Chaos

On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrast to VAR and DMD

Cite as: Chaos 31, 013108 (2021); https://doi.org/10.1063/5.0024890
Submitted: 14 August 2020 . Accepted: 30 November 2020 . Published Online: 04 January 2021

Erik Bollt

nature COMMUNICATIONS

ARTICLE

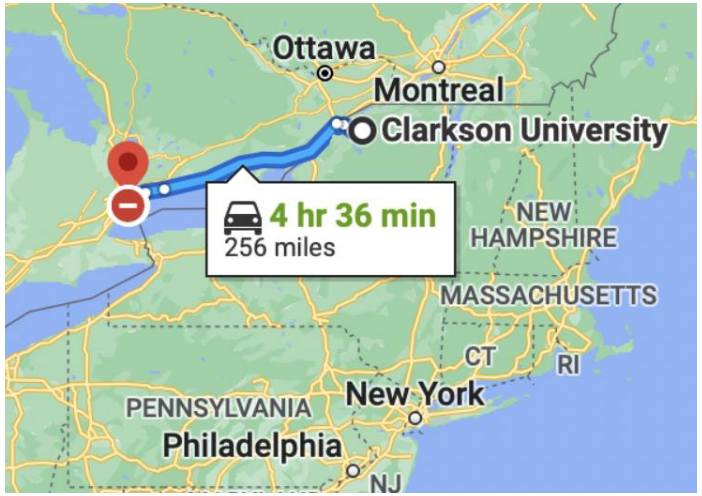
https://doi.org/10.1038/s41467-021-25801-2 OPEN

Next generation reservoir computing

Daniel J. Gauthier [1,2], Erik Bollt[3,4], Aaron Griffith [1] & Wendson A. S. Barbosa [1]

Clarkson UNIVERSITY

Erik Bollt



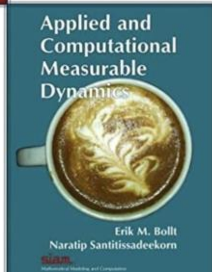
mathematics MDPI

; https://

Article

Randomized Projection Learning Method for Dynamic Mode Decomposition

Sudam Surasinghe [1,*,†,‡] and Erik M. Bollt [2,‡]



-**bolltem@clarkson.edu,**
-**http://www.clarkson.edu/~bolltem**

A Reservoir Computer, RC

Is a kind of neural network - for forecasting dynamical systems
but most of the (millions of) parameters are chosen randomly.

Clearly its cheap

Surprisingly - it actually works!

And surprisingly, it works really well.

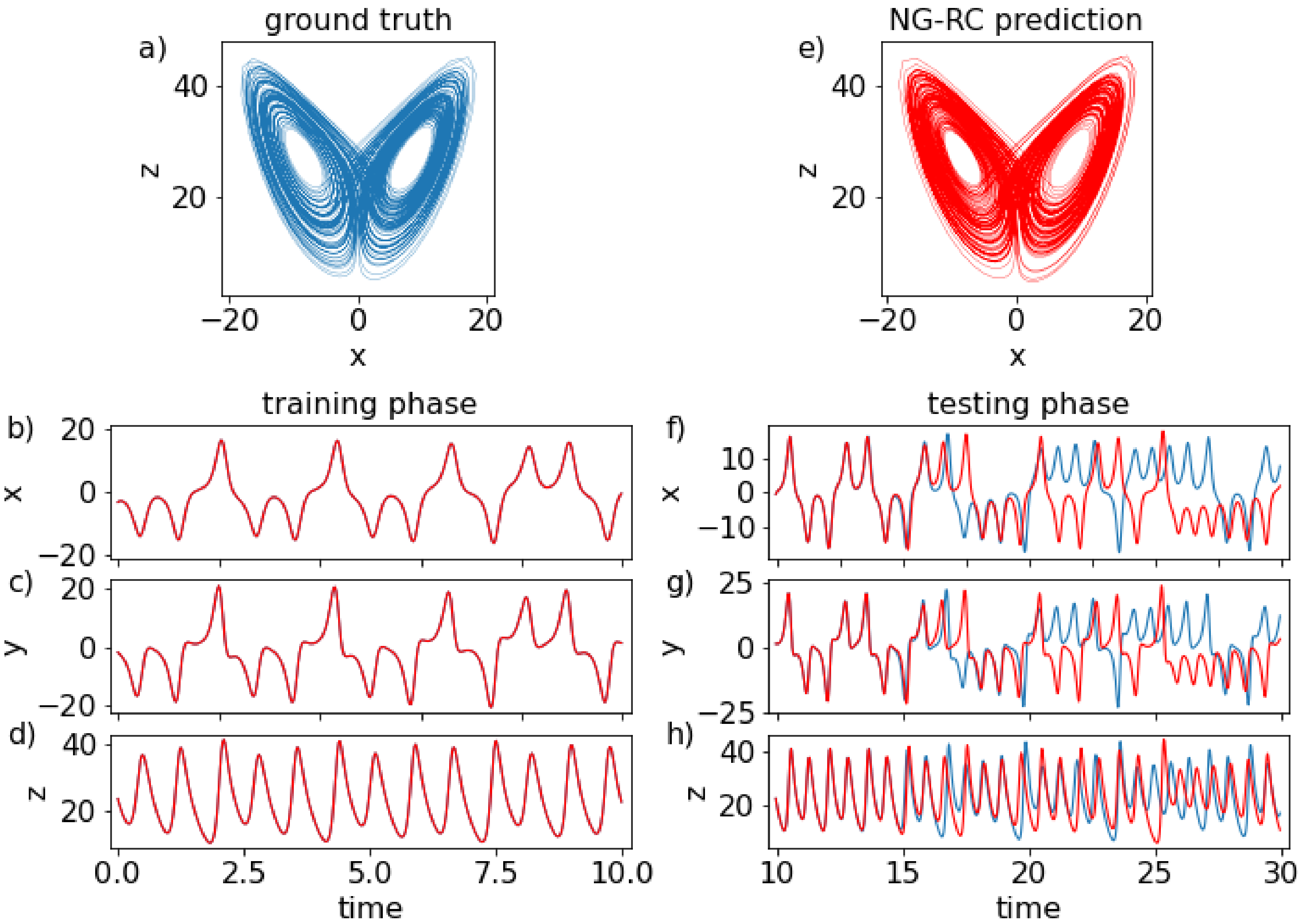
(This talk is not about neat things you can do with RC)
(This talk is about how/why/bridge-equivalent to something else)

ESN-Jaeger 2001, Jaeger-Haas, 2004.

Conclude: Works really really well – and drastically MUCH less data hungry

-linear RC with nonlinear readout = NVAR **AND this leads to** NG-RC

-VAR vs VMA which follows classic representation theorem by WOLD thm - also relates to DMD-Koopman

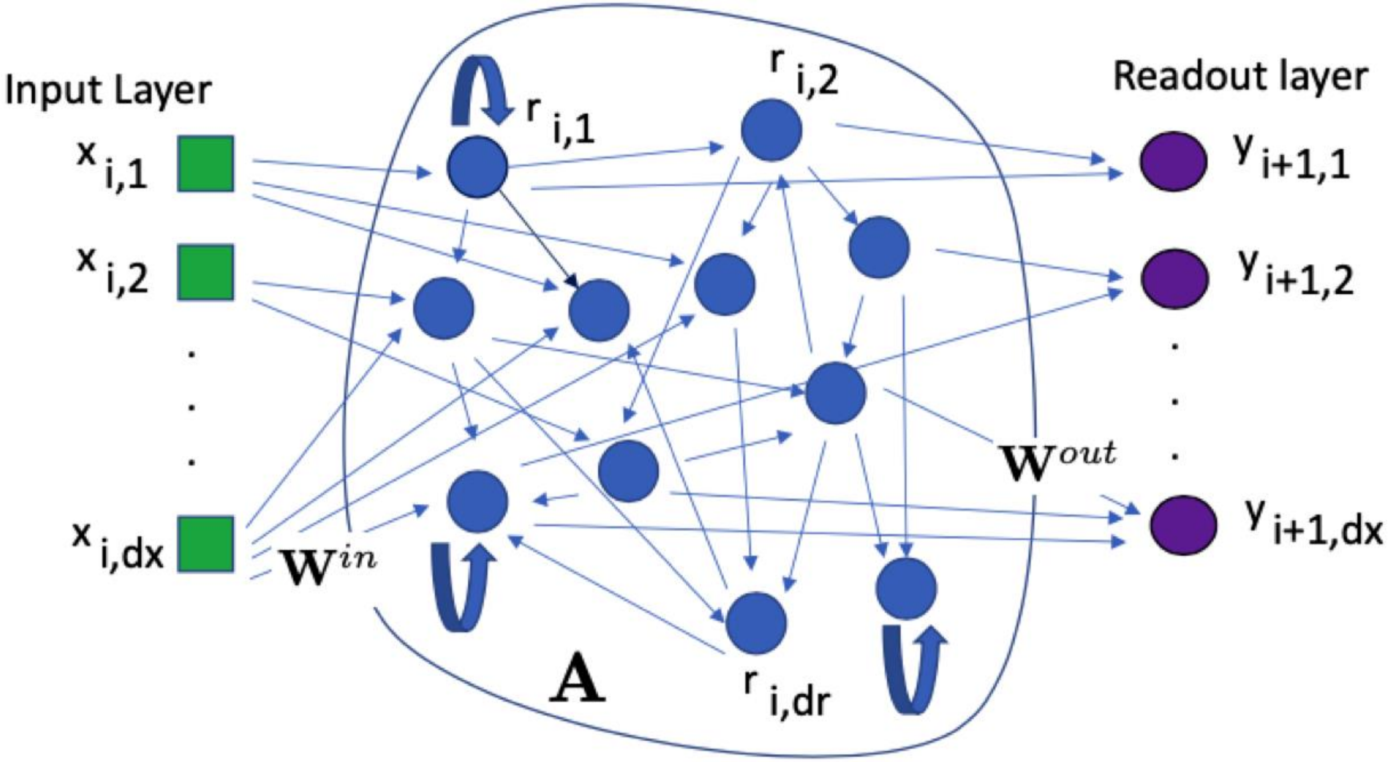


NG-RC is 1. simple – 2. MUCH less data hungry – 3. few parameters – 4. flexible feature

An RC a kind of random RNN – very nice for time series forecasting - works GREAT!

My question – why does RC work at all - all sorts of random parameters

Answer: **time soaks up the random**



Show an equivalence – a logical bridge - **to NVAR** (VAR is a star of econometrics)

and also **to Koopman**

and along the way - an improvement that we call NG-RC

Notable from Literature
-Billings and NARX

Reservoir computing – a special case of RNN

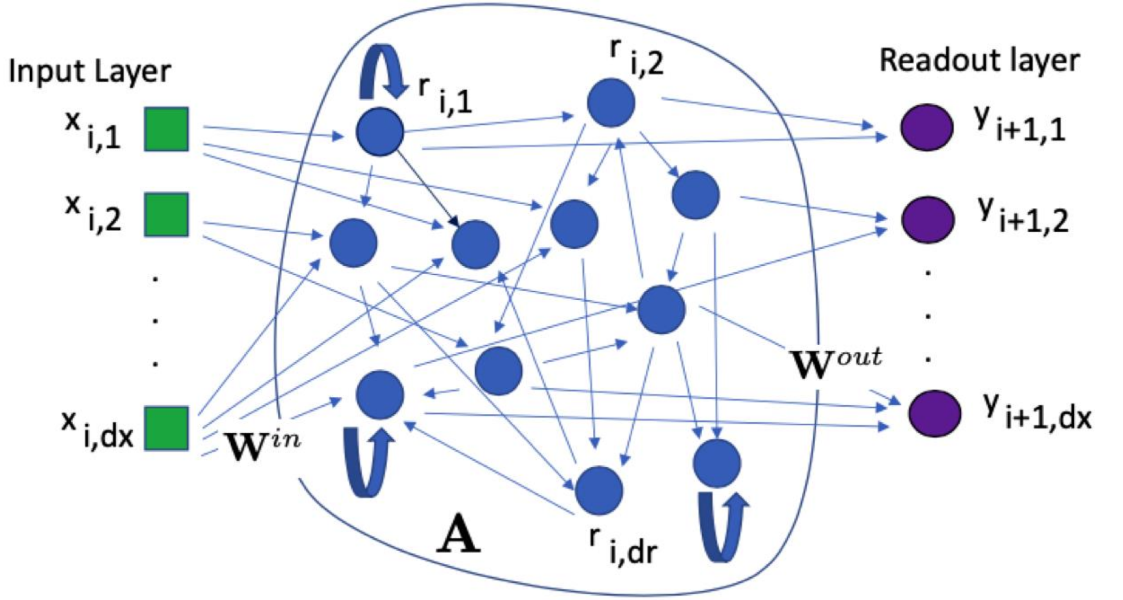
spec case ANN, Jaeger-Hass 2004, ESN-Jaeger 2001.

$$\{\mathbf{x}_i\}_{i=1}^{N} \subset \mathbb{R}^{d_x} \qquad d_r > d_x$$

$$\mathbf{u}_i = \mathbf{W}^{in}\mathbf{x}_i, \qquad \mathbf{r}_i \in \mathbb{R}^{d_r}$$

$$\mathbf{r}_{i+1} = (1-\alpha)\mathbf{r}_i + \alpha q(\mathbf{A}\mathbf{r}_i + \mathbf{u}_i + \mathbf{b}),$$

$$\mathbf{y}_{i+1} = \mathbf{W}^{out}\mathbf{r}_{i+1}.$$



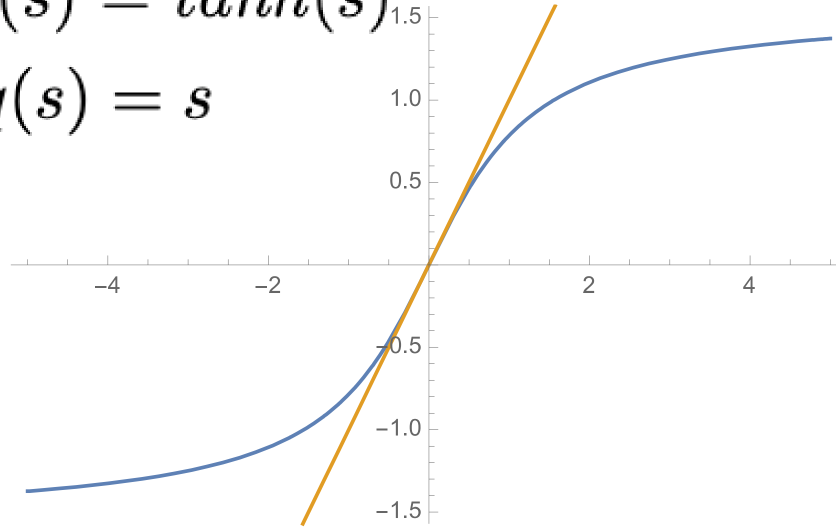
$\mathbf{W}^{in}_{i,j} \sim U(0,\gamma)$ $\quad d_r \times d_x$ read in matrix

$\mathbf{A}_{i,j} \sim U(-\beta,\beta)$, with β to scale the spectral radius

$d_x \times d_r$ *trained read-out matrix* matrix $\mathbf{W}^{out}$

$q(s) = tanh(s)$

→ $q(s) = s$



Surprise – A and $\mathbf{W}^{in}$ are random but it still works!

Notable Litt: Gonon - Ortega 19’, 20’ – RC enjoys a universal approximation theorem.

Even if linear with nonlinear readout.

Turns out that a Reservoir Computer is some kind of random RNN – but relates to a classical VAR(k) – *a star from Econometrics and stochastic processes*

an autoregressive model of order p can be written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

So what?

