Probabilistic Machine Learning
Lecture 20
Latent Dirichlet Allocation

Philipp Hennig

29 June 2021

Eberhard Karls Universität Tübingen

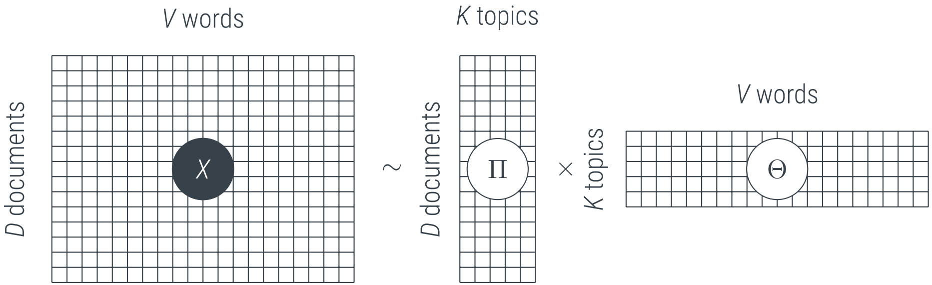
Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

Designing a probabilistic machine learning method:

1. get the **data**
   1.1 try to collect as much meta-data as possible
2. build the **model**
   2.1 identify quantities and datastructures; assign names
   2.2 design a generative process (graphical model)
   2.3 assign (conditional) distributions to factors/arrows (use exponential families!)
3. design the **algorithm**
   3.1 consider conditional independence
   3.2 try standard methods for early experiments
   3.3 run unit-tests and sanity-checks
   3.4 identify bottlenecks, find customized approximations and refinements

A Mixture of Probabilities?

Desiderata



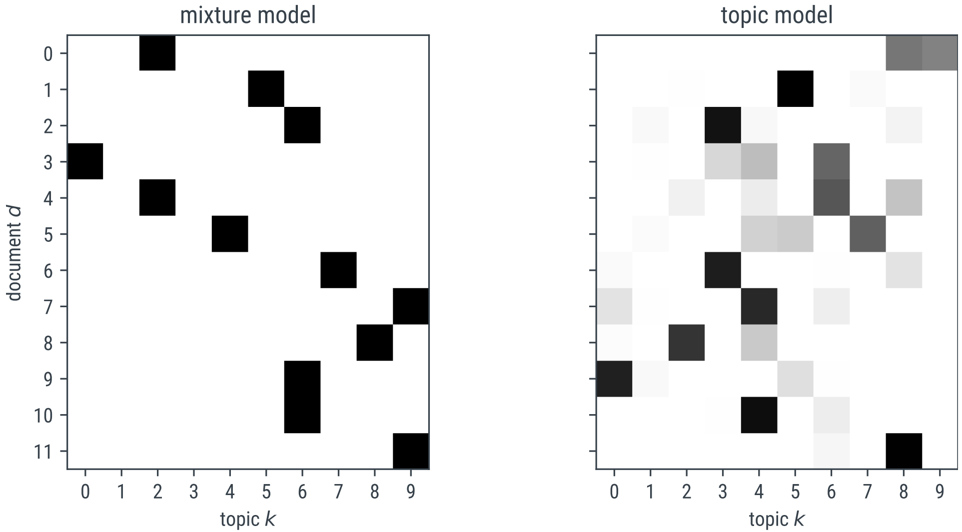
- topics should be probabilities: $p(\boldsymbol{x}_d \mid k) = \prod_{v=1}^{V} \prod_{i=1}^{x_{dv}} \theta_{k(i),v}$
- but documents should have *several* topics! Let π_{dk} be the *probability* to draw a word from topic k

doc sparsity each document d should only contain a small number of topics

word sparsity each topic k should only contain a small number of the words v in the vocabulary

