



TEXAS A&M UNIVERSITY  
Engineering



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

# Petrobras Workshop: Scientific Machine Learning (SciML) and Data-Driven Model Reduction for Reservoir Simulation

Eduardo Gildin  
Petroleum Engineering Department at  
Texas A&M University

August 11-15, 2025



TEXAS A&M UNIVERSITY  
Engineering



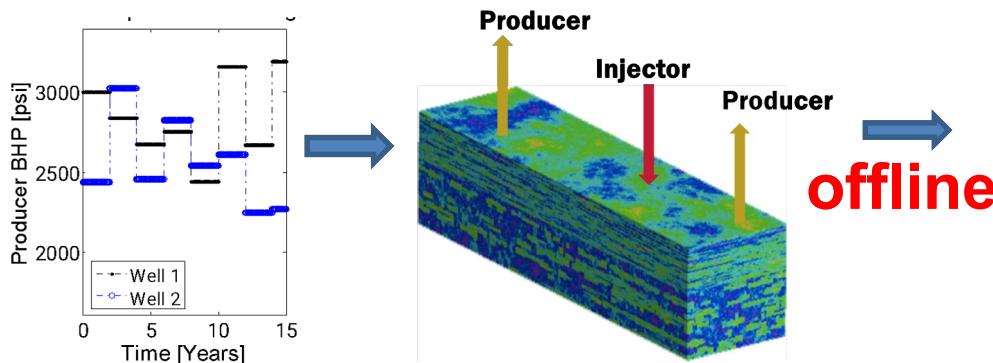
TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

# Data-Driven Model Reduction (Classical Methods)

# POD-Hybrid-based ROM



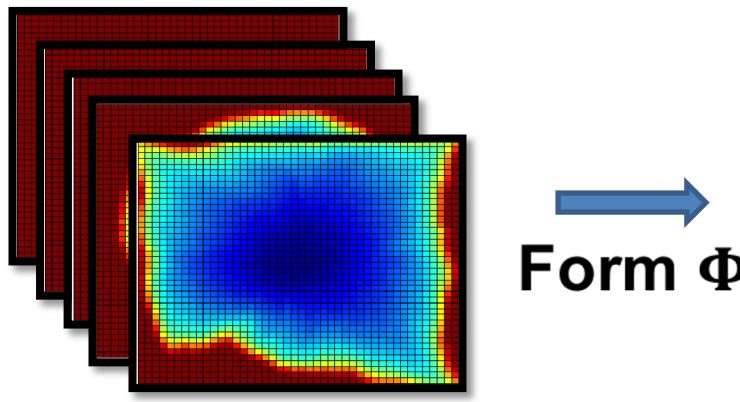
TEXAS A&M UNIVERSITY  
Harold Vance Depa  
Petroleum Enginee



$$\mathbf{R}(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^{n+1}) = \mathbf{F}(\mathbf{x}^{n+1}) + \mathbf{A}(\mathbf{x}^{n+1}, \mathbf{x}^n) + \mathbf{Q}(\mathbf{x}^{n+1}, \mathbf{u}^{n+1})$$

$$\mathcal{J} = \frac{\partial \mathcal{R}}{\partial \mathbf{x}}$$

$\mathbf{x} \rightarrow$  state (pressures, saturations)



$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}$$

$$\mathbf{r} = \dot{\mathbf{x}}_r - f(\Phi \mathbf{x}_r) - g(\Phi \mathbf{x}_r)\mathbf{u}$$

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = SVD(\mathbf{X})$$

Approximate  
 $\mathbf{x} \approx \Phi \mathbf{x}_r$   
 $\Phi \in \mathbb{R}^{N \times r}$

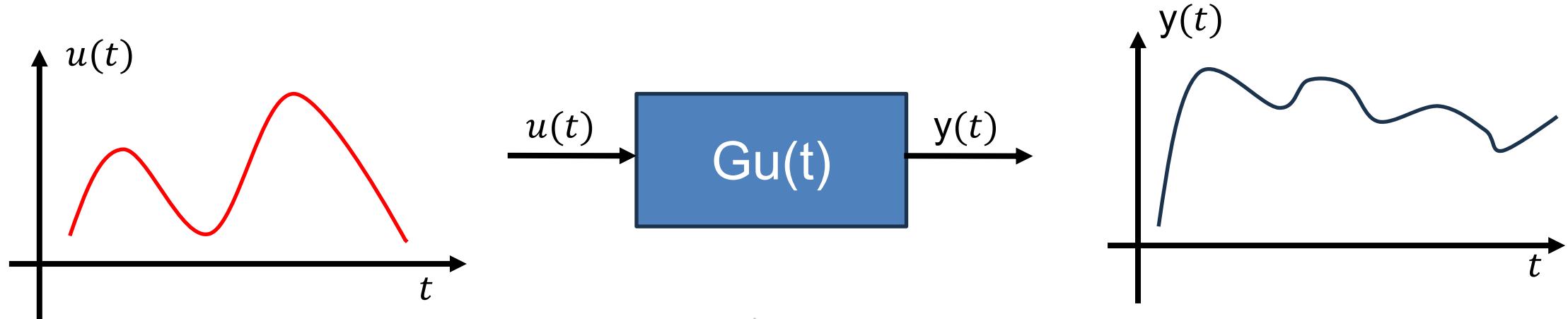
$$\Phi = \mathbf{U}[:, 1:r]$$

$$\dot{\mathbf{x}}_r = \Phi^T f(\Phi \mathbf{x}_r) + \Phi^T g(\Phi \mathbf{x}_r)\mathbf{u}$$

Project  
 $\Psi^\top \mathbf{r} = 0$   
 (Galerkin:  $\Psi = \Phi$ )

Question 1 → how to find proper projections?

# Operator Inference – Dynamical Systems

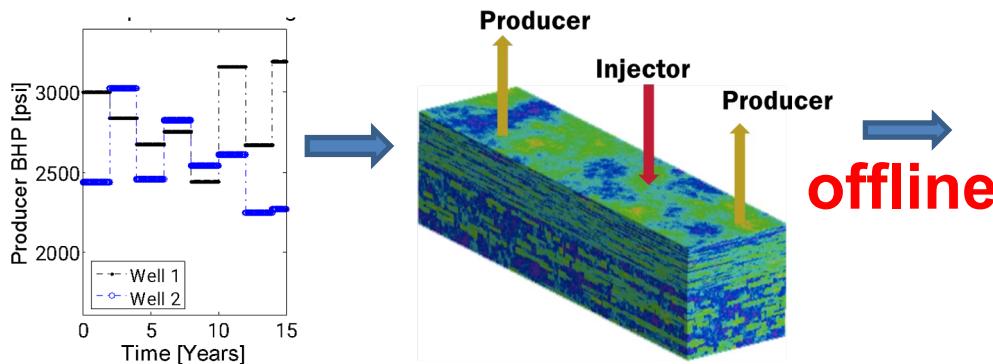


$$\mathbf{x}(t) = \Phi(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^t \Phi(t, \tau)\mathbf{B}(\tau)\mathbf{u}(\tau)d\tau$$

$$Gu(t) = x(t|u(t)) = x(0) + \int_0^t f(Gu(\tau), u(\tau))d\tau$$

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \quad \longrightarrow \quad G(s) = C(sI - A)^{-1}B + D$$

# Operator Learning



$$\mathbf{R}(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^{n+1}) = \mathbf{F}(\mathbf{x}^{n+1}) + \mathbf{A}(\mathbf{x}^{n+1}, \mathbf{x}^n) + \mathbf{Q}(\mathbf{x}^{n+1}, \mathbf{u}^{n+1})$$
$$\mathcal{J} = \frac{\partial \mathcal{R}}{\partial \mathbf{x}}$$

$\mathbf{x} \rightarrow$  state (pressures, saturations)

**Question 2 → can we directly identify the operator?**

$$\begin{cases} z^{n+1} &= \mathbf{A}^n z^n + \mathbf{B}^n u^n \\ y^{n+1} &= \mathbf{C}^n z^{n+1} + \mathbf{D}^n u^n \end{cases}$$

→ We aim to learn reduced models non-intrusively from training data *while maintaining some of the theoretical guarantees provided by intrusive methods.*

**Solution →**

(bi)DMD  
p-DMD  
Lifting

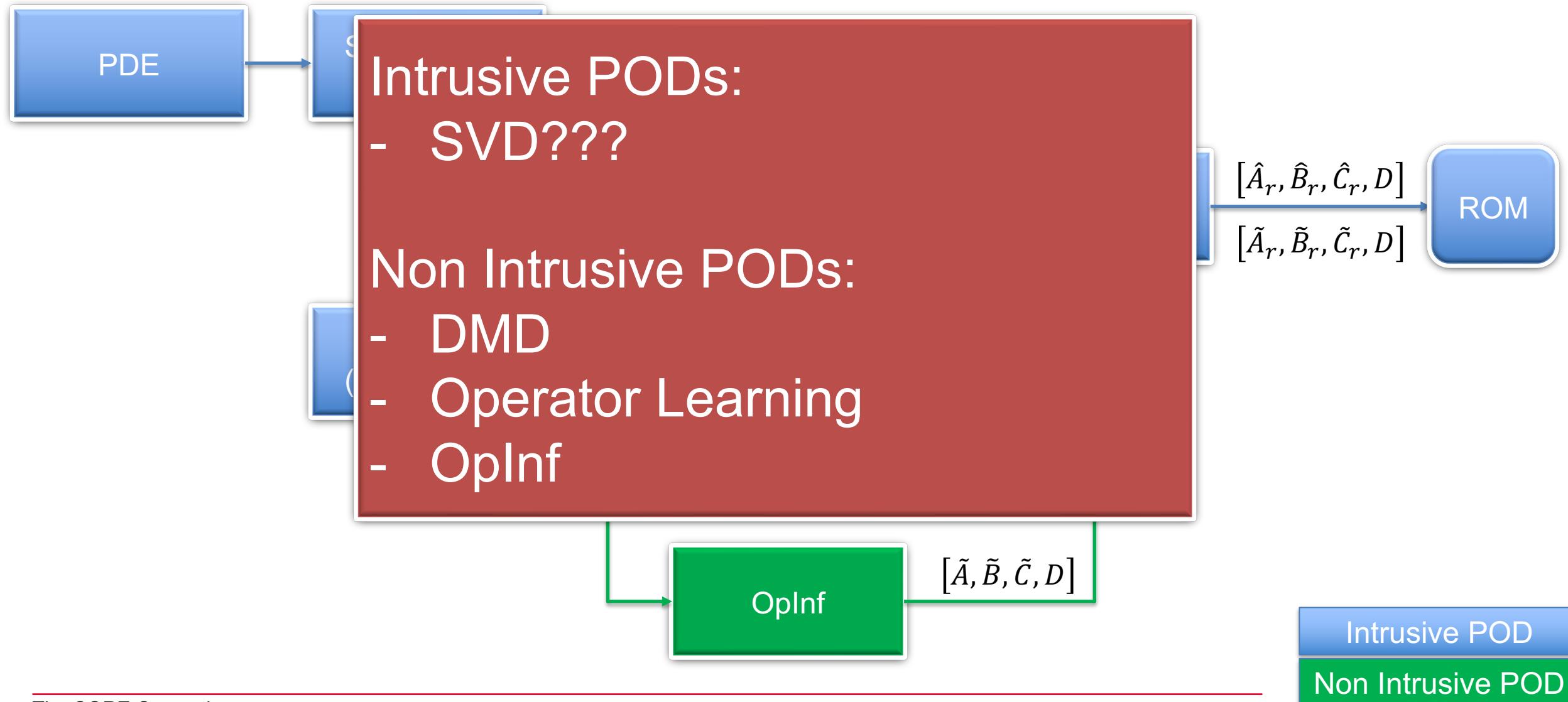
DMD for  
LPV

Koopman

E2CO

FNO  
HNO

# Motivation – Intrusive vs non-intrusive POD



# ERA – eigensystem realization



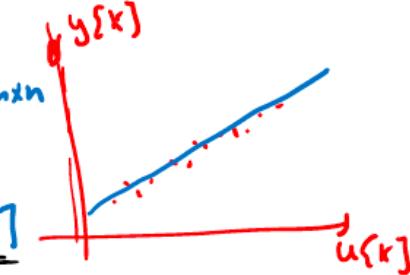
ERA: given  $u(k)$   $y(k)$  → Find  $A, B, C, D$   
thus is the discrete version

$$x[k+1] = Ax[k] + Bu[k] \rightarrow \text{Dynamic Eq}$$

$$\rightarrow y[k] = Cx[k] + Du[k] \rightarrow \text{Static Eq}$$



Assume we can find  $x[k]$



Consider for now a scalar case  $y[k] \in \mathbb{R}^1$ , But  $u[k] \in \mathbb{R}^m$  ( $m$  inputs)

$$y[k] = [c_1 c_2 \dots c_n] \begin{bmatrix} x_1[k] \\ x_2[k] \\ \vdots \\ x_n[k] \end{bmatrix} + [d_1 \dots d_m] \begin{bmatrix} u_1[k] \\ u_2[k] \\ \vdots \\ u_m[k] \end{bmatrix}$$

$$C = [\dots \dots]$$

$$\rightarrow D(u)$$

$$y[k] = \underbrace{\begin{bmatrix} x_1[k] & \dots & x_n[k] \end{bmatrix}}_z \dots \underbrace{\begin{bmatrix} u_1[k] & \dots & u_m[k] \end{bmatrix}}_w \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ d_1 \\ \vdots \\ d_m \end{bmatrix}}_{\omega}$$

$w = ?$

So far  $\rightarrow$  ill-posed problem  $\rightarrow$  Have at least  $(n+m)$  rows

$$\begin{bmatrix} y[1] \\ \vdots \\ y[N] \end{bmatrix} = \begin{bmatrix} x_1[1] & \dots & x_n[1] & u_1[1] & \dots & u_m[1] \\ x_1[2] & \dots & x_n[2] & u_1[2] & \dots & u_m[2] \\ \vdots & & & & & \\ x_1[N] & \dots & x_n[N] & u_1[N] & \dots & u_m[N] \end{bmatrix}_{N \times (n+m)} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \\ d_1 \\ \vdots \\ d_m \end{bmatrix}_{n+m}$$

$Z \in \mathbb{R}^{N \times (n+m)}$

$w = ?$

*Experiments*

Use Least Squares  $\rightarrow$

$$\min \| y - Z w \|_F^2$$

$w^*$  = optimal value

$$w^* = [Z^T Z]^{-1} Z^T y$$

$Z^T$  = pseudo inverse

How to compute Pseudo-inverse without forming  $\bar{Z}$ .  
→ One can show  $\rightarrow \bar{Z}_r \stackrel{\text{def}}{=} U_r \Sigma_r V_r^T$

$$U_r^T U_r = I$$
$$V_r^T V_r = I$$

$$\boxed{Z^+ = V \Sigma^{-1} U^T}$$

Consider the dynamical system.  
 $X[k+1] = A X[k] + B u[k]$  →  
Assume can measure all states  $X[k]$

Find  $A, B$

take only one row:  
 $\rightarrow x_i[k+1] = [a_{i1} \dots a_{in}] \begin{pmatrix} x_1[k] \\ x_2[k] \\ \vdots \\ x_n[k] \end{pmatrix} + [b_{i1} \dots b_{im}] \begin{pmatrix} u_1[k] \\ u_2[k] \\ \vdots \\ u_m[k] \end{pmatrix}$

Concatenate:

$$x_i[k+1] = \begin{bmatrix} x^T[k] & u^T[k] \end{bmatrix} \begin{bmatrix} a_{i,1} \\ \vdots \\ a_{i,n} \\ b_{i,1} \\ \vdots \\ b_{i,m} \end{bmatrix}$$

$$\rightarrow y_i = z_i w_j$$

$\Rightarrow$  many i's to work with  $\rightarrow$  independent.

↓  
same L.S.