There is a very well developed theory for AR and VAR

– notably
Existence of Representation Theorem by WOLD

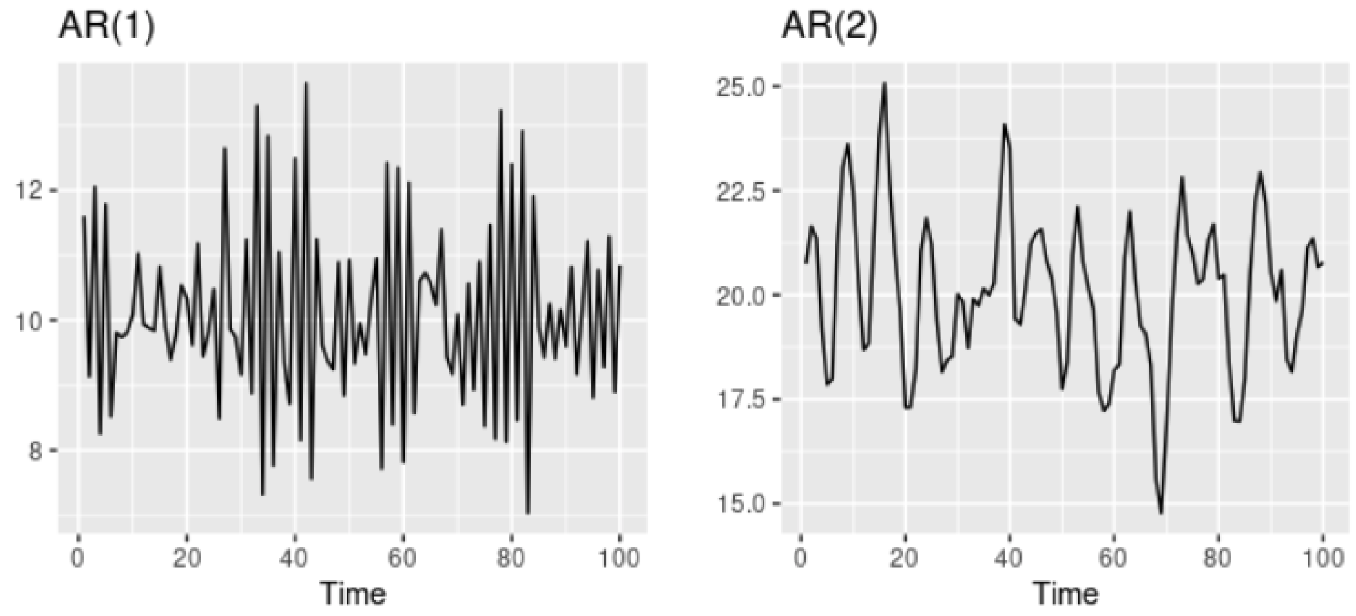
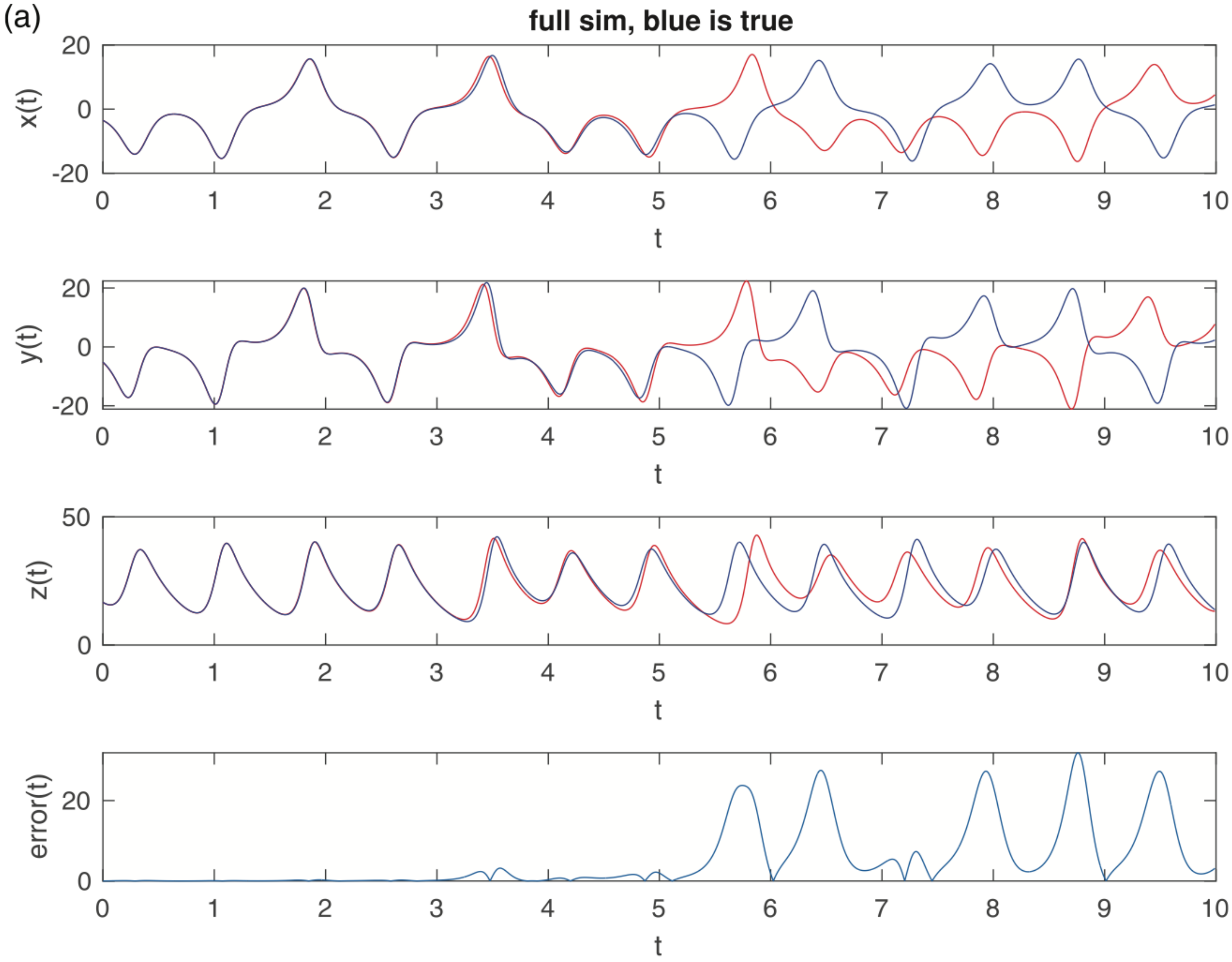
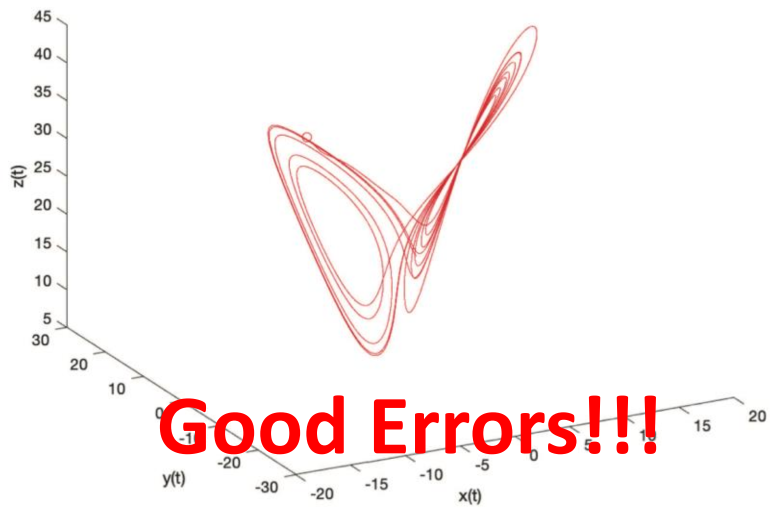


Figure 8.5: Two examples of data from autoregressive models with different parameters. Left: AR(1) with $y_t = 18 - 0.8y_{t-1} + \varepsilon_t$. Right: AR(2) with $y_t = 8 + 1.3y_{t-1} - 0.7y_{t-2} + \varepsilon_t$. In both cases, ε_t is normally distributed white noise with mean zero and variance one.

What-Why – Reservoir computing – forecast future from time series data of chaotic process or stochastic process





Good Errors!!!

$$\{X_t : t \in T\}$$

$$\begin{aligned}\dot{x} &= \sigma(y - x)\\ \dot{y} &= rx - y - xz\\ \dot{z} &= xy - bz\end{aligned}$$

Lorenz63

My question is – *why does it work at all* with all sorts of random parameters?

Things people do to make it work better – parameters and hyperparameters

-distribution to randomly select **A** (e.g. by sparsity and scaling) to control spectral radius

-a better distribution for read-in $\mathbf{W}^{in}$ to control scale

Here: Strip we away to a simplified version, maybe even “make it worse” – for purpose to interpret analytically.

-we choose simple distributions for read W_in and A --- *a linear - identity threshold q(s)=s*

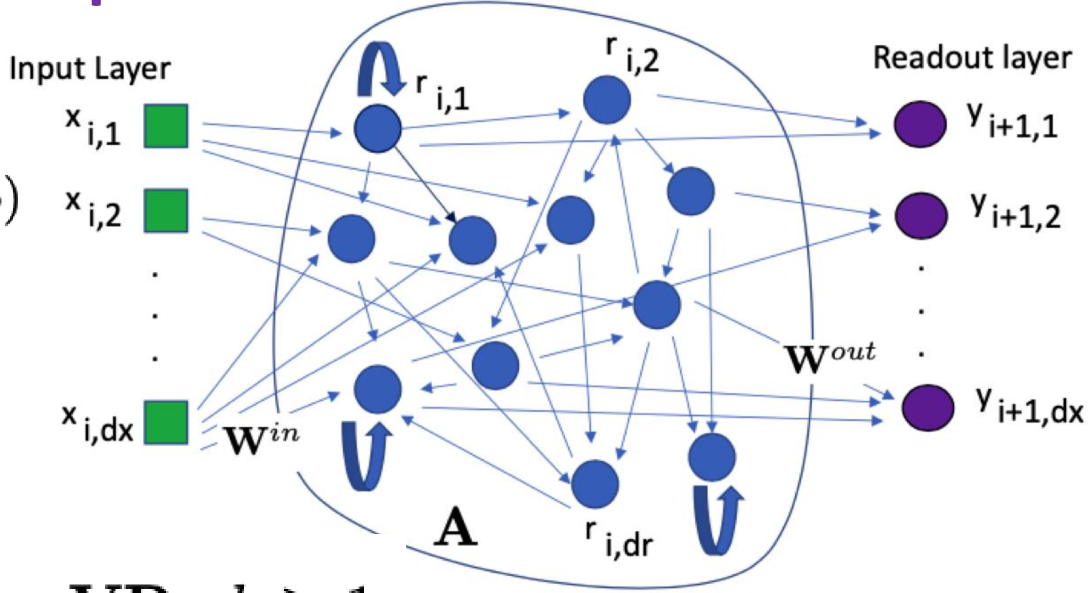
Punchline - now it become directly comparable to a vector autoregressive process – VAR –

-and with the VAR vs VMA which allows a representation theorem by WOLD

-also it is a bit like DMD-Koopman.

AND -linear RC with nonlinear readout = NVAR => a NG-RC variant of NVAR and vice versa.

Fitting the readout matrix by (regularized) least squares – the usual RC



Inner variables

$$\mathbf{r}_{i+1} = (1-\alpha)\mathbf{r}_i + \alpha q(\mathbf{A}\mathbf{r}_i + \mathbf{u}_i + \mathbf{b})$$

$$\mathbf{R} = [\mathbf{r}_{k+1}|\mathbf{r}_{k+1}|\dots|\mathbf{r}_N],\ k \geq 1.$$

True variables

$$\mathbf{X} = [\mathbf{x}_{k+1}|\mathbf{x}_{k+1}|\dots|\mathbf{x}_N] = [\mathbf{V}\mathbf{r}_{k+1}|\mathbf{V}\mathbf{r}_{k+2}|\dots|\mathbf{V}\mathbf{r}_N] = \mathbf{V}\mathbf{R},\ k \geq 1$$

$$\mathbf{W}_{out} = \underset{\mathbf{V}\in\mathbb{R}^{d_x \times d_r}}{\arg\min} \|\underline{X} - \mathbf{V}\mathbf{R}\|_F = \underset{\mathbf{V}\in\mathbb{R}^{d_x \times d_r}}{\arg\min} \sum_{i=k}^{N} \|\mathbf{x}_i - \mathbf{V}\mathbf{r}_i\|_2,$$

Train *just* the output

(Tikhonov regularized – ridge regression) least squares solution – helps prevent overfitting

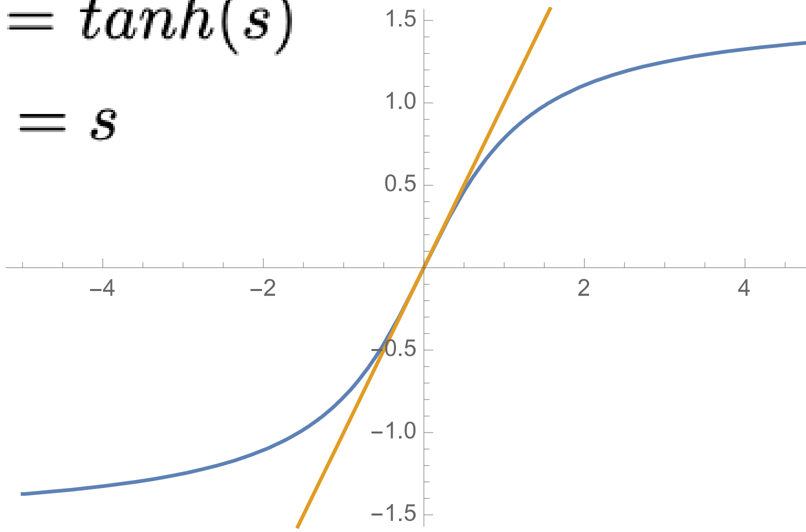
$$\mathbf{W}^{out} := \mathbf{X}\mathbf{R}^T(\mathbf{R}\mathbf{R}^T + \lambda\mathbf{I})^{-1}$$

RC With A Fully Linear Activation, $q(s) = s$, Yields a VAR(k)

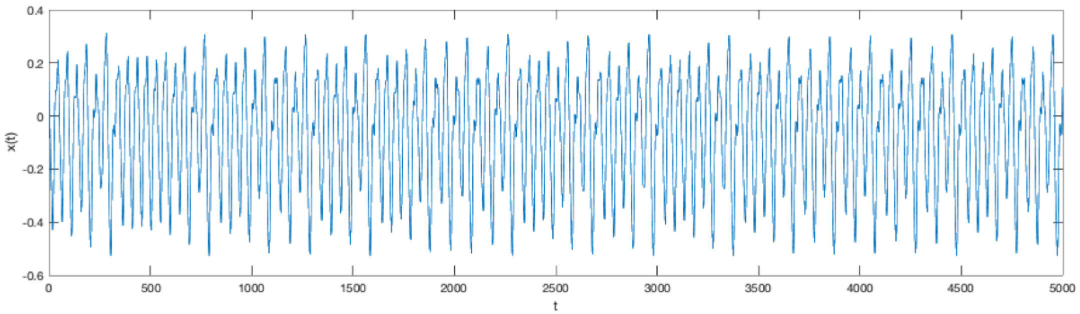
Note: $q(s) = tanh(s) \approx s - s^3/3 \ldots,$

