A Machine Learning Framework for High-Dimensional Optimal Transport

Deep Learning and Medical Applications, IPAM January 2020

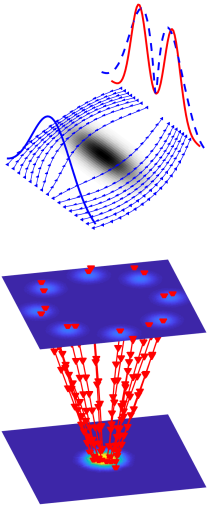
Lars Ruthotto[1], Stanley Osher[2], Wuchen Li[2], Levon Nurbekyan[2], Samy Wu Fung[2]

[1]Departments of Mathematics and Computer Science, Emory University
[2]Department of Mathematics, University of California Los Angeles

lruthotto@emory.edu
@lruthotto

Agenda: Machine Learning for Optimal Transport

- **Background: Low-Dimensional OT**
  - (Mass-Preserving) Image Registration
- **Motivation: High-Dimensional OT**
  - Generative modeling
  - Challenge: curse of dimensionality
- **Three Options for OT as Mean Field Game**
  - macroscopic view: variational problem
  - microscopic view: coupled optimization problem
  - Hamilton-Jacobi-Bellman (HJB) equation
- **Our Framework: Variational approach with**
  - Lagrangian solver for continuity equation
  - Neural Network Parameterization
  - Penalize violations of HJB
- **Numerical Experiments**



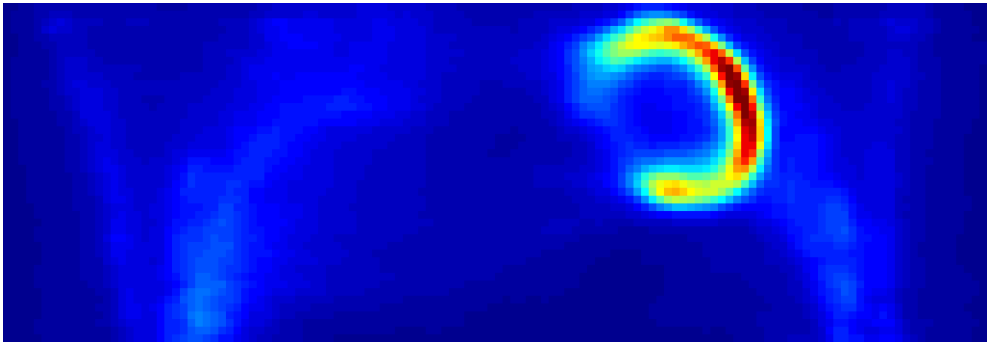
LR, S Osher, W Li, L Nurbekyan, S Wu Fung
A Machine Learning Framework for Solving High-Dimensional Mean Field Game and Mean Field Control Problems, arXiv, 2019.

A Mang, LR.
A Lagrangian Gauss–Newton–Krylov Solver for Mass- and Intensity-Preserving Diffeomorphic Image Registration. *SIAM SISC*,39(5), B5860-B5885 2017.

Background: Mass-Preserving Image Registration

Motion Correction of Cardiac PET

- Fabian Gigengack, Martin Burger, European Institute of Molecular Imaging, University of Münster
- Otmar Schober, Department of Nuclear Medicine, University of Münster



Thoracic FDG^{18} PET, acquisition time $\approx$ 20 min

F. Gigengack, LR, M. Burger, C.H. Wolters, X. Jiang, K.P. Schäfers
Motion correction in dual gated cardiac PET using mass-preserving image registration.
IEEE Trans. Med. Imag.; 31(3):698–712, 2012.

LR, F. Gigengack, M. Burger, C.H. Wolters, X. Jiang, K.P. Schäfers, J. Modersitzki
A Simplified Pipeline for Motion Correction in Dual Gated Cardiac PET.
Bildverarbeitung f.d. Medizin: 51–56, 2012.

Optimal Control Formulation of Image Registration

reference $\mathcal{R}$

template $\mathcal{T}$

Optimal Control Problem

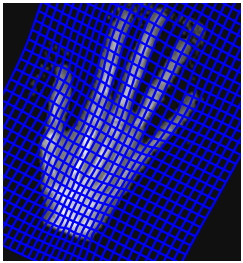
Given a template image $\mathcal{T}$ and a reference image $\mathcal{R}$ find a smooth velocity field v, such that the final state $u(\cdot, 1)$ is similar to $\mathcal{R}$, i.e., solve

$$\text{minimize}_{v,u} \quad \text{distance}[u(\cdot,1),\mathcal{R}] + \text{regularizer}[v]$$

$$\text{subject to} \quad u(\cdot,0) = \mathcal{T} \text{ and } \begin{cases} \partial_t u + v \cdot \nabla u = 0, & \text{intensity preservation} \\ \partial_t u + \nabla \cdot (uv) = 0, & \text{mass-preservation.} \end{cases}$$

Optimal Control Formulation of Image Registration

reference $\mathcal{R}$



template $\mathcal{T}$ +$y(v)$

$u(\cdot, 1)$

Optimal Control Problem

Given a template image $\mathcal{T}$ and a reference image $\mathcal{R}$ find a smooth velocity field v, such that the final state $u(\cdot, 1)$ is similar to $\mathcal{R}$, i.e., solve

$$\text{minimize}_{v,u} \quad \text{distance}[u(\cdot,1), \mathcal{R}] + \text{regularizer}[v]$$

$$\text{subject to} \quad u(\cdot,0) = \mathcal{T} \text{ and } \begin{cases} \partial_t u + v \cdot \nabla u = 0, & \text{intensity preservation} \\ \partial_t u + \nabla \cdot (uv) = 0, & \text{mass-preservation.} \end{cases}$$

Numerical Methods for Forward Problem

Notation: $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N$ states, $\mathbf{y}_0$ regular grid, $\Delta t = 1/N$

Eulerian methods approximate states on fixed grid for all times

- Example (Lax-Friedrichs): for all cells, $i = 1, \ldots, n$ do

$$(\mathbf{u}_{k+1})_i = \frac{1}{2}\left((\mathbf{u}_k)_{i+1} + (\mathbf{u}_k)_{i-1}\right) + \frac{\Delta t}{2\Delta x}\left((\mathbf{v}_k \odot \mathbf{u}_k)_{i+1} - (\mathbf{v}_k \odot \mathbf{u}_k)_{i-1}\right).$$

- Others: Upwind, Lax-Wendroff, ENO (non-differentiable) / WENO (nonlinear in initial condition),...

Numerical Methods for Forward Problem

Notation: $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N$ states, $\mathbf{y}_0$ regular grid, $\Delta t = 1/N$

Eulerian methods approximate states on fixed grid for all times

- Example (Lax-Friedrichs): for all cells, $i = 1, \ldots, n$ do

$$(\mathbf{u}_{k+1})_i = \frac{1}{2}\left((\mathbf{u}_k)_{i+1} + (\mathbf{u}_k)_{i-1}\right) + \frac{\Delta t}{2\Delta x}\left((\mathbf{v}_k \odot \mathbf{u}_k)_{i+1} - (\mathbf{v}_k \odot \mathbf{u}_k)_{i-1}\right).$$

- Others: Upwind, Lax-Wendroff, ENO (non-differentiable) / WENO (nonlinear in initial condition),...

Lagrangian Methods follow characteristics. Let $\mathbf{I}$ be an interpolation matrix (sparse, $\mathbf{I} \geq 0$, row-sums of 1)

$$\mathbf{u}_N = \mathbf{I}(\mathbf{y}(v, \mathbf{y}_1, 1, 0))\,\mathbf{u}_0 \quad \text{or} \quad \mathbf{u}_N = \mathbf{I}(\mathbf{y}(v, \mathbf{y}_0, 0, 1))^\top\,\mathbf{u}_0$$

Numerical Methods for Forward Problem

Notation: $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N$ states, $\mathbf{y}_0$ regular grid, $\Delta t = 1/N$

Eulerian methods approximate states on fixed grid for all times

- Example (Lax-Friedrichs): for all cells, $i = 1, \ldots, n$ do

$$(\mathbf{u}_{k+1})_i = \frac{1}{2}\left((\mathbf{u}_k)_{i+1} + (\mathbf{u}_k)_{i-1}\right) + \frac{\Delta t}{2\Delta x}\left((\mathbf{v}_k \odot \mathbf{u}_k)_{i+1} - (\mathbf{v}_k \odot \mathbf{u}_k)_{i-1}\right).$$

- Others: Upwind, Lax-Wendroff, ENO (non-differentiable) / WENO (nonlinear in initial condition),...