↓  
Find  $A_1, B_1, C_1, D$

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

$$\begin{aligned} \hat{A} &= T^{-1}AT \\ \hat{B} &= T^{-1}B \\ \hat{C} &= CT \\ \hat{D} &= D \end{aligned}$$

$$G(s) = C(sI - \hat{A})^{-1}\hat{B}$$

Find  $T, \hat{A}, \hat{B}$

$$\begin{cases} \dot{x} = \hat{A}\hat{T}x + \hat{B}u \\ y = \hat{C}\hat{T}x + \hat{D}u \end{cases}$$

Recall  $H_0$

$$H_0 = \begin{bmatrix} C_B & CA & CA^2B & \dots & CA^{k-1}B \\ 1 & 1 & 1 & \vdots & 1 \\ CA^{k-1}B & 1 & 1 & \vdots & 1 \end{bmatrix}$$

$H_1 = \begin{bmatrix} \text{shifted } H_0 \end{bmatrix}$

↓

$$H_0 = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{k-1} \end{bmatrix} \begin{bmatrix} B & AB & A^2B & \dots & A^{k-1}B \end{bmatrix}$$
$$= O_K(A_{1L}) \times C_K(A_{nB})$$

$$H_1 = \begin{bmatrix} CAB & CA^2B & \dots & CA^k B \\ \vdots & & & \\ CA^k B & \dots & CA^{2r-1}B \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{r-1} \end{bmatrix} A \begin{bmatrix} B \\ AB \\ \dots \\ A^{k-1}B \end{bmatrix}$$

Consequence: rank  $H_0 \equiv$  order the minimal realization of the system

$$\boxed{\text{rank } H_0 = r}$$

Fact:

$$\Theta_k(A, c) \times B = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{k-1} \end{bmatrix}^B = \begin{bmatrix} CB \\ CAB \\ CA^2B \\ \vdots \\ CA^{k-1}B \end{bmatrix} \Rightarrow H_0 e_1 = \begin{bmatrix} h_{f1} \\ h_{c1} \\ h_{s1} \\ \vdots \\ 1 \end{bmatrix}$$

Similarly

$$C C(A, B) = \begin{bmatrix} C_0 & CAB - CA^{k+1}B \\ h[1] & h[2] - h[k] \end{bmatrix} = \Rightarrow e_1^T H_0$$

take  $A_b \rightarrow SVD$

$$A_b = U \Sigma V^T = U_s \Sigma_s V_s^T$$

Since  $A_b = \Sigma^{1/2} U_s \Theta_k(A) C_k(A, B) V_s \Sigma_s^{-1/2}$

$$\left[ \Sigma_s^{-1/2} U_s^T \Theta_k(A, C) \right] \left[ C_k(A, B) V_s \Sigma_s^{-1/2} \right] = I_r$$

$$\boxed{T := C_k(A, B) V_s \Sigma_s^{-1/2}}$$
$$\bar{T} := \Sigma_s^{-1/2} U_s^T \Theta_k(A, C)$$

take

the transformation

$$\hat{A} = T^{-1}AT^T = \Sigma_i^{-1/2} U_i^T \Theta_{rk}(A, C) A C_k(A, B) V_i \Sigma_i^{-1/2}$$

$H_1$

$$\hat{A} = \Sigma_i^{-1/2} U_i^T H_i V_i \Sigma_i^{-1/2} \rightarrow \text{only depends on data!}$$

Similarly

$$\hat{B} = T^{-1}B = \Sigma_i^{1/2} V_i^T e_i$$

$$\hat{C} = CT = e_i^T U_i \Sigma_i^{1/2}$$

ERA ProofsDefinition Impulse Response

If  $u[k]$  is the impulse input, the output obtained is called Impulse response  $h(k)$ , as is given by

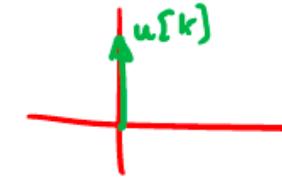
$$h(0) = 0$$

$$h(k) = C A^{k-1} B \quad \text{for } k > 0$$

$$x[k+1] = A x[k] + B u[k]$$

$$y[k] = C x[k] + D u[k]$$

$$\begin{aligned} u(0) &= 1 \\ u(n) &= 0 \quad n > 0 \end{aligned}$$



$$\begin{aligned} x[0] &= 0 & \downarrow^0 & \uparrow^1 \\ x[1] &= A x[0] + B u[0] & & \\ [y[1] = C B] &\rightarrow h(1) & & \\ x[2] &= A x[1] + B u[1] & & \\ y[2] &= C x[1] + D u[1] = h(2) & & \\ & \quad \text{CAB} & & \\ & \vdots & & \end{aligned}$$

ERA Algorithm:

- Given  $\rightarrow$  Impulse Responses  $y(k) = h(i)$   $i = 1, \dots, 2k$
- Find  $\hat{A}, \hat{B}, \hat{C}, \hat{D}$

1) Construct the Hankel Matrices

$$H_0 = \begin{bmatrix} h(1) & h(2) & \dots & h(k) \\ h(2) & h(3) & \dots & h(k+1) \\ \vdots & & & \\ h(k) & \dots & & h(2k-1) \end{bmatrix}$$

$$u[k] = \begin{bmatrix} u[1] \\ u[2] \\ \vdots \\ u[k] \end{bmatrix}$$

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k] \\ x[0] &= 0 \end{aligned}$$

$$x[1] = Bu[0]$$

$$H_1 = \begin{bmatrix} h(2) & h(3) & \dots & h(k+1) \\ \vdots & & & \\ h(k+1) & \dots & & h(2k) \end{bmatrix}$$

2) Compute SVD of  $H_0 = [U_1 | U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U_1 \Sigma_1 V_1^T$

3) Construct similarity transform matrices

*Reduced order matrices*

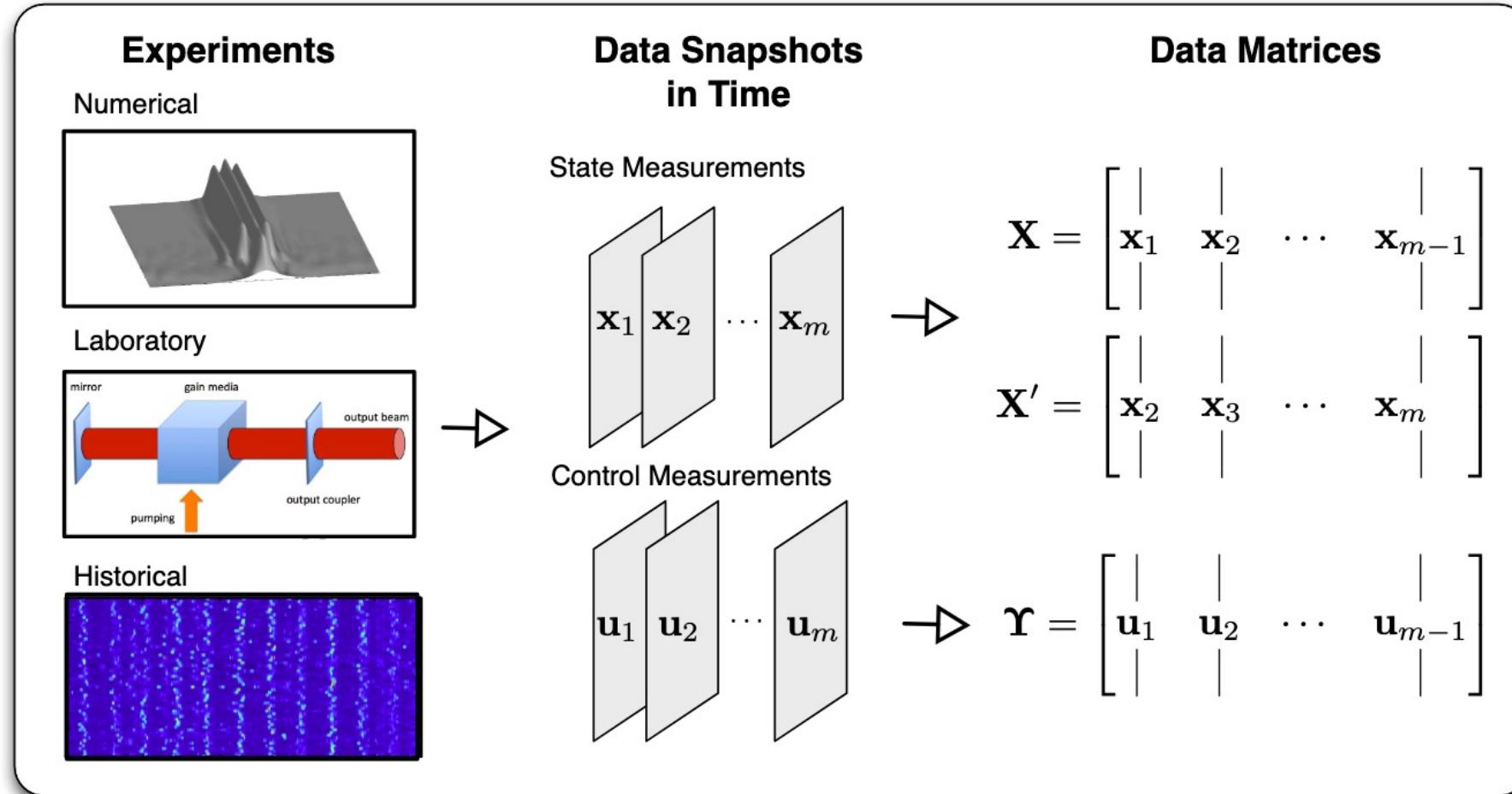
$$\begin{cases} \hat{A} = \Sigma_1^{-1/2} U_1^T H_1 V_1 \Sigma_1^{-1/2} \\ \hat{B} = \Sigma_1^{1/2} V_1^T e_1 \\ \hat{C} = e_1^T U_1 \Sigma_1^{1/2} \end{cases}$$

take the first column of  $\Sigma_1^{1/2} V_1^T$

take the first row of  $e_1^T U_1 \Sigma_1^{1/2}$

$\hat{D} = D$

# Data Collection

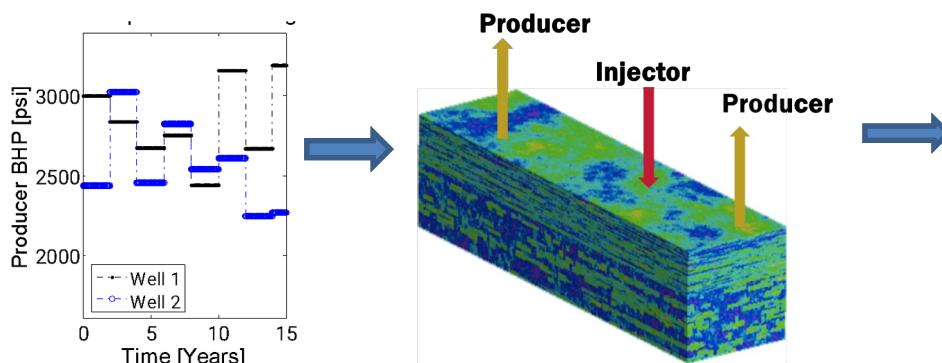


# Dynamic Mode Decomposition (DMD)



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

$$\dot{x} = f(x(t), u(t), t) \rightarrow \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \rightarrow \begin{cases} \dot{x}(t) = Ax(t) \\ x(0) = x_0 \end{cases}$$



$$X = \begin{bmatrix} x_1 & x_1 & \cdots & x_1 \\ x_2 & x_2 & \cdots & x_2 \\ x_3 & x_3 & \cdots & x_3 \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_n & \cdots & x_n \end{bmatrix} \quad t_1 \quad t_2 \quad t_i \quad t_N$$

$$\{x_{k+1} = Ax_k$$

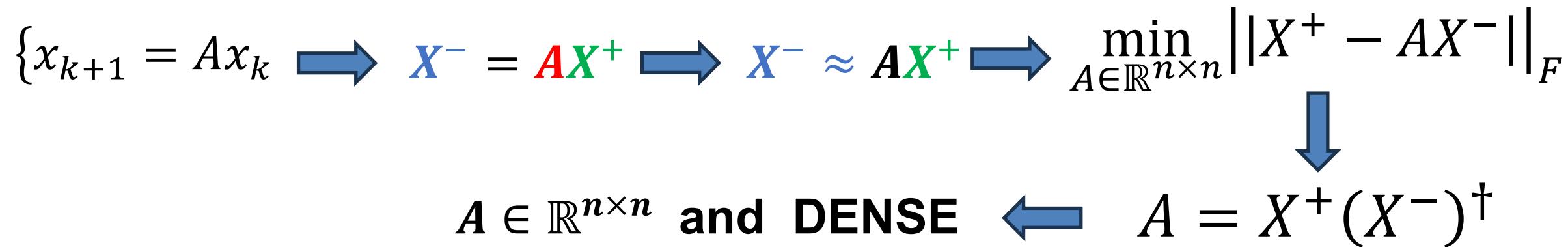
$$X^- = AX^+$$

$$X^- = [x^1 \quad x^2 \quad \cdots \quad x^{N-1}]$$

$$X^+ = [x^2 \quad x^3 \quad \cdots \quad x^N]$$

$$X^- = [x^1 \quad x^2 \quad \dots \quad x^{N-1}]$$

$$X^+ = [x^2 \quad x^3 \quad \dots \quad x^N]$$



DMD look for a reduced representation of A

$$X^- \approx U\Sigma V^T$$



$$A_r = U^T X^+ V \Sigma^{-1}$$

# Model Reduction

## Dynamic Mode Decomposition (DMD)

Find the dynamic properties of  $\mathbf{A}$

$$\mathbf{X}' = \mathbf{AX}$$

- Find the truncated SVD of  $\mathbf{X}$

$$\mathbf{X} \approx \tilde{\mathbf{U}} \tilde{\Sigma} \tilde{\mathbf{V}}^*$$

- Compute reduced-order approximation  $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} = \tilde{\mathbf{U}} \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1}$$

- Investigate the dynamic properties of  $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} \mathbf{W} = \mathbf{W} \Lambda$$

- Solve for the dynamic modes of  $\mathbf{A}$

$$\Phi = \mathbf{X}' \mathbf{V} \Sigma^{-1} \mathbf{W}$$

## DMD with Control (DMDc)

Find the dynamic properties of  $\mathbf{A}$  and  $\mathbf{B}$

$$\mathbf{X}' = \mathbf{AX} + \mathbf{BY}$$

- Construct the input data matrix

$$\Omega = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix}$$

- Find the truncated SVD of input matrix  $\Omega$

$$\Omega \approx \tilde{\mathbf{U}} \tilde{\Sigma} \tilde{\mathbf{V}}^* = \begin{bmatrix} \tilde{\mathbf{U}}_1 \\ \tilde{\mathbf{U}}_2 \end{bmatrix} \tilde{\Sigma} \tilde{\mathbf{V}}^*$$

- Find the truncated SVD of output matrix  $\mathbf{X}'$

$$\mathbf{X}' \approx \hat{\mathbf{U}} \hat{\Sigma} \hat{\mathbf{V}}^*$$

- Compute reduced-order approximation of  $\mathbf{A}$

$$\tilde{\mathbf{A}} = \hat{\mathbf{U}}^* \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1} \tilde{\mathbf{U}}_1^* \hat{\mathbf{U}}$$

- Investigate the dynamic properties of

$$\tilde{\mathbf{A}} \mathbf{W} = \mathbf{W} \Lambda$$

- Solve for the dynamic modes of

$$\Phi = \mathbf{X}' \mathbf{V} \Sigma^{-1} \tilde{\mathbf{U}}_1^* \hat{\mathbf{U}} \mathbf{W}$$



Algorithm : DMD

- 1) Given  $\dot{x} = f(t, x, u) \rightarrow$  non linear function (Full order Simulator)  
 $\dot{x} = f(t, x, u)$   $\rightarrow$  state snapshots ( $N$  snapshots)  $\rightarrow$  collect data
  - 2) Perform simulations to get state snapshots ( $N$  snapshots)
  - 3) Form these matrices  
 $z = [x_1 \ x_2 \dots \ x_{N-1}]$   
 $z' = [x_2 \dots \ x_N]$
  - 4) Compute low rank approx of  $z$   
 $\downarrow$  r rank
  - 5) Perform POD on  $A$   
 $\tilde{A} = U_r^* A U_r$
- $\downarrow$
- $$z_r = U_r \Sigma_r V_r^*$$
- $\downarrow$
- $$z' = A U_r \Sigma_r V_r^*$$
- $\longrightarrow$
- $\tilde{A} = U^* z' V \Sigma^{-1}$

6) Compute  $\text{eig}(\tilde{A})$

$$\tilde{A} W = W \Lambda$$

↑  
eigenvalues

↑  
eigenvectors

eigenvalues of (A)

7) Form Projector (BMO)