$q(s) = tanh(s)$

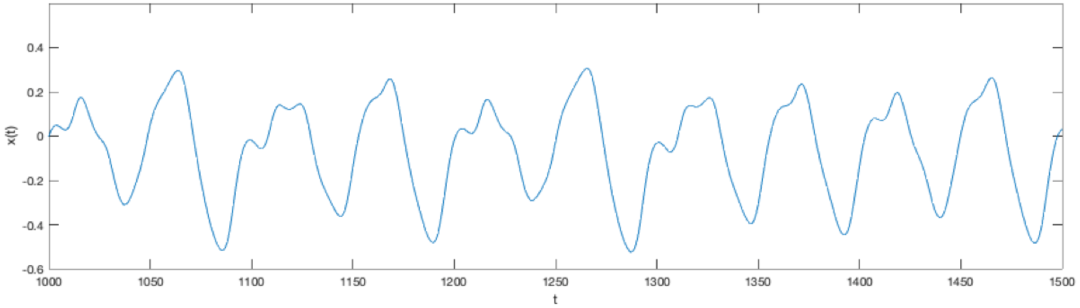
$\rightarrow q(s) = s$

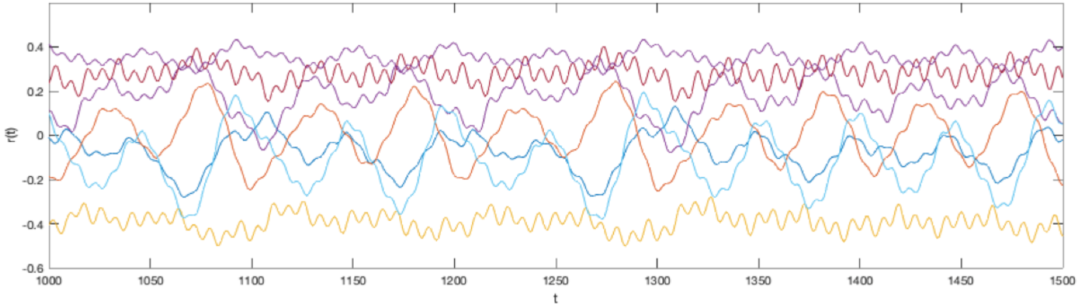


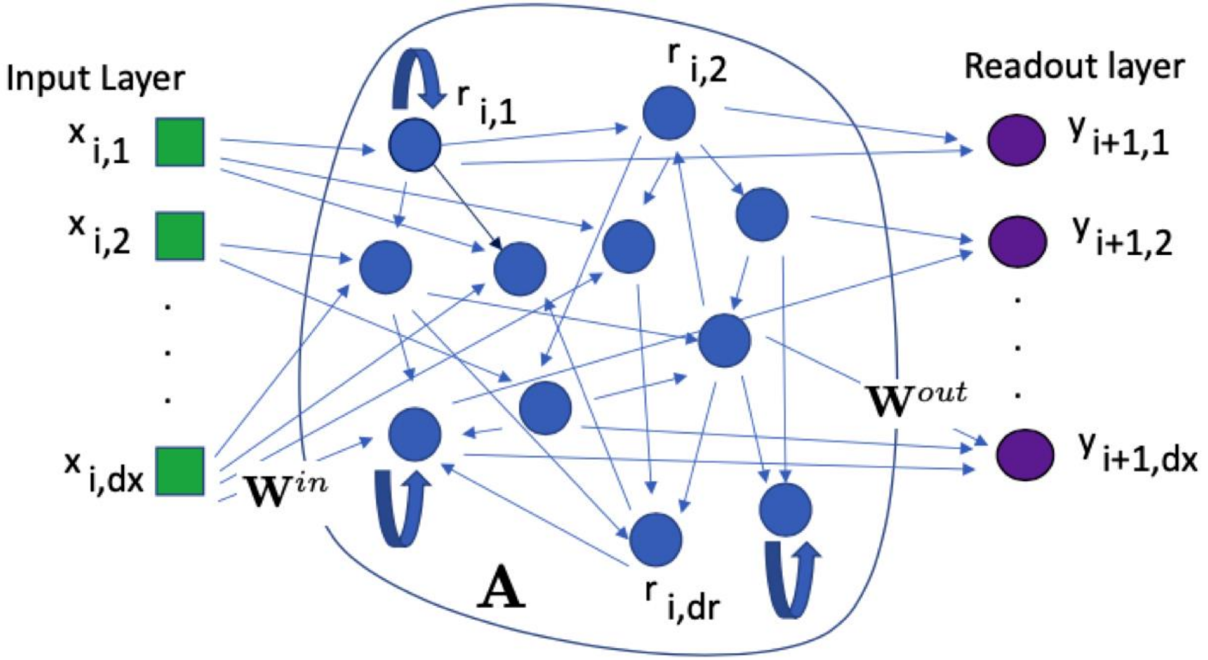
For small s, for small r, what if we just choose? $q(s) = s.$



Example RC for Mackey-Glass







$$\mathbf{r}_{i+1} = (1-\alpha)\mathbf{r}_i + \alpha q(\mathbf{A}\mathbf{r}_i + \mathbf{u}_i + \mathbf{b})$$

How to: Then just iterate– RC is a simple linear iteration with q(s)=s activation

$$
\begin{aligned}
\mathbf{r}_2 &= \mathbf{A}\mathbf{r}_1 + \mathbf{u}_1 = \mathbf{u}_1 = \mathbf{W}^{in}\mathbf{x}_1, \quad \mathbf{u}_1 = \mathbf{W}^{in}\mathbf{x}_1, \text{ but also we choose, } \mathbf{r}_1 = 0. \\
\mathbf{r}_3 &= \mathbf{A}\mathbf{r}_2 + \mathbf{u}_2 \\
&= A\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{in}\mathbf{x}_2 \\
\mathbf{r}_4 &= \mathbf{A}\mathbf{r}_3 + \mathbf{u}_3 \\
&= \mathbf{A}(\mathbf{A}\mathbf{r}_2 + \mathbf{u}_2) + \mathbf{u}_3 \\
&= \mathbf{A}^2\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{A}\mathbf{W}^{in}\mathbf{x}_2 + \mathbf{W}^{in}\mathbf{x}_3 \\
&\vdots \\
\mathbf{r}_{k+1} &= \mathbf{A}\mathbf{r}_k + \mathbf{u}_k \\
&= \mathbf{A}(\mathbf{A}\mathbf{r}_{k-1} + \mathbf{u}_{k-1}) + \mathbf{u}_k \\
&\vdots \\
&= \mathbf{A}^{k-1}\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{A}^{k-2}\mathbf{W}^{in}\mathbf{x}_2 + \ldots + \mathbf{A}\mathbf{W}^{in}\mathbf{x}_{k-1} + \mathbf{W}^{in}\mathbf{x}_k \\
&= \sum_{j=1}^{k} \mathbf{A}^{j-1}\mathbf{u}_{k-j+1} = \sum_{j=1}^{k} \mathbf{A}^{j-1}\mathbf{W}^{in}\mathbf{x}_{k-j+1}, \quad \mathbf{A}^0 = I
\end{aligned}
$$

just iterate on hidden variable

A linear RC, linear readout = implicit vector autoregressive

$$
\begin{aligned}
\mathbf{y}_{k+1} &= \mathbf{W}^{out}\mathbf{r}_{k+1} \\
&= \mathbf{W}^{out}\sum_{j=1}^{k}\mathbf{A}^{j-1}\mathbf{W}^{in}\mathbf{x}_{k-j+1} \\
&= \mathbf{W}^{out}\mathbf{A}^{k-1}\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{out}\mathbf{A}^{k-2}\mathbf{W}^{in}\mathbf{x}_2 + \ldots + \mathbf{W}^{out}\mathbf{A}\mathbf{W}^{in}\mathbf{x}_{k-1} + \mathbf{W}^{out}\mathbf{W}^{in}\mathbf{x}_k \\
&= a_k\mathbf{x}_1 + a_{k-1}\mathbf{x}_2 + \ldots + a_2\mathbf{x}_{k-1} + a_1\mathbf{x}_k,
\end{aligned}
$$

Remind anyone of Arnoldi?

with notation,

$$a_j = \mathbf{W}^{out}\mathbf{A}^{j-1}\mathbf{W}^{in}, \ j = 1, 2, ..., k.$$

coefficients a_j are $d_x \times d_x$ matrices

Conclude:

A linear RC - linear readout **=** vector autoregressive of k-delays estimator of a stochastic process –a classical VAR(k) –*from Econometrics and stochastic processes*

$$\mathbf{y}_{k+1} = c + a_k\mathbf{x}_1 + a_{k-1}\mathbf{x}_2 + \ldots + a_2\mathbf{x}_{k-1} + a_1\mathbf{x}_k + \boldsymbol{\xi}_{k+1}$$

Existence – WOLD theorem - And just this already this works “**pretty well**”

Explicit logical **Bridge**: **RC =** A Lovely **VAR(k)**

VAR: a star from Econometrics – works "ok" here – will do better

$$\begin{bmatrix} | & | & | & | \\ \mathbf{y}_{k+1} & \mathbf{y}_{k+2} & \dots & \mathbf{y}_N \\ | & | & | & | \end{bmatrix} = [[a_1] \quad [a_2] \quad \dots \quad [a_k]] \begin{bmatrix} | & | & \vdots & | \\ \mathbf{x}_k & \mathbf{x}_{k+1} & \dots & \mathbf{x}_{N-1} \\ | & | & \vdots & | \\ \mathbf{x}_{k-1} & \mathbf{x}_k & \dots & \mathbf{x}_{N-2} \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_{N-k-1} \\ | & | & \vdots & | \end{bmatrix}$$

$$\mathbf{a}^* = \mathbf{X}\mathbb{X}^T(\mathbb{X}\mathbb{X}^T + \lambda I)^{-1} := \mathbf{X}\mathbb{X}_\lambda^\dagger$$

With the Relationship between var coefficients and RC

$$a_j = \mathbf{W}^{out}\mathbf{A}^{j-1}\mathbf{W}^{in}, \; j = 1, 2, ..., k.$$

The directly fitted VAR coefficients

$$\mathbf{W}^{out} := \mathbf{v}^* = \mathbf{a}^*\mathbb{A}_\lambda^\dagger = \mathbf{X}\mathbb{X}_\lambda^\dagger\mathbb{A}_\lambda^\dagger$$

|
V

Relate the RC and the VAR: $\mathbf{Y} = \mathbf{a}\mathbb{X} = \mathbf{v}\mathbb{A}\mathbb{X}.$

$$\mathbb{A} = [\mathbf{W}^{in}|\mathbf{A}\mathbf{W}^{in}|\dots|\mathbf{A}^{k-2}\mathbf{W}^{in}|\mathbf{A}^{k-1}\mathbf{W}^{in}]$$

Already - this works “**pretty well**”

(we will do much better shortly)

Fully linear RC, q(x)=x, d_r=1000

KEY TAKE AWAY at this point:

--

-but don’t do it this RC way…. ***** Do it the VAR way**

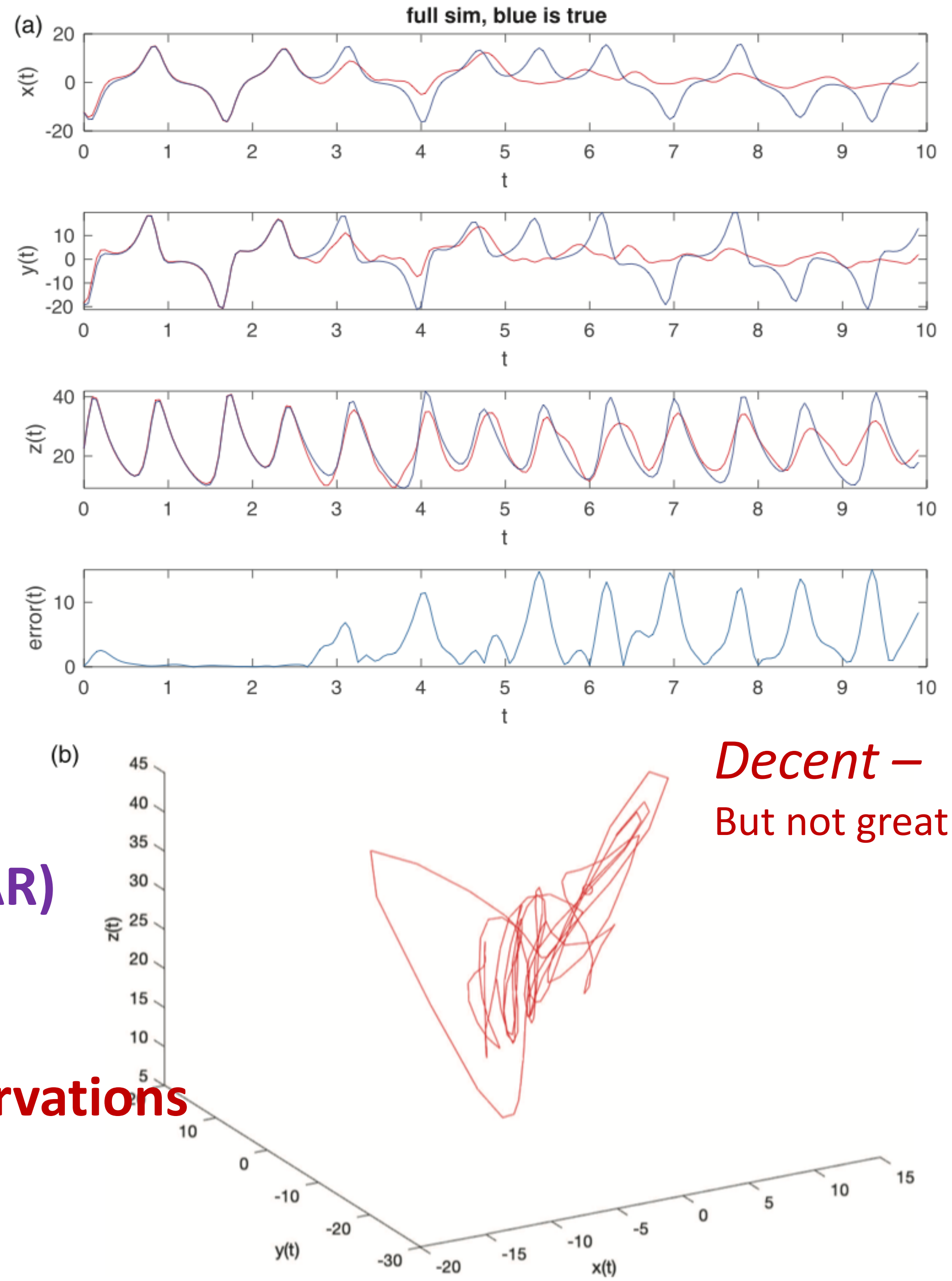
-for each “good” RC there is a corresponding VAR (NVAR)

-where did the random go?

