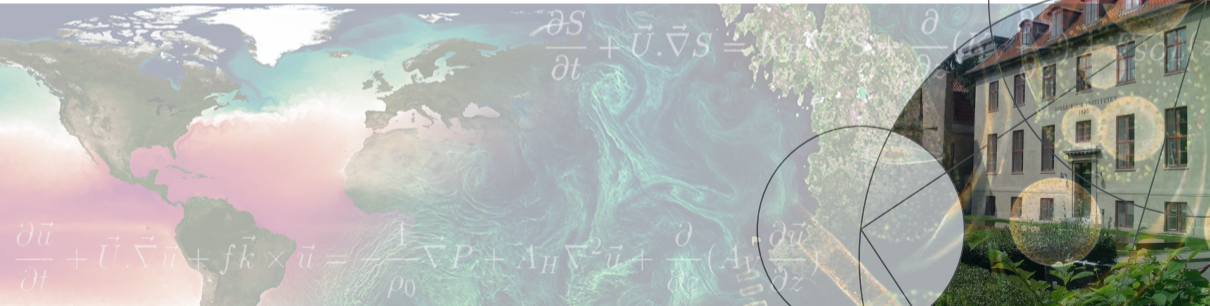




Faculty of Science

Echo State Networks and Anomaly Detection in Simulation Data

James Avery (avery@nbi.dk)
Niels Bohr Institute, University of Copenhagen



The Trouble with RNN



Difficulties in Training RNN

RNN are extremely versatile, but the high expressiveness comes at cost.

Under the right definition, RNN are:

Turing complete \implies universal (\implies training uncomputable).

In practice:

Restricted but very large expressiveness, with expensive but (usually) computable training.



Difficulties in Training RNN

Back-Propagation Through Time (BPTT) unrolls recurrent network graph (like unrolling a loop in a computer program), yielding static but very deep network.

Two problems arise:

- ① Optimizing becomes extremely resource intensive
- ② ...and can fail completely (due to e.g. vanishing gradients, bifurcations in the state space).



What are state space bifurcations?

Consider an RNN with just a single variable:

$$x_{t+1} = \tanh(wx_t + b)$$

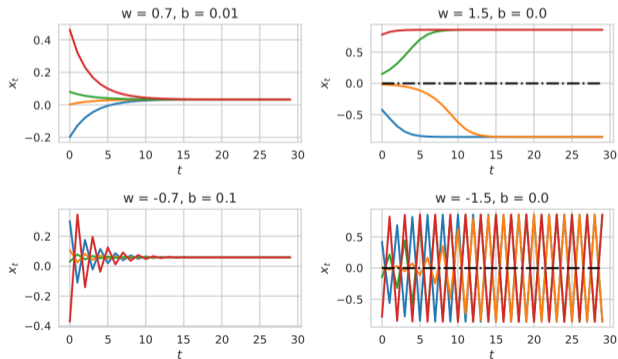


What are state space bifurcations?

Consider an RNN with just a single variable:

$$x_{t+1} = \tanh(wx_t + b)$$

Stable and unstable fixed-points (attractors and repellers):

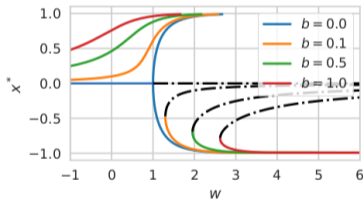
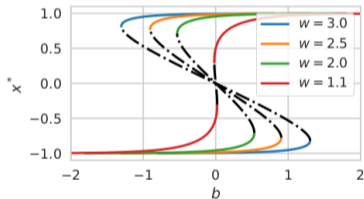


What are state space bifurcations?

Consider an RNN with just a single variable:

$$x_{t+1} = \tanh(wx_t + b)$$

Bifurcations happen when attractors split into two (or more):

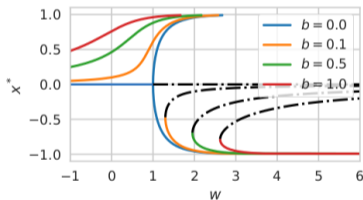
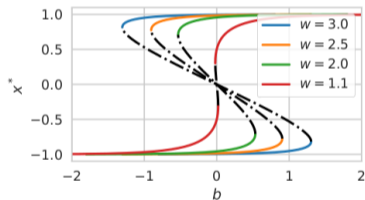


What are state space bifurcations?