Lagrangian Methods follow characteristics. Let $\mathbf{I}$ be an interpolation matrix (sparse, $\mathbf{I} \geq 0$, row-sums of 1)

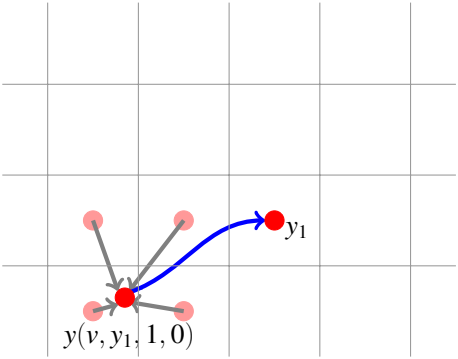
$$\mathbf{u}_N = \mathbf{I}(\mathbf{y}(v, \mathbf{y}_1, 1, 0))\, \mathbf{u}_0 \quad \text{or} \quad \mathbf{u}_N = \mathbf{I}(\mathbf{y}(v, \mathbf{y}_0, 0, 1))^\top\, \mathbf{u}_0$$

Semi-Lagrangian follow characteristics for short time, interpolate/push, repeat

$$\mathbf{u}_N = \mathbf{I}(\mathbf{y}(v, \mathbf{y}_1, \Delta t, 0))^N \mathbf{u}_0 \quad \text{or} \quad \mathbf{u}_N = (\mathbf{I}(\mathbf{y}(v, \mathbf{y}_0, 0, \Delta t))^\top)^N \mathbf{u}_0.$$

Illustration: Lagrangian Methods

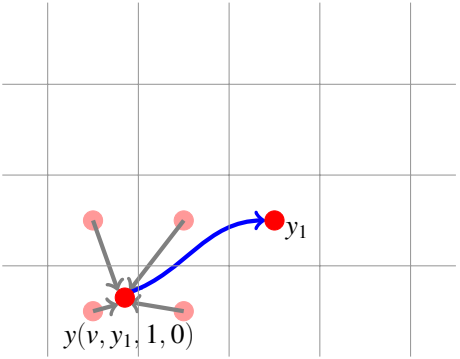
Let $\mathbf{I}(\cdot) \in \mathbb{R}^{m \times n}$ be an interpolation matrix ($\mathbf{I} \geq 0$, sparse, rows sum to one), $\mathbf{y}_0$ be fixed(!) grid point, $v : \Omega \times [0,1] \to \mathbb{R}^d$ velocity



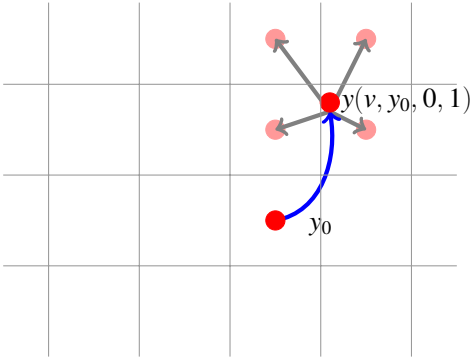
advection, $u(\mathbf{y}_1, 1) = \mathbf{I}(y(v, \mathbf{y}_1, 1, 0))u_0$

Illustration: Lagrangian Methods

Let $\mathbf{I}(\cdot) \in \mathbb{R}^{m \times n}$ be an interpolation matrix ($\mathbf{I} \geq 0$, sparse, rows sum to one), $\mathbf{y}_0$ be fixed(!) grid point, $v : \Omega \times [0,1] \to \mathbb{R}^d$ velocity



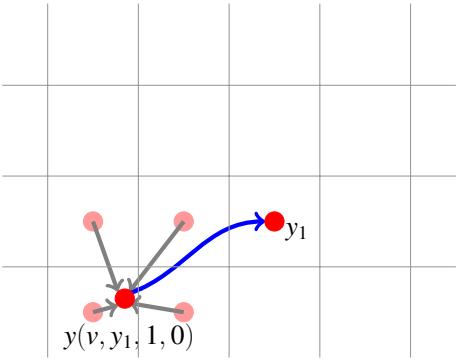
advection, $u(\mathbf{y}_1, 1) = \mathbf{I}(y(v, \mathbf{y}_1, 1, 0))u_0$



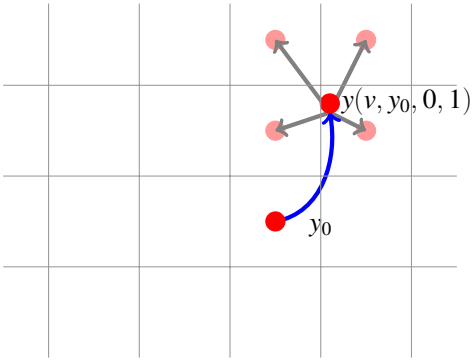
continuity, $u(\mathbf{y}_0, 1) = \mathbf{I}(y(v, \mathbf{y}_0, 0, 1))^\top u_0$

Illustration: Lagrangian Methods

Let $\mathbf{I}(\cdot) \in \mathbb{R}^{m \times n}$ be an interpolation matrix ($\mathbf{I} \geq 0$, sparse, rows sum to one), $\mathbf{y}_0$ be fixed(!) grid point, $v : \Omega \times [0,1] \to \mathbb{R}^d$ velocity



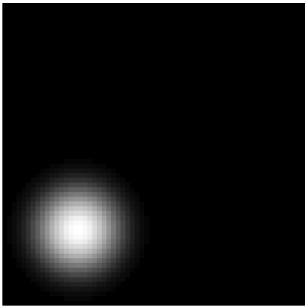
advection, $u(\mathbf{y}_1, 1) = \mathbf{I}(y(v, \mathbf{y}_1, 1, 0))u_0$



continuity, $u(\mathbf{y}_0, 1) = \mathbf{I}(y(v, \mathbf{y}_0, 0, 1))^\top u_0$

J. Fohring, E. Haber, LR.
Geophysical Imaging of Fluid Flow in Porous Media.
SIAM SISC, 36 (5): S218–S236, Oct. 2014.

Particle Methods: A 2D Example



ρ_0

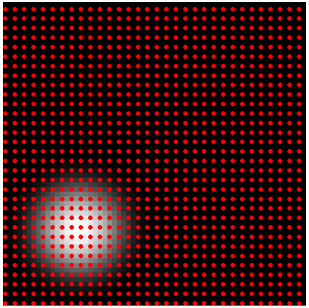
Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2$$

$$\text{with} \quad \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5).$$

$$\vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5)$$

Particle Methods: A 2D Example



ρ_0 + cell-centred grid

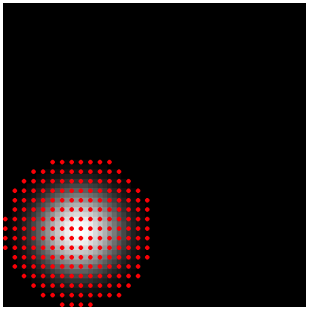
Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2$$

$$\text{with} \quad \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5).$$

$$\vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5)$$

Particle Methods: A 2D Example



ρ_0 + particles

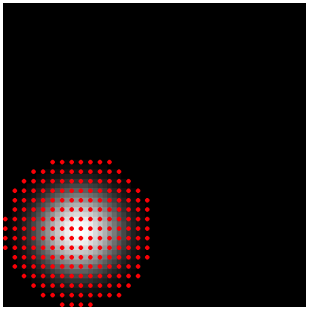
Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2$$

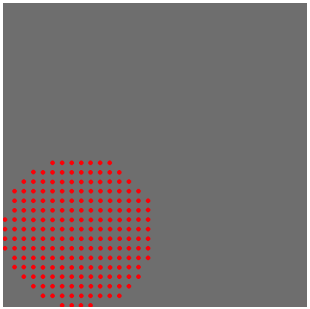
$$\text{with} \quad \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5).$$

$$\vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5)$$

Particle Methods: A 2D Example



ρ_0 + particles



particle simulation

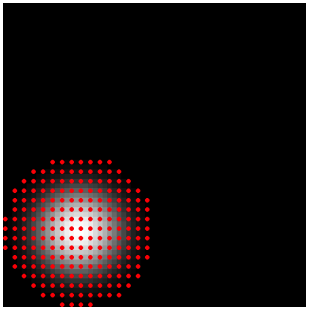
Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2$$

