

# Deep Residual Learning and PDEs on Manifold

Zhen Li <sup>\*</sup>      Zuoqiang Shi<sup>†</sup>

## Abstract

In this paper, we formulate the deep residual network (ResNet) as a control problem of transport equation. In ResNet, the transport equation is solved along the characteristics. Based on this observation, deep neural network is closely related to the control problem of PDEs on manifold. We propose several models based on transport equation, Hamilton-Jacobi equation and Fokker-Planck equation. The discretization of these PDEs on point cloud is also discussed.

**keywords:** Deep residual network; control problem; manifold learning; point cloud; transport equation; Hamilton-Jacobi equation; Fokker-Planck equation.

## 1 Deep Residual Network

Deep convolution neural networks have achieved great successes in image classification. Recently, an approach of deep residual learning is proposed to tackle the degradation in the classical deep neural network [6, 7]. The deep residual network can be realized by adding shortcut connections in the classical CNN. A building block is shown in Fig. 1. Formally, a building block is defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{\mathbf{W}_i\}) + \mathbf{x}.$$

Here  $\mathbf{x}$  and  $\mathbf{y}$  are the input and output vectors of the layers. The function  $\mathcal{F}(\mathbf{x}, \{\mathbf{W}_i\})$  represents the residual mapping to be learned. In Fig. 1,  $\mathcal{F} = \mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 \cdot \mathbf{x})$  in which  $\sigma$  denotes ReLU.

---

<sup>\*</sup>Department of Mathematics, Hong Kong University of Science & Technology, Hong Kong. *Email:* mazli@ust.hk.

<sup>†</sup>Yau Mathematical Sciences Center, Tsinghua University, Beijing, China, 100084. *Email:* zqshi@tsinghua.edu.cn.

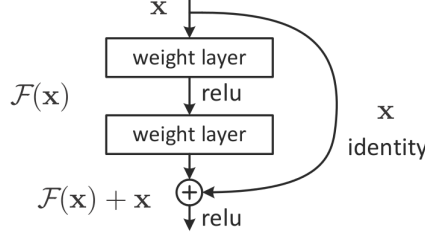


Figure 1: Building block of residual learning (Fig. 1 in [6]).

## 2 Transport Equation and ResNet

Consider the the terminal value problem of linear transport equation in  $\mathbb{R}^d$ :

$$\begin{cases} \frac{\partial u}{\partial t} - \mathbf{v}(\mathbf{x}, t) \cdot \nabla u = 0, & \mathbf{x} \in \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, T) = f(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^d. \end{cases} \quad (1)$$

where  $\mathbf{v}(\mathbf{x}, t)$  is a given velocity field.  $T > 0$  is a positive terminal time.

It is well-known that this problem can be solved along the characteristics.

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{v}(\mathbf{X}(t), t) \quad (2)$$

Along the characteristic lines,  $u$  is a constant.

$$\begin{aligned} \frac{du(\mathbf{X}(t), t)}{dt} &= \frac{\partial u}{\partial t}(\mathbf{X}(t), t) - \frac{d\mathbf{X}(t)}{dt} \cdot \nabla u(\mathbf{X}(t), t) \\ &= \frac{\partial u}{\partial t}(\mathbf{X}(t), t) - \mathbf{v}(\mathbf{x}, t) \cdot \nabla u(\mathbf{X}(t), t) = 0 \end{aligned} \quad (3)$$

which gives

$$u(\mathbf{X}(0), 0) = u(\mathbf{X}(T), T) = f(\mathbf{X}(T)). \quad (4)$$

Let  $(t_k)_{k=0}^L$  with  $t_0 = 0$  and  $t_L = T$  be a partition of  $[0, T]$ . The characteristic of the transport equation (1) can be solved by simple forward Euler discretization,

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \Delta t \mathbf{v}(\mathbf{X}_k, t_k) \quad (5)$$

where  $\Delta t$  is the time step.

Now, we choose a special velocity field

$$\mathbf{v}(\mathbf{x}, t) = \left( \mathbf{W}^{(2)}(t) \cdot \mathbf{a} \left( \mathbf{W}^{(1)}(t) \cdot \mathbf{x} + \mathbf{b}^{(1)}(t) \right) + \mathbf{b}^{(2)}(t) \right), \quad (6)$$

where  $\mathbf{W}^{(1)}(t), \mathbf{W}^{(2)}(t) \in \mathbb{R}^{d \times d}$  are matrices-valued functions of  $t$ .  $\mathbf{b}^{(1)}(t), \mathbf{b}^{(2)}(t) \in \mathbb{R}^d$  are two  $\mathbb{R}^d$ -valued functions of  $t$ .  $\mathbf{a}$  is also a  $\mathbb{R}^d$ -valued function,

$$\mathbf{a}(\mathbf{x}) = (a(x_1), a(x_2), \dots, a(x_3)) \in \mathbb{R}^d, \quad (7)$$

and  $a$  is a given function which is called activation function in neural network.

