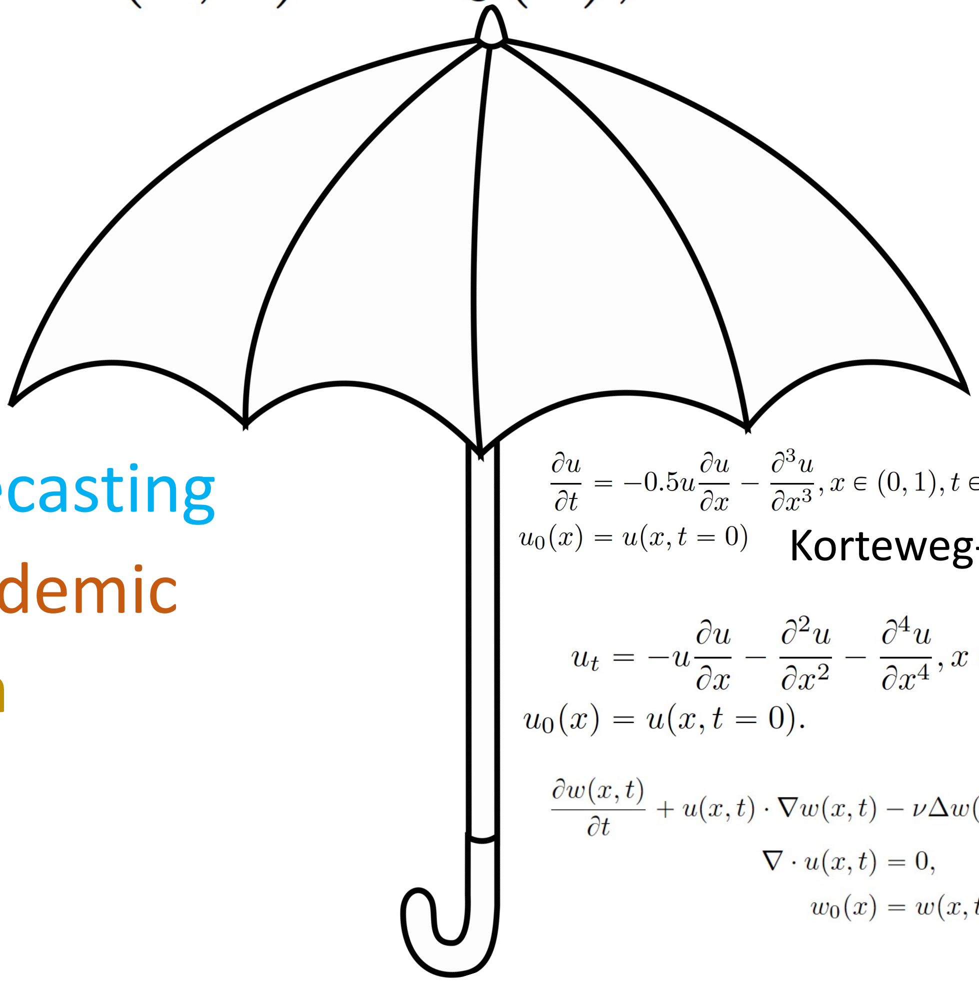
# Non-linear Operator Approximations for Initial Value Problems

**Gaurav Gupta, Xiongye Xiao, Radu Balan, Paul Bogdan**

# Initial Value Problems

$$u_t = \mathcal{F}(t, u), \quad x \in \Omega$$
$$u(x, 0) = u_0(x), \quad x \in \Omega$$



Dynamical systems, Forecasting

Kermack-McKendrik- epidemic

Hodgkin-Huxley - neuron

$$\frac{\partial u}{\partial t} = -0.5u\frac{\partial u}{\partial x} - \frac{\partial^3 u}{\partial x^3}, x \in (0,1), t \in (0,1]$$
$$u_0(x) = u(x, t=0)$$

Korteweg-de Vries

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + v\frac{\partial^2 u}{\partial x^2}, x \in (0, 2\pi), t \in (0,1]$$
$$u_0(x) = u(x, t=0).$$