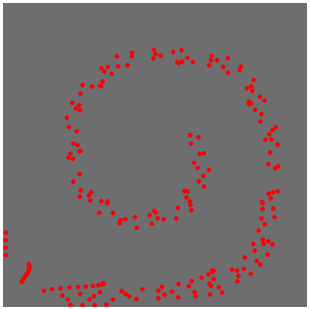
$$\text{with} \quad \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5).$$

$$\vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5)$$

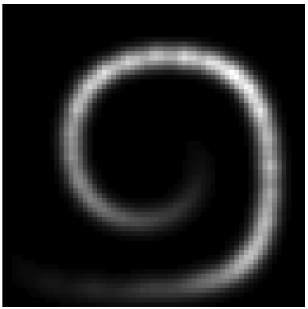
Particle Methods: A 2D Example



ρ_0 + particles



particles, $t = 2.5$

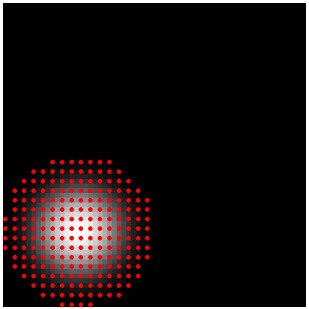


$\rho(\cdot, 2.5)$

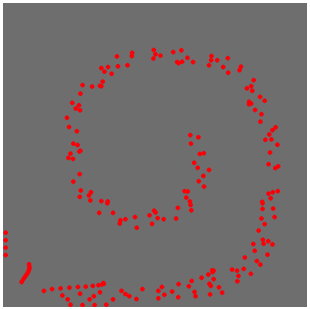
Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\begin{aligned} & \rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2 \\ \text{with} \quad & \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5). \\ & \vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5) \end{aligned}$$

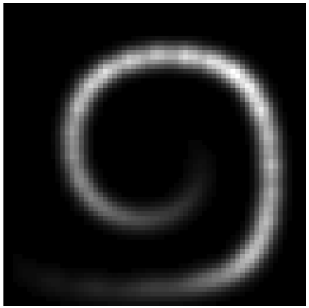
Particle Methods: A 2D Example



ρ_0 + particles



particle simulation

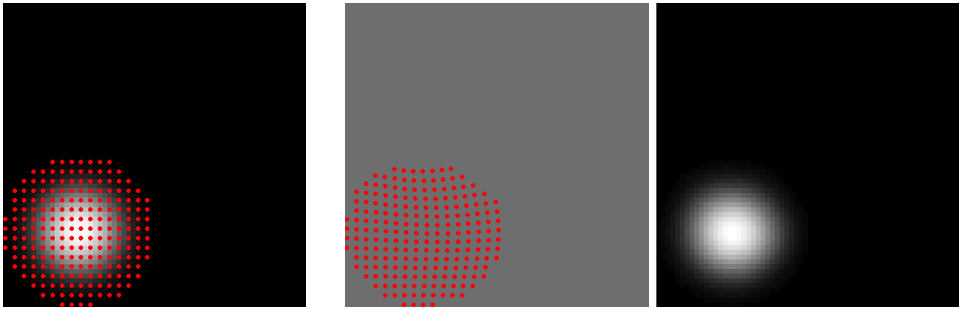


$\rho(\cdot, 2.5)$

Example: Advection in incompressible flow, Chertock and Kurganov (2006)

$$\begin{aligned} & \rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2 \\ \text{with} \quad & \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5). \\ & \vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5) \end{aligned}$$

Particle Methods: A 2D Example



ρ_0 + particles | particles, $t = 5$ | $\rho(\cdot, 5)$

Example: Advection in incompressible flow, Chertock and Kurganov (2006)

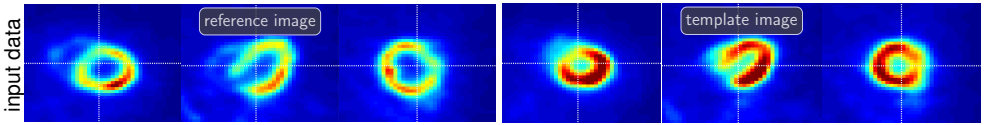
$$\rho_t + \nabla \cdot (\vec{u}\rho) = 0, \quad \text{on } \Omega = [0,1]^2$$

$$\text{with} \quad \vec{u}_1(x,t) = (\sin(\pi x_1))^2 \cdot \sin(2\pi x_2) \cdot \cos(\pi t/5).$$

$$\vec{u}_2(x,t) = (\sin(\pi x_2))^2 \cdot \sin(2\pi x_1) \cdot \cos(\pi t/5)$$

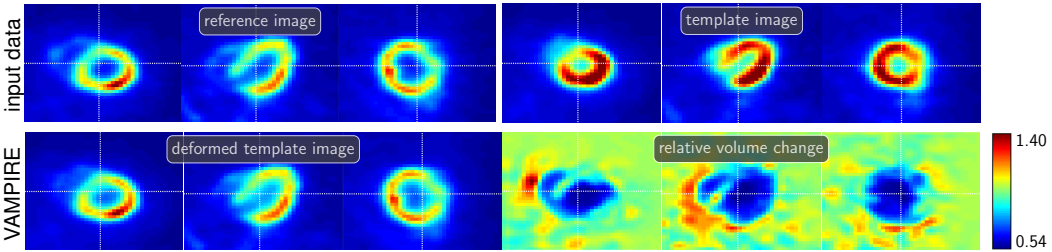
Example: 3D Mass-preserving Registration

levels: 10^3, 20^3, 40^3
VAMPIRE: $\alpha_{\text{length}} = 1000$, $\alpha_{\text{vol}} = 10$, Jacobi-PCG (matrix-free)
MP-LDDMM: non-stationary diffusion, $n_t = 1$, $\alpha = 100$, Spectral-PCG, RK4 $N = 2$



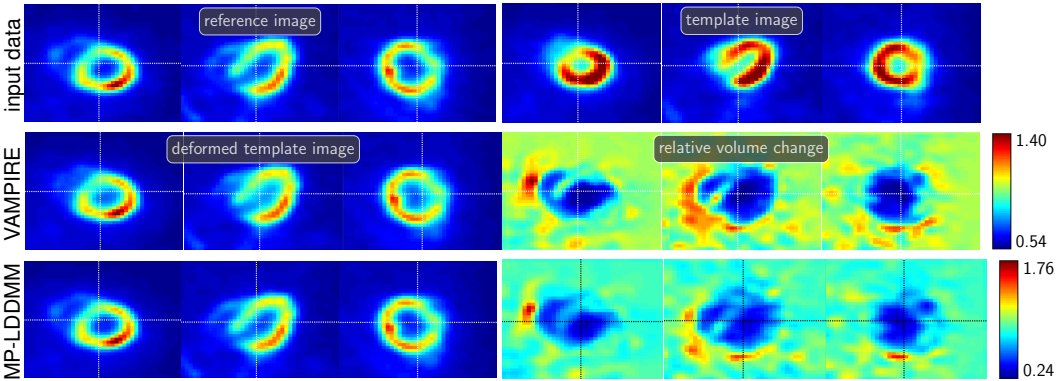
Example: 3D Mass-preserving Registration

levels: 10^3, 20^3, 40^3
VAMPIRE: $\alpha_{\text{length}} = 1000$, $\alpha_{\text{vol}} = 10$, Jacobi-PCG (matrix-free)
MP-LDDMM: non-stationary diffusion, $n_t = 1$, $\alpha = 100$, Spectral-PCG, RK4 $N = 2$



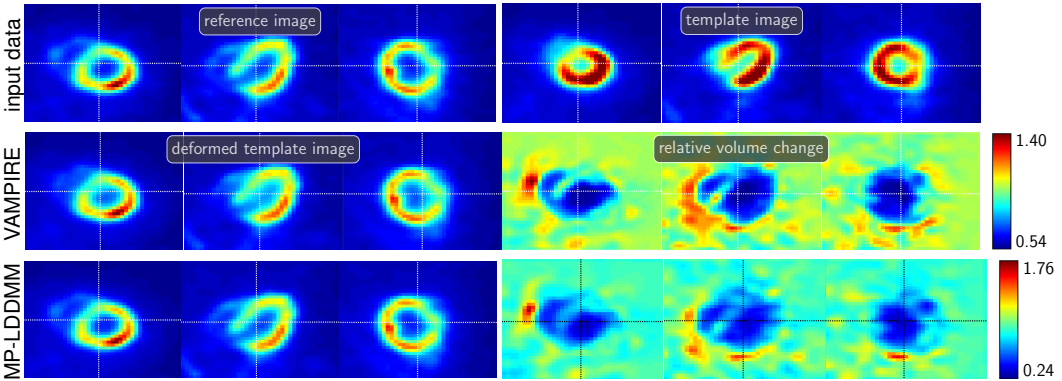
Example: 3D Mass-preserving Registration

levels: 10^3, 20^3, 40^3
VAMPIRE: $\alpha_{\text{length}} = 1000$, $\alpha_{\text{vol}} = 10$, Jacobi-PCG (matrix-free)
MP-LDDMM: non-stationary diffusion, $n_t = 1$, $\alpha = 100$, Spectral-PCG, RK4 $N = 2$



