自己組織化マップ
Self-Organizing Map
SOM

明治大学 理工学部 応用化学科
データ化学工学研究室　金子 弘昌

自己組織化マップ (SOM) とは？

✓ニューラルネットワークの１つ

✓データを可視化・見える化するための非線形手法

✓主成分分析などとは異なり、はじめに二次元平面の座標を作ってしまい、それを実際の多次元空間のサンプルに合わせ込むというスタンス

✓オーバーフィッティングを起こしやすいので注意が必要

✓SOMのいろいろな問題点を解決した、上位互換の手法に Generative Topographic Mapping (GTM) がある

- GTMに対するSOMのメリットは、手法の説明が簡単、コーディングがしやすい、くらい

SOMを作る おおまかな流れ

✓２次元マップのサイズを決める
- 10×10 とか、4×6 とか

✓２次元の各グリッドにニューロンを配置する
- 10×10なら、100個のニューロン
- 各ニューロンは、データセットの変数の数と同じ要素数をもつベクトル
- 要素の値はランダム

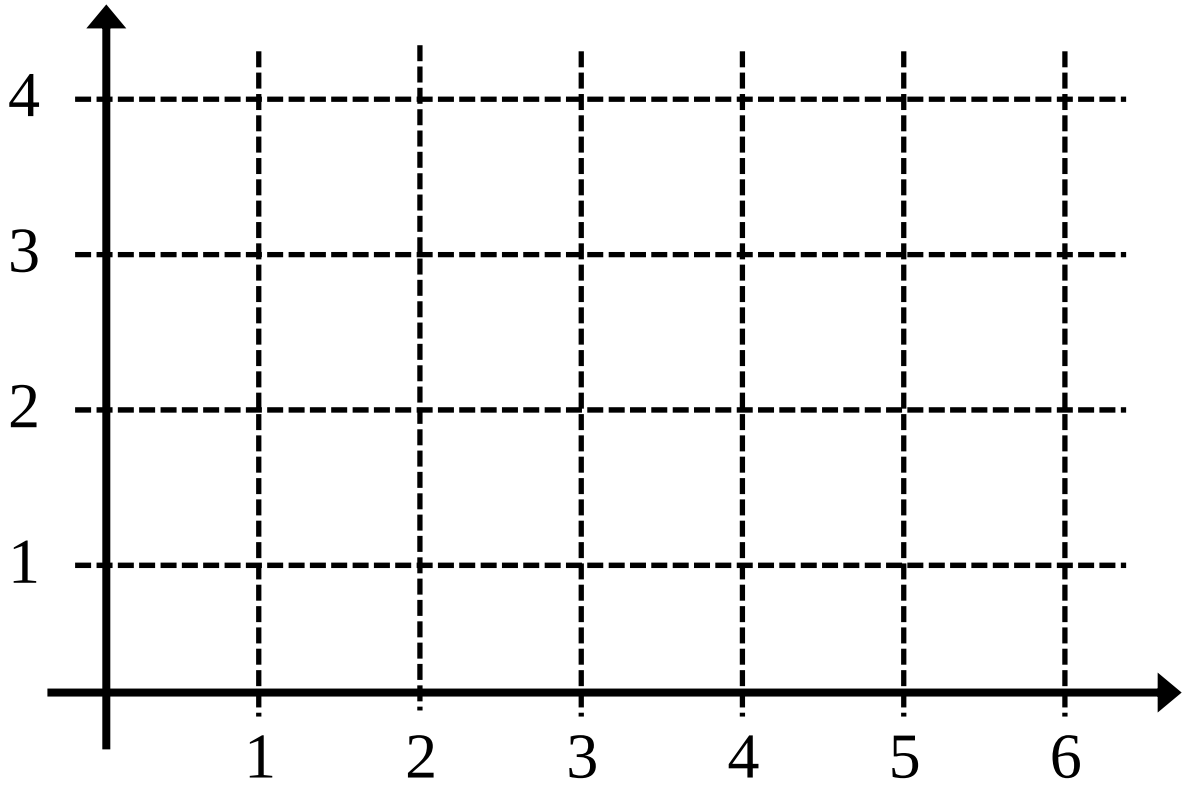
✓以下を繰り返す
- データセットのサンプルごとに最もユークリッド距離の近いニューロン (勝者ニューロン) を見つける
- 勝者ニューロンをそのサンプルに少し近づける
- 勝者ニューロンに近いニューロンも、そのサンプルに少し近づける

