

# Fourth Midterm - AntisymmetricRNN an architecture for long term dependancies preservation!

Author : Vincenzo Gargano

## Introduction

AntisymmetricRNN propose a slight change in the usual architecture of RNNs and Gated architecture by treating them first by a dynamical sytem perspective and then leveraging on the stability of these system in order to introduce two main tools: using Antisymmetrical matrixes as building block for stable architectures per initialization and introduce way to stabilize the discretization with a new hyperparameter called Diffusion.

## Dynamical Systems review

As Steven Strogatz says in his book:

” Cite

*"Non linear systems are really difficult to solve analytically, linear systems are easy because they are decomposable, the idea is like those of signals that can be decomposed using Fourier Analysis, instead non linear systems are everywhere in nature, and hearing two beautiful songs don't get us twice the fun."*

Dynamical and Complex Systems, are described by a mathematical model called State-Space Model, there we think in terms of set of *state variables* whose values (at any instant of time) can be used to predict future evolution of this system.

The fundamental part is that these are systems that work on sequences of any kind and time is sequential: seems just the definition of RNNs.

## On convergence of dynamical systems

What we usually do in the context of non-linear system, in order to study some of properties of our interest, is to do linear approximation on a certain `point of equilibria`. So by converting our RNNs into ODEs and then analyzing the behaviour around that point we can gain a lot of insights about what the system's properties are, whether it is unstable or stable and so on. But what are we interested in? The answer is the *Jacobian Matrix* and its *eigenvalues*.

## Attractors and Hyperbolic attractors

Thinking of dissipation we can picture it as a loss of energy over time of our system, in this case *Attractors and stable points*.

Dissipative systems are characterized by the presence of attracting sets or manifolds of dimensionality lower than the state space. For a Manifold or **attractor** we can have stable trajectories, these represent the *equilibrium states* of a dynamical system, for more insights: Appendix (B)



Note

Where for *trajectory* we mean the direction "drawn" by our system through time, it's a function that maps a time dependent variable to the state assumed at a certain time, for each time  $t$ . We will see these trajectories later in the paper

Consider a point attractor whose nonlinear dynamic equations are linearized around the equilibrium state, let  $A$  be the Jacobian evaluated, the attractor is said to be `hyperbolic attractor` if the eigenvalues of the Jacobian have all absolute values less than 1. Hyperbolic attractors are particularly interesting for Lesson 14 - RNN Sequential Models and Gated RNN > Gradients problem discovered : **Vanishing Gradient**.

---

## AntisymmetricRNN

The paper presents this new architecture ***AntisymmetricRNN*** that is able to capture long-term dependencies using the stability property of its differential equation, as we discussed previously.

### Definition - Stability in RNN

They define like this: a **solution**  $h(t)$  with initial condition  $h(0)$  as *stable* if for any  $\epsilon > 0 \exists \delta > 0$  such that any other solution  $\tilde{h}(t)$  of the ODE with initial condition  $\tilde{h}(0)$  satisfy the fact to be **Uniformly stable**

We need uniformly stable network : that if a small perturbation let's say  $\delta$  on the initial state  $h(0)$ , the effect of this perturbation on the states that follows won't be bigger than a certain  $\epsilon$ . The eigenvalues  $\lambda$  of the Jacobian matrix will play a central role in stability, as we know from the Lyapunov Theorems see Appendix (A)

We have a stable solution of the ODE if:

$$\max_{i=1,2,..n} \text{Re}(\lambda_i(J(t))) \leq 0, \quad \forall t \geq 0$$

But as we discussed previously a stable system like this would results in a **Lossy system**, that brings again to unwanted consequences, so we must stay in the middle between instability and stability, we need a critical point.

This condition is called **critical criterion**, here the system preserve long-term dependancies of the inputs while being stable, this is the pivot, longer we maintain this property the more dependancies we can "store" in our RNN recurrent weights.

$$\max_{i=1,2,..n} \text{Re}(\lambda_i(J(t))) \approx 0, \quad \forall t \geq 0$$

## Stability and Trainability

Trainability of our nets are important, how do we connect it to the stability of it's ODE?

Since we're studying the **sensitivity** of a solution, how the solution changes with changes in initial conditions.

How we can see in the equation above by differentiating the ODE of our RNN

$$\frac{d}{dt} \left( \frac{\partial h(t)}{\partial h(0)} \right) = J(t) \frac{\partial h(t)}{\partial h(0)}$$

We're characterizing what happens trough time to the changes of the current state given changes in initial condition. This is equal to our  $J$  that represents the sensitivity on the current state  $h(t)$ , with respect to the previous one, by multiplying  $J$  with that change this allows us to se how the initial state will have effect and propagate through the RNN.

