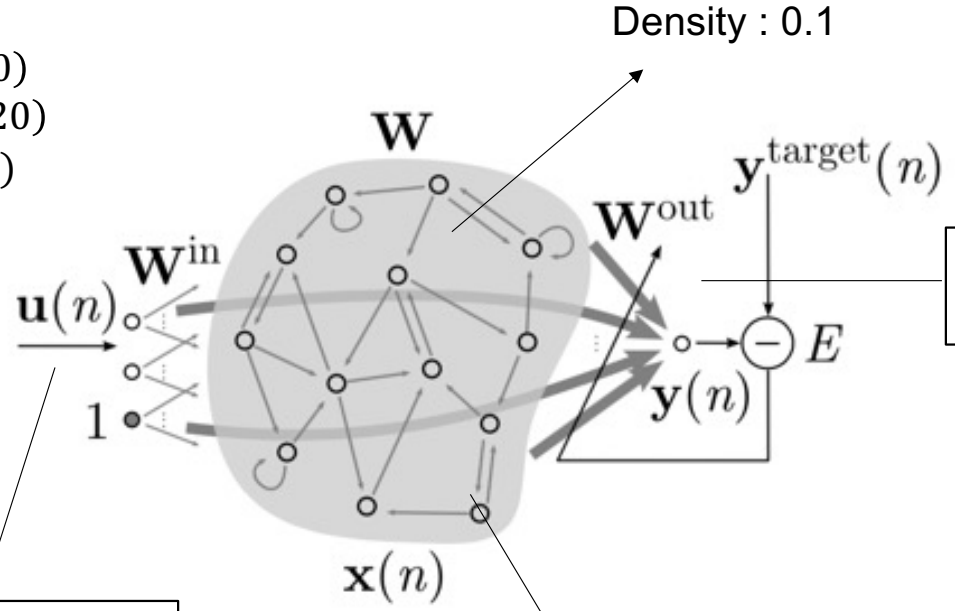
具体的なとモデル

W^{in} : $(1, 20)$
W : $(20,20)$
W^{out}: $(20,1)$



Output 層

$$\boldsymbol{y}(n) = W^{out}\boldsymbol{x}(n)$$

Input 層

$$\boldsymbol{x}(n) = W^{in}\boldsymbol{u}(n)$$

リザーバー 層

$$\boldsymbol{x}(n+1) = (1-\alpha)\boldsymbol{x}(n) + \alpha f(W^{in}\boldsymbol{u}(n+1) + W\boldsymbol{x}(n))$$

W^{out}の計算　　$W^{out} = (w_0, w_1, \cdots, w_{19})$

$$x = (x_0, x_1, \cdots, x_{19})^{\mathrm{T}}$$

$$\begin{aligned} y_{pred} &= x_0 w_0 + x_1 w_1 + \cdots + x_{19} w_{19} \\ &= W^{out}\boldsymbol{x} \end{aligned}$$

コスト関数　　$J(w_i) = \left(y_{pred} - y_{true}\right)^2$

$$\frac{\partial J}{\partial w_i} = 2x_i(y_{pred} - y_{true})$$

$$\frac{\partial J}{\partial W^{out}} = 2\,(W^{out}\boldsymbol{x} - y_{true})\boldsymbol{x}^{\mathrm{T}} = 0$$

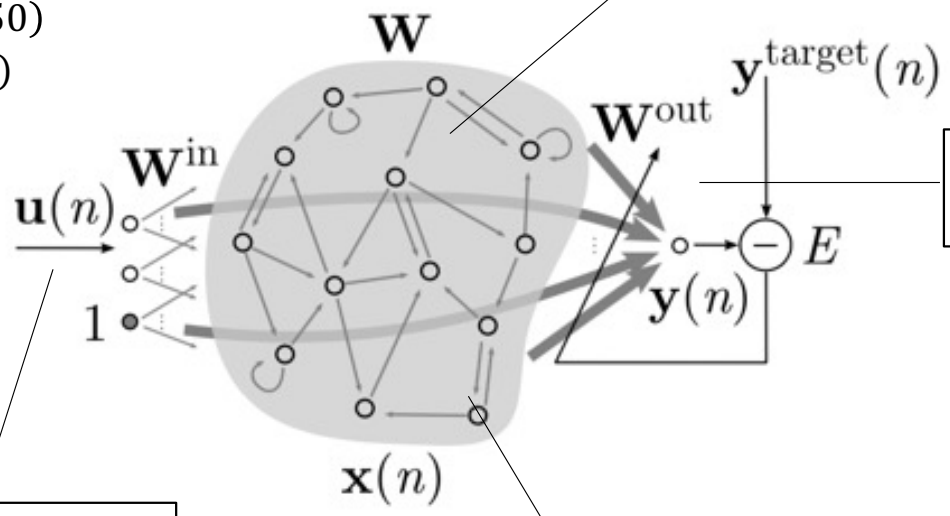
$$W^{out}\left(\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}\right) = y_{true}\boldsymbol{x}^{\mathrm{T}}$$

$$W^{out} = y_{true}\boldsymbol{x}^{\mathrm{T}}\left(\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}\right)^{-1}$$

