Supplementary Material: ODE-inspired Network Design for Single Image Super-Resolution

Xiangyu He[1,2*], Zitao Mo[1,2*], Peisong Wang[1], Yang Liu[4], Mingyuan Yang[4], Jian Cheng[1,2,3] ✉
[1] NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Center for Excellence in Brain Science and Intelligence Technology, CAS, Beijing, China
[4] Alibaba Group, Hangzhou, China
{xiangyu.he, zitao.mo, peisong.wang, jcheng}@nlpr.ia.ac.cn

1. Numerical solution for ODEs

In this section, we briefly revisit the numerical methods of ODEs for those who are not familiar with it, including Leapfrog method and Runge-Kutta method that we used to develop the CNN blocks.

1.1. Leapfrog method

First, we reformulate the initial value problem of the following first-order differential equation

$$\frac{\mathrm{d}y}{\mathrm{d}x} = f(x,y), \quad y(x_0) = y_0. \tag{1}$$

Consider the second-degree Taylor polynomial approximation at x_n

$$y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{h^2}{2!}y''(x_n) + \frac{1}{3!}y'''(\xi_1), \tag{2}$$

$$y(x_{n-1}) = y(x_n) - hy'(x_n) + \frac{h^2}{2!}y''(x_n) - \frac{1}{3!}y'''(\xi_2), \tag{3}$$

where h is the step size, $x_n < \xi_1 < x_{n+1}$, $x_{n-1} < \xi_2 < x_n$. Sustract (3) from (2), drop the higher order term and make use of (1), we will obtain the Leapfrog scheme

$$y_{n+1} = y_{n-1} + 2hf(x_n, y_n). \tag{4}$$

The local truncation error is $\mathcal{O}(h^3)$, while the total error will be accumulated to $\mathcal{O}(h^2)$. Therefore, Leapfrog is a second-order method.

1.2. Runge-Kutta method

2-stage Runge-Kutta: We use the trapezoid formula to approximate y_{n+1}

$$y_{n+1} = y_n + \frac{h}{2}[f(x_n, y_n) + f(x_{n+1}, y_{n+1})]. \tag{5}$$

However, we have no knowledge of the exact value of y_{n+1}, thus we turn to the first-order approximation

$$y_{n+1} \approx y_n + hf(x_n, y_n). \tag{6}$$

These formulas can be rewritten into a 2-stage Runge-Kutta scheme

$$y_{n+1} = y_n + \frac{1}{2}(G_1 + G_2), \tag{7}$$

$$G_1 = hf(x_n, y_n), \tag{8}$$

$$G_2 = hf(x_n + h, y_n + G_1). \tag{9}$$

The local truncation error is $\mathcal{O}(h^3)$, and it is a second-order scheme which is also known as Heun's method.

Runge-Kutta family: Arbitray s (a positive integer) stage Runge-Kutta method takes the form

$$y_{n+1} = y_n + \sum_{i=1}^{s} \gamma_i G_i, \tag{10}$$

$$G_1 = hf(x_n, y_n),$$

$$G_i = hf(x_n + \alpha_i h, y_n + \sum_{j=1}^{i-1} \beta_{ij} G_j)$$

(we recommend [1] for further readings). As shown above, Runge-Kutta schemes make use of multiple steps to improve approximation accuracy. The coefficient α, β, γ can be determined by Taylor series. Here, we take $s = 3$ as an example, which is the same case we used to develop RK3-block. First we reformulate the 3-stage Runge-Kutta scheme

$$y_{n+1} = y_n + \gamma_1 G_1 + \gamma_2 G_2 + \gamma_3 G_3, \tag{11}$$

$$G_1 = hf(x_n, y_n), \tag{12}$$

$$G_2 = hf(x_n + \alpha_2 h, y_n + \beta_{21} G_1), \tag{13}$$

$$G_3 = hf(x_n + \alpha_3 h, y_n + \beta_{31} G_1 + \beta_{32} G_2). \tag{14}$$

[*] These authors contributed equally to this work.

For the sake of clarity, we omit the variable in the derivatives. Consider the Taylor expansion of $y(x)$ for G_2 at (x_n, y_n), drop the higher-order terms and substitue G_1 with formula (12), we have

$$\begin{aligned} G_2 &= h[f + (\alpha_2 h f_x + \beta_{21} h f f_y) \\ &\quad + \frac{1}{2}(\alpha_2^2 f_{xx} + \beta_{21}^2 f^2 f_{yy} + 2\alpha_2\beta_{21} f_{xy} f)h^2] \\ &= fh + (\alpha_2 f_x + \beta_{21} f_y f)h^2 \\ &\quad + \frac{1}{2}(\alpha_2^2 f_{xx} + \beta_{21}^2 f^2 f_{yy} + 2\alpha_2\beta_{21} f_{xy} f)h^3. \end{aligned} \tag{15}$$

Similarly, we obtain

$$\begin{aligned} G_3 = {} & fh + (\alpha_3 f_x + \beta_{31} f f_y + \beta_{32} f_y f)h^2 \\ & + \{\beta_{32} f_y(\alpha_2 f_x + \beta_{21} f_y f) + \frac{1}{2}[\alpha_3^2 f_{xx}^2 + (\beta_{31}^2 f^2 \\ & + \beta_{32}^2 f^2 + 2\beta_{31}\beta_{32} f^2) + 2\alpha_3(\beta_{31} f f_{xy} + \beta_{32} f f_{xy})]\}h^3. \end{aligned} \tag{16}$$