Burgers’

$$u_t = -u\frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^4 u}{\partial x^4}, x \in (0,1), t \in (0,1]$$
$$u_0(x) = u(x, t=0).$$

Kuramoto-Sivashinsky

$$\frac{\partial w(x,t)}{\partial t} + u(x,t) \cdot \nabla w(x,t) - \nu \Delta w(x,t) = f(x), \quad x \in (0,1)^2, t \in (0,T]$$
$$\nabla \cdot u(x,t) = 0, \quad x \in (0,1)^2, t \in [0,T]$$
$$w_0(x) = w(x, t=0), \quad x \in (0,1)^2$$

Navier-Stokes

# Exponential Operators

| Equation | Solution |
| --- | --- |
| $\frac{d\mathbf{u}}{dt} = \mathbf{A}\mathbf{u}, u(t=0) = \mathbf{u}_0$ (Linear ODE) | $\mathbf{u}(t=\tau) = e^{t\mathbf{A}}\mathbf{u}_0$ |
| $u_t = \frac{\partial^2 u}{\partial x^2}, u(x,0) = u_0(x)$ (Heat equation) | $u(x,\tau) = e^{\tau \frac{\partial^2}{\partial x^2}} u_0(x)$ |
| $u_t = \mathcal{L}u + \mathcal{N}f(u), u(x,0) = u_0(x)$ | $u(x,\tau) = e^{\tau\mathcal{L}}u_0(x) + \int_0^\tau e^{(\tau-t)\mathcal{L}}\mathcal{N}f(u(x,t))dt$ |

Exponential operators

Non-linear

# Data Efficiency

## Neural Operator

$$T = T_1 \circ \underbrace{\sigma}_{\substack{\text{Non-linearity} \\ \text{e.g. ReLU}}} \circ T_2 \ldots \sigma \circ \underbrace{T_N}_{\text{Canonical linear operator}}$$

🤔

Can we do better for getting compact models?

Explicitly embed exponential operators

# Padé Approximation

Exponential attains Taylor series but truncation not efficient

Less error by Padé approximation

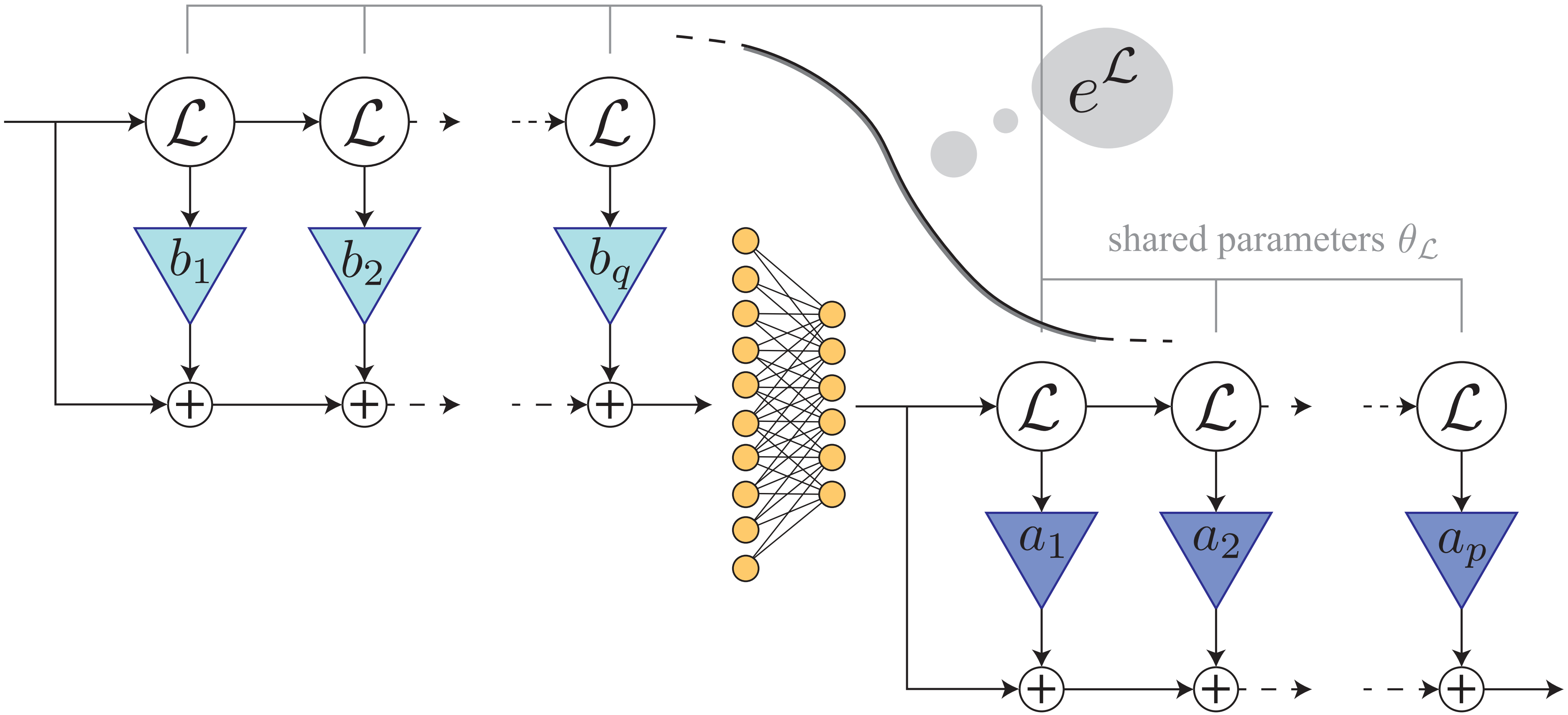
Used by Matlab, Scipy for Matrix exponentials `expm`