こんなデータセットがあるとする

		変数						
		1	2	・・・	i	・・・	m-1	m
サンプル	1							
	2							
	⋮							
	j				$x_i^{(j)}$			
	⋮							
	n-1							
	n							

２次元マップのサイズを決める

✓ここでは簡単のため、４× ６にします

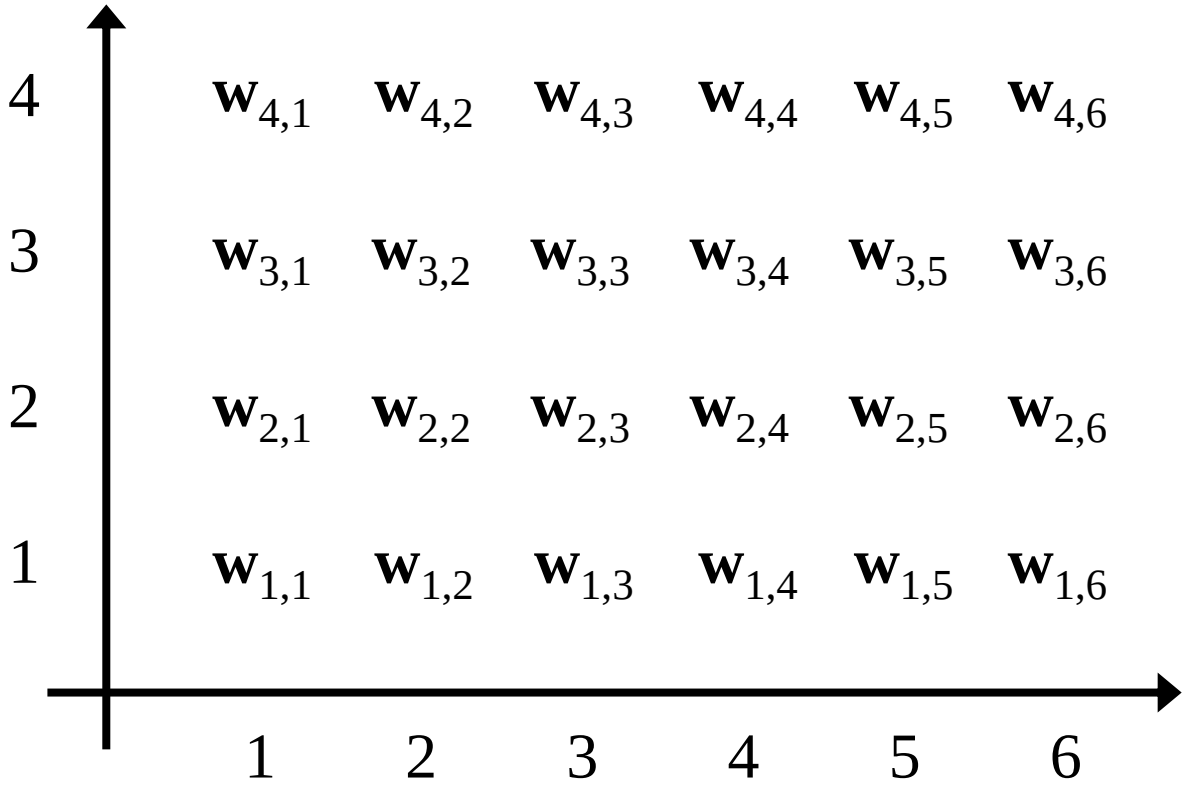


2 次元の各グリッドにニューロンを配置する

✓各ニューロン $\mathbf{w}_{i,j}$ は変数の数 n の長さをもつベクトル

- $\mathbf{w}_{i,j} = [\ w_{i,j,1} \quad w_{i,j,2} \quad \cdots \quad w_{i,j,k} \quad \cdots \quad w_{i,j,m\text{-}1} \quad w_{i,j,m}\ ]$

✓最初は $w_{i,j,k}$ を乱数とする (ニューロンの初期化)