Consider an RNN with just a single variable:

$$x_{t+1} = \tanh(wx_t + b)$$

Bifurcations happen when attractors split into two (or more):



Resulting huge gradients can destroy hundreds of learning steps in a single iteration.



Echo State Networks



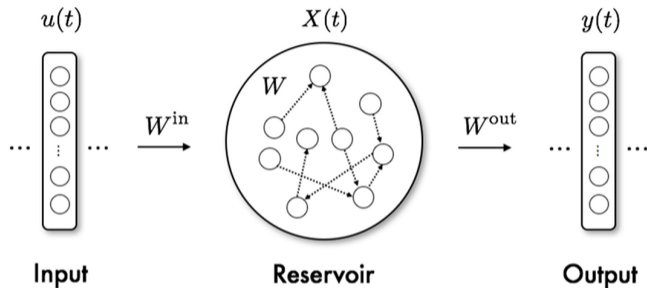
Comparison between LSTM and ESN

ESN are *much* weaker than general RNN, and weaker than LSTM, but are simple, extremely fast to train, and most importantly: stable.

	RNN	LSTM	ESN
Expressive Power	∞	Very strong	Medium
Vanishing Gradients	Yes	No	No
Many local minima	Yes	Yes	No
Training Bifurcations	Yes	Yes	No
Convergence can fail	Yes	Yes	Nothing to Converge
Training time	∞	Very Slow	Very Fast



How is it Done?



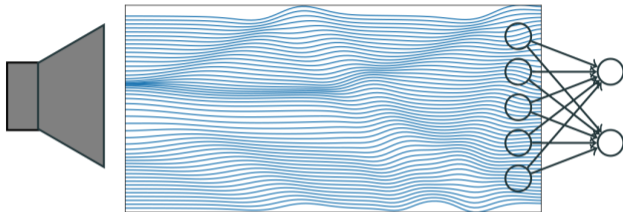
$$\vec{x}_{t+1} = \varphi(\mathbf{W}^{\text{in}} \vec{u}_t + \mathbf{W} \vec{x}_t)$$

$$\vec{y}_{t+1} = \psi(\mathbf{W}^{\text{out}} \vec{x}_{t+1})$$

Only output matrix \mathbf{W}^{out} is trained. If ψ is identity, training is just solving a least-squares problem!



An ESN Metaphor: Pattern Recognition in a Bucket

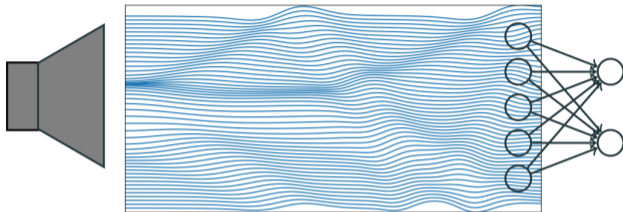


$$\vec{x}_{t+1} = \varphi (\mathbf{W}^{\text{in}} \vec{u}_t + \mathbf{W} \vec{x}_t)$$

$$\vec{y}_{t+1} = \psi (\mathbf{W}^{\text{out}} \vec{x}_{t+1})$$



An ESN Metaphor: Pattern Recognition in a Bucket



$$\vec{x}_{t+1} = \varphi(\mathbf{W}^{\text{in}} \vec{u}_t + \mathbf{W} \vec{x}_t)$$
$$\vec{y}_{t+1} = \psi(\mathbf{W}^{\text{out}} \vec{x}_{t+1})$$

Only output matrix \mathbf{W}^{out} is trained. If ψ is identity, training is just solving a least-squares problem!



Hidden matrix **W** must be sparse, but connected:

- ① Sparsity provides “locality” condition: Every hidden variable gets a “neighbourhood” of its neighbours, from which information can gradually propagate.
- ② Connectedness makes sure that information eventually reaches everywhere.
- ③ Bonus: Sparsity makes matrix-vector multiplication $\mathcal{O}(n)$ instead of $\mathcal{O}(n^2)$.



The *eigenvalues* of the hidden matrix \mathbf{W} control how memory attenuates.

In particular, we need to scale \mathbf{W} to have a desired *spectral radius* $\rho(\mathbf{W}) \simeq 1$, so that it neither **explodes** ($\rho \gg 1$) or **dies out** ($\rho \ll 1$).

In the “waves in a bucket” metaphor, ρ is like the viscosity of the fluid.



Performance

The main selling point of ESN is how efficiently they can be trained:

Instead of gradient descent (heavy, can fail to converge), training is a single deterministic least-squares calculation.

