Operator Regression

Abstract problem

Reference "Real Analysis" by Folland

First some informal definitions

def Hilbert space – linear combinations of functions w/ an inner product

ex $L^2(\Omega) = \{ f : \int_\Omega f^2 \, dx < \infty \}$

w/ inner product $(f, g) = \int_\Omega f g \, dx$

ex $P^1(T) = \{\text{piecewise linear FEM}\}$

def linear functional A linear map from $V \to \mathbb{R}$ where V is HS

a functional is bnded if $|\tau(u)| < \infty$

ex point eval $\tau = \delta_{x_i} \circ u$, derivative $= \delta_{x_i} \circ \frac{d}{dx} u$

def dual space The set of bounded linear functionals on V is denoted V^* with the operator norm

$$\|\tau\| = \sup \{ \|Tx\|_V : \|x\|_V = 1 \}$$

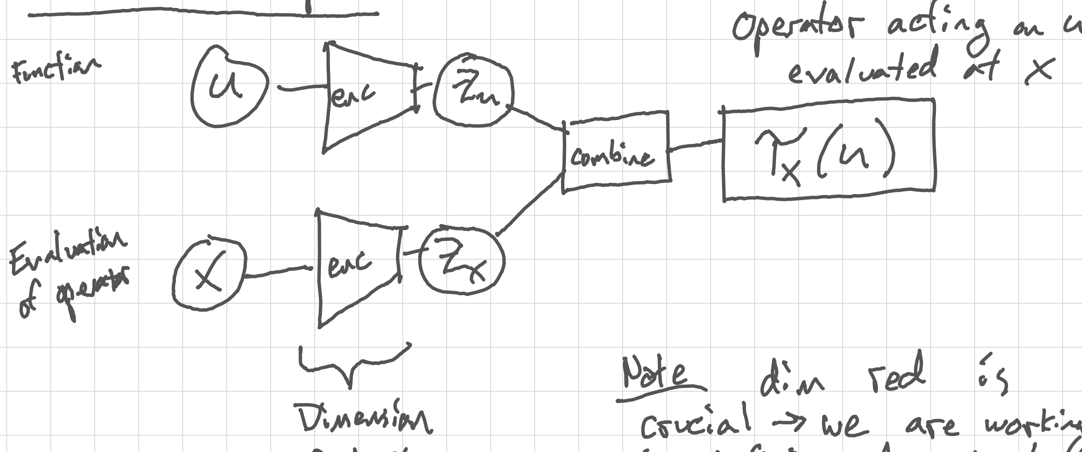
ex Consider $V = \mathbb{R}^N$, then $V^* = \mathbb{R}^N$

$\vec{u} \in V$, $w^T u = c \in \mathbb{R}$ $\Rightarrow \tau(u) = w^T u \in V^*$

In

Could take a whole course on functional analysis but we are using these to understand that operator regression amounts to developing latent representations of V, V^* and their relationship

General blueprint



Note dim red is crucial → we are working in infinite dimensional spaces

---

We will consider several architectures fitting this blueprint

Method 1

"Model Reduction & NNs for Parametric PDEs"
Bhattacharya et al 2021

ex $\quad -\nabla \cdot a(x) \nabla u(x) = f(x)$

(A) Solution operator : $L: f \to u$

(B) Parametric solution $L: a \to u_a$

Given input $\mathcal{X}$, output $\mathcal{Y}$, want to characterize the map $\Psi : \mathcal{X} \to \mathcal{Y}$

$$
\begin{tikzcd}
\mathcal{X} \arrow[r, "F_{\mathcal{X}}"] \arrow[d, "\Psi"'] & \mathbb{R}^{d_{\mathcal{X}}} \arrow[r, "G_{\mathcal{X}}"] \arrow[d, "\phi"'] & \mathcal{X} \arrow[d, "\Psi"] \\
\mathcal{Y} \arrow[r, "F_{\mathcal{Y}}"'] & \mathbb{R}^{d_{\mathcal{Y}}} \arrow[r, "G_{\mathcal{Y}}"'] & \mathcal{Y}
\end{tikzcd}
$$

