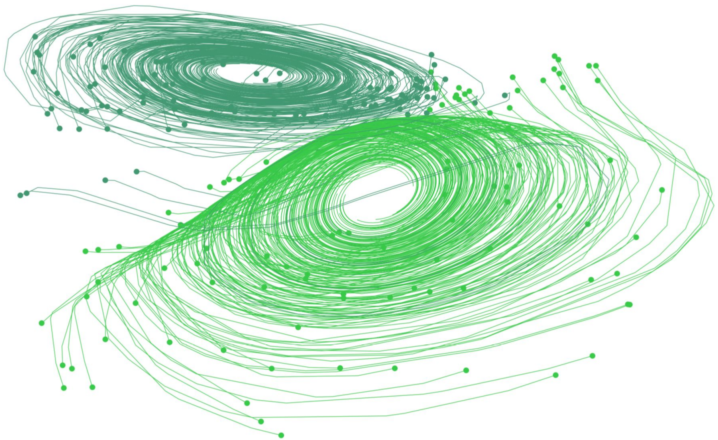
Neural Ordinary Differential Equations



HIGHER SCHOOL OF ECONOMICS
NATIONAL RESEARCH UNIVERSITY

Alexander Markovich

Moscow 2019

How to solve ODE

Problem $\dot{y} = y,\ y(0) = 1$

Solution №1

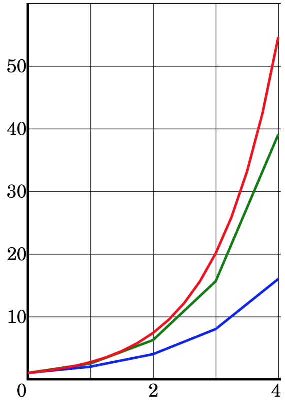
$$\frac{dy}{dt} = y \quad \frac{dy}{y} = dt \quad \int \frac{dy}{y} = \int dt$$

$$\ln|y| = t \quad \Rightarrow \quad y = e^t$$

Solution №2

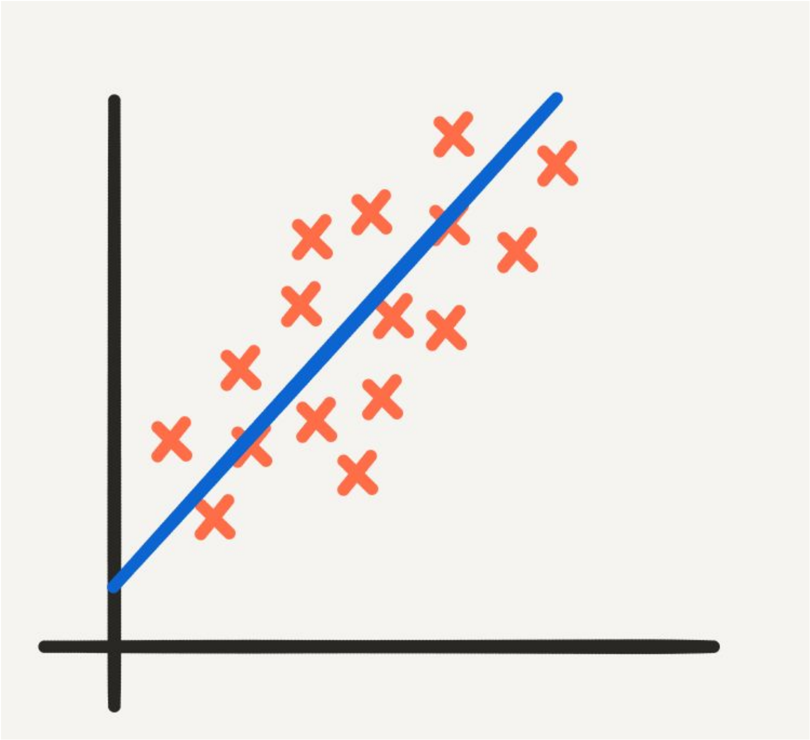
$$\frac{y_{n+1} - y_n}{\underbrace{t_{n+1} - t_n}_{h}} = f(t_n, y_n)$$

$$y_{n+1} = y_n + hf\,(t_n, y_n)$$



n	y_n	t_n	$f\,(t_n, y_n)$	h	Δy	y_{n+1}
0	1	0	1	1	1	2
1	2	1	2	1	2	4
2	4	2	4	1	4	8
3	8	3	8	1	8	16

ODE and ML



Classical ML

$$f : \mathcal{X} \to \mathcal{Y}$$

$$f(x) = \hat{y} = ax + b$$

ML with ODE

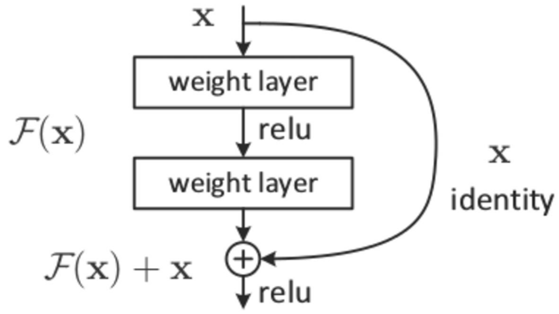
$$y = f(x)$$

$$\frac{dy}{dx} = f'(x)$$

ResNet and Euler's method

For n-th layer:

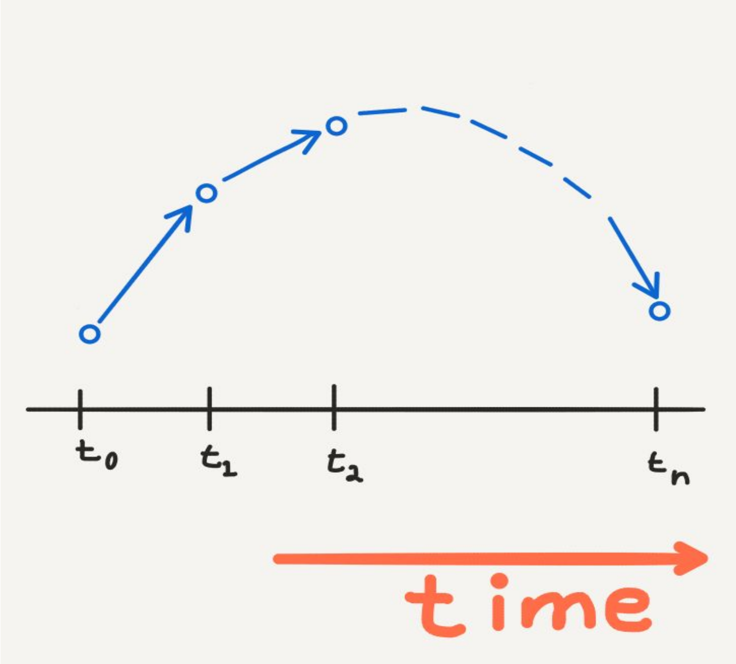
$$x_{n+1} = x_n + F(x_n)$$

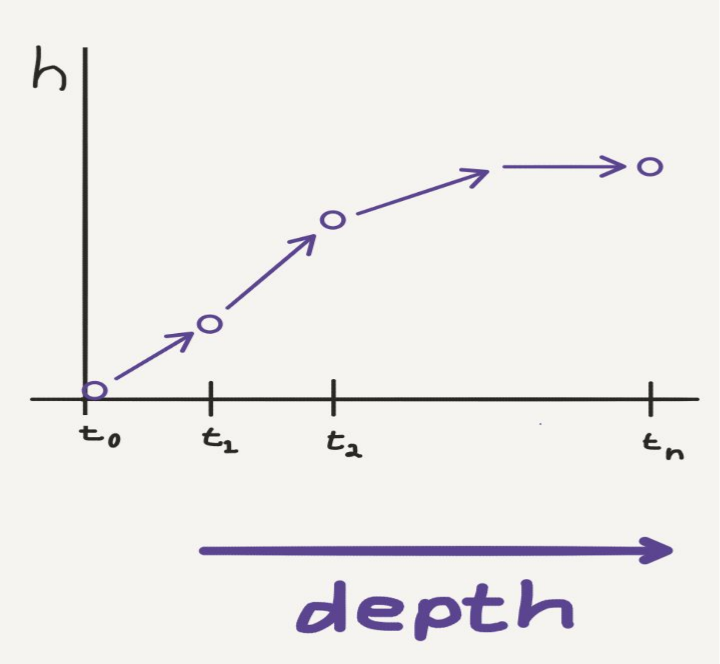


What to add to get the Euler method?

$$x_{n+1} = x_n + hF(x_n), h = 1 = n + 1 - n$$

ResNet and Euler's method





$$y_{n+1} = y_n + hf(t_n, y_n)$$

$$y_{n+1} = y_n + F(y_n)$$

NN as ODE

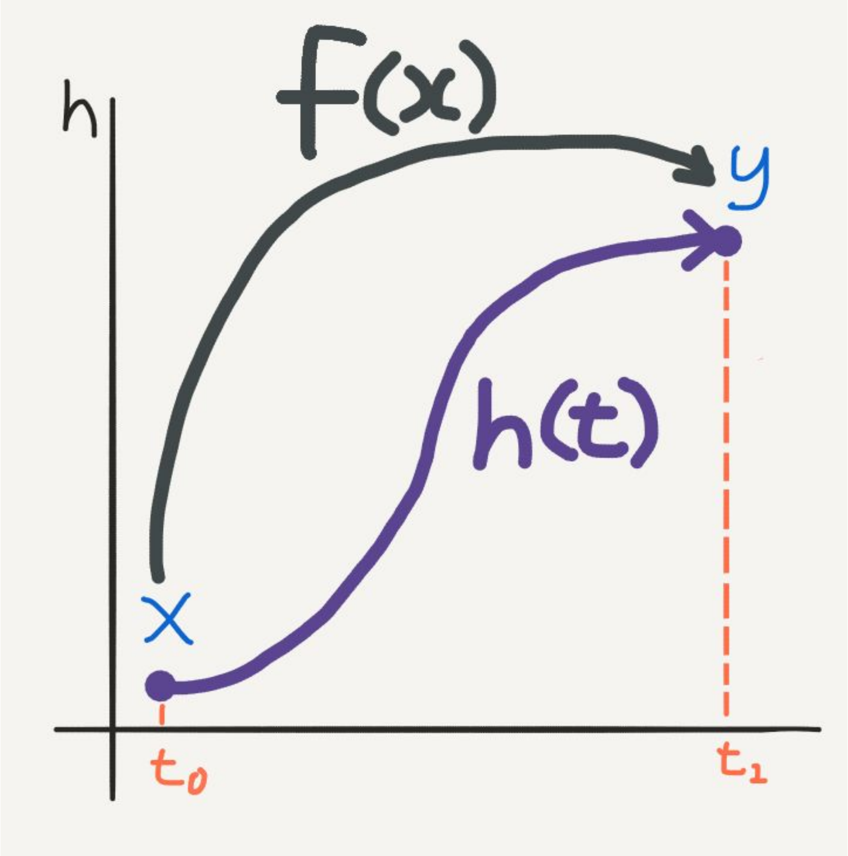
- We have NN with hidden states
- Each state depend on parameters $$h_{t+1} = h_t + f(h_t, \theta_t)$$
- NN are discretisation of hidden states in a latent space

$$\text{Number of layers} \to \infty$$
$$\text{Step size} \to 0$$

$$\frac{dh(t)}{dt} = f(t, h(t), \theta_t)$$

$$f \text{ is NN}, \ t \text{ is layer}$$

NN as ODE



$$h(t)\ -\ ?$$

Bad idea $$h(t) = \int f(t, h(t), \theta_t)dt$$

Good idea Numerical ODE Method

$$h(t_0) = x \qquad h(t_1) = y$$

$$\hat{y} = h(t_1) = \mathrm{ODESolver}(h(t_0), t_0, t_1, \theta, f)$$

How to train?

Backprop Through Depth

$$\mathscr{L}(t_0, t_1, \theta_t) = \mathscr{L}(\mathrm{ODESolver}(h(t_0), t_0, t_1, \theta, f))$$

$$\frac{\partial \mathscr{L}}{\partial h(t)} - ?$$

$$a(t) = -\frac{\partial \mathscr{L}}{\partial h(t)}$$

$$\frac{da(t)}{dt} = -a(t)^T \frac{\partial f(t, h(t), \theta_t)}{\partial h(t)}$$

$$\frac{\partial \mathscr{L}}{\partial h(t)} = \int a(t)^T \frac{\partial f(t, h(t), \theta_t)}{\partial h(t)} dt$$

$$\mathrm{ODESolver}(\ \cdot\ )$$

Everything together

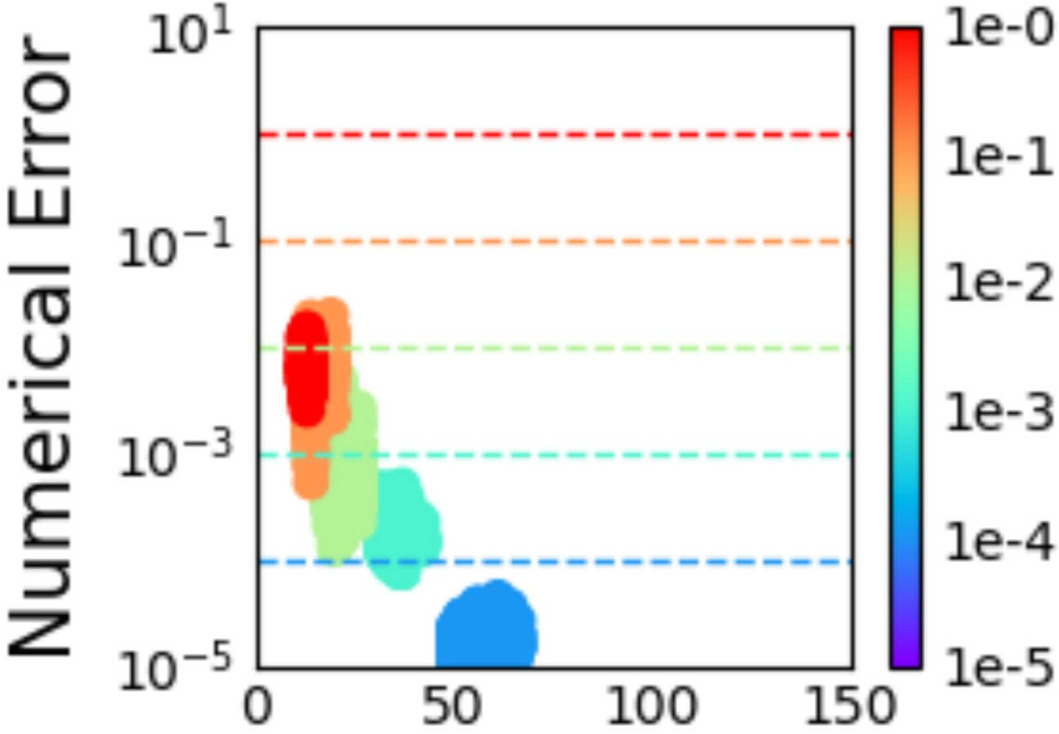
- We have a set of NN pairs of data points $$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$
- Model the derivative $$\frac{dy}{dx} = f(x, y)$$
- Parameterize approximation by NN with hidden states $$\frac{dh(t)}{dt} = f(t, h(t), \theta)$$
- ODE Solver $$\hat{y} = h(t_1) = \mathrm{ODESolver}(h(t_0), t_0, t_1, \theta, f)$$
- Backprop Through Depth $$\frac{\partial \mathcal{L}}{\partial \theta}, \frac{\partial \mathcal{L}}{\partial h(t)}$$

MNIST and NODE

Replace 6 ResNet blocks with ODE-Net

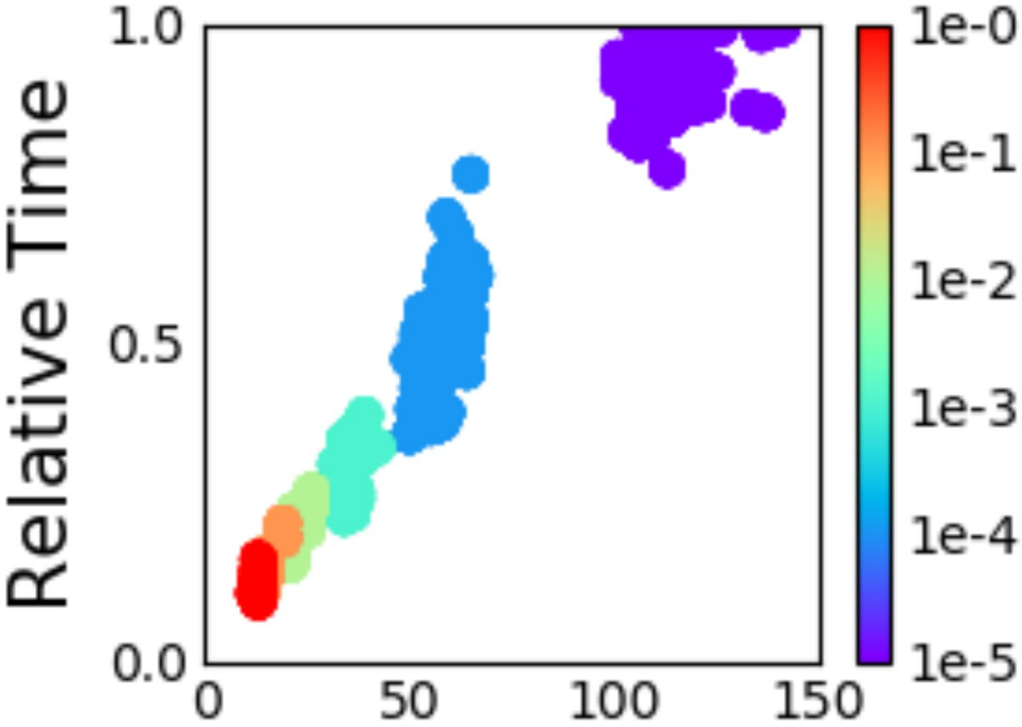
	Test Error	# Params	Memory	Time
1-Layer MLP†	1.60%	0.24 M	-	-
ResNet	0.41%	0.60 M	$\mathcal{O}(L)$	$\mathcal{O}(L)$
RK-Net	0.47%	0.22 M	$\mathcal{O}(\tilde{L})$	$\mathcal{O}(\tilde{L})$
ODE-Net	0.42%	0.22 M	$\boldsymbol{\mathcal{O}(1)}$	$\mathcal{O}(\tilde{L})$

Explicit Error Control



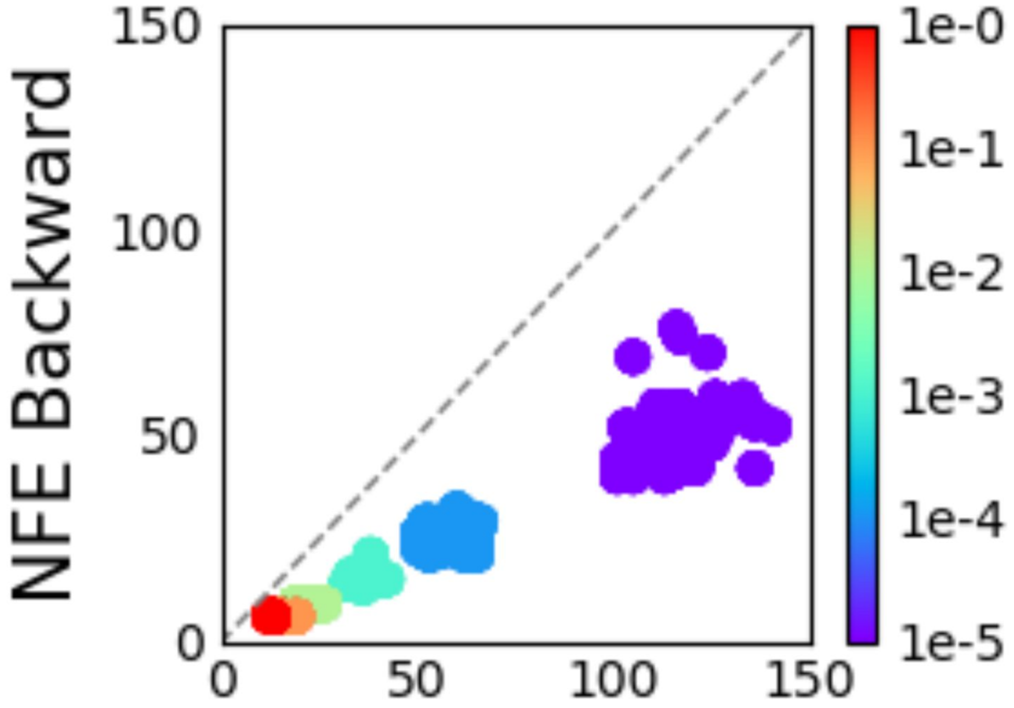
(a) NFE Forward

Speed-Accuracy Tradeoff



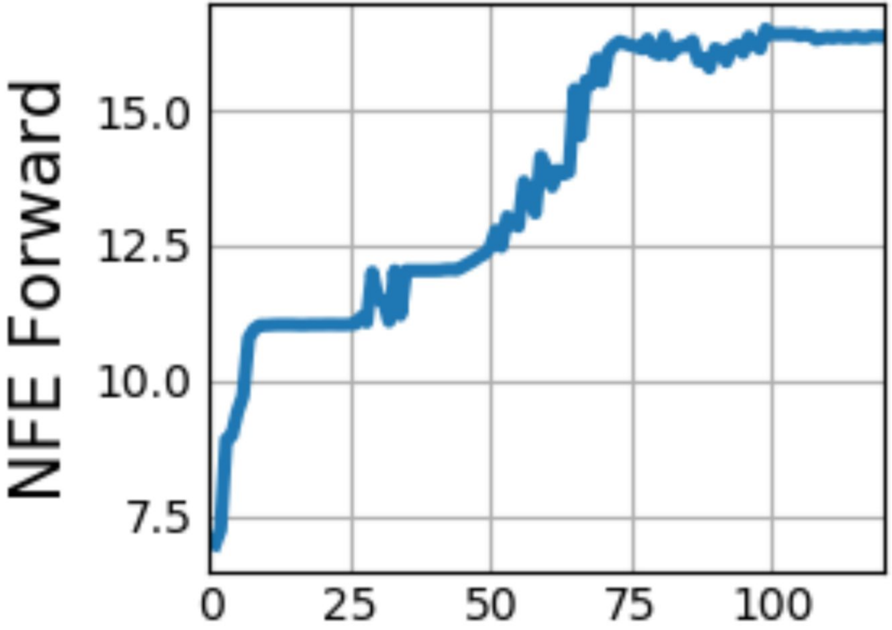
(b) NFE Forward

Reverse vs Forward Cost



(c) NFE Forward

How complex are the dynamics?



(d) Training Epoch

Normalization Flows

$$\mathbf{z}_1 = f(\mathbf{z}_0) \implies \log p(\mathbf{z}_1) = \log p(\mathbf{z}_0) - \log\left|\det \frac{\partial f}{\partial \mathbf{z}_0}\right|$$

- Determinant of Jacobian has cost O(D^3)

- Matrix determinant lemma gives O(DH^3) cost

- Example of planar NF:

$$\mathbf{z}(t+1) = \mathbf{z}(t) + uh\big(w^\top \mathbf{z}(t) + b\big), \quad \log p(\mathbf{z}(t+1)) = \log p(\mathbf{z}(t)) - \log\left|1 + u^\top \frac{\partial h}{\partial \mathbf{z}}\right|$$

Continuous Normalization Flows

- What if we move to continuous transformations?

$$\frac{\partial \log p(\mathbf{z}(t))}{\partial t} = -\operatorname{tr}\left(\frac{df}{d\mathbf{z}(t)}\right)$$

- Time-derivative only depends on trace of Jacobian

$$\frac{d\mathbf{z}(t)}{dt} = uh\left(w^\top \mathbf{z}(t) + b\right), \qquad \frac{\partial \log p(\mathbf{z}(t))}{\partial t} = -u^\top \frac{\partial h}{\partial \mathbf{z}(t)}$$

- Trace of sum is sum of traces - O(HD) cost!

$$\frac{d\mathbf{z}(t)}{dt} = \sum_{n=1}^{M} f_n(\mathbf{z}(t)), \qquad \frac{d \log p(\mathbf{z}(t))}{dt} = \sum_{n=1}^{M} \operatorname{tr}\left(\frac{\partial f_n}{\partial \mathbf{z}}\right)$$

CNF Experiments