Machine Learning I: Supervised Methods

B. Keith Jenkins

Announcements

- Course Project Assignment has been posted on D2L
  - Includes descriptions of 3 datasets
  - Requirements
  - Tips to help you
  - Grading criteria
- There will be 2 more homeworks (HW7, HW8)
- No Slido poll questions today

* - Added post-lecture:
  - Equation numbers for all $w_{pq}^{(\ell)}$;
  - Last comment on p. 12.

Reading

- Bishop 9.1
  - K-means clustering (we will use this with Radial Basis Function ANNs)

Today's lecture

- Multiple layer feedforward ANNs (part 2)
  - Feedforward ANNs as universal function approximators
    - Proof by construction (D=1)
    - Summary and comments

Feed forward ANN's: fundamental capabilities and limitations

- Classification
- Regression
- Function approximation.

ANN, for any input $\underline{x}$, will give outputs: $\hat{y}_i(\underline{x})$, $i=1,\cdots,d_L$.

> Regression: outputs $\hat{y}_1(\underline{x}), \hat{y}_2(\underline{x}), \cdots, \hat{y}_{d_L}(\underline{x})$.

Like "approximating" some unknown function $y_i(\underline{x})$

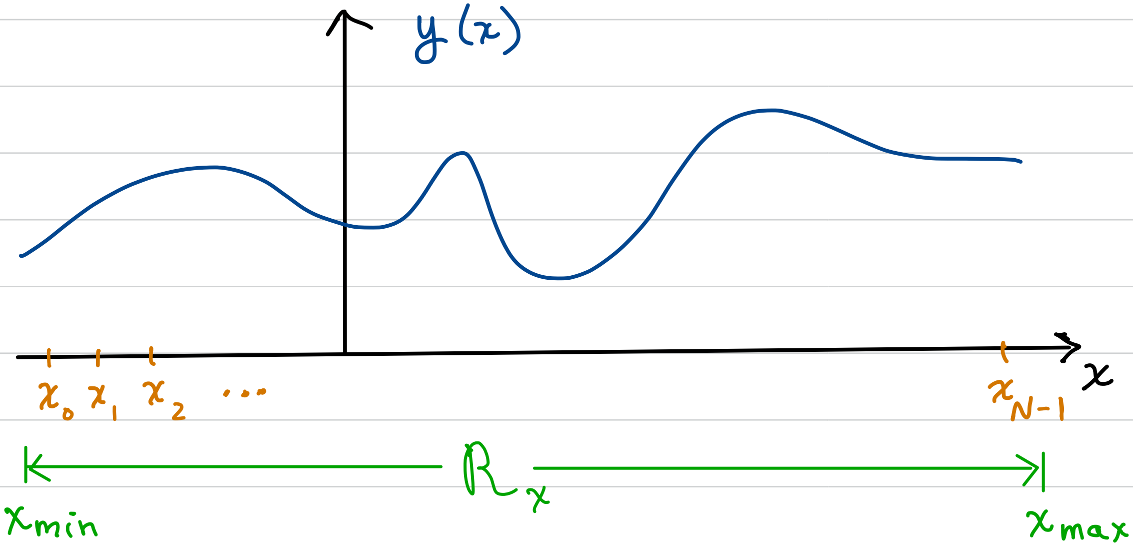
> Classification: (i) approx. binary-valued (unknown) fcns. $y_i(\underline{x})$
(ii) approx. discriminant fcns. $g_k(\underline{x})$ or $g_{kj}(\underline{x})$.

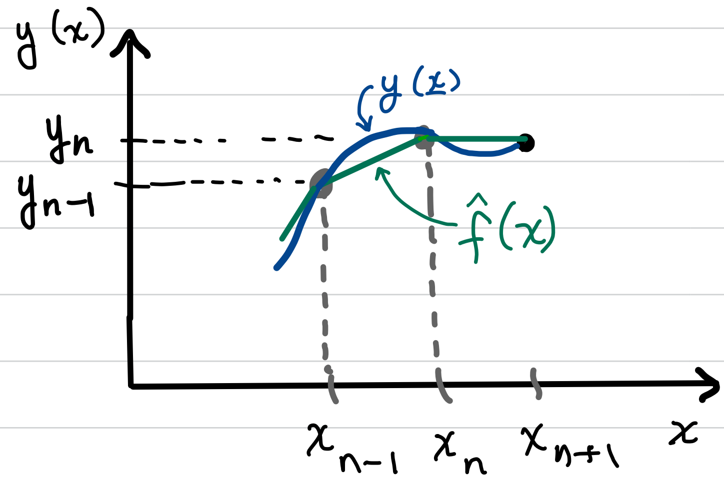
For ML (regr. & class'n) the fcn. we are approximating is <u>unknown.</u>

<u>Question:</u> how general of a function can a multilayer feedforward ANN approximate, to (ideally) any degree of accuracy?

