Freezing the Max-Pool Layers

1D max-pool layers in our architecture (Goodfellow et al., 2016) picks out the larger of every two entries. We freeze these layers by writing the selection as a multiplication by a permutation matrix, and viewing the permutation to be input-independent and fixed. For example,

$$z = [-0.0803,\ -0.3806,\ -0.2336, 0.0252]^T, \quad \texttt{pool(z)} = [-0.0803, 0.0252]^T \tag{A17}$$

The frozen max-pool layer would correspond to the linear mapping represented by the constant matrix multiplication

$$z \mapsto \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} z. \tag{A18}$$

Data Availability Statement

The code used to produce results in this work are archived in a Zenodo code repository (Rim & Suri, 2024). We have made use of the data and the NN model from (Melgar, 2016; Rim et al., 2022). The geodetic data therein was generated using the MudPy software (Melgar, 2020) that generates random earthquakes (LeVeque et al., 2016), synthetic GNSS data and sea floor deformations. The resulting tsunami waveform data was generated using the GeoClaw software from Clawpack (Berger et al., 2011; Clawpack Development Team, 2020). Computational experiments involving NNs were conducted using the PyTorch Library (Paszke et al., 2019). All of the software used is open source and freely available.

Acknowledgments
The authors thank Robert Baraldi and Christopher M. Liu for carefully reading the manuscript and providing valuable feedback. The authors acknowledge the Research Infrastructure Services (RIS) group at Washington University in St. Louis for providing computational resources and services needed to generate the research results delivered within this paper (URL ris.wustl.edu). DR thanks Gerrit Welper, Ari Stern, Kui Ren and Weilin Li for helpful discussions. DR and RJL thank Kenjiro Terada's group and Tohoku University for many useful discussions and financial support to visit Sendai and collaborate on real-time tsunami forecasting using machine learning.

References

Bai, T., Luo, J., Zhao, J., Wen, B., & Wang, Q. (2021). Recent advances in adversarial training for adversarial robustness. In Z.-H. Zhou (Ed.), *Proceedings of the thirtieth international joint conference on artificial intelligence, IJCAI-21* (pp. 4312–4321). International Joint Conferences on Artificial Intelligence Organization.

Berger, M. J., George, D. L., LeVeque, R. J., & Mandli, K. T. (2011). The GeoClaw software for depth-averaged flows with adaptive refinement. *Advances in Water Resources*, *34*(9), 1195–1206. https://doi.org/10.1016/j.advwatres.2011.02.016

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., et al. (2013). Evasion attacks against machine learning at test time. In H. Blockeel, K. Kersting, S. Nijssen, & F. Železný (Eds.), *Machine learning and knowledge discovery in databases* (pp. 387–402).

Choromanska, A., Henaff, M., Mathieu, M., Ben Arous, G., & LeCun, Y. (2015). The loss surfaces of multilayer networks. In G. Lebanon & S. V. N. Vishwanathan (Eds.), *Proceedings of the eighteenth international conference on artificial intelligence and statistics* (Vol. 38, pp. 192–204).

Clawpack Development Team. (2020). Clawpack software. Retrieved from http://www.clawpack.org

Cohen, J., Rosenfeld, E., & Kolter, Z. (2019). Certified adversarial robustness via randomized smoothing. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 1310–1320).

DeVore, R., Hanin, B., & Petrova, G. (2021). Neural network approximation. *Acta Numerica*, *30*, 327–444. https://doi.org/10.1017/s0962492921000052

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. Retrieved from http://www.deeplearningbook.org

Groetsch, C. (2011). Linear inverse problems. In O. Scherzer (Ed.), *Handbook of mathematical methods in imaging* (pp. 3–41). Springer New York.

Han, X., Hu, Y., Foschini, L., Chinitz, L., Jankelson, L., & Ranganath, R. (2020). Deep learning models for electrocardiograms are susceptible to adversarial attack. *Nature Medicine*, *26*(3), 360–363. https://doi.org/10.1038/s41591-020-0791-x

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., & Madry, A. (2019). Adversarial examples are not bugs, they are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32).