$$e^{\mathcal{L}} \approx r_{pq}(\mathcal{L}) = \left( \sum_{j=0}^{q} b_j \underbrace{\mathcal{L} \circ \mathcal{L} \circ \ldots \circ \mathcal{L}}_{j\text{-times}} \right)^{-1} \left( \sum_{j=0}^{p} a_j \underbrace{\mathcal{L} \circ \mathcal{L} \circ \ldots \circ \mathcal{L}}_{j\text{-times}} \right)$$

$b_j$, $a_j$ → pre-determined

# Padé Neural Operator

$$e^{\mathcal{L}} \approx r_{pq}(\mathcal{L}) = \left( \sum_{j=0}^{q} b_j \underbrace{\mathcal{L} \circ \mathcal{L} \circ \ldots \circ \mathcal{L}}_{j-\text{times}} \right)^{-1} \left( \sum_{j=0}^{p} a_j \underbrace{\mathcal{L} \circ \mathcal{L} \circ \ldots \circ \mathcal{L}}_{j-\text{times}} \right)$$



# Gradient bounds

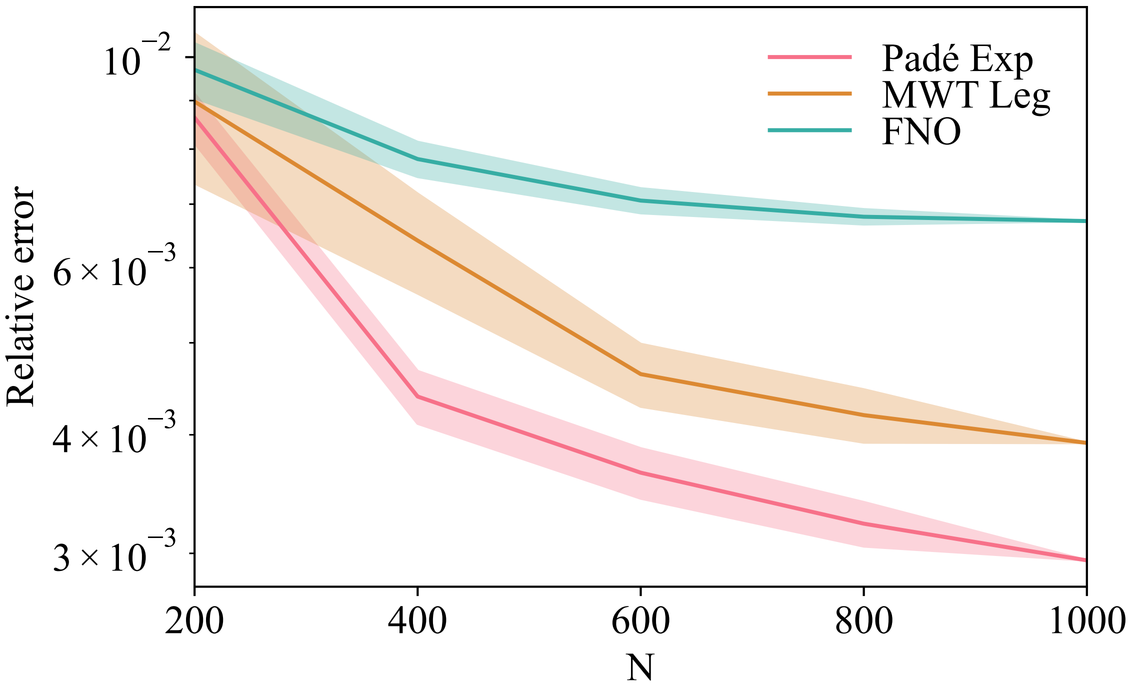
Theorem: Given a linear operator $\mathcal{L}$ and non-linear layer $\sigma(Wx + b)$, the gradients of the operation $y := F(x, \theta_{\mathcal{L}}, W, b) = [p/q]e^{\mathcal{L}}(x)$ are bounded as

$$\left\| \frac{\partial y}{\partial \theta_{\mathcal{L}}} \right\| \leqslant \exp(\|\mathcal{L}\|) \left(\|b\|_2 + \|W\| \|x\|_2\right) \left( \sum_{j=1}^{n_\theta} \left\| \frac{\partial \mathcal{L}}{\partial \theta_j} \right\|^2 \right)^{1/2},$$

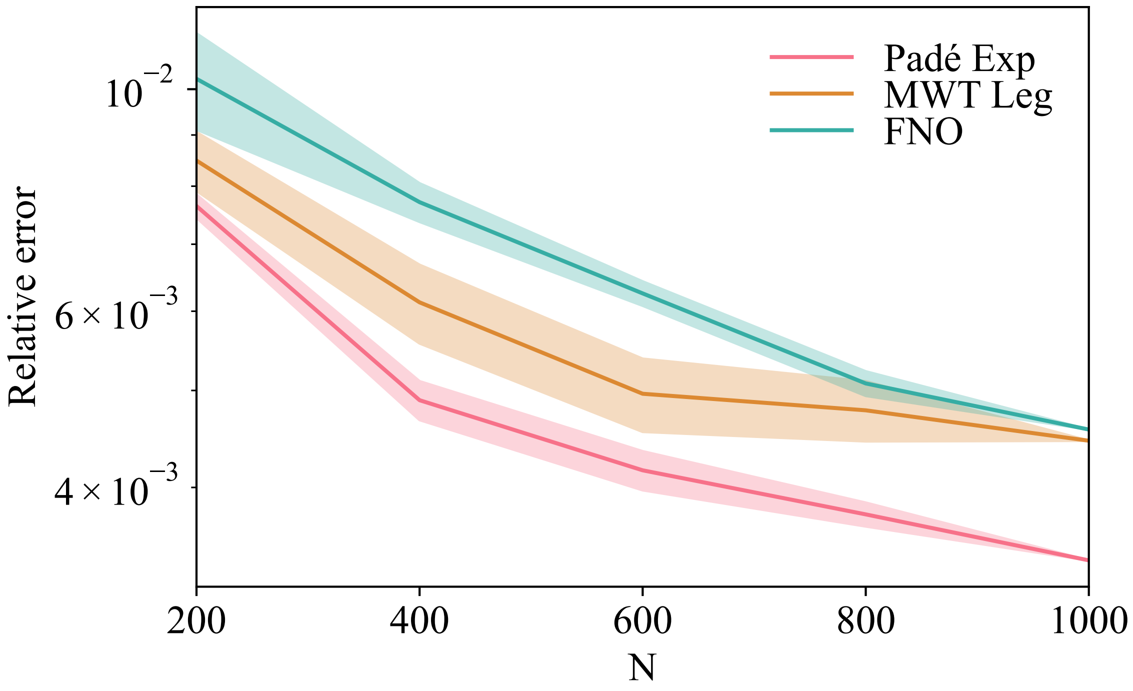
$$\left\| \frac{\partial y}{\partial W} \right\| \leqslant \exp(\|\mathcal{L}\|) \|x\|_2,$$