Example: 3D Mass-preserving Registration

levels: 10^3, 20^3, 40^3
VAMPIRE: $\alpha_{\text{length}} = 1000$, $\alpha_{\text{vol}} = 10$, Jacobi-PCG (matrix-free)
MP-LDDMM: non-stationary diffusion, $n_t = 1$, $\alpha = 100$, Spectral-PCG, RK4 $N = 2$



time-to-solution: VAMPIRE $\approx$ 74 sec, MP-LDDMM $\approx$ 36 sec

Motivation: High-Dimensional Optimal Transport

Generative Modeling with Normalizing Flows

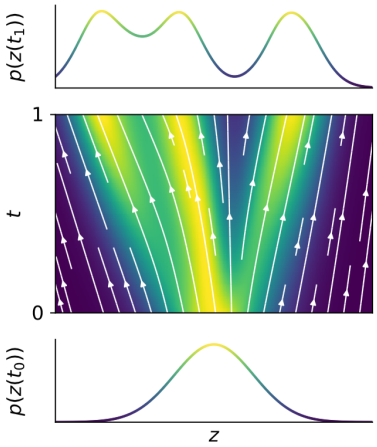


image: Grathwohl et al. 2019

D Rezende, S Mohamed
Variational Inference with Normalizing Flows. *SIAM SISC*,39(5), arXiv, 2015.

W Grathwohl et al.
FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. *arXiv*, 2018.

L Yang, GE Karniadakis
Potential Flow Generator with L_2 OT Regularity for Generative Models. *arXiv:1908.11462v1*, 2018.

L Zhang, Weinan E, L Wang
Monge-Ampère Flow for Generative Modeling, *arXiv:1809.10188v1*, 2018.

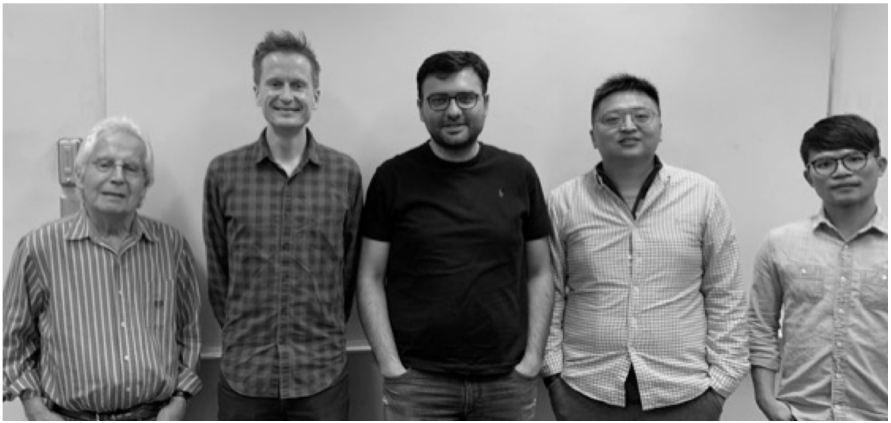
J Lin, K Lensink, E Haber
Fluid Flow Mass Transport for Generative Networks, *arXiv:1910.01694v2*, 2019.

Potential for medical imaging: data-driven priors and regularizers

Problem: curse of dimensionality!

Theory: OT as MFG $\rightsquigarrow$ HJB

Team and Acknowledgements

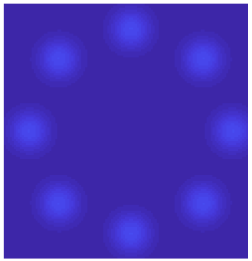


Emory Funding: DMS 1751636 US-Israel BSF 2018209

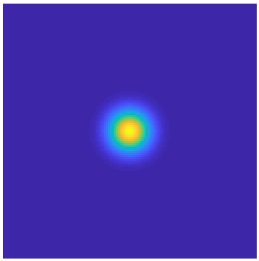
UCLA Funding: AFOSR MURI FA9550-18-1-0502, AFOSR FA9550-18-1-0167, ONR N00014-18-1-2527

Special thanks: Organizers and staff of IPAM Long Program MLP 2019.

Optimal Mass Transport



initial density, ρ_0



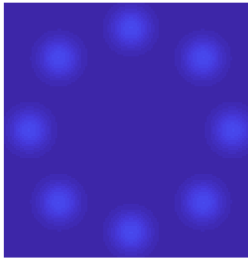
target density, ρ_1

Optimal Mass Transport

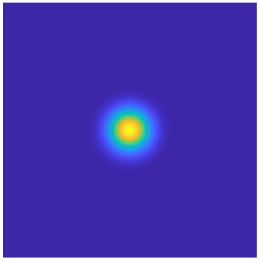
Given an initial density, ρ_0, and a target density, ρ_1, find the velocity field v that renders the push-forward of ρ_0 equal to ρ_1 and minimizes the transport costs, i.e., solve

$$\begin{aligned} \text{minimize}_{v,\rho} \quad & \int_0^1 \int_\Omega \|v(x,t)\|^2 \rho(x,t) dx dt \\ \text{subject to} \quad & \partial_t \rho + \nabla \cdot (\rho v) = 0, \quad \rho(\cdot, 0) = \rho_0(\cdot), \quad \rho(\cdot, 1) = \rho_1(\cdot) \end{aligned}$$

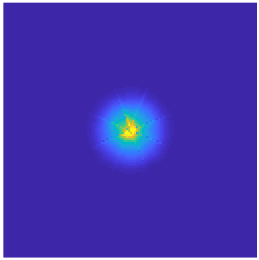
Optimal Mass Transport



initial density, ρ_0



target density, ρ_1



push-fwd of ρ_0

Optimal Mass Transport

Given an initial density, ρ_0, and a target density, ρ_1, find the velocity field v that renders the push-forward of ρ_0 equal to ρ_1 and minimizes the transport costs, i.e., solve

$$\begin{aligned} \text{minimize}_{v,\rho} \quad & \int_0^1 \int_\Omega \|v(x,t)\|^2 \rho(x,t) dx dt \\ \text{subject to} \quad & \partial_t \rho + \nabla \cdot (\rho v) = 0, \quad \rho(\cdot, 0) = \rho_0(\cdot), \quad \rho(\cdot, 1) = \rho_1(\cdot) \end{aligned}$$

Mean Field Games / Mean Field Control

MFGs model large populations of rational agents playing non-cooperative differential game.

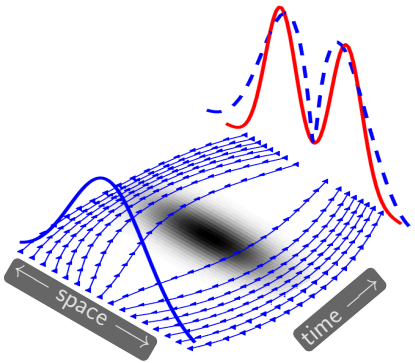
Mean Field Games / Mean Field Control

MFGs model large populations of rational agents playing non-cooperative differential game.

$$\text{minimize}_{v,\rho}\ \mathcal{J}_{\text{MFG}} := \int_0^1 \int_{\mathbb{R}^d} L\left(x, v(x,t)\right) \rho(x,t) dx dt + \int_0^1 \mathcal{F}(\rho(\cdot,t)) dt + \mathcal{G}(\rho(\cdot,1))$$

$$\text{subject to} \quad \partial_t \rho(x,t) + \nabla \cdot (\rho(x,t) v(x,t)) = 0, \quad \rho(x,0) = \rho_0(x),$$

- ρ_0: initial density (given)
- L: transport costs.
  Today: $L(x,v) = \frac{1}{2}\|v\|^2$
- $\mathcal{F}$: running costs.
  E.g., congestion, preference
  For OT: $\mathcal{F} \equiv 0$
- $\mathcal{G}$: terminal costs.
  E.g., cross entropy of ρ_1 and $\rho(\cdot,1)$