- F, G are encoder decoder pairs
- ϕ is operator mapping in the latent space

3 types of maps

$$\left.\begin{aligned} G_{\mathcal{X}} \circ F_{\mathcal{X}} &= I_{\mathcal{X}} \\ G_{\mathcal{Y}} \circ F_{\mathcal{Y}} &= I_{\mathcal{Y}} \end{aligned}\right\} \text{Auto encoder error}$$

$$\left. G_{\mathcal{Y}} \circ \phi \circ F_{\mathcal{X}} = \Psi \right\} \text{Operator error}$$

<u>Encoder / Decoder</u>

- Calculate reduced basis w/ PCA

$$P_{\mathcal{X}} = PCA(x_1, \dots x_N)$$

$$F_{\mathcal{X}} = \Pi x := (P^T P)^{-1} P x \quad \in \mathbb{R}^{d_{\mathcal{X}}}$$

$$G_{\mathcal{X}} = P z$$

- Given reduced representations

$$z_x = \Pi_X \, x$$

$$z_y = \Pi_y \, y$$

Use DNN to approximate

$$z_y \approx \phi(z_x) := \mathrm{DNN}(z_x ; \theta)$$

Method 2 Projection Based ROM

(Benner 2015)

$$\begin{tikzcd}
X \arrow[r, "F_X"] \arrow[d, "\Psi"'] & \mathbb{R}^{d_x} \arrow[r, "G_X"] \arrow[d, "\phi"'] & X \arrow[d, "\Psi"] \\
Y \arrow[r, "F_Y"'] & \mathbb{R}^{d_y} \arrow[r, "G_Y"'] & Y
\end{tikzcd}$$

Need a PDE.

- Identical setting for encoder/decoder
- To identify ϕ, Revisit the Galerkin approximation using the PCA basis instead

→ Original FEM $\quad u(x) = P(x)^T \hat{u}$

→ ROM $\quad \Pi u(x) = (V P(x))^T \hat{u}$

→ Galerkin ROM - test, trial $\in \mathrm{span}(V P(x))$

$$V^T S V \hat{u} = V^T b \rightarrow \phi := (V^T S V)^{-1} V^T b$$

Method 1	Method 2
No governing eqn needed	Intrusive
Limited to DNN accuracy	Convergent

Both use a linear encoding via PCA

Method 3 "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional auto encoders" Lee, Carlberg

$$Z_x = NN_e(X)$$

$$X = NN_d(Z_x)$$

Solve

$$\nabla^2 u(NN_d(Z)) = f$$

Tricks → Non-linear
→ Loss of symmetry needs "special" finite elements (Petrov Galerkin)

Method 4 Fourier Embeddings

(Several groups concurrently:
1. Stuart, Anandkumar FNO
2. Patel, Trask MOR-Physics
3. Dongbin Xiu

First some background on Fourier transform

def Fourier series

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{2\pi i n x}, \quad x \in [-\tfrac{1}{2}, \tfrac{1}{2}]$$

or in light of Eulers formula

$$e^{ix} = \cos x + i \sin x$$

$$f(x) = \sum_{n=-\infty}^{\infty} c_n \left(\cos 2\pi n x + i \sin 2\pi n x \right)$$

We can move between function and its fourier series expansion via the Fourier transform

<u>def</u> fourier transform

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) \exp(-2\pi i \xi x) \, dx$$

<u>def</u> inverse fourier transform

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) \exp(2\pi i \xi x) \, d\xi$$

Fourier representation has many useful properties

① $$\widehat{f'(x)} = \int_{-\infty}^{\infty} f'(x) \exp(-2\pi i \xi x) \, dx$$

$$= - \int_{-\infty}^{\infty} f(x) \frac{d}{dx} \exp(-2\pi i \xi x) \, dx$$

$$= \underbrace{-2\pi i \xi}_{\substack{\text{symbol of} \\ \text{differential operator}}} \widehat{f(x)}$$

α in general

$$\widehat{\frac{d^\alpha}{dx^\alpha} f} = (2\pi i \xi)^\alpha \hat{f}$$

$\Rightarrow$ Arbitrary differential operators may be represented as multiplication in Fourier space