Note that we have dropped the $\mathcal{O}(h^4)$ term, then cosider the derivatives of y at x_n

$$\frac{\mathrm{d}y(x_n)}{\mathrm{d}x_n} = f(x_n, y_n) = f, \tag{17}$$

$$\frac{\mathrm{d}^2 y(x_n)}{\mathrm{d}x_n^2} = \frac{\mathrm{d}f(x_n, y_n)}{\mathrm{d}x_n} = f_x + f_y f, \tag{18}$$

$$\frac{\mathrm{d}^3 y(x_n)}{\mathrm{d}x_n^3} = f_{xx} + 2 f f_{xy} + f_{yy} f^2 + f_x f_y + f_y^2 f. \tag{19}$$

Use the Taylor expansion

$$y_{n+1} = y_n + y'(x_n)h + \frac{h^2}{2} y''(x_n) + \frac{h^3}{6} y'''(x_n) + \mathcal{O}(h^4), \tag{20}$$

and compare the coefficients of (11) and (20), we have the following equations

$$\begin{aligned} & \gamma_1 + \gamma_2 + \gamma_3 = 1, \\ & \gamma_2\beta_{21} + \gamma_3(\beta_{31} + \beta_{32}) = \frac{1}{2}, \\ & \gamma_2\alpha_2 + \gamma_3\alpha_3 = \frac{1}{2}, \\ & \gamma_2\alpha_2^2 + \gamma_3\alpha_3^2 = \frac{1}{3}, \\ & \gamma_2\beta_{21}^2 + \gamma_3(\beta_{31}^2 + \beta_{32}^2 + 2\beta_{31}\beta_{32}) = \frac{1}{3}, \\ & \alpha_2\beta_{32}\gamma_3 = \frac{1}{6}, \\ & \beta_{21}\beta_{32}\gamma_3 = \frac{1}{6}, \\ & \alpha_2\beta_{21}\gamma_2 + \alpha_3(\beta_{31} + \beta_{32})\gamma_3 = \frac{1}{3}. \end{aligned} \tag{21}$$

The solution is not unique. In our designs, we take

$$\begin{aligned} & \gamma_1 = \frac{1}{6}, \gamma_2 = \frac{2}{3}, \gamma_3 = \frac{1}{6}, \\ & \beta_{21} = \frac{1}{2}, \beta_{31} = -1, \beta_{32} = 2, \\ & \alpha_2 = \frac{1}{2}, \alpha_3 = 1, \end{aligned} \tag{22}$$

which is also known as Kutta's method. Since we have dropped the term with order higher than $\mathcal{O}(h^4)$, the local truncation error is $\mathcal{O}(h^4)$, i.e., a third-order method.

2. Dynamical system

In this section, we briefly revisit the concept of dynamical system used in this paper. Here we adopt the definition in [2]. Generally speaking, a dynamical system is a monoid $\mathcal{G}$ acting on a set M. More precisely, there is a map

$$\Phi : \mathcal{G} \times M \to M \tag{23}$$

$$(g, x) \mapsto \Phi_g(x) \tag{24}$$

for $\forall g, t \in \mathcal{G}$ and the identity element $e \in \mathcal{G}$, satisfies

$$\Phi_g \circ \Phi_t = \Phi_{g \circ t}, \quad \Phi_e = \mathcal{I}. \tag{25}$$

In the semantics of SISR, $\Phi_t(x)$ can be granted as a map from the input sample x to the output high-resolution image through time t, since we regard t as the element of $\mathcal{G}$. When $t = 0$, $\Phi_0(x) = x$ is an identity mapping, then we would like to approach $\Phi_t(x)$ through an ODE (e.g. formula (1)) and interprete it as a CNN. In order to achieve this process, we design finer CNN blocks and choose appropriate t, which is corresponded to block numbers (detailed in Table 1).

Table 1. Residual blocks with their correpsonding numerical methods. This table lists the performance on DIV2K validation set and shows the impact of different numerical schemes. For a fair comparsion, we set the number of multiply-accumulate operations (MAC) and parameters of our methods to be the same as EDSR-baselines.

Scale	Method	PSNR	Numerical Method	G
×2	EDSR	34.61	Forward Euler	Original
×2	LF	**34.67**	Leapfrog	v2
×2	RK2	34.63	Second-order Runge-Kuta	v3
×3	EDSR	30.92	Forward Euler	Original
×3	LF	**30.98**	Leapfrog	v2
×3	RK2	30.94	Second-order Runge-Kuta	v3
×4	EDSR	28.95	Forward Euler	Original
×4	LF	**29.02**	Leapfrog	v2
×4	RK2	28.99	Second-order Runge-Kuta	v3

References

[1] R. L. Burden and J. D. Faires. Numerical analysis. *Cengage Learning*, 9, 2010.

[2] G. Teschl. *Ordinary differential equations and dynamical systems*, volume 140. American Mathematical Soc., 2012.