サンプルとニューロンとの距離を計算する

✓各サンプルを $\mathbf{x}^{(j)} = [\ x_1^{(j)} \quad x_2^{(j)} \quad \cdots \quad x_i^{(j)} \quad \cdots \quad x_{m\text{-}1}^{(j)} \quad x_m^{(j)}\ ]$ とする

✓１つのサンプルと、すべてのニューロンとの間でユークリッド距離を計算する

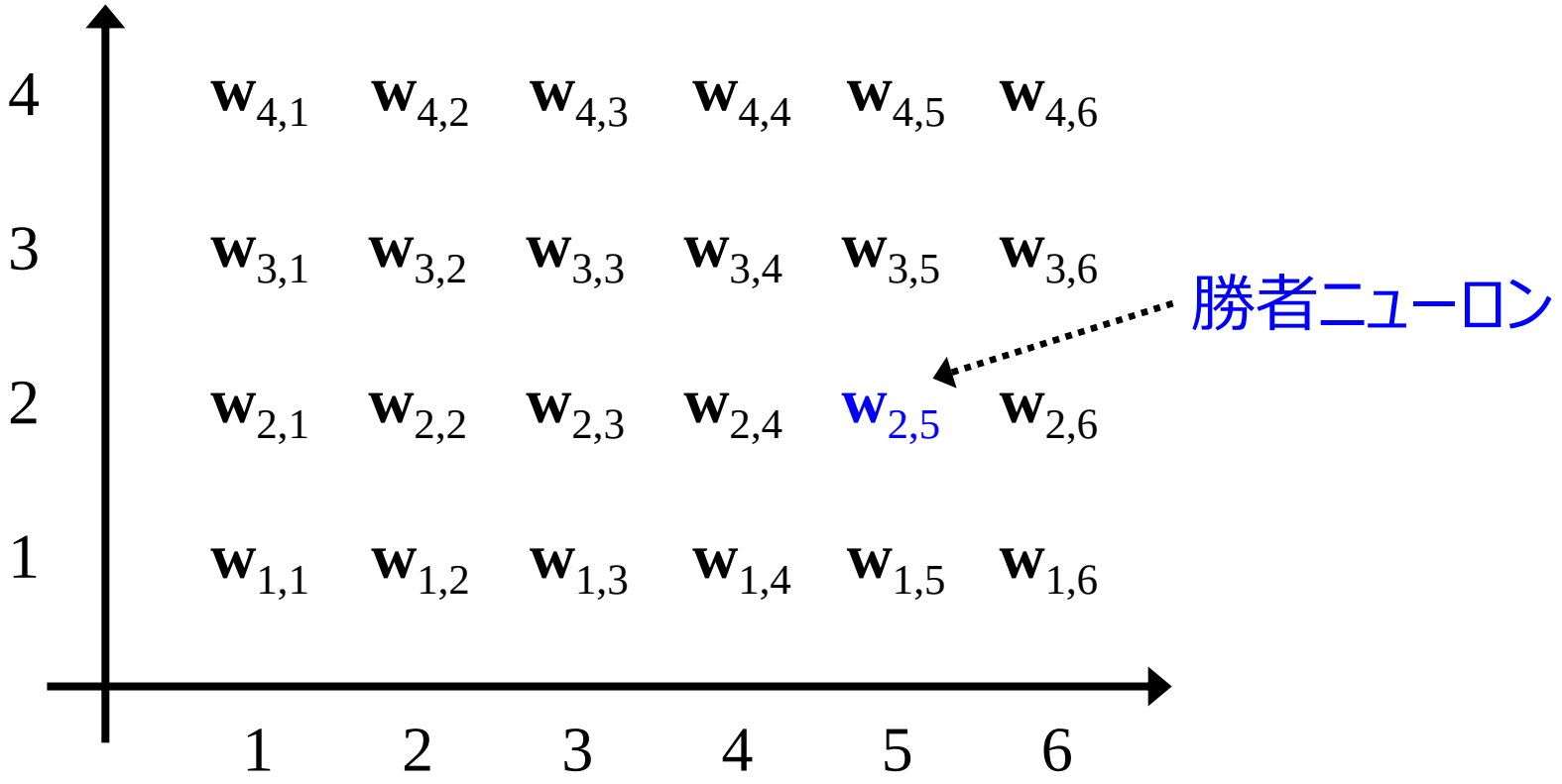
- 例) $\mathbf{x}^{(1)}$ と $\mathbf{w}_{4,3}$ との間のユークリッド距離 d

$$d = \left\| \mathbf{x}^{(1)} - \mathbf{w}_{4,3} \right\| = \sqrt{\sum_{i=1}^{n} \left(x_i^{(1)} - w_{4,3,i} \right)^2}$$

