M2 Statistics & Data Science

# Advanced Statistics & Machine Learning

Faicel Chamroukhi
Professeur
https://chamroukhi.com/

## Overview

1 Topographic Learning

## Topographic Learning for clustering/visualisation

- Self-Organizing Maps (SOMs)
- Generative Topographic Mapping (GTM)
- Formulation of the Generative Topographic Mapping (GTM)
- GTM Through Time
- Formulation of the GTM Through Time (GTM-TT)

## Introduction I

Topographic learning has become a widely used approach for the analysis, dimensionality reduction, visualization of high-dimensional data.

One of the most used topographic approaches is the self-organizing map (SOM) (Kohonen, 2001) and its generative version : the Generative Topographic Mapping (GTM) (Bishop and Williams, 1998).

The SOM idea consists in an unsupervised learning approach based on artificial neural networks and was inspired from the competitive learning.

## Competitive learning I

Competitive learning (Kong and Kosko, 1991) is an unsupervised adaptive process during which the neurons of a neural network (units) compete for the right to respond to a subset of the input data

Each unit (neuron) of the network gradually becomes specialized of a subset of the data

When an input is presented, the neuron that is best to represent it in the sense of a chosen similarity measure, typically an Euclidean distance, wins the competition and is allowed to "learn" from it, this is the well-known **"winner-take-all"** rule.

During the learning process, the neurons (units) **compete** for the right to respond to a subset of the input data and each unit of the network becomes specialized of a subset of the data.

## Competitive learning II

When an input $\mathbf{x}_i$ is presented, the neuron that is best to represent it (in the sense of a chosen similarity measure, typically a distance (Euclidean)) wins the competition and it allowed to learn from it.

Only the winner neuron (also called the best matching unit (BMU)) is updated, that is :

$$\boldsymbol{\mu}_{z_i}^{(q+1)} = \boldsymbol{\mu}_{z_i}^{(q)} + \alpha^{(q)}(\mathbf{x}_i - \boldsymbol{\mu}_{z_i}^{(q)}) \tag{1}$$

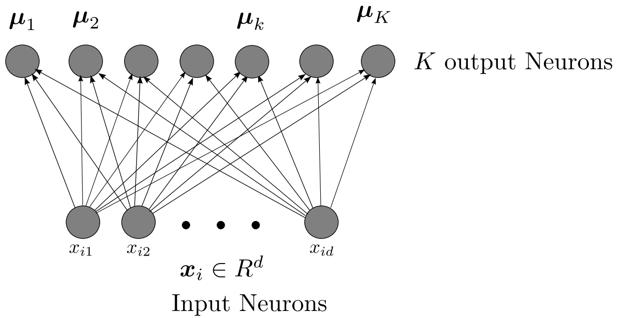
where

$$z_i = \arg\min_{k \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{2}$$

is the index of the winner prototype and $0 < \alpha^{(q)} < 1$ is the learning rate which decreases monotonically as the learning proceeds.

## Competitive learning III

$\boldsymbol{\mu}_k = (\mu_{k1}, \ldots, \mu_{kd})' \in R^d$ prototype (vector of weights) associated to neuron $k$