RNNからESNを理解する

W^{in} : $(1, 50)$
W : $(50,50)$
W^{out}: $(50,1)$

Density : 0.1



Output 層

$$\boldsymbol{y}(n) = W^{out}\boldsymbol{x}(n)$$

Input 層

$$\boldsymbol{x}(n) = W^{in}\boldsymbol{u}(n)$$

リザーバー 層

$$\boldsymbol{x}(n+1) = (1-\alpha)\boldsymbol{x}(n) + \alpha f(W^{in}\boldsymbol{u}(n+1) + W\boldsymbol{x}(n))$$

少なくとも今回の例では，リザーバー層の結合・重みの選択にほとんど依存しない
→ リザーバーの結合・重みってそんなになんでもいいの？？

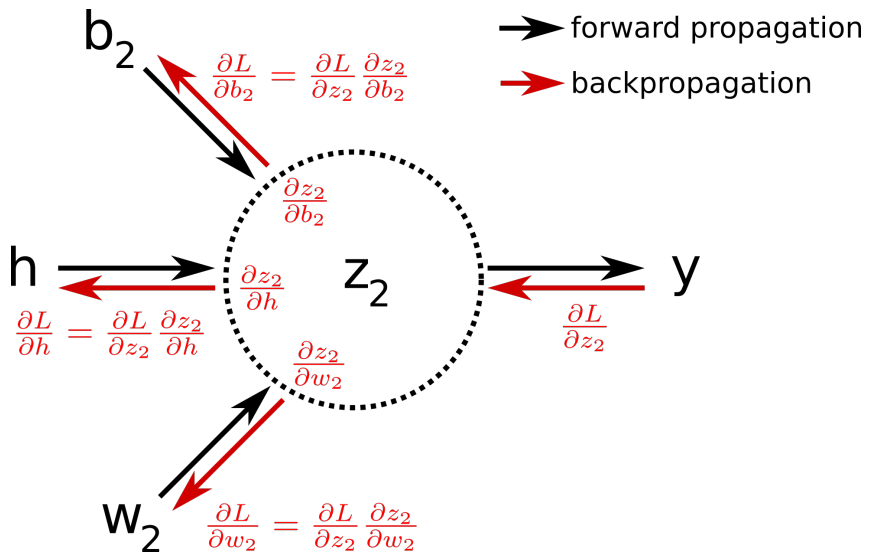
RNNからESNを理解する

Back Propagationの計算を考えると，出力に近い側の重みが大きく更新される　＝　重要

→ 逆に入力に近い側では，勾配が小さくなり更新が小さい？

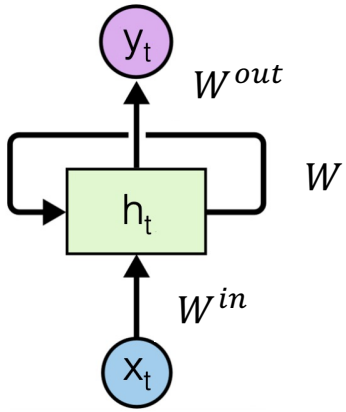
→ 極端な話，問題によっては出力層のみの重みの更新だけで十分　＝　ESN

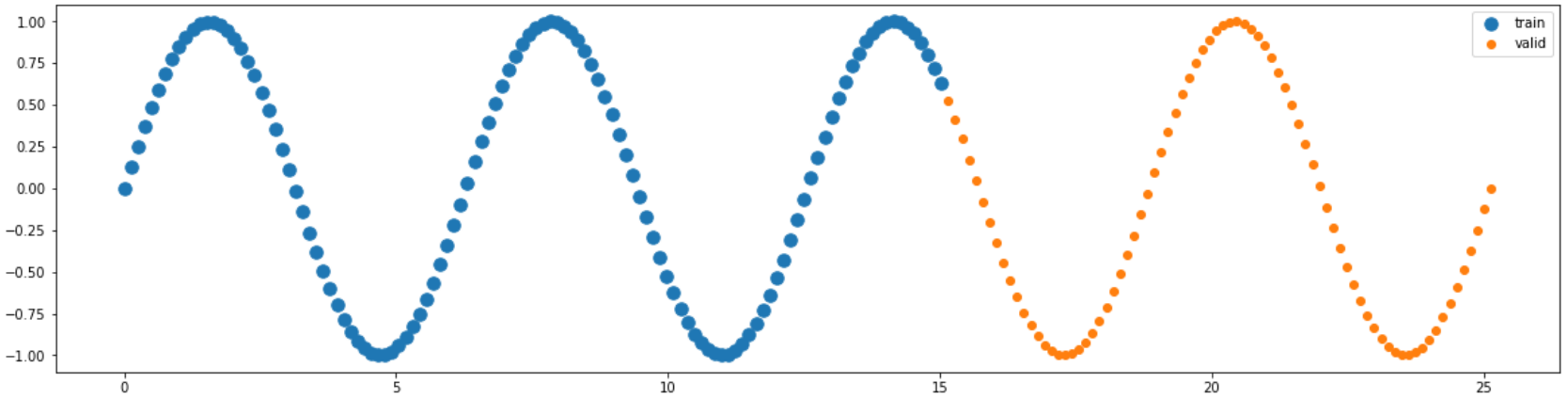
特に，tanhなど，勾配が1を超えずこの傾向が顕著　→ ここからtanhがよく利用される？