With above velocity field, the discrete characteristics becomes

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \left( \bar{\mathbf{W}}^{(2)}(t_k) \cdot \mathbf{a} \left( \mathbf{W}^{(1)}(t_k) \cdot \mathbf{X}_k + \mathbf{b}^{(1)}(t_k) \right) + \bar{\mathbf{b}}^{(2)}(t_k) \right) \quad (8)$$

where  $\bar{\mathbf{W}}^{(2)}(t) = \Delta t \mathbf{W}^{(2)}(t)$ ,  $\bar{\mathbf{b}}^{(2)}(t) = \Delta t \mathbf{b}^{(2)}(t)$ .

Notice that the characteristics can be represented as a residual network, Fig. 2.

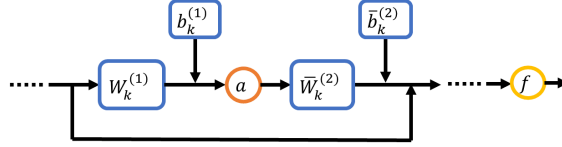


Figure 2: 2-layers shortcut residual neural network.

This is exactly a residual network with bias [6].

Based on above analysis, we see that the training process of ResNet can be formulated as an control problem of a transport equation in  $\mathbb{R}^d$ .

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) - \mathbf{v}(\mathbf{x}, t) \cdot \nabla u(\mathbf{x}, t) = 0, & \mathbf{x} \in \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 1) = f(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^d, \\ u(\mathbf{x}_i, 0) = g(\mathbf{x}_i), & \mathbf{x}_i \in T. \end{cases} \quad (9)$$

where  $T$  denotes the training set.  $g(\mathbf{x}_i)$  is the labeled value at sample  $\mathbf{x}_i$ . Function  $u$  may be scalar or vector value in different applications.

In above control problem formulation (9), the terminal value  $u(\mathbf{x}, 1) = f(\mathbf{x})$  and the model of the velocity field  $\mathbf{v}(\mathbf{x}, t)$  are very important. The whole model is fixed as long as the terminal value and the model of the velocity field are given.

**Terminal Value** The terminal value is corresponding to the output layer in ResNet. Usually, it is consists of a few full connected layers and an output function layer. The often used output functions are sigmoid function  $\sigma(x) = \frac{1}{1+\exp(-x)}$  or softmax function  $\mathbf{softmax}(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$ .

From the control problem point of view, the sigmoid or softmax output function are not good terminal conditions, since they are pre-determined, may be far from the real value. It would be very helpful if we can get a good terminal value. First, it makes the optimization converge quickly. Moreover, a good initialization means that the map between the initial function and the real function is close to identity. Then one natural assumption is that the velocity field is simple and close to zero.

A good initialization may be obtained from weighted nonlocal Laplacian [10]. Denote  $T$  as the training set,  $S$  is a subset of the training set.  $S$  can be chosen at random or use

the idea of cross-validation. Then, an interpolation on  $T$  is given by solving

$$\sum_{\mathbf{x}_j \in T} (w(\mathbf{x}_i, \mathbf{x}_j) + w(\mathbf{x}_j, \mathbf{x}_i))(u(\mathbf{x}_i) - u(\mathbf{x}_j)) + \frac{|T|}{|S|} \sum_{\mathbf{x}_j \in S} w(\mathbf{x}_j, \mathbf{x}_i)(u(\mathbf{x}_i) - g(\mathbf{x}_j)) = 0, \quad \mathbf{x}_i \in T. \quad (10)$$

Here,  $g(\mathbf{x}_i)$  is labeled value on  $\mathbf{x}_i \in S$ ,  $w(\mathbf{x}_i, \mathbf{x}_j)$  is a given weight function. If the control problem (9) is still solved along the characteristics, we need the initial value has an explicit form. Fortunately, weighted nonlocal Laplacian also gives a natural interpolation on  $\mathbb{R}^d$ .

