Gated Complex Recurrent Neural Networks

Moritz Wolter and Anglea Yao
Institute of Computer Science
Bonn University
wolter@cs.uni-bonn.de

June 9, 2018

Abstract

Complex numbers have long been favoured for digital signal processing, yet complex representations rarely appear in deep learning architectures. RNNs, widely used to process time series and sequence information, could greatly benefit from complex representations. We present a novel complex gate recurrent cell. When used together with norm-preserving state transition matrices, our complex gated RNN exhibits excellent stability and convergence properties. We demonstrate competitive performance of our complex gated RNN on the synthetic memory and adding task, as well as on the real-world task of human motion prediction.

1 Introduction

Recurrent neural networks (RNNs) are widely used for processing time series and sequential information. The difficulties of training RNNs, especially when trying to learn long-term dependencies, are well-established, as RNNs are prone to vanishing and exploding gradients [2, 14, 33]. Heuristics developed to alleviate some of the optimization instabilities and learning difficulties include gradient clipping [11, 31], gating [5, 14], and using norm-preserving state transition matrices [1, 15, 17, 45].

Gating, as used in gated recurrent units (GRUs) [5] and long short-term memory (LSTM) networks [14], have become common-place in recurrent architectures. Gates facilitate the learning of longer term temporal relationships [14]. Furthermore, in the presence of noise in the input signal, gates can protect the cell state from undesired updates, thereby improving overall stability and convergence.

A matrix $\mathbf{W}$ is norm-preserving if its repeated multiplication with a vector leaves the vector norm unchanged, iti.$e.\|\mathbf{W}h\|_2 = \|h\|_2$. Norm-presrving state transition matrices are particularly interesting for RNNs because they preserve gradients over time [1], thereby preventing both the vanishing and exploding gradient problem. To be norm-preserving, state transition matrices need to be

either orthogonal or unitary[1]. Working with unitary matrices, in the complex domain, can be advantageous since unitary matrices are a larger and more expressive set than orthogonal matrices. Unitary matrices can spread their eigenvalues over the entire unit circle, whereas orthogonal matrices can only have eigenvalues of ± 1.

Complex numbers have long been favoured for signal processing [13, 26, 29]. A complex signal does not simply double the dimensionality of the signal. Instead, the representational richness of complex signals is rooted in its physical relevance and the mathematical theory of complex analysis. Complex arithmetic, and in particular multiplication, is different from its real counterpart and allows us to construct novel network architectures with several desirable properties. Despite networks being complex-valued, however, it is often necessary to work with real-valued cost functions and or existing real-valued network components. Mappings from $\mathbb{C} \to \mathbb{R}$ are therefore indispensable. Unfortunately such functions violate the Cauchy-Riemann equations and are not complex- differentiable in the traditional sense. We advocate the use of Wirtinger calculus [44] (also known as CR-calculus [23]), which makes it possible to define complex (partial) derivatives, even when working with non-holomorph or non-analytic functions.

Complex-valued representations have begun receiving some attention in the the deep learning community but they have been applied only to the most basic of architectures [1, 12, 40]. For recurrent networks, complex representations could gain more acceptance if they were shown to be compatible with more commonly used gated architectures and also competitive for real-world data. This is exactly our aim in this current work, where we propose a complex-valued gated recurrent network and show how it can easily be implemented with a standard deep learning library (TensorFlow) that offers limited support for complex calculus. Our contributions can be summarized as follows.

- We introduce a novel complex-gated recurrent unit; to the best of our knowledge, we are the first to explore such a structure using complex number representations.
- We compare experimentally the effects of a bounded versus unbounded non-linearity in recurrent networks, finding evidence countering the commonly held heuristic that only bounded non-linearities should be applied in RNNs. Unbounded non-linearities, in our case, perform better, but must be coupled with the stabilizing measure of using norm-preserving state transition matrices.
- Our complex gated network is stable and fast to train; it outperforms state-of-the-art with equal parameters on both synthetic tasks and the real-world application of predicting poses in human motion capture.

[1] Unitary matrices are the complex analogue of orthogonal matrices, iti.*e.* a complex matrix W is unitary if $W\overline{W} = \overline{W}W = I$, where $\overline{W}$ is its conjugate transpose and I is the identity matrix.

2 Related work

The current body of literature in deep learning focuses predominantly on real-valued neural networks. Theory for learning with complex-valued data, however, was established long before the breakthroughs of deep learning. This includes the development of complex non-linearities and activation functions [8, 19, 41], the computation of complex gradients and Hessians [42, 48], and complex backpropagation [3, 4, 7, 10, 20, 25].