RNNからESNを理解する：実験

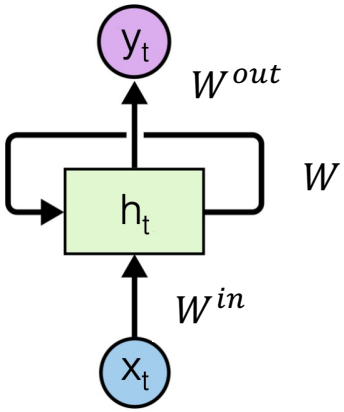
RNNを用いた，sin関数の推定
　青：Training data
　オレンジ：Validation data

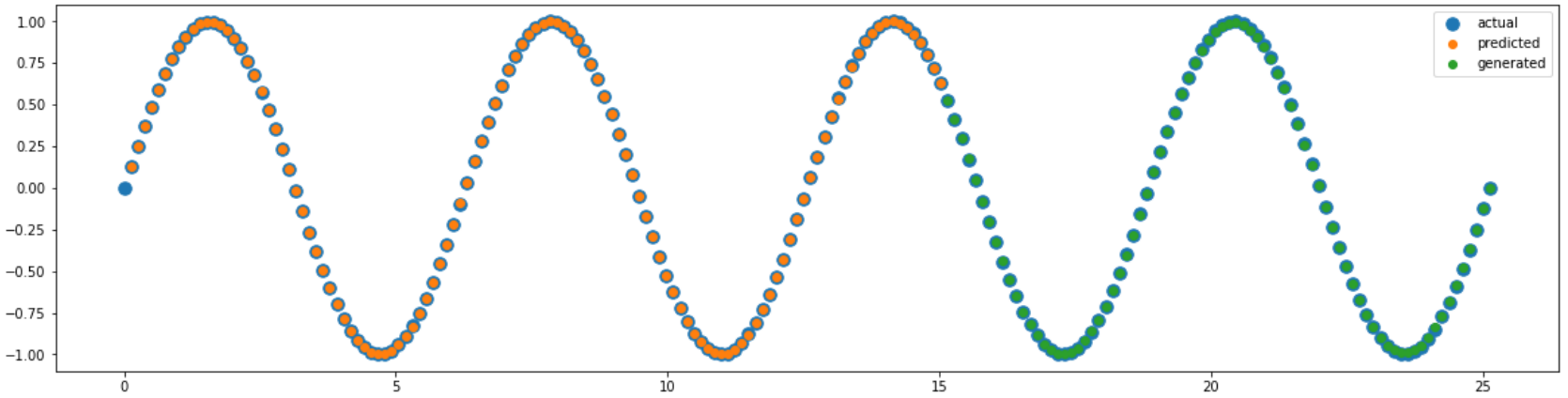




RNNからESNを理解する:結果1

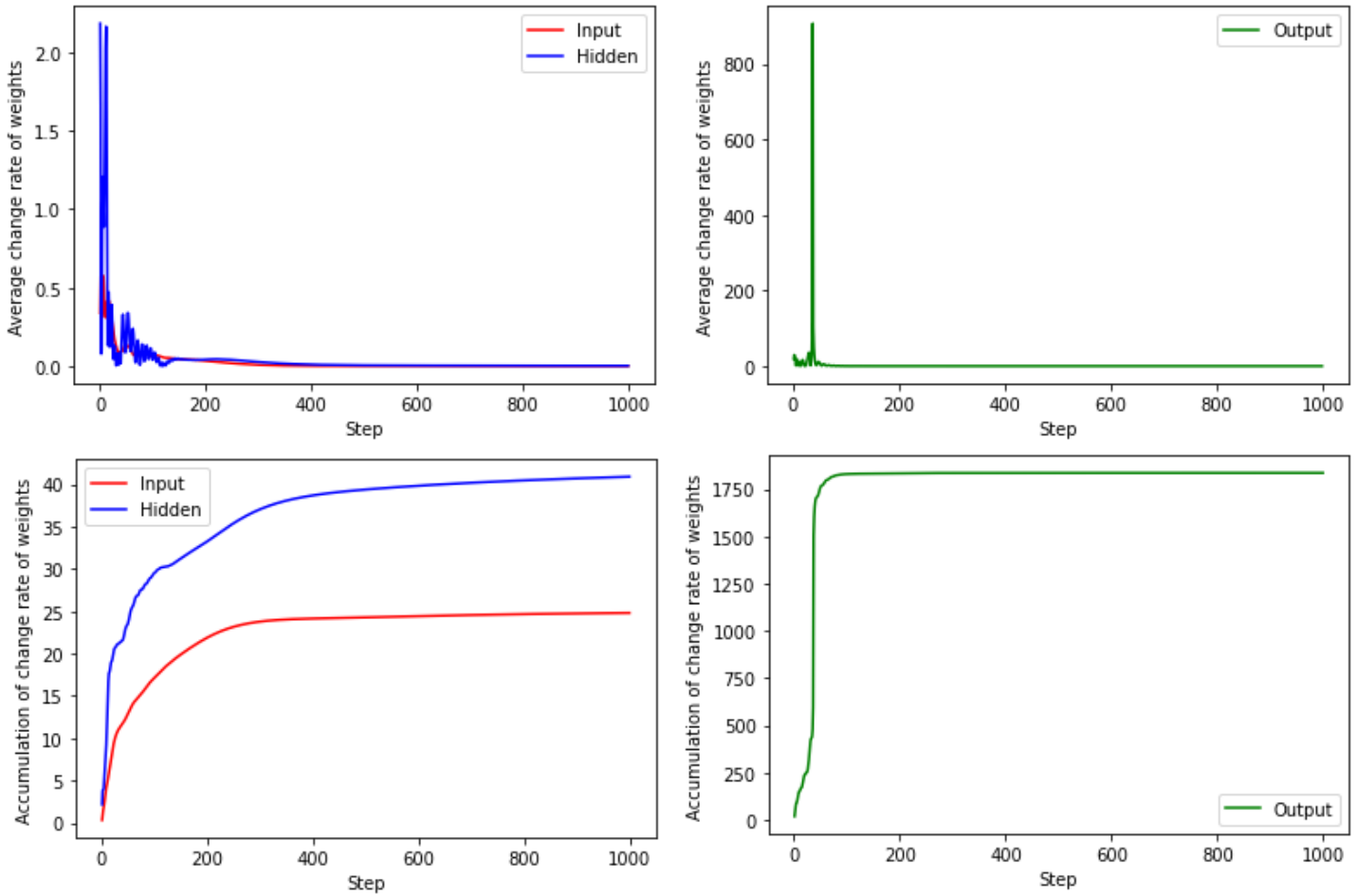
通常のRNNの予測結果

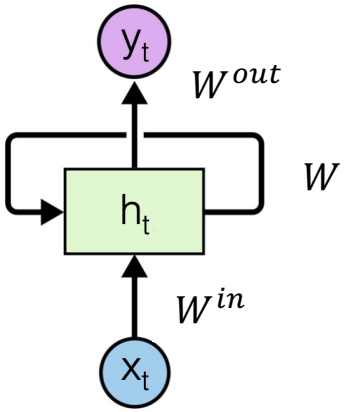




RNNからESNを理解する：結果1