The predictions are: $\mathbf{y}_t = \mathbf{W}^{\text{out}}\mathbf{x}_t$, where \mathbf{x} is the input concatenated with the current state.



Performance

The main selling point of ESN is how efficiently they can be trained:

Instead of gradient descent (heavy, can fail to converge), training is a single deterministic least-squares calculation.

The predictions are: $\mathbf{y}_t = \mathbf{W}^{\text{out}}\mathbf{x}_t$, where \mathbf{x} is the input concatenated with the current state.

We can write $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_T]$, and the desired outputs as: $\mathbf{W}^{\text{out}}\mathbf{X} \simeq \mathbf{D}$.



Performance

The main selling point of ESN is how efficiently they can be trained:

Instead of gradient descent (heavy, can fail to converge), training is a single deterministic least-squares calculation.

The predictions are: $\mathbf{y}_t = \mathbf{W}^{\text{out}}\mathbf{x}_t$, where \mathbf{x} is the input concatenated with the current state.

We can write $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_T]$, and the desired outputs as: $\mathbf{W}^{\text{out}}\mathbf{X} \simeq \mathbf{D}$.

This can be optimized in a single step by solving the $N_{\text{hidden}} \times N_{\text{out}}$ least-squares problem

$$\mathbf{W}^{\text{out}} = \arg \min_{\mathbf{W}} \|\mathbf{W}\mathbf{X} - \mathbf{D}\|^2 + \beta^2 \|\mathbf{W}\|^2$$

This is so fast, that we can do online training!



An Example: A Chaotic Time Sequence

Mackey-Glass sequence:

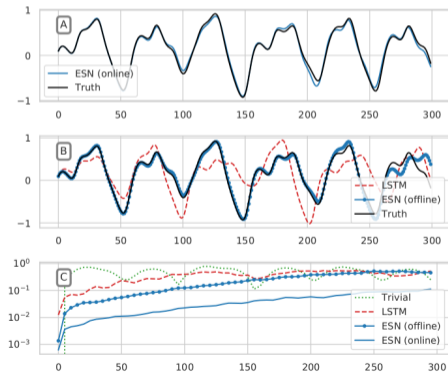
$$\frac{\partial y(t)}{\partial t} = \beta \frac{y(t - \tau)}{1 + y^n(t - \tau)} - \gamma y(t)$$



An Example: A Chaotic Time Sequence

Mackey-Glass sequence:

$$\frac{\partial y(t)}{\partial t} = \beta \frac{y(t - \tau)}{1 + y^n(t - \tau)} - \gamma y(t)$$



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:



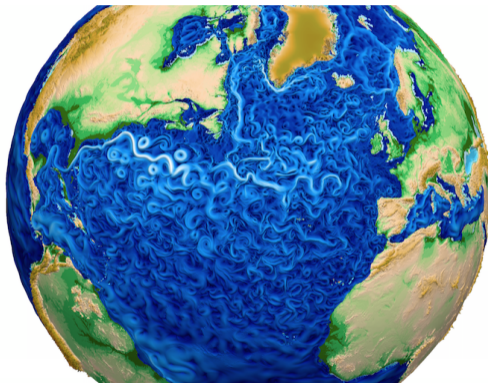
Challenge:

MSc Students: Niklas Heim, Jacob Ungar Felding



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:



Challenge: Enormous data sets, gigabytes per time step, terabytes in total

MSc Students: Niklas Heim, Jacob Ungar Felding



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:



Challenge: Enormous data sets, gigabytes per time step, terabytes in total.
High-dimensional; ESN only really work for 1D sequences.

MSc Students: Niklas Heim, Jacob Ungar Felding



Extending ESN to Finite-Difference Simulations



Extending ESN to Finite-Difference Simulations

Finite-difference simulations are:

- ① Simulation of fields with values on a regular grid, similar to *images*.
- ② Generally smooth or close to it: few, localized discontinuities, if any.
- ③ Spatially and temporally correlated: guided by differential equations that determine value from neighbours in time and space.



Extending ESN to Finite-Difference Simulations

Finite-difference simulations are:

- ① Simulation of fields with values on a regular grid, similar to *images*.
- ② Generally smooth or close to it: few, localized discontinuities, if any.
- ③ Spatially and temporally correlated: guided by differential equations that determine value from neighbours in time and space.

Let's exploit this structure!



Extending ESN to Finite-Difference Simulations

Recall: In standard ESN, we need to map input to a much higher-dimensional hidden space, where the recurrence happens.