Complex-valued representations were first used in deep networks to model phase dependencies for more biologically plausible neurons [35] and to augment the memory of LSTMs [6]. Recently, complex-valued CNNs have been proposed as an alternative for classifying natural images [12, 40] and inverse mapping of MRI signals [43]. Complex CNNs were found to be competitive or better than state-of-the-art [40] and significantly less prone to over-fitting [12].

For working with temporal sequences, complex-valued RNNs have also been explored [1, 15, 18, 45], though interest in complex representations stems from improved learning stability. In [1], norm-preserving state transition matrices are used to prevent vanishing and exploding gradients. Since it is difficult to parameterize real-valued orthogonal weights, [1] recommends shifting to the complex domain, resulting in a unitary RNN (uRNN). The weights of the uRNN in [1], for computational efficiency, are constructed as a product of component unitary matrices. As such, they span only a reduced subset of unitary matrices and do not have the expressiveness of the full set. Alternative methods of parameterizing the unitary matrices have been explored [15, 18, 45]. Our proposed complex gated RNN builds on these works in that we also use unitary state transition matrices. In particular, we adopt the parameterization of [45] in which weights are parameterized by full-dimensional unitary matrices, though any of the other parameterizations [1, 15, 18] can also be substituted.

3 Preliminaries

We represent a complex number $z \in \mathbb{C}$ as $z = x + ib$, where $x = \Re(z)$ and $y = \Im(z)$ are the real and imaginary parts respectively. The complex conjugate of z is $\bar{z} = x - iy$. In polar coordinates, z can be expressed as $z = |z|e^{i\theta_z}$, where $|z|$ and θ are the magnitude and phase respectively and $\theta_z = \text{atan2}(x, y)$. Note that $z_1 \cdot z_2 = |z_1||z_2|e^{i(\theta_1+\theta_2)}$, $z_1 + z_2 = x_1 + x_2 + i(y_1 + y_2)$ and $s \cdot z = s \cdot re^{i\theta}, s \in \mathbb{R}$. The expression $s \cdot z$ scales z's magnitude, while leaving the phase intact.

3.1 Complex Gradients

A complex-valued function $f : \mathbb{C} \to \mathbb{C}$ can be expressed as $f(z) = u(x, y) + iv(x, y)$ where $u(\cdot, \cdot)$ and $v(\cdot, \cdot)$ are two real-valued functions. The complex derivative of $f(z)$, or the $\mathbb{C}$-derivative, is defined if and only if f is *holomorph*. In such a case, the partial derivatives of u and v must not only exist but also satisfy the Cauchy-Riemann equations, where $\partial u/\partial x = \partial v/\partial y$ and $\partial v/\partial x = -\partial u/\partial y$.

Strict holomorphy can be overly stringent for deep learning purposes. In fact, Liouville's theorem [27] states that the only complex function which is both holomorph and bounded is a constant function. This implies that for complex (activation) functions, one must trade off either boundedness or differentiability. One can forgo holomorphy and still leverage the theoretical framework of Wirtinger or CR-calculus [23, 29] to work separately with the $\mathbb{R}$- and $\overline{\mathbb{R}}$-derivatives[2]

$$\mathbb{R}\text{-derivative} \triangleq \frac{\partial f}{\partial z}|_{z=\text{const}} \frac{1}{2}(\frac{\partial f}{\partial x} - i\frac{\partial f}{\partial y}),\ \overline{\mathbb{R}}\text{-derivative} \triangleq \frac{\partial f}{\partial \bar{z}}|_{z=\text{const}} \frac{1}{2}(\frac{\partial f}{\partial x} + i\frac{\partial f}{\partial y}). \tag{1}$$

Based on these derivatives, one can define the chain rule for a function $g(f(z))$ as follows:

$$\frac{\partial g(f(z))}{\partial z} = \frac{\partial g}{\partial f}\frac{\partial f}{\partial z} + \frac{\partial g}{\partial \bar{f}}\frac{\partial \bar{f}}{\partial z} \qquad \text{where} \qquad \bar{f} = u(x,y) - iv(x,y). \tag{2}$$

Since mappings from $\mathbb{C} \to \mathbb{R}$ can generally be expressed in terms of the complex variable z and its conjugate $\bar{z}$, the Wirtinger-Caclulus allows us to formulate and theoretically understand the gradient of real-valued loss functions in an easy yet principled way.