Theoretical example of universal function approximation using ReLU

Problem: approximate any function $y(x)$ using a 2-layer ANN, over the region R_x.
Assume a 1D problem, ReLU for the hidden layer.





Use a piecewise linear approximation.
Choose data points (grid points) (x_i, y_i), $y_i = y(x_i)$, $i = 0, 1, 2, \cdots, N-1$.

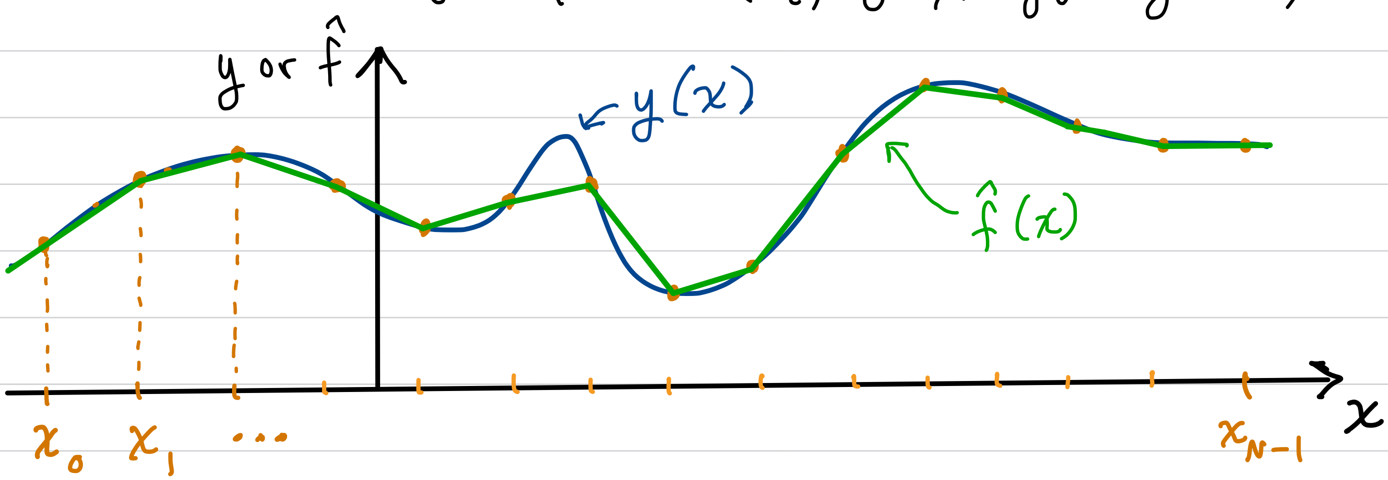
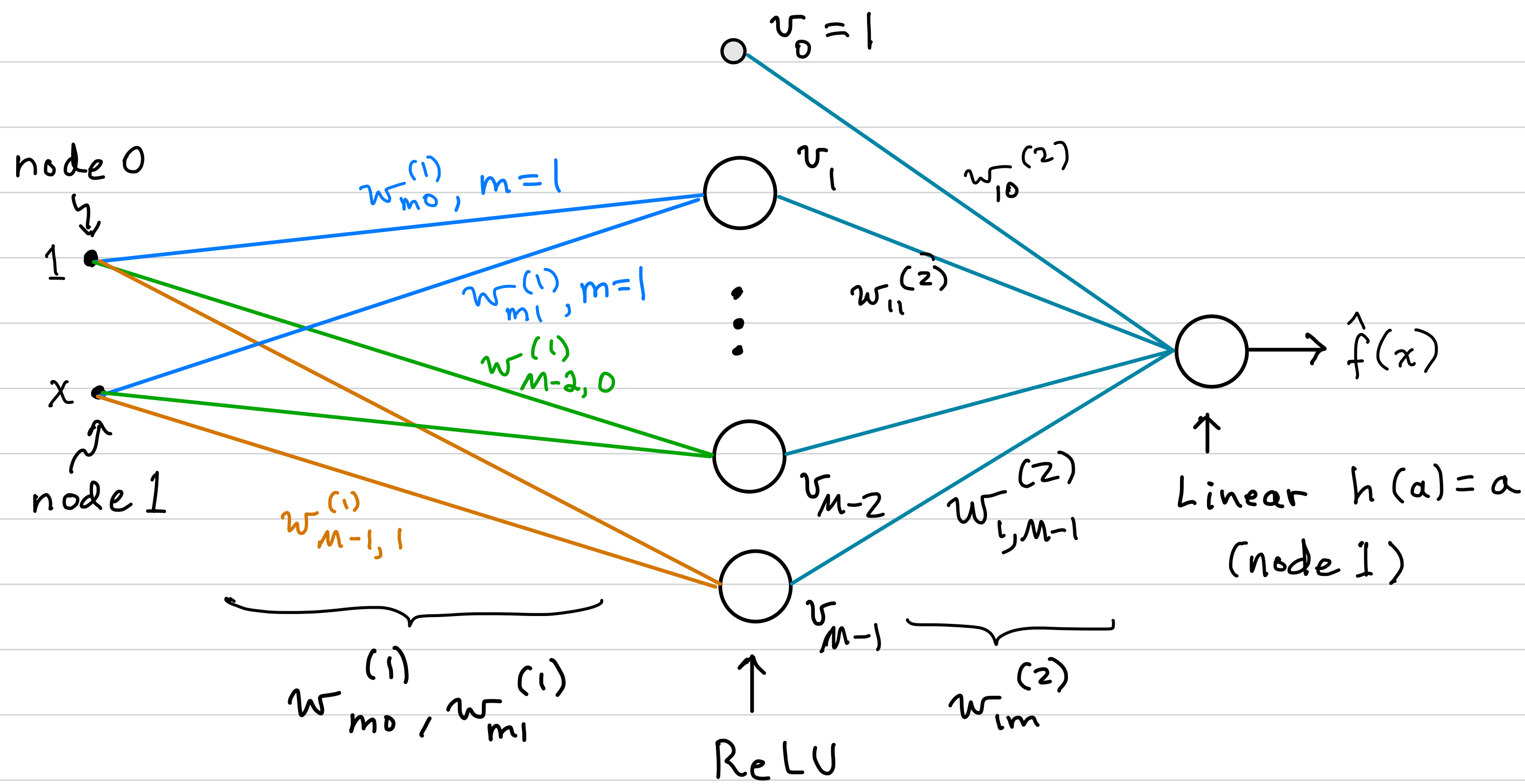