通常のRNN重みの更新を観察（中間層ノード：50）





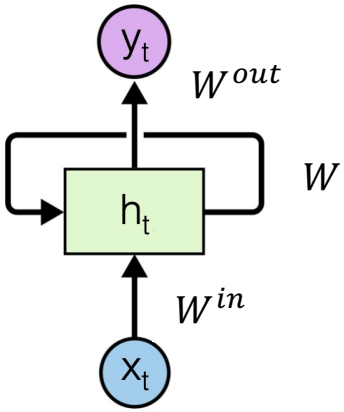
上段：重みの変化率の平均
下段：変化率の累積

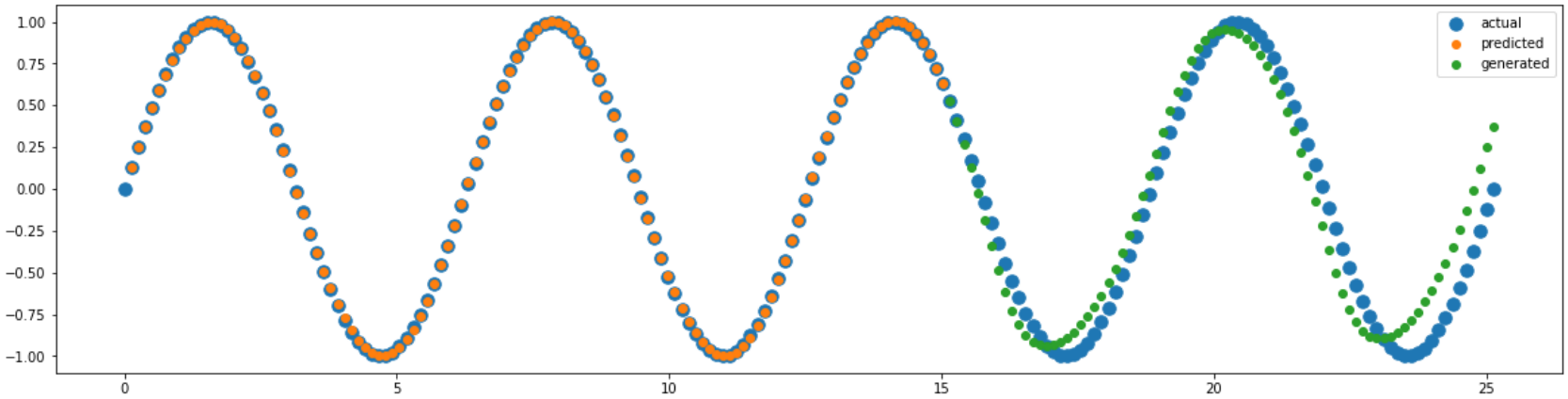
中間層を複雑にするほど顕著

RNNからESNを理解する：結果2

通常のRNN構造で中間層の更新を行わない場合

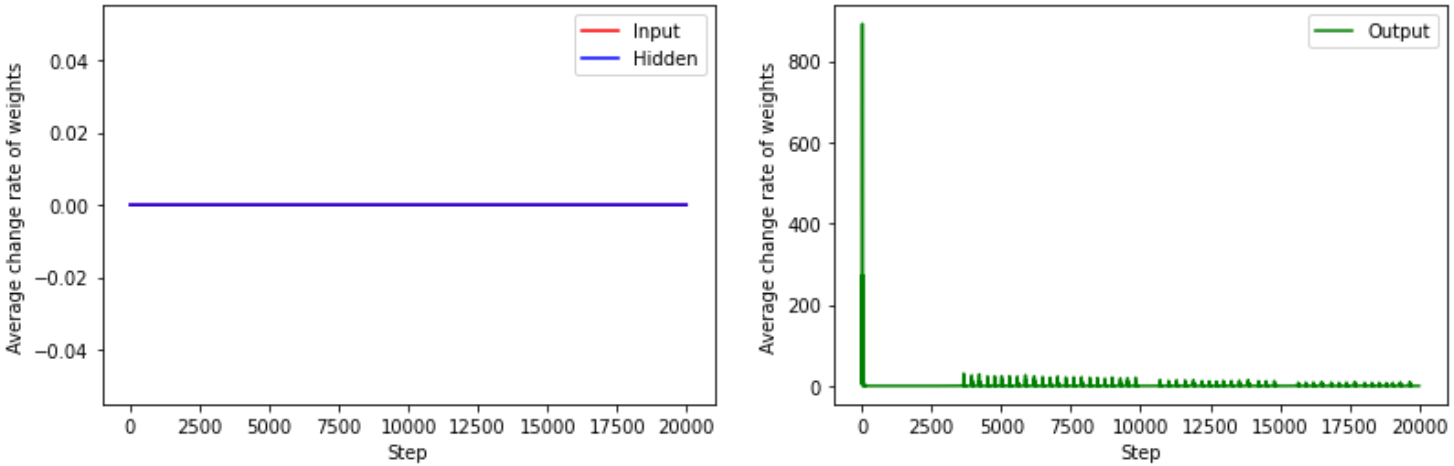
- 精度は落ちるが，おおよそ追従できている
- 計算速度は，早くなる
  → 今回は2倍程度
  → 中間層が複雑になると顕著になると予想される
- ただし，収束は遅い