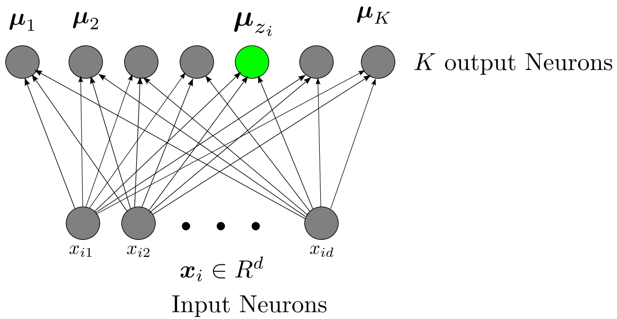


# Competitive learning IV

Step1 : Determine the **winner neuron**

$z_i = \arg \min_{1<k<K} \| \boldsymbol{x}_i - \boldsymbol{\mu}_k \|^2$

$z_i$ being the index of the **winner neuron** for $\boldsymbol{x}_i$

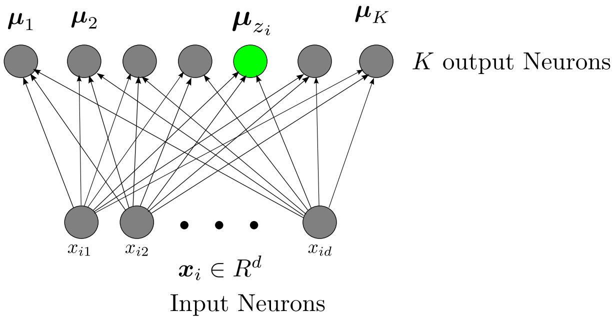


## Competitive learning V

Step 2 : Update **only** the winner neuron (Winner-Take-All):

$$\boldsymbol{\mu}_{z_i}^{\text{new}} = \boldsymbol{\mu}_{z_i}^{\text{old}} + \alpha^{\text{old}}(\boldsymbol{x}_i - \boldsymbol{\mu}_{z_i}^{\text{old}})$$

$z_i$ being the index of the **winner neuron** for $\boldsymbol{x}_i$



## Competitive learning VI

If $\alpha^{(q)} = \frac{1}{(\#\text{class } z_i)^{(q)}+1}$, one obtains the sequential $K$-means algorithm with $(\#\text{class } z_i)^{(q)}$ being the number observations assigned to the neuron $z_i$ at the previous iteration.

**Some limitations :**

In the standard formulation of the competitive learning, only the winner neuron is updated, this is the well-known "winner-take-all" rule.

In addition, it does not consider the order between the neurons (i.e, when the neurons are located on a map lattice).

$\Rightarrow$ The self-organizing map (SOM) generalizes the competitive learning by allowing also the neighbors of the winner to be updated and for which the neurons become ordered on a map lattice.

## Self-Organizing Maps (SOMs) I

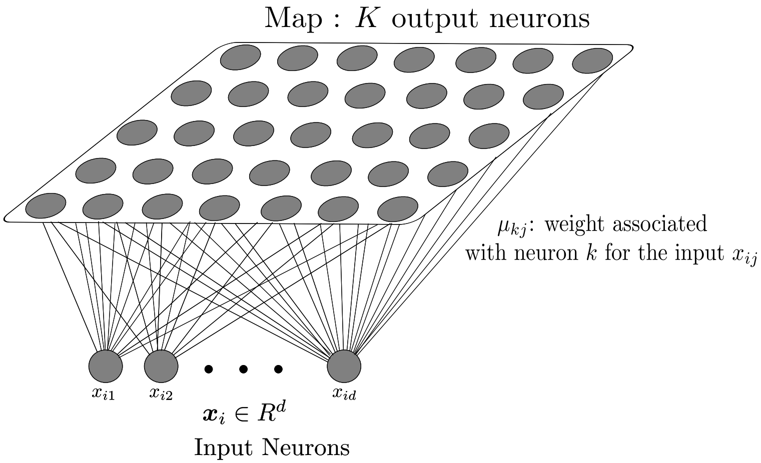
The self-organizing map (SOM) (Kohonen, 2001, 1989; Kohonen et al., 2000) is a neural-based approach for the exploration and visualization of high-dimensional data.

It derives an orderly mapping of multidimensional data onto a regular typically 2-dimensional grid (map).

It is a non linear projection method that converts complex nonlinear relationships in the high-dimensional space into simpler geometric relationships in the plan such that the important topological and metric relationships are conveyed.

The data are organized on the map in such a way that observations that are close together in the high-dimensional space are also closer to each other on the map.

