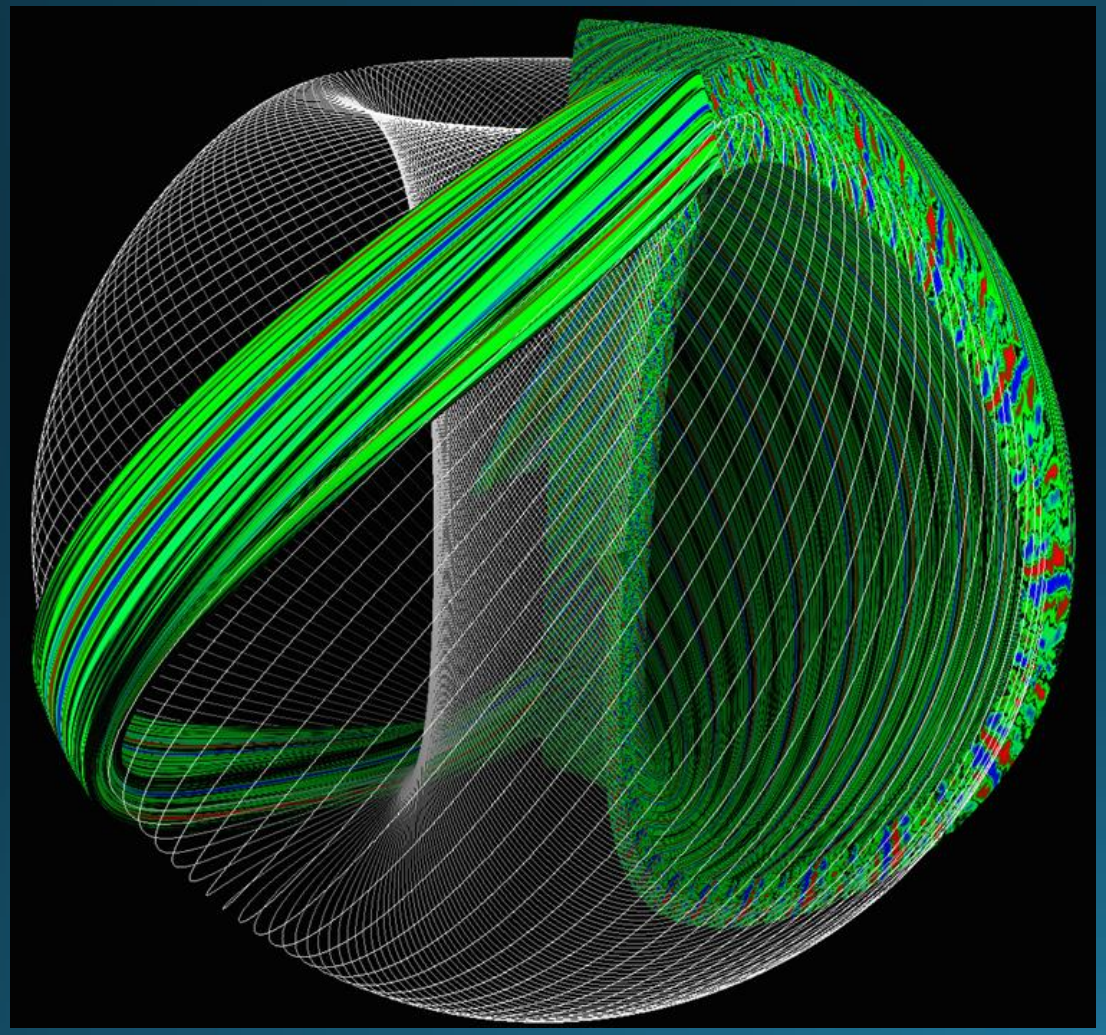
Modeling turbulence in tokamak plasmas with reservoir computing

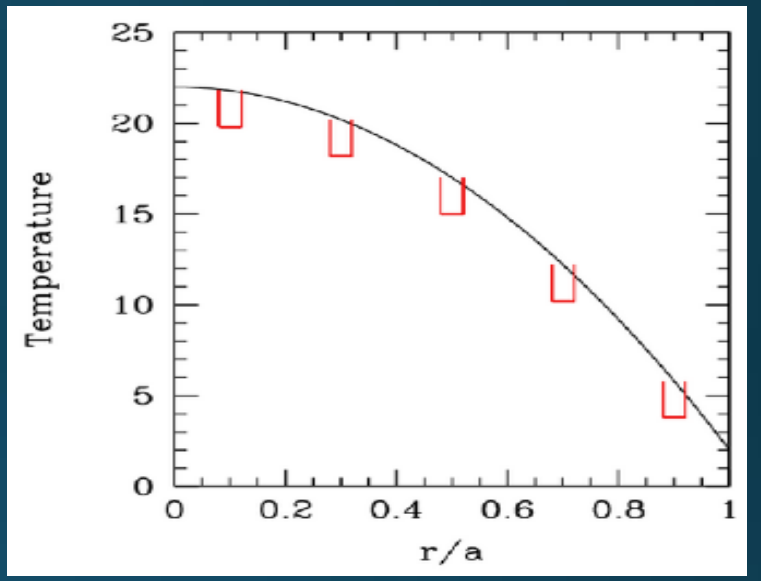
Nathaniel Barbour
Advisor: Professor Bill Dorland
Department of Physics

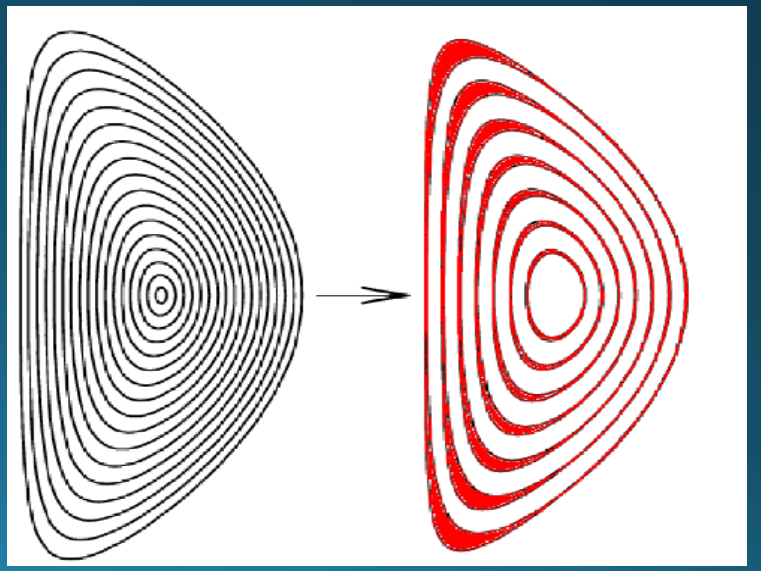
Overview for today

- Recap of full project
- Reservoir implementation
- Current results
- Plan for project completion

Gyrokinetic modeling







Big Picture

- **Guiding question:** Can we accurately predict turbulent heat flux more quickly than a direct numerical solution?
- Facilitates comparisons of potential future fusion experiments before construction
- Turbulence simulation -> fast scale (~10^5 Hz)
- Radial heat diffusion -> slow scale (~4 Hz)