$$u(\mathbf{x}) = \frac{1}{\bar{w}(\mathbf{x})} \sum_{\mathbf{x}_j \in T} w(\mathbf{x}, \mathbf{x}_j)u(\mathbf{x}_j) + \frac{|T|}{|S|} \sum_{\mathbf{x}_j \in S} w(\mathbf{x}, \mathbf{x}_j)g(\mathbf{x}_j), \quad (11)$$

where

$$\bar{w}(\mathbf{x}) = \sum_{\mathbf{x}_j \in T} w(\mathbf{x}, \mathbf{x}_j) + \frac{|T|}{|S|} \sum_{\mathbf{x}_j \in S} w(\mathbf{x}, \mathbf{x}_j).$$

**Velocity Field Model** In ResNet, the model of the velocity field  $\mathbf{v}(\mathbf{x}, t)$  is

$$\mathbf{v}(\mathbf{x}, t) = \mathbf{W}^{(2)}(t) \cdot \mathbf{a} \left( \mathbf{W}^{(1)}(t) \cdot \mathbf{x} \right). \quad (12)$$

where  $\mathbf{a}$  is the activation function which is chosen to be ReLU function.  $\mathbf{W}^{(1)}(t)$ ,  $\mathbf{W}^{(2)}(t) \in \mathbb{R}^{d \times d}$  are convolution matrices in  $\mathbb{R}^d$ . The matrices  $\mathbf{W}^{(1)}(t)$ ,  $\mathbf{W}^{(2)}(t)$  are further assumed to be piecewise constant in  $t$ . The number of pieces is corresponding to the number of layers in ResNet. As we can see, the regularity of the velocity field in  $\mathbf{x}$  is related to the activation function and the regularity in  $t$  is determined by the number of layers.

As we mentioned before, the velocity field model may be simplified with good terminal value. For instance, the number of layers is reduced, such that the number of parameters are reduced consequently. The other possible choice of the velocity field model is

$$\mathbf{v}(\mathbf{x}, t) = \sum_{k=1}^K \mathbf{W}_k(t) \cdot \phi_k(\mathbf{x}). \quad (13)$$

Here  $\mathbf{W}_k(t)$  and  $\phi_k(\mathbf{x})$  are basis in  $t$  and  $\mathbf{x}$  respectively. The great success of the convolution neural network (CNN) suggest that  $\mathbf{W}_k(t)$  should be convolution matrices with different support sizes. The activation functions in deep learning may give the basis function  $\phi_k(\mathbf{x})$ .

As we analyzed in this section, ResNet can be formulated as a control problem of the transport equation (9) and the transport equation is solved by characteristic method. Based on this observation, we have many models for ResNet. In the next section, we will discuss some models based on PDEs on manifold.

### 3 Control Problems on Manifold

In the past several decades, many numerical methods for transport equations were developed. Roughly speaking, all the methods can be classified to either Lagrangian method or Eulerian method. Characteristic method is a Lagrangian method. On the other hand, Eulerian methods have been proved to be very successful in many applications. In this section, we will discuss the possibility to solve the control problem in Eulerian grid.

**Transport equation** It is difficult to solve the control problem (9) directly on Eulerian grid due to the high dimensionality of the data ( $d \gg 1$ ). To reduce the dimension, we use the idea of manifold learning and assume that the data set sample a low dimensional manifold  $\mathcal{M} \subset \mathbb{R}^d$ . Then, (9) is actually a PDE evolve in a low dimensional manifold.

$$\begin{cases} \frac{\partial u}{\partial t} - \mathbf{v}(\mathbf{x}, t) \cdot \nabla_{\mathcal{M}} u = 0, & \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 0) = f(\mathbf{x}), & \mathbf{x} \in \mathcal{M}, \\ u(\mathbf{x}_i, 1) = g(\mathbf{x}_i), & \mathbf{x}_i \in T. \end{cases} \quad (14)$$

$\nabla_{\mathcal{M}}$  denotes the gradient on manifold  $\mathcal{M}$ . Let  $X : V \subset \mathbb{R}^m \rightarrow \mathcal{M} \subset \mathbb{R}^d$  be a local parametrization of  $\mathcal{M}$  and  $\theta \in V$ . For any differentiable function  $u : \mathcal{M} \rightarrow \mathbb{R}$ , let  $U(\theta) = f(X(\theta))$ , define

$$D_k u(X(\theta)) = \sum_{i,j=1}^m g^{ij}(\theta) \frac{\partial X_k}{\partial \theta_i}(\theta) \frac{\partial U}{\partial \theta_j}(\theta), \quad k = 1, \dots, d. \quad (15)$$

where  $(g^{ij})_{i,j=1,\dots,m} = G^{-1}$  and  $G(\theta) = (g_{ij})_{i,j=1,\dots,m}$  is the first fundamental form which is defined by

$$g_{ij}(\theta) = \sum_{k=1}^d \frac{\partial X_k}{\partial \theta_i}(\theta) \frac{\partial X_k}{\partial \theta_j}(\theta), \quad i, j = 1, \dots, m. \quad (16)$$