② <u>def</u> convolution

<u>Note</u> continuous version compare to CNN

$$f \star g = \int f(x)\, g(x-y)\, dy$$

$$\widehat{f \star g} = \iint f(x)\, g(x-y)\, e^{-2\pi i x \xi}\, dy\, dx$$

$$= \iint f(x)\, g(x-y) \exp\Big(-2\pi i \xi \big(x \underbrace{- y + y}_{\text{add zero}}\big)\Big)\, dy\, dx$$

$$= \int f(x) e^{-2\pi i \xi x} \left(\int g(x-y)\, e^{-2\pi i \xi (x-y)}\, dy \right) dx$$

$$= \hat{f}\ \hat{g}$$

$\Rightarrow$ Convolutions (nasty integrals) turn into products

③ Fundamental Solutions of linear PDE

- Consider heat eqn

$$\partial_t u + \partial_{xx} u = 0$$

- Take FFT

$$\partial_t \hat{u} - 4\pi^2 \xi^2 \hat{u} = 0$$

$$\Rightarrow \quad \hat{u} = \exp(-4\pi^2 \xi t)\, \hat{u}(t=0)$$

Define $\hat{F} = \exp(-4\pi^2 \xi t)$

Then $u(t) = F \star u(t=0)$

And we can represent the solution operator as a convolution with a "fundamental solution"

Architecture

$$\begin{tikzcd}
\mathcal{X} \arrow[r, "FFT"] \arrow[d, "\psi"'] & \mathbb{R}^{d_x} \arrow[r, "IFFT"] \arrow[d, "\phi"', "DNN"] & \mathcal{X} \arrow[d, "\psi"] \\
\mathcal{Y} \arrow[r, "FFT"'] & \mathbb{R}^{d_y} \arrow[r, "IFFT"'] & \mathcal{Y}
\end{tikzcd}$$

<u>Idea</u> Hardcode FFT encoder
- good for differential operators
- bad for nonlinearities
- bad for non-periodic boxes

Method 5   DeepOnet   Lu, Jin, Karniadakis 2020

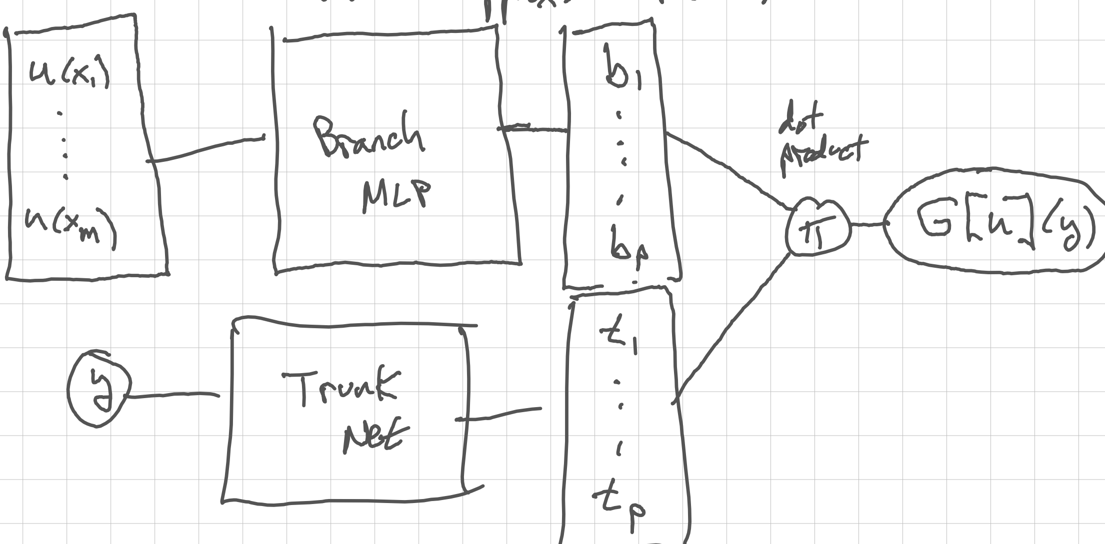
IDEA learn it all in one shot.