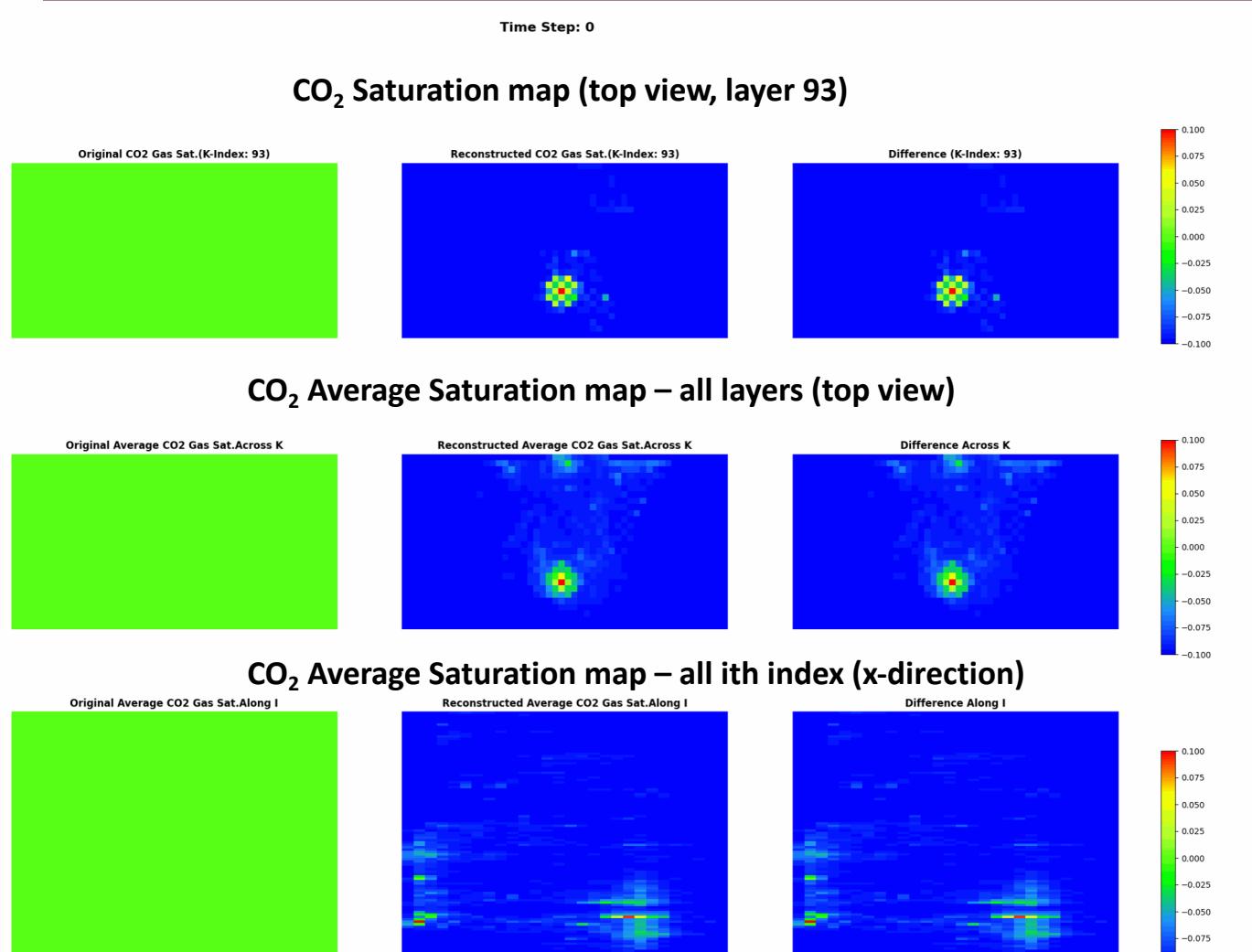
$$\boxed{\hat{\Phi} = Z V \Sigma^{-1} W} \rightarrow \text{eigenvectors of } A$$

8) Evolve in time:

$$\boxed{\tilde{x}(k\Delta t) = \hat{\Phi} \Delta^t b_0}$$

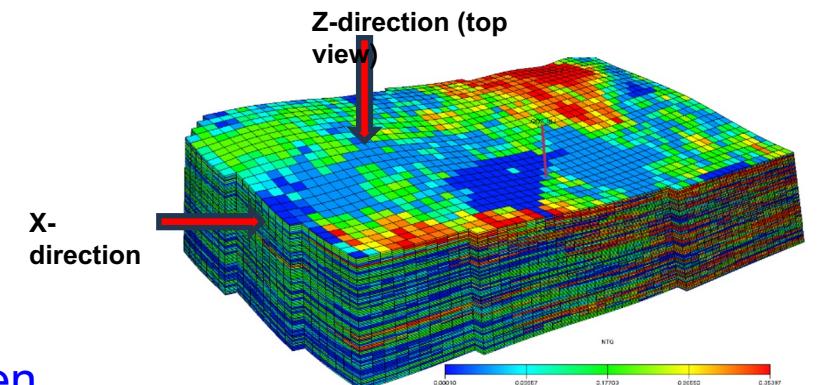
I.C

# Sparse DMD Method for CCS



## Results : CO<sub>2</sub> plume evolution

	Numerical model	Sp-DMD
<b>Run time (seconds)</b>	<b>6000 (approx. 2hrs)</b>	<b>123 secs</b>



Rapid Geological CO<sub>2</sub> Storage Forecast and Optimization: A Data-driven Dynamic Mode Decomposition Model Order Reduction Approach

# Why not only DMD?



DMD

- *best-fits linear operators to state trajectories*
- *Methods based on Koopman operators have been developed to extend DMD to nonlinear*
- *challenge is selecting observables such that the dynamics become close to linear.*

OplInf

- *Operator Inference fits operators of reduced models to data;*
- *however, it allows for nonlinear terms and thus can capture nonlinear dynamics.*
- *Operator Inference explicitly embeds the underlying physics through the structured form of the reduced model it learns.*

[Data-driven operator inference for nonintrusive projection-based model reduction](#)

B. Peherstorfer and K. Willcox, Computer Methods in Applied Mechanics and Engineering, 2016

# OplInf – Non-linear systems



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

## Step 1 : Data collection + Lift to higher dimensions

$$X = \begin{bmatrix} t_1 & t_2 & t_i & t_N \\ x_1 & x_1 & \cdots & x_1 \\ x_2 & x_2 & \cdots & x_2 \\ x_3 & x_3 & \cdots & x_3 \\ \vdots & \vdots & \cdots & \vdots \\ x_n & x_n & \cdots & x_n \end{bmatrix}$$

$$\dot{x}(t) = ax(t) + e^{-x(t)}.$$

Set  $x_1 = x$  and take  $x_2 = e^{-x_1}$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} a & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -a & -1 \end{bmatrix} \left( \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \otimes \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \right)$$

$$f(x(t; \mu), u(t); \mu) = A_1(\mu)x(t; \mu) + A_2x^2(t; \mu) + B(\mu)u(t).$$

[Learning nonlinear reduced models from data with operator inference](#)

[B. Kramer](#), [B. Peherstorfer](#), and [K. Willcox](#) Annual Review of Fluid Mechanics, 2024

## Step 2 : Construct low-dimensional (latent space)

$$[U, S, V] = SVD(X) \xrightarrow[\text{POD}]{\text{Projection}} x \approx V\hat{x}$$

## Step 3 : reduced model construction by OplInf

$$\frac{d}{dt}x(t; \mu) = f(x(t; \mu), u(t); \mu) = \sum_{i=1}^{\ell} A_i(\mu)x^i(t; \mu) + B(\mu)u(t),$$

$$\frac{d}{dt}\hat{x}(t; \mu) = \sum_{i=1}^{\ell} \hat{A}_i(\mu)\hat{x}^i(t; \mu) + \hat{B}(\mu)u(t),$$



# Least square solutions

$$J_j(\hat{A}_1(\mu_j), \dots, \hat{A}_\ell(\mu_j), \hat{B}(\mu_j)) = \sum_{k=1}^K \left\| \sum_{i=1}^\ell \hat{A}_i(\mu_j) \check{x}^i(t_k; \mu_j) + \hat{B}(\mu_j) \mathbf{u}(t_k; \mu_j) - \check{x}'(t_k; \mu_j) \right\|_2^2,$$

## For Linear Systems (first order)

$$\dot{\hat{\mathbf{X}}}_r = \hat{\mathbf{A}}_r \hat{\mathbf{X}}_r + \hat{\mathbf{B}}_r \mathbf{U} \quad \xrightarrow{\text{green arrow}} \quad \dot{\hat{\mathbf{X}}}_r = \mathbf{O}\boldsymbol{\Gamma}$$
$$\Gamma_{r+P \times K} = \begin{bmatrix} \hat{\mathbf{X}}_r^T & \mathbf{U}^T \end{bmatrix}^T$$
$$\mathbf{O}_{r \times r+P} = \begin{bmatrix} \hat{\mathbf{A}}_r & \hat{\mathbf{B}}_r \end{bmatrix}$$
$$\hat{\mathbf{O}} = \arg \min_{\mathbf{O}} \| \dot{\hat{\mathbf{X}}}_r - \mathbf{O}\boldsymbol{\Gamma} \|^2$$

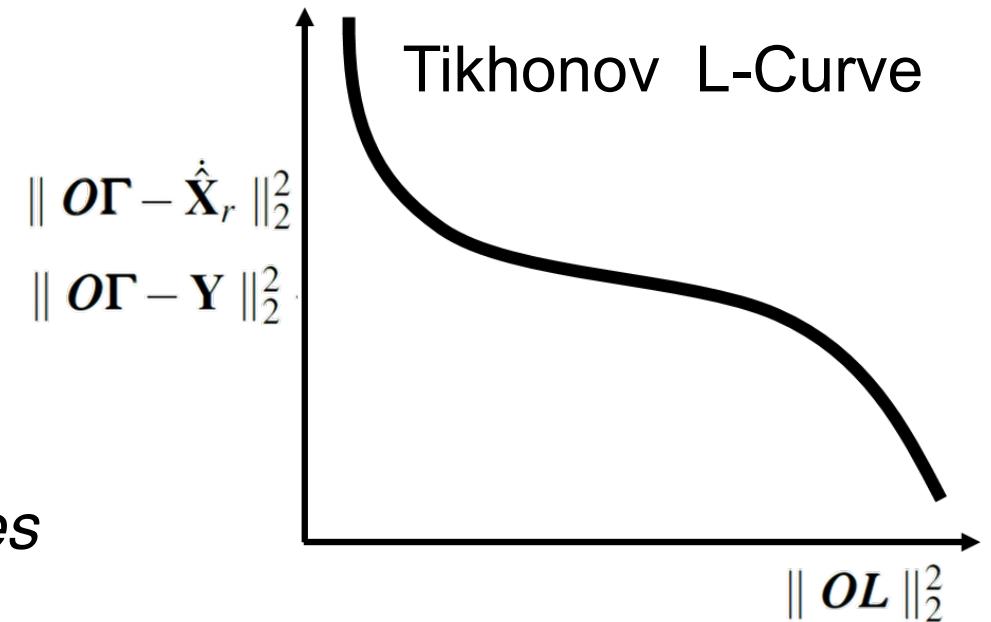
# Operator Inference - OplInf



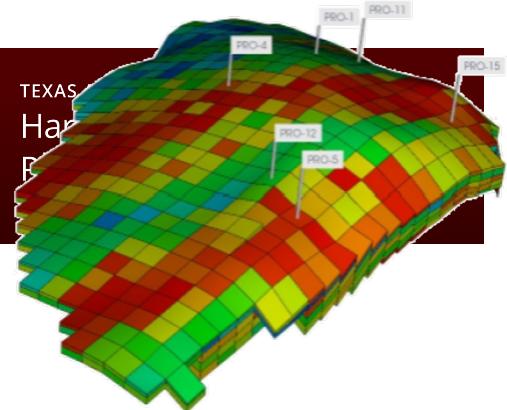
*Tikhonov regularization has been proposed to enhance Operator Inference by preventing overfitting*

$$\hat{\boldsymbol{\theta}} = \min_{\boldsymbol{\theta}} \left[ \| \boldsymbol{O}\boldsymbol{\Gamma} - \dot{\mathbf{X}}_r \|_2^2 + \lambda \| \boldsymbol{OL} \|_2^2 \right]$$

*Regularization imposes a bias on the learning process to guide Operator Inference toward meaningful models that have predictive capabilities and that generalize well to unseen parameters, inputs, and initial conditions.*



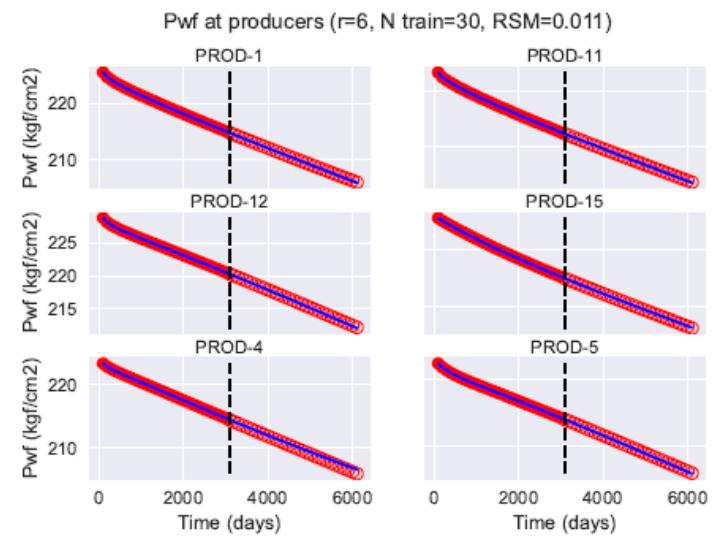
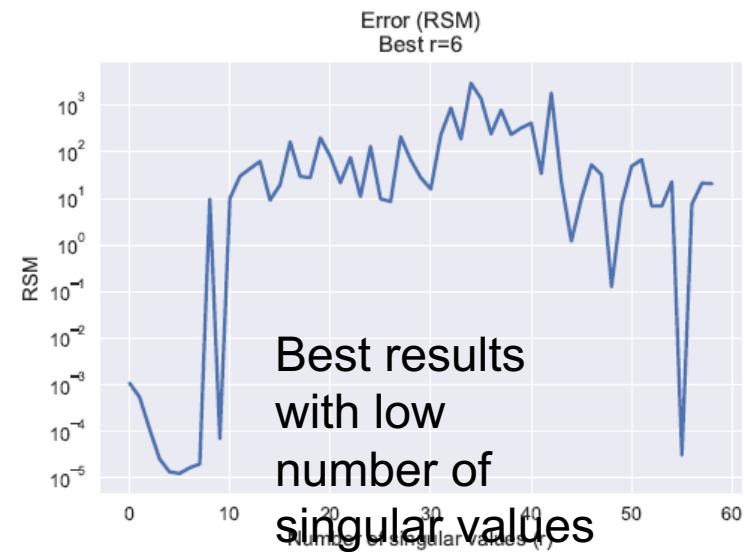
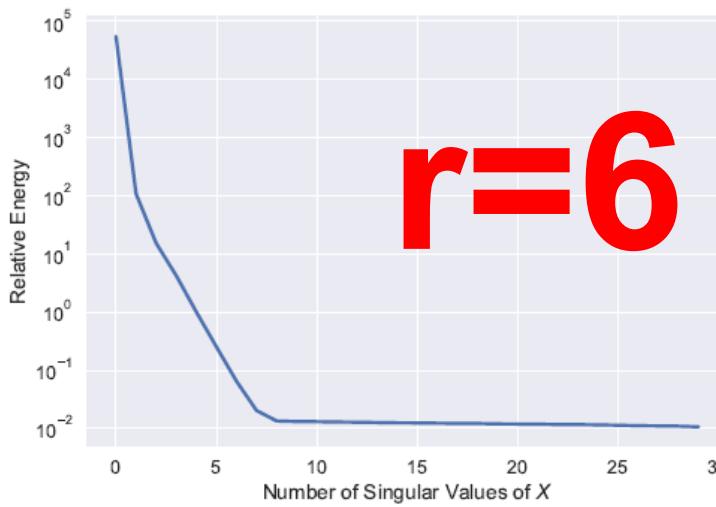
# Oplnf – Example 1



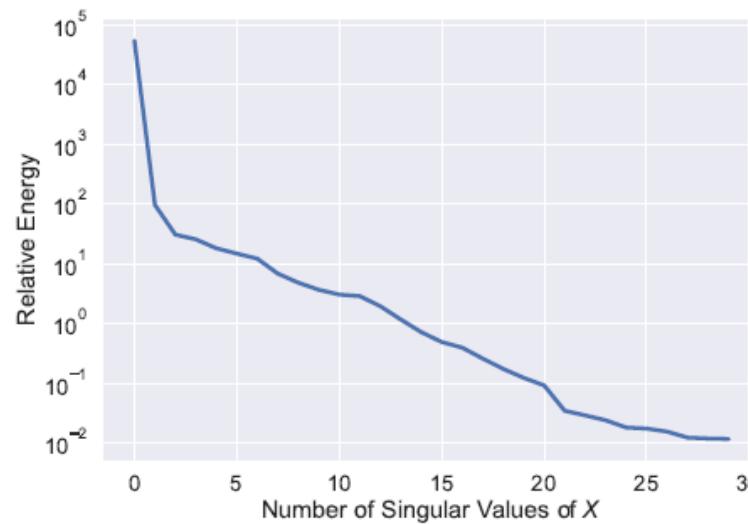
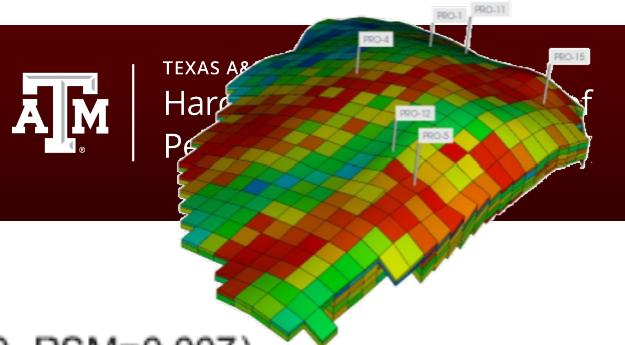
## PUNQ MODEL (Single-Phase) $19 \times 28 \times 5 = 2660$ cells

$$\mathbf{X}_{N \times K} = \begin{bmatrix} p_1^{(1)} & p_1^{(2)} & p_1^{(3)} & \dots & p_1^{(K)} \\ p_2^{(1)} & p_2^{(2)} & p_2^{(3)} & \dots & p_2^{(K)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_N^{(1)} & p_N^{(2)} & p_N^{(3)} & \dots & p_N^{(K)} \end{bmatrix}$$