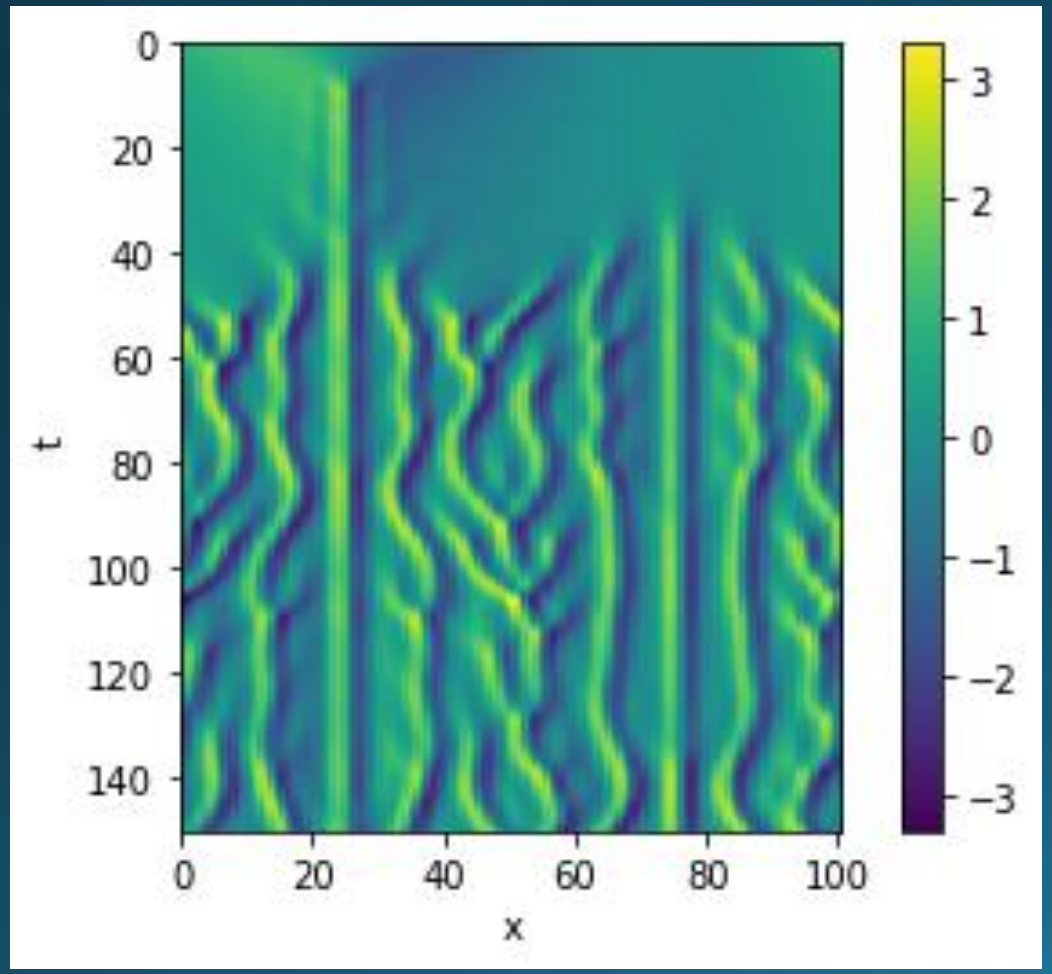
- Replace PDE solver with matrix multiplication and an element-wise vector operation
- Test method on smaller, relevant problem first

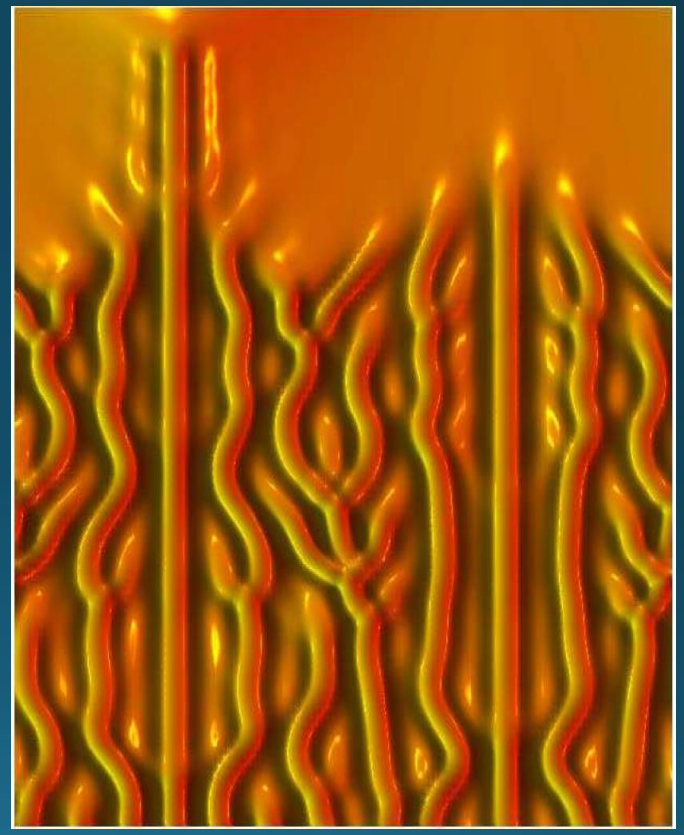
1-D Kuramoto-Sivashinsky (KS) Equation

$$u_t + u_{xxxx} + u_{xx} + uu_x = 0$$

- $u(x,t)$ is periodic on $[0, L)$
- Arises in plasma physics: trapped ion mode instabilities
- Quadratic nonlinear term
- Higher-order dissipation
- Nontrivial chaotic dynamics
- Implemented spectral RK_4 solver in C++

C++ Solution vs Published Solution





Reservoir Computing Overview

- Recurrent ANN with distinctions such as:
  - Connections defined by **random and sparse** adjacency matrix.
  - Input and internal **weights remain fixed.**
- Advantages:
  - Simpler training process
  - Output parameters can quickly be reused at a later time.
- Disadvantages:
  - Like other ANN methods, black box
  - Reservoir size scales with problem size unfavorably

Single Reservoir Implementation

- Input of dimension N
- Network of D_R neuron units with sparse adjacency matrix **A**
  - Elements drawn from uniform[0,1]
- $\mathbf{W}_{in}$ Input coupling matrix of dimension D_R x N
  - Elements drawn from uniform[-σ,σ]
- $\mathbf{W}_{out}$ Output coupling matrix of dimension N x D_R
  - N row vectors of parameters trained by linear regression
- State vector $\mathbf{r}(t + \Delta t) = \tanh[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{in}\mathbf{u}(t)]$
- $\tilde{\mathbf{u}}_R = \mathbf{W}_{out}\mathbf{r}$
- Implemented in Python using NumPy and SciPy

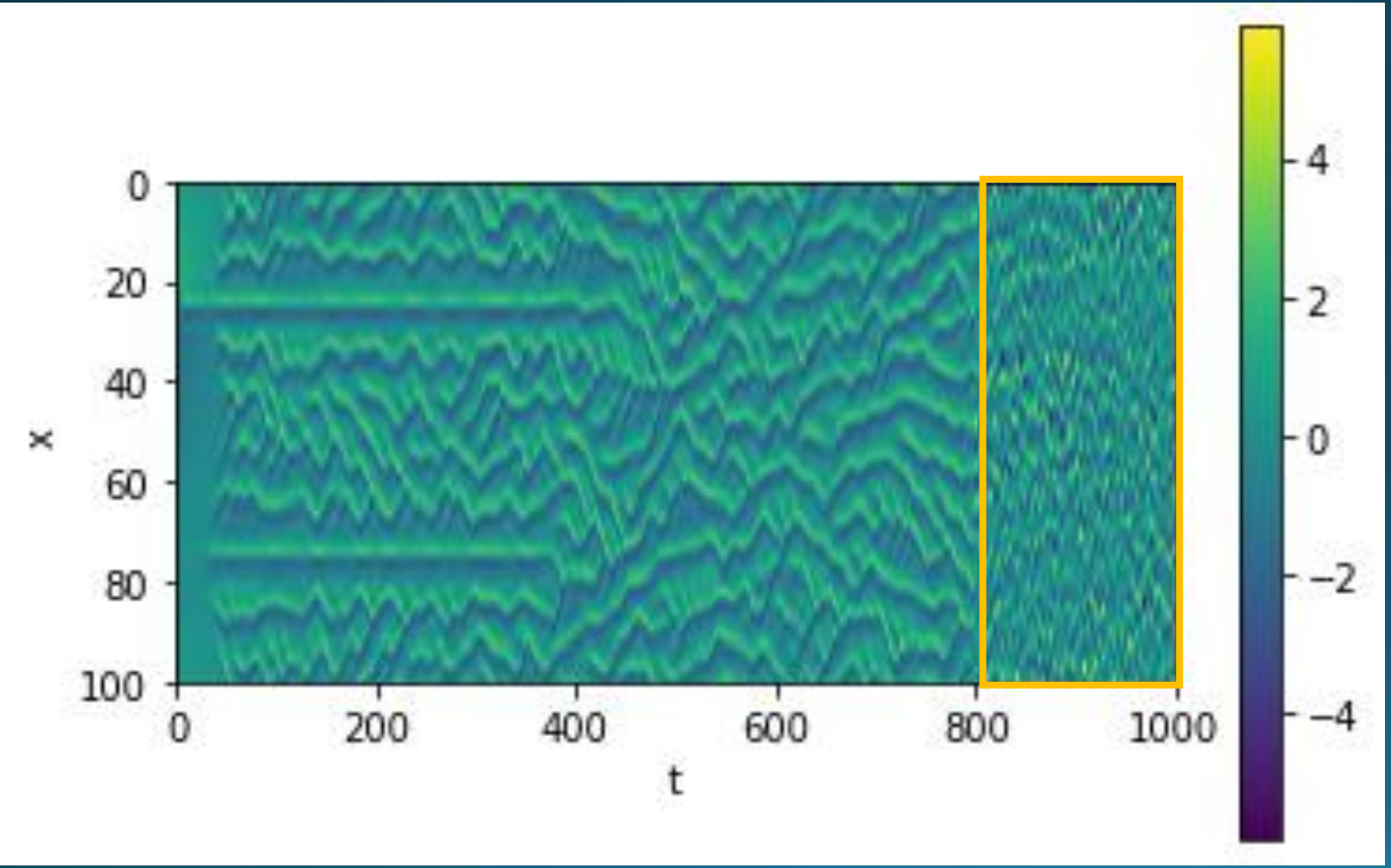
Training

- Tikhonov-Regularized Linear Regression
- Minimize:

$$\sum_{m=0}^{T/\Delta t} \|\mathbf{u}(m\Delta t) - \tilde{\mathbf{u}}_R(m\Delta t)\|^2 + \beta \|\mathbf{W}_{\text{out}}\|^2$$

