Fourier Series Iteration of a Neural Network

July 14, 2020

1 Notation

1.1 Fourier Series

We write the Fourier series expansion $\hat{f} : \mathbb{R} \to \mathbb{R}$ of a function $f : \mathbb{R} \to \mathbb{R}$ as

$$\hat{f}(x) = \frac{1}{\sqrt{2\pi}} \sum_{n=-\infty}^{\infty} c_n e^{inx}, \tag{1}$$

where

$$c_n = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-inx} dx. \tag{2}$$

When expanding an m-dimensional function, we use the following generalization.

$$\hat{f}(x_1, \ldots, x_m) = \frac{1}{(2\pi)^{m/2}} \sum_{n_1=-\infty}^{\infty} \cdots \sum_{n_m=-\infty}^{\infty} c_{n_1,\ldots,n_m} e^{i(n_1 x_1 + \cdots + n_m x_m)} \tag{3}$$

where

$$c_{n_1,\ldots,n_m} = \frac{1}{(2\pi)^{m/2}} \int_{\mathbb{R}^m} f(x_1, \ldots, x_m) e^{-i(n_1 x_1 + \cdots + n_m x_m)} dx. \tag{4}$$

If we wish to truncate a Fourier series to $N_1, \ldots, N_m$ terms, we use the truncated Fourier series denoted by $\hat{f}_{N_1,\ldots,N_M}$ which is defined by

$$\hat{f}_{N_1,\ldots,N_m}(x_1, \ldots, x_m) = \frac{1}{(2\pi)^{m/2}} \sum_{n_1=-N_1}^{N_1} \cdots \sum_{n_m=-N_M}^{N_M} c_{n_1,\ldots,n_m} e^{i(n_1 x_1 + \cdots + n_m x_m)}. \tag{5}$$

1.2 Neural Network

We represent an arbitrary L-layer neural network by the function

$$y(x) = \sigma_L(W_L \cdots \sigma_1(W_1 x) \cdots). \tag{6}$$

where W_i is the weight matrix for the i-th layer and σ_i is the activation function for the i-th layer. We can rewrite this iteratively.

$$x_1(x) = x \tag{7}$$
$$x_{i+1}(x) = \sigma_i(W_i x_i(x)) \tag{8}$$

so that $y = x_{L+1}$.

2 Approximation

We wish to construct a Fourier approximation to a neural network layer by layer. For computational feasibility, we will assume that our Fourier series approximation is truncated to N terms. That is, the i-th layer of a neural network is a function of the input x defined by

$$\hat{f}_1(x) = \hat{x}_1(x) = \sum_{n=-N}^{-1} i\frac{(-1)^n}{n} + \sum_{n=1}^{N} i\frac{(-1)^n}{n}, \tag{9}$$

$$\hat{f}_i(x) = \hat{x}_i(x) = \widehat{(\sigma_{i-1} \circ w_{i-1} \circ \hat{f}_{i-1})}(x). \tag{10}$$

We can compute $\hat{f}_i(x)$ using quadrature or a discrete Fourier transform over many gridpoints of its definition. The algorithm to perform the approximation iteratively is as follows:

Algorithm 1 Iterative Approximation Algorithm

procedure FOURIER APPROXIMATION$((\sigma_1, W_1), \ldots, (\sigma_L, W_L))$
  $\hat{f}_1 \leftarrow \hat{x}_1 = \sum_{n=-N}^{-1} i\frac{(-1)^n}{n} + \sum_{n=1}^{N} i\frac{(-1)^n}{n}$
  for $i = 2, \ldots, L+1$ **do**
    for $n = -N, \ldots, N$ **do**
      $c_{i,n} \leftarrow \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \sigma_{i-1}(W_{i-1} f_{i-1}(x)) e^{-inx} dx$ ▷ Use Quadrature or DFT
    $\hat{f}_i(x) \leftarrow \frac{1}{\sqrt{2\pi}} \sum_{n=-\infty}^{\infty} c_{i,n} e^{inx}$
  $y \leftarrow f_{L+1}(x)$

This algorithm operates on a single input dimension but can easily be extend to multidimensional approximations.