We'll call  $A(t) : \frac{\partial h(t)}{\partial h(0)}$  the  $J$  of an hidden state  $h_t$  with respect to the initial hidden state  $h_0$ . When the *critical criterion* is met, so  $\text{Re}(\Lambda(J)) \approx 0$ . And thus we get that that the magnitude of the  $A(t)$  is constant in time, this means that we have no exploding nor vanishing gradients. This is why we want a system remains **Marginally Stable**! Our

trajectories may oscillate or remain stationary. We want a **Neutral stability** to avoid the two scenarios we can also say to have a center manifold.

## The reason behind Vanishing Gradients

In the paper we can clearly see how this problem emerges: by assuming diagonalization of  $A$ .

$$A(t) = \exp(Jt) = P e^{\Lambda(J)t} P^{-1}$$

By *critical criterion* we have that if  $\Lambda(J)$  are near zero, the magnitude of the term  $A(t)$  stay marginally stable as we discussed, for positive we diverge and negative we have vanishing gradients.

## Antisymmetry is all you need

The thing is we want to design ODEs that satisfy that criterion. So we introduce the **Antisymmetric Matrix**: a square matrix whose transpose equals its negative so:

$$M^T = -M$$

The property of interest is that all eigenvalues of  $M$  are imaginary. So we have no real part! They are zero.

Then they present a discretization of that ODE, and call this : **AntisymmetricRNN**

$$h_t = h_{t-1} + \epsilon \tanh((W_h - W_h^T)h_{t-1} + V_h x_t + b_h)$$

Our  $\epsilon$  would represent the small step we do when approximating. Since antisymmetric matrix has less degrees of freedom we can use the triangular matrix, making parameter efficient this net.


## Diffusion as stabilizer

This along the introduction of Antisymmetric matrix is the at heart of this architecture and of course is an hyperparameter. Now we have a stable ODE, but discretization can be still unstable, so we need a stability condition on the discretization method.

$$\max_{i=1,2,\dots,n} |1 + \epsilon \lambda_i(J_t)| \leq 1$$

The ODE defined previously is incompatible with the stability condition of Euler method because the eigenvalues are all imaginary that quantity is greater than 1 and this make our AntisymmetricRNN unstable.

One way to fix this is adding *Diffusion* to our system, we subtract a small  $\gamma > 0$  from the diagonal elements of the transition matrix (), so we need to add this hyperparameter to ensure stability.

 Important

This diffusion hyperparameter can be seen as a "stabilizer" for the vector field but it's, that my intuition: by applying diffusion we **constrain** the vector field to be stable.

# Gating Mechanism

Gating is a crucial mechanism for RNNs, firstly these were introduced because of the CeC, the error and useless informations were never discarded, to the idea was to filter out the useless information with gates, idea is: use sigmoid to constraint in  $[0, 1]$  and then apply product wise multiplication to discard with respect to the output of sigmoid, these filters has to be learn. Gates are crucial for long term dependancies\* Ablation study cited in paper

How can we make hold the critical criterion here?

Well they propose adding gate  $z_t$  to control the flow of information into the hidden states

Defined

$$z_t = \sigma((W_h - W_h^T - \gamma I)h_{t-1} + V_zx_t + b_z)$$

and  $h_t$

$$h_t = h_{t-1} + \epsilon z_t \odot \tanh((W_h - W_h^T - \gamma I)h_{t-1} + V_hx_t + b_h)$$

So the  $z_t$  uses Sigmoid as activation and the hidden state used Tanh. This is how is defined the Input gate in usual LSTM architectures (and also update gate in GRUs)

$$I_t = \sigma(W_{Ih}h_{t-1} + W_{In}x_t + b_I)$$

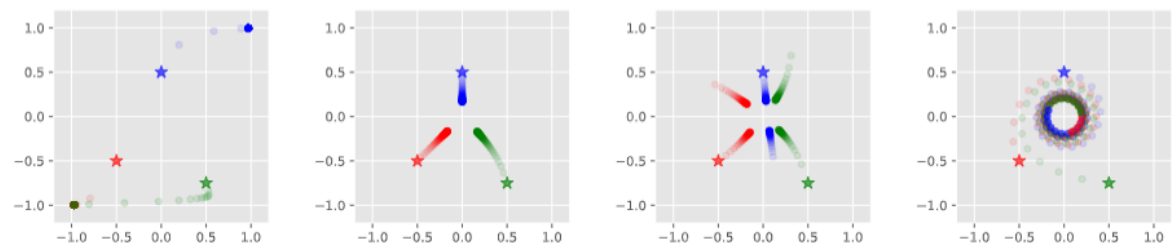
So we get a new shared weights matrix that is antisymmetric, model parameter will increase but not drastically. But the fact that matters is that the Jacobian of this gate would results in a diagonal matrix multiplied by our antisymmetric, the real part of the eigenvalues will be still close to zero due to this.