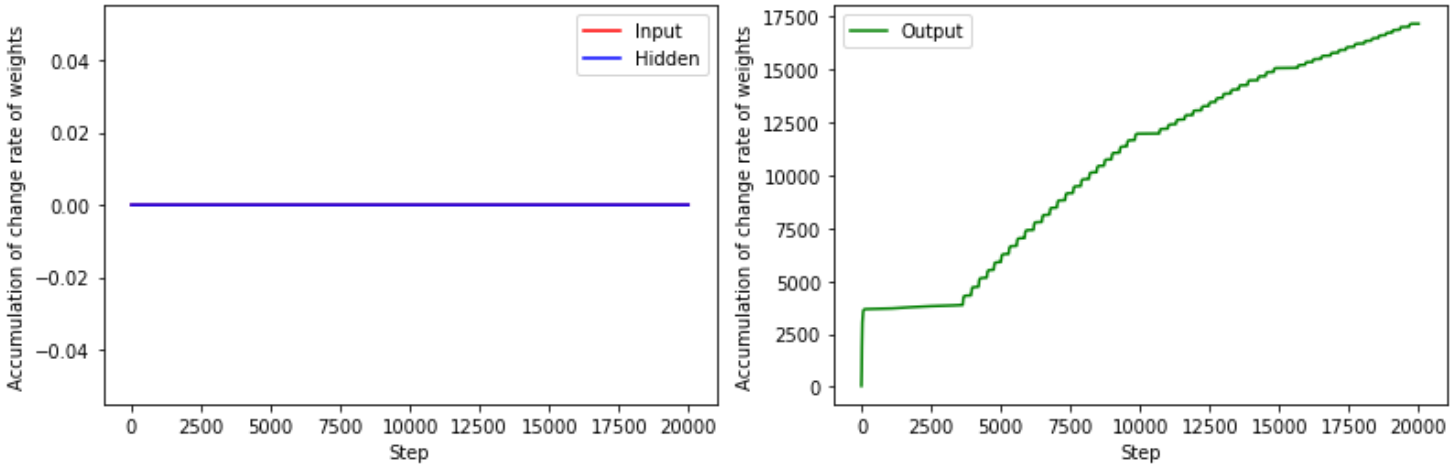


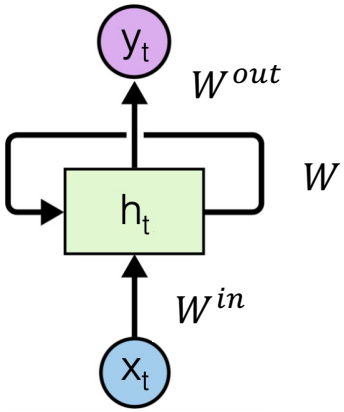


RNNからESNを理解する：結果2

重みの更新を観察（中間層ノード：50）







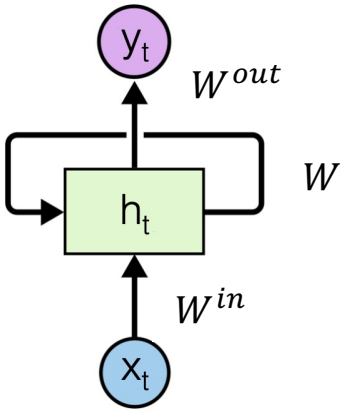
上段：重みの変化率の平均
下段：変化率の累積