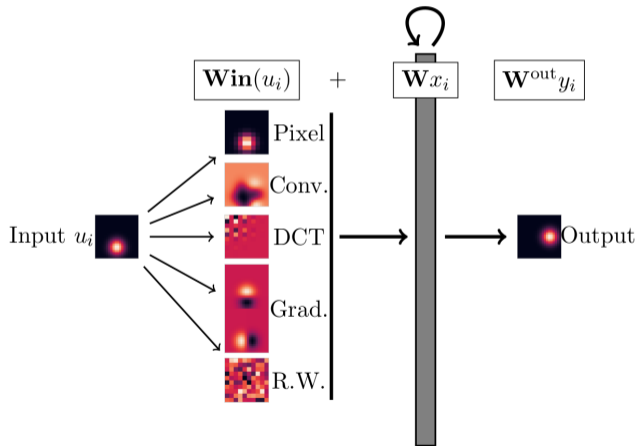
This map, \mathbf{W}^{in} is uniformly random, and throws away all information about spatial and temporal locality, and other interdependence between variables. Let's not do that!

Idea: Replace random map with one that preserve meaningful features.



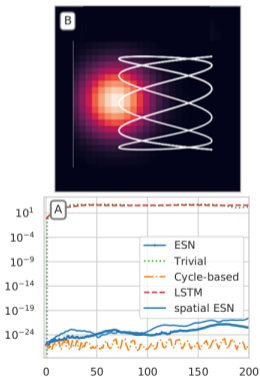
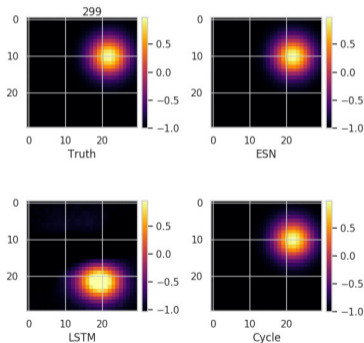
Extending ESN to Finite-Difference Simulations

We stack convolutions, derivatives, cosine transform. Other useful maps could be divergence, curl, turbulence, etc.



Video of Blob in Periodic Lissajous Pattern

$$(x, y) = (\cos(\alpha t), \sin(\beta t))$$

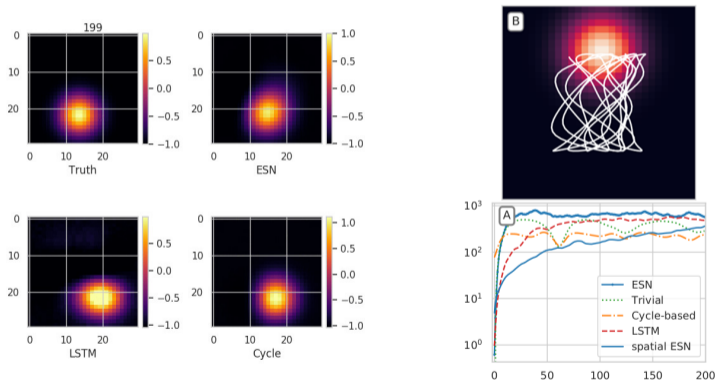


Trained on 2000 steps, predicting 300 into the future.



Video of Blob in Chaotic Mackey-Glass Pattern

$$(x, y) = (\text{mackey}(t), \sin(t))$$



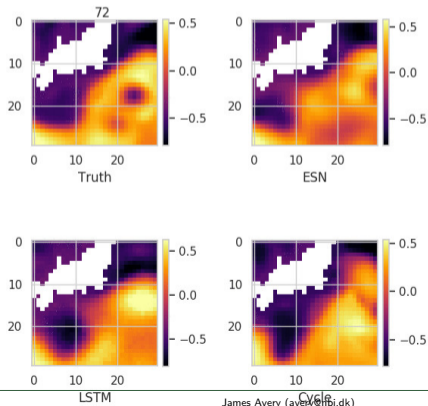
Trained on 2000 steps, predicting 300 into the future.