$\nabla_{\mathcal{M}}$  is defined as

$$\nabla_{\mathcal{M}} u = (D_1 u, D_2 u, \dots, D_d u). \quad (17)$$

The manifold  $\mathcal{M}$  is sampled by the data set  $P$ .  $P$  includes the training set and the test set. Using this manifold formulation, the degree of freedom is reduced to  $|P|$  which is computable in the current computer. However, the other problem emerges in this formulation. The sample of  $\mathcal{M}$ , data set  $P$ , consists of unstructured high dimensional points. Unlike the classical numerical methods which solve PDE on regular grids (or meshes), in this case, we need to discretize PDE on unstructured high dimensional point cloud  $P$ . We know very few information of the underlying manifold  $\mathcal{M}$ . To handle this kind of problems, recently,

point integral method (PIM) was developed to solve PDE on point cloud. In PIM, gradient on point cloud is approximate by an integral formula [9].

$$D_k u(\mathbf{x}) = \frac{1}{w_\delta(\mathbf{x})} \int_{\mathcal{M}} (u(\mathbf{x}) - u(\mathbf{y})) (x^k - y^k) R_\delta(\mathbf{x}, \mathbf{y}) d\mathbf{y}. \quad (18)$$

where  $w_\delta(\mathbf{x}) = \int_{\mathcal{M}} R_\delta(\mathbf{x}, \mathbf{y}) d\mathbf{y}$ ,

$$R_\delta(\mathbf{x}, \mathbf{y}) = R\left(\frac{\|\mathbf{x} - \mathbf{y}\|^2}{\delta^2}\right). \quad (19)$$

The kernel function  $R(r) : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  is assumed to be a  $C^2$  function with compact support.

Corresponding discretization is

$$D_k u(\mathbf{x}) = \frac{1}{\bar{w}_\delta(\mathbf{x})} \sum_{\mathbf{y} \in P} (u(\mathbf{x}) - u(\mathbf{y})) (x^k - y^k) R_\delta(\mathbf{x}, \mathbf{y}) V(\mathbf{y}). \quad (20)$$

$\bar{w}_\delta(\mathbf{x}) = \sum_{\mathbf{y} \in P} R_\delta(\mathbf{x}, \mathbf{y}) V(\mathbf{y})$  and  $V(\mathbf{y})$  is the volume weight of  $\mathbf{y}$ .

**Viscous transport equation** To make the model more stable, we can consider to add a viscous term. Now we get a control problem of viscous transport equation.

$$\left\{ \begin{array}{l} \frac{\partial u}{\partial t} - \mathbf{v}(\mathbf{x}, t) \cdot \nabla_{\mathcal{M}} u = \mu \Delta_{\mathcal{M}} u, \quad \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 0) = f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M}, \\ u(\mathbf{x}_i, t) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in S, \\ u(\mathbf{x}_i, 1) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in T \setminus S, \\ \frac{\partial u}{\partial \mathbf{n}}(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial \mathcal{M}. \end{array} \right. \quad (21)$$

where  $\Delta_{\mathcal{M}} u = \sum_{k=1}^d D_k(D_k(u))$  is the Laplace-Beltrami operator on  $\mathcal{M}$ .  $\partial \mathcal{M}$  is the boundary of  $\mathcal{M}$  and  $\mathbf{n}$  is the out normal at  $\partial \mathcal{M}$ .

In this case, we need to add constraints,  $u(\mathbf{x}_i, t) = g(\mathbf{x}_i)$ ,  $\mathbf{x}_i \in S \subset T$  in a subset  $S$  of the training set  $T$ . Otherwise, the solution will be too smooth to fit the data due to the viscosity. The choice of  $S$  is an issue. The simplest way is to choose  $S$  at random. In some sense, the viscous term maintains the regularity of the solution and the convection term is used to fit the data. On the point cloud, the Laplace-Beltrami operator along with the constraints  $u(\mathbf{x}_i, t) = g(\mathbf{x}_i)$ ,  $\mathbf{x}_i \in S \subset T$  can be discretized by the weighted nonlocal Laplacian [10].