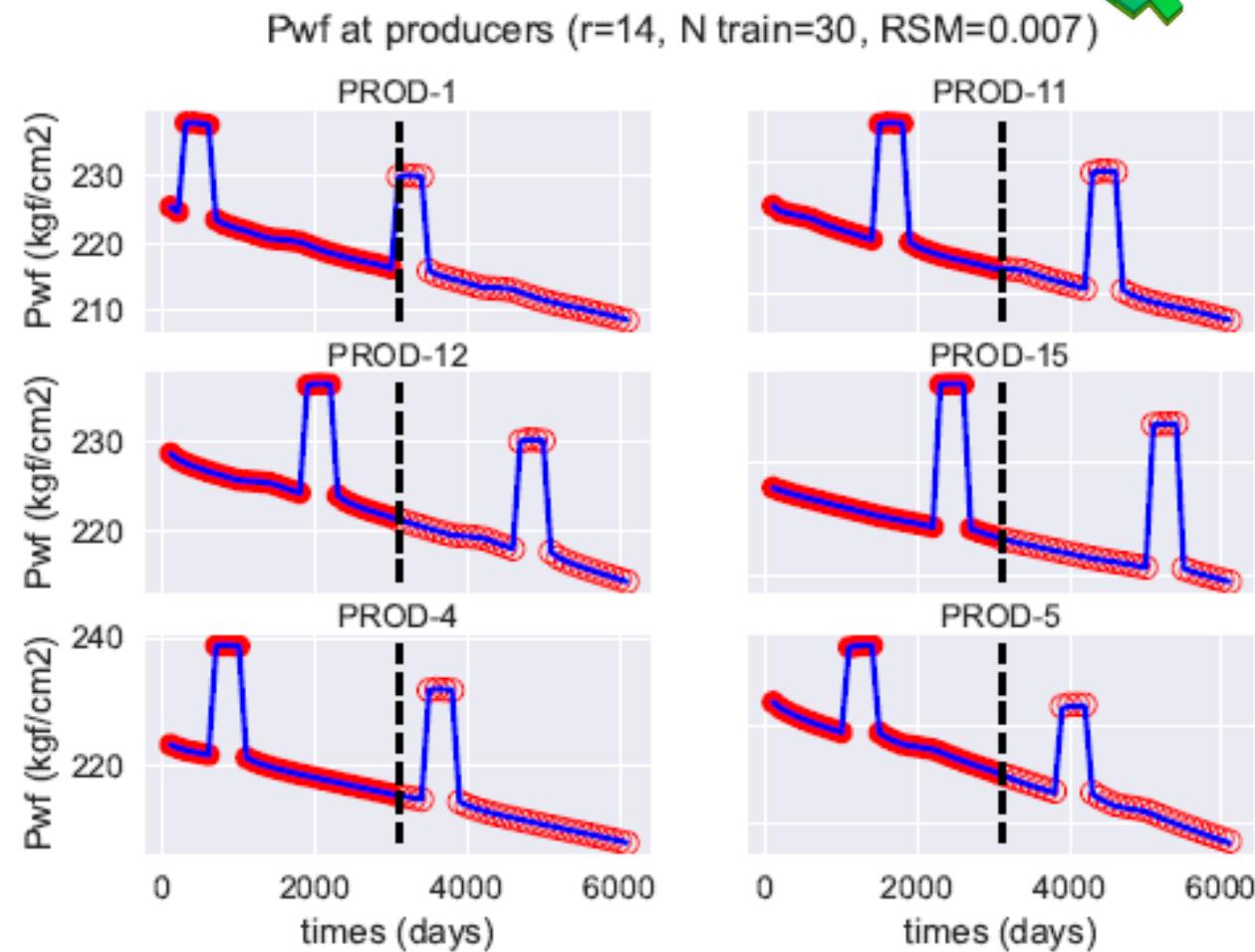
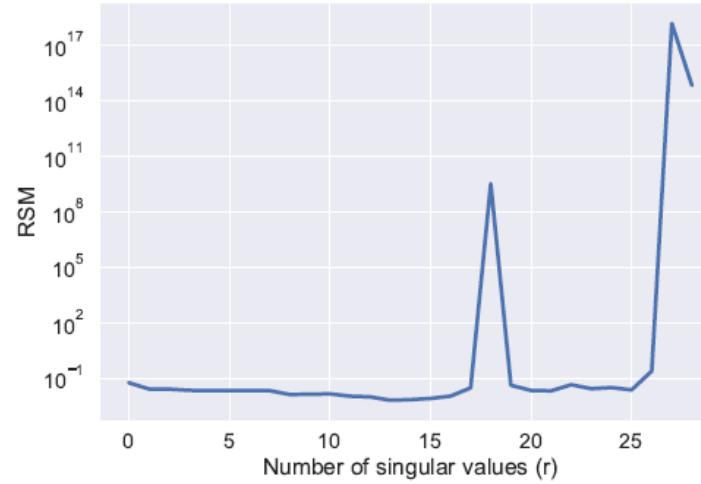
$$\mathbf{Y}(t) = \begin{bmatrix} p_{wf}^{(1)} & p_{wf}^{(2)} & p_{wf}^{(3)} & \dots & p_{wf}^{(K)} \\ BSW^{(1)} & BSW^{(2)} & BSW^{(3)} & \dots & BSW^{(K)} \end{bmatrix}$$



# OplInf – Example 1



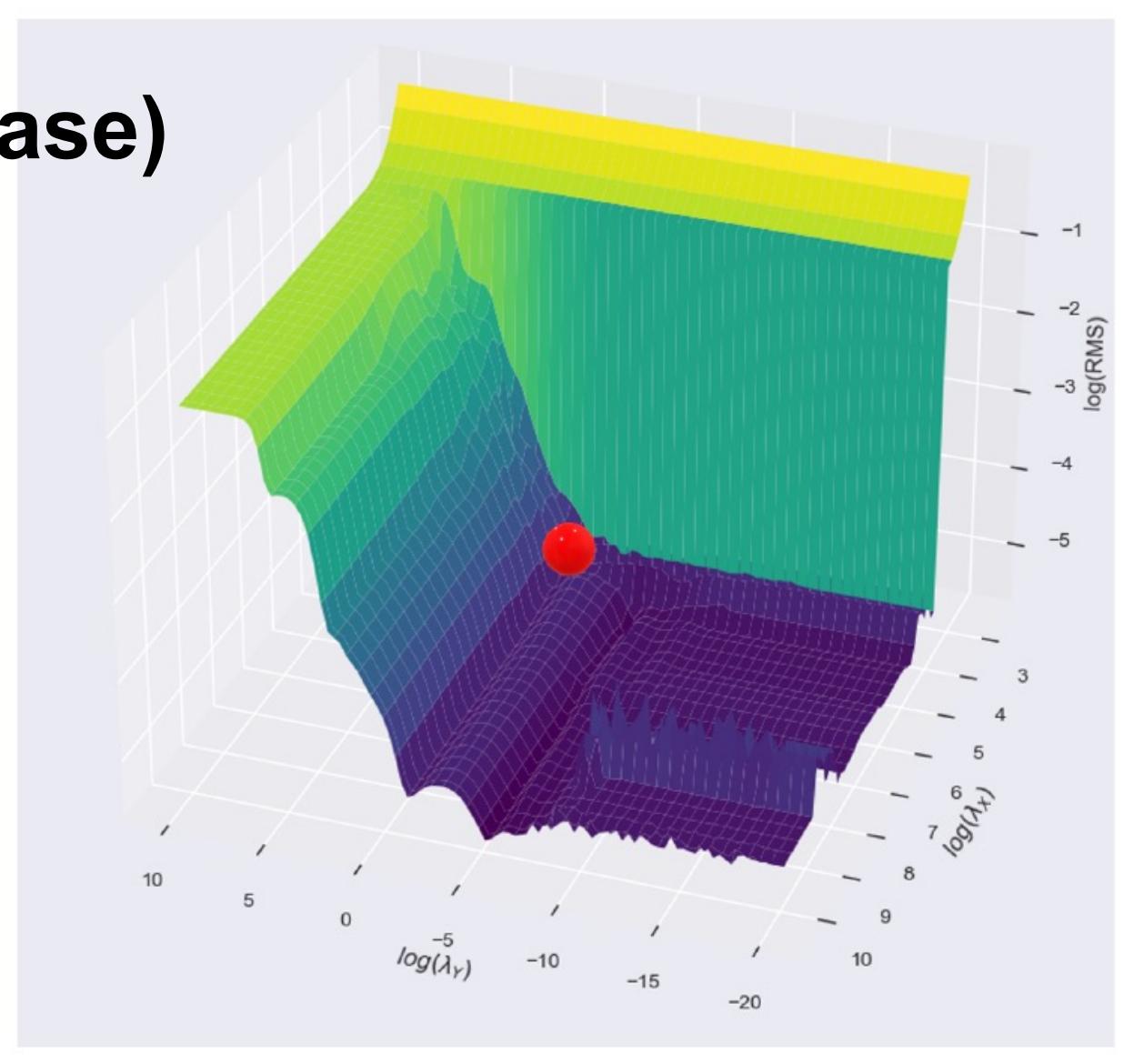
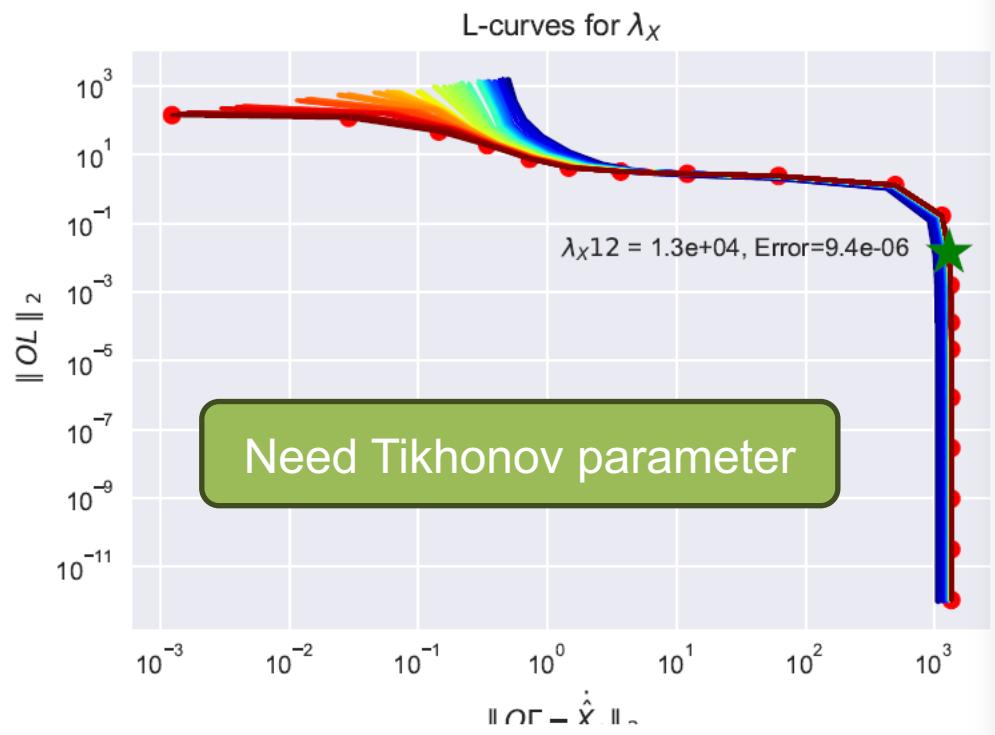
Error (RSM)  
Best r=14

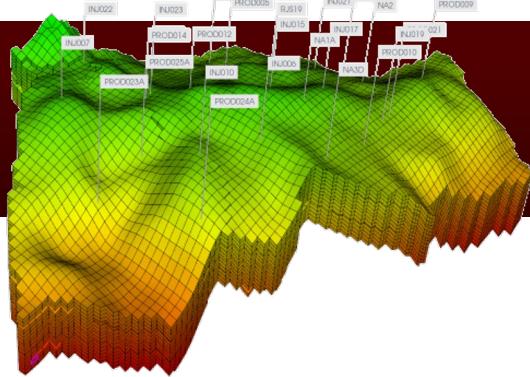


# Example 2

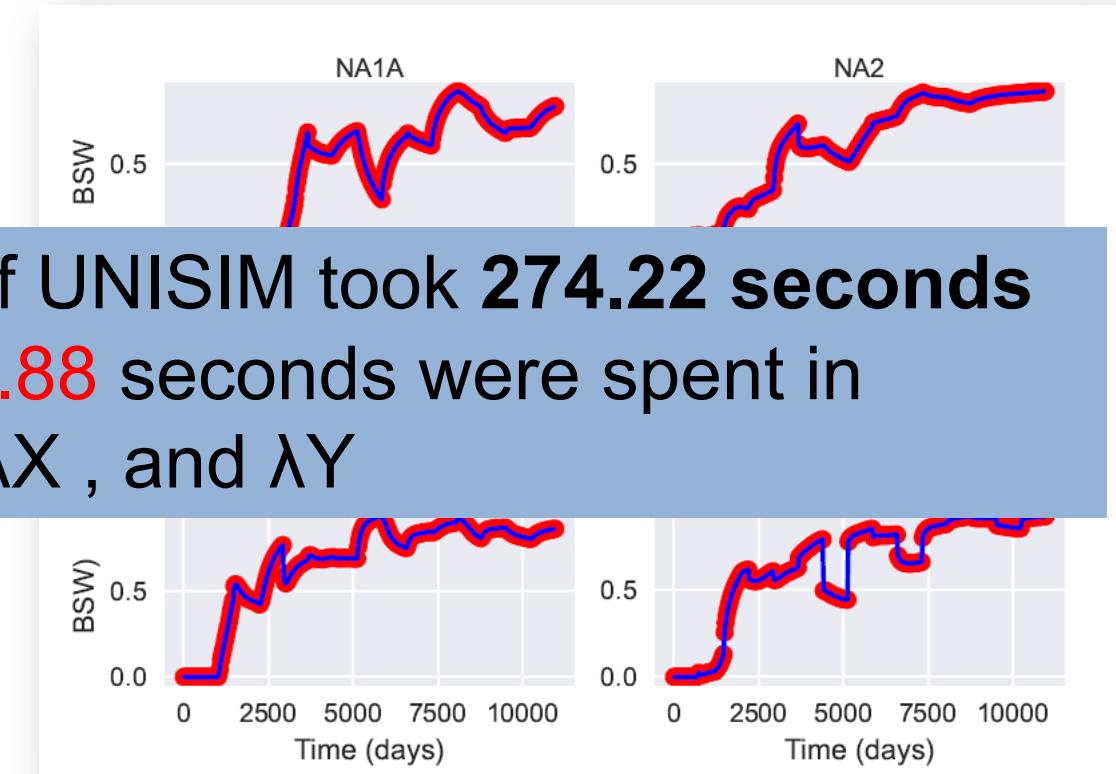
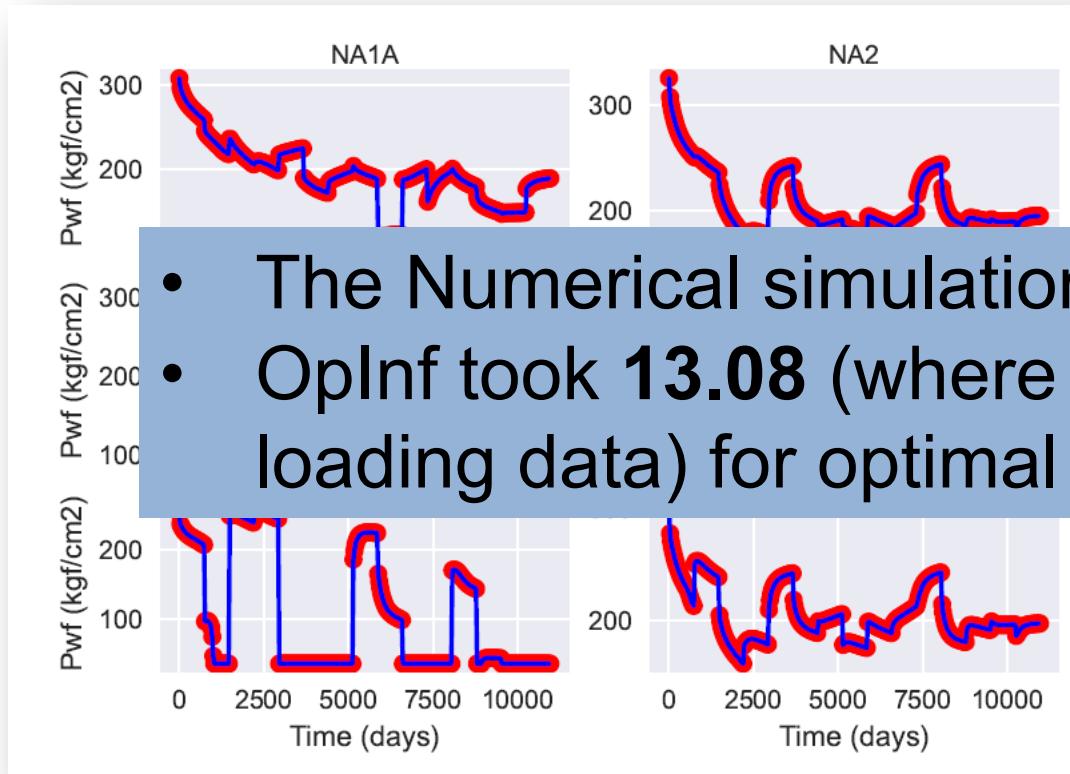
## UNISIM – I – M (Two-Phase)