Fig. 1



$\Rightarrow$ hidden unit outputs $v_m(x_n)$, $m = 1, \cdots, M-1$; $M-1 = \#$ hidden units.
Hidden bias node output $v_0 = 1$.

Let $M = N = \#$ "data points" (or "grid points")

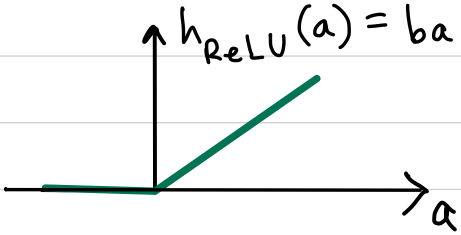
Goal: find weights that will give $\hat{f}(x)$ as a piecewise-linear approximation to $y(x)$

2nd layer $\qquad\qquad$ 1st layer

activation fcn. of v_m

$$\hat{f}(x) = w_{10}^{(2)} v_0(x) + \sum_{m=1}^{M-1} w_{1m}^{(2)} v_m(x), \qquad v_m(x) = h_{ReLU}\left(w_{m1}^{(1)} x + w_{m0}^{(1)}\right)$$

$$h_{ReLU}(a) = ba$$



$$v_m(x) = \max\{0,\ b\,(w_{m1}^{(1)} x + w_{m0}^{(1)})\}$$

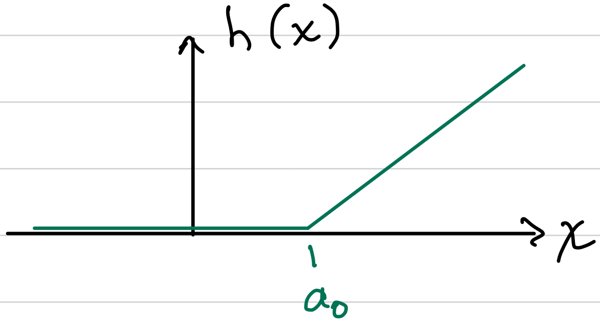
(1)

Let ReLU parameter $b = 1$ ($\Rightarrow$ slope 1).
Let $w_{m1}^{(1)} = 1 \quad \forall m$

$$\Rightarrow v_m(x) = \max\{0,\ x + w_{m0}^{(1)}\}$$

$$\hat{f}(x) = w_{10}^{(2)} v_0(x) + \sum_{m=1}^{M-1} w_{1m}^{(2)} \max\{0,\ x + w_{m0}^{(1)}\}$$

We will want:



How to express using h_{ReLU}?

$$\rightarrow h(x) = h_{ReLU}(x - a_0)$$
$$= \max\{0,\ x - a_0\}$$

Order the N data points so that: $x_0 < x_1 < x_2 < \cdots < x_{N-1}$
in which duplicate data points are omitted.