最も距離の近いニューロンを見つける

✓勝者ニューロン：あるサンプルとの距離が最も小さいニューロン

- 例) $\mathbf{x}^{(1)}$ について、勝者ニューロンは $\mathbf{w}_{2,5}$



勝者ニューロンをサンプルに少し近づける

✓勝者ニューロンを $\mathbf{w}_{2,5}$ とすると、修正後のニューロン $\mathbf{w}_{\mathbf{new}2,5}$ は、

$$\mathbf{w}_{\mathbf{new}\,2,5} = \mathbf{w}_{2,5} + \alpha\left(\mathbf{x}^{(1)} - \mathbf{w}_{2,5}\right)$$

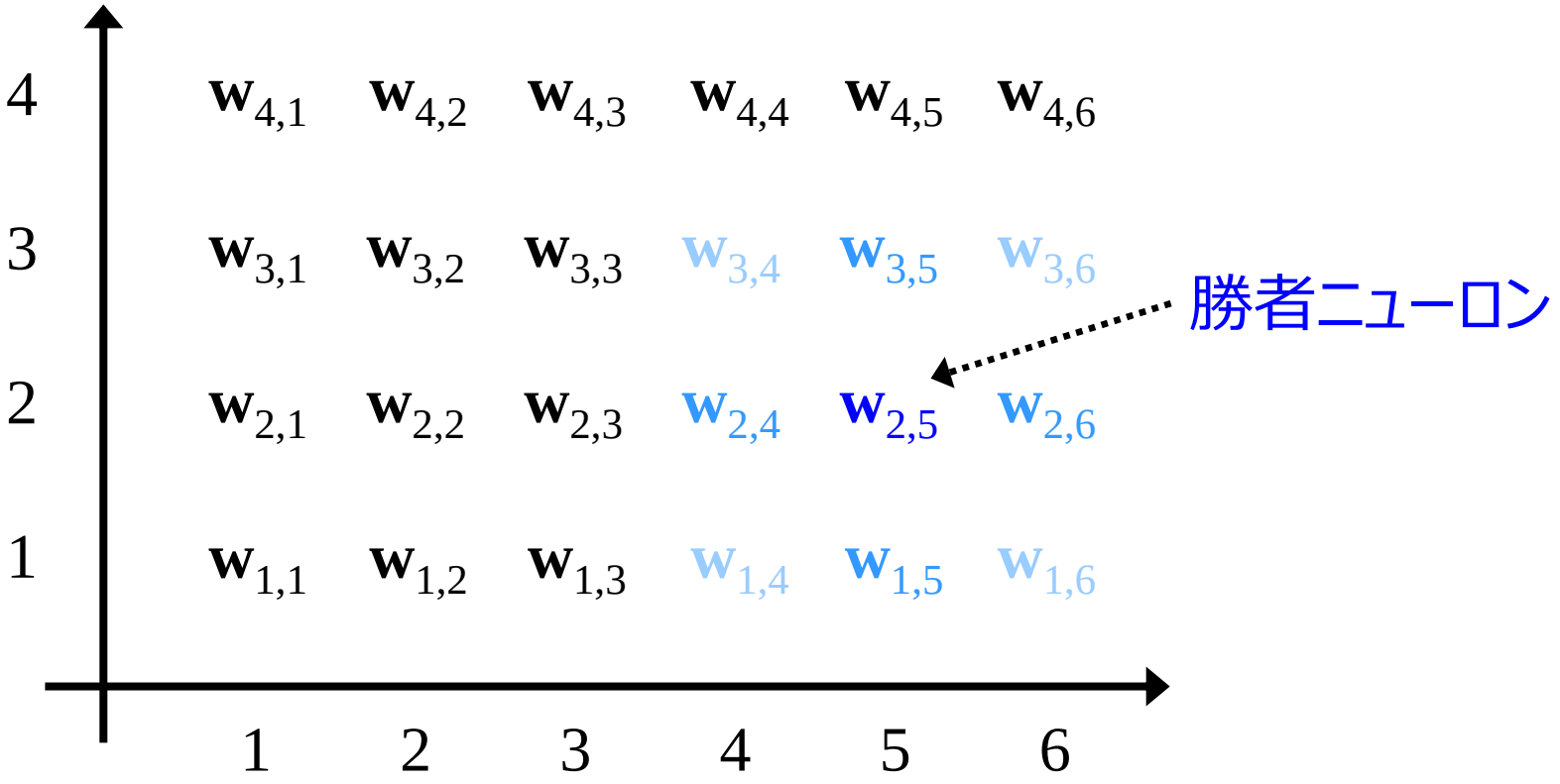
α：学習率 ($0 < \alpha < 1$)