$81 \times 58 \times 20 = 93,960$  cells





# r=25



- The Numerical simulation of UNISIM took **274.22 seconds**
- Oplnf took **13.08** (where **10.88** seconds were spent in loading data) for optimal  $r$ ,  $\lambda X$  , and  $\lambda Y$



TEXAS A&M UNIVERSITY  
Engineering



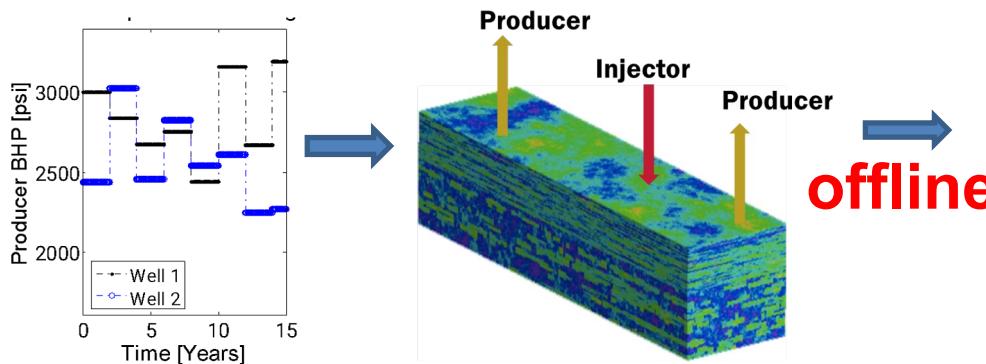
TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

# Data-Driven MOR (Based on Deep Learning)

# Problem Setting - Operator Learning

TEXAS A&M UNIVERSITY

Harold Vance Department of  
Petroleum Engineering



$$\mathbf{R}(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^{n+1}) = \mathbf{F}(\mathbf{x}^{n+1}) + \mathbf{A}(\mathbf{x}^{n+1}, \mathbf{x}^n) + \mathbf{Q}(\mathbf{x}^{n+1}, \mathbf{u}^{n+1})$$
$$\mathcal{J} = \frac{\partial \mathcal{R}}{\partial \mathbf{x}}$$

$\mathbf{x} \rightarrow$  state (pressures, saturations)

**Question → can we directly identify the Operator ( or dynamical system)?**

$$F: \mathcal{A} \times \Theta \rightarrow u$$

$$\begin{cases} z^{n+1} &= \mathbf{A}^n z^n + \mathbf{B}^n u^n \\ y^{n+1} &= \mathbf{C}^n z^{n+1} + \mathbf{D}^n u^n \end{cases}$$

→ We aim to learn reduced models non-intrusively from training data *while maintaining some of the theoretical guarantees provided by intrusive methods.*

**Solution →**

(bi)DMD  
p-DMD  
Lifting

DMD for  
LPV

Koopman

E2CO

FNO  
HNO

# What is (Fourier) Neural Operators?

TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

## Problem setting

$$-\nabla \cdot (K(x)\nabla u(x)) = f(x), \text{ with proper BC on } \partial D$$

## Solution $\rightarrow$ Green's function

G  $\rightarrow$  Green's function or Kernel Function

$$u(x) = \int_D G_K(x, y)f(y)dy + \text{non zero BC}$$

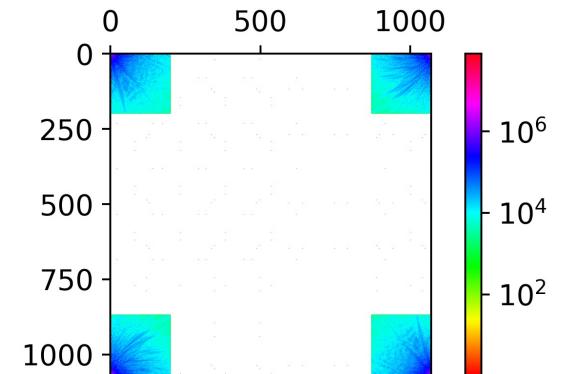
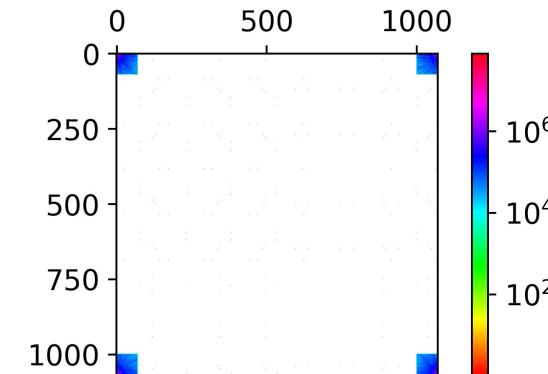
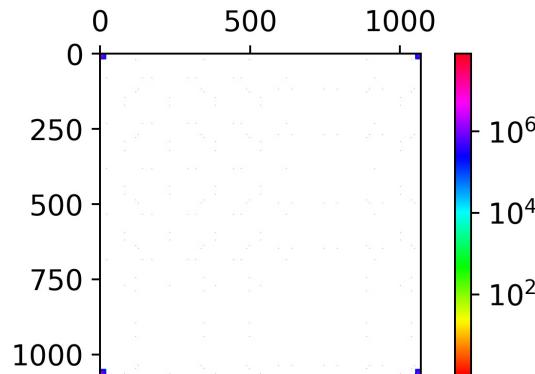
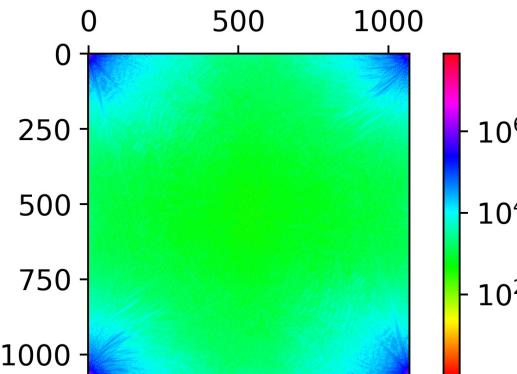
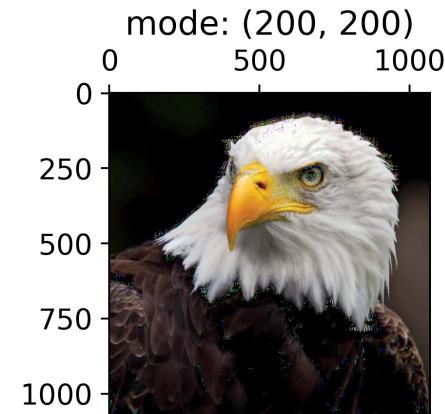
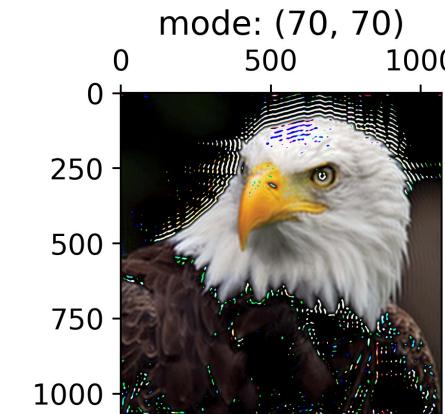
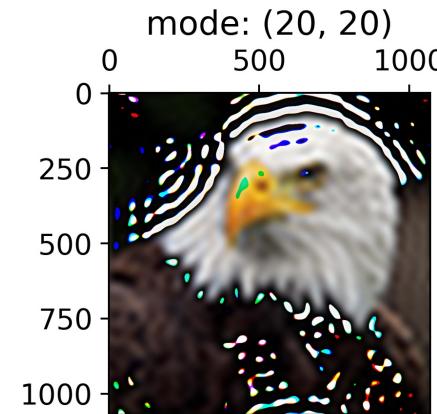
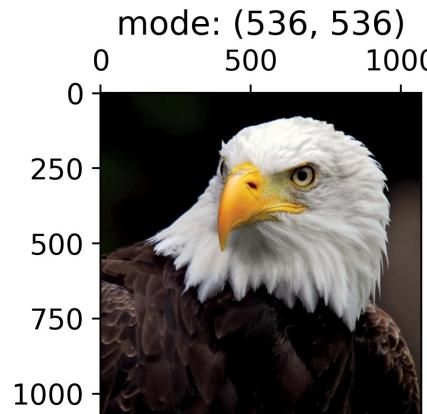
## Neural Operators $\rightarrow$ Approximate the Kernel (or convolution) by a NN

### Architecture

$$v_{l+1}(s) = \sigma \left( W_l v_l(s) + \int_D \kappa_l(s, z)v_l(z) dz + b_l(s) \right)$$

# Image recovery considering Fourier high modes

- High modes are positioned at the corners of the image in Fourier space
- Inverting back high models to the real space recovers the image to a great accuracy.
- Gibbs phenomena at sharp edges.

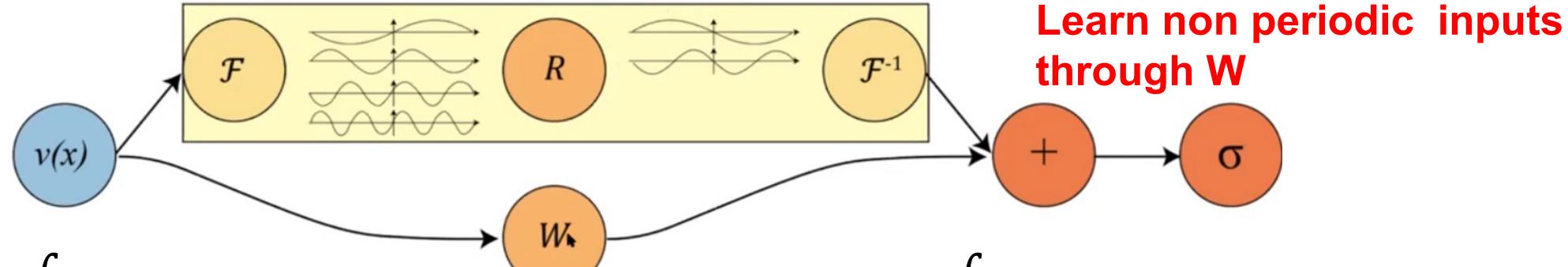


# Fourier Neural Operator (FNO)



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

FNO leverages neural networks with Fourier transforms



$$(\mathcal{F}f)_j(k) = \int_D f_j(x) e^{-2\pi i \langle x, k \rangle} dx ; \quad (\mathcal{F}^{-1}f)_j(x) = \int_D f_i(k) e^{2\pi i \langle x, k \rangle} dk$$

$$(\mathcal{K}(a; \phi)v_t)(x) = \mathcal{F}^{-1} \left( \mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v_t) \right) (x)$$

$$(\mathcal{K}(\phi)v_t)(x) = \mathcal{F}^{-1} \left( R_\phi \cdot \mathcal{F}(v_t) \right) (x)$$

$R_\phi$  is the Fourier transform of a periodic function parameterized by  $\phi$ .

Kovachki et al, Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs Journal of Machine Learning Research 24 (2023)

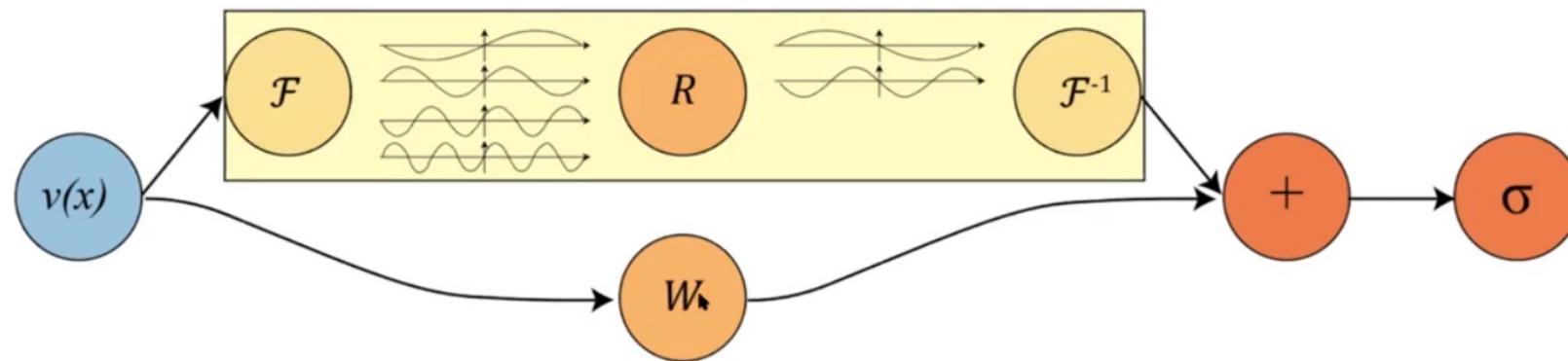
The CORE Consortium.

# Fourier Neural Operator (FNO)



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

FNO leverages neural networks with Fourier transforms to map its input function to the target function in the frequency domain, showcasing notably superior performance compared to traditional Convolutional Neural Networks (CNNs)



Learn non periodic inputs through W

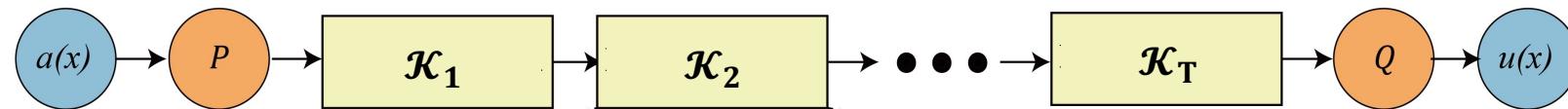
$$v_{l+1}(s) = \sigma \left( W_l v_l(s) + \int_D \kappa_l(s, z) v_l(z) dz + b_l(s) \right), \quad l = 0, \dots, L-1$$

# How about non-linear PDE's

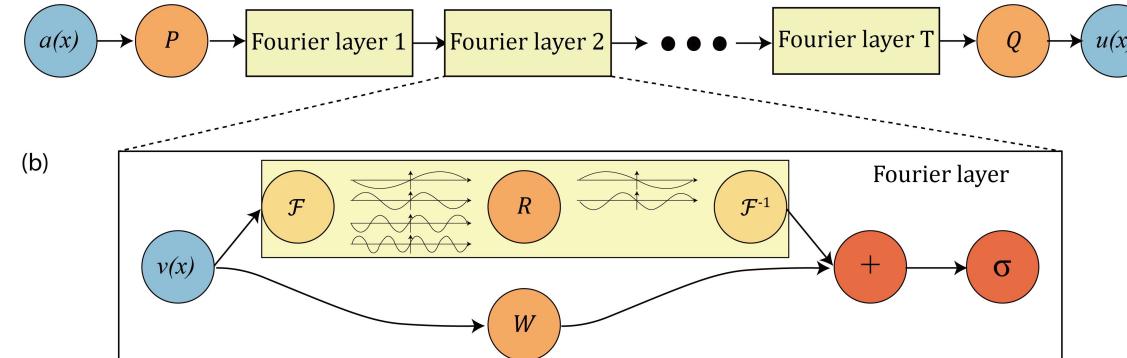


TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

- Embeddings → Lift to higher dimensions (P, Q)
- Compose several layers for non-linear pde's



- The R tensor is a multi-dimensional tensor of learnable weights that learns the high Fourier modes.