- Regularization parameter to mitigate potential overfitting
- Chose β = 1e-4 consistent with Pathak et al.
- $\mathbf{W}_{\text{out}} = \mathbf{U}\mathbf{R}^{\mathbf{T}}\left(\mathbf{R}\mathbf{R}^{\mathbf{T}} + \beta\mathbf{I}\right)^{-\mathbf{1}}$

Results from December 2020

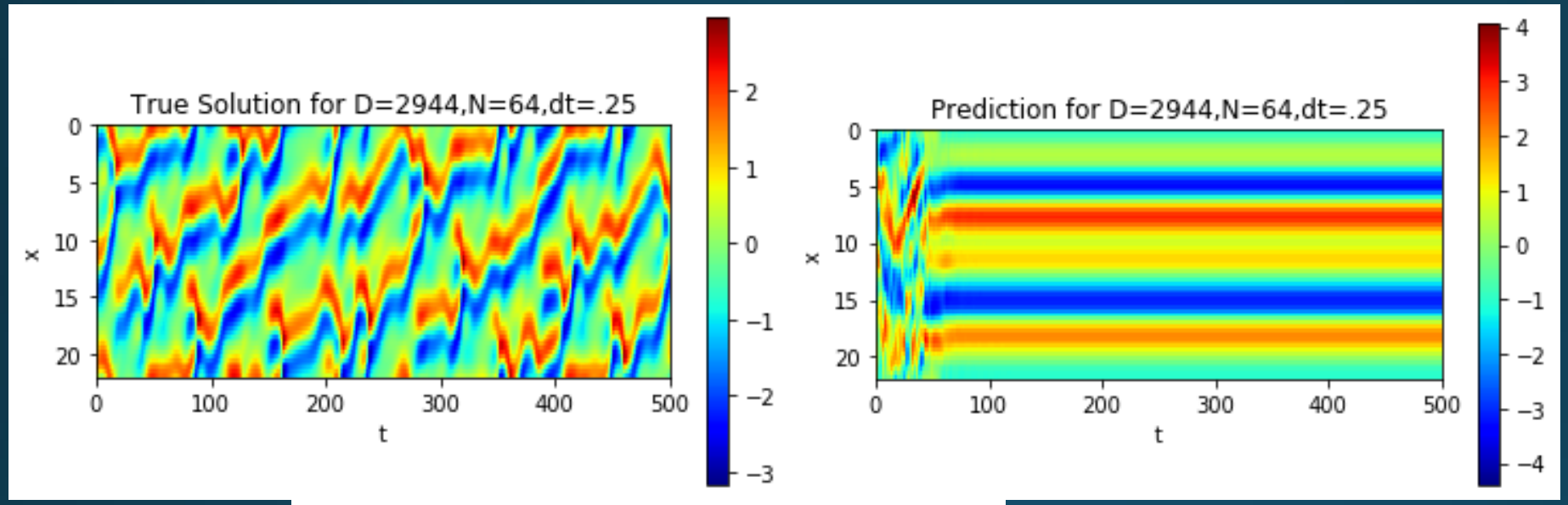


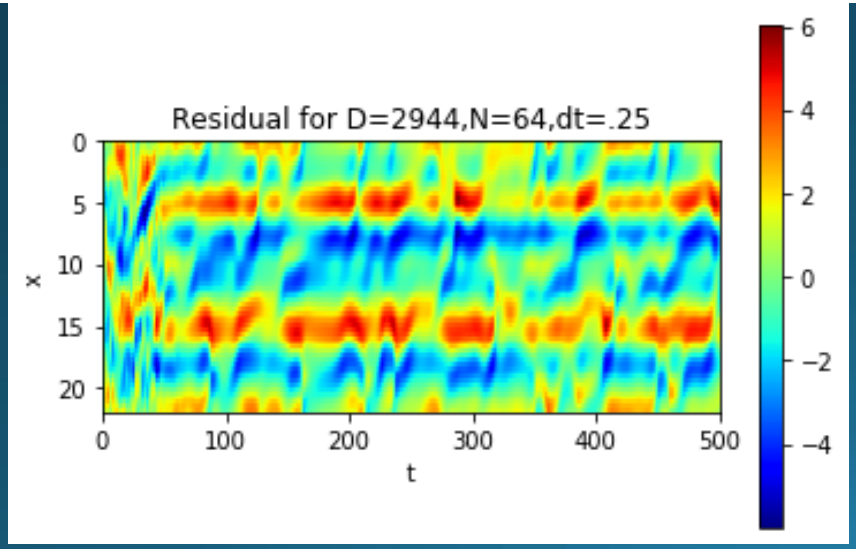
- High spatial frequency
- Some characteristic structure preserved

Potential Solutions

- Use larger ratio of reservoir units to spatial points
- Tune the hyperparameters of the model
- Break symmetry in the reservoir solution consistent with Pathak et al. 2018
- Authors posit that odd symmetry in tanh function creates artificial symmetry not present in KS system
- $\mathbf{r}_i^* = \mathbf{r}_i$ for even i and $\mathbf{r}_i^2$ for odd i
- $\tilde{\mathbf{u}}_R = \mathbf{W}_{\mathbf{out}}\mathbf{r}^*$
- $\mathbf{W}_{\mathrm{out}} = \mathbf{U}\mathbf{R}^{*\mathbf{T}}\left(\mathbf{R}^*\mathbf{R}^{*\mathbf{T}} + \beta\mathbf{I}\right)^{-\mathbf{1}}$

Results with modifications





Training MSE: 7.87e-4

Solution: Redefined W_{in}

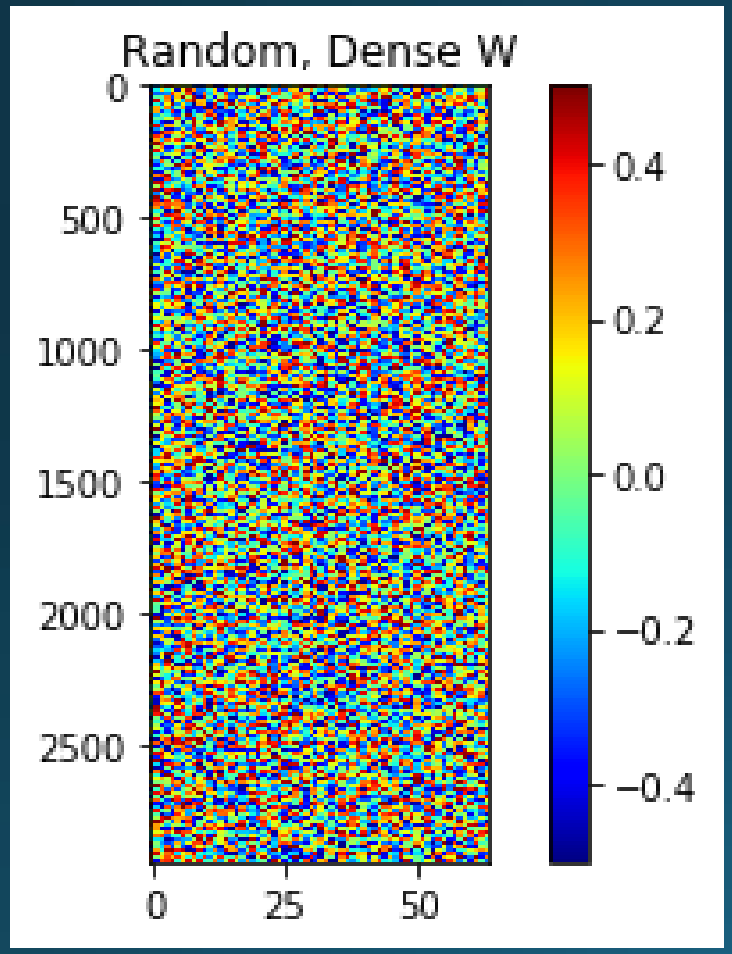
- Contacted author who forwarded public repository of Matlab code for the project
- W_{in} in author's code defined differently from straightforward paper description