## Self-Organizing Maps (SOMs) II

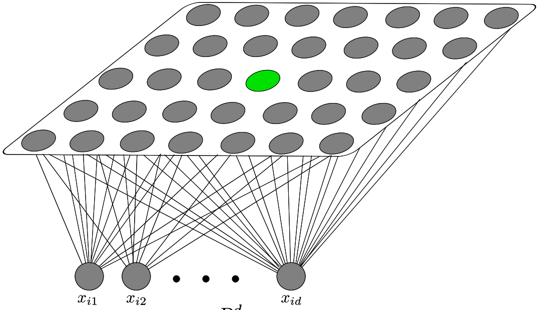

Map : $K$ output neurons
$\mu_{kj}$: weight associated with neuron $k$ for the input $x_{ij}$
$x_{i1}$
$x_{i2}$
$x_{id}$
$\boldsymbol{x}_i \in R^d$
Input Neurons

# Self-Organizing Maps (SOMs) III

Step 1: **Competetion** : determine the **winner neuron**

$$z_i = \arg \min_{1<k<K} \| \boldsymbol{x}_i - \boldsymbol{\mu}_k \|^2$$

$z_i$ being the index of the **winner neuron** for $\boldsymbol{x}_i$



$\boldsymbol{x}_i \in R^d$

Input Neurons

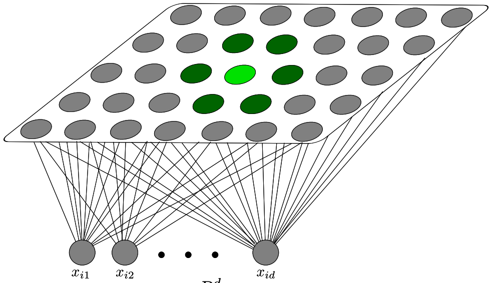
## Self-Organizing Maps (SOMs) IV

Step 2: **cooperation** : Update **all** the neurons in a weighted manner

$$\boldsymbol{\mu}_k^{\text{new}} = \boldsymbol{\mu}_k^{\text{old}} + \alpha^{\text{old}}\mathcal{K}^{\text{old}}(\mathbf{r}_{z_i}, \mathbf{r}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{\text{old}})$$

$z_i$ being the index of the **winner neuron** for $\boldsymbol{x}_i$

$\mathbf{r}_k$ denotes the coordinates of neuron $k$ on the map



$\boldsymbol{x}_i \in R^d$

Input Neurons

## Self-Organizing Maps (SOMs) V

**The Learning process of the SOM**

There are two methods used in learning the SOM : incremental (sequential) learning and batch learning.

**The incremental** learning method is an iterative procedure

we start with a data point $\mathbf{x}_i$ and a set of $d$-dimensional model vectors $\boldsymbol{\mu}_k$ called neurons, units or prototypes.

Each neuron (prototype) is associated with a coordinate vector $\mathbf{r}_k$ on a 2-D map lattice and starts with some initial value $\boldsymbol{\mu}_k^{(q=0)}$).

At each iteration, a vector $\mathbf{x}_i$ is selected and the distance (typically Euclidean, but other choices are possible) between it and all the prototypes is calculated (for the competition).

## Self-Organizing Maps (SOMs) VI

The best-matching unit (BMU) or prototype (the winner of the competition) is found and is denoted by $\boldsymbol{\mu}_{z_i}$

$$z_i = \arg\min_{k \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2. \quad (3)$$

Once the closest prototype $\boldsymbol{\mu}_{z_i}$ is found, the prototypes are updated so that $\boldsymbol{\mu}_{z_i}$ is moved closer to the data vector $\mathbf{x}_i$. The neighbors of the BMU $\boldsymbol{\mu}_{z_i}$ are also updated, in a weighted manner (the **cooperation**) as follows :

$$\boldsymbol{\mu}_k^{(q+1)} = \boldsymbol{\mu}_k^{(q)} + \alpha^{(q)} \mathcal{K}^{(q)}(z_i, k)(\mathbf{x}_i - \boldsymbol{\mu}_k^{(q)}) \quad (4)$$

$q$ denotes iteration number, $0 < \alpha^{(q)} < 1$ is the learning rate which decreases monotonically as the learning proceeds

$\mathcal{K}(z_i, k)$ is a chosen neighborhood function around the winner unit $z_i$ (in general we have $\mathcal{K}^{(q)}(z_i, z_i) = 1$).