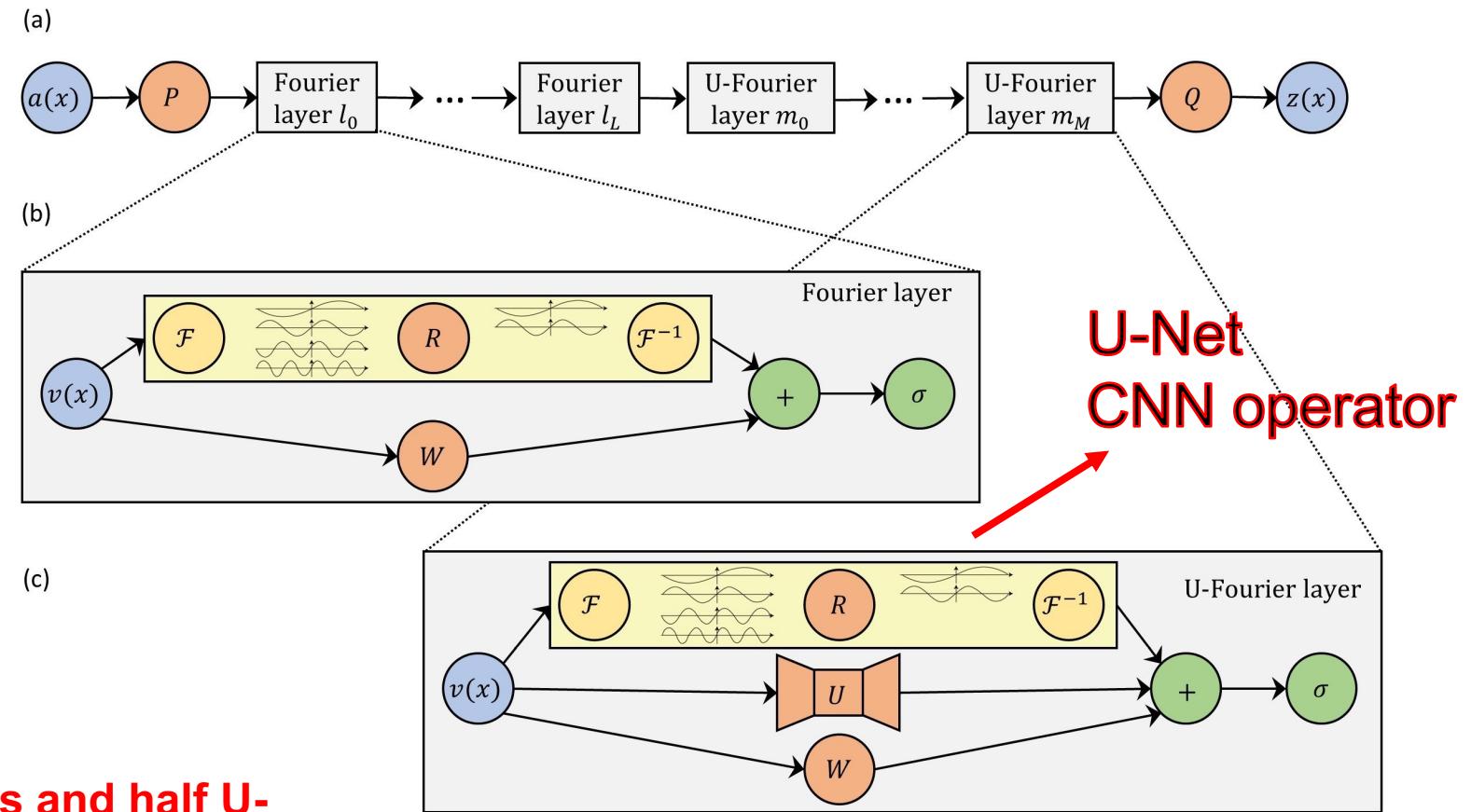
# U-FNO for better accuracy



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

The U-Net processes local convolution to enrich the representation power of the U-FNO in higher frequencies information

For the multi-phase flow problems →  
the architecture with half Fourier layers and half U-Fourier layers performs better



Wen et al, U-FNO - an enhanced Fourier neural operator-based deep-learning model for multiphase flow [Advances in Water Resources Volume 163, May 2022, 104180](#)

# Fourier neural operator (FNO) applied to Reservoir Simulation



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

## Goal:

Train a FNO that enables pressure and saturation predictions on:

- New/unseen permeability fields
- New/unseen well locations.
- New/unseen well schedules (controls).
- New number of wells.

## Input design:

The input tensor  $a(x) = (N_s, N_c, N_x, N_y, N_t)$

## where:

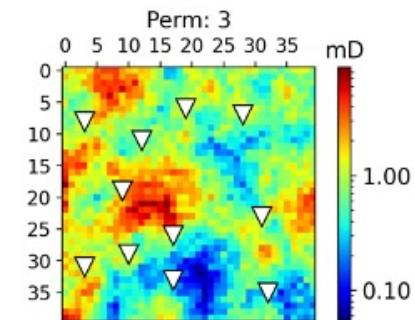
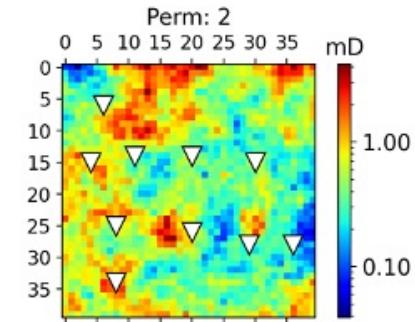
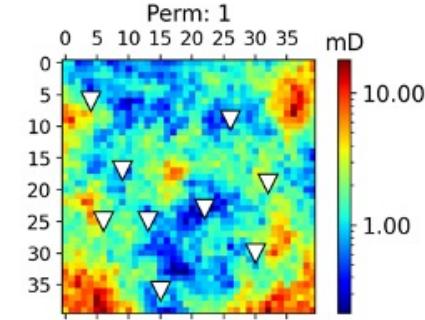
$N_s$ : number of samples

$N_c$ : number of channels

$N_x$ : number of x nodes

$N_y$ : number of y nodes

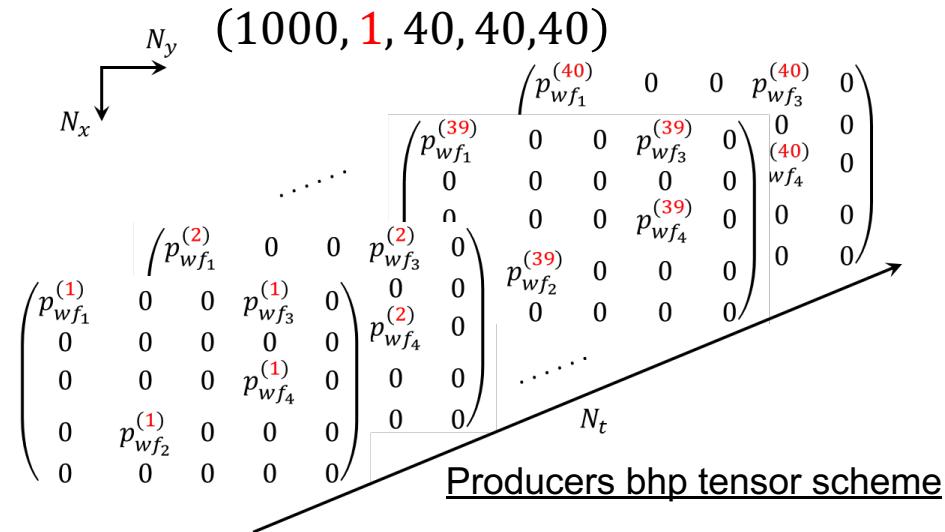
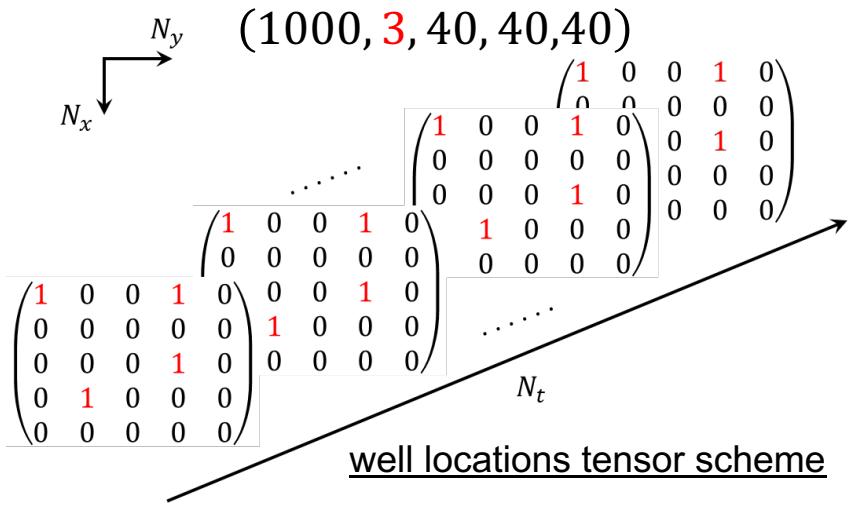
$N_t$ : number of t nodes



# U – Fourier neural operator: Training

## Training and testing:

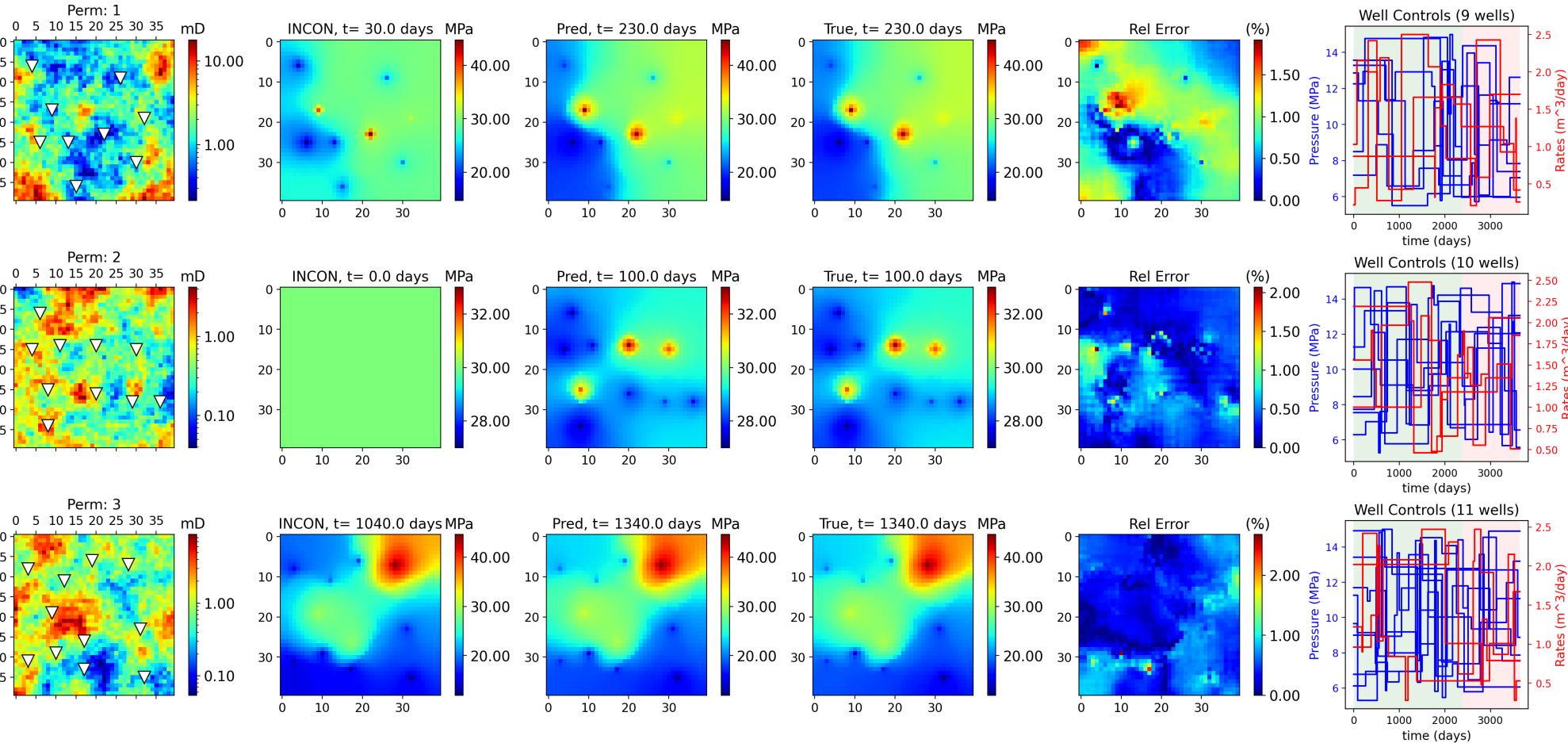
- Generate 4200 simulations for 40x40 different permeability fields with 6 wells (4 prod, 2 inj) in each field randomly distributed in space. The simulation runs from 0 to 2400 days, with results cache every 10 days.
  - 3500 for training, and 500 for validation, 200 for testing.
  - Testing samples include from 3 up to 12 wells, whereas training and validation samples are restricted to 6 wells only.
  - The training input tensor shape is: (1000, 6, 40, 40, 40).  
The 7 channels are: (permeability, prod\_bhps, inj\_rates, well\_locs, p\_incon, sw\_incon)
  - The output tensor shape is: (1000, 2, 40, 40, 40)



# U-FNO – PRESSURE PREDICTIONS

TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

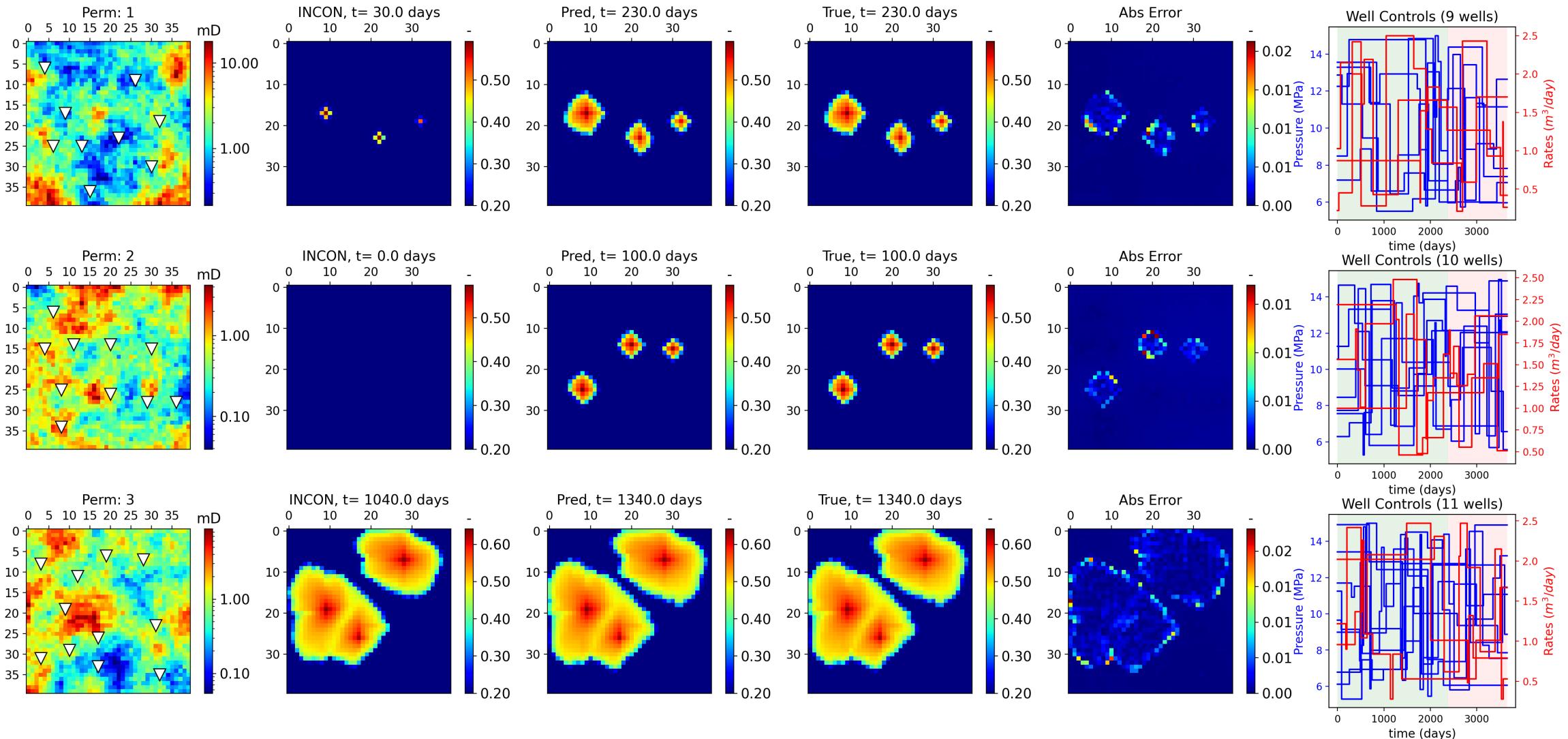
- 3500 training samples, 500 validation , 200 testing
- Training and validation samples have 6 wells (4 producers, 2 injectors)
- Testing samples could have from 3 up to 12 wells.