3.2 A Split Complex Approach

We work with a split-complex approach, where we apply real-valued non-linear activations to process the real and imaginary parts of the complex number separately. This makes it convenient for implementation purposes, since standard deep learning libraries are not designed to work with complex representations natively. Instead, we can simply keep track of complex numbers as two separate real-valued components. Split complex activation functions process either the magnitude and phase, or the real and imaginary components separately with two real-valued nonlinear functions and then recombine the two into a new complex quantity,

While some may argue this takes away from the utility of having complex representations, we prefer this to fully complex functions. Fully complex nonlinearities do exist and may seem favourable (as argued by [40], one would then only need to keep track of two derivatives) but due to Liouville's theorem, we must forgo boundedness and then deal with forward pass instablities.

[2]For holomorph functions the $\overline{\mathbb{R}}$-derivative is zero and the $\mathbb{C}$- derivative is equal to the $\mathbb{R}$-derivative.

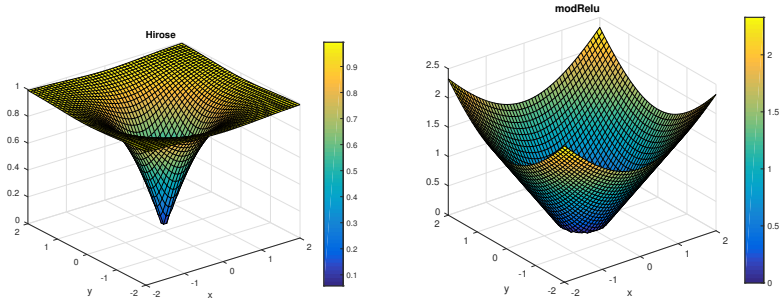


Figure 1: Surface plots of the magnitude of the Hirose and modRelu activation functions, the modRelu uses b=-0.5.

4 Complex Gated RNNs

4.1 Basic Complex RNN Formulation

Without any assumptions on real versus complex representations, we define a basic RNN as follows:

$$\mathbf{z}_t = \mathbf{W}\mathbf{h}_{t-1} + \mathbf{V}\mathbf{x}_t + \mathbf{b} \tag{3}$$

$$\mathbf{h}_t = f_a(\mathbf{z}_t) \tag{4}$$

where $\mathbf{x}_t$ and $\mathbf{h}_t$ represent the input and hidden unit vectors at time t. f_a is a point-wise non-linear activation function, and W and V are the hidden and input state transition matrices respectively. In working with complex networks, $\mathbf{x}_t \in \mathbb{C}^{n_h \times 1}$, $\mathbf{h}_t \in \mathbb{C}^{n_h \times 1}$, $\mathbf{W} \in \mathbb{C}^{n_h \times n_h}$ and $\mathbf{V} \in \mathbb{C}^{n_h \times n_{in}}$, where n_x and n_h are the dimensionalities of the input and hidden states respectively.

4.2 Complex Non-linear Activation Functions

Choosing a non-linear activation function f_a for complex networks can be non-trivial. Liouville's theorem [27] stipulates the necessity of trading off boundedness for differentiability. Though holomorph non-linearities using transcendental functions have also been explored in the literature [29], the presence of singularities makes them difficult to learn in a stable manner. Instead, bounded non-holomorph non-linearities tend to be favoured [13, 29], where bounded real-valued non-linearities are applied to the split complex. This also parallels the convention of using (bounded) tanh non-linearities in real RNNs.

A common split is with respect to the magnitude and phase. This non-linearity was popularized by Hirose [13] and scales the magnitude by a factor m^2 before passing it through a tanh:

$$f_{\text{Hirose}}(z) = \tanh\left(\frac{|z|}{m^2}\right) e^{-i\cdot\theta_z} = \tanh\left(\frac{|z|}{m^2}\right) \frac{z}{|z|}. \tag{5}$$

In other areas of deep learning, the rectified linear unit (ReLU) is now the go-to choice as a non-linearity. In comparison to sigmoid / tanh activations, they are computationally cheap, and have been shown to not only expedite convergence [24] but also perform better [32, 28, 47]. However, there is no direct extension into the complex domain, and as such, modified versions have been proposed [12, 43], with the most popular being the modReLU [1]. The modReLU is a variation of the Hirose non-linearity, where the tanh on the magnitude is replaced with a ReLU:

$$f_{\text{modReLU}}(z) = \text{ReLU}(|z| + b)e^{-i\cdot\theta_z} = \text{ReLU}(|z| + b)\frac{z}{|z|}, \tag{6}$$

and b denotes an offset parameter.

4.3 $\mathbb{R} \to \mathbb{C}$ input and $\mathbb{C} \to \mathbb{R}$ output mappings

While several time series problems are inherently complex, especially when considering their Fourier representations, the majority of benchmark problems in machine learning are still only defined in the real number domain. However, one can still solve these problems with complex representations, since a real z has simply a zero imaginary component, iti.$e.\Im(z) = 0$ and $z = x + i \cdot 0$.

