

Solving inverse problems with PINNs

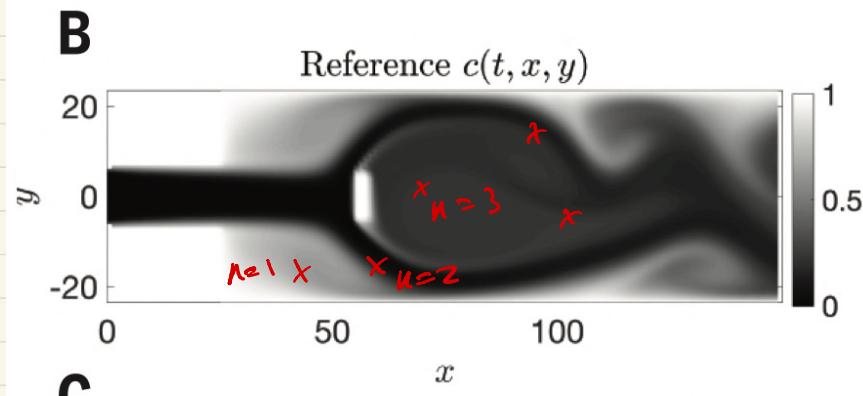
Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations

MAZIAR RAISI , ALIREZA YAZDANI , AND GEORGE EM KARNIADAKIS  [Authors Info & Affiliations](#)

SCIENCE • 30 Jan 2020 • Vol 367, Issue 6481 • pp. 1026-1030 • DOI: 10.1126/science.aaw4741

Inverse problem learning goal

Given: Training data = $\{t^n, x^n, y^n, z^n, c^n\}_{n=1}^N$



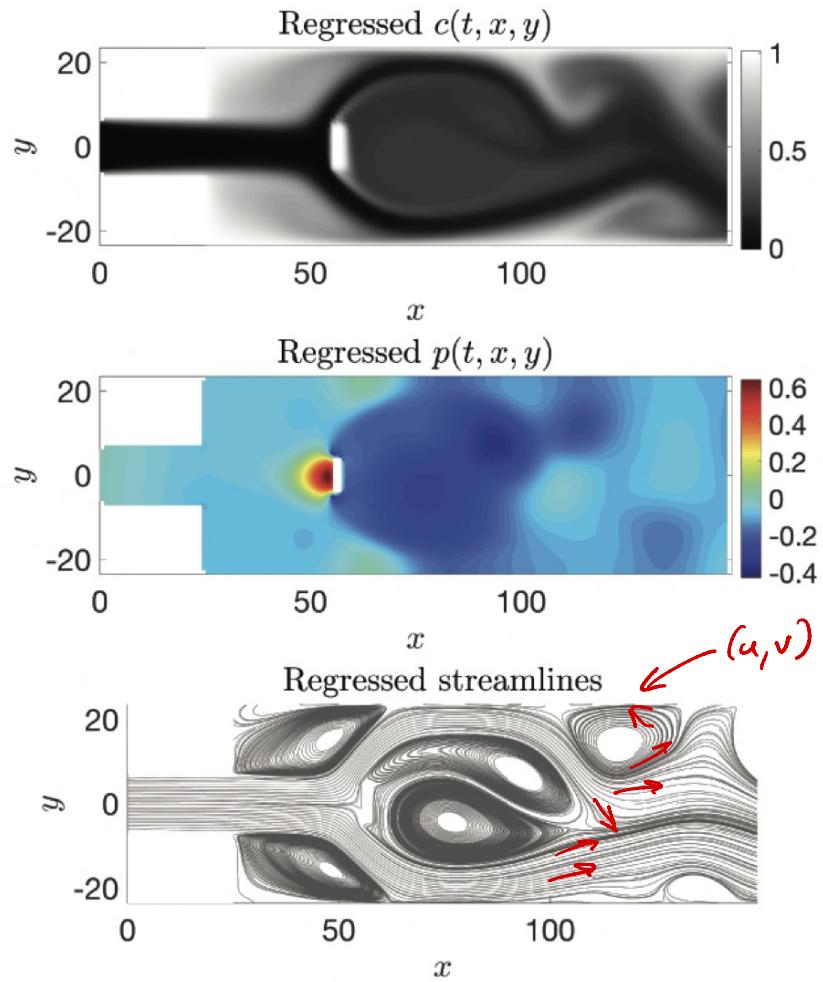
(u, v, w) velocity in (x, y, z) dirs.