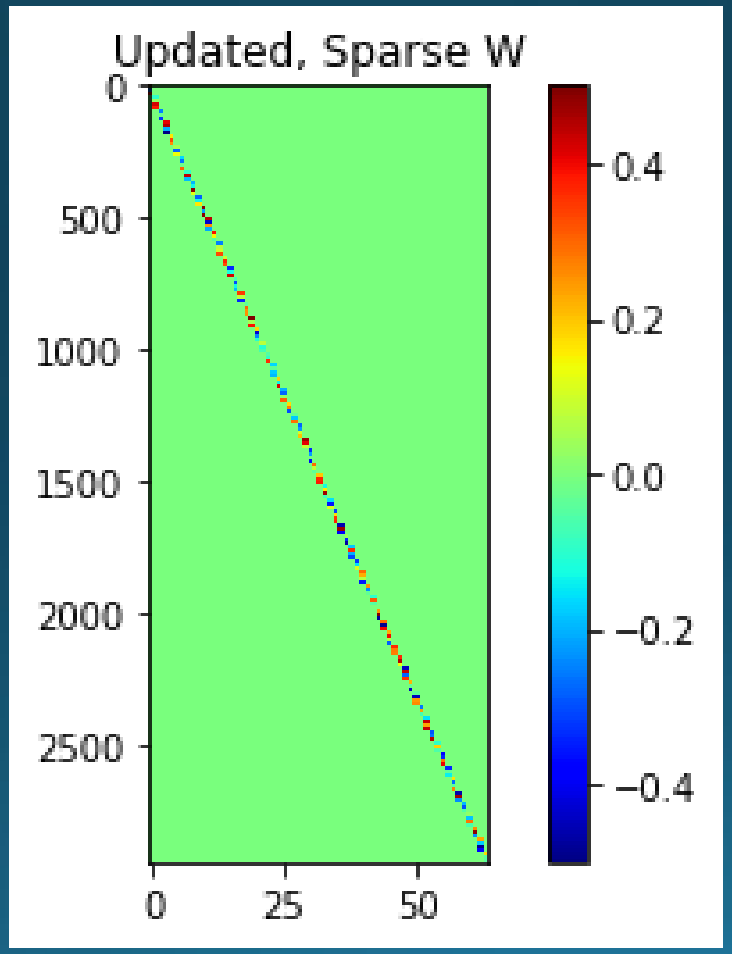
Ref. [8]. (We emphasize here that our choices are illustrative and that many others are possible.) The I/R coupler is $\mathbf{W}_{\text{in}}(\mathbf{u}) = \mathbf{W}_{\text{in}}\mathbf{u}$ (where $\mathbf{W}_{\text{in}}$ is a $D_r \times D_{\text{in}}$ matrix whose input elements are drawn from a uniform distribution in $[-\sigma, \sigma]$). The reservoir is a large, low-degree (κ), directed

```
function [x, wout, A, win] = train_reservoir(resparams, data)

A = generate_reservoir(resparams.N, resparams.radius, resparams.degree);
q = resparams.N/resparams.num_inputs;
win = zeros(resparams.N, resparams.num_inputs);
for i=1:resparams.num_inputs
    rng(i)
    ip = resparams.sigma*(-1 + 2*rand(q,1));
    win((i-1)*q+1:i*q,i) = ip;
end
%win = resparams.sigma*(-1 + 2*rand(resparams.N, resparams.num_inputs));
```

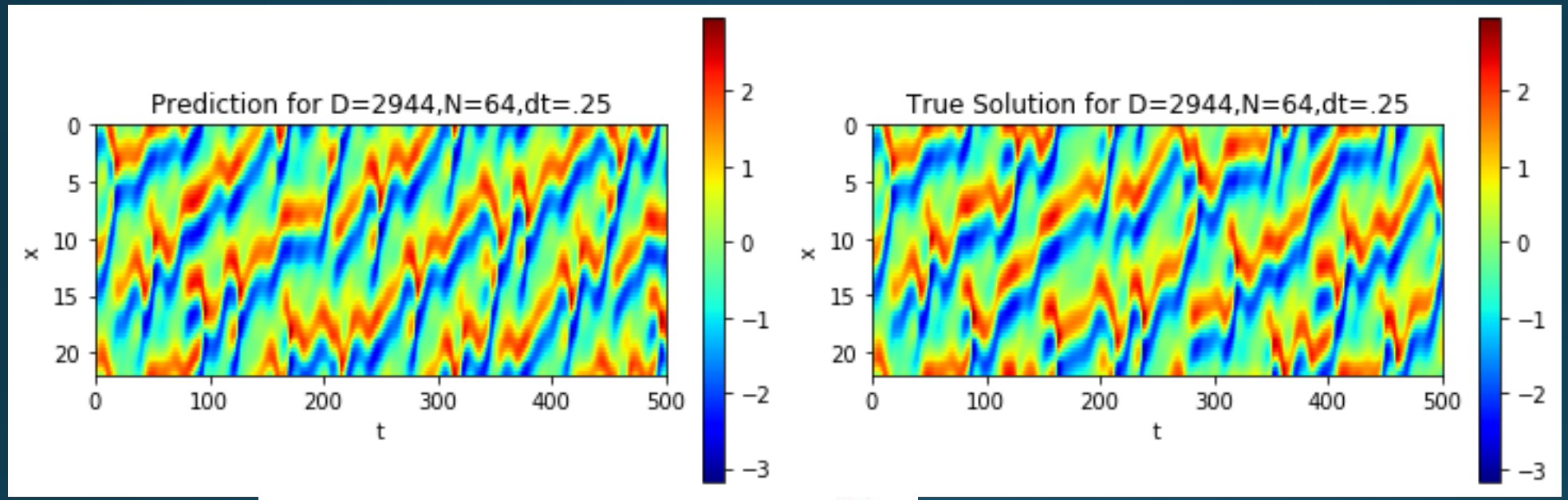
Comparison of Input Coupling Matrices

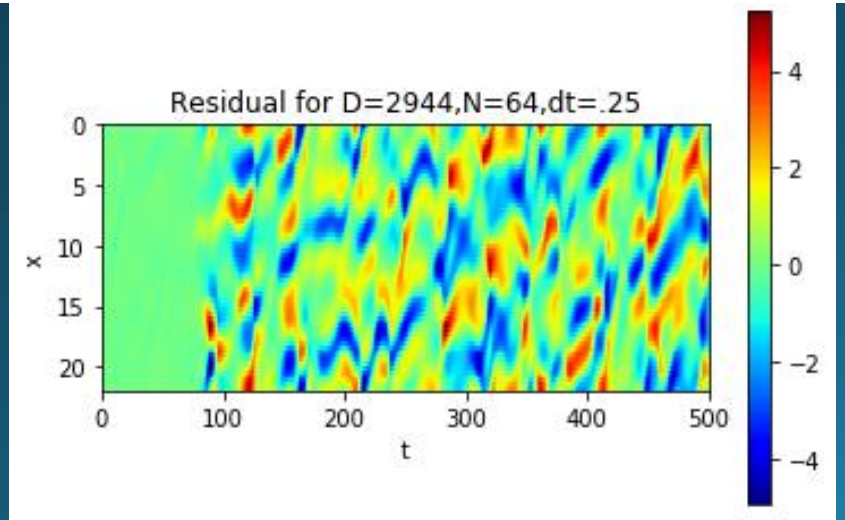




For each $\mathbf{W_{in}}$ has

Results for Sparse Input Coupling





Training MSE: 1.988e-06

Accurate predictions for duration consistent with Pathak et al.

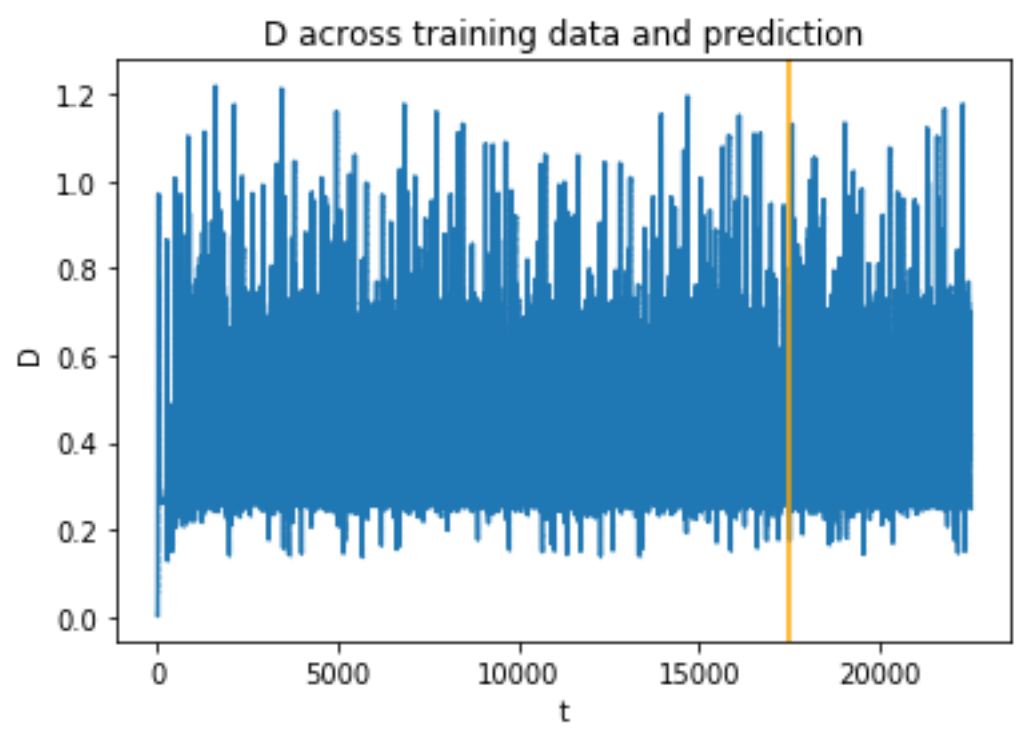
Physical Intuition of u(x,t)