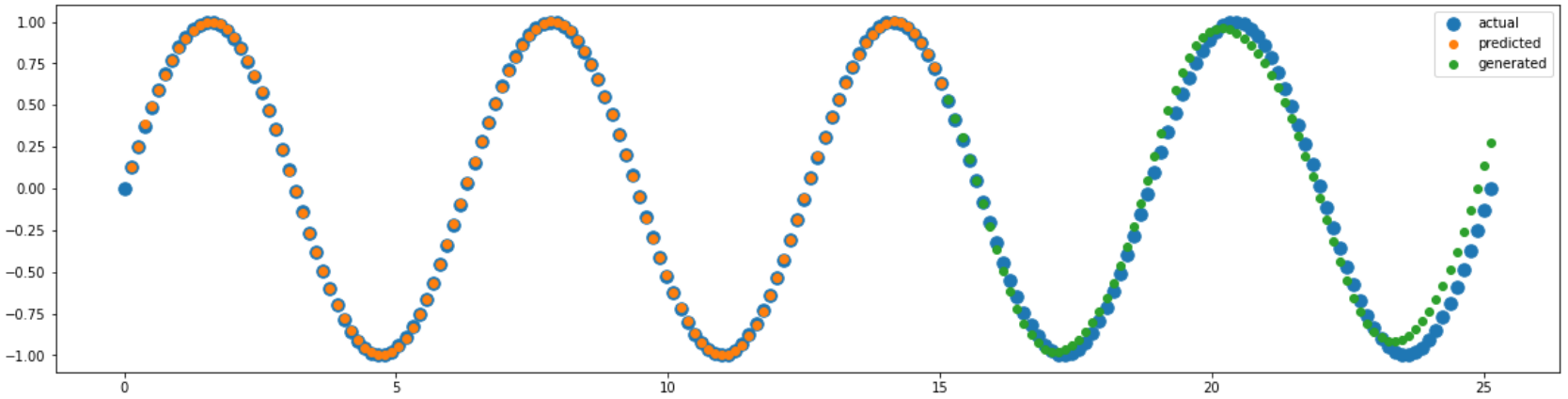
Input，Output層は0

RNNからESNを理解する：結果3

通常のRNN構造で中間層の更新を行わない かつ リンクをスパース（20 %）に

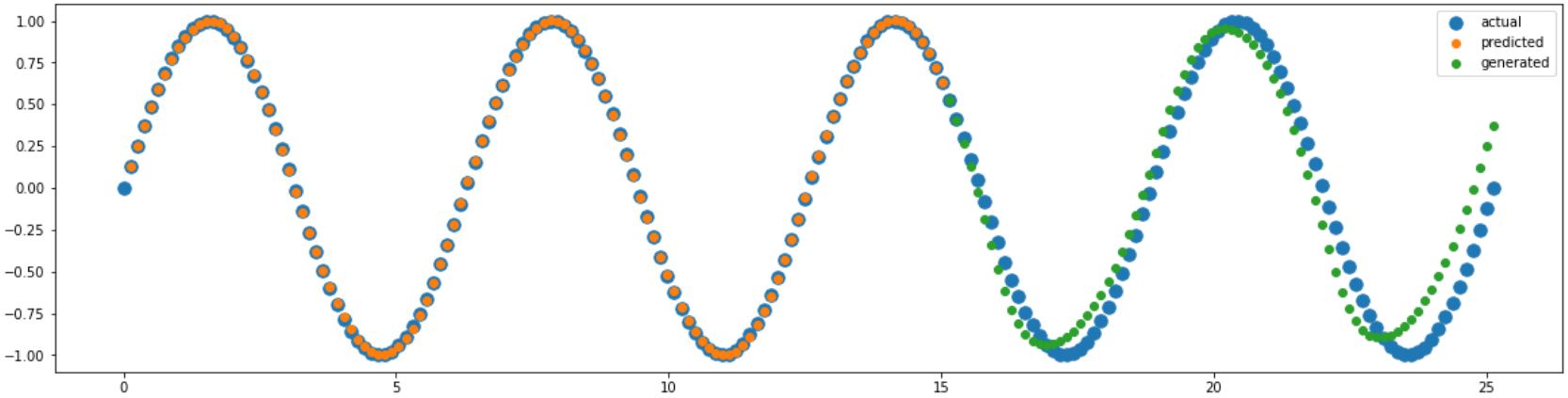
- 原理的にESNに一致すると理解している（アルゴリズムは異なる）
- 数値的にも視覚的にも精度が上がっている