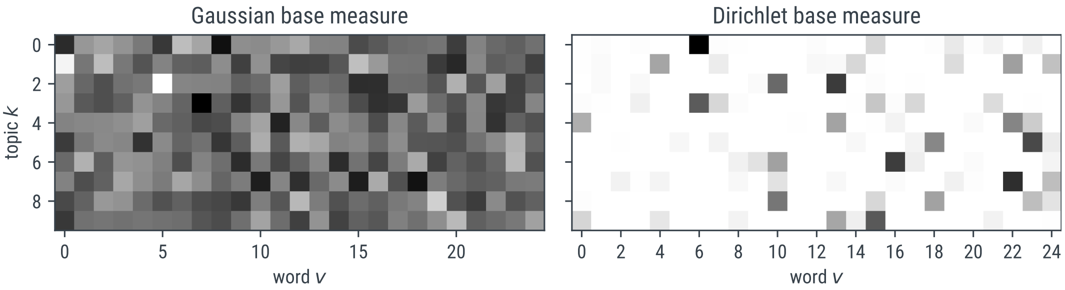
Each document discusses *a few* topics

Dirichlet sparsity in the document-topic distribution



Each topic contains *a few* words

Dirichlet sparsity in the topic-word distribution



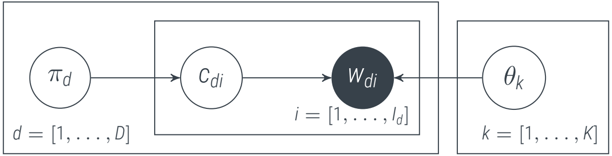
Designing a probabilistic machine learning method:

1. get the **data**
   1.1 try to collect as much meta-data as possible
2. build the **model**
   2.1 identify quantities and datastructures; assign names
   2.2 design a generative process (graphical model)
   2.3 assign (conditional) distributions to factors/arrows (use exponential families!)
3. design the **algorithm**
   3.1 consider conditional independence
   3.2 try standard methods for early experiments
   3.3 run unit-tests and sanity-checks
   3.4 identify bottlenecks, find customized approximations and refinements

A Discrete Topic Model

almost there

EBERHARD KARLS UNIVERSITÄT TÜBINGEN



To draw I_d words $w_{di} \in [1, \dots, V]$ of document $d \in [1, \dots, D]$:

- Draw topic assignments $c_{di} = [c_{di1}, \dots, c_{diK}] \in \{0; 1\}^K, \sum_k c_{dik} = 1$ of word w_{id} from

$$\Rightarrow C \in \{0; 1\}^{D \times I_d \times K} \qquad \text{with} \qquad p(C \mid \Pi) = \prod_{d=1}^{D} \prod_{i=1}^{I_d} \prod_{k=1}^{K} \pi_{dk}^{c_{dik}}$$

- Draw word w_{di} from

$$p(w_{di} = v \mid c_{di}, \Theta) = \prod_k \theta_{kv}^{c_{dik}}$$

But we need priors for Π, Θ. And we'd like them to be sparse!

Reminder: The Dirichlet Distribution

a sparsity prior for probability distribution

$$p(\boldsymbol{x} \mid \boldsymbol{\pi}) = \prod_{i=1}^{n} \pi_{x_i} \qquad x \in \{1;\dots,K\}$$

$$= \prod_{k=1}^{K} \pi_k^{n_k} \qquad n_k := |\{x_i \mid x_i = k\}|$$

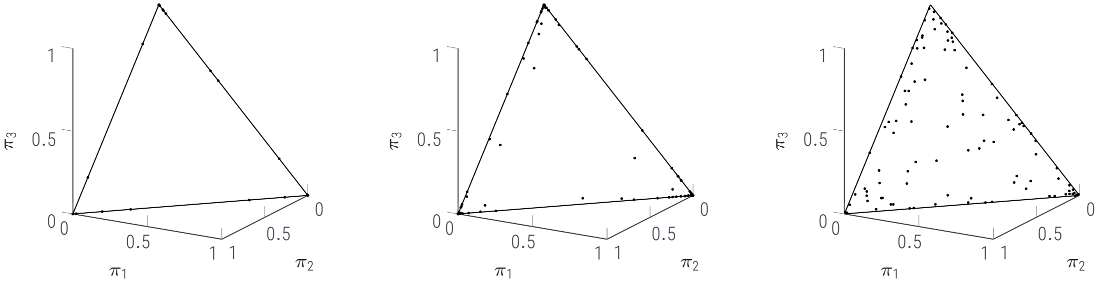
$$p(\boldsymbol{\pi} \mid \boldsymbol{\alpha}) = \mathcal{D}(\boldsymbol{\alpha}) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1}$$

$$p(\boldsymbol{\pi} \mid \boldsymbol{x}) = \mathcal{D}(\boldsymbol{\alpha} + \boldsymbol{n})$$

Peter Gustav Lejeune Dirichlet
(1805–1859)