## Self-Organizing Maps (SOMs) VII

The neighborhood function can for example be a Gaussian centered at the best-matching unit :

$$\mathcal{K}^{(q)}(z_i, k) = \exp\left(-\frac{\|\mathbf{r}_k - \mathbf{r}_{z_i}\|^2}{2\sigma^{2(q)}}\right)$$

$\mathbf{r}$ denotes the coordinates of the prototypes on the map, $\sigma^{(q)}$ the width of the neighborhood which decreases monotonically as the learning proceeds.

$\Rightarrow$ Due to the neighborhood function, the units which are closer to the BMU will be more affected than the others.

The previous steps are repeated until all the patterns $\mathbf{x}_i$ $(i = 1, \ldots, n)$ in the training set have been processed (at iteration $q$).

To achieve a better convergence towards the desired mapping it is usually required to repeat the previous loop until some convergence criteria are met (loop on iteration $q$ $(q = 1, \ldots, q_{\max})$).

## Self-Organizing Maps (SOMs) VIII

After the training step, we have the set of prototypes over the 2-D coordinates on the map.

In a clustering context, to find a partition of the data one can run a stand clustering algorithm on the prototypes, such as $K$-means or hierarchical clustering

Thus, from this point of view of clustering, SOM can be seen as a particular initialization for a later clustering step

## Self-Organizing Maps (SOMs) IX

**The batch training**

The batch training is also iterative, but, at each iteration, it uses the whole data set before adjustments are made rather than a single data vector.

At each step of the algorithm, the data set is partitioned such that each observation is associated with its nearest model vector (prototype) in the sense of the Euclidean distance : $z_i = \arg\min_{k \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2$

The updated prototypes (units) $\boldsymbol{\mu}_k^{(q+1)}$ are found as a weighted average of the data, where the weight of each observation is the value of the neighborhood function at its BMU $z_i$, that is $\mathcal{K}^{(q)}(z_i, k)$ :

$$\boldsymbol{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^{n} \mathcal{K}^{(q)}(z_i, k)\mathbf{x}_i}{\sum_{i=1} \mathcal{K}^{(q)}(z_i, k)}. \qquad (5)$$

## Self-Organizing Maps (SOMs) X

**SOM as optimizing a cost function**

We note that while the SOM can be seen as an unsupervised learning algorithm (stochastically) minimizing the following cost function (Kaski, 1997; Kohonen, 2001)

$$J^{\mathrm{SOM}}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z}) = \sum_{k=1}^{K} \sum_{i=1}^{n} \mathcal{K}(z_i, k) \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2, \tag{6}$$

The previous learning rules for the SOM correspond to a gradient descent in minimizing this SOM cost function

Several methods exist for visualizing the resulting map and prototypes, among them one can cite the U-matrix (Ultsch and Siemon, 1990) that is often used to locate clusters in the data.

## Self-Organizing Maps (SOMs) XI

**Limitations**

Note that, at the origin, the SOM algorithm is based on heuristics and is not derived from the optimization of an objective function.

In addition, the preservation of the neighborhood structure is not guaranteed by the SOM method, and there could be problems with convergence of the prototype vectors.

The SOM does not define a density model, the choice of how the neighborhood function should shrink during training is also sensitive (the parameter $\sigma$).

$\Rightarrow$ The generative topographic mapping (GTM) (Bishop and Williams, 1998) was inspired by the SOM and attempts to overcome its limitations.

## Self-Organizing Maps (SOMs) XII

The GTM is described in terms of a latent variable model (or space) with dimensionality $d$ (Bishop and Williams, 1998).

