Pushforward / Jvp rule for function broadcasting

(Neural Network nonlinear activation)

$$f(\underline{x}) = \underline{y} \qquad f\left(\begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}\right) = \begin{bmatrix} \sigma(x_1) \\ \sigma(x_2) \\ \vdots \\ \sigma(x_N) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

$\underline{x} \in \mathbb{R}^N$

$\underline{y} \in \mathbb{R}^N$

$f: \mathbb{R}^N \to \mathbb{R}^N$

$\sigma: \mathbb{R} \to \mathbb{R}$

$= \sigma.(\underline{x})$

in MATLAB or Julia

(autobroadcasting in NumPy)

task: propagate $\dot{\underline{x}} \in \mathbb{R}^N$ to $\dot{\underline{y}} \in \mathbb{R}^N$

$$\dot{\underline{y}} = \frac{\partial f}{\partial \underline{x}} \dot{\underline{x}} = \underbrace{\begin{bmatrix} \overset{\sigma'(x_1)}{\frac{\partial f_1}{\partial x_1}} & \overset{=0}{\frac{\partial f_1}{\partial x_2}} & \cdots & \\ \overset{0}{\frac{\partial f_2}{\partial x_1}} & \overset{\sigma'(x_2)}{\frac{\partial f_2}{\partial x_2}} & & \\ \vdots & & \ddots & \\ & & & \underset{\sigma'(x_N)}{\frac{\partial f_N}{\partial x_N}} \end{bmatrix}}_{\text{diagonal Jacobian matrix}} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_N \end{bmatrix}$$

because there is no interaction between the dimensions in $\underline{x}$

$$\dot{\underline{y}} = \begin{bmatrix} \sigma'(x_1) & & 0 \\ & \sigma'(x_2) & \\ 0 & \ddots & \\ & & \sigma'(x_N) \end{bmatrix} \dot{\underline{x}}$$

$$= \text{diag}(\sigma'.(\underline{x}))\, \dot{\underline{x}}$$

$$= \sigma'.(\underline{x}) \odot \dot{\underline{x}}$$

the derivative broadcastedly applied to the primal inputs

element-wise multiplication

Full pushforward rule

$$\boxed{\mathcal{F}\left(f, (\underline{x},), (\dot{\underline{x}},)\right) = \left(\underbrace{(\sigma.(\underline{x}),)}_{\underline{y}}, \underbrace{(\sigma'.(\underline{x}) \odot \dot{\underline{x}},)}_{\dot{\underline{y}}}\right)}$$

e.g.: sigmoid: $\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$

tanh: $\sigma'(x) = 1 - \sigma^2(x)$