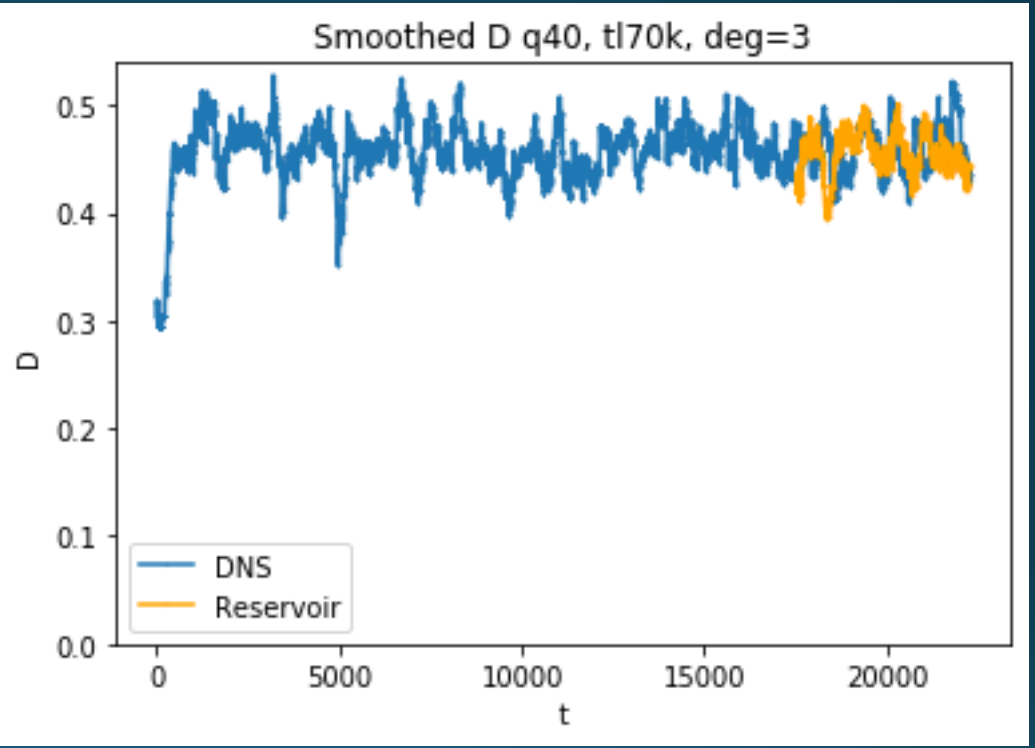
- Domain is periodic 1-D line, approx. a circle
- Calculate normalized electrostatic potential
- Electric field is tangential, Magnetic field is into the plane
- D is a diffusion coefficient.

With the fluctuating electrostatic potentials driven by the instability calculated in the two limits $\alpha \sim 1$ and $\alpha \ll 1$, we can compute the particle transport by use of the expression

$$D = \frac{\epsilon^{1/2}}{\nu_-}\left(\frac{cT}{eB}\right)^2\left\langle\left(\frac{\partial\Phi}{\partial y}\right)^2\right\rangle = \frac{\epsilon^{3/2}}{\nu_-}\left(\frac{\omega_0}{\nu_-}\right)^2\left(\frac{cT}{er B}\right)^2\left\langle\left(\frac{\partial\psi}{\partial\xi}\right)^2\right\rangle \qquad (19)$$

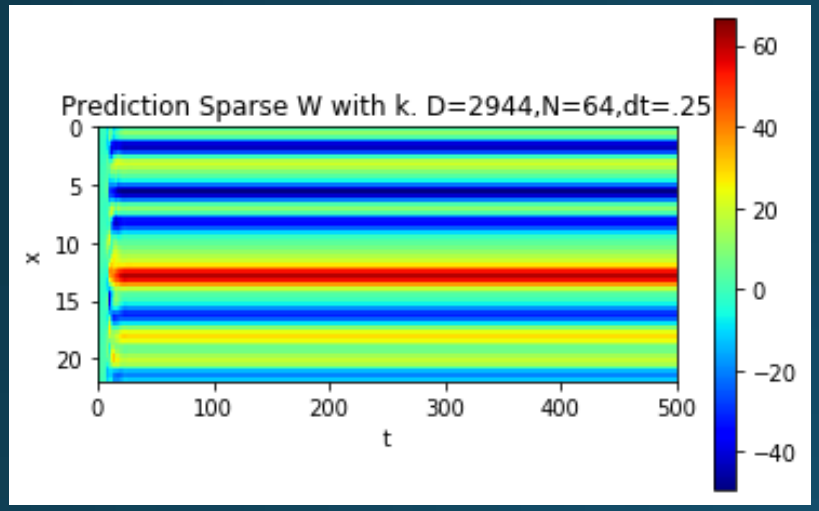
Diffusion Coefficient for KS

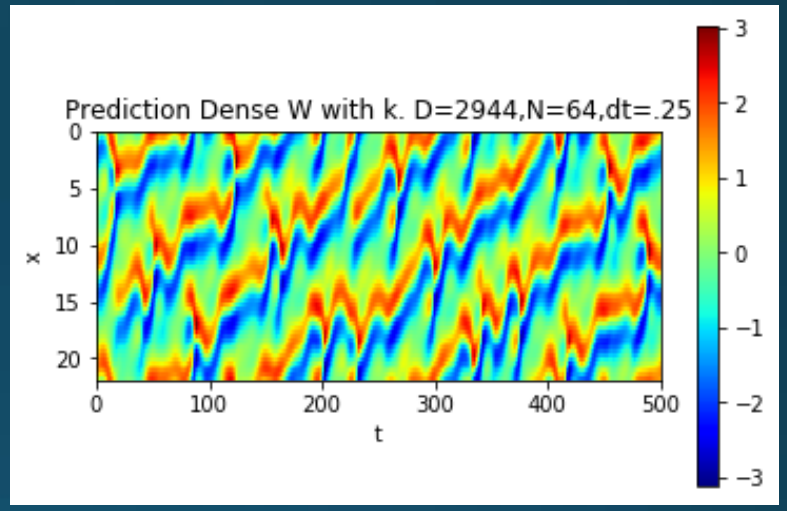


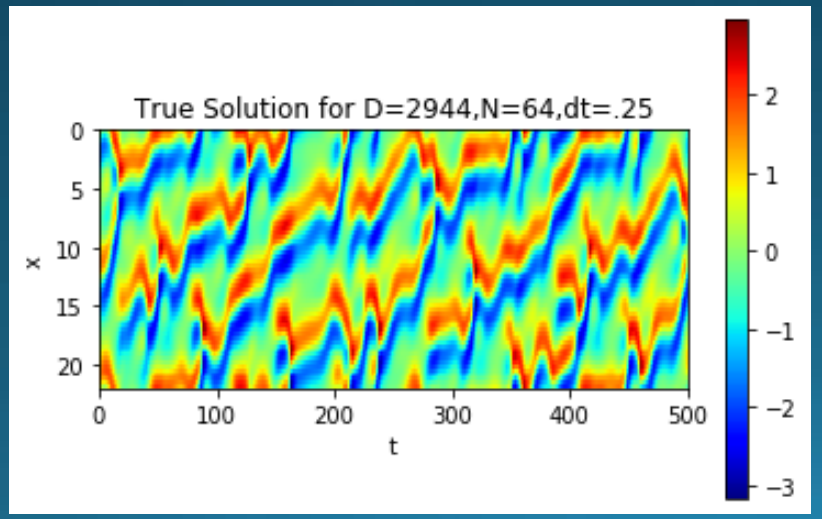


Reservoir diffusion coefficient mean: 0.45387
DNS diffusion coefficient mean: 0.45651
Error: 0.59%