The goal is to find a representation for the distribution $p(\mathbf{x})$ of $d$-dimensional data, in terms of a smaller number of $p$ latent variables where $p < d$ and often take d = 2 for ease of visualization.

Additionally, the model parameters learning is performed the Expectation-Maximization (EM) algorithm in a maximum likelihood framework.

Both convergence and topographic ordering are guaranteed with the GTM.

## Generative Topographic Mapping (GTM) I

The Generative Topographic Mapping (GTM) (Bishop and Williams, 1998), is a non-linear probabilistic projection method for data visualization, dimensionality reduction, etc

It was inspired by the SOM and attempts to overcome its limitations through a **probabilistic formulation**.

The GTM is described in terms of a **latent variable (or space) model** with dimensionality $L$

$\Rightarrow$ the goal is to find a representation for the distribution $p(\mathbf{y})$ of $d$-dimensional data, in terms of a smaller number of $L$ latent variables where $L < d$ (often take $L = 2$ for visualization in the plan)

$\Rightarrow$ the model parameters learning is performed by the well-established **EM algorithm**

## Generative Topographic Mapping (GTM) II

GTM has a well established statistical background and relies on the well-known stability and convergence properties of the Expectation-Maximization (EM) algorithm (Dempster et al., 1977).

Both convergence and topographic ordering are guaranteed with the GTM.

In addition, GTM performs soft clustering in contrary to SOM which assigns the neurons to the clusters in a hard way.

Further comparisons with the SOM algorithm can be found in (Bishop and Williams, 1998).

## Generative Topographic Mapping (GTM) III

Let $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ be a set of $n$ iid multidimensional data vectors $\mathbf{y}_i = (y_{i1}, \ldots, y_{id})^T \in \mathbb{R}^d$

Let $\mathbf{z} = (z_1, \ldots, z_n)$ be the associated unknown (hidden) states with $z_i \in \{1, \ldots, K\}$.

Now consider a two-dimensional latent space (the map) $\mathbf{x} = (x_1, x_2)^T$ on which we aim to visualize the data.

## Generative Topographic Mapping (GTM) IV

the GTM model is a latent data (space) model asd is based on the finite mixture model formulation (McLachlan and Peel., 2000)

The aim is to represent the distribution of the observed data $p(\mathbf{y}_i)$ in the data space $\mathbb{R}^d$ in terms of a number of $L < d$-dimensional latent variables $\mathbf{x}$ with prior distribution $p(\mathbf{x})$.

The distribution $p(\mathbf{y})$ is then obtained by integration over the distribution of $\mathbf{x}$ by considering a specified conditional density $p(\mathbf{y}|\mathbf{x})$, that is :

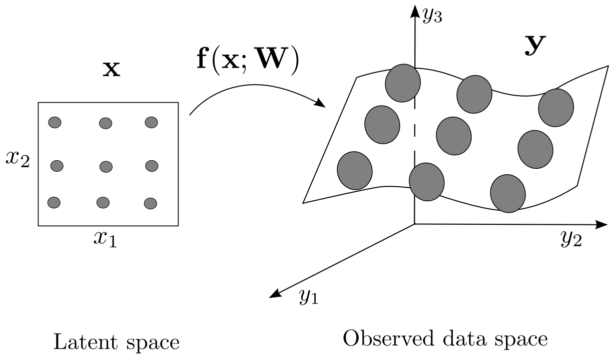
$$p(\mathbf{y}_i) = \int_{\mathcal{X}} p(\mathbf{y}_i|\mathbf{x})p(\mathbf{x})d\mathbf{x}. \quad (7)$$

For computational tractability of this integral, the GTM model assumes that the latent variables $\mathbf{x}$ have a prior Dirac mixture density given by

$$p(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K}\delta(\mathbf{x} - \mathbf{x}_k) \quad (8)$$

$\mathbf{x}_k$ represents the coordinates of the Dirac placement on the latent space.