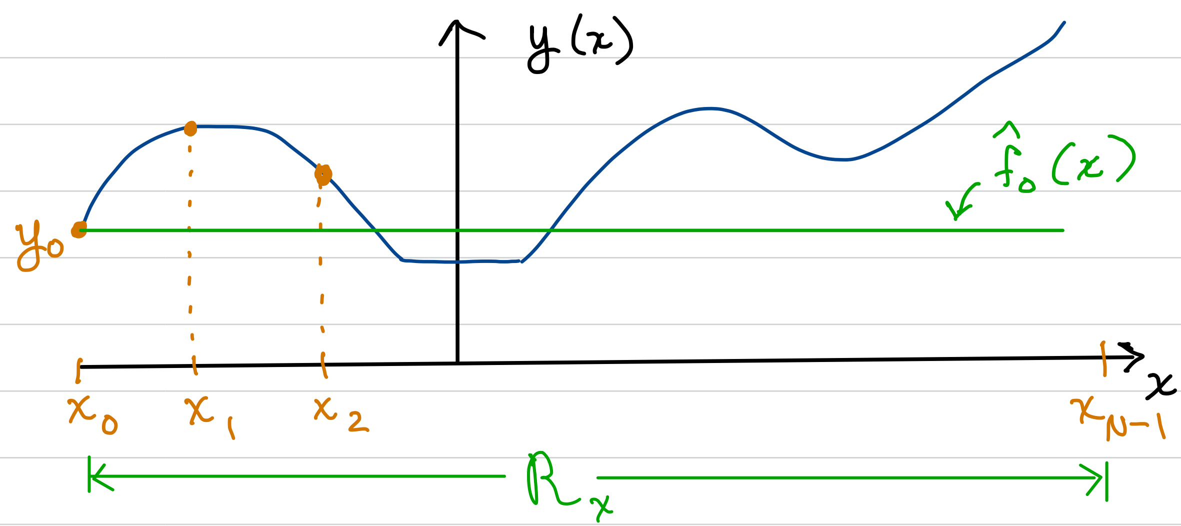
For simplicity, assume $x_0 = x_{min}$ and $x_{N-1} = x_{max}$ (we have data points at edges of region R_x).

Comment: this proof also applies (with minor modifications) if $x_{min} < x_0$ and $x_{N-1} < x_{max}$.

Let $\hat{f}_{n-1}(x)$ be the function approximation based on data points $x_0, x_1, \ldots, x_{n-1}$.

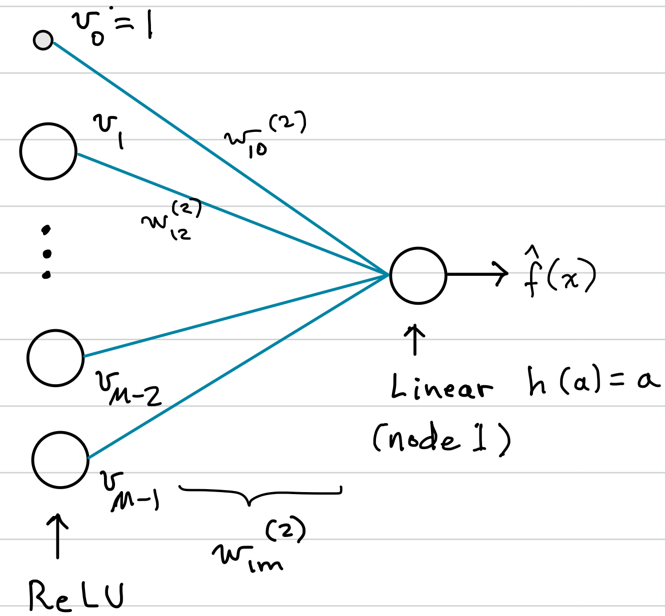
$\Rightarrow$ $\hat{f}_0(x)$ uses only data point (x_0, y_0).

(2) Let: $\hat{f}_0(x) = y_0$.



In the ANN (Fig. 1), this can be done by:

(3) [$w_{10}^{(2)} = y_0$ $\quad\Rightarrow \hat{f}_0(x) = y_0$



For $\hat{f}_1(x)$ use data $(x_0, y_0), (x_1, y_1)$.

→ Or, use $\hat{f}_0(x)$ and (x_1, y_1).