**Hamilton-Jacobi equation** Notice that by introducing  $\bar{v}(\mathbf{x}, t) = \mathbf{v}(\mathbf{x}, t) \cdot \mathbf{n}(\mathbf{x}, t)$  and  $\mathbf{n}(\mathbf{x}, t) = \frac{\nabla_{\mathcal{M}} u}{|\nabla_{\mathcal{M}} u|}$ , transport equation in (14) can be rewritten as a Hamilton-Jacobi equation.

$$\left\{ \begin{array}{l} \frac{\partial u}{\partial t} - \bar{v}(\mathbf{x}, t) \cdot |\nabla_{\mathcal{M}} u| = 0, \quad \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 0) = f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M}, \\ u(\mathbf{x}_i, 1) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in T. \end{array} \right. \quad (22)$$

In (14), the velocity field,  $\mathbf{v}(\mathbf{x}, t)$ , is a  $d$ -dimensional vector field. It needs a lot of degree of freedom to model a high dimensional vector field. In the Hamilton-Jacobi model, we only need to model a scalar function  $\bar{v}(\mathbf{x}, t)$ . The number of parameters can be reduced tremendously. The trade-off is that Hamilton-Jacobi equation is a nonlinear equation which is more difficult to solve. Recently, fast solver of Hamilton-Jacobi equation attracts lots of attentions and many efficient methods are developed [1, 2, 3, 4, 5]. On the point cloud, one possible choice to discretize  $|\nabla_{\mathcal{M}}u|$  is

$$|\nabla_{\mathcal{M}}u(\mathbf{x})| = \left( \int_{\mathcal{M}} w(\mathbf{x}, \mathbf{y})(u(\mathbf{x}) - u(\mathbf{y}))^2 d\mathbf{y} \right)^{1/2}. \quad (23)$$

And we can use the radial basis function to model  $\bar{v}(\mathbf{x}, t)$ ,

$$\bar{v}(\mathbf{x}, t) = \sum_{k=1}^K c_k(t) \phi(\mathbf{x}; \mathbf{x}_k), \quad (24)$$

where  $\phi(\mathbf{x}; \mathbf{x}_k)$  is the radial basis function centered at  $\mathbf{x}_k$ .  $c_k(t)$  is chosen to be piecewise constant in  $t$  similar as that in ResNet.

We can also consider to add a viscous term in the Hamilton-Jacobi equation,

$$\left\{ \begin{array}{l} \frac{\partial u}{\partial t} - \bar{v}(\mathbf{x}, t) \cdot |\nabla_{\mathcal{M}}u| = \mu \Delta_{\mathcal{M}}u, \quad \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 0) = f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M}, \\ u(\mathbf{x}_i, t) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in S, \\ u(\mathbf{x}_i, 1) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in T \setminus S, \\ \frac{\partial u}{\partial \mathbf{n}}(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial \mathcal{M}. \end{array} \right. \quad (25)$$

The viscous term can be handled by the weighted nonlocal Laplacian [10].

**Fokker-Planck equation** The other idea to simplify the model of the velocity field is to introduce a potential function  $U(\mathbf{x}, t)$  and assume that

$$\mathbf{v}(\mathbf{x}, t) = \nabla_{\mathcal{M}}U(\mathbf{x}, t). \quad (26)$$

Then, the viscous transport equation in (21) becomes a Fokker-Planck equation.

$$\left\{ \begin{array}{l} \frac{\partial u}{\partial t} + \nabla_{\mathcal{M}}U(\mathbf{x}, t) \cdot \nabla_{\mathcal{M}}u = \mu \Delta_{\mathcal{M}}u, \quad \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}, 0) = f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M}, \\ u(\mathbf{x}_i, t) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in S, \\ u(\mathbf{x}_i, 1) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in T \setminus S, \\ \frac{\partial u}{\partial \mathbf{n}}(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial \mathcal{M}. \end{array} \right. \quad (27)$$

Now, we also only need to model a scalar function  $U(\mathbf{x}, t)$  and the equation is still linear with given potential.