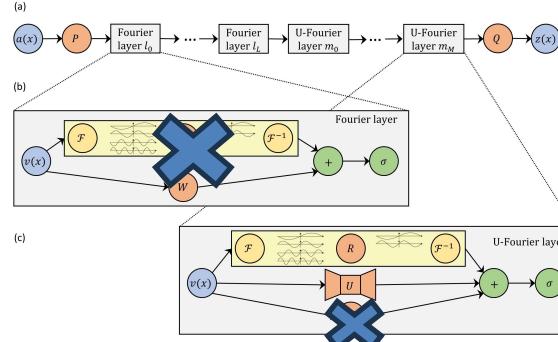
~ 3-4 hrs training

# U-FNO – SATURATRION PREDICTIONS

TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

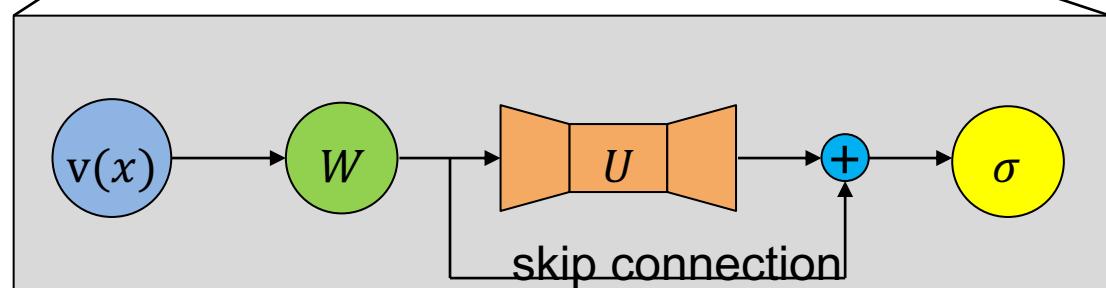
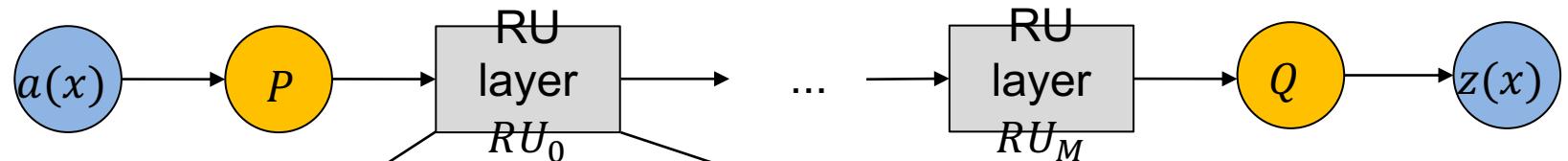


# OUR NEW ARCHITECTURE – RESIDUAL U-NET



**OUR model is 40% faster to train than U-FNO**

- The residual connection replaces the Fourier block.
- In contrast to Fourier, there is no lost information when considering the high modes only, because the skip connection will retain all information.
- 3RU-NET includes 3 RU layers
- 5RU-NET includes 5 RU layers



Added skip  
connection

# OURS vs U-FNO – ERRORS

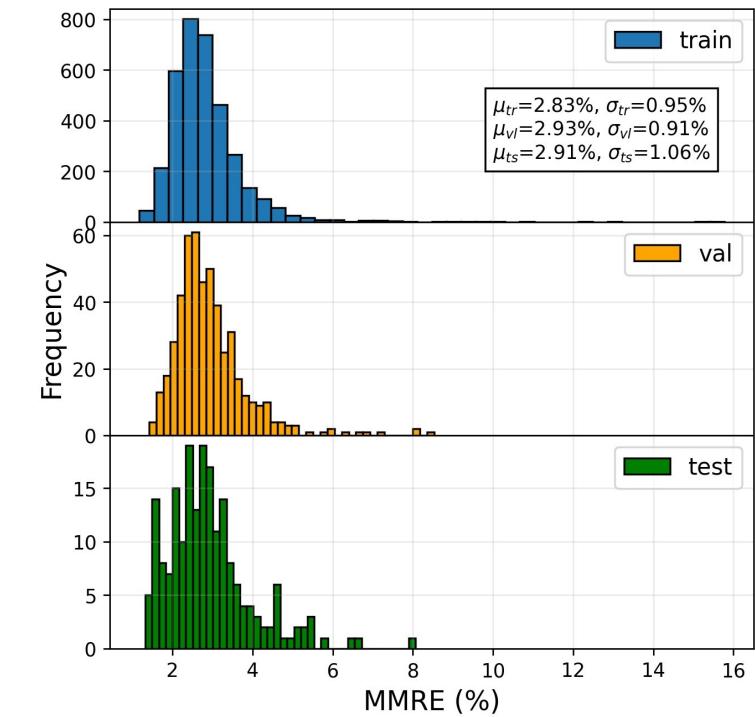
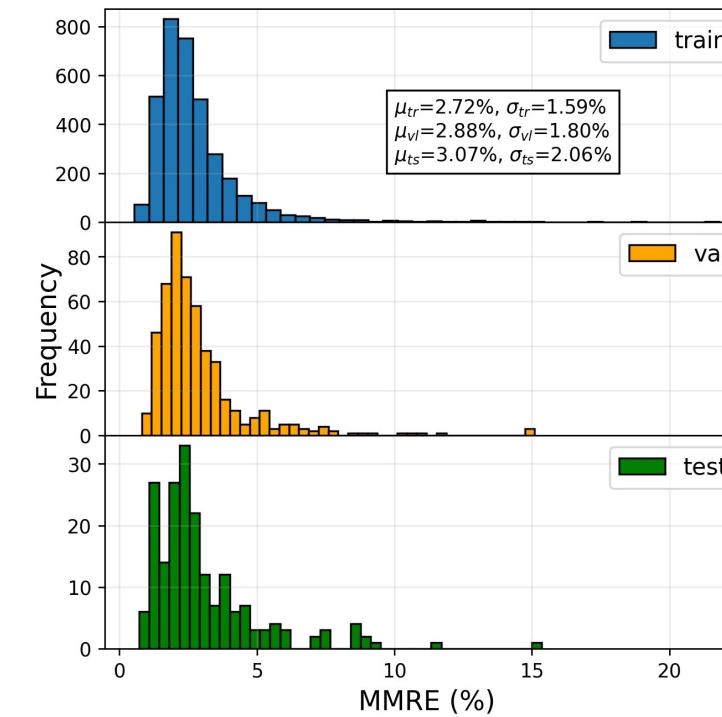
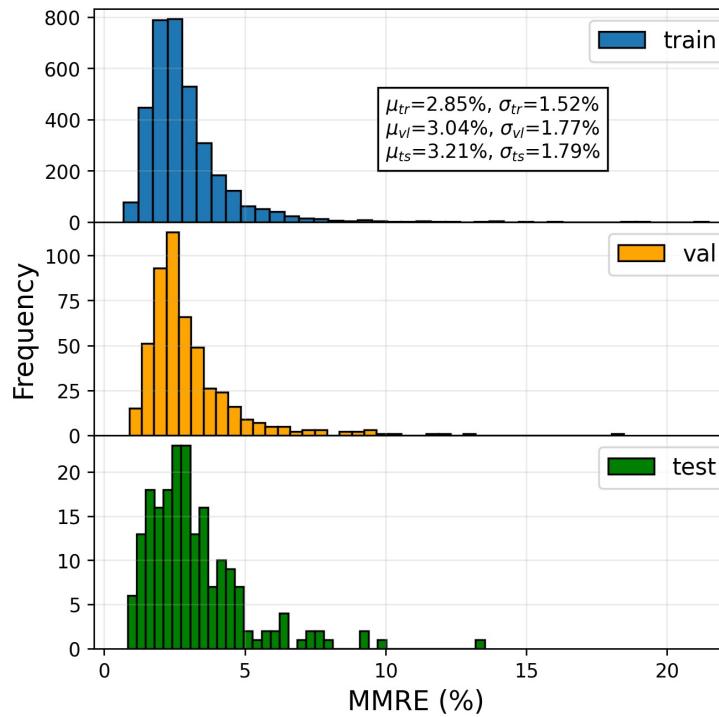


TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

- 3500 training samples, 500 validation , 200 testing
- Training and validation samples have 6 wells (4 producers, 2 injectors)
- Testing samples have from 3 up to 12 wells.

**OUR model is 40% faster to train than U-FNO**

U-FNO (3 U-Fourier blocks): 221 sec/epoch

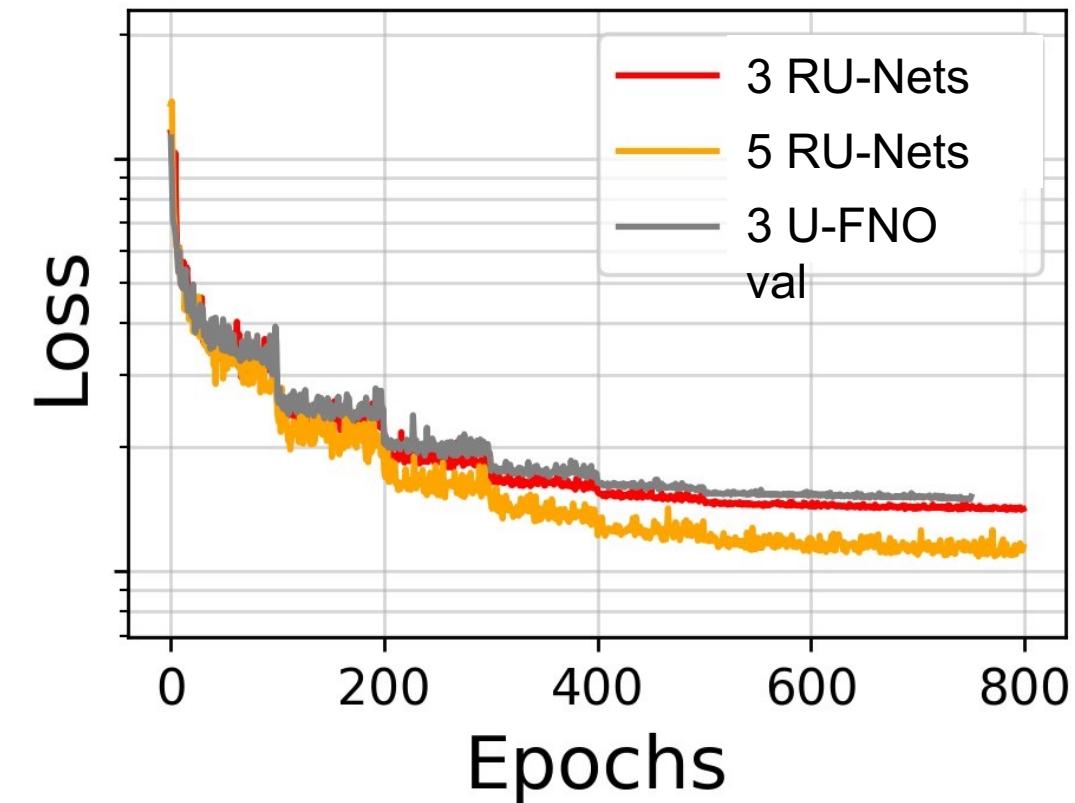
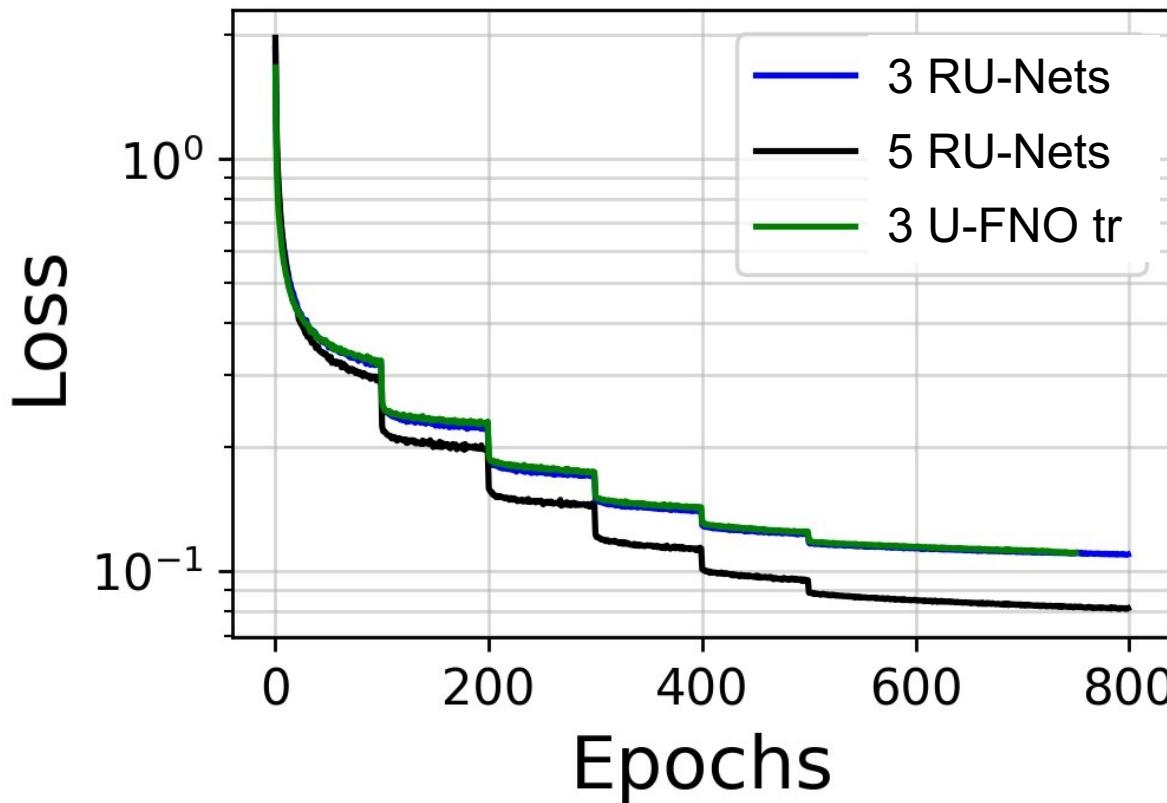


# OURS vs U-FNO – Training and Validation Losses



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

- 3500 training samples, 500 validation , 200 testing
- Training and validation samples have 6 wells (4 producers, 2 injectors)
- Testing samples have from 3 up to 12 wells.