To map the complex state $\mathbf{h}$ into a real output $\mathbf{o}_r$, we use a linear combination of the real and imaginary components, similar to [1], with $\mathbf{W}_o$ and $\mathbf{b}_o$ as weights and offset:

$$\mathbf{o}_r = \mathbf{W}_o[\Re(\mathbf{h})\ \Im(\mathbf{h})] + \mathbf{b}_o. \tag{7}$$

4.4 Optimization on the Stiefel Manifold for Norm Preservation

In [1], it was proven that a unitary [3] $\mathbf{W}$ would prevent vanishing and exploding gradients of the cost function C with respect to h_t, since the gradient magnitude is bounded. However, this proof hinged on the critical assumption that the derivative of f_a is also unity. This assumption is valid if the pre-activations are real and one chooses the ReLU as the non-linearity. For complex pre-activations, however, this is no longer a valid assumption. Neither the Hirose non-linearity (Eq. 5) nor the modReLU (Eq. 6) can guarantee stability (despite the suggestion otherwise in the original proof [1]).

Even though it is not possible to guarantee stability, we strongly advocate using norm-preserving state transition matrices, since they do still have excellent stabilizing effects. This was proven experimentally in [1, 15, 45] and we find similar evidence in our own experiments (see Fig. 2). Ensuring that $\mathbf{W}$ remains unitary during the optimization can be challenging, especially since the group of unitary matrices is not closed under addition. As such, it is not possible to

[3] Since $\mathbb{R} \subseteq \mathbb{C}$, we use the term unitary to refer to both real orthogonal and complex unitary matrices and make a distinction for clarity purposes only where necessary.

learn $\mathbf{W}$ with standard update-based gradient descent. Alternatively, one can learn $\mathbf{W}$ on the Stiefel manifold [45], with the $k+1$ update $\mathbf{W}_{k+1}$ given by [38]

$$\mathbf{W}_{k+1} = (\mathbf{I} + \frac{\lambda}{2}\mathbf{A}_k)^{-1}(\mathbf{I} - \frac{\lambda}{2}\mathbf{A}_k)\mathbf{W}_k \qquad \text{where} \qquad \mathbf{A} = \mathbf{W}\nabla_{\overline{\mathbf{W}}}F - \overline{\mathbf{W}}\nabla_{\mathbf{W}}F \tag{8}$$

and where λ is the learning rate, $\mathbf{I}$ the identity matrix, and F the cost function.

4.5 Complex-Valued Gating Units

In keeping with the spirit that gates determine the amount of a signal to pass, we construct a complex gate as a $\mathbb{C}^{n_h \times n_h} \to \mathbb{R}^{n_h \times 1}$ mapping. Like in real gated RNNs, the gate is applied as an element-wise product, iti.e.$\mathbf{g} \odot \mathbf{h} = \mathbf{g} \odot |\mathbf{h}|e^{i\theta_h}$. In our complex case, this type of operation results in an element-wise scaling of the hidden state's magnitude. When the gate is 0, it completely reset a signal, whereas if it is 1, then it ensures that the signal is passed entirely. We introduce our gates into the RNN in a similar fashion as the classic GRU [5]:

$$\widetilde{\mathbf{z}}_t = \mathbf{W}(\mathbf{g}_r \odot \mathbf{h}_{t-1}) + \mathbf{V}\mathbf{x}_t + \mathbf{b} \tag{9}$$

$$\mathbf{h}_t = \mathbf{g}_z \odot f_a(\widetilde{\mathbf{z}}_t) + (1 - \mathbf{g}_z) \odot \mathbf{h}_{t-1}, \tag{10}$$

where $\mathbf{g}_r$ and $\mathbf{g}_z$ represent reset and update gates respectively and are defined with corresponding subscripts r and z as

$$\mathbf{g}_r = f_g(\mathbf{z}_r), \qquad \text{where} \qquad \mathbf{z}_r = \mathbf{W}_r\mathbf{h} + \mathbf{V}_r\mathbf{x}_t + \mathbf{b}_r, \tag{11}$$

$$\mathbf{g}_z = f_g(\mathbf{z}_z), \qquad \text{where} \qquad \mathbf{z}_z = \mathbf{W}_z\mathbf{h} + \mathbf{V}_z\mathbf{x}_t + \mathbf{b}_z. \tag{12}$$