# Simulating RNNs dynamics

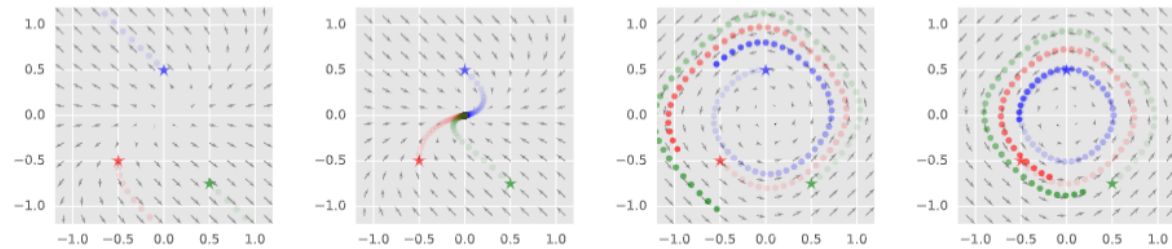
Equilibrium Type	Eigenvalue Type	Behaviour
Stable node	Real and negative	Stable and attractive
Stable focus	Complex conjugate with negative real parts	Stable and spiraling inward
Conservative	Purely imaginary	Conservative and oscillatory
Unstable node	Real and positive	Unstable and repulsive
Unstable focus	Complex conjugate with positive real parts	Unstable and spiraling outwards
Saddle point	Real with opposite signs	Saddle point

We can see in the Table 1 above different behaviours and then we can visualize them below

Ok so we've seen how theory works behind the dynamics of RNNs, let's see what happens if we use different matrixes for the weights. We can refer to the table we see above. These are the *trajectories* mentioned before!



(a) Vanilla RNN with a random weight matrix. (b) Vanilla RNN with an identity weight matrix. (c) Vanilla RNN with a random orthogonal weight matrix (seed = 0). (d) Vanilla RNN with a random orthogonal weight matrix (seed = 1).



(e) RNN with feedback with positive eigenvalues. (f) RNN with feedback with negative eigenvalues. (g) RNN with feedback with imaginary eigenvalues. (h) RNN with feedback with imaginary eigenvalues and diffusion.

We can observe in the (g) and (h) pictures, that without diffusion the equilibria tend to spiral outwards, instead by using the correct diffusion parameter we can manage to get a cycling pattern. By moving tangentially and increasing the distance from origin lead to instability, this is when  $\gamma$  comes in if we choose our diffusion and subtract it from diagonal ( $-\gamma I$ ) we get a slightly negative eigenvalue that tilts vectors field towards the origin so we have an "opposite" force that lead to constant distance from origin. From Appendix (B)

These are RNNs that lacks bias and input.

And this is the difference between vanilla and those with feedback.

$$\text{vanilla: } \mathbf{h}_t = \tanh(\mathbf{W}\mathbf{h}_{t-1}), \quad \text{feedback: } \mathbf{h}_t = \mathbf{h}_{t-1} + \epsilon \tanh(\mathbf{W}\mathbf{h}_{t-1}).$$

These are the matrix used for represent the four RNN with feedback (figures e-f)

$$\mathbf{W}_+ = \begin{pmatrix} 2 & -2 \\ 0 & 2 \end{pmatrix}, \mathbf{W}_- = \begin{pmatrix} -2 & 2 \\ 0 & -2 \end{pmatrix}, \mathbf{W}_0 = \begin{pmatrix} 0 & -2 \\ 2 & 0 \end{pmatrix}, \mathbf{W}_{\text{diff}} = \begin{pmatrix} -0.15 & -2 \\ 2 & -0.15 \end{pmatrix}.$$