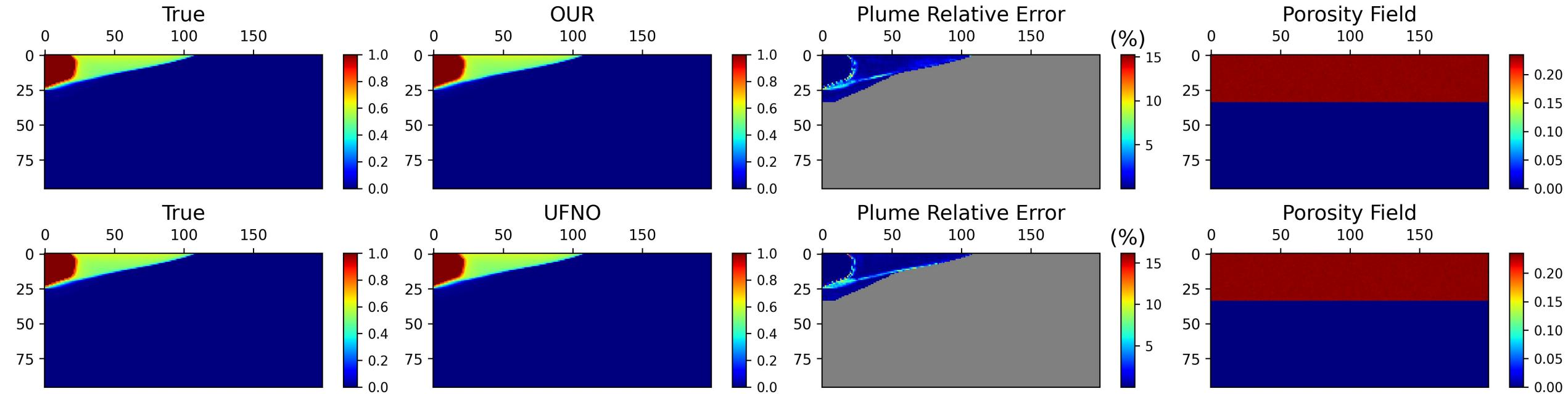


# OURS vs U-FNO BENCHMARKING – SATURATION PREDICTION



TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

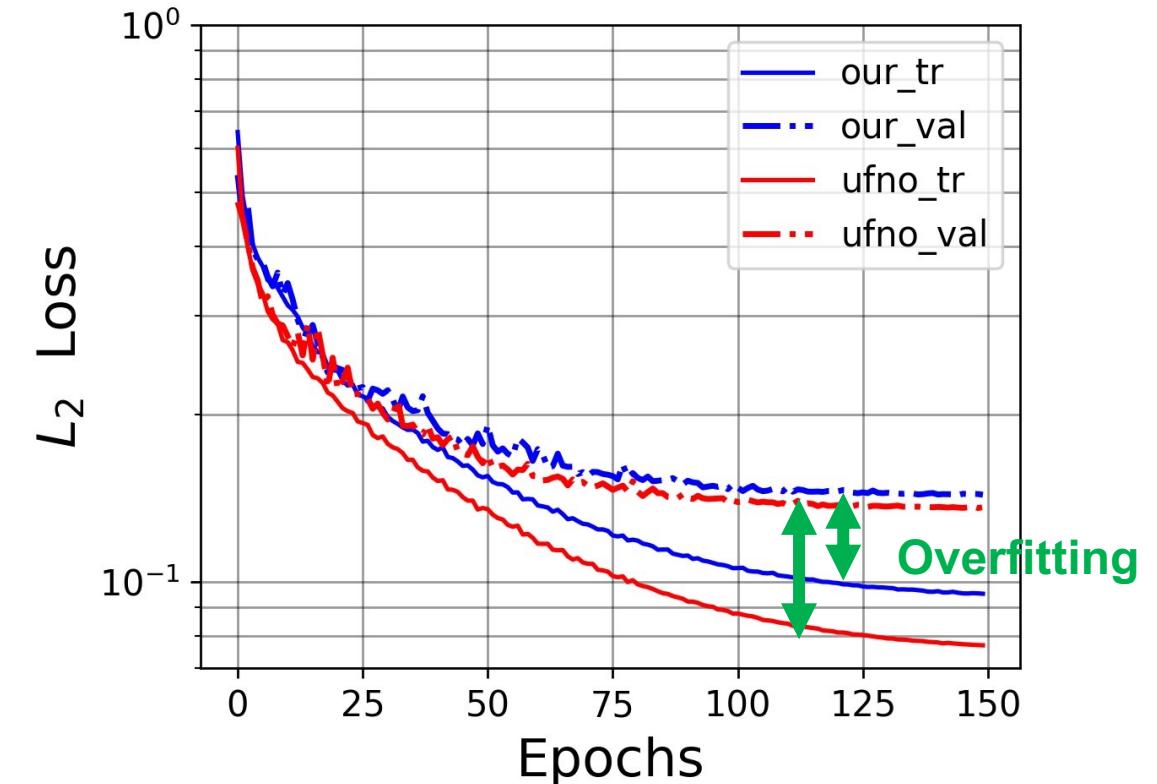
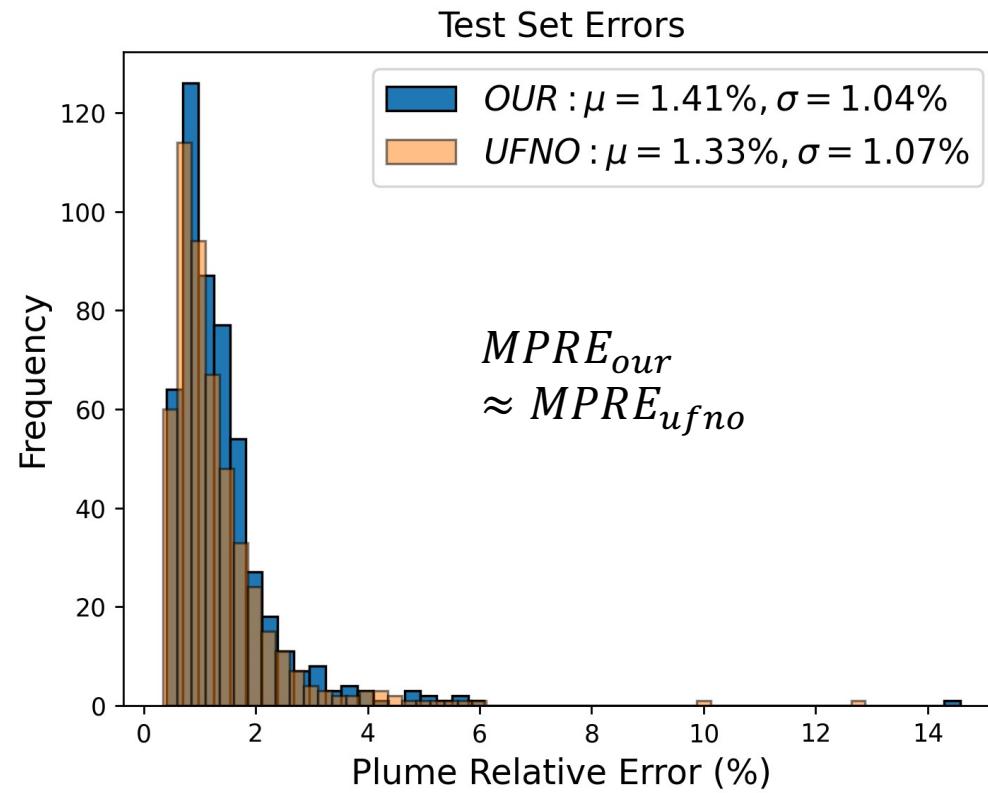
Benchmarking our model against U-FNO →  
with the same dataset used in the U-FNO paper (CCS case)



Wen et al, U-FNO - an enhanced Fourier neural operator-based deep-learning model for multiphase flow [Advances in Water Resources Volume 163, May 2022, 104180](#)

# OURS vs U-FNO BENCHMARKING ERRORS

Less overfitting in our model – the distance between training loss and validation loss denotes overfitting.



# Why this works? Perhaps an explanation...

NIKOLA B. KOVACHKI *et al.*, *OPERATOR LEARNING: ALGORITHMS AND ANALYSIS*,  
*arXiv:2402.15715*, Feb 2024

## ***Special case of an averaging neural operator***

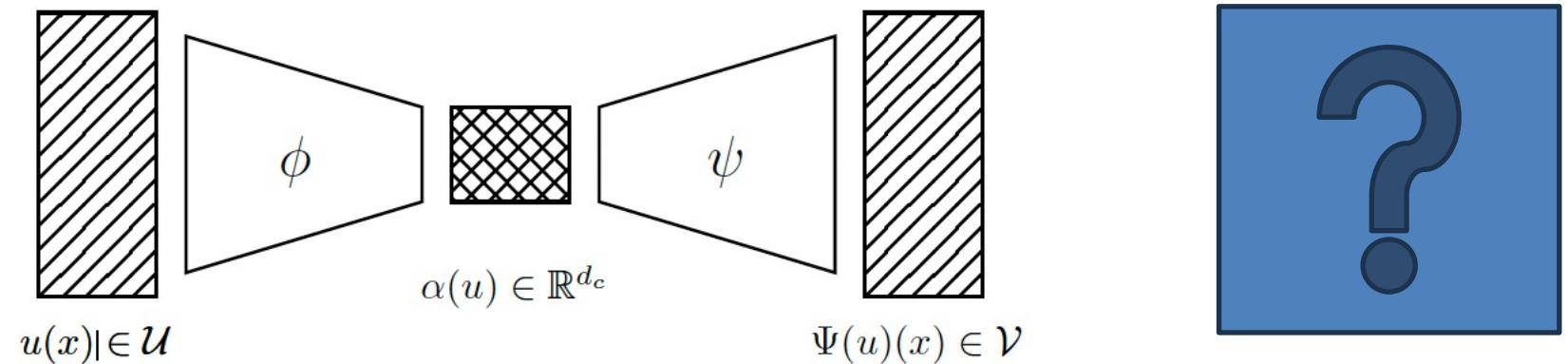
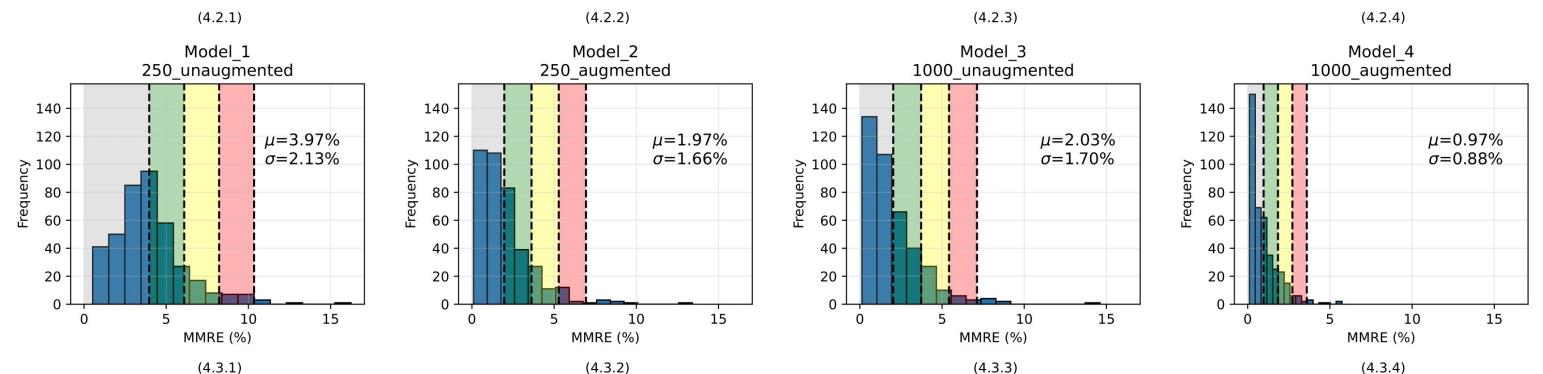
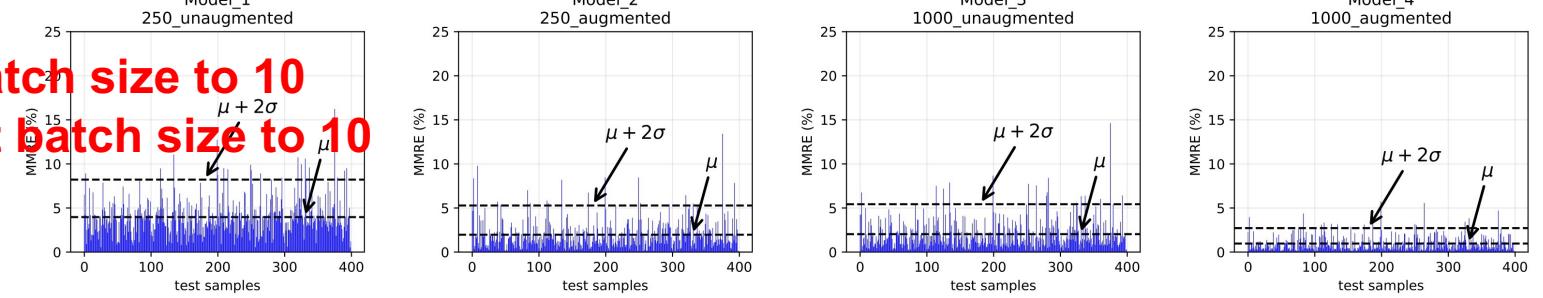
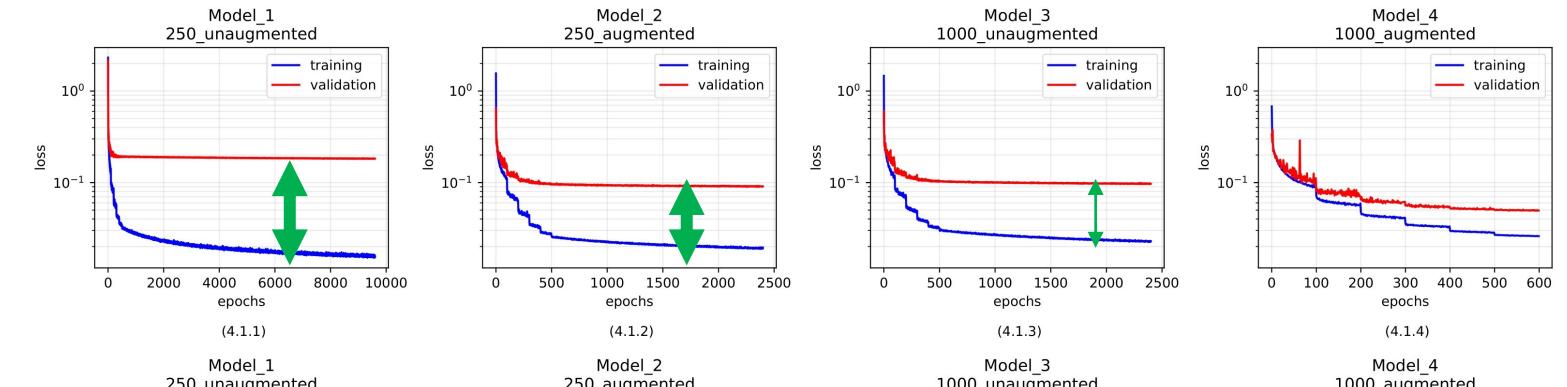


FIGURE 5. Special case of an averaging neural operator, illustrated as a nonlinear encoder-decoder architecture; with encoder  $u \mapsto \alpha = \int_{\mathfrak{D}} \phi(u(y), y) dy$ , and decoder  $\alpha \mapsto \psi(\alpha, \cdot)$ .

# DATA AUGMENTATION REDUCES OVERFITTING

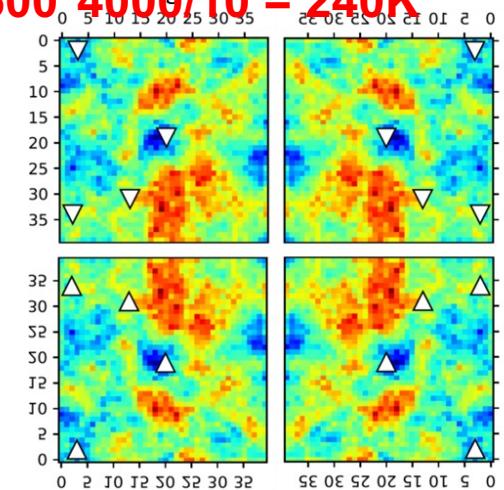
	# samples	Epochs
Model 1	250 base	9600
Model 2	250 base + aug	2400
Model 3	1000 base	2400
Model 4	1000 base + aug	600



Model 1 →  $9600 * 250 / 10 = 240K \rightarrow$  Kept batch size to 10

Model 2/3 →  $2400 * 1000 / 10 = 240K \rightarrow$  Kept batch size to 10

Model 4 →  $600 * 4000 / 10 = 240K$



# OUR MODEL FOR CCS ON IRREGULAR GEOMETRIES SATURATION PREDICTIONS – ONGOING WORK

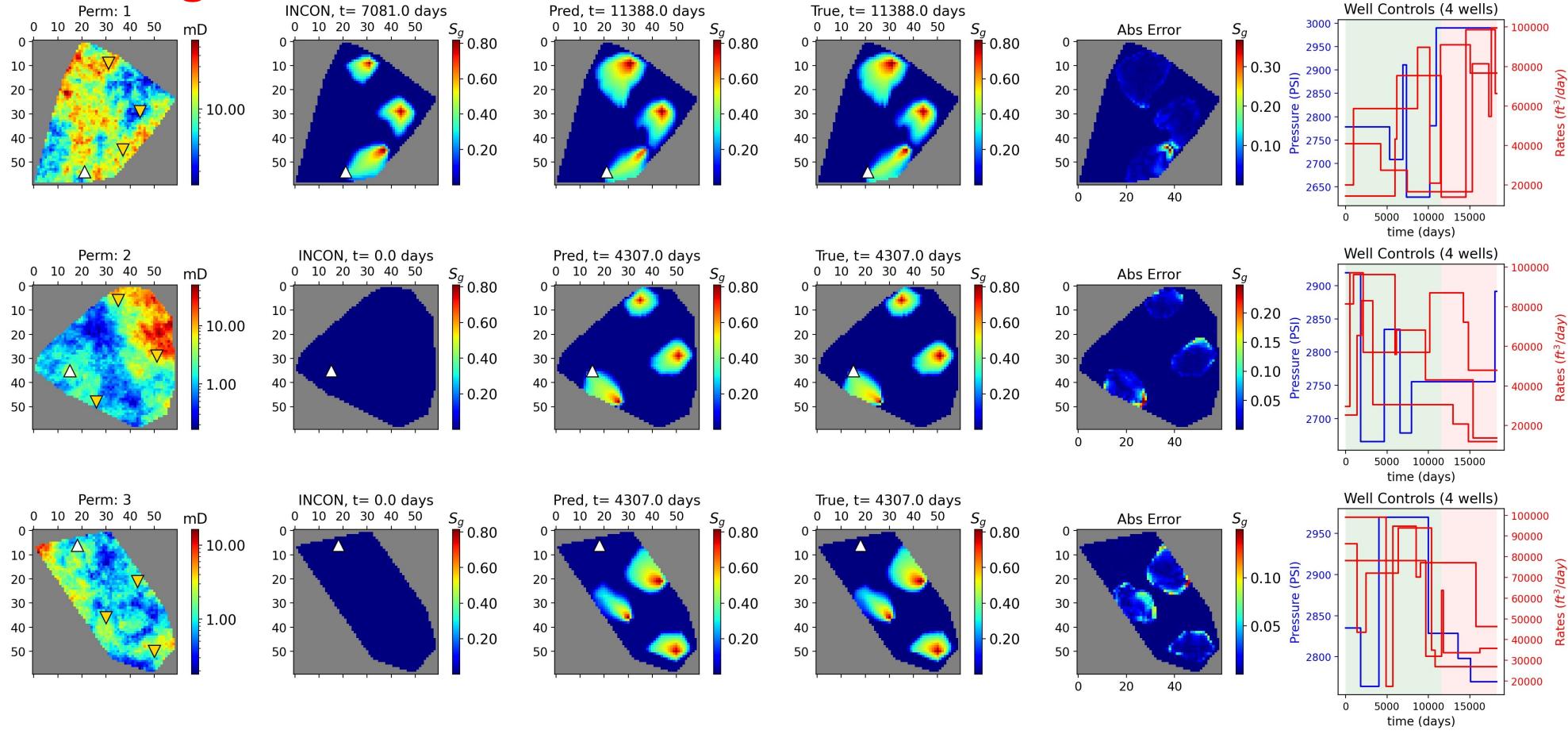


TEXAS A&M UNIVERSITY  
Harold Vance Department of  
Petroleum Engineering

## Changing the geometry of the reservoir

60X60 → 1500 ft each grid block

3 CO<sub>2</sub> injectors, 1 producer



Predicting on irregular geometries allows using all depleted reservoirs simulation data to train a global reservoir digital twin that is suitable for all shapes and geologies of different reservoirs.