Find: Learned model : $(t, x, y, z) \mapsto (c, u, v, w, p)$

Should match data and "obey physics"

Learned model

$$(t, x, y, z) \longmapsto (c, u, v, w, p)$$



Physics

Continuity equation:

$$c_t + u c_x + v c_y + w c_z = \text{Pe}^{-1} (c_{xx} + c_{yy} + c_{zz}),$$

Navier - Stokes equations:
Burger's eqn

$$u_t + u u_x + v u_y + w u_z = -p_x + \text{Re}^{-1} (u_{xx} + u_{yy} + u_{zz}),$$

$$v_t + u v_x + v v_y + w v_z = -p_y + \text{Re}^{-1} (v_{xx} + v_{yy} + v_{zz}),$$

$$w_t + u w_x + v w_y + w w_z = -p_z + \text{Re}^{-1} (w_{xx} + w_{yy} + w_{zz}),$$

$$u_x + v_y + w_z = 0.$$

Training setup

Residual points : $\{t^m, x^m, y^m, z^m\}_{m=1}^M$

Residuals :

- $r_1: e_1 := c_t + uc_x + vc_y + wc_z - \text{Pe}^{-1}(c_{xx} + c_{yy} + c_{zz}),$
- $r_2: e_2 := u_t + uu_x + vu_y + wu_z + p_x - \text{Re}^{-1}(u_{xx} + u_{yy} + u_{zz}),$
- $r_3: e_3 := v_t + uv_x + vv_y + wv_z + p_y - \text{Re}^{-1}(v_{xx} + v_{yy} + v_{zz}),$
- $r_4: e_4 := w_t + uw_x + vw_y + ww_z + p_z - \text{Re}^{-1}(w_{xx} + w_{yy} + w_{zz}),$
- $r_5: e_5 := u_x + v_y + w_z.$

Loss :

$$\text{MSE} = \frac{1}{N} \sum_{n=1}^N |c(t^n, x^n, y^n, z^n) - c^n|^2$$

← NN

$$+ \sum_{i=1}^5 \frac{1}{M} \sum_{m=1}^M |e_i(t^m, x^m, y^m, z^m)|^2$$

← PI

+ BCS ?

Notes

- Network: 10 layers
250 neurons per layer

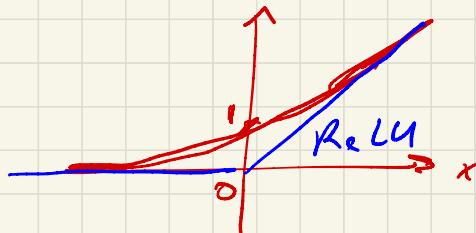
- swish activation function:

$$\text{swish}(x) := x \text{ sigmoid}(x) = x / (1 - \exp(-x))$$

sigmoid : $\frac{1}{1+e^{-x}}$

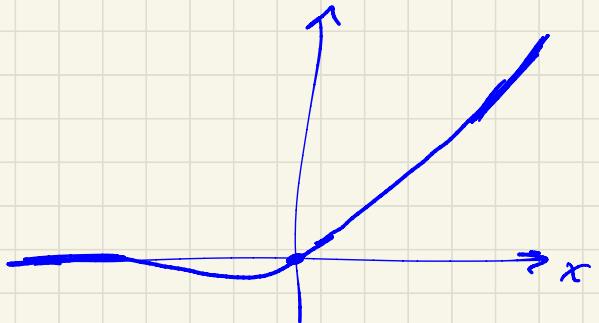


swish : $\frac{x}{1-e^{-x}}$

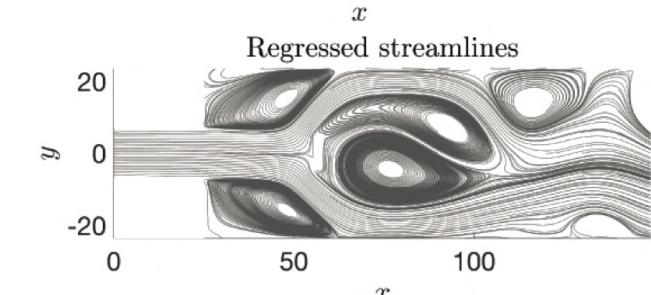
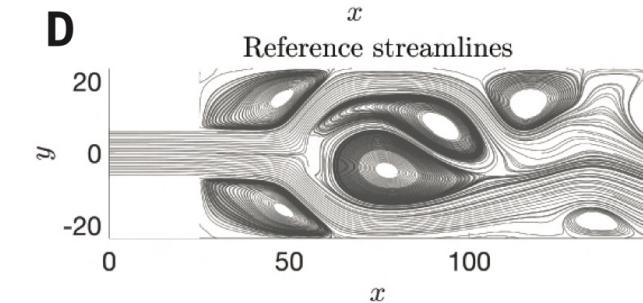
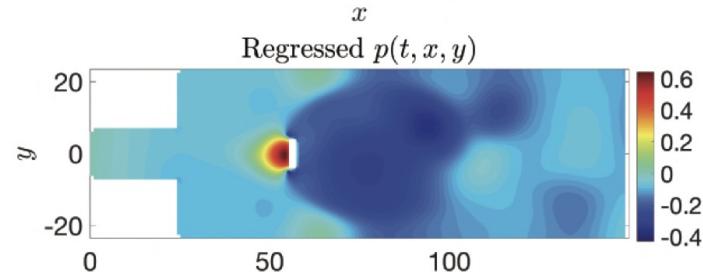
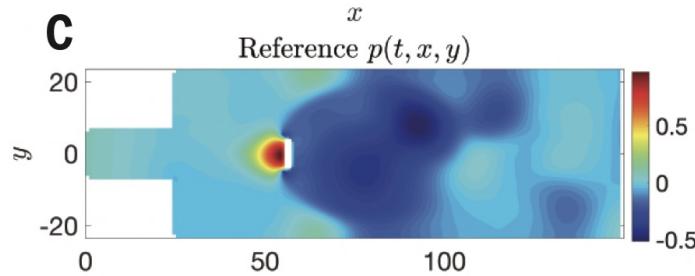
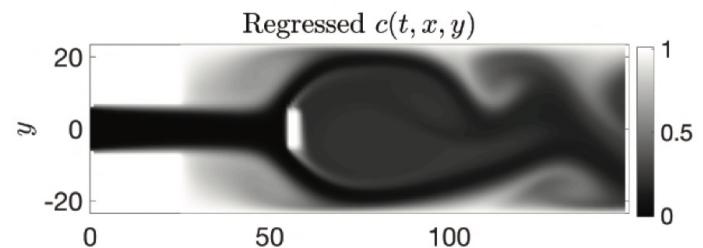
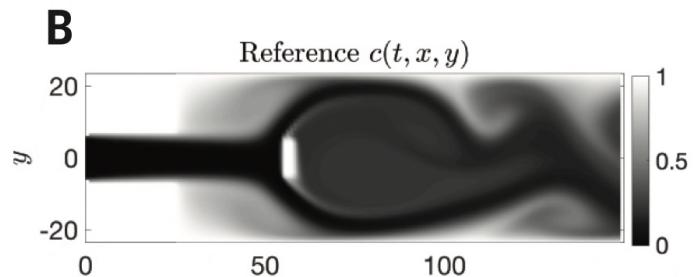