## Generative Topographic Mapping (GTM) V



# Generative Topographic Mapping (GTM) VI

To specify the conditional density of the observations $\mathbf{y}$ on the latent variables $\mathbf{x}$, For the GTM, this is achieved by considering a parametric non-linear mapping function $\mathbf{f}(\mathbf{x};\mathbf{W})$ that maps the latent data $\mathbf{x}$ from the latent space to corresponding points in the data space.

$\Rightarrow$ the conditional density of the observations is then given as a Gaussian density centered at the projected points $\mathbf{f}(\mathbf{x};\mathbf{W})$ with variance $\beta^{-1}$ :

$$p(\mathbf{y}_i|\mathbf{x}_k) = \left(\frac{\beta}{2\pi}\right)^{d/2} \exp\left\{-\frac{\beta}{2}\|\mathbf{y}_i - \mathbf{f}(\mathbf{x}_k;\mathbf{W})\|^2\right\} = \mathcal{N}(\mathbf{y}_i;\mathbf{f}(\mathbf{x}_k;\mathbf{W}),\beta^{-1}\mathbf{I}_d) \quad (9)$$

$\mathbf{f}(\mathbf{x}_k;\mathbf{W}) = \mathbf{W}\breve{}(\mathbf{x}_k)$ is a $d$-dimensional point in the manifold embedded in data space

$\mathbf{W}$ is a $d \times M$ matrix of parameters that govern the mapping,

$\breve{}(\mathbf{x}_k) = (\Phi_1(\mathbf{x}_k), \ldots, \Phi_M(\mathbf{x}_k))$ consists of $M$ non-linear basis functions (in the standard model $\phi_m(\mathbf{x}_k)$ is a Gaussian : $\Phi_m(\mathbf{x}_k) = \exp\left\{-\frac{\|\mathbf{x}_k - \mu_m\|^2}{2\sigma^2}\right\}$).

## Generative Topographic Mapping (GTM) VII

The GTM density (7) finally results in the following mixture density :

$$p(\mathbf{y}_i; \mathbf{W}, \beta) = \int_{\mathcal{X}} p(\mathbf{y}_i|\mathbf{x})p(\mathbf{x})d\mathbf{x} = \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{y}_i|\mathbf{x}_k) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{N}(\mathbf{y}_i; \mathbf{f}(\mathbf{x}_k; \mathbf{W}), \beta^{-1}\mathbf{I}_d). \tag{10}$$

The estimation of the GTM models parameters $(\mathbf{W}, \beta)$ from and i.i.d data sample is performed by maximizing the observed-data likelihood

$$p(\mathbf{Y}; \bar{}) = \prod_{i=1}^{n} \frac{1}{K}\sum_{k=1}^{K} \mathcal{N}(\mathbf{y}_i; \mathbf{f}(\mathbf{x}_k; \mathbf{W}), \beta^{-1}\mathbf{I}_d). \tag{11}$$

via the EM algorithm (see the previous chapters or (Bishop and Williams, 1997)(Bishop et al., 1998)).

## Generative Topographic Mapping (GTM) VIII

**Some limitations**

The standard GTM model (Bishop and Williams, 1997; Bishop et al., 1998) is dedicated to i.i.d data

The independence assumption becomes however very restricting for analyzing sequences.

$\Rightarrow$ the GTM Through Time (GTM-TT) (Bishop et al., 1997) overcomes these two limitations by relying on a hidden Markov model (HMM) formulation

## GTM Through Time I

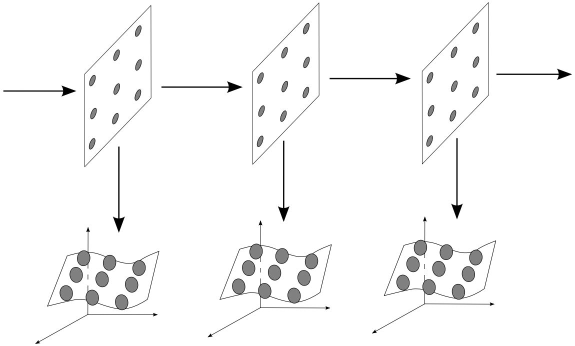
The GTM through time (GTM-TT) model extends the standard GTM model to learn from sequences by relaxing the independence assumption.