Fourier space prediction fails without dense input coupling



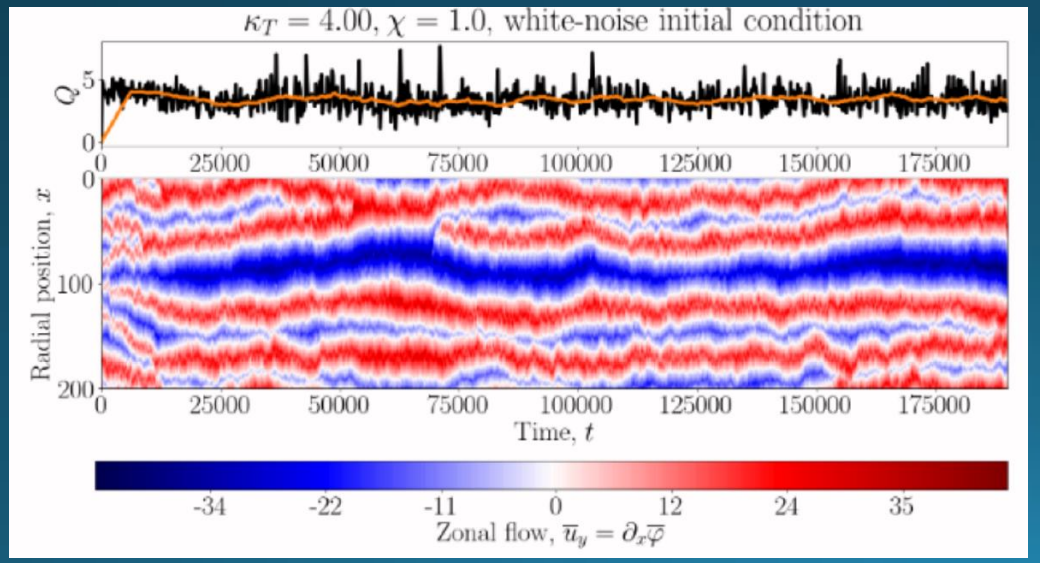




2-D system (Ivanov et al. 2020)

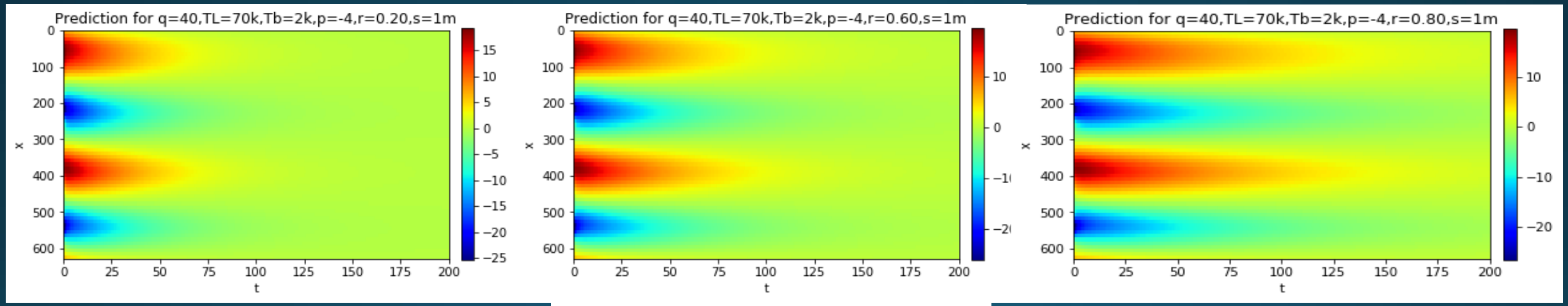
- High-collision limit of the gyrokinetic equation
- Ignores density gradient
- No variation permitted in direction of magnetic field lines

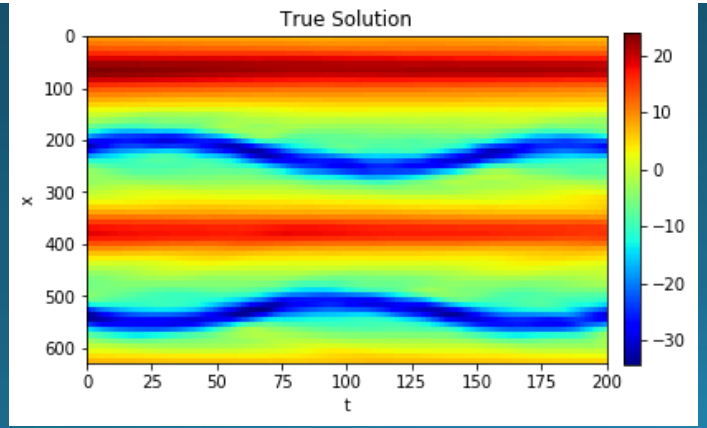
$$\frac{\partial}{\partial t}\left(h - \frac{ZeF_i}{T_i}\langle\phi\rangle_R\right) + \boldsymbol{V}_D \cdot \frac{\partial h}{\partial \boldsymbol{R}} + \langle \boldsymbol{V}_E\rangle_R \cdot \left[\frac{\partial h}{\partial \boldsymbol{R}} - \hat{\boldsymbol{x}}\left(\frac{v^2}{v_{ti}^2} - \frac{3}{2}\right)\frac{F_i}{L_T}\right] = \langle C_l[h]\rangle_R. \quad \text{(A 2)}$$



Spectral radius scan (high β)

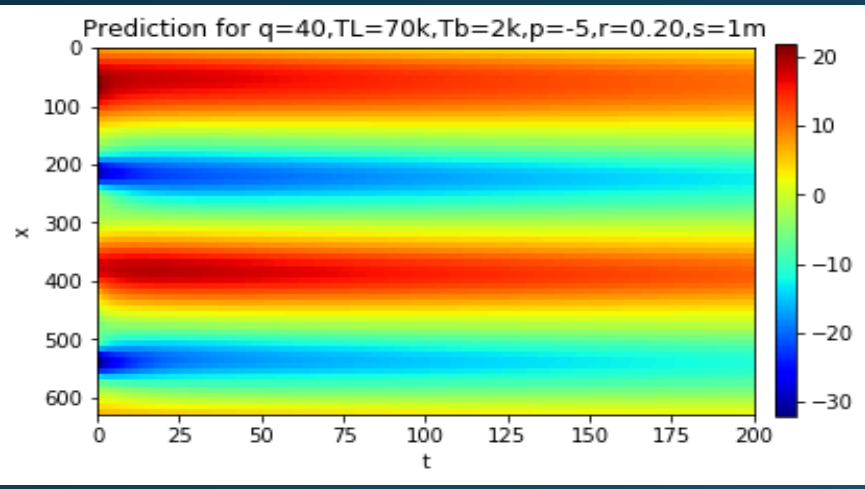
- Isolated impact of adjacency matrix using low input weights
- As anticipated, spectral radius controls decay rate of solution

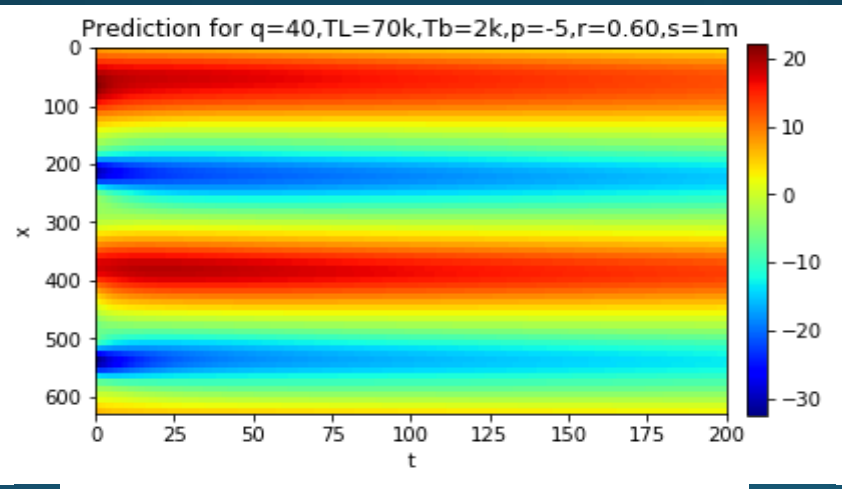


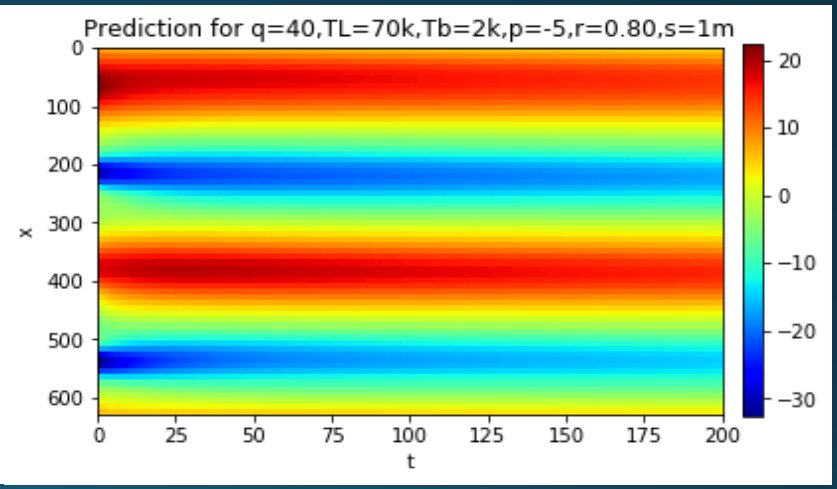


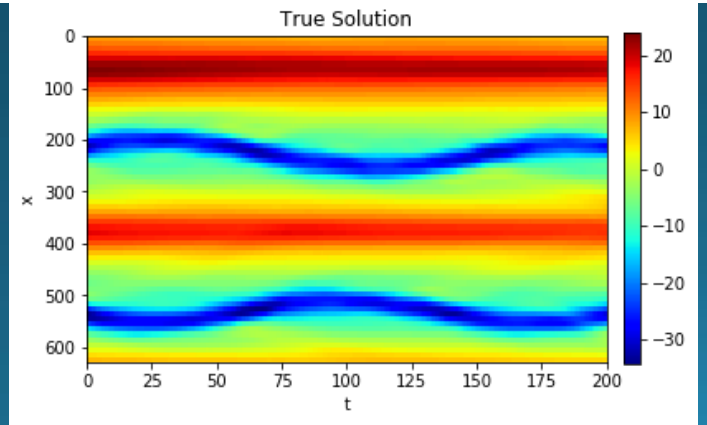
Spectral radius scan (low β)