Above, f_g denotes the gate activation, $\mathbf{W}_r \in \mathbb{C}^{n_h \times n_h}$ and $\mathbf{W}_z \in \mathbb{C}^{n_h \times n_h}$ denote state to state transition matrices, $\mathbf{V}_r \in \mathbb{C}^{n_h \times n_i}$ and $\mathbf{V}_z \in \mathbb{C}^{n_h \times n_i}$ the input to state transition matrices, and $\mathbf{b}_r \in \mathbb{C}^{n_h}$ and $\mathbf{b}_z \in \mathbb{C}^{n_h}$ the biases. f_g is a non-linear gate activation function. We choose for f_g a sigmoid product:

$$f_{\text{prod}}(\mathbf{z}) = \sigma(\Re(\mathbf{z})) \cdot \sigma(\Im(\mathbf{z})). \tag{13}$$

This formulation is active in the first quadrant, where both real and imaginary part are positive.

As mentioned previously, even with unitary state transition matrices, this type of gating is not mathematically guaranteed to be stable. However, the effects of vanishing gradients are mitigated by the fact that the derivatives are distributed over a sum [14, 5]. Exploding gradients are clipped.

5 Experimentation

5.1 Synthetic Tasks: Memory and Adding Problem

We test our complex gated RNN on two benchmark synthetic tasks: the memory problem and the adding problem [14]. These problems are designed especially to

challenge RNNs, and require the networks to store information over time scales on the order of hundreds of time steps.

In the **Memory Problem**, the RNN should remember n inputs symbols over a time period of length $T + 2n$ based on a dictionary set $\{s^1, s^2, ..., s^n, s^b, s^d\}$, where s^1 to s^n are symbols to memorize and s^b and s^i are blank and delimiter symbols respectively. The input sequence, of length $T + 2n$, is composed of n symbols drawn randomly with replacement from $\{s^1, ..., s^n\}$, followed by $T - 1$ repetitions of s^b, s^d, and another n repetitions of s^b. The objective of the RNN, after being presented with the initial n symbols, is to generate an output sequence of length $T + S$, with T repetitions of s^b, and upon seeing s^d, recall the original n input symbols. A network without memory would output s^b and once presented with s^d, randomly predict any of the original n symbols; this results in a categorical cross entropy of ${}^{(n\,+\,1\log(8))}/_{(T\,+\,2(n\,+\,2))}$.

In the **Adding Problem**, two sequences of length T are given as input, where the first sequence consists of numbers randomly sampled from $\mathcal{U}[0, 1]$[4], while the second is an indicator sequence of all $0's$ and exactly two $1's$, with the first 1 placed randomly in the first half of the sequence and the second 1 randomly in the second half. The objective of the RNN is to predict the sum of the two entries of the first input sequence once the second 1 is presented in the indicator input sequence. A naive baseline would predict 1 at every time step, regardless of the input indicator sequence's value; this produces an mean squared error (MSE) of 0.167, iti.*e.*the variance of the sum of two independent uniform distributions.

In our experiments, we choose $n = 8$ and $T = 250$ for the memory problem and $T = 250$ for the adding problem. Previous works [1, 15, 45] have found that gating is not helpful for the memory problem, but can greatly help with the adding problem since it is able to ignore non-relevant inputs.

5.2 Real-World Task: Human Motion Prediction

We also apply our complex gated RNN to the real-world task of human motion prediction, iti.*e.* predicting likely future $3D$ poses of a person given the past motion. This task is of interest to diverse areas of research, including 3D tracking in computer vision [37, 46], motion synthesis for graphics [22, 36] as well as pose and action predictions for assistive robotics [21]. We follow the same experimental setting as [30], working specifically with the walking sequences of the Human 3.6M dataset [16]. For training, we use six of the seven actors and test on actor five. We use the pre-processed data of [39], which converts the motion capture into exponential map representations of each joint. Based on an input sequence of body poses from 50 frames, the future 25 frames are predicted, with error measured by the difference in Euler angles with respect to the ground truth poses.

[4]Note that this is a variant of [14]'s original adding problem, which draws numbers from $\mathcal{U}[-1, 1]$ and used three indicators $\{-1, 0, 1\}$. Our variant is consistent with state-of-the-art [1, 15, 45]

5.3 RNN Implementation Details

We work in Tensorflow, using RMS-prop to update standard weights and the multiplicative Stiefel-manifold update as described Eq. 8 for all unitary state transition matrices. The unitary state transition matrices are initialized the same as [1] as the product of component unitary matrices. All other weights are initialized using the uniform initialisation method recommended in [9], iti.$e.U[-l, l]$ with $l = \sqrt{6/(n_{in} + n_{out})}$, where n_{in} and n_{out} are the input and output dimensions of the tensor to be initialised. All biases are intialized as zero, with the exception of the gate biases $\mathbf{b}_r$ and $\mathbf{b}_z$, which are initialized at 4 to ensure fully open gates and linear behaviour at the start of training. All synthetic tasks are run for $2 \cdot 10^4$ iterations with a batch-size of 50 and a constant learning rate of 0.001 for both the RMS-Prop and the Stiefel-Manifold updates.