Agent-Based MFG Model

Let (x, t) be state of a single agent that aims at choosing v that minimizes

$$J_{x,t}(v) = \int_t^T \|v(s)\|^2 + F\left(z(s), \rho(z(s), s)\right) ds + G\left(z(T), \rho(z(T), T)\right),$$

where position changes as in

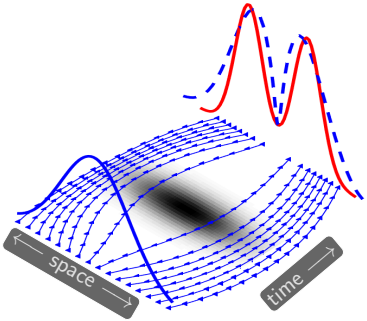
$$\partial_t z(t) = v\left(t\right), \; 0 \le t \le T, \quad z(0) = x.$$

- F: running costs (Ex: congestion, preference)
- G: terminal costs (Ex: cross entropy)
- Define value function

$$\Phi(x, t) = \inf_v J_{x,t}(v)$$

- MFG is called potential MFG if

$$F(x, \rho) = \frac{\delta \mathcal{F}(\rho)}{\delta \rho}(x), \quad G(x, \rho) = \frac{\delta \mathcal{G}(\rho)}{\delta \rho}(x),$$



Hamilton-Jacobi-Bellman (HJB) Equation

Lasry & Lions '06: First-order optimality conditions of potential MFG are

$$-\partial_t \Phi(x,t) + \frac{1}{2}\|\Phi(x,t)\|^2 = F(x,\rho(x,t)), \qquad \Phi(x,1) = G(x,\rho(x,1)), \tag{HJB}$$

and

$$\partial_t \rho(x,t) - \nabla \cdot (\rho(x,t)\nabla\Phi(x,t)) = 0, \quad \rho(x,0) = \rho_0(x) \tag{CE}$$

Three options for solving the problem

- minimize $\mathcal{J}_{\text{MFG}}$ w.r.t. v or $-\nabla\Phi$ (variational problem)
- minimize $J_{x,t}$ w.r.t. v or $-\nabla\Phi$ for all agents (microscopic view)
- solve (HJB) and (CE) (high-dimensional PDEs)

Hamilton-Jacobi-Bellman (HJB) Equation

Lasry & Lions '06: First-order optimality conditions of potential MFG are

$$-\partial_t \Phi(x,t) + \frac{1}{2}\|\Phi(x,t)\|^2 = F(x,\rho(x,t)), \qquad \Phi(x,1) = G(x,\rho(x,1)), \tag{HJB}$$

and

$$\partial_t \rho(x,t) - \nabla \cdot (\rho(x,t)\nabla\Phi(x,t)) = 0, \quad \rho(x,0) = \rho_0(x) \tag{CE}$$

Three options for solving the problem

- minimize $\mathcal{J}_{\text{MFG}}$ w.r.t. v or $-\nabla\Phi$ (variational problem)
- minimize $J_{x,t}$ w.r.t. v or $-\nabla\Phi$ for all agents (microscopic view)
- solve (HJB) and (CE) (high-dimensional PDEs)

Today: Combine advantages of the above to tackle curse of dimensionality

- formulate as variational problem. minimize $\mathcal{J}_{\text{MFG}}(-\nabla\Phi)$
- eliminate (CE) with Lagrangian PDE solver $\rightsquigarrow$ mesh-free, parallel
- parameterize Φ with Neural Network $\rightsquigarrow$ mesh-free, efficient(?)
- penalize violations of (HJB) $\rightsquigarrow$ regularity, global convergence(?)

Lagrangian Method

Lagrangian Method for Continuity Equation

Assume Φ given. Then, the solution to

$$\partial_t \rho(x,t) - \nabla \cdot (\rho(x,t) \nabla \Phi(x,t)) = 0, \quad \rho(x,0) = \rho_0(x)$$

satisfies

$$\rho(z(x,t),t) \det \nabla z(x,t) = \rho_0(x)$$

along the characteristic curve given by

$$\partial_t z(x,t) = -\nabla \Phi(z(x,t)), \quad z(x,0) = x.$$

Problem: Computing $\det \nabla z(x,t)$ needs $\mathcal{O}(d^3)$ flops.

Lagrangian Method for Continuity Equation

Assume Φ given. Then, the solution to

$$\partial_t \rho(x,t) - \nabla \cdot (\rho(x,t) \nabla \Phi(x,t)) = 0, \quad \rho(x,0) = \rho_0(x)$$

satisfies

$$\rho(z(x,t),t) \det \nabla z(x,t) = \rho_0(x)$$

along the characteristic curve given by

$$\partial_t z(x,t) = -\nabla \Phi(z(x,t)), \quad z(x,0) = x.$$

Problem: Computing $\det \nabla z(x,t)$ needs $\mathcal{O}(d^3)$ flops.

Idea: Use Jacobi's identity to compute $l(x,t) = \log \det(\nabla z(x,t))$

$$\partial_t l(x,t) = \Delta \Phi(z(x,t),t), \quad l(x,0) = 0.$$

Note: Can combine the ODE solves for position and $\log \det$ along $z(x,\cdot)$

Lagrangian Method for Optimal Transport

$$\text{minimize}_{\Phi} \mathbb{E}_{\rho_0}\big(c_{\mathrm{L}}(x,1) + G(z(x,1)) + \alpha_1 c_{\mathrm{H}}(x,1) + \alpha_2 \|\Phi(z(x,1),1) - G(z(x,1))\|\big)$$

$$\text{subject to} \quad \partial_t \begin{pmatrix} z(x,t) \\ l(x,t) \\ c_{\mathrm{L}}(x,t) \\ c_{\mathrm{H}}(x,t) \end{pmatrix} = \begin{pmatrix} -\nabla\Phi(z(x,t),t) \\ -\Delta\Phi(z(x,t),t) \\ \frac{1}{2}\|\nabla\Phi(z(x,t),t)\|^2 \\ \left|\partial_t\Phi(z(x,t),t) - \frac{1}{2}\|\nabla\Phi(z(x,t),t)\|^2\right| \end{pmatrix}, \quad t \in (0,1]$$

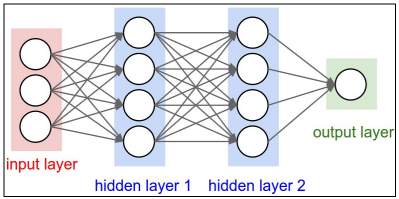
$$z(x,0) = 0, \quad l(x,0) = c_{\mathrm{L}}(x,0) = c_{\mathrm{H}}(x,0) = 0$$

- z and $l = \log\det$ needed to solve continuity eq.
- c_{L} and c_{H} accumulate cost along characteristic
- α_1, α_2: penalty parameters for HJB violation
- discretize with Monte Carlo
- can use SA (SGD, ADAM,...) or SAA (BFGS, Newton,...) methods
- 🌞 no grid needed and 🌞 computation can be parallelized over x

Next, parameterize Φ with NN. Needed: $\nabla\Phi$ and $\Delta\Phi$

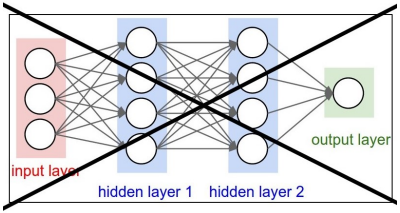
Neural Network Model

Deep Learning Revolution (?)



- deep learning: use neural networks (from $\approx$ 1950's) with many hidden layers
- able to "learn" complicated patterns from data
- applications: image classification, face recognition, segmentation, driverless cars, ...
- recent success fueled by: massive data sets, computing power
- A few recent references:
  - **A radical new neural network design could overcome big challenges in AI**, MIT Tech Review '18
  - Data Scientist: Sexiest Job of the 21st Century, Harvard Business Rev '17

Deep Learning Revolution (?)



$$\begin{cases} \mathbf{Y}_{j+1} &= \sigma(\mathbf{K}_j\mathbf{Y}_j + \mathbf{b}_j) \\ \mathbf{Y}_{j+1} &= \mathbf{Y}_j + \sigma(\mathbf{K}_j\mathbf{Y}_j + \mathbf{b}_j) \\ \mathbf{Y}_{j+1} &= \mathbf{Y}_j + \sigma\left(\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_{j,1}) + \mathbf{b}_{j,2}\right) \\ & \vdots \end{cases}$$

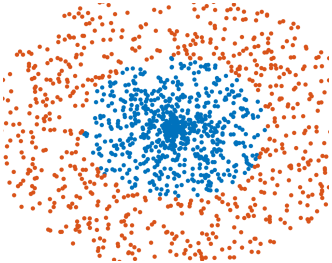
(Notation: $\mathbf{Y}_j$: features, $\mathbf{K}_j, \mathbf{b}_j$: weights, σ : activation)

- deep learning: use neural networks (from $\approx$ 1950's) with many hidden layers
- able to "learn" complicated patterns from data
- applications: image classification, face recognition, segmentation, driverless cars, ...
- recent success fueled by: massive data sets, computing power
- A few recent references:
  - **A radical new neural network design could overcome big challenges in AI**, MIT Tech Review '18
  - Data Scientist: Sexiest Job of the 21st Century, Harvard Business Rev '17

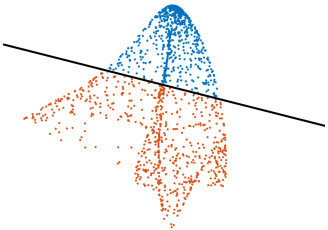
Deep Residual Neural Networks (simplified)

Award-winning forward propagation

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + \mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$



input features, $\mathbf{Y}_0$



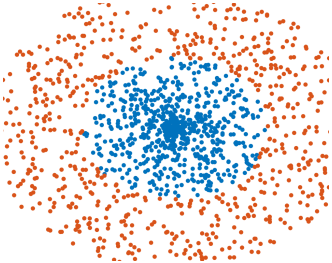
propagated features, $\mathbf{Y}_N$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.