Kuroshio Ocean Current off the Coast of Japan

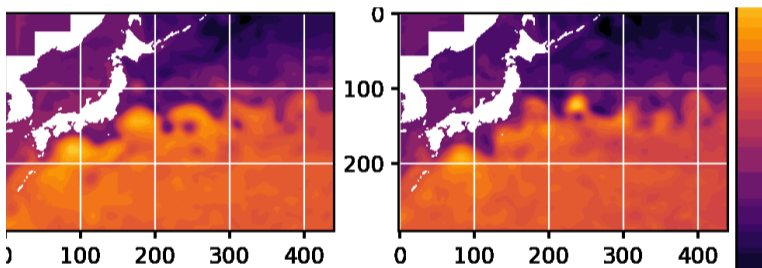
MSc thesis: Niklas Heim, *Automated Anomaly Detection in Chaotic Time Series*

(Best Danish CS Thesis in 2018 - won by a physicist! :))



Making Spatial ESN Scale to Cover the Earth

MSc thesis: Jacob, *A Scalable Spatially Aware ESN*



Trained on first 10 years of 5-day means (290×440 by 730 steps), predict 4 years (300 steps) into the future. Training time: 4 minutes!



Anomaly Detection



Anomaly Detection

Once we have an RNN that can “understand” a sequence well enough to predict it somewhat reliably, we can use it to automatically discover when it starts behaving strange.

Knowing that something weird is going on, even a little bit in advance, can save lives and resources. . .



Anomaly Detection

Once we have an RNN that can “understand” a sequence well enough to predict it somewhat reliably, we can use it to automatically discover when it starts behaving strange.

Knowing that something weird is going on, even a little bit in advance, can save lives and resources. . . everywhere!



Anomaly Detection

Once we have an RNN that can “understand” a sequence well enough to predict it somewhat reliably, we can use it to automatically discover when it starts behaving strange.

Knowing that something weird is going on, even a little bit in advance, can save lives and resources. . . everywhere!

E.g.: Detecting failure in nuclear reactors, engine break down, weather events, etc. before they happen.



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:

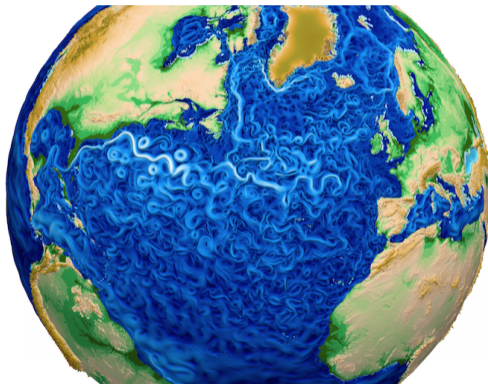


Challenge:



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:



Challenge: Enormous data sets, gigabytes per time step, terabytes in total.



Anomaly Detection in Climate Simulations

We'll look at discovering weird phenomena in *climate simulation data*:



Challenge: Enormous data sets, gigabytes per time step, terabytes in total.
High-dimensional; ESN only really work for 1D sequences.



Anomaly Detection from Predictions

Idea: Given a good method for prediction, we quantify normal behaviour as *how well we were able to predict it*.

Choose a window size T at the same scale as the anomalies we wish to discover. To evaluate the *normality* of step t , start at $t - T/2$, and predict T steps, until $t + T/2$.

Now compare $y_{t-T/2}, \dots, y_{t+T/2}$ to the real data. The integral of the error is used to calculate a *normality score*. If the error suddenly becomes much bigger than usual, we may have discovered an *anomaly*.



Example in 1D: Mackey-Glass

We use the chaotic equation from before

$$\frac{\partial y(t)}{\partial t} = \beta \frac{y(t - \tau)}{1 + y^n(t - \tau)} - \gamma y(t)$$

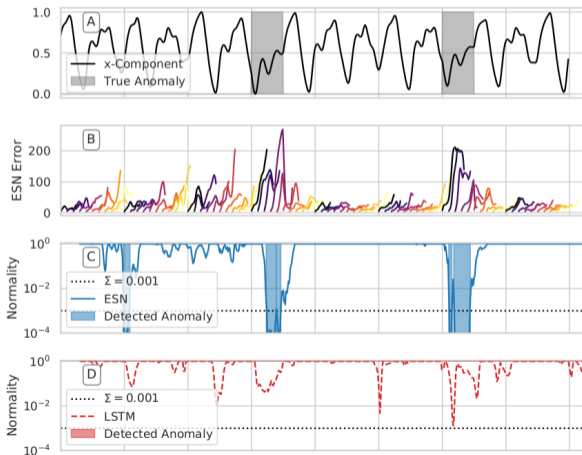
and train on 2000 steps with $\gamma = 0.10$.

In the anomaly-detection phase, we change γ to 0.13 for 50 steps at random steps.



Result:

The difference is imperceptible to humans, but we catch both instances with our ESN predictor.