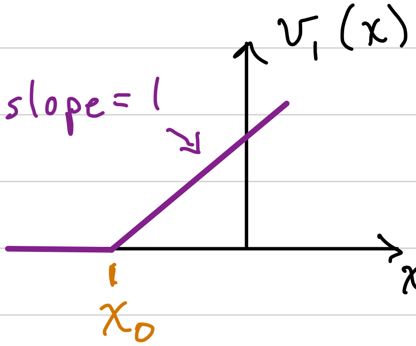
What is $\hat{f}_0(x_1)$? y_0

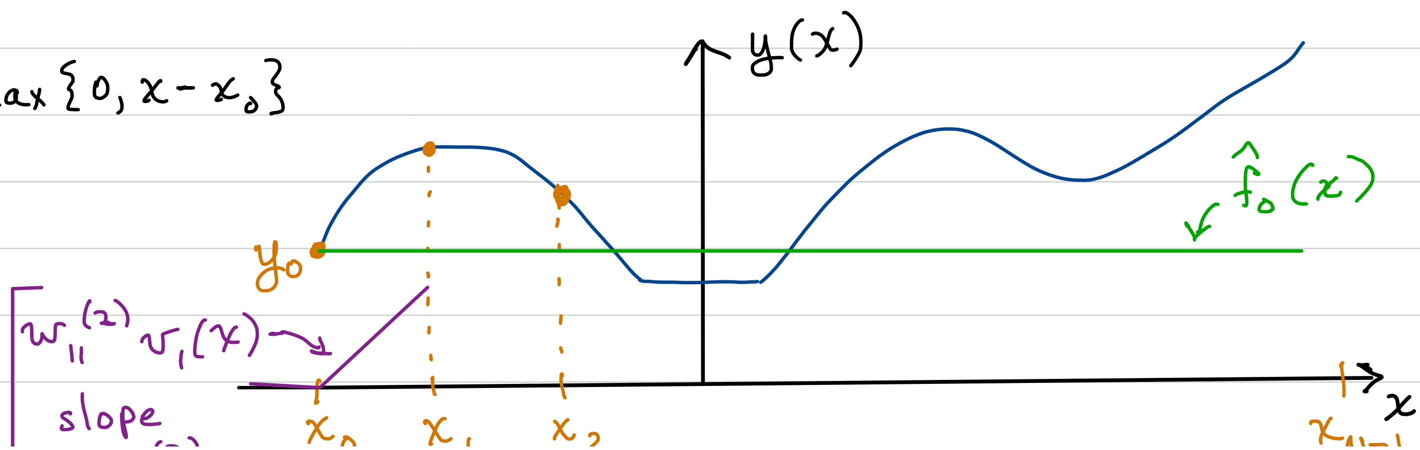
$\Rightarrow \hat{f}_0(x_1) \neq y_1$ in general

Use $v_1(x)$ to correct $\hat{f}_0(x_1)$, so that $\hat{f}_1(x_1) = y_1$.

Place "hinge" of the ReLU for v_1 at x_0:

$v_1(x) = \max\{0, x - x_0\}$





$$(1) \Rightarrow \hat{f}_1(x) = \overbrace{\underbrace{w_{10}^{(2)} v_0(x)}_{y_0}}^{\hat{f}_0(x)} + w_{11}^{(2)} \max\{0,\ x + \underbrace{w_{10}^{(1)}}\}$$

(4) $\left[\ \text{Choose } w_{10}^{(1)} = -x_0\right.$

$$\hat{f}_1(x) = \hat{f}_0(x) + w_{11}^{(2)} \max\{0,\ x - x_0\}$$

$$\left[\hat{f}_1(x) = \hat{f}_0(x) + w_{11}^{(2)} v_1(x)\right. \qquad = \hat{f}(x) \text{ if } v_m(x) = 0,\ m > 1.$$

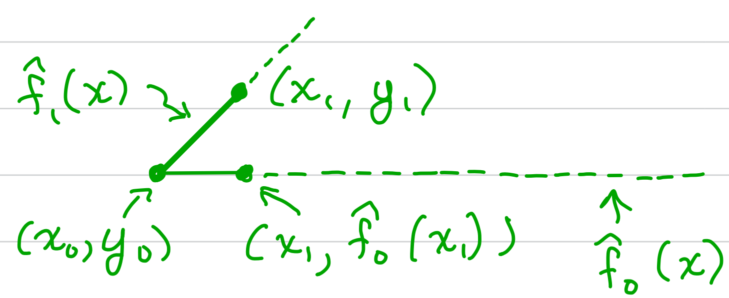
What slope $w_{11}^{(2)}$ will make $\hat{f}_1(x_1) = y_1$?

$$\hat{f}_1(x) - \hat{f}_0(x) = w_{11}^{(2)} v_1(x) \qquad \text{slope} = w_{11}^{(2)} = \frac{y_1 - y_0}{x_1 - x_0}$$