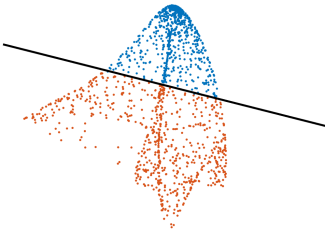
Deep Residual Neural Networks (simplified)

Award-winning forward propagation

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$



input features, $\mathbf{Y}_0$



propagated features, $\mathbf{Y}_N$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.

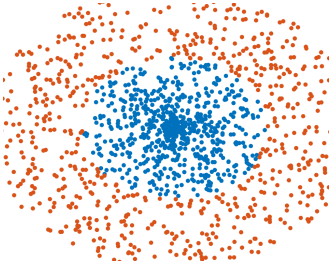
Deep Residual Neural Networks (simplified)

Award-winning forward propagation

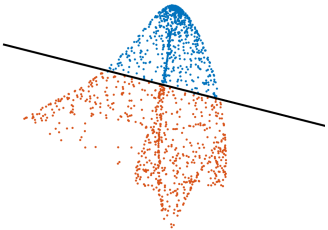
$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$

ResNet is forward Euler discretization of

$$\partial_t \mathbf{y}(t) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right), \qquad \mathbf{y}(0) = \mathbf{y}_0.$$



input features, $\mathbf{Y}_0$



propagated features, $\mathbf{Y}_N$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.

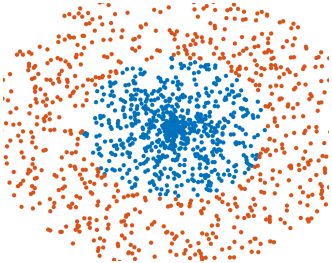
Deep Residual Neural Networks (simplified)

Award-winning forward propagation

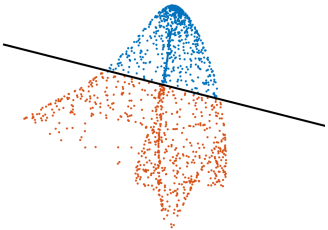
$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$

ResNet is forward Euler discretization of

$$\partial_t \mathbf{y}(t) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right), \qquad \mathbf{y}(0) = \mathbf{y}_0.$$



input features, $\mathbf{Y}_0$



propagated features, $\mathbf{Y}_N$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.

Deep Residual Neural Networks (simplified)

Award-winning forward propagation

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$

ResNet is forward Euler discretization of

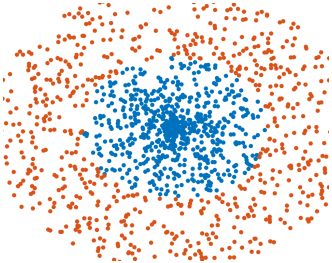
$$\partial_t \mathbf{y}(t) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right), \qquad \mathbf{y}(0) = \mathbf{y}_0.$$

Notation: $\boldsymbol{\theta}(t) = (\mathbf{K}_1(t), \mathbf{K}_2(t), \mathbf{b}(t))$ and

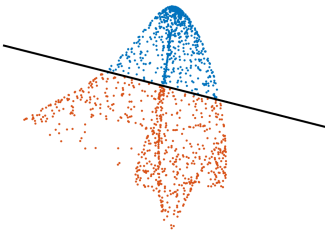
$$\partial_t \mathbf{y}(t) = f(\mathbf{y}, \boldsymbol{\theta}(t)), \quad \mathbf{y}(0) = \mathbf{y}_0$$

where $f(\mathbf{y}, \boldsymbol{\theta}) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right).$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.

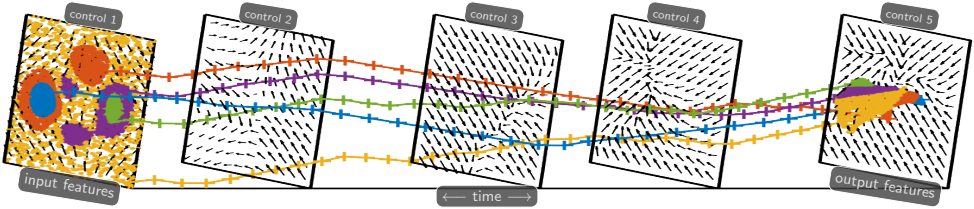


input features, $\mathbf{Y}_0$



propagated features, $\mathbf{Y}_N$

Optimal Control Approaches to Deep Learning



Deep Learning meets optimal control / parameter estimation.

- new ways to analyze and design neural networks
- expose similarities to trajectory problem, optimal transport, image registration, ...
- training algorithms motivated by (robust) optimal control
- discrete ResNet $\rightsquigarrow$ continuous problem $\rightsquigarrow$ discrete architecture

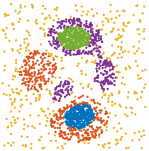
Overview: Deep Neural Networks motivated by PDEs

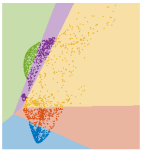
Optimal control formulation

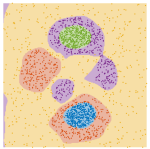
- new insights, theory, algorithms
- stability and impact on convergence/generalization

Stability and well-posedness

- stability: continuous (constraints/design) vs. discrete
- Verlet: reversible and stable networks (memory-free)







E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

E Holtham et al.
Learning Across Scales.
AAAI, 2018.

B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Overview: Deep Neural Networks motivated by PDEs

Optimal control formulation

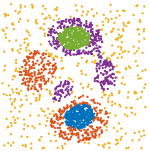
- new insights, theory, algorithms
- stability and impact on convergence/generalization

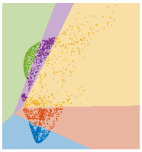
Stability and well-posedness

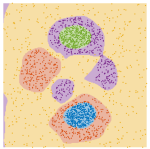
- stability: continuous (constraints/design) vs. discrete
- Verlet: reversible and stable networks (memory-free)

Properties of PDE-inspired CNNs

- parabolic: numerical stability, multiscale, . . .
- hyperbolic: reversible, preserve high-frequency features,. . .
- IMEX: stability, global coupling in feature space
- Lean: drastical reduction in #weights and flops







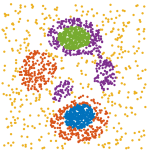
E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

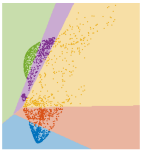
E Holtham et al.
Learning Across Scales.
AAAI, 2018.

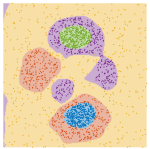
B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Overview: Deep Neural Networks motivated by PDEs







Optimal control formulation

- new insights, theory, algorithms
- stability and impact on convergence/generalization

Stability and well-posedness

- stability: continuous (constraints/design) vs. discrete
- Verlet: reversible and stable networks (memory-free)

Properties of PDE-inspired CNNs

- parabolic: numerical stability, multiscale, . . .
- hyperbolic: reversible, preserve high-frequency features,. . .
- IMEX: stability, global coupling in feature space
- Lean: drastical reduction in #weights and flops

See also recorded talks in IPAM Long Program MLP 2019

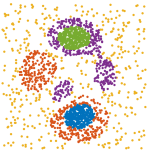
E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

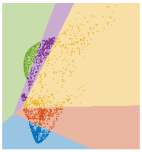
E Holtham et al.
Learning Across Scales.
AAAI, 2018.