For the human motion prediction task, we use the same learning rate as [30], 0.005, though we reduce our state size to 512 (to be compatible with their 1024^5).

5.4 Results

5.4.1 Impact of Gating

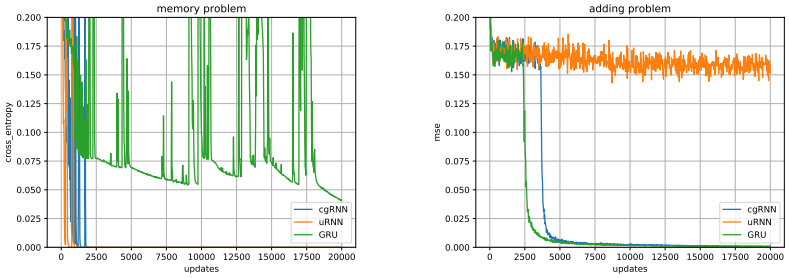
We first look at the impact that gating has on the synthetic tasks by comparing our complex gated RNN with the gateless uRNN from [1] (see Fig. 2). Both networks use complex representations and also unitary state transition matrices. As additional baselines, we also compare with TensorFlow's out-of-the-box GRU and LSTM. We choose the hidden state size n_h of each network to ensure that the resulting number of parameters is approximately equivalent (~ 165000)[6].

We find that our complex gated RNN successfully solves both the memory problem as well as the adding problem. On the memory problem, gating does not play a role. Instead, having norm-preserving weight matrices is key to ensure stability during the learning. Without it, both the GRU and the LSTM are highly unstable and while the GRU is eventually able to solve the problem, it takes many more iterations. The LSTM fails to solve the problem and stays at the baseline. Our gated complex RNN achieves very similar performance as the uRNN. This has to do with the fact that we initialize our gate bias term to be fully open, iti.$e.\mathbf{g}_r = \mathbf{1}$, $\mathbf{g}_z = \mathbf{1}$. Under this setting, the formulation is the same as the uRNN, and the $\mathbf{W}$ will dominate the cell's dynamics.

For the adding problem, previous works [1, 15, 45] have suggested that gates are beneficial. We confirm this result and speculate that the advantage comes from the gates shielding the network from the irrelevant inputs of the adding problem, hence the success of our complex gated RNN as well as the GRU and LSTM, but not the uRNN. We are very surprised to discover that the standard

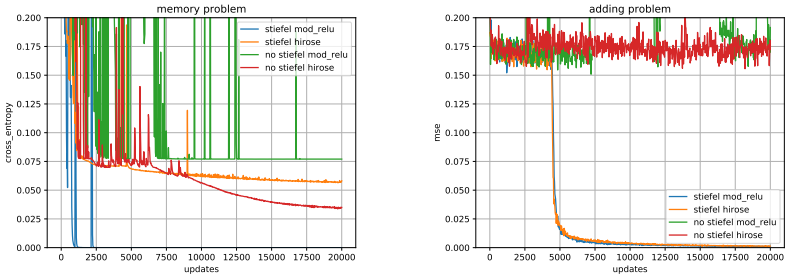
[5] Note that this is a larger reduction than necessary – the approximately equal state size is $\sqrt{\frac{1024^2}{2}}$

[6] The total number of LSTM weights depends on the output dimension, which is nine in the case of the memory problem and one for the adding problem, we adapted the LSTM state size to $n_h = 800$ for the adding problem to keep the total number of parameters at approximately 165k.



(a) (b)

Figure 2: Comparison of our complex gated RNN (cgRNN, blue, $n_h = 80$) with the unitary RNN [1](uRNN, orange, $n_h = 140$) and standard GRU [5](orange, $n_h = 112$) on the memory (a) and adding (b) problem for $T = 250$. The hidden state size n_h for each network are chosen so as to approximately match the number of parameters (approximately 42k parameters total). Figure best viewed in colour. On the memory problem, having norm-preserving state transition matrices is critical for stable learning, while on the adding problem, having gates is important.