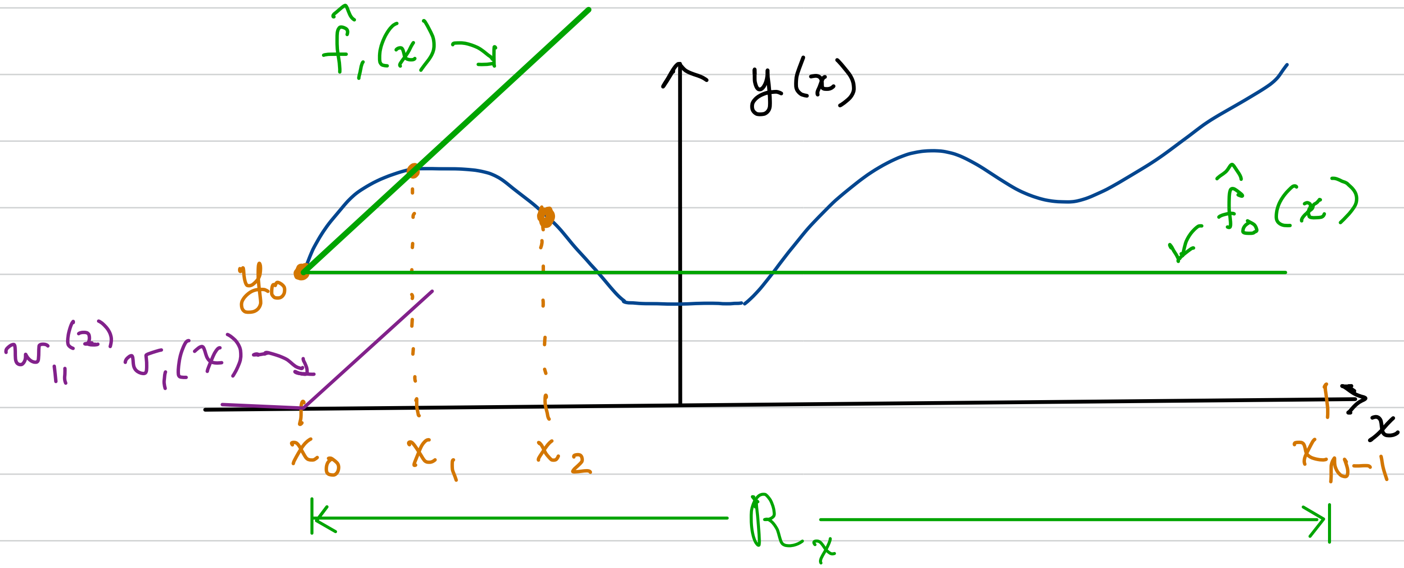
$$\hat{f}_1(x_1) - \hat{f}_0(x_1) = w_{11}^{(2)} v_1(x_1)$$

$$y_1 - y_0 = w_{11}^{(2)}(x_1 - x_0) \quad \text{because } x_1 > x_0.$$

(5) $\left[\ w_{11}^{(2)} = \dfrac{y_1 - \hat{f}_0(x_1)}{x_1 - x_0}\right.$



$$\Rightarrow \hat{f}_1(x_1) = y_1 \Rightarrow 0 \text{ error at } x_1.$$



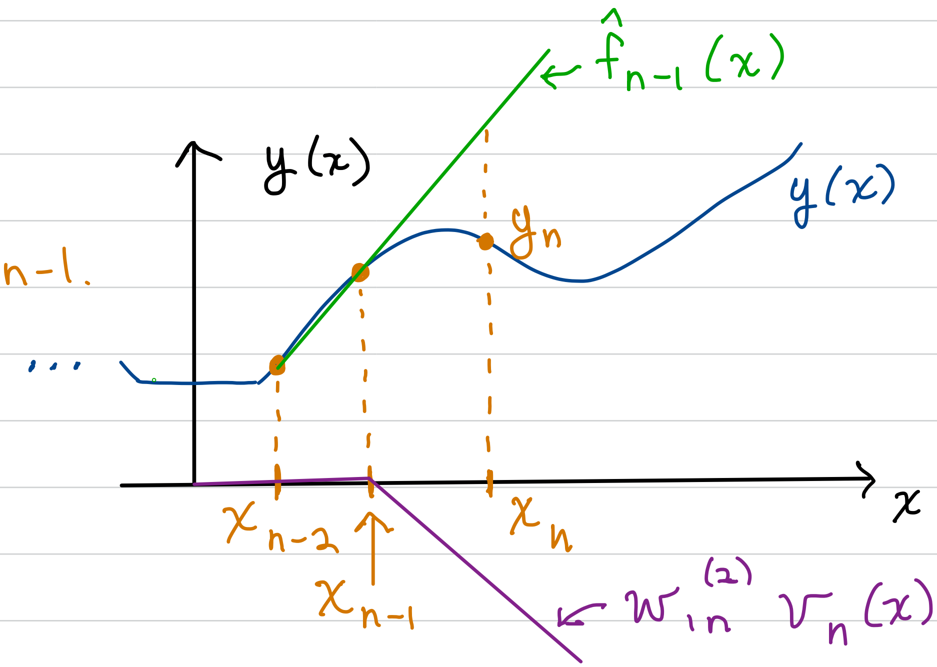
Now consider $n^{\underline{th}}$ segment (for $n \geq 2$):

Given:

$\hat{f}_{n-1}(x)$

x_n, y_n

x_m, y_m, $m \leq n-1$.



$$(1) \Rightarrow \quad \hat{f}(x) = \overbrace{\underbrace{w_{10}^{(2)}}_{y_0} v_0(x)}^{\hat{f}_0(x)} + \sum_{m=1}^{M-1} w_{1m}^{(2)} \max\{0,\ x + w_{m0}^{(1)}\}$$

(6) [Choose $w_{m0}^{(1)} = -x_{m-1}$, $\quad \forall\, m > 1.$

$$\hat{f}(x) = \underbrace{\hat{f}_0(x) + \sum_{m=1}^{n-1} w_{1m}^{(2)} \max\{0, x - x_{m-1}\}}_{\triangleq\ \hat{f}_{n-1}(x)} + \underbrace{\sum_{m=n+1}^{M-1} w_{1m}^{(2)} \max\{0, x - x_{m-1}\}}_{=0 \text{ in region } x_{n-1} \le x \le x_n \text{ because } x \le x_n}$$

$$+ \, w_{1n}^{(2)} \max\{0, x - x_{n-1}\}$$

This is the estimate based on (x_i, y_i), $i = 0, 1, \cdots, n$.