-Linear RC with *linear* readout = implicit VAR

-Random projects out **– time & successive observations**

$$a_j = \mathbf{W}^{out}\mathbf{A}^{j-1}\mathbf{W}^{in},\ j = 1, 2, ..., k.$$

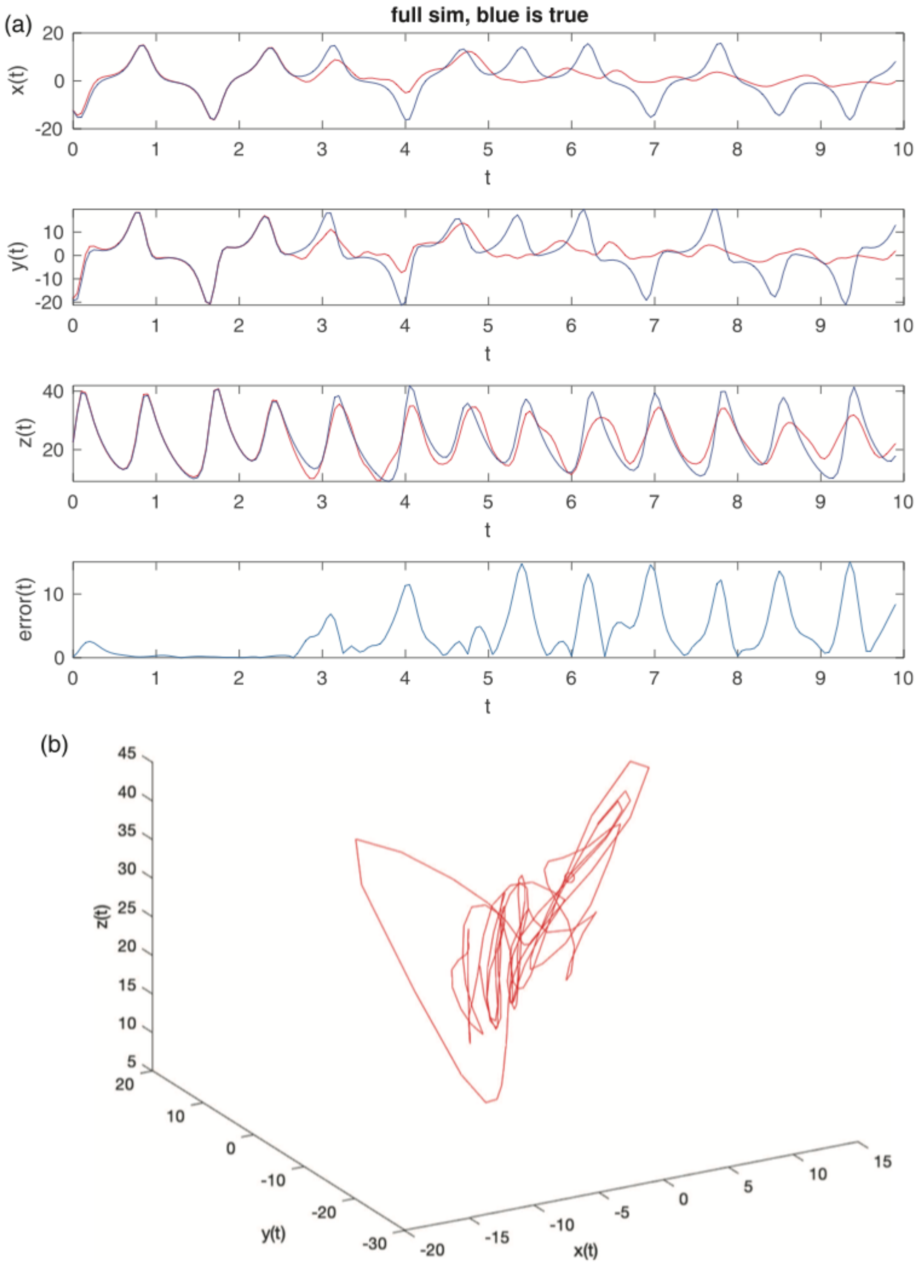


Decent –

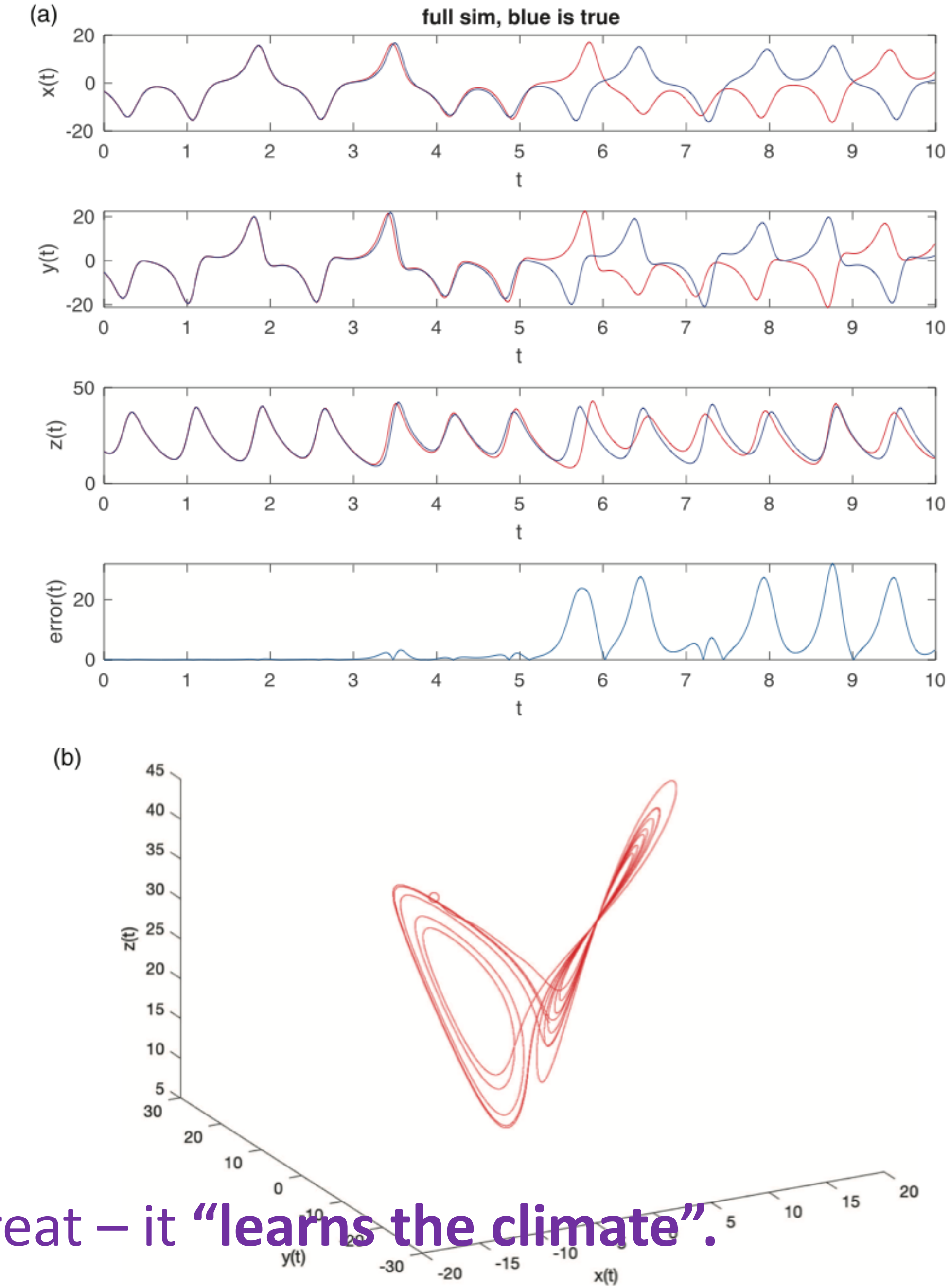
But not great

Already works "**pretty well**"

Works Great! – linear RC training with nonlinear readout



Fully linear RC, q(x)=x, d_r=1000

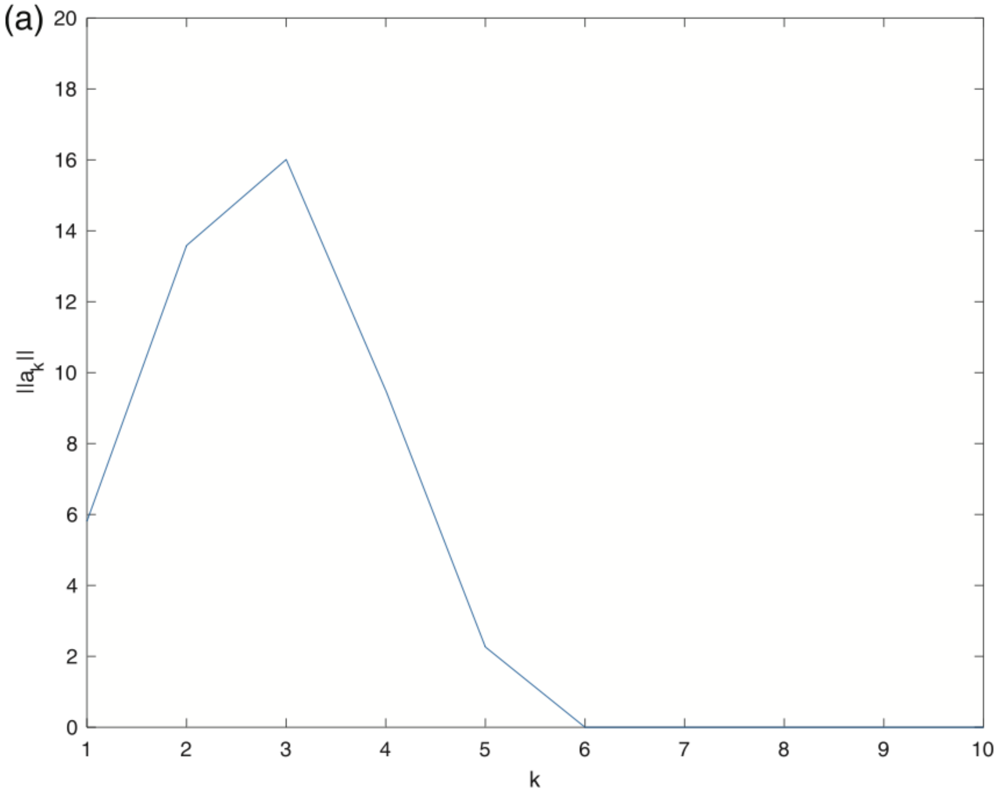


by works great – it **"learns the climate".**

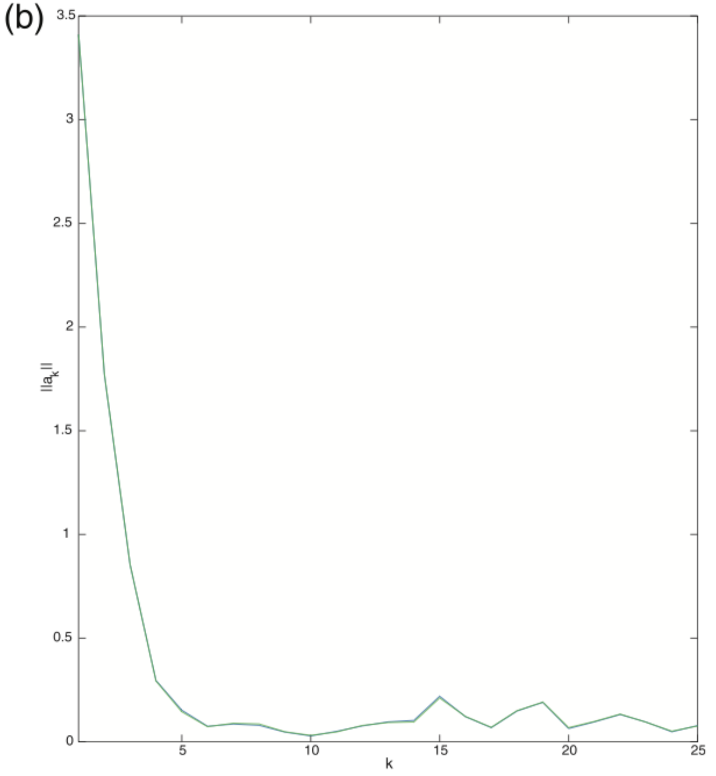
Naturally – Fading memory – time scale regarding A

$$\begin{aligned}\|a_j\|_\star &= \|\mathbf{W}^{out}\mathbf{A}^{j-1}\mathbf{W}^{in}\|_\star \\ &\leq \|\mathbf{W}^{out}\|_\star \|\mathbf{A}^{j-1}\|_\star \|\mathbf{W}^{in}\|_\star \\ &\leq \|\mathbf{W}^{out}\|_\star \|\mathbf{A}\|_\star^{j-1} \|\mathbf{W}^{in}\|_\star .\end{aligned}$$

Mackey-Glass



Lorenz63



Now explicit connection between NVAR=linear RC w' nonlinear readout

$$\mathbf{R}_1 = [\mathbf{r}_k \quad |\mathbf{r}_{k+1} \quad |\cdots \quad |\mathbf{r}_N],$$
$$\mathbf{R}_2 = [\mathbf{r}_k \circ \mathbf{r}_k \quad |\mathbf{r}_{k+1} \circ \mathbf{r}_{k+1} \quad |\cdots \quad |\mathbf{r}_N \circ \mathbf{r}_N]$$
$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \end{bmatrix}.$$

$$\mathbb{X}_1 = \begin{bmatrix} | & | & \vdots & | \\ \mathbf{x}_k & \mathbf{x}_{k+1} & \cdots & \mathbf{x}_{N-1} \\ | & | & \vdots & | \\ \mathbf{x}_{k-1} & \mathbf{x}_k & \cdots & \mathbf{x}_{N-2} \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{N-k} \\ | & | & \vdots & | \end{bmatrix}, \mathbb{X}_2 = \begin{bmatrix} | & | & \vdots & | \\ p_2(\mathbf{x}_k,\mathbf{x}_k) & p_2(\mathbf{x}_{k+1},\mathbf{x}_{k+1}) & \cdots & p_2(\mathbf{x}_{N-1},\mathbf{x}_{N-1}) \\ | & | & \vdots & | \\ p_2(\mathbf{x}_{k-1},\mathbf{x}_k) & p_2(\mathbf{x}_k,\mathbf{x}_{k+1}) & \cdots & p_2(\mathbf{x}_{N-2},\mathbf{x}_{N-1}) \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ p_2(\mathbf{x}_1,\mathbf{x}_k) & p_2(\mathbf{x}_2,\mathbf{x}_{k+1}) & \cdots & p_2(\mathbf{x}_{N-k-1},\mathbf{x}_{N-1}) \\ | & | & \vdots & | \\ p_2(\mathbf{x}_k,\mathbf{x}_{k-1}) & p_2(\mathbf{x}_{k+1},\mathbf{x}_k) & \cdots & p_2(\mathbf{x}_{N-1},\mathbf{x}_{N-2}) \\ | & | & \vdots & | \\ p_2(\mathbf{x}_{k-1},\mathbf{x}_{k-1}) & p_2(\mathbf{x}_{k+1},\mathbf{x}_{k-1}) & \cdots & p_2(\mathbf{x}_{N-2},\mathbf{x}_{N-2}) \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ p_2(\mathbf{x}_1,\mathbf{x}_1) & p_2(\mathbf{x}_2,\mathbf{x}_2) & \cdots & p_2(\mathbf{x}_{N-k},\mathbf{x}_{N-k}) \\ | & | & \vdots & | \end{bmatrix}.$$

Stack the monomials

$$\mathbf{W}^{out} = \begin{bmatrix} \mathbf{W}_1^{out} \\ \mathbf{W}_2^{out} \end{bmatrix} \qquad \mathbf{W}^{out} := \mathbf{X}\mathbf{R}^T(\mathbf{R}\mathbf{R}^T + \lambda \mathbf{I})^{-1}$$

$$p_2(\mathbf{v},\mathbf{w}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n^2},$$
$$(\mathbf{v},\mathbf{w}) \mapsto [v_1w_1|v_1w_2|\cdots|v_1w_n|v_2w_1|v_2w_2|\cdots|v_nw_n]^T,$$

Then again we get a version of

$$\mathbb{X} = \begin{bmatrix} \mathbb{X}_1 \\ \mathbb{X}_2 \end{bmatrix} \qquad \mathbf{Y} = \mathbf{a}\mathbb{X}$$

-said as NVAR

$$\mathbf{y}_{\ell+1} = a_\ell \mathbf{x}_1 + a_{\ell-1}\mathbf{x}_2 + \cdots + a_2\mathbf{x}_{\ell-1} + a_1\mathbf{x}_\ell + a_{2,(\ell,\ell)}p_2(\mathbf{x}_1,\mathbf{x}_1) + a_{2,(\ell-1,\ell)}p_2(\mathbf{x}_2,\mathbf{x}_1) + \cdots + a_{2,(1,1)}p_2(\mathbf{x}_\ell,\mathbf{x}_\ell),$$

Specifically - NVAR coeff relate to RC parameters

$$a_j = \mathbf{W}_1^{out}\mathbf{A}^{j-1}\mathbf{W}^{in}, j = 1,2,\ldots,\ell, \qquad a_{2,(i,j)} = \mathbf{W}_2^{out}P_2(A^{i-1}\mathbf{W}^{in}, A^{j-1}\mathbf{W}^{in}), i,j = 1,\ldots,\ell.$$

nature COMMUNICATIONS

ARTICLE

https://doi.org/10.1038/s41467-021-25801-2 OPEN

Next generation reservoir computing

Daniel J. Gauthier [1,2], Erik Bollt [3,4], Aaron Griffith [1] & Wendson A. S. Barbosa [1]

Linear RC with nonlinear readout = implicit NVAR ===> NG-RC

An implicit RC means we can skip RC – instead do NG-RC - efficient – less data hungry – skips the middle-man –
Less parameters and hyperparameters to worry about.