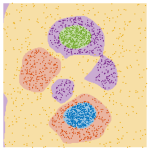
B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Overview: Deep Neural Networks motivated by PDEs







Optimal control formulation

- new insights, theory, algorithms
- stability and impact on convergence/generalization

Stability and well-posedness

- stability: continuous (constraints/design) vs. discrete
- Verlet: reversible and stable networks (memory-free)

Properties of PDE-inspired CNNs

- parabolic: numerical stability, multiscale, . . .
- hyperbolic: reversible, preserve high-frequency features,. . .
- IMEX: stability, global coupling in feature space
- Lean: drastical reduction in #weights and flops

See also recorded talks in IPAM Long Program MLP 2019

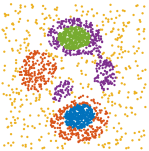
E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

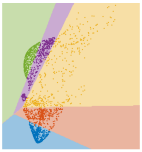
E Holtham et al.
Learning Across Scales.
AAAI, 2018.

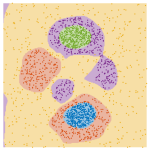
B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Overview: Deep Neural Networks motivated by PDEs







Optimal control formulation

- new insights, theory, algorithms
- stability and impact on convergence/generalization

Stability and well-posedness

- stability: continuous (constraints/design) vs. discrete
- Verlet: reversible and stable networks (memory-free)

Properties of PDE-inspired CNNs

- parabolic: numerical stability, multiscale, . . .
- hyperbolic: reversible, preserve high-frequency features,. . .
- IMEX: stability, global coupling in feature space
- Lean: drastical reduction in #weights and flops

See also recorded talks in IPAM Long Program MLP 2019

E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

E Holtham et al.
Learning Across Scales.
AAAI, 2018.

B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Neural Network Model for Mean Field Games

Estimate mapping from $s = (x, t) \in \mathbb{R}^{d+1}$ to $\Phi(x, t) \in \mathbb{R}$ via (NN + quadratic)

$$\Phi(s, \theta) = w^\top N(s, \theta_N) + \frac{1}{2} s^\top A s + c^\top s + b, \quad \theta = (w, \theta_N, \mathrm{vec}(A), c, b)$$

We use M-layer ResNet with weights $\theta_N = (\mathrm{vec}(K_0), \ldots, \mathrm{vec}(K_M), b_0, \ldots, b_M)$.

Neural Network Model for Mean Field Games

Estimate mapping from $s = (x,t) \in \mathbb{R}^{d+1}$ to $\Phi(x,t) \in \mathbb{R}$ via (NN + quadratic)

$$\Phi(s,\theta) = w^\top N(s,\theta_N) + \frac{1}{2} s^\top A s + c^\top s + b, \quad \theta = (w, \theta_N, \mathrm{vec}(A), c, b)$$

We use M-layer ResNet with weights $\theta_N = (\mathrm{vec}(K_0), \ldots, \mathrm{vec}(K_M), b_0, \ldots, b_M)$.

forward propagation:
$w^\top N(s,t)) = w^\top u_M$

$$\begin{aligned} u_0 &= \sigma(K_0 s + b_0) \\ u_1 &= u_0 + h\sigma(K_1 u_0 + b_1) \\ &\vdots \qquad\qquad \vdots \\ u_M &= u_{M-1} + h\sigma(K_M u_{M-1} + b_M), \end{aligned}$$

Neural Network Model for Mean Field Games

Estimate mapping from $s = (x,t) \in \mathbb{R}^{d+1}$ to $\Phi(x,t) \in \mathbb{R}$ via (NN + quadratic)

$$\Phi(s,\theta) = w^\top N(s,\theta_N) + \frac{1}{2} s^\top A s + c^\top s + b, \quad \theta = (w, \theta_N, \mathrm{vec}(A), c, b)$$

We use M-layer ResNet with weights $\theta_N = (\mathrm{vec}(K_0), \ldots, \mathrm{vec}(K_M), b_0, \ldots, b_M)$.

forward propagation:
$w^\top N(s,t)) = w^\top u_M$

$$\begin{aligned} u_0 &= \sigma(K_0 s + b_0) \\ u_1 &= u_0 + h\sigma(K_1 u_0 + b_1) \\ &\vdots \qquad\qquad \vdots \\ u_M &= u_{M-1} + h\sigma(K_M u_{M-1} + b_M), \end{aligned}$$

backward propagation: $\nabla_s (w^\top N(s,t))$

$$\begin{aligned} z_M &= w + hK_M^\top \mathrm{diag}(\sigma'(K_M u_{M-1} + b_M))w, \\ &\vdots \qquad\qquad \vdots \\ z_1 &= z_2 + hK_1^\top \mathrm{diag}(\sigma'(K_1 u_0 + b_1))z_2, \\ z_0 &= K_0^\top \mathrm{diag}(\sigma'(K_0 s + b_0))z_1, \end{aligned}$$

Next: Compute $\Delta\Phi(s,\theta) = \mathrm{tr}\left(E^\top (\nabla_s^2 (N(s,\theta_N)w) + A)E\right)$,

If $E \in \mathbb{R}^{m \times d}$ this can be done in $\mathcal{O}(m^2 \cdot d \cdot M)$ FLOPS

Numerical Experiments

Optimal Transport Experiment - Setup

- initial density

$$\rho_0(x) = \frac{1}{8}\sum_{j=1}^{8} \rho_G\left(x, \mathbf{m}_j, 0.3 \cdot \mathbf{I}\right),$$

$$\mathbf{m}_j = 4 \cdot \cos\left(\frac{2\pi}{9} j\right) \mathbf{e}_1 + 4 \cdot \sin\left(\frac{2\pi}{9} j\right) \mathbf{e}_2$$

- target density

$$\rho_1(x) = \rho_G\left(x, \mathbf{0}, 0.3 \cdot \mathbf{I}\right).$$

Optimal Transport Experiment - Setup

- initial density

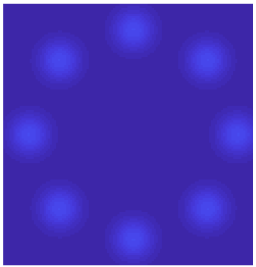
$$\rho_0(x) = \frac{1}{8}\sum_{j=1}^{8} \rho_G\left(x, \mathbf{m}_j, 0.3 \cdot \mathbf{I}\right),$$

$$\mathbf{m}_j = 4 \cdot \cos\left(\frac{2\pi}{9} j\right) \mathbf{e}_1 + 4 \cdot \sin\left(\frac{2\pi}{9} j\right) \mathbf{e}_2$$