✓トーラスマッピングにすると端のニューロンの不公平感をなくせる

- トーラスマッピング：二次元マップの一番右の右は左、
  一番上の上は下、とすること、マップはドーナツ状

勝者ニューロンに近いのもサンプルに近づける

✓勝者ニューロンを $\mathbf{w}_{2,5}$ とすると、その近くに存在するニューロンも、
$\mathbf{w}_{\mathbf{new}2,5}$ ほどではないがサンプル $\mathbf{x}^{(1)}$ に近づける



勝者ニューロンに近いのもサンプルに近づける

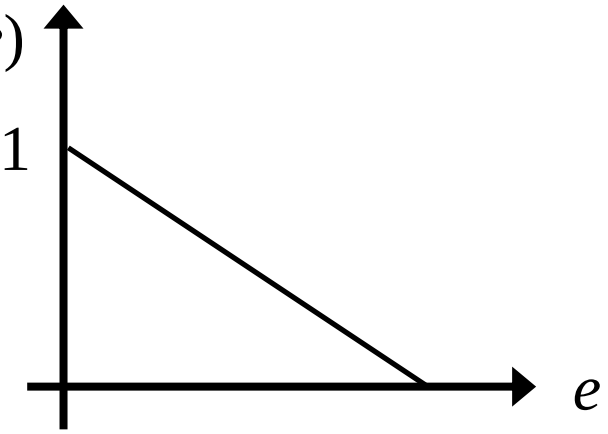
✓勝者ニューロンを $\mathbf{w}_{2,5}$ とすると、たとえば、その近くのニューロン $\mathbf{w}_{2,4}$ の修正後のニューロン $\mathbf{w}_{\mathbf{new}2,4}$ は、

$$\mathbf{w}_{\mathbf{new}\,2,4} = \mathbf{w}_{2,4} + \alpha\left(\mathbf{x}^{(1)} - \mathbf{w}_{2,4}\right) \times g\left(e\right)$$

$g(e)$：近傍関数

e：二次元マップ上での勝者ニューロンとの距離

近傍関数の例　$g(e)$



二次元マップの学習を繰り返す

✓学習：勝者ニューロン・その近くのニューロンをサンプルに近づけること

✓サンプルを順番に学習させる

✓すべてのサンプルを学習させ終わったら、もう一巡

✓何順させるか：学習回数
- 事前に決めておく

✓一巡するごとに、サンプルの順番をシャッフルさせることで、
均等に学習させることができる

SOMの特徴

✓学習が終わった後、サンプルごとの勝者ニューロンを見ることで、
二次元マップ上での可視化が達成される

✓ニューロン間の距離を見ることで、クラスタリングも検討できる

- ニューロン間の距離が大きいところは、クラスターの境目
- ただ、狙ってクラスタリングしたわけではなく、たまたまクラスターの境目になることもあるため、別途クラスタリングをしたほうが無難

SOMの問題点

✓事前に学習回数・学習率を決めなければならない

✓学習回数を多くしたからといって、二次元マップが収束するとは限らない

✓二次元マップのサイズ・学習回数・学習率・近傍関数をすべて適切に決めないと、

- 二次元マップが各サンプルにオーバーフィットしてしまう
- 二次元マップが実際の多次元空間において滑らかにならない

SOMの問題点の解決策

✓Generative Topographic Mapping (GTM) を用いる

- 先にあげた問題点を解決できる