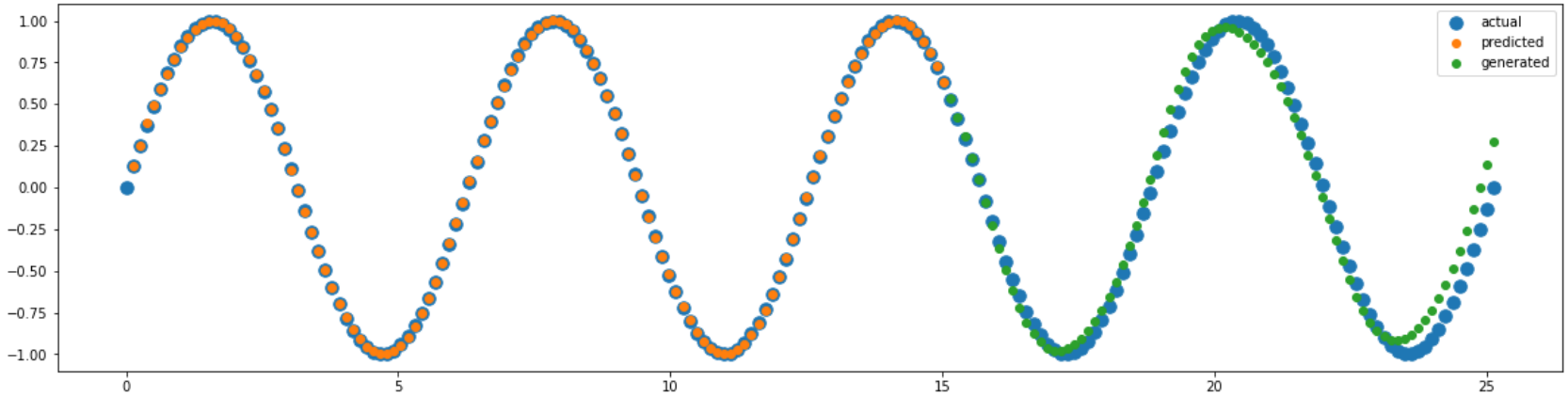


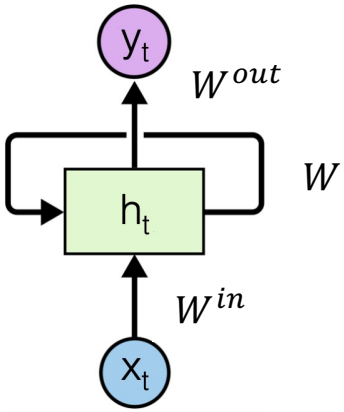
RNNからESNを理解する：結果2と3の比較

- 全結合



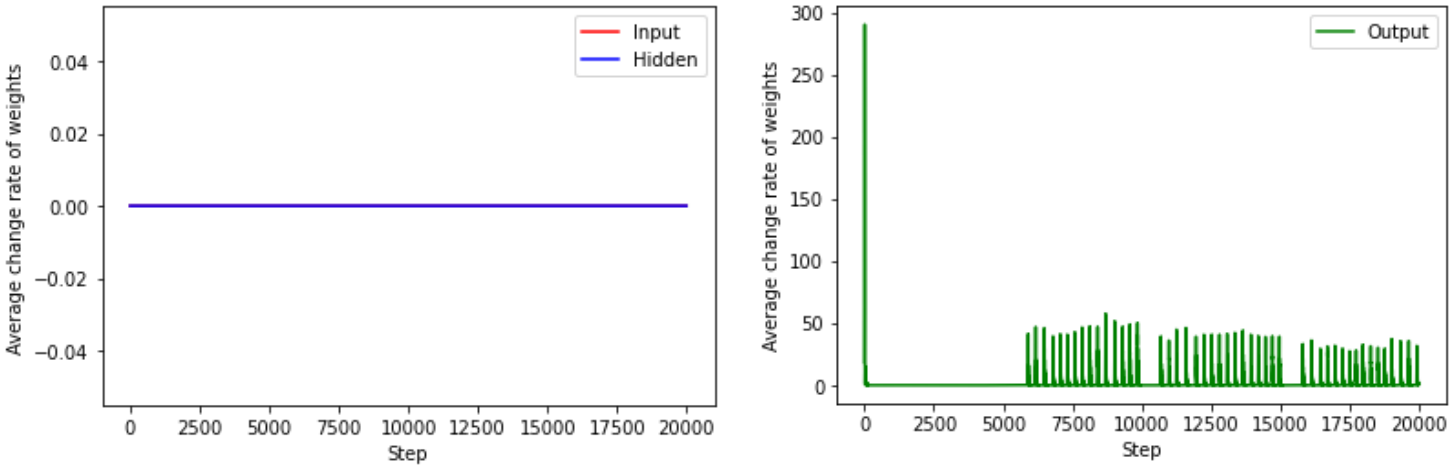
- スパースなリンク(20 %)

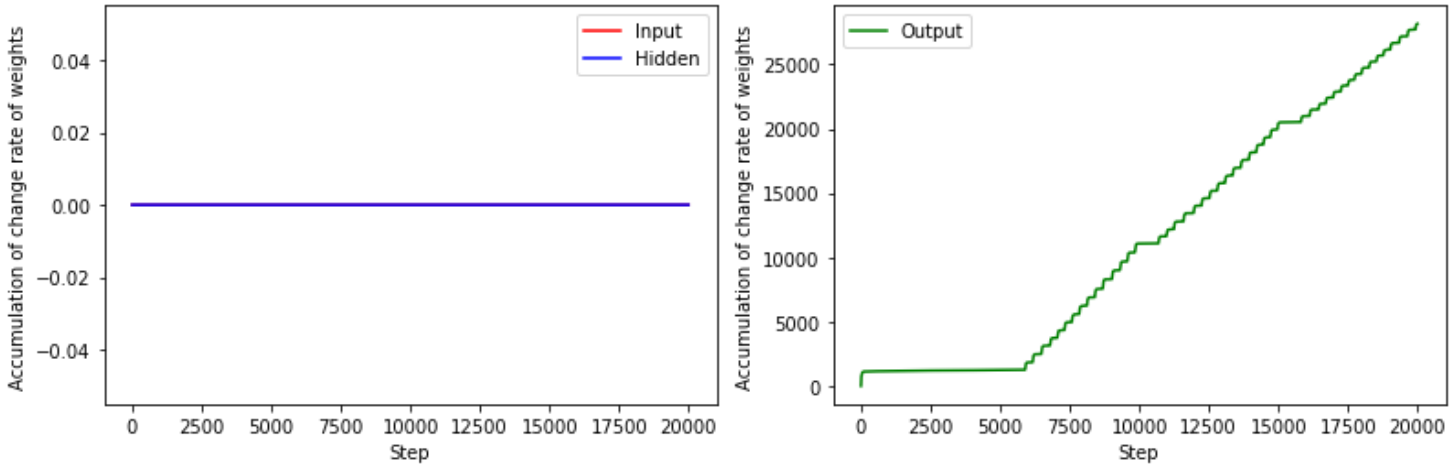


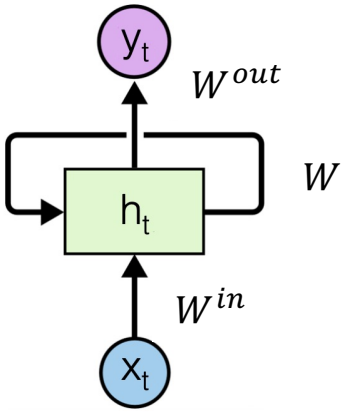


RNNからESNを理解する：結果3

重みの更新を観察（中間層ノード：50）







上段：重みの変化率の平均
下段：変化率の累積

Input，Output層は0

いくつかの結論

- ESN = RNN +　重みの部分的更新 + 結合のスパース化

- RNNに関して中間層が十分に複雑な場合，出力層の重みの最適化だけで十分なことがある
  → RNNを適用してみて各層の重みの変化率を調べることでESNのモデリングの指針になる？
  → 出力層が卓越していれば，ESNの適用を検討する

- 逆に中間層のノードが小さい時は，中間層も更新される
  → 各ノードへの要求が厳しくなる（少数精鋭）

- 少なくともESNはRNNの高速化手法の結果得られると理解すると自然
  → ネットワークの重要な部分のみを更新させる
  → RNNほど汎用的ではないが，高速

感想

- sin波でも少しチューニングが必要でした
- 個人的には，RNNから少しづつESNに変形する過程でしっくりきました