(a) (b)

Figure 3: Comparison of non-linearities and norm preserving state transition matrices on the complex gated RNNs for the memory (a) and adding (b) problems for T=250. We find that the unbounded modReLU (see Eq. 6) performs best for both problems, but only if the state transition matrices are kept unitary. Without unitary state-transition matrices, the bounded Hirose non-linearity (see Eq. 5) performs better. We use $n_h = 80$ for all experiments.

GRU baseline, without any norm-preserving state transition matrices works very well on the adding problem; in fact marginally outperforms our complex gated RNN. However, we believe this result does not speak to the inferiority of complex representations, rather that the adding problem as a synthetic task is not able to leverage the advantages offered by the representation.

5.4.2 Non-Linearity Choice and Norm Preservation

We compare the bounded Hirose tanh non-linearity versus the unbounded modReLU (see Section 4.2) in our complex gated RNN in Figure 3 and discover a strong interaction effect from the norm-preservation. First, we find that optimiz-

ing on the Stiefel manifold to preserve norms for the state transition matrices significantly improves learning, regardless of the non-linearity. In both the memory and the adding problem, keeping the state transition matrices unitary ensures faster and smoother convergence of the learning curve.

Without unitary state transition matrices, the bounded tanh non-linearity, iti.*e.*the conventional choice is better than the unbounded modReLU. However, with unitary state transition matrices, the modReLU pulls ahead. Again, this result is consistent for both the memory and the adding problem. We speculate that the modReLU, like the ReLU in the real setting, is a better choice of non-linearity. The advantages afforded upon it by being unbounded, however, also makes it more sensitive to stability, which is why these advantages are present only when the state-transition matrices are kept unitary. Similar effects were observed in real RNNs in [34], in which batch normalization was required in order to learn a standard RNN with the ReLU non-linearity.

5.4.3 Choosing a gate activation

TODO...

5.4.4 Human Motion Prediction

Figure 4 shows the test-set Euler-errors of six different walking sequences during training. As suggested by [30] a test run is done every 1000 iterations. Our complex gated RNN consistently improves the prediction results in comparison to a GRU with more than twice the state size. All weights except for the unitary state transition matrices are updated using stochastic gradient descent. The final Euler error for the sequences are shown in the table in Figure 4; we observe that our complex gated RNN has about 10% lower error than the standard GRU.

6 Conclusion

In this paper, we have proposed a novel complex gated recurrent unit which we use together with unitary state transition matrices to form a stable and fast to train recurrent neural network. To enforce unitarity, we optimize the state transition matrices on the Stiefel manifold, which we show to work well with the modReLU. Our complex gated RNN achieves state-of-the-art performance on the adding problem while remaining competitive on the memory problem. We further demonstrate the applicability of our network on the real-world task of human motion prediction, in which we show superior performance with respect to a real-valued GRU. In future work, we intend to apply Fourier and Hilbert transform pre-processing on the temporal data to fully leverage the complex representations.

References

[1] Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *ICML*, 2016.

[2] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[3] Nevio Benvenuto and Francesco Piazza. On the complex backpropagation algorithm. *IEEE Transactions on Signal Processing*, 40(4):967–969, 1992.

[4] D.H. Brandwood. A complex gradient operator and its application in adaptive array theory, 1983.

[5] Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, October 2014.

[6] Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. Associative long short-term memory. In *ICML*, 2016.

[7] D. Franken. Complex digital networks: a sensitivity analysis based on the wirtinger calculus, 1997.

[8] George M Georgiou and Cris Koutsougeras. Complex domain backpropagation. *IEEE transactions on Circuits and systems II: analog and digital signal processing*, 39(5):330–334, 1992.

[9] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[10] Su Lee Goh and Danilo P Mandic. A complex-valued rtrl algorithm for recurrent neural networks. *Neural computation*, 16(12):2699–2713, 2004.

[11] Goodfellow. *Deep Learning*. MIT Press, 2017.

[12] Nitzan Guberman. On complex valued convolutional neural networks. Technical report, The Hebrew University of Jerusalem Israel, 2016.

[13] Akira Hirose. *Complex-valued neural networks: Advances and applications*, volume 18. John Wiley & Sons, 2013.

[14] Sepp Hochreiter and Juergen Schmidhuber. Long short term memory. *Neural Computation*, 1997.

[15] Stephanie L. Hyland and Gunnar Raetsch. Learning unitary operators with help from u(n). In *AAAI*, 2017.

[16] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.

[17] Li Jing, Caglar Gulcehre, John Peurifoy, Yichen Shen, Max Tegmark, Marin Solja, and Yoshua Bengio. Gated orthogonal recurrent units: On learning to forget. In *AAAI Workshops*, 2018.