Almost no metaparameters for an Next Generation RC!

- Sample time of input data dt, total training time T_{train}

- Number of time delay taps *k* and the number of sample steps to "skip" *s*

$$\mathbf{F}_{lin} = \left[\mathbf{U}(t), \mathbf{U}(t - s\,dt), \mathbf{U}(t - 2\,s\,dt), \ldots, \mathbf{U}(t - k\,s\,dt)\right]^T$$

Linear part of feature vector

- Nonlinear form of output vector, *e.g.*,

Nonlinear part of feature vector

$$\mathbf{F}_{nonlinear} = \left[\mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin}, \mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin}, \mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin} \lceil \otimes \rceil \mathbf{F}_{lin}\right]^T$$

$$\mathbf{F}_{total} = \left[\mathbf{F}_{lin}, \mathbf{F}_{nonlinear}\right]^T$$

$\lceil \otimes \rceil$

Flatten, unique terms of outer product

- Ridge regression parameter α

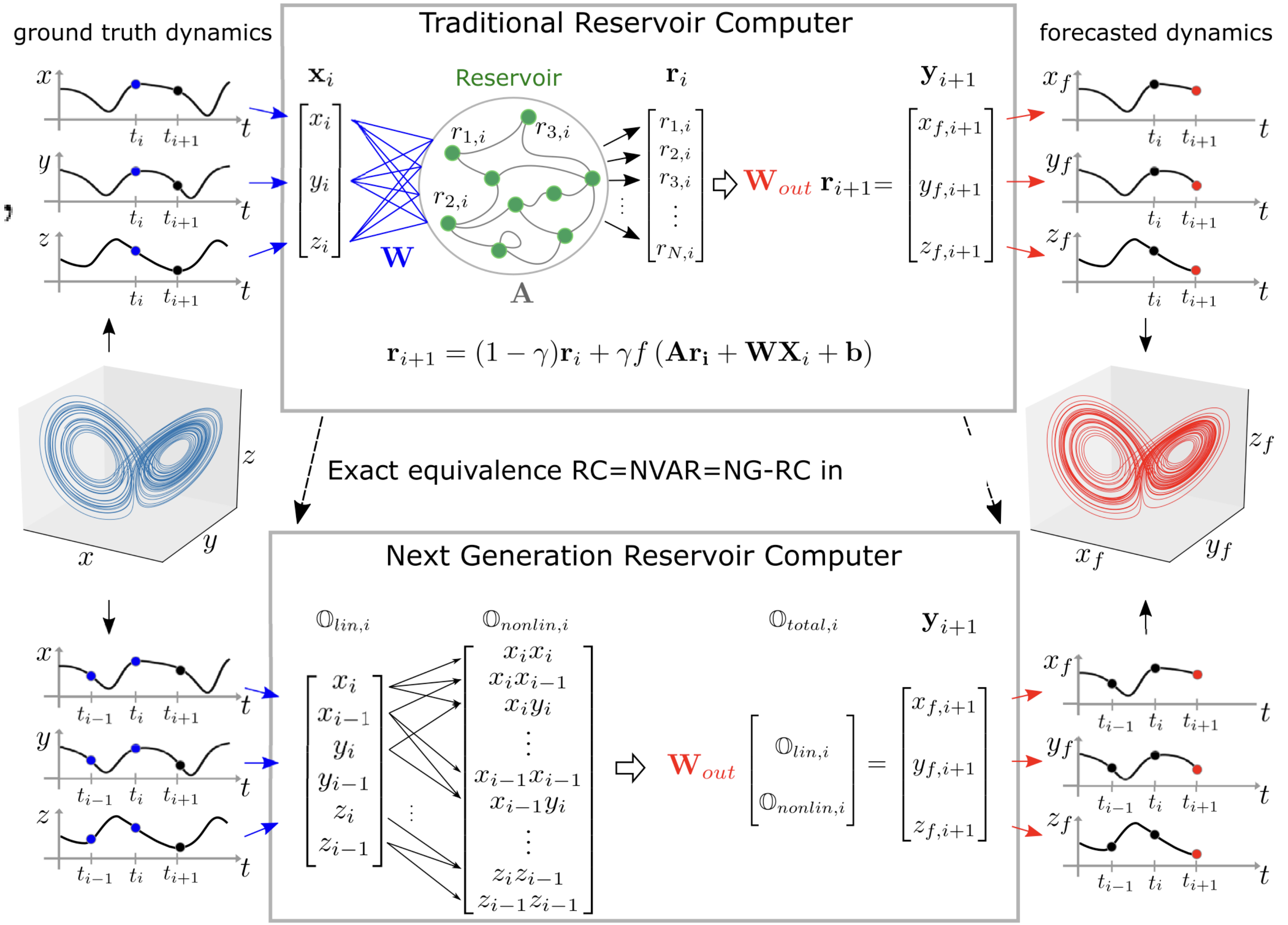
$$\mathbf{W}_{out} = \mathbf{Y}_{des}\mathbf{U}^T\left(\mathbf{U}\mathbf{U}^T + \alpha\mathbf{I}\right)^{-1}$$

Move the nonlinear from the activation function instead to a feature vector of inner state, Ortega, and also Bollt,

A linear reservoir with nonlinear output equivalently powerful as universal approximator with similar performance as Standard RC – equivalent to NVAR – equivalent to NG-RC - but with reliability and simplicity advantages.

$$\mathbf{Y}_{i+1} = \mathbf{W}_{\text{out}} \mathbb{O}_{\text{total},i+1},$$

$$\mathbf{W}_{\text{out}} = \mathbf{Y}_d \mathbb{O}_{\text{total}}{}^T \left(\mathbb{O}_{\text{total}} \mathbb{O}_{\text{total}}{}^T + \alpha \mathbf{I}\right)^{-1},$$



nature COMMUNICATIONS

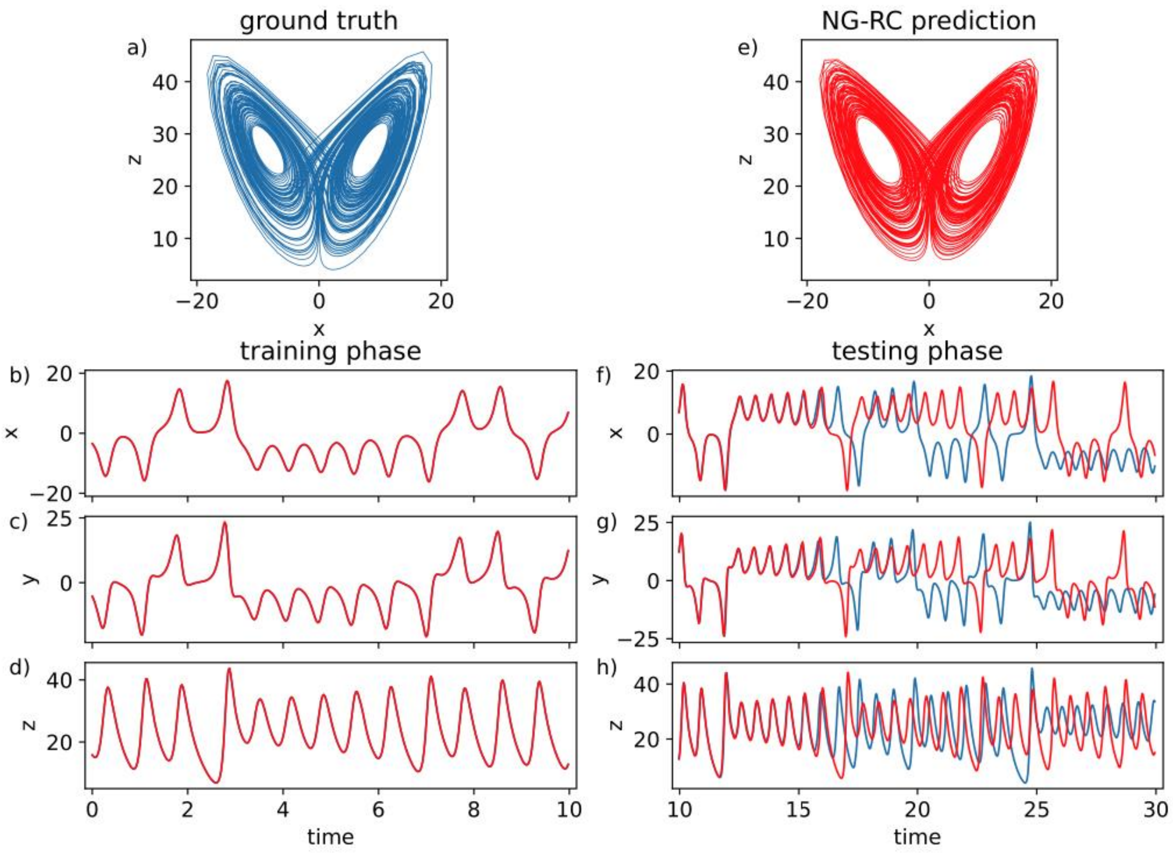
ARTICLE

https://doi.org/10.1038/s41467-021-25801-2 OPEN

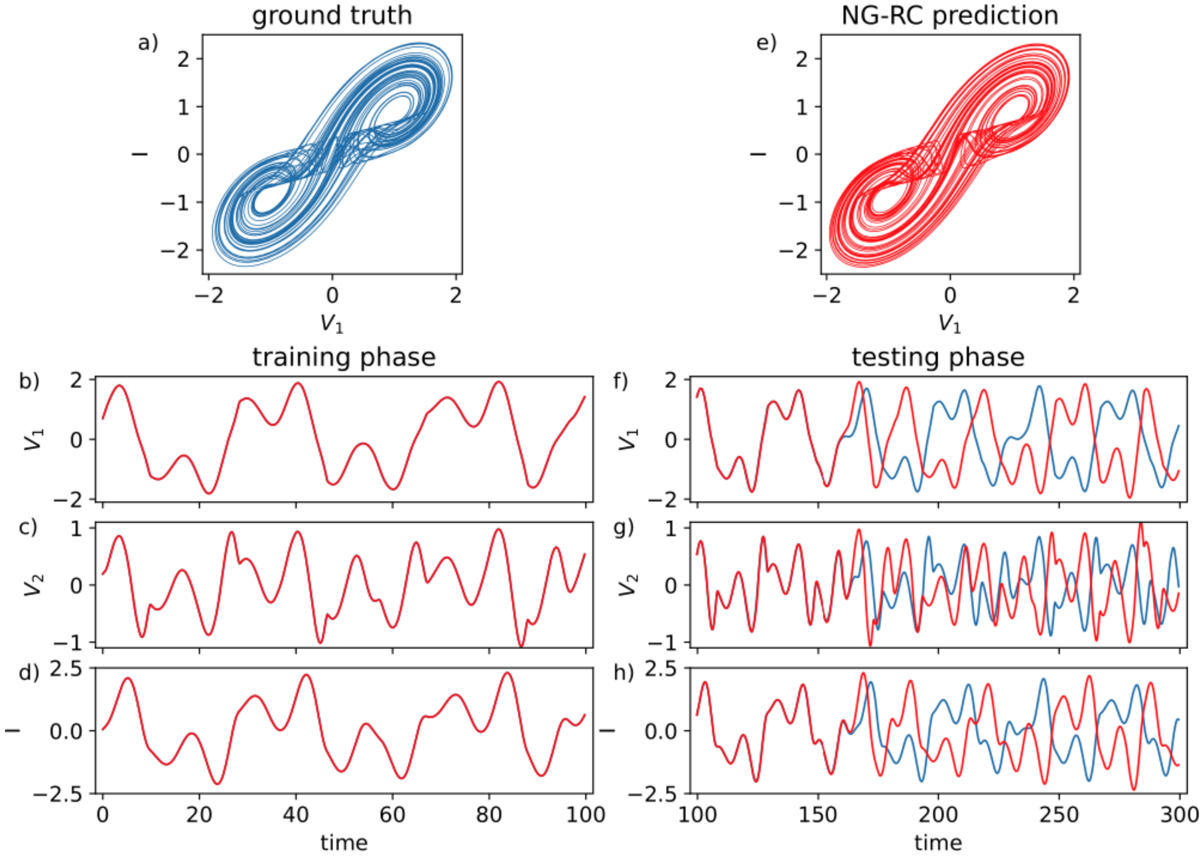
Next generation reservoir computing

Daniel J. Gauthier[1,2], Erik Bollt[3,4], Aaron Griffith[1] & Wendson A. S. Barbosa[1]

NG-RC works very well, with very few points, almost no tunable parameters



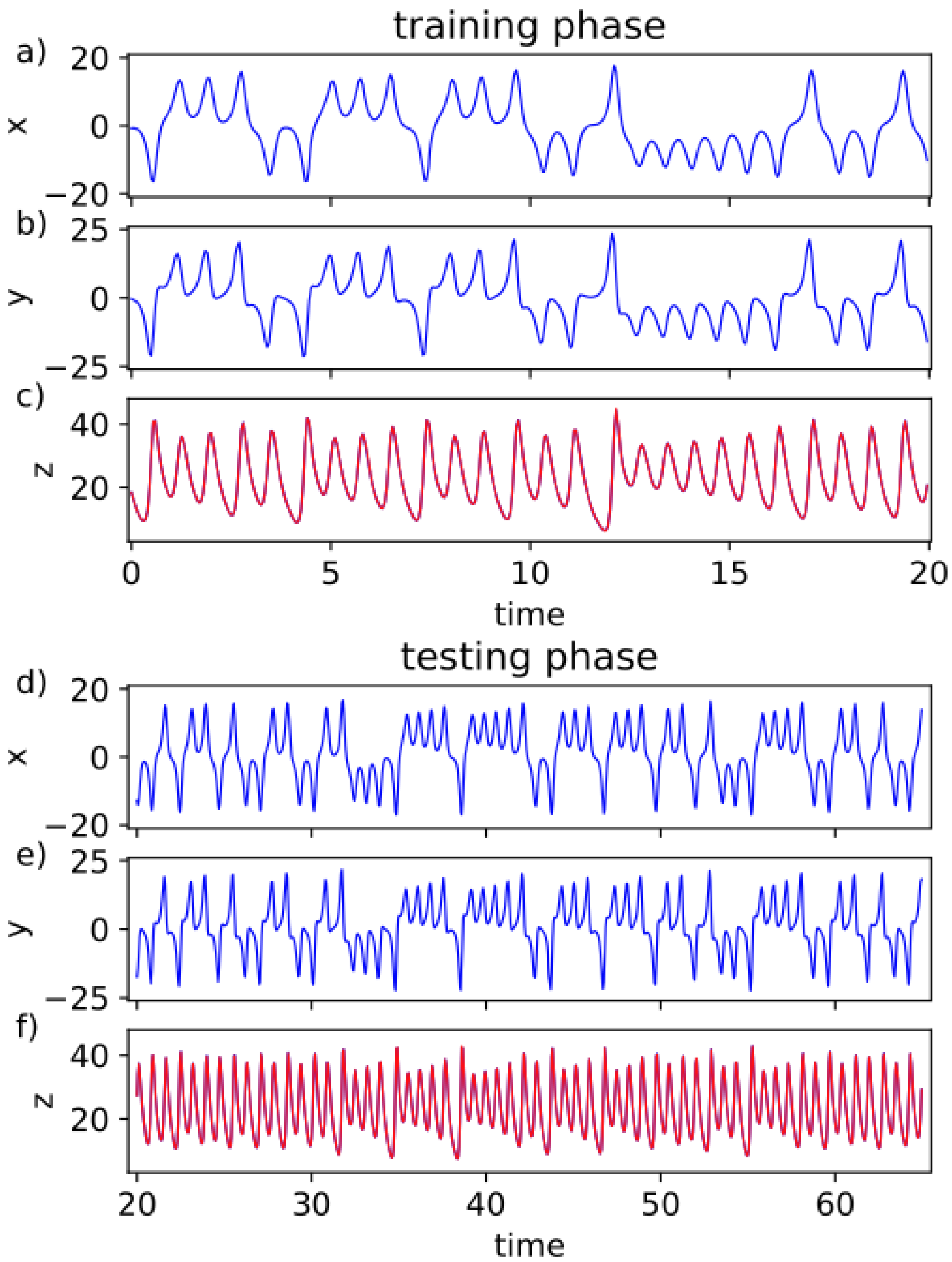
Forecasting a dynamical system using the NG-RC.
Lorenz63 strange attractors.