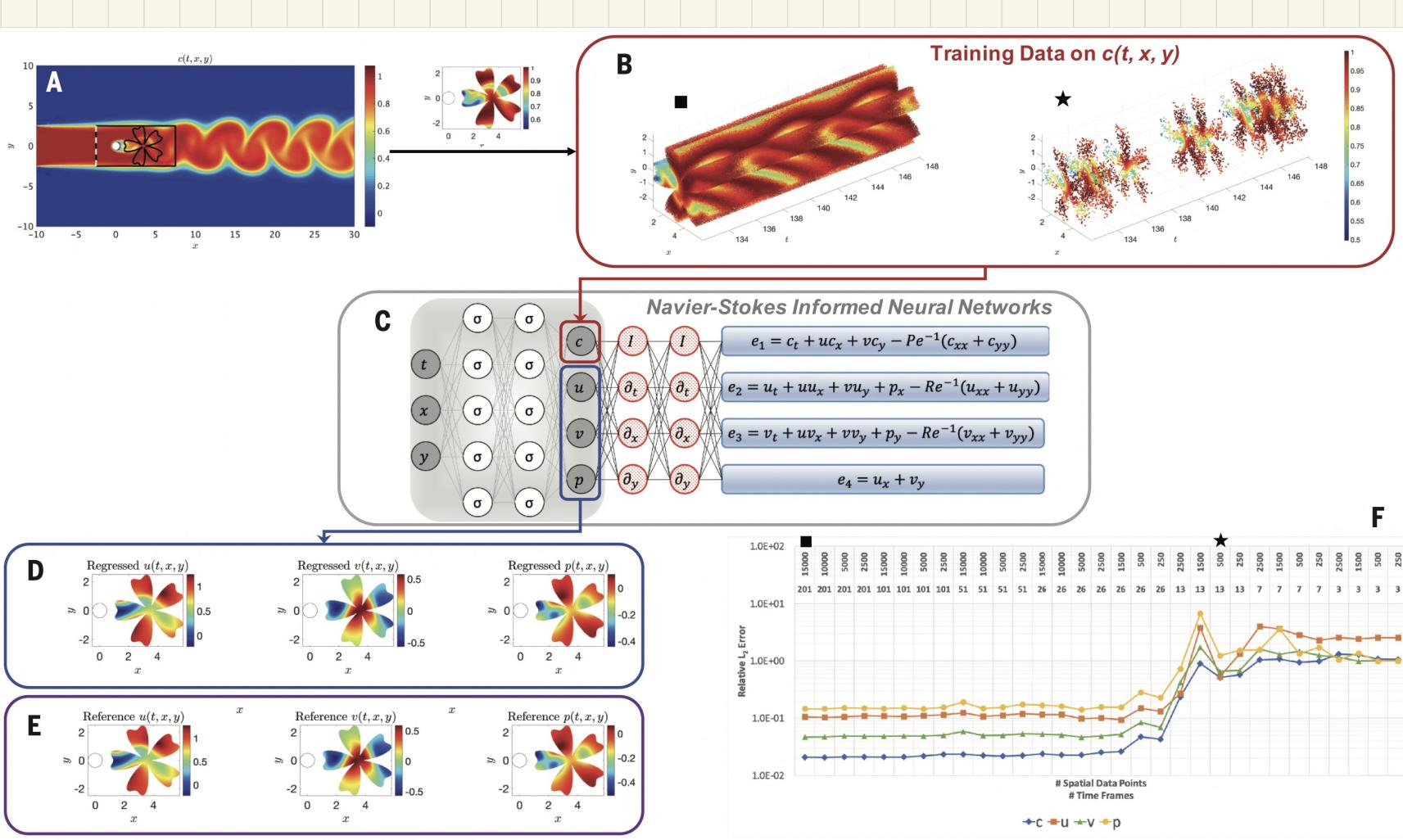
$$\frac{x}{1-e^{-x}} = \frac{x}{1-(1-x+x^2+\dots)} = \frac{x}{x-x^2} = \frac{1}{1-x} = 1+x$$