Dirichlets can encode sparsity

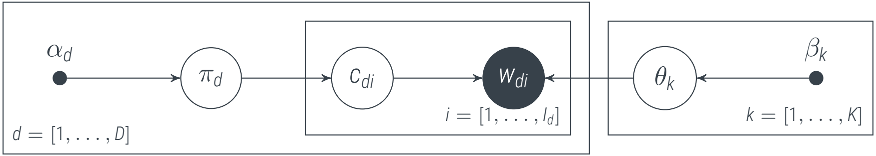
for $\alpha \sim 0.01, 0.1, 0.5$ (100 samples each)



Latent Dirichlet Allocation

our generative model

[Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003) JMLR 3, 993–1022]



To draw I_d words $w_{di} \in [1, \dots, V]$ of document $d \in [1, \dots, D]$:

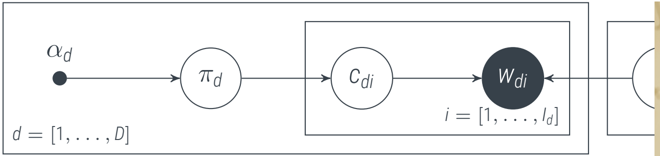
- Draw K topic distributions θ_k over V words from $\quad p(\Theta \mid \boldsymbol{\beta}) = \prod_{k=1}^{K} \mathcal{D}(\theta_k; \beta_k)$
- Draw D document distributions over K topics from $\quad p(\Pi \mid \boldsymbol{\alpha}) = \prod_{d=1}^{D} \mathcal{D}(\pi_d; \alpha_d)$
- Draw topic assignments c_{dik} of word w_{di} from $\quad p(C \mid \Pi) = \prod_{i,d,k} \pi_{dk}^{c_{dik}}$
- Draw word w_{di} from $\quad p(w_{di} = v \mid c_{di}, \Theta) = \prod_k \theta_{kv}^{c_{dik}}$

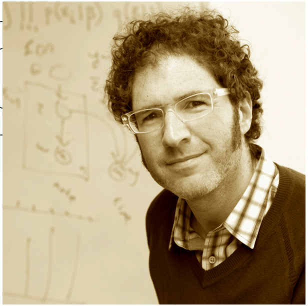
Useful notation: $n_{dkv} = \#\{i : w_{di} = v, c_{dik} = 1\}$. Write $n_{dk:} := [n_{dk1}, \dots, n_{dkV}]$ and $n_{dk\cdot} = \sum_v n_{dkv}$, etc.

Latent Dirichlet Allocation

our generative model

[Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003) JMLR 3, 993–1022]





David M. Blei

image: Columbia U

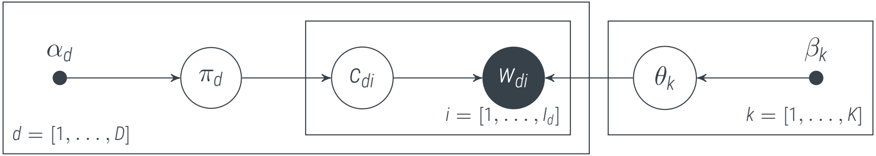
To draw I_d words $w_{di} \in [1, \dots, V]$ of document $d \in [1, \dots, D]$:

- Draw K topic distributions θ_k over V words from
- Draw D document distributions over K topics from
- Draw topic assignments c_{dik} of word w_{di} from
- Draw word w_{di} from

Useful notation: $n_{dkv} = \#\{i : w_{di} = v, c_{dik} = 1\}$. Write $n_{dk:} := [n_{dk1}, \dots, n_{dkV}]$ and $n_{dk\cdot} = \sum_v n_{dkv}$, etc.

Designing a probabilistic machine learning method:

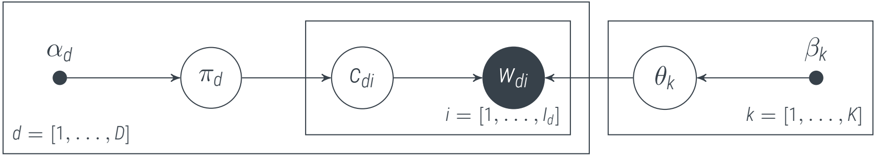
1. get the **data**
   1.1 try to collect as much meta-data as possible
2. build the **model**
   2.1 identify quantities and datastructures; assign names
   2.2 design a generative process (graphical model)
   2.3 assign (conditional) distributions to factors/arrows (use exponential families!)
3. design the **algorithm**
   3.1 consider conditional independence
   3.2 try standard methods for early experiments
   