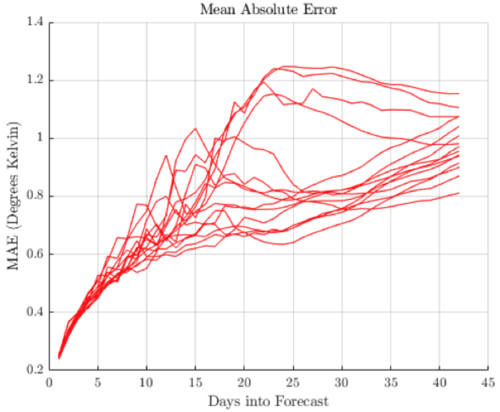
Forecasting the double-scroll system using the NG-RC

Another fun task – *look Ma! – no z*!

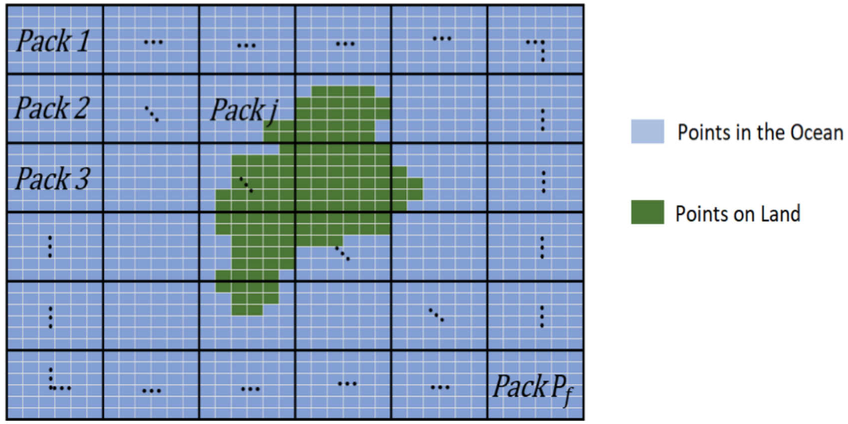
Inference using an NG-RC. a–c Lorenz63 variables during the training phase (blue) and prediction (c, red)

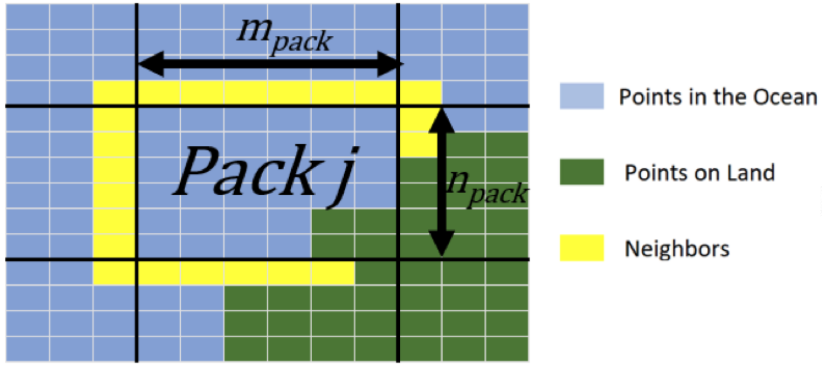


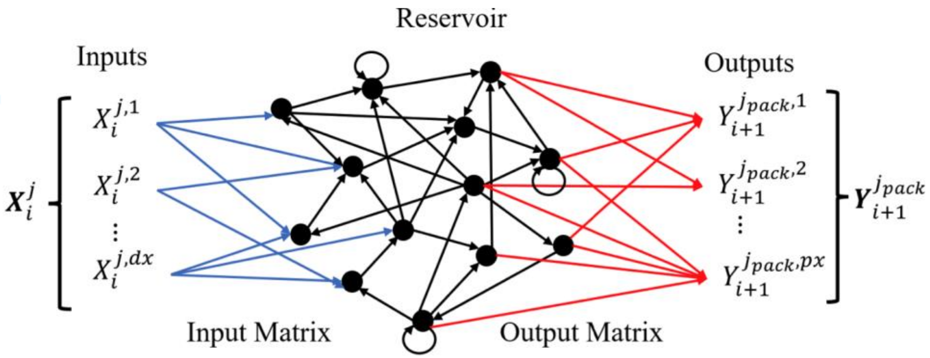
A substantial and spatiotemporally complex data set of significance – **SST Earth**



The mean absolute error for the 6 week forec[redacted]





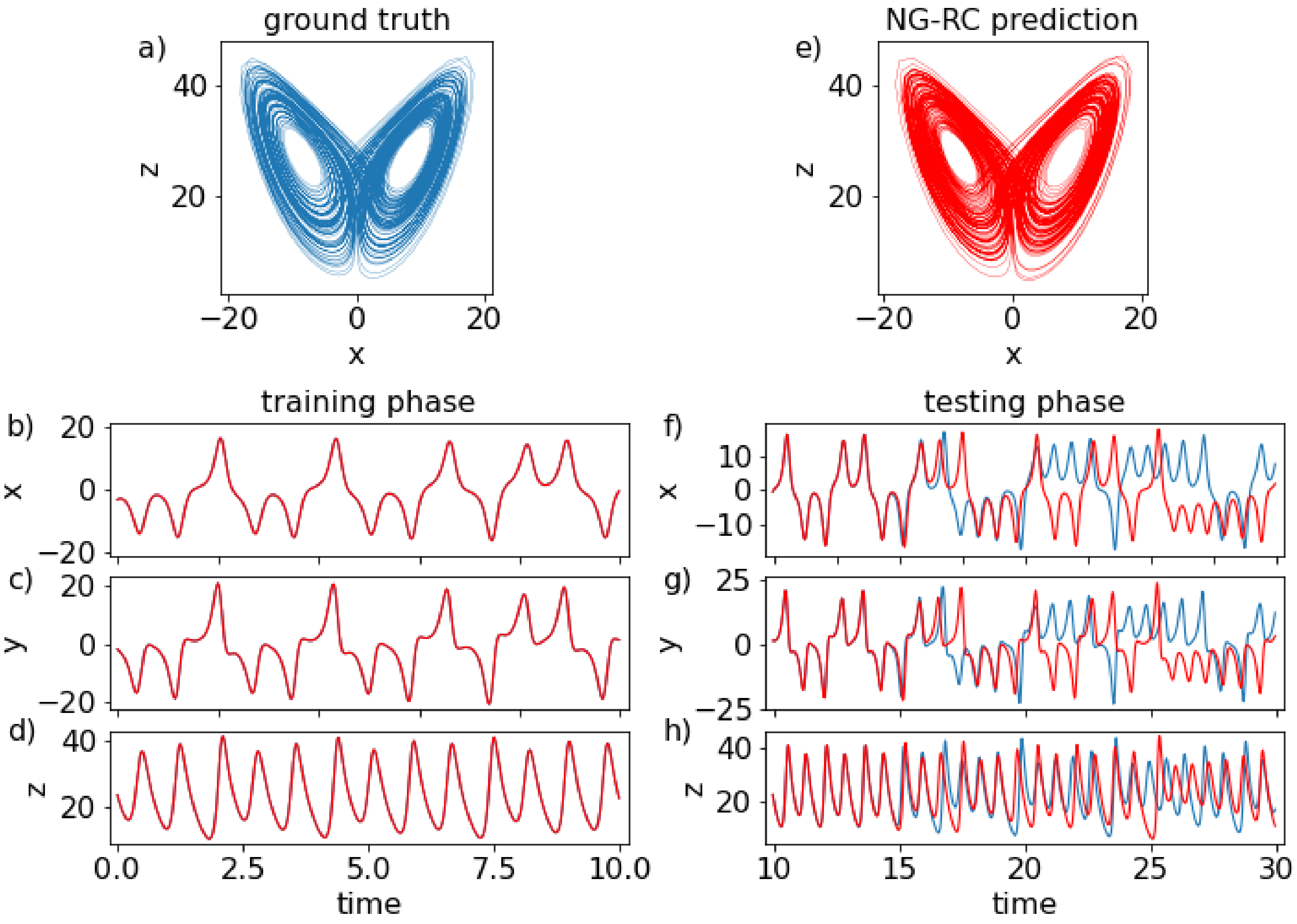


Walleshauser, Bollt. "Predicting Sea Surface Temperatures with Coupled Reservoir Computers." *Nonlinear Processes in Geo Disc*(2022): 1-19.

Conclude: Works really really well – and drastically MUCH less data hungry

-linear RC with nonlinear readout = implicit NVAR **AND this leads to** NG-RC

-VAR vs VMA which follows classic representation theorem by WOLD thm - also relates to DMD-Koopman



Data from Lorenz63,

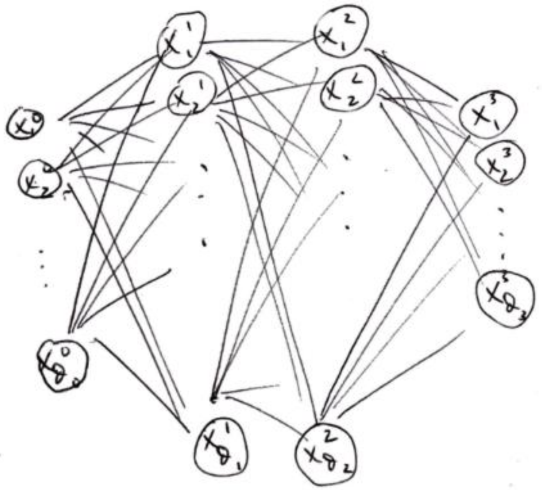
$$\dot{x} = 10\left(y - x\right),$$

$$\dot{y} = x\left(28 - z\right) - y,$$

$$\dot{z} = xy - \frac{8}{3}z,$$

NG-RC is 1. simple – 2. MUCH less data hungry – 3. few parameters – 4. flexible feature

On ELM – Extreme Learning Machine – Feedforward but Random Weights Variant of ANN
Much like RC you train just the output layer.



Again – obvious why it would be nice – cheap – but does it work?

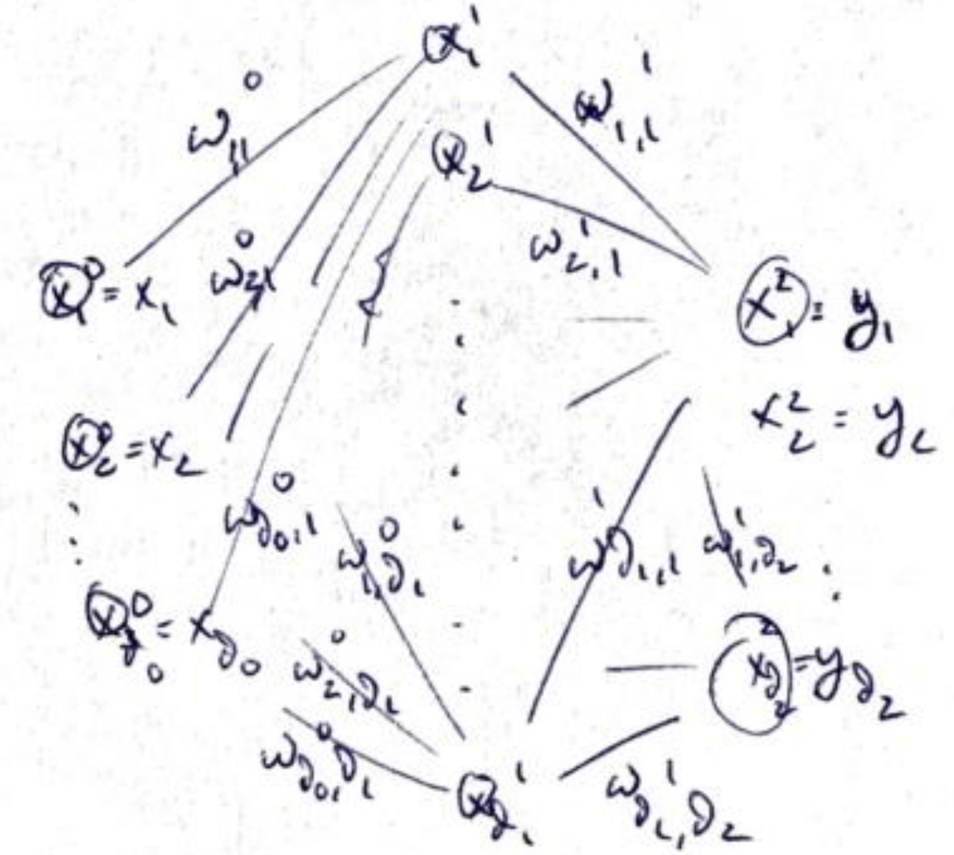
So ELM is usually stated as SLFNN

$$\sigma_r(s) = ReLu(s) = max(s, 0), r < q, \sigma_q(s) = s$$

$$F_{r,\Theta_r}(X^r) = \sigma_r(W^r X^r + B^{r+1})$$

$$F(X, \Theta) = F_{q,\Theta_q} \circ F_{q-1,\Theta_{q-1}} \circ \ldots \circ F_{0,\Theta_0}(X)$$

$$\mathcal{L}(\mathcal{D}; \Theta) = \sum_{i=1}^{N} \|F(X_i, \Theta) - Y_i\|_2 + \lambda \mathcal{R}(\Theta).$$



Just counting:

$$\Theta = \cup_{r=0}^{q} \Theta_r, \quad \Theta_r = \{\{w_{j,k}^r\}_{j=1,k=1}^{d_r, d_{r+1}}, \{b_k^{r+1}\}_{k=1}^{d_{r+1}}\}.$$