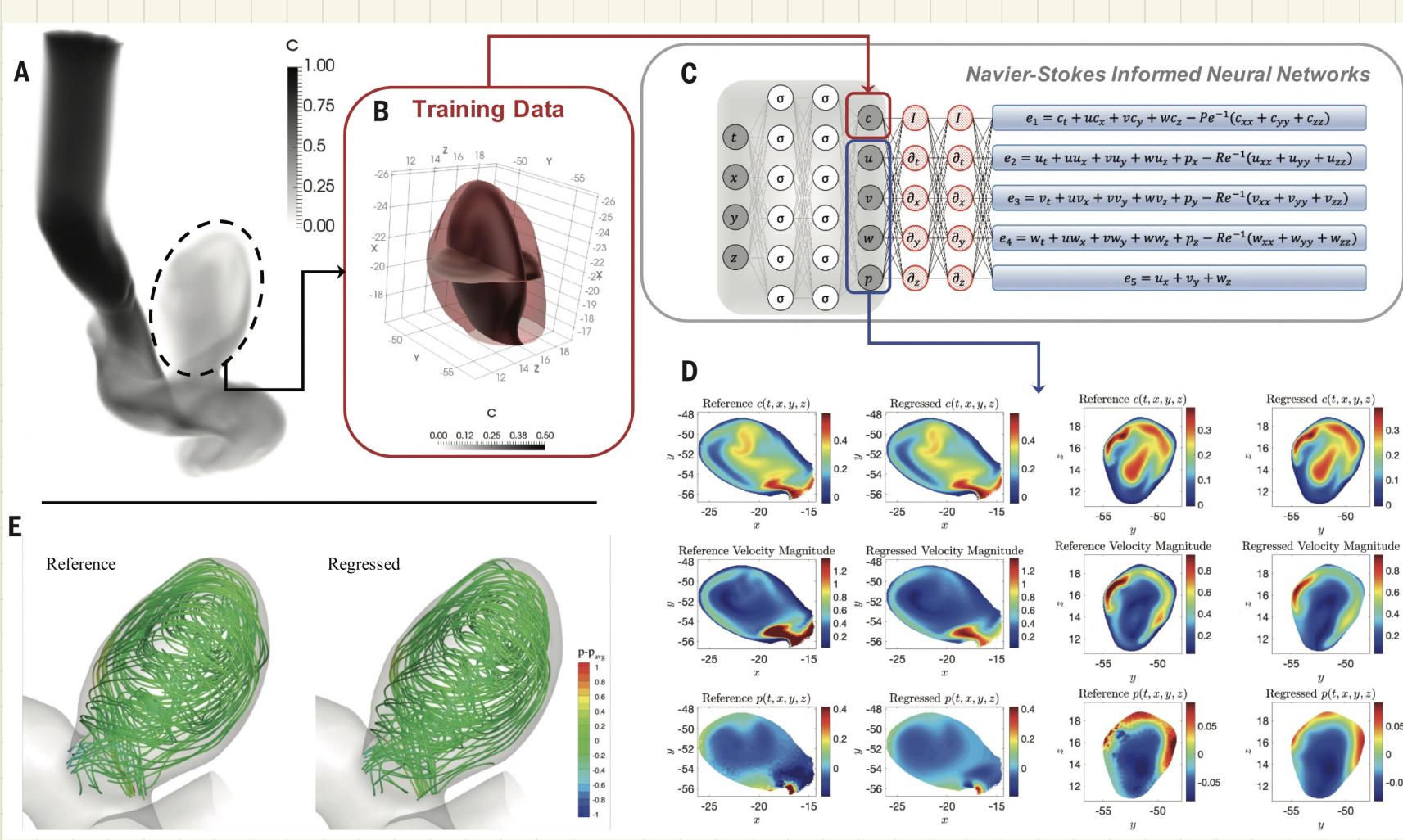
$$\frac{x}{1+e^{-x}}$$



Verification against test data







Why not ReLU?