$$\left\| \frac{\partial y}{\partial b} \right\| \leqslant \exp\left( \frac{p}{p+q} \|\mathcal{L}\| \right).$$

# Data efficiency on 1D datasets



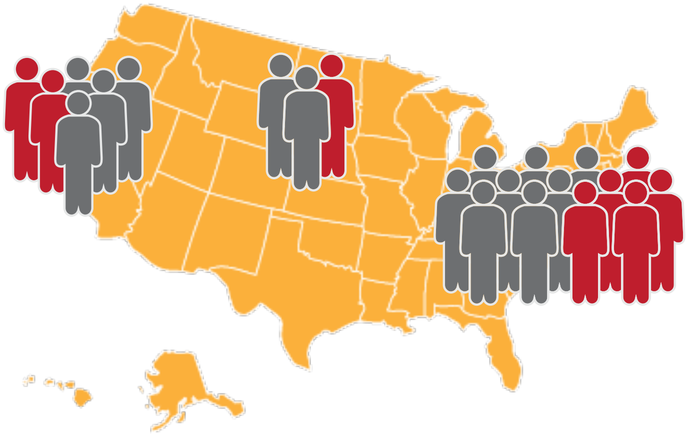

KdV equation




KS equation

# Epidemic Forecasting (COVID19)

Predict Confirmed (C) Recovered (R) Deaths (D)



Less data: only ~450 samples in total

Concatenate 50 US states x 3 compartments ⟶ 2D array

$$T(\underbrace{d_{-14}, d_{-13}, \ldots, d_{-1}}_{u_0(x)}) = (\underbrace{d_0, d_1, \ldots, d_6}_{u(\tau,x)})$$

$u_0(x)$ ⟶ T ⟶ $u(\tau,x)$

# COVID19 forecasting benchmarks

| Networks | MAE | | | Relative L2 error | Net. vs FC |
|---|---|---|---|---|---|
| | C | R | D | | |
| Padé Exp | **1219 ± 130** | **1752 ± 666** | **211 ± 31** | **0.0155 ± 0.0034** | 82.14% (+652K) |
| MWT Leg | 3554 ± 1157 | 2928 ± 1338 | 284 ± 209 | 0.0245 ± 0.0043 | 62.0% (+18M) |
| FNO 3D | 4213 ± 391 | 3391 ± 1233 | 592 ± 157 | 0.0301 ± 0.0045 | 54.0% (+1.02M) |
| LNO 3D | 28502 ± 12698 | 6586 ± 3442 | 1465 ± 965 | 0.1056 ± 0.0394 | -105.0% (+238K) |
| Neural ODE | 4339 ± 1174 | 3443 ± 1408 | 443 ± 192 | 0.0310 ± 0.0069 | 53.8% (+172K) |
| Seq2Seq | 2798 ± 456 | 3317 ± 1690 | 346 ± 83 | 0.0273 ± 0.0058 | 63.7%(+1.8M) |
| Transformer | 7087 ± 972 | 6613 ± 2853 | 1722 ± 320 | 0.0501 ± 0.0094 | 13.4% (+15.2K) |
| FC | 10305 ± 2818 | 5885 ± 1609 | 1634 ± 686 | 0.0609 ± 0.0111 | (37.2K) |

Compact neural operator with best performance

# COVID19 forecasting sample