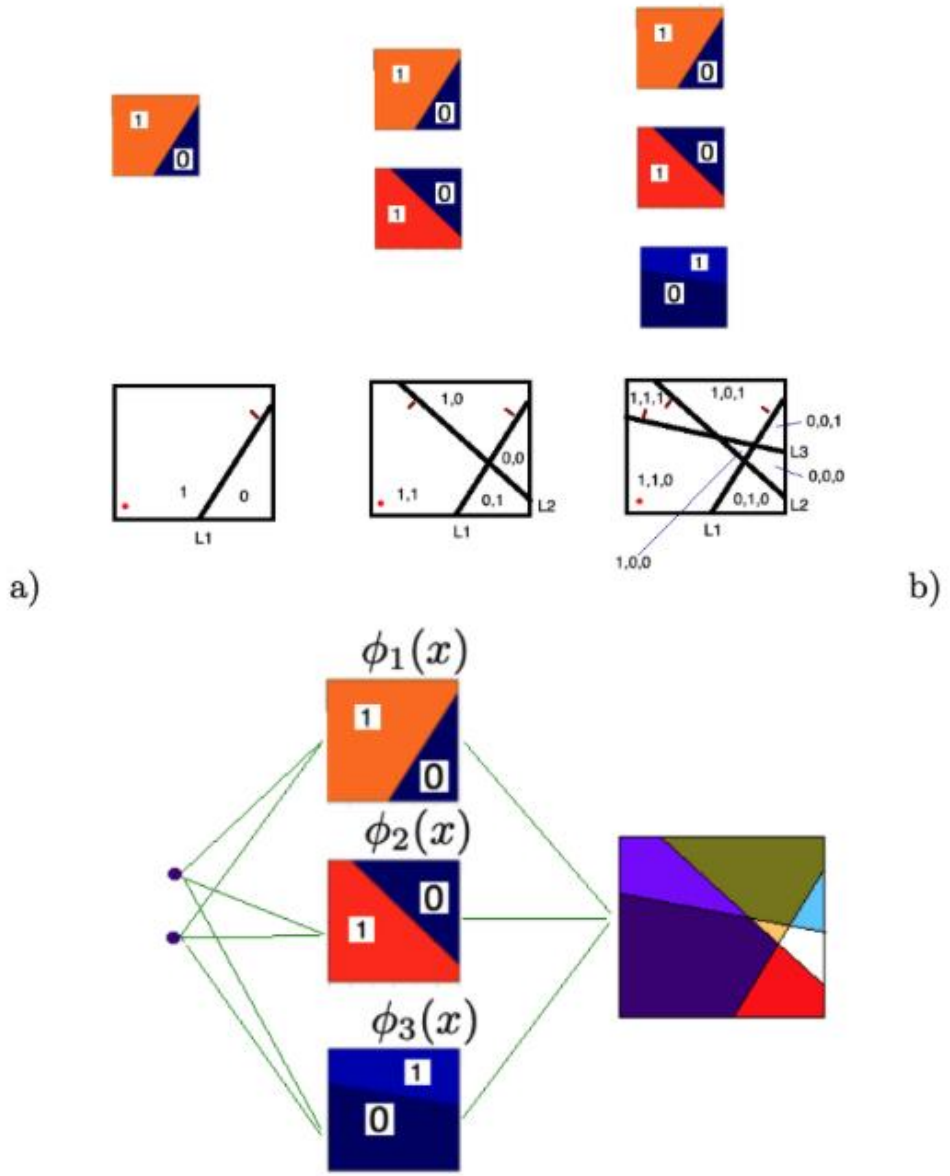
$$|\Theta| = d_0 d_1 + d_1 d_2 + d_1$$

Vs

$$|\Theta_{out}| = d_1 d_2 << |\Theta| = d_0 d_1 + d_1 d_2 + d_1 + d_2.$$

On ELM – the random shallow case, with ReLu, is especially easy to understand.

-Linear combinations of ReLu functions result and so
-domains of piecewise linear continuous function results.



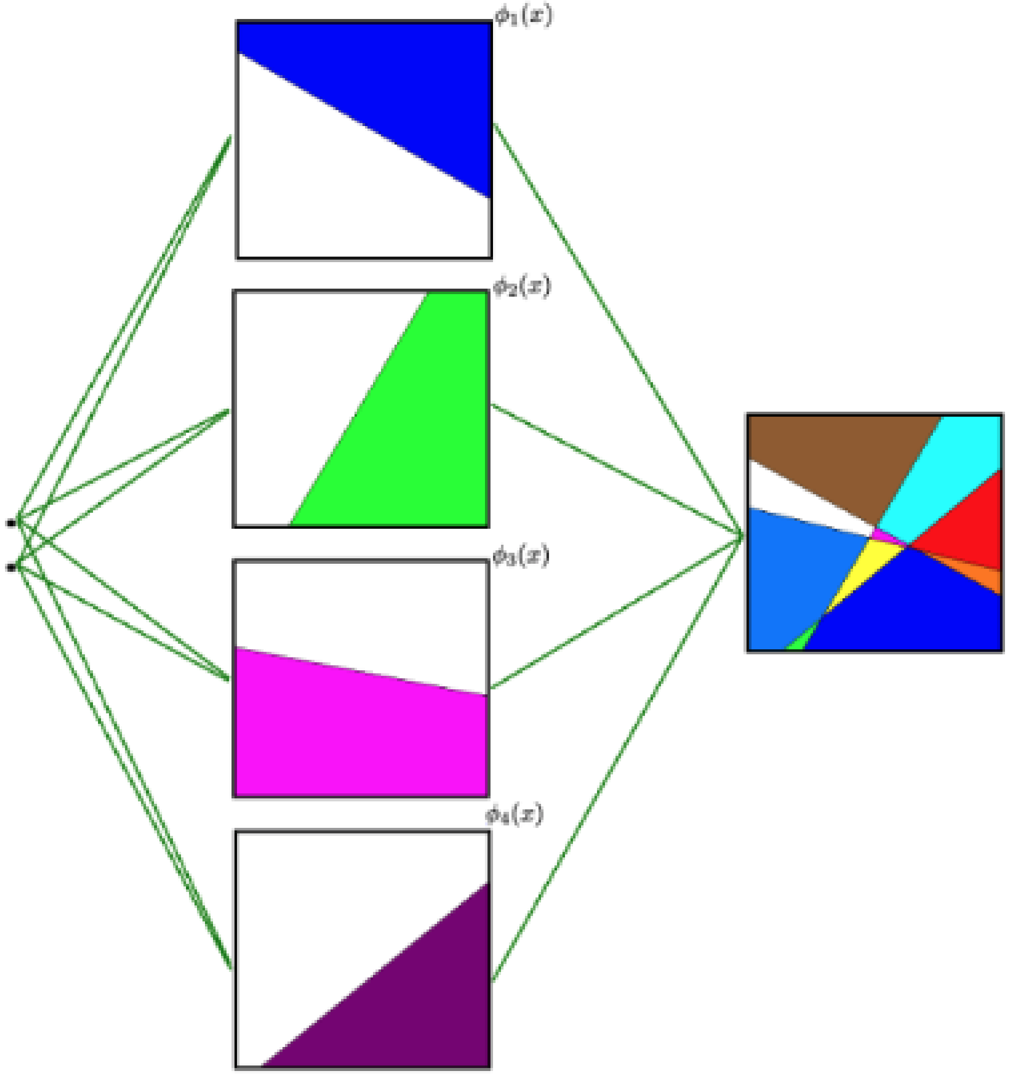
Expressive?

On refinement with growing layer
-a classic question of partition of d_0 dim space
by n-hyperplanes

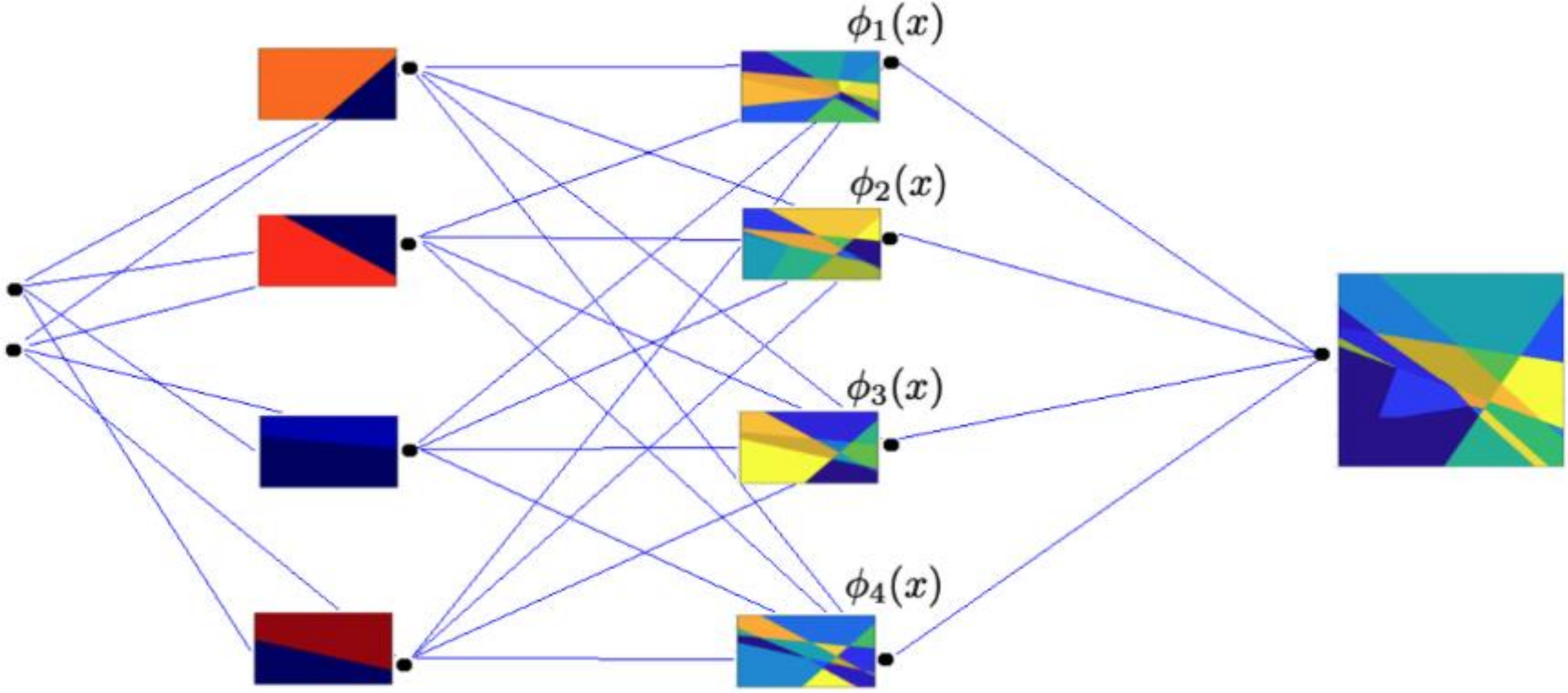
$$M(d_0, n) = \sum_{i=0}^{d_0} \binom{n}{i}$$

$$B(d_0, n) < M(d_0, n)$$

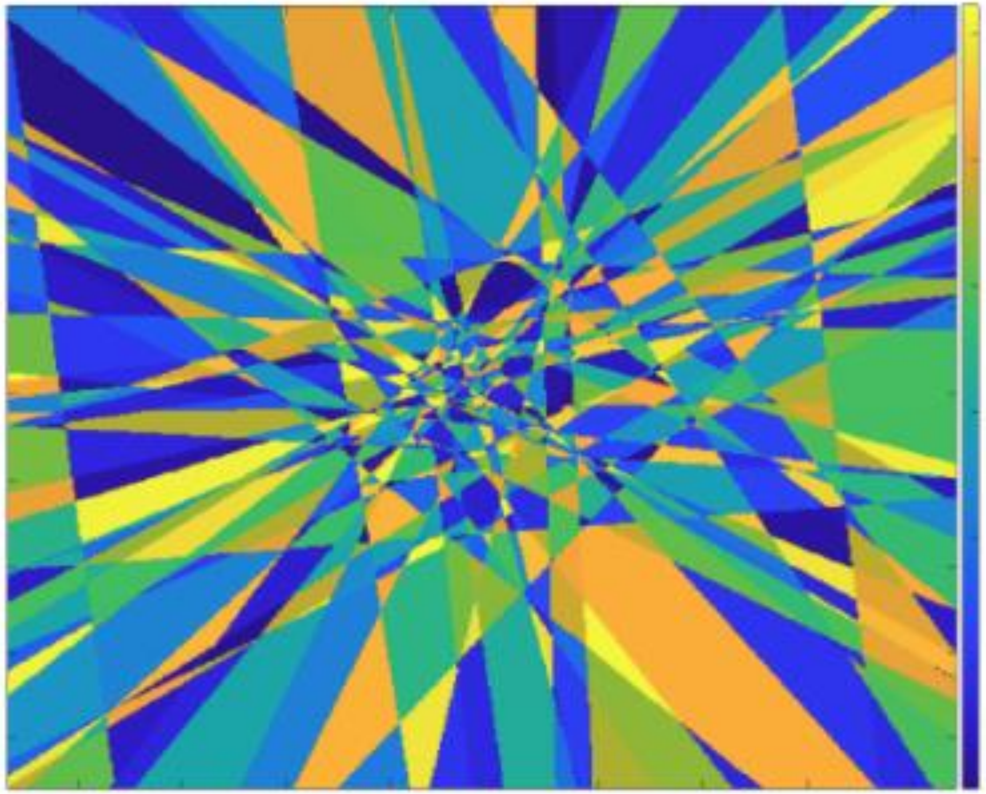
$$B(d_0, n) = \binom{n-1}{d_0}$$



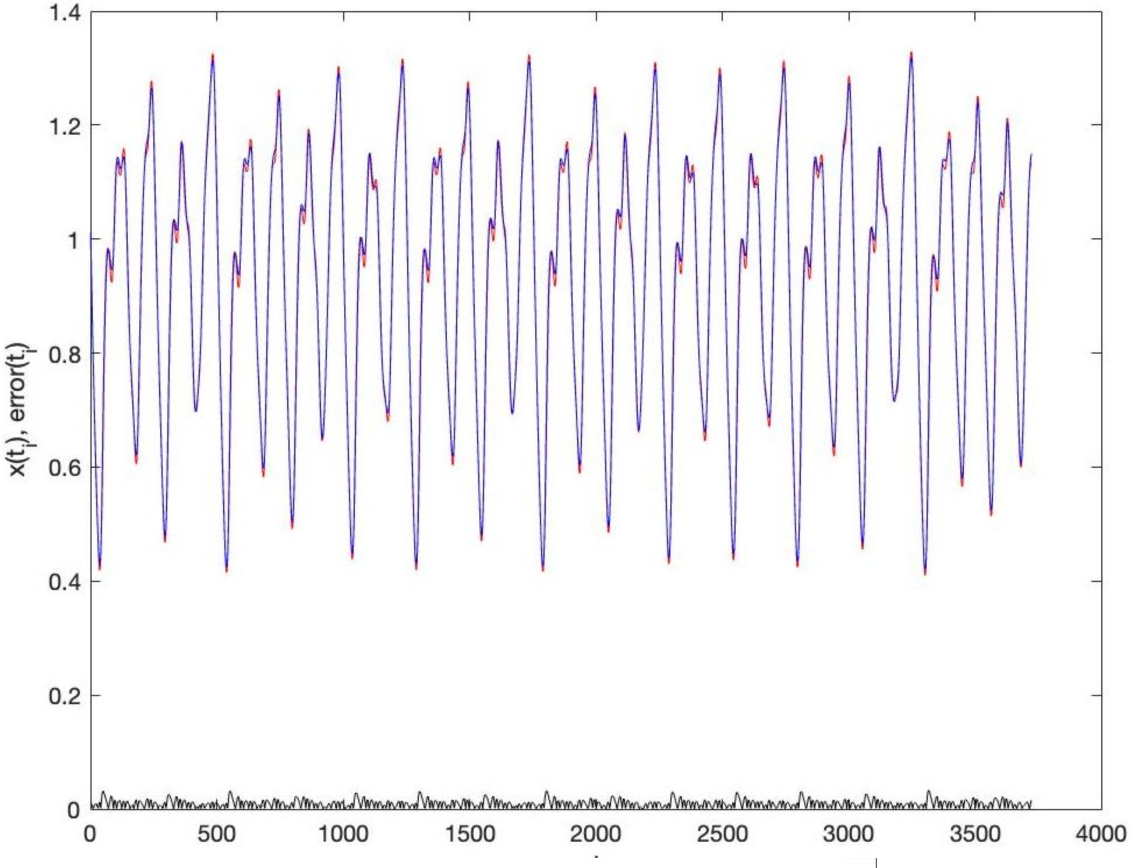
Expressive?