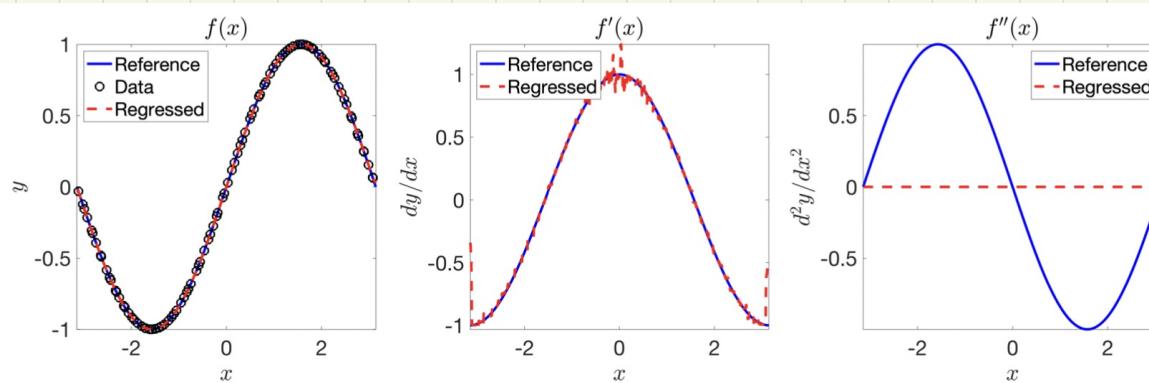
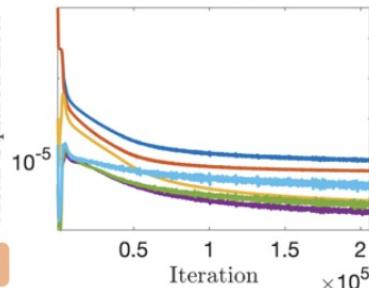


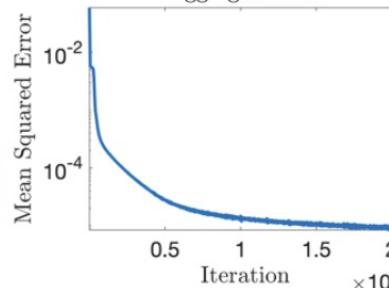
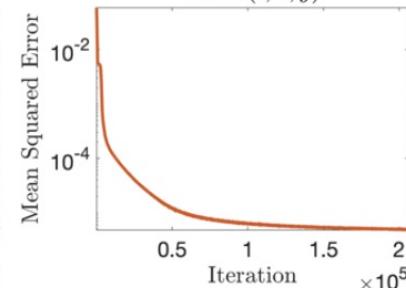
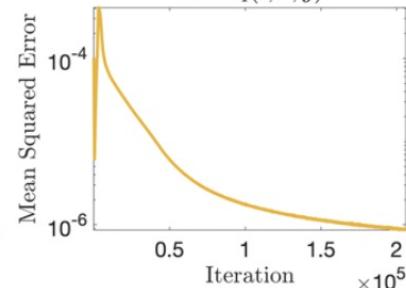
Fig. S2. Regressing data generated from $f(x) = \sin(x)$ using the ReLU activation function. A neural network with ReLU activation can accurately approximate $f(x)$ (left panel), while it leads to a non-smooth regression for the first derivative of f (middle panel) and a vanishing second derivative of f (right panel).