Thm Universal Approximation (Chen & Chen 95)

Given continuous σ, continuous operator G
there exists parameters such that for all $\varepsilon > 0$

$$\left| G[u](y) - \sum_{k=1}^{p} \sum_{i=1}^{n} \underbrace{c_i^k \sigma\left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k\right)}_{\text{branch}} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{\text{trunk}} \right| < \varepsilon$$

Remark Be careful putting too much faith in universal approx. results



Lecture Notes 3/26

HW notes

- extension to next week
- question 3 → do not use packages, alter the code from class
- question 2 — two components

① Modify code to generate data

Recall for a second-order ODE

To solve: $\ddot{z} = -\alpha z - \beta \dot{z}$

Write as system of first order ODES

$$\dot{z}_1 = z_2$$

$$\dot{z}_2 = -\alpha z_1 - \beta z_2$$

$$\frac{d}{dt}\vec{z} = Az \quad , \quad A = \begin{pmatrix} 0 & 1 \\ -\alpha & -\beta \end{pmatrix} \quad z = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$$

Solve eigenvalue problem

$$\det(A - I\lambda) = 0$$

$$-\lambda(-\beta - \lambda) + \alpha = 0$$

$$-\lambda(-\beta-\lambda) + \alpha = 0$$

$$\lambda^2 + \beta\lambda + \alpha = 0$$

$$\lambda = \frac{-\beta \pm \sqrt{\beta^2 - 4\alpha}}{2}$$

$$z(t) = c_1 e^{i\lambda_1 t} + c_2 e^{i\lambda_2 t}$$

<u>OR</u> Guess and check

- $z(t) = z_0 e^{-\lambda t} \cos \theta t$
- Plug into residual, solve ~~for~~ for λ, θ

$$\dot{z} = z_0 \left(-\lambda e^{-\lambda t} \cos\theta t - \theta e^{-\lambda t} \sin\theta t\right)$$

$$\ddot{z} = \quad \cdots$$

$0 = z - \alpha\dot{z} - \beta\ddot{z}$, solve for λ, θ

<u>OR</u> Use ODE int

Second ingredient - alter prior to account for dissipation

$$\frac{d}{dt}\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = Az \rightarrow \frac{z_1^{n+1} - z_1^{n}}{h} = z_2^{n+1}$$

implicit Euler $\rightarrow$ $\dfrac{z_2^{n+1} - z_2^{n}}{h} = -\alpha z_1^{n+1} - \beta z_2^{n+1}$

Work out probability

$$p(z_1^{n+1}, z_2^{n+1} \mid z_1^{n}, z_2^{n}) = N(\mu, \Sigma)$$

work these out

Note that we now need a 2D latent space (z_1, z_2 not just z)

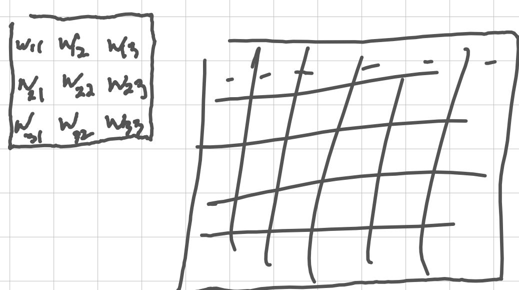
Notes for the exam

- Bring a single sheet w/ whatever you want on both sides
- Focus comfortable w/ simple probability definitions & derivations

---

Next section - focusing on unstructured data

Recall translation invariance of CNNs



In general data may vary in shape, size

ex Mesh geometries, paired w/ maximum stress, gives supervised problem. Given a new mesh, predict stress.

How to generalize across different meshes?

Physics 4 AI vs AI 4 Physics

Mathematical Machinery - Graphs

A graph $G(N, E)$ is a collection of nodes $N \in \mathbb{R}^{N_n}$ and edges $E \in \mathbb{R}^{N_e}$

Edges may be directed $\quad (i) \rightarrow (j)$

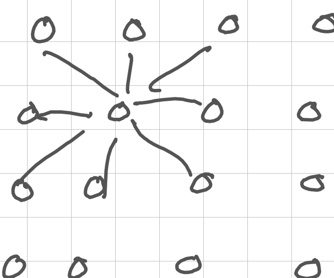
or undirected $\quad (i) - (j)$

Can associate labels w/ nodes or edges

Examples of graphs

images / CNNs

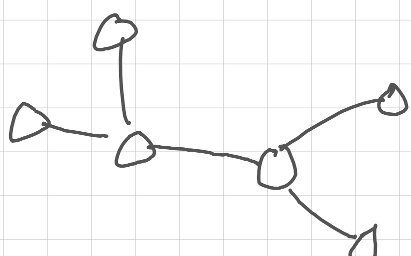
pixels = nodes
connect neighboring pixels