- Isolated impact of adjacency matrix using low input weights
- Again, spectral radius controls decay rate of solution



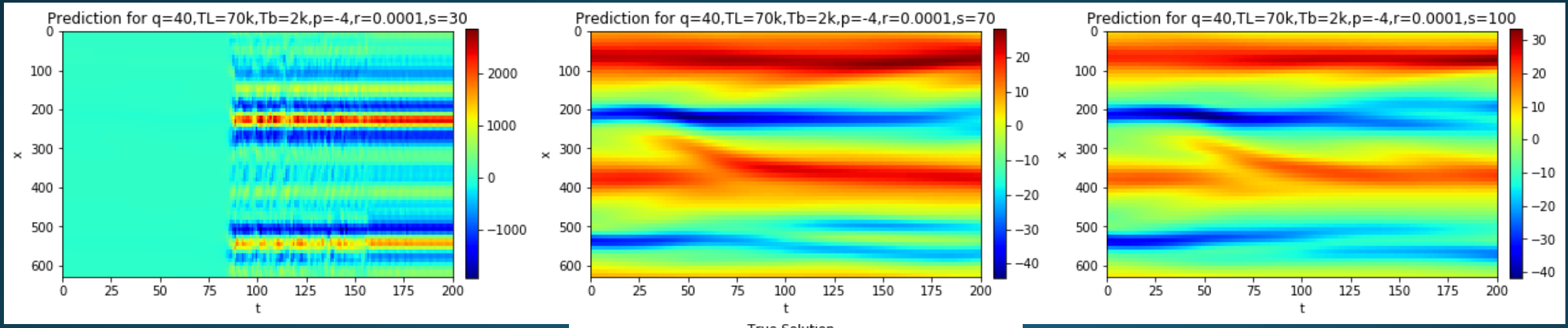


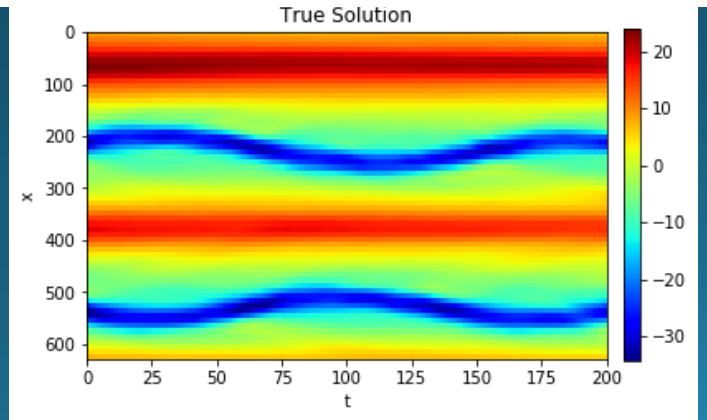




Input coupling scan (high β)

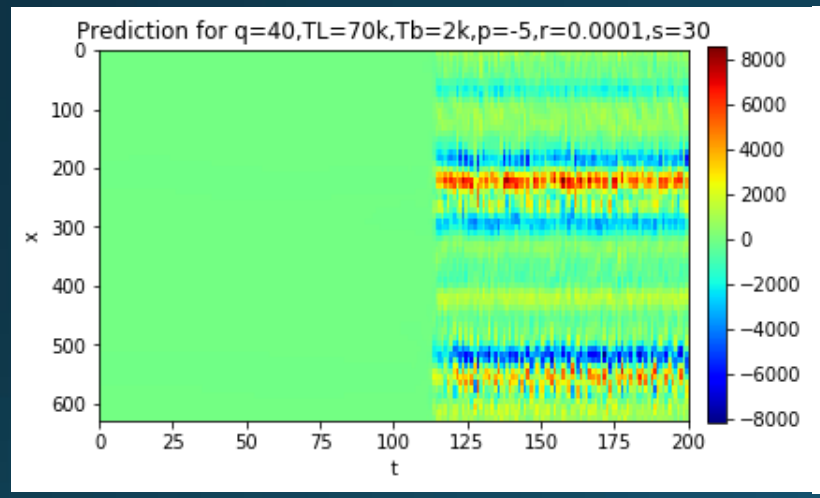
- Isolated impact of input coupling using low spectral radius
- Input coupling weights must scale with range of training data

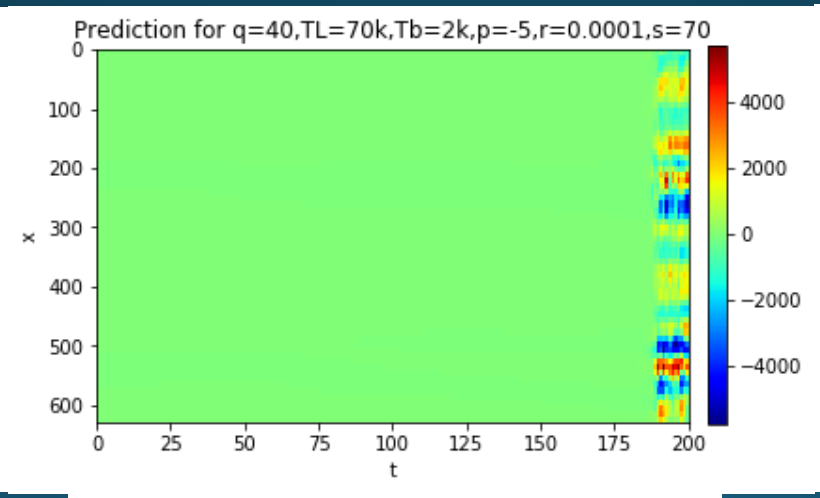


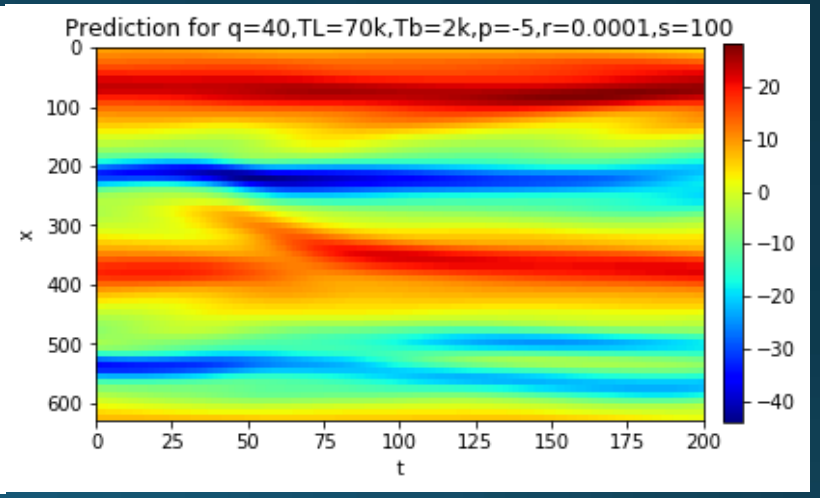


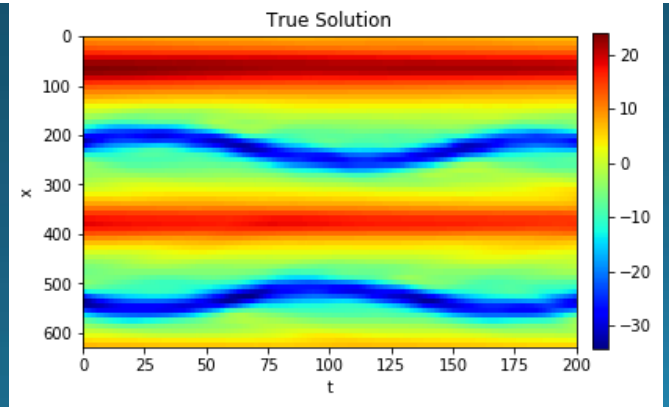
Input coupling scan (low β)