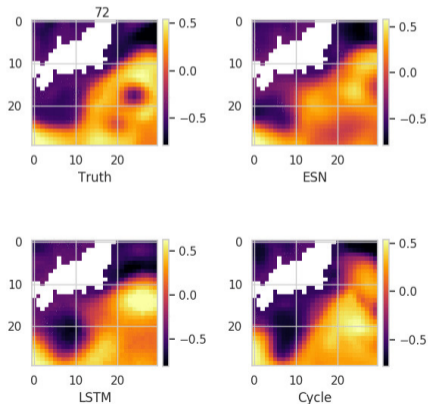


Finding Anomalies in Ocean Simulation Data



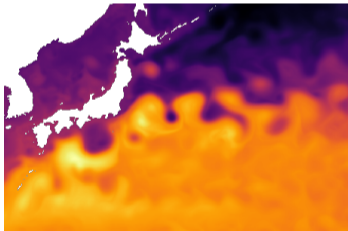
Kuroshio Ocean Current off the Coast of Japan

MSc thesis: Niklas Heim, *Automated Anomaly Detection in Chaotic Time Series* (Best Danish CS Thesis in 2018 - won by a physicist! :))



We trained on 10 years of simulation data, and then evaluated normality through 5 years. For each step, we predicted a year into the future, and calculated a normality score from the total error during the period.

The goal was to automatically discover a known ocean phenomenon: The Kuroshio (“Black Tide”).



Localizing an anomaly in space

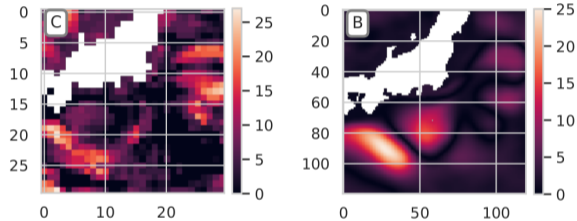
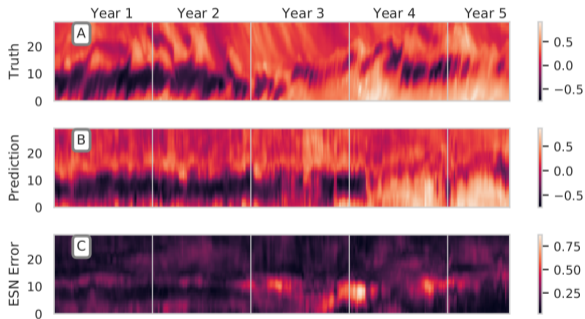


Figure: Left: a “heat map” of low normality scores, accumulated over the 5 year prediction period. Right: Known localization of the Kuroshio phenomenon.

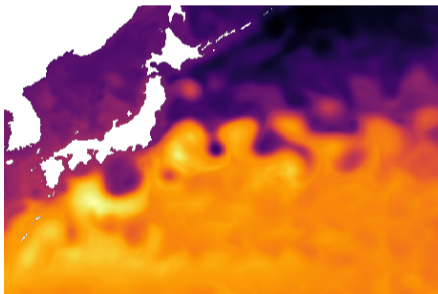


Localizing an anomaly in time



Making Spatial ESN Scale to Cover the Earth

MSc thesis: Jacob Ungar Felding, *Automated Anomaly Detection in Chaotic Time Series*



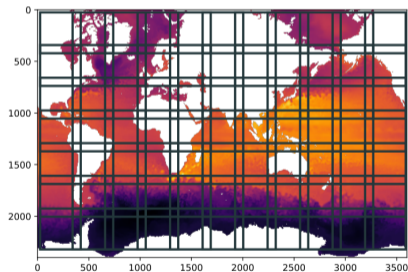
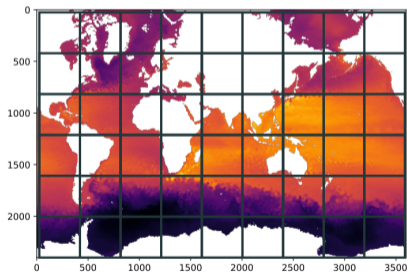
$290 \times 440 = 127600$ variables! Trained on first 10 years of 5-day means (730 steps), predict 4 years (300 steps) into the future. Training time: 4 minutes!



The Next Stage

Jacob's new development allow us to scale up the methods to large regions of the world.

Time to scour the (simulated) Seven Seas for new oceanographic phenomena!



Possibility for new MSc project shared between eScience and Team Ocean.