[$\hat{f}_n(x) = \hat{f}_{n-1}(x) + w_{1n}^{(2)} v_n(x)$, $\quad v_n(x) = \max\{0, x - x_{n-1}\}$

Assume we have $\hat{f}_{n-1}(x_{n-1}) = y_{n-1}$ $\Rightarrow$ 0 error at x_{n-1}.

Generally $\hat{f}_{n-1}(x_n) \neq y_n$.

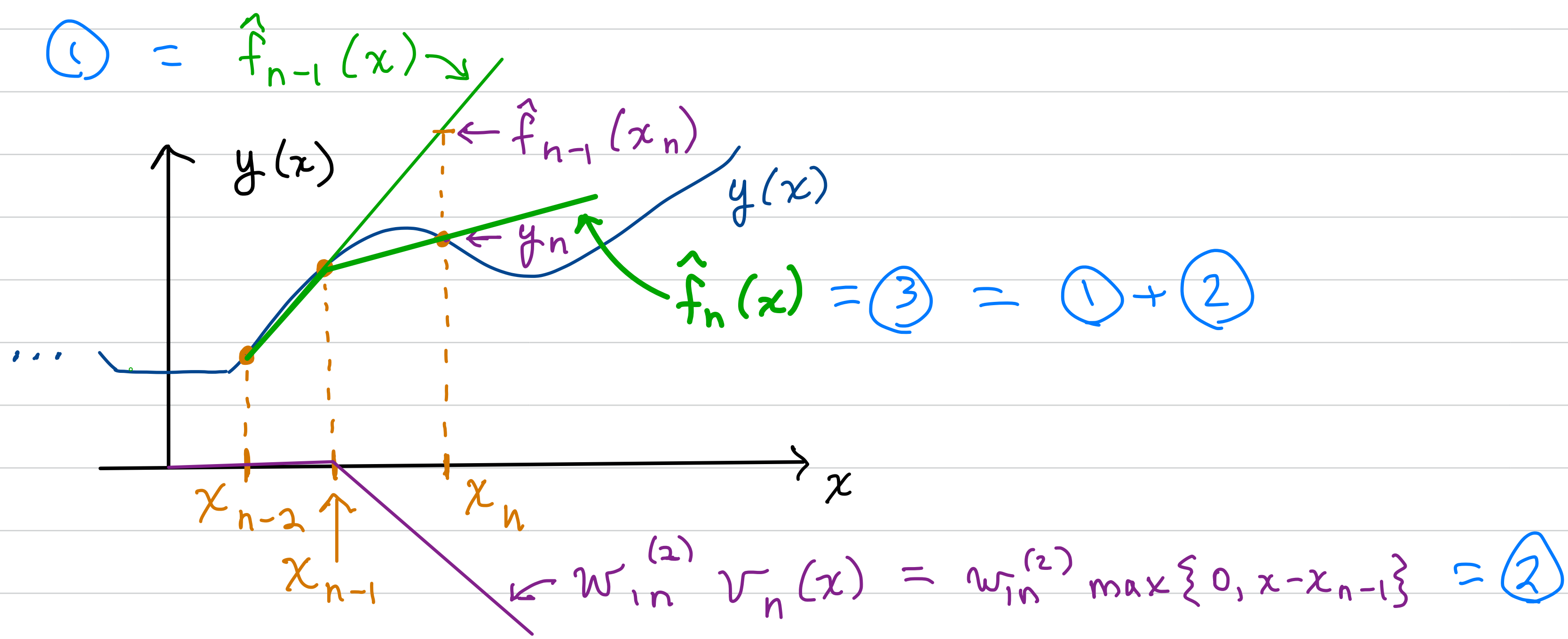
$$\hat{f}_n(x) = \hat{f}_{n-1}(x) + \underbrace{w_{1n}^{(2)} \max\{0, x - x_{n-1}\}}_{\text{Choose to make } \hat{f}_n(x_n) = y_n}$$

$$\hat{f}_n(x_n) = \hat{f}_{n-1}(x_n) + w_{1n}^{(2)}(x_n - x_{n-1}) \quad \text{because } x_n > x_{n-1}.$$

$$\hat{f}_n(x_n) - \hat{f}_{n-1}(x_n) = w_{1n}^{(2)}(x_n - x_{n-1})$$

$$y_n - \hat{f}_{n-1}(x_n) = w_{1n}^{(2)}(x_n - x_{n-1})$$

(7) $$w_{1n}^{(2)} = \frac{y_n - \hat{f}_{n-1}(x_n)}{x_n - x_{n-1}}, \quad n \geq 2.$$



By induction, error can be made 0 at every data point, and $\hat{f}(x)$ gives linear approximation to $y(x)$ between neighboring data points.