[18] Li Jing, Yichen Shen, Tena Dubček, John Peurifoy, Scott Skirlo, Yann LeCun, Max Tegmark, and Marin Soljačić. Tunable efficient unitary neural networks (eunn) and their application to rnns. In *ICML*, 2017.

[19] Taehwan Kim and Tülay Adali. Complex backpropagation neural network using elementary transcendental activation functions. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, volume 2, pages 1281–1284. IEEE, 2001.

[20] Taehwan Kim and Tülay Adali. Fully complex multi-layer perceptron network for nonlinear signal processing. *Journal of VLSI signal processing systems for signal, image and video technology*, 32(1-2):29–43, 2002.

[21] Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):14–29, 2016.

[22] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 473–482. ACM, 2002.

[23] Ken Kreutz-Delgado. The complex gradient operator and the cr-calculus. *arXiv preprint arXiv:0906.4835*, 2009.

[24] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[25] Henry Leung and Simon Haykin. The complex backpropagation algorithm. *IEEE Transactions on Signal Processing*, 39(9):2101–2104, 1991.

[26] Hualiang Li and Tülay Adal. Complex-valued adaptive signal processing using nonlinear functions, 2008.

[27] Joseph Liouville. Leçons sur les fonctions doublement périodiques. *Journal für die reine und angewandte Mathematik / Zeitschriftenband (1879)*, 1879.

[28] A. Maas, A. Hannun, and A. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

[29] Danilo P Mandic and Vanessa Su Lee Goh. *Complex valued nonlinear adaptive filters: noncircularity, widely linear and neural models*, volume 59. John Wiley & Sons, 2009.

[30] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4674–4683. IEEE, 2017.

[31] Thomas Mikolov. Statistical language models based on neural networks. Technical report, Brno University of Technology, 2012.

[32] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *ICML*, 2010.

[33] Pascanu. On the difficulty of training recurrent neural networks. *Journal of Machine Learning Research*, 2013.

[34] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Improving speech recognition by revising gated recurrent units. In *INTERSPEECH*, 2017.

[35] David P Reichert and Thomas Serre. Neuronal synchrony in complex-valued deep networks. In *ICLR*, 2014.

[36] Alla Safonova and Jessica K Hodgins. Construction and optimal search of interpolated motion graphs. In *ACM Transactions on Graphics (TOG)*, volume 26, page 106. ACM, 2007.

[37] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1-2):4, 2010.

[38] Hemant Tagare. Notes on optimization on stiefel manifolds. Technical report, Yale University, 2011.

[39] Graham W. Taylor, Geoffrey E Hinton, and Sam T. Roweis. Modeling human motion using binary latent variables. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1345–1352. MIT Press, 2007.

[40] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, Joao Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Joshua Bengio, and Christopher J Pal. Deep complex networks. In *ICLR*, 2018.

[41] Aurelio Uncini, Lorenzo Vecci, Paolo Campolucci, and Francesco Piazza. Complex-valued neural networks with adaptive spline activation function for digital-radio-links nonlinear equalization. *IEEE Transactions on Signal Processing*, 47(2):505–514, 1999.

[42] A Van Den Bos. Complex gradient and hessian. *IEE Proceedings-Vision, Image and Signal Processing*, 141(6):380–382, 1994.

[43] Patrick Virtue, Stella X. Yu, and Michael Lustig. Better than real: Complex-valued neural nets for mri fingerprinting. 2017.

[44] W. Wirtinger. Zur formalen theorie der funktionen von mehr komplexen veränderlichen, 1927.

[45] Scott Wisdom, Thomas Powers, John R. Hershey, Jonathan Le Roux, , and Les Atlas. Full-capacity unitary recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2016.

[46] Angela Yao, Juergen Gall, and Luc Van Gool. Coupled action recognition and pose estimation from multiple views. *International journal of computer vision*, 100(1):16–37, 2012.

[47] M. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, and G. E. Hinton. On rectified linear units for speech processing. In *ICASSP*, 2013.

[48] Hans Georg Zimmermann, Alexey Minin, and Victoria Kusherbaeva. Comparison of the complex valued and real valued neural networks trained with gradient descent and random search algorithms. In *ESANN*, 2011.

seed	cgRNN-error	GRU-error
0080	**1.13**	1.24
0160	**1.14**	1.30
0320	**1.19**	1.31
0400	**1.17**	1.34
0560	**1.23**	1.39
1000	**1.39**	1.51
average	**1.21**	1.35

(a)

(b)

Figure 4: Motion prediction Euler angle errors for the complex gated RNN (green) versus GRU (blue), where each line indicates a separate test sequence. The final error after $20,000$ iterations is shown in the adjacent table.