Mean Squared Error

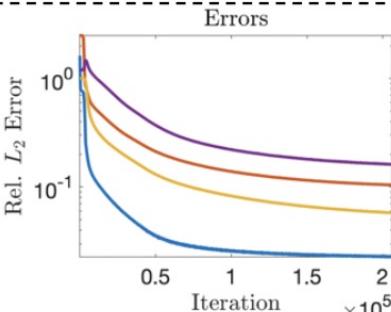
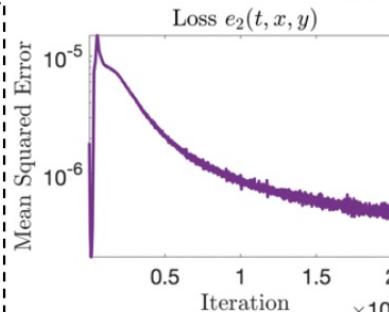
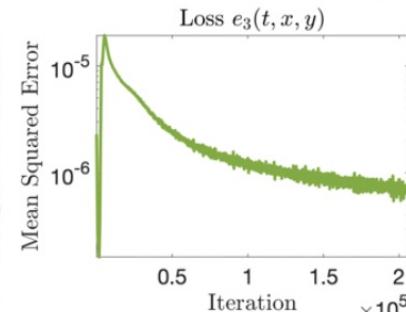
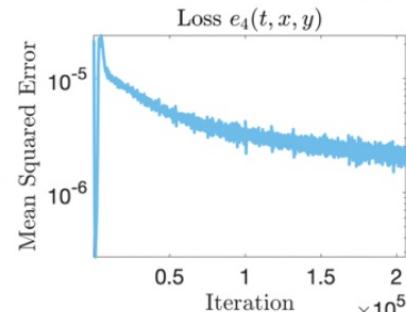
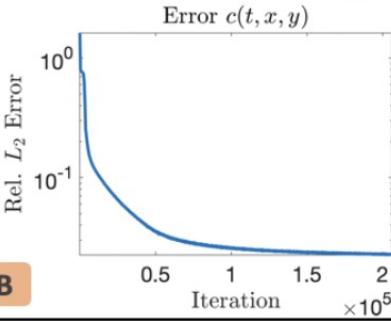
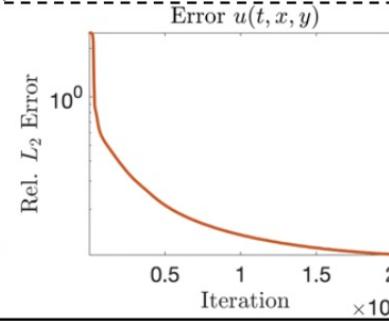
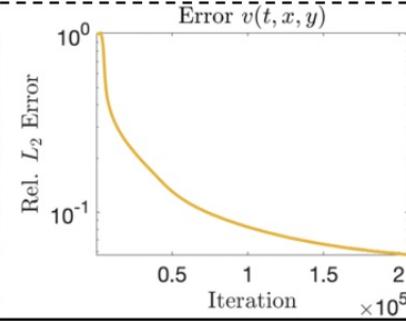
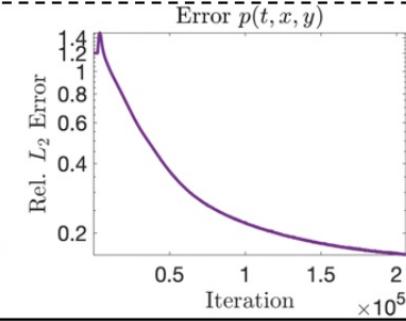
Losses



Aggregate Loss

Loss $c(t, x, y)$ Loss $e_1(t, x, y)$ Rel. L_2 Error

Errors

Loss $e_2(t, x, y)$ Loss $e_3(t, x, y)$ Loss $e_4(t, x, y)$ Rel. L_2 ErrorError $c(t, x, y)$ Error $u(t, x, y)$ Error $v(t, x, y)$ Error $p(t, x, y)$ 

B