Summary

- Feedforward ANN with 1 hidden layer (2 layers of weights).
- Weights $w_{m1}^{(1)} = 1 \quad \forall m$ (all weights from input x in first layer)
- Hidden units use ReLU activation functions $h(a) = \max\{0, a\}$ (unit slope)
- Weights $w_{m0}^{(1)} = -x_{m-1}$ give offset of each hidden-unit ReLU

$$v_m(x) = \max\{0, x + w_{m0}^{(1)}\} = \max\{0, x - x_{m-1}\}$$

(x_{m-1} is hinge point of $v_m(x)$ ReLU function.)

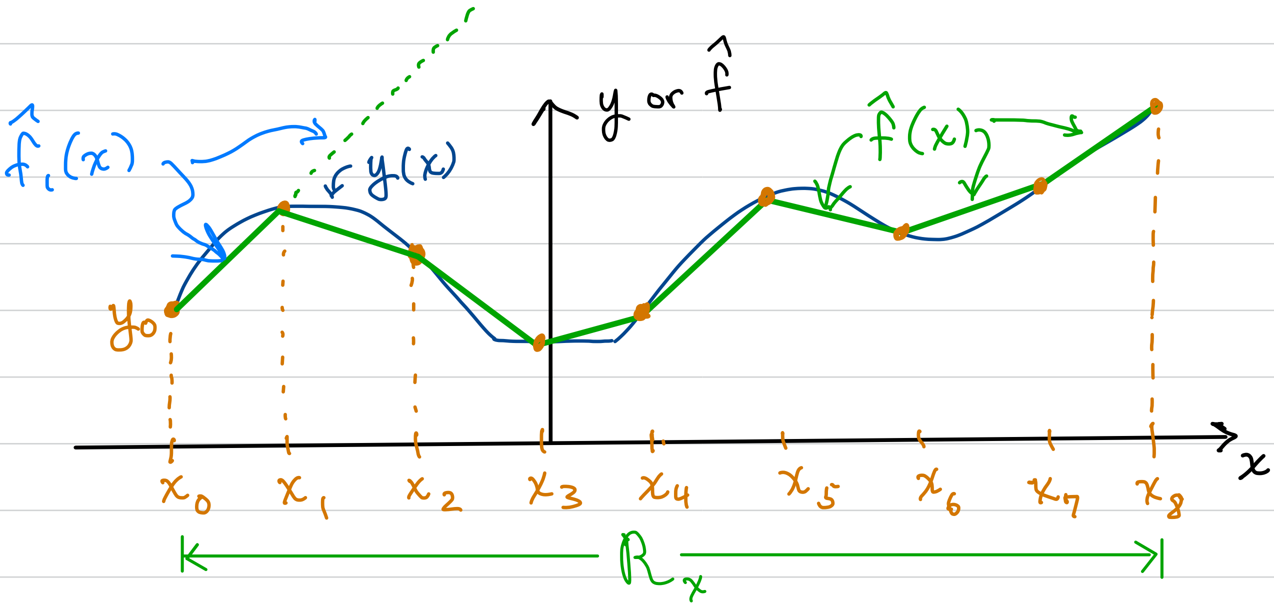
- Second layer weights $w_{1n}^{(2)}$ give slope of ReLU of $v_n(x)$ so that

$$\hat{f}_n(x) = \underbrace{\hat{f}_0(x) + \left(\sum_{m=1}^{n-1} w_{1m}^{(2)} v_m(x)\right)}_{\text{Sum of contributions from all } v_m(x),\ m<n.} + w_{1n}^{(2)} v_n(x)$$

gives error-free estimate at x_n: $\hat{f}_n(x_n) = y_n$.

* ○ All weight values are given in Eqs. (1), (3), (4)–(7).

- Example result ($N=9$ data points)



Comments

1. $\hat{f}(x)$ is a piecewise-linear approximation of $y(x)$.

2. By taking N $(=M)$ sufficiently large, error of approximation

$$\mathcal{E} = \frac{1}{x_{max} - x_{min}} \int_{R_x} \left(\hat{f}(x) - y(x) \right)^2 dx$$

can be made arbitrarily small, for essentially any realistic function function $y(x)$.

3. Thus, this ANN shows that a (1D) universal function approximator is possible with a 2-layer feedforward ANN.

4. Can be extended to higher dimensional inputs $D \geq 2$.
(D > 2 most easily proven using Fourier-series approach instead of piecewise-linear).