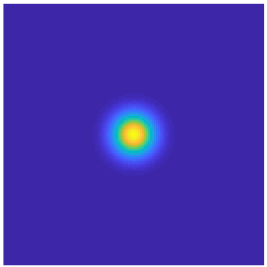
- target density

$$\rho_1(x) = \rho_G\left(x, \mathbf{0}, 0.3 \cdot \mathbf{I}\right).$$

initial density

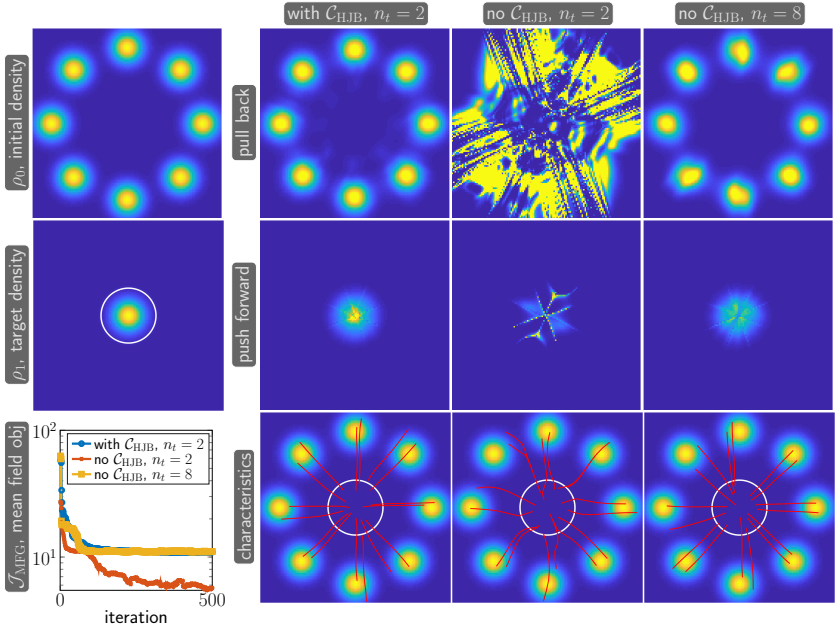


target density



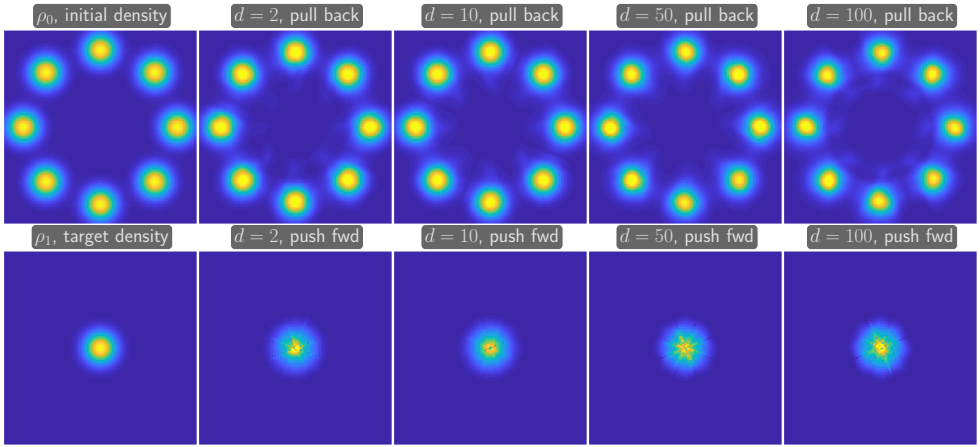
- small ResNet (two layers, width 16)
- optimization strategy
  - SAA approach with BFGS
  - 500 iter / resample every 25 iter
  - simple Armijo Backtracking

Experiment 1: Benefit of HJB Penalty



HJB penalty improves accuracy and(!) lowers computational costs

Experiment 2: Scalability to Higher Dimensions

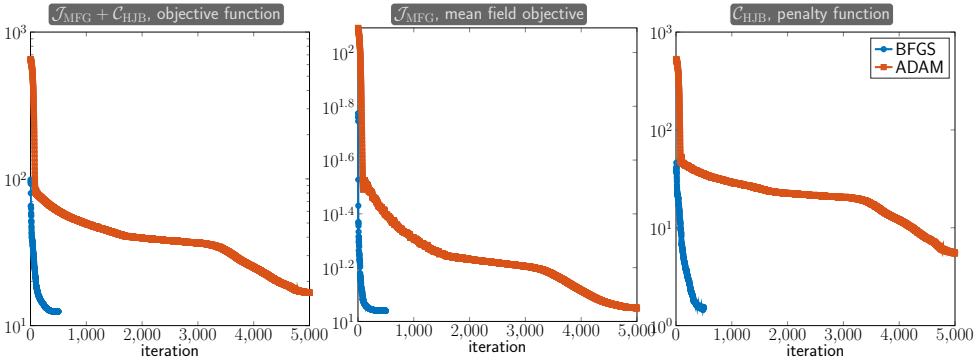


d	N	$\mathcal{L}$	$\mathcal{G}$	$\mathcal{C}_{\text{HJB}}$	time/iter (s)
2	1,024	1.08e+01	1.41e-01	1.53e+00	1.437
10	4,096	1.08e+01	1.85e-01	1.25e+00	8.408
50	16,384	1.10e+01	2.41e-01	4.85e+00	65.706
100	36,864	1.11e+01	3.22e-01	7.37e+00	283.259

qualitatively similar results in all dimensions / moderate growth of runtime

Experiment 3: SAA vs. SA

Compare SAA method with BFGS to ADAM



objective function approximated using 1,024 validation samples.

BFGS converges faster and reduces HJB penalty more substantially

Experiment: Comparison with Eulerian Solver

Eulerian scheme:

- dynamical OT formulation
- conservative finite volume
- leads to convex optimization
- solved to high accuracy with Newton's method

E Haber, R Horesh
A Multilevel Method for the Solution of Time Dependent Optimal Transport, NM-TMA 8(1), 2015.

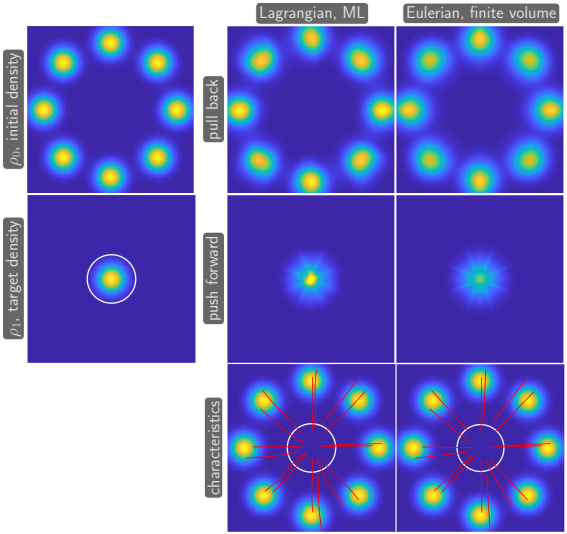
Experiment: Comparison with Eulerian Solver

Eulerian scheme:

- dynamical OT formulation
- conservative finite volume
- leads to convex optimization
- solved to high accuracy with Newton's method

Comparison:

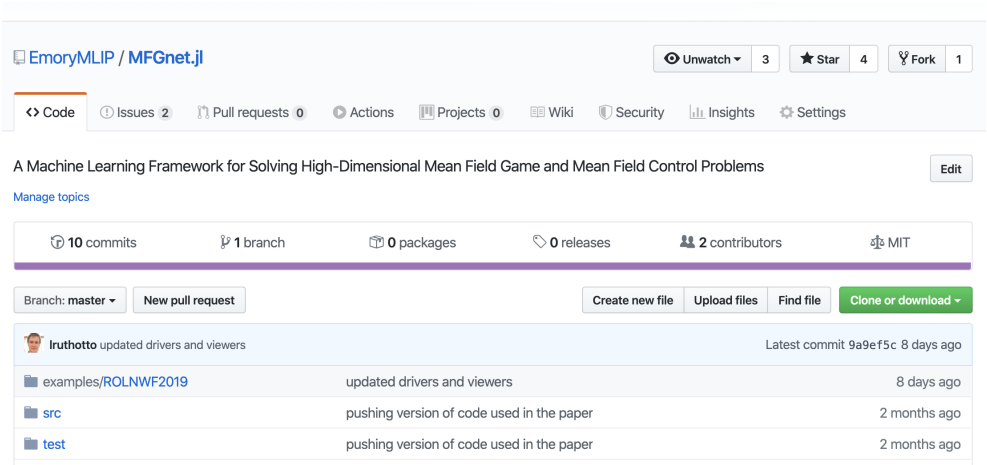
- no. of parameters: 3,080,448 vs. 365
- suboptimality of NN $\approx 0.5\%$



E Haber, R Horesh
A Multilevel Method for the Solution of Time Dependent Optimal Transport, NM-TMA 8(1), 2015.

Summary

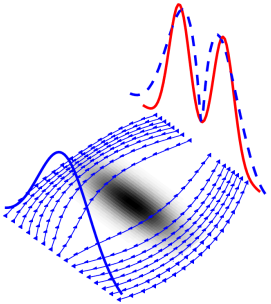
MFGnet.jl - Julia Package

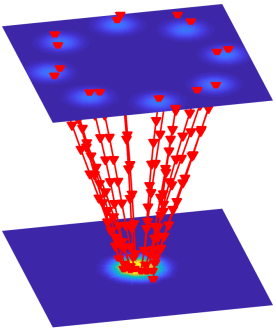


https://github.com/EmoryMLIP/MFGnet.jl

Σ: Machine Learning for High-Dimensional OT

- **Math: Lagrangian Method for MFG**
  - MFG theory: HJB, potential, and optimality conditions
  - combine microscopic computation and macroscopic penalty
  - mesh-free PDE solver and integration
- **ML: Neural Network for Potential**
  - allows to monitor and enforce HJB
  - direct computation of Laplacian (for now)
  - ResNet: multilevel, stability
- **Optimal Transport**
  - HJB penalty improves accuracy and efficiency
  - Scalability to higher dimensions
  - Open issue: optimization





more examples and generalizations at
`github.com/EmoryMLIP/MFGnet.jl` and in:

LR, S Osher, W Li, L Nurbekyan, S Wu Fung
A Machine Learning Framework for Solving High-Dimensional Mean Field Game and Mean Field Control Problems, arXiv, 2019.