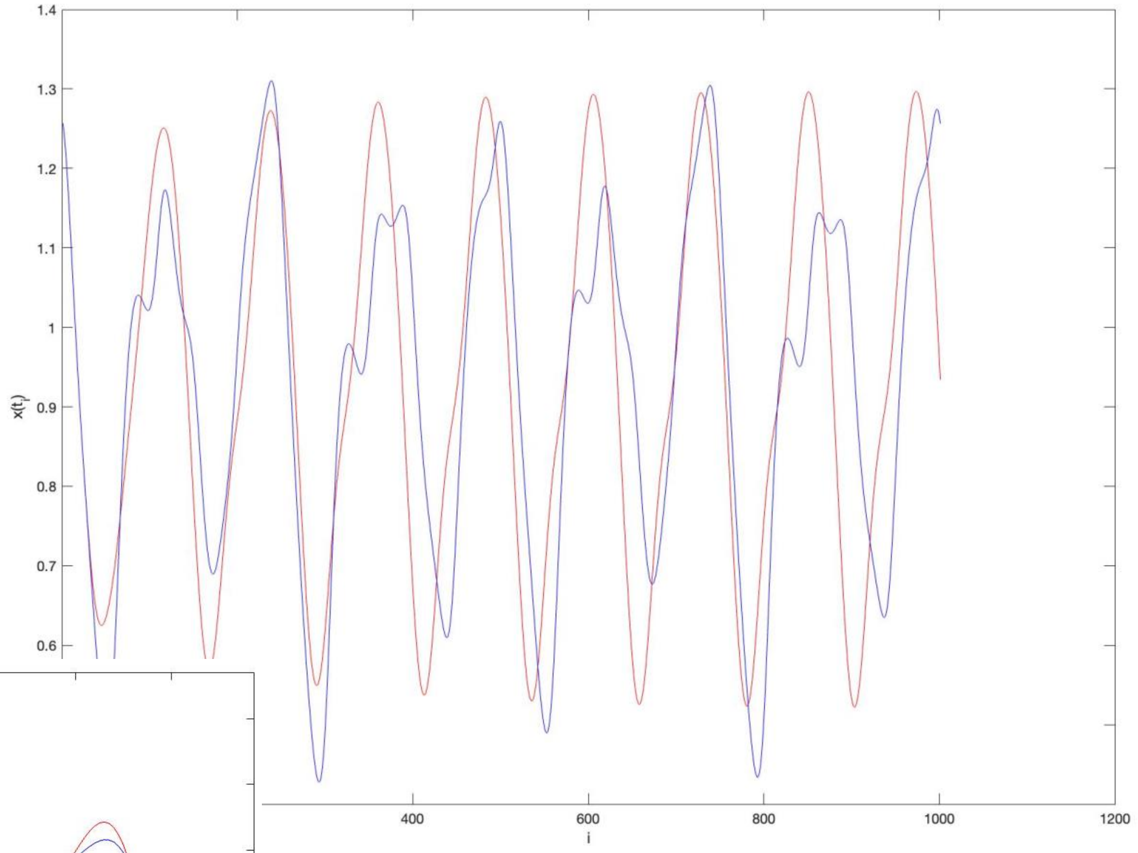


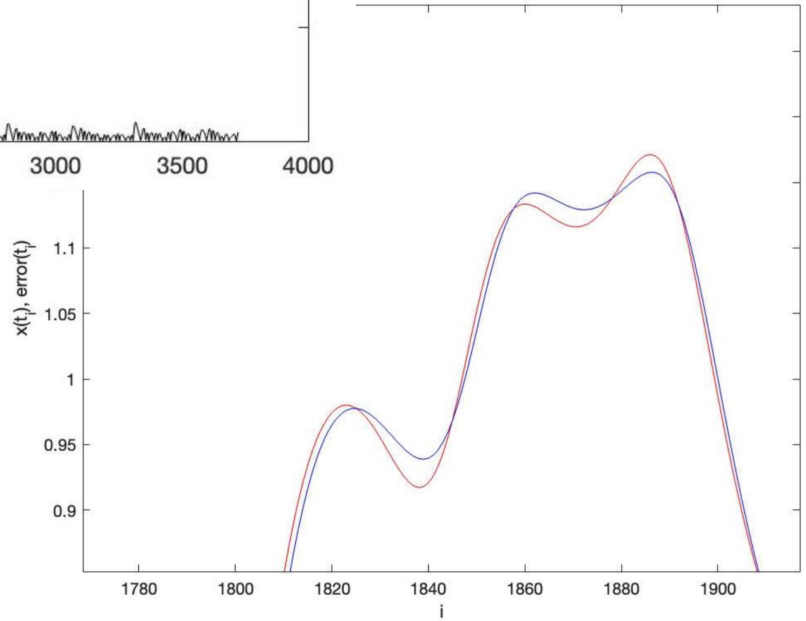
Expressive?



ELM forecasting, of Lasota Mackey, on x(t) and 14 delays.







Meh – so far – decent but not great.

EXISTENCE of the representation: **Wold theory** about zero mean covariance stationary vector processes
-there is a VMA - possibly infinite history => for invertible delay processes described by a VAR and approx by a VAR(k).

Theorem 1 **(Wold Theorem,** *A zero mean covariance stationary vector process* $\{\mathbf{x}_t\}$ *admits a representation,*

$$\mathbf{X}_t = C(L)\boldsymbol{\xi}_t + \boldsymbol{\mu}_t,$$

where $C(L) = \sum_{i=0}^{\infty} C_i L^i$ *is a polynomial delay operator polynomial, the* C_i *are the moving average matrices, and* $L^i(\boldsymbol{\xi}_t) = \boldsymbol{\xi}_{t-i}$. *The term* $C(L)\boldsymbol{\xi}$ *is the stochastic part of the decomposition. The* $\boldsymbol{\mu}_t$ *term is the deterministic (perfectly predictable) part as a linear combination of the past values of* $\mathbf{X}_t$. *Furthermore,*

- $\boldsymbol{\mu}_t$ *is a d-dimensional linearly deterministic process.*
- $\boldsymbol{\xi}_t \sim WN(0, \Omega)$ *is white noise.*
- *Coefficient matrices are square summable,*

$$\sum_{i=0}^{\infty} \|C_i\|^2 < \infty.$$

- $C_0 = I$ *is the identity matrix.*
- *For each t,* $\boldsymbol{\mu}_t$ *is called the innovation or the linear forecast errors.*

$$\mathbf{X}_t = C(L)\boldsymbol{\xi}_t \implies B(L)\mathbf{X}_t = \boldsymbol{\xi}_t,$$

Clarifying notation of the delay operator polynomial, with an example, let

$$C(L) = \begin{bmatrix} 1 & 1+L \\ -\frac{1}{2}L & \frac{1}{2} - L \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & \frac{1}{2} \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ -\frac{1}{2} & -1 \end{bmatrix}$$

$$L = C_0 + C_1 L, \text{ and } C_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \text{ if } i > 1;$$

therefore if, for example, $\mathbf{x}_t \in \mathbb{R}^2$,

$$C(L)\mathbf{x}_t = \begin{bmatrix} 1 & 1+L \\ -\frac{1}{2}L & \frac{1}{2} - L \end{bmatrix} \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix} = \begin{bmatrix} x_{1,t} + x_{2,t} + x_{2,(t-1)} \\ \frac{1}{2}x_{1,(t-1)} + \frac{1}{2}x_{2,t} - x_{2,(t-1)} \end{bmatrix}$$

Koopman Konnection - The RC can be written in a way that reminds us of DMD regression

$$\begin{bmatrix} | & | & \vdots & | \\ \mathbf{x}_{k+1} & \mathbf{x}_{k+2} & \dots & \mathbf{x}_N \\ | & | & \vdots & | \\ \mathbf{x}_k & \mathbf{x}_{k+1} & \dots & \mathbf{x}_{N-1} \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ \mathbf{x}_2 & \mathbf{x}_3 & \dots & \mathbf{x}_{N-k} \\ | & | & \vdots & | \end{bmatrix} = \mathcal{K} \begin{bmatrix} | & | & \vdots & | \\ \mathbf{x}_k & \mathbf{x}_{k+1} & \dots & \mathbf{x}_{N-1} \\ | & | & \vdots & | \\ \mathbf{x}_{k-1} & \mathbf{x}_k & \dots & \mathbf{x}_{N-2} \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_{N-k-1} \end{bmatrix},$$

"exact DMD" solution

$$\mathcal{K} = \arg\min_K \|\mathbb{X}' - K\mathbb{X}\|_F,$$

$$\mathcal{K} = \mathbb{X}'\mathbb{X}^\dagger,$$

|
V

$$\mathbb{X}' = \mathcal{K}\mathbb{X},$$

vs

$$\begin{bmatrix} | & | & | & | \\ \mathbf{y}_{k+1} & \mathbf{y}_{k+2} & \dots & \mathbf{y}_N \\ | & | & | & | \end{bmatrix} = \begin{bmatrix} [a_1] & [a_2] & \dots & [a_k] \end{bmatrix} \begin{bmatrix} | & | & \vdots & | \\ \mathbf{x}_k & \mathbf{x}_{k+1} & \dots & \mathbf{x}_{N-1} \\ | & | & \vdots & | \\ \mathbf{x}_{k-1} & \mathbf{x}_k & \dots & \mathbf{x}_{N-2} \\ | & | & \vdots & | \\ \vdots & \vdots & \vdots & \vdots \\ | & | & \vdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_{N-k-1} \\ | & | & \vdots & | \end{bmatrix}$$

In practice – train the linear RC to **polynomial readout** of **hidden r**

$$\mathbf{R}_1 = \left[\mathbf{r}_k \quad |\mathbf{r}_{k+1} \quad | \cdots \quad |\mathbf{r}_N\right],$$

| Hadamard product
V

$$\mathbf{R}_2 = \left[\mathbf{r}_k \circ \mathbf{r}_k \quad |\mathbf{r}_{k+1} \circ \mathbf{r}_{k+1} \quad | \cdots \quad |\mathbf{r}_N \circ \mathbf{r}_N\right]$$

$$\mathbf{R} = \begin{bmatrix}\mathbf{R}_1\\ \mathbf{R}_2\end{bmatrix}. \qquad \mathbf{W}^{out} = \begin{bmatrix}\mathbf{W}_1^{out}\\ \mathbf{W}_2^{out}\end{bmatrix} \qquad \mathbf{W}^{out} := \mathbf{X}\mathbf{R}^T(\mathbf{R}\mathbf{R}^T + \lambda\mathbf{I})^{-1}$$

Turns out this yields not a VAR but an **NVAR** – works much better! – Just like before **– iterate**......

$$Bw \circ Bw = (w_1\mathbf{b}_1 + w_2\mathbf{b}_2 + \cdots + w_n\mathbf{b}_n) \circ (w_1\mathbf{b}_1 + w_2\mathbf{b}_2 + \cdots + w_n\mathbf{b}_n)$$

"key trick"

$$= [\mathbf{b}_1 \circ \mathbf{b}_1 | \mathbf{b}_1 \circ \mathbf{b}_2 | \cdots | \mathbf{b}_n \circ \mathbf{b}_n] \begin{bmatrix} w_1^2 \\ w_1 w_2 \\ \vdots \\ w_1 w_n \\ w_2 w_1 \\ w_2^2 \\ \vdots \\ w_n^2 \end{bmatrix} := P_2(B, B)p_2(\mathbf{w}, \mathbf{w}).$$

$$\mathbf{r}_2 \circ \mathbf{r}_2 = (\mathbf{W}^{in}\mathbf{x}_1) \circ (\mathbf{W}^{in}\mathbf{x}_1)$$
$$= P_2(\mathbf{W}^{in}, \mathbf{W}^{in})p_2(\mathbf{x}_1),$$
$$\mathbf{r}_3 \circ \mathbf{r}_3 = (A\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{in}\mathbf{x}_2) \circ (A\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{in}\mathbf{x}_2)$$
$$= P_2(A\mathbf{W}^{in}, A\mathbf{W}^{in})p_2(\mathbf{x}_1, \mathbf{x}_1) + P_2(A\mathbf{W}^{in}, \mathbf{W}^{in})p_2(\mathbf{x}_1, \mathbf{x}_2)$$
$$+ P_2(\mathbf{W}^{in}, A\mathbf{W}^{in})p_2(\mathbf{x}_2, \mathbf{x}_1) + P_2(\mathbf{W}^{in}, \mathbf{W}^{in})p_2(\mathbf{x}_2, \mathbf{x}_2),$$
$$\vdots$$

$$P_2 : \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n^2},$$

$m \times n^2$ matrix of Hadamard products

$p_2(\mathbf{v}, \mathbf{w}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n^2}$, $n^2 \times 1$ vector of quadratic monomials

$$(\mathbf{v}, \mathbf{w}) \mapsto [v_1 w_1 | v_1 w_2 | \cdots | v_1 w_n | v_2 w_1 | v_2 w_2 | \cdots | v_n w_n]^T,$$

The iteration thing again,
Now gives monomials

$$
\begin{aligned}
\mathbf{r}_2 \circ \mathbf{r}_2 &= (\mathbf{W}^{in}\mathbf{x}_1) \circ (\mathbf{W}^{in}\mathbf{x}_1) \\
&= P_2(\mathbf{W}^{in}, \mathbf{W}^{in}) p_2(\mathbf{x}_1), \\
\mathbf{r}_3 \circ \mathbf{r}_3 &= (A\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{in}\mathbf{x}_2) \circ (A\mathbf{W}^{in}\mathbf{x}_1 + \mathbf{W}^{in}\mathbf{x}_2) \\
&= (A\mathbf{W}^{in}\mathbf{x}_1) \circ (A\mathbf{W}^{in}\mathbf{x}_1) + (A\mathbf{W}^{in}\mathbf{x}_1) \circ (\mathbf{W}^{in}\mathbf{x}_2) \\
&\quad + (\mathbf{W}^{in}\mathbf{x}_2) \circ (A\mathbf{W}^{in}\mathbf{x}_1) + (\mathbf{W}^{in}\mathbf{x}_2) \circ (\mathbf{W}^{in}\mathbf{x}_2) \\
&= P_2(A\mathbf{W}^{in}, A\mathbf{W}^{in}) p_2(\mathbf{x}_1, \mathbf{x}_1) + P_2(A\mathbf{W}^{in}, \mathbf{W}^{in}) p_2(\mathbf{x}_1, \mathbf{x}_2) \\
&\quad + P_2(\mathbf{W}^{in}, A\mathbf{W}^{in}) p_2(\mathbf{x}_2, \mathbf{x}_1) + P_2(\mathbf{W}^{in}, \mathbf{W}^{in}) p_2(\mathbf{x}_2, \mathbf{x}_2), \\
&\vdots \\
\mathbf{r}_{k+1} \circ \mathbf{r}_{k+1} &= \sum_{i=1}^{k} (A^{i-1}\mathbf{W}^{in}\mathbf{x}_{k+1-i}) \circ \left(\sum_{j=1}^{k} A^{j-1}\mathbf{W}^{in}\mathbf{x}_{k+1-j} \right) \\
&= \sum_{i,j=1}^{k} P_2(A^{i-1}\mathbf{W}^{in}, A^{j-1}\mathbf{W}^{in}) p_2(\mathbf{x}_{k+1-i}, \mathbf{x}_{k+1-j}) \\
&:= \mathbb{A}_2[\mathbb{X}_2]_k.
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{A}_2 = [&P_2(\mathbf{W}^{in}, \mathbf{W}^{in}) | P_2(A\mathbf{W}^{in}, \mathbf{W}^{in}) | P_2(A^2\mathbf{W}^{in}, \mathbf{W}^{in}) | \cdots \\
&\cdots | P_2(A^{k-1}\mathbf{W}^{in}, \mathbf{W}^{in}) | P_2(\mathbf{W}^{in}, A\mathbf{W}^{in}) | P_2(A\mathbf{W}^{in}, A\mathbf{W}^{in}) \\
&\times | P_2(A^2\mathbf{W}^{in}, A\mathbf{W}^{in}) | \cdots \\
&\cdots | P_2(A^{k-2}\mathbf{W}^{in}, A^{k-1}\mathbf{W}^{in}) | P_2(A^{k-1}\mathbf{W}^{in}, A^{k-1}\mathbf{W}^{in})]
\end{aligned}
$$

is a $d_r \times kd_x^2$ matrix.