More specifically, the GTM-TT model incorporates the standard GTM model as the emission density in a Hidden Markov Model (HMM) (HMM will be studied later) as follows :

The hidden sequence $(z_1, \ldots, z_n)$ indicating the location on the latent space $(\mathbf{x}_{z_t})$ at each time step is a Markov chain with initial distribution $\pi$ and transition matrix $\mathbf{A}$ : $\pi_k = p(z_1 = k)$ and $\mathbf{A}_{\ell k} = p(z_t = k | z_{t-1} = \ell)$

The conditional emission density function is the one of the GTM model, that is $p(\mathbf{y}_t | \mathbf{x}_{z_t}) = \left(\frac{\beta}{2\pi}\right)^{d/2} \exp\left\{-\frac{\beta}{2}\|\mathbf{y}_t - \mathbf{f}(\mathbf{x}_{z_t}; \mathbf{W})\|^2\right\}$ where $z_t$ denotes the state at time $t$.

## GTM Through Time II



## GTM Through Time III

### Learning the GTM-TT

The model parameters $(\boldsymbol{\pi}, \mathbf{A}, \beta, \mathbf{W})$ are estimated by maximizing the observed data likelihood, which is expressed as the one of a standard HMM as follows

$$p(\mathbf{Y}; \bar{}) = \sum_{z_1} \ldots \sum_{z_n} p(z_1) p(\mathbf{y}_1 | \mathbf{x}_{z_1}) \prod_{t=2}^{n} p(z_t | z_{t-1}) p(\mathbf{y}_t | \mathbf{x}_{z_t}). \quad (12)$$

The maximization is performed by the EM (Baum-Welch) algorithm (Bishop et al., 1997)(Dempster et al., 1977)(Baum et al., 1970) where the E-step includes a forward-backward recursion to evaluate the posterior state distribution and to compute the likelihood.

## Bibliography I

Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. Annals of Mathematical Statistics, 41 :164–171.

Bishop, C. M., Hinton, G. E., and Strachan, I. G. D. (1997). Gtm through time. In IEE Fifth International Conference on Artificial Neural Networks, pages 111–116.

Bishop, C. M., Svensén, M., and Williams, C. K. I. (1998). Gtm : The generative topographic mapping. Neural Computation, 10 :215–234.

Bishop, C. M. and Williams, C. K. I. (1997). Gtm : A principled alternative to the self-organizing map. In Advances in Neural Information Processing Systems, pages 354–360. Springer-Verlag.

Bishop, C. M. and Williams, C. K. I. (1998). Gtm : The generative topographic mapping. Neural Computation, 10 :215–234.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of The Royal Statistical Society, B, 39(1) :1–38.

Kaski, S. (1997). Data Exploration Using Self-Organizing Maps. PhD thesis, Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering.

Kohonen, T. (1989). Self-organization and associative memory. Springer-Verlag New York, Inc. New York, NY, USA.

Kohonen, T. (2001). Self-Organizing Maps. Information Sciences. Springer, third edition edition.

Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., and Saarela, A. (2000). Self organization of a massive document collection. IEEE Transactions on Neural Networks, 11(3) :574–585.

Kong, S. and Kosko, B. (1991). Differential competitive learning for centroid estimation and phoneme recognition. IEEE Transactions on Neural Networks, 2(1) :118–124.

McLachlan, G. J. and Peel., D. (2000). Finite mixture models. New York : Wiley.

Ultsch, A. and Siemon, H. P. (1990). Kohonen's self organizing feature maps for exploratory data analysis. In Proceedings of International Neural Networks Conference (INNC), pages 305–308. Kluwer Academic Press.