- Isolated impact of input coupling using low spectral radius
- Input coupling weights must scale with range of training data

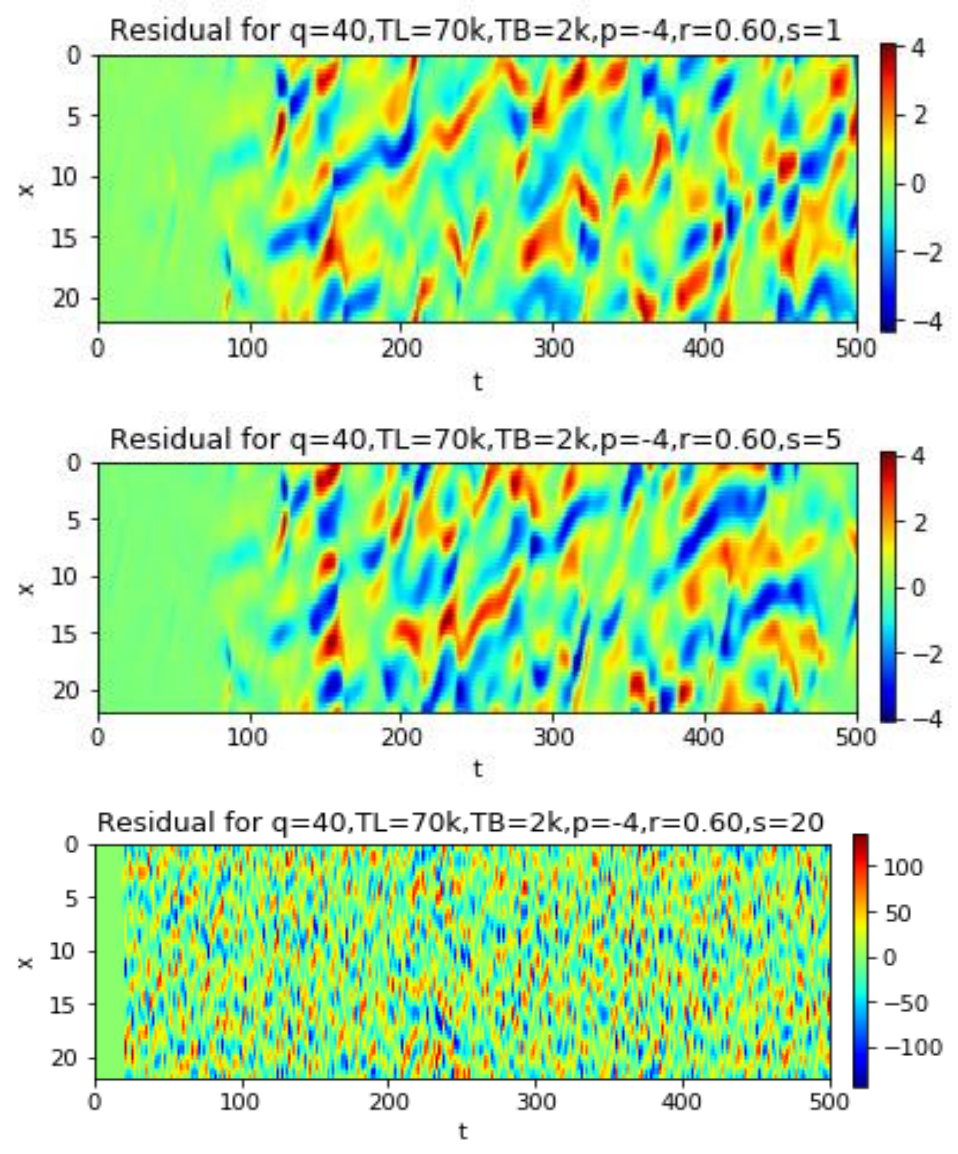


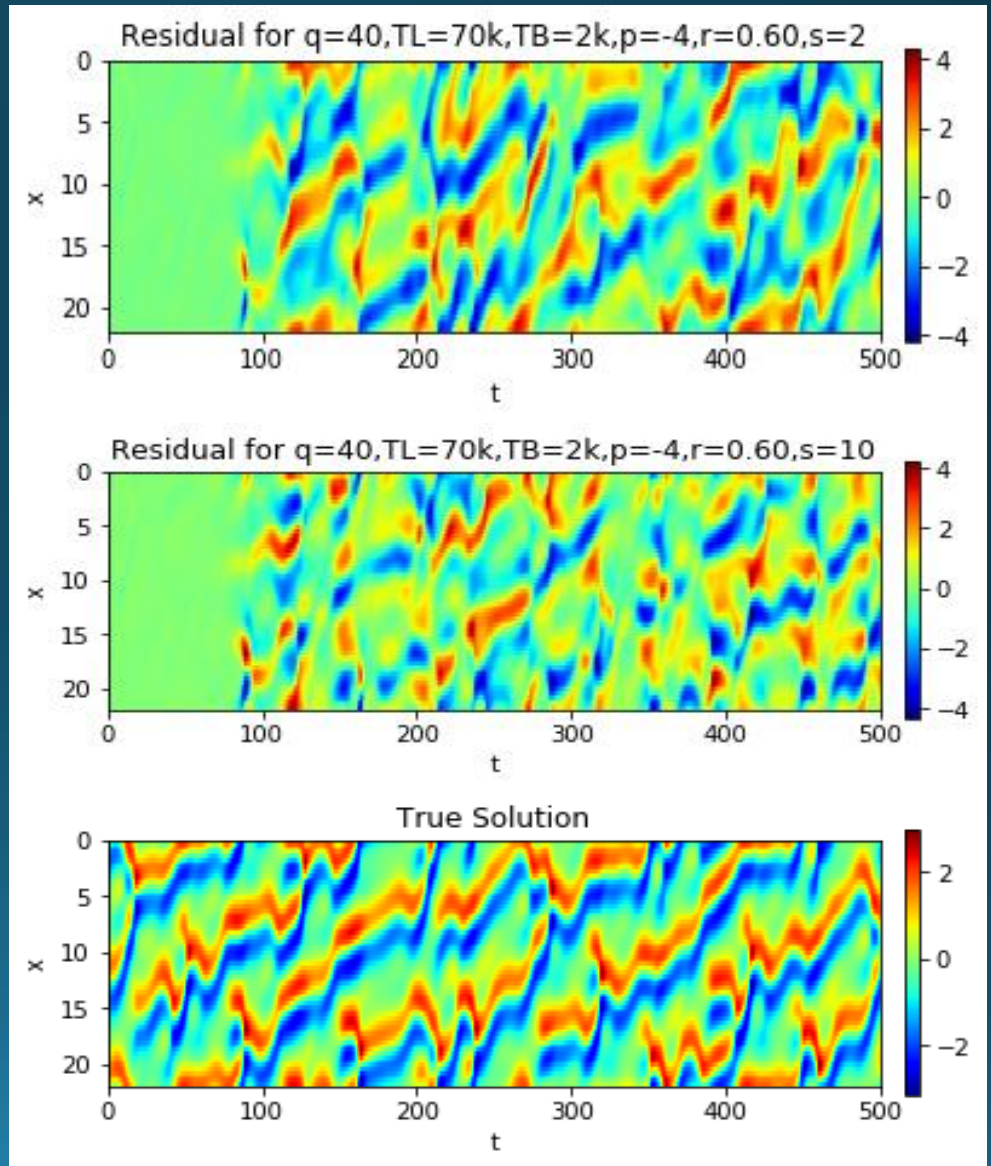






Input coupling scan (KS)





Next Steps

- Finalize CUDA implementation for speed improvement in large system
- Complete the hyperparameter scan to improve higher dimensional case
- Implement reservoir for full 5-D gyrokinetic system
- Implement hybrid reservoir for coarser time-resolution in DNS
- Further investigate generalizability of reservoir training

References

- M. Barnes. Ph.D. Thesis (2008) arxiv:0901.2868
- W. Dorland et al. *Phys. Rev. Lett*. **85**, 5579 (2000).
- P. Ivanov et al. (2020). *J. Plas. Phys.*, **86**(5), 855860502 (2020). doi:10.1017/S0022377820000938
- J. Jiang and Y. Lai. *Phys. Rev. Research* **1**, 033056 (2019).
- A. Kassam and L. N. Trefethen *SIAM J. Sci. Comput.*, **26**, 4 (2005).
- R. E. LaQuey et al. *Phys. Rev. Lett.* **34**, 391 (1975).
- J. Pathak et al. Chaos **28**, 041101 (2018).
- J. Pathak et al. *Phys. Rev. Lett.* **120**, 024102 (2018).