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., et al. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Kurakin, A., Goodfellow, I. J., & Bengio, S. (2017). Adversarial machine learning at scale. In *International conference on learning representations*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. https://doi.org/10.1162/neco.1989.1.4.541

LeVeque, R. J., Waagan, K., González, F. I., Rim, D., & Lin, G. (2016). Generating random earthquake events for probabilistic tsunami hazard assessment. *Pure and Applied Geophysics*, *173*(12), 3671–3692. https://doi.org/10.1007/978-3-319-55480-8_2

Liu, C. M., Rim, D., Baraldi, R., & LeVeque, R. J. (2021). Comparison of machine learning approaches for tsunami forecasting from sparse observations. *Pure and Applied Geophysics*, *178*(12), 5129–5153. https://doi.org/10.1007/s00024-021-02841-9

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International conference on learning representations*.

Makinoshima, F., Oishi, Y., Yamazaki, T., Furumura, T., & Imamura, F. (2021). Early forecasting of tsunami inundation from tsunami and geodetic observation data with convolutional neural networks. *Nature Communications*, *12*(1), 2253. https://doi.org/10.1038/s41467-021-22348-0

Melgar, D. (2016). Cascadia FakeQuakes waveform data and scenario plots [Dataset]. *Journal of Geophysical Research*. https://doi.org/10.5281/zenodo.59943

Melgar, D. (2020). MudPy software. https://doi.org/10.5281/zenodo.3703200

Melgar, D., Crowell, B. W., Melbourne, T. I., Szeliga, W., Santillan, M., & Scrivner, C. (2020). Noise characteristics of operational real-time high-rate GNSS positions in a large aperture network. *Journal of Geophysical Research: Solid Earth*, *125*(7), e2019JB019197. https://doi.org/10.1029/2019jb019197

Melgar, D., LeVeque, R. J., Dreger, D. S., & Allen, R. M. (2016). Kinematic rupture scenarios and synthetic displacement data: An example application to the Cascadia Subduction Zone. *Journal of Geophysical Research: Solid Earth*, *121*(9), 6658–6674. https://doi.org/10.1002/2016jb013314

Miller, J. P., Taori, R., Raghunathan, A., Sagawa, S., Koh, P. W., Shankar, V., et al. (2021). Accuracy on the line: On the strong correlation between out-of-distribution and in-distribution generalization. In *Proceedings of the 38th international conference on machine learning* (Vol. 139, pp. 7721–7735).

Mulia, I. E., Ueda, N., Miyoshi, T., Gusman, A. R., & Satake, K. (2022). Machine learning-based tsunami inundation prediction derived from offshore observations. *Nature Communications*, *13*(1), 5489. https://doi.org/10.1038/s41467-022-33253-5

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems* (Vol. 32).

Rim, D., Baraldi, R., Liu, C. M., LeVeque, R. J., & Terada, K. (2022). Tsunami early warning from global navigation satellite system data using convolutional neural networks. *Geophysical Research Letters*, *49*(20), e2022GL099511. https://doi.org/10.1029/2022gl099511

Rim, D., & Suri, S. (2024). Code repository. https://doi.org/10.5281/zenodo.12663552

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., et al. (2019). Adversarial training for free. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32).

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. In *International conference on learning representations*.

Tramér, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., & McDaniel, P. (2018). Ensemble adversarial training: Attacks and defenses. In *International conference on learning representations*.

Trefethen, L. N., & Bau, D. (1997). *Numerical linear algebra*. Society of Industrial and Applied Mathematics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (Vol. 30).

Williamson, A., Melgar, D., & Rim, D. P. (2019). The effect of earthquake kinematics on tsunami propagation. *Journal of Geophysical Research: Solid Earth*, *124*(11), 11639–11650. https://doi.org/10.1029/2019jb017522

Zhang, H., Yu, Y., Jiao, J., Xing, E., Ghaoui, L. E., & Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 7472–7482).

Zhu, L., & Rivera, L. A. (2002). A note on the dynamic and static displacements from a point source in multilayered media. *Geophysical Journal International*, *148*(3), 619–627. https://doi.org/10.1046/j.1365-246X.2002.01610.x