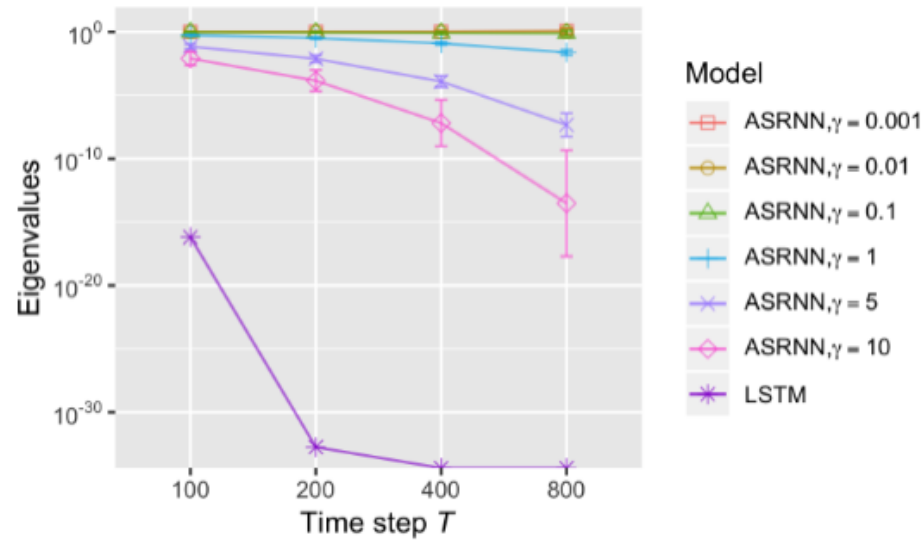
## Experiments

They tested the long term dependancies of these nets against the LSTM and other models, first experimenting in predicting MNIST digits by pixel by pixel outperforming old model having even less parameters to train, and using  $T = 784$  timesteps, then they tried a more difficult task with CIFAR-10,  $32 \times 32$  RGB images in 10 classes, initially with one pixel per channel (*input* :  $m = 3$ ) with  $T = 1024$  steps.

Surprisingly they tried to use an entire row each time of CIFAR-10, after 32 steps the salient information are taken but they used 968 more steps to create artificially longer dependancies from row to row. Still AntisymmetricalRNNs outperformed with less parameters LSTMs

## Is Exploding and Vanishing solved?

Well no, but these architectures indeed mitigate a lot this issue.



Here we can see that regular LSTM the *unitary eigenvalues* quickly fade to zero indicating vanishing gradient, while the AntisymmetricalRNNs are more robust to time steps maintaining the  $\mu$  close to 1 and  $\sigma$  close to zero. But if the values of the  $\gamma$  is too big we tend to get vanishing gradient as well.

## Conclusion and Further Works

Basically the problem posed by the paper boils down to find the right initialization for initial value problem such that the critical criterion is satisfied. In order to satisfy it we can use a special type of matrix that for eigenvalues has only the imaginary part. Another plus is that with this matrix we get less parameters because it has less degrees of freedom, so we improve and simplify an existing model.

These architectures are still limited by the fact that training is sequential, but recently seems that further improvements can be done (see [RWKV paper](#).) We can conclude by evidencing the importance of analyzing these architecture by the point of view of Dynamical System, by keeping these marginally stable we can be resistant to long term relationship in our data. Moreover if our data are far complex than MNIST for example in [this paper](#), we can see that for spatio-temporal data that can "shift over time", these suffers of problems like forgetting information over time, having an architecture that can model these data would be really useful.

## References

Nonlinear Dynamics and Chaos by Steven Strogatz.

## Appendix



## (A) Lyapunov's Theorems

Now the objective is determine stability, we could try all possible state space equation but the apporach is not doable, an elegant approach is founded by Lyapunov. We can leverage on a function called `Lyapunov Function`.

1. Theorem: The equilibrium state is **stable** if in a small neighborhood of  $\bar{x}$ , there exists a positive-definite function  $V(x)$  such that its derivative with respect to time is negative semidefinite.
2. The equilibrium state is **asymptotically stable** if in small neighborhood there exists a positive definite function  $V(x)$  such that the derivative is negative definite in that region.

The function  $V(x)$  that satisfies the requirements of these teorems is called Lyapunov function for that equilibrium state. Of course what we are intrested is to have *stabiity* rather than *asymptotically stability*. See Simulating RNNs dynamics table.

## (B) Divergence theorem

Having intuition about this is quite important and i think that visualization of our systems is what makes them really easy to understand the divergence represents how much a vector field "*spreads out*" or "*diverges*" from a point.

### Important

If the divergence  $\nabla \cdot F(x)$  (which is a scalar) is zero, the system is conservative, and if it negative the system is dissipative.

For example what happens if a RNNs is dissipative? In the worst case as also mentioned in the paper we could lose previously stored information during the forward step.