Molecules

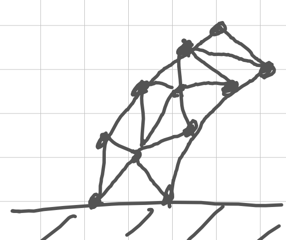
node = atom

edge = bond



PDEs discretized w/ FEM

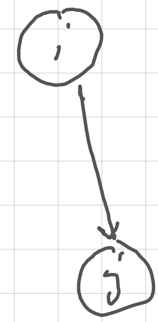
node - vertex
edge - adjacent vertices



Social Networks etc

Graph fundamentals

<u>def</u> Graph Gradient



$$Gu_{ij} = u_j - u_i$$

- Oriented difference per edge
- Anti-symmetric

$$Gu_{ij} = -Gu_{ji}$$

<u>def</u> Graph divergence

$$Du_i = \sum_{j \sim i} u_{ij}$$

nodes j sharing an edge with i

Note that G and D are matrices adjoint with respect to l_2 inner product

def $\langle x, y \rangle = \sum_i x_i y_i$

inner product on edges

$$\langle v, Gu \rangle = \sum_{e_{ij}} v_{ij} (u_j - u_i)$$

$$= \sum_{i,j} v_{ij} u_j - \sum_{i,j} v_{ij} u_i$$

$$\langle v, Gu \rangle = \sum_{e_{ij}} v_{ij} (u_j - u_i)$$

$$= \sum_{j \sim i} v_{ij} u_j - \sum_{i, j \sim i} v_{ij} u_i$$

$$= \sum_{j, i} v_{ji} u_i - \quad ''$$

$$= \sum_i u_i \left(\sum_{j \sim i} v_{ji} - v_{ij} \right)$$

$$= \sum_i u_i \left(-2 \sum_{j \sim i} v_{ij} \right)$$

$$= \langle u, Dv \rangle$$ inner product on edges

i.e $G^T = D$

If we instead pick a matrix inner product, we can define a

<u>weighted graph div</u> $D_w v = \sum_{j \sim i} w_{ij} v_{ij}$ positive weights

Defining $\langle x, y \rangle_W = x^T W y$

$$\langle v, Gu \rangle_W = \langle D_w v, u \rangle$$

Just like vector calculus, we can build a weigthed graph Laplacian

$$\Delta_g f = D_w G f$$

$$\Delta_g f_i = \sum_{j\sim i} w_{ij} (f_j - f_i)$$

GL are symmetric positive definite

pf Recall A is SPD if $x^T A x > 0$ for any x

Defining $D_w = D W$

$W = \mathrm{diag}(w_{ij})$

Then $\Delta_g = D W G$

$= G^T W G$

$w_{ij} > 0 \Rightarrow W = \sqrt{W}\sqrt{W}$

$\Delta_g = (\sqrt{W} G)^T (\sqrt{W} G)$

$\forall x,$

$x^T \Delta_g x = (\sqrt{W} G x)^T (\sqrt{W} G x)$

Let $y = \sqrt{W} G x \Rightarrow x^T \Delta_g x = y^T y$

> 0 ∎

Graphs have a deep theory as well as ties to physical systems

ex Resistive network

Ohms law $\rightarrow$ $J_{ij} = R_{ij}(V_j - V_i)$

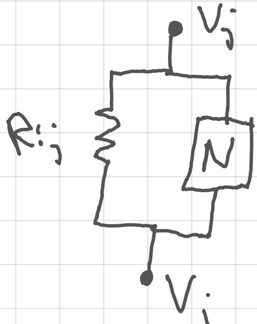
Kirchoffs law $\rightarrow$ $\sum_{j\sim i} J_{ij} = f_i$

Is equivalent to solving

$$\Delta_g V = G^T W G V = f$$

where $W = \mathrm{diag}(R_{ij})$

In general we can consider families of circuit models where each edge consists of a resistor in parallel w/ a non-linearity



$$J_{ij} = R_{ij}(V_j - V_i) + F(V_i, V_j)$$

ex diode
inductor
etc

For these models, the governing equation looks like

$$\text{Kirchoff} \rightarrow \sum_{j \sim i} J_{ij} = f_i$$

$$J_{ij} = R_{ij} G V_{ij} + N(V_i, V_j)$$

or ★ $$\Delta_g V + G^T N(V) = f$$

Many systems admit these types of "circuit analogy" models

- lumped capacitance heat transfer
- hydraulic circuit
- spring/damper mechanics

We will start on graphs by fitting one of these circuit models to data

Thm If N is lipschitz (Trask 22)

$$|N(v_1) - N(v_2)| \leq L_N \|v_1 - v_2\|$$

with L_N smaller than the smallest eigenvalue of Δ_g, then ★ has a unique soln.

We won't worry about the math,
just how to fit model to data

Define $\mathcal{L}[u;\theta,w] = G^T \underset{\text{resistance}}{W} G u + G^T \underset{\text{DNN}}{N[u;\theta]} - \underset{\text{source/sink}}{f}$

Solve

$$\min_{\theta, W} \sum_d \| u^d - u^d_{data} \|^2$$

$$\text{s.t. } \mathcal{L}[u^d;\theta,w] = 0$$

Add Lagrange Multiplier

$$\underset{\text{loss}}{L} = (u - u_{data})^T (u - u_{data}) + \lambda^T \mathcal{L}[u;\theta,w]$$

From KKT

① $\partial_\lambda L = 0 \Rightarrow \boxed{\mathcal{L}[u;\theta,w] = 0}$

"Forward Problem" - solve w/ current guess for params

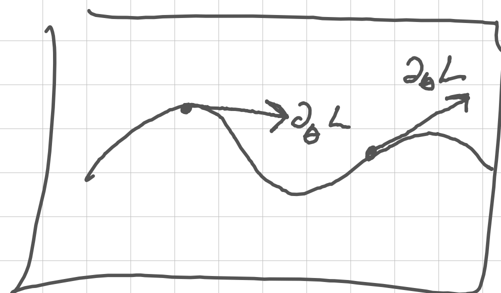
② $\partial_u L = 0 \Rightarrow \boxed{-2(u - u_{data}) = \left(\frac{\partial \mathcal{L}}{\partial u}\right)^T \lambda}$

"Adjoint Problem" - solve for λ

(3) Model update

Hit remaining terms w/ an Adam update

Draw cartoon showing constraint manifold



Graph Attention networks (Velickovic 2017)

Back to ML from physics world

<u>Goal</u> Given supervised data

$$D = \{(G_i, y_i)\}_{i=1}^{N}$$

where G_i is a graph w/ varying size

y_i is set of nodal labels

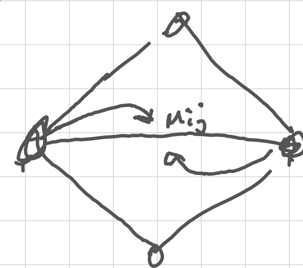
A Graph Attention layer (GAT) consists of 3 steps

1 Message Passing

$$m_{ij} = N(x_i^n, x_j^n \,|\, \theta)$$

message on edge i,j

labels on nodes of graph



2 Aggregate

$$x_i^{n+1} = \sigma\left(\sum_{j\sim i} \alpha_{ij}\, m_{ij}\right)$$

attention weight

3 Attention

preattention mechanism

$$\alpha_{ij}(m) = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

$$e_{ij} = a(h_i, h_j)$$

Challenges w/ GATs

"Oversquashing / Oversmoothing"

Cant go very deep w/o output collapsing to a single vector independent of input

Tricks

Graph Rewiring - add connections

Physics - inspired dynamics

GRAND - Graph Neural Diffusion (Chamberlain 21)

$$x^{n+1} = \sigma\left(\sum_j a(x_i, x_j) \left(x_j^n - x_i^n \right) \right)$$

non-linear attention serves as inner product