Regarding the discretization of Fokker-Planck equation on point cloud, notice that

$$-\mu\Delta_{\mathcal{M}}u + \nabla_{\mathcal{M}}U(\mathbf{x}, t) \cdot \nabla_{\mathcal{M}}u = -\mu e^{U/\mu} \operatorname{div} \left( e^{-U/\mu} \nabla_{\mathcal{M}}u \right) \quad (28)$$

The elliptic operator with isotropic coefficients also has an integral approximation [8],

$$\begin{aligned} -\int_{\mathcal{M}} \left[ \frac{1}{p^2(\mathbf{y})} \operatorname{div}(p^2(\mathbf{y}) \nabla u(\mathbf{y})) \right] \bar{R}_{\delta}(\mathbf{x}, \mathbf{y}) p(\mathbf{y}) d\mathbf{y} &\approx \frac{4}{\delta^2} \int_{\mathcal{M}} R_{\delta}(\mathbf{x}, \mathbf{y}) (u(\mathbf{x}) - u(\mathbf{y})) p(\mathbf{y}) d\mathbf{y} \\ &- 2 \int_{\partial\mathcal{M}} \frac{\partial u}{\partial \mathbf{n}}(\mathbf{y}) \bar{R}_{\delta}(\mathbf{x}, \mathbf{y}) p(\mathbf{y}) dS_{\mathbf{y}}. \end{aligned} \quad (29)$$

with  $p(\mathbf{x}) = \exp\left(-\frac{U}{2\mu}\right)$  and

$$\bar{R}_{\delta}(\mathbf{x}, \mathbf{y}) = \bar{R}\left(\frac{\|\mathbf{x} - \mathbf{y}\|^2}{\delta^2}\right), \quad \bar{R}(r) = \int_r^{+\infty} R(s) ds. \quad (30)$$

It is easy to develop discretization based on above integral approximation.

We can further simplify the potential function by considering the steady Fokker-Planck equation. In this case, the potential function is independent on  $t$ .

$$\begin{cases} -\mu\Delta_{\mathcal{M}}u + \nabla_{\mathcal{M}}U(\mathbf{x}) \cdot \nabla_{\mathcal{M}}u = 0, & \mathbf{x} \in \mathcal{M} \subset \mathbb{R}^d, t \geq 0, \\ u(\mathbf{x}_i) = g(\mathbf{x}_i), & \mathbf{x}_i \in S \subset T. \end{cases} \quad (31)$$

The potential function  $U$  is learned to fit the data in  $T \setminus S$ ,

$$u(\mathbf{x}_i) = g(\mathbf{x}_i), \quad \mathbf{x}_i \in T \setminus S. \quad (32)$$

The discretization of the steady Fokker-Planck equation can be derived from (28) and (29).

## 4 Discussion

In this paper, we establish the connection between the deep residual network (ResNet) and the transport equation. ResNet can be formulated as solving a control problem of transport equation along the characteristics. Based on this observation, we propose several PDE models on the manifold sampled by the data set. We consider the control problem of transport equation, Hamilton-Jacobi equation and Fokker-Planck equation. The discretization of these PDEs on point cloud are discussed.

This is a very preliminary discussion on the relation between deep learning and PDEs. There are many important issues remaining unresolved including the model of the velocity field, the back propagation method in the control problem and so on. This is just the first step in exploring the relation between deep learning and control problems of PDEs.



## References

- [1] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for certain non-convex hamilton-jacobi equations, projections and differential games. *UCLA CAM-Report 16-27*, 2016.
- [2] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for time-dependent non-convex hamilton-jacobi equations arising from optimal control and differential games problems. *UCLA CAM-Report 16-62*, 2016.
- [3] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for state-dependent hamilton-jacobi equations. *UCLA CAM-Report 17-16*, 2017.
- [4] J. Darbon and S. Osher. Algorithms for overcoming the curse of dimensionality for certain hamilton-jacobi equations arising in control theory and elsewhere. *UCLA CAM-Report 15-50*, 2015.
- [5] J. Darbon and S. Osher. Splitting enables overcoming the curse of dimensionality. *UCLA CAM-Report 15-69*, 2015.
- [6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, 2016.
- [7] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *arXiv:1603.05027*, 2016.
- [8] Z. Li and Z. Shi. A convergent point integral method for isotropic elliptic equations on point cloud. *SIAM: Multiscale Modeling & Simulation*, pages 874–905, 2016.
- [9] Z. Li, Z. Shi, and J. Sun. Point integral method for elliptic equations with variable coefficients on point cloud. *mathscidoc:1708.25001*, 2017.
- [10] Z. Shi, S. Osher, and W. Zhu. Weighted nonlocal laplacian on interpolation from sparse data. *accepted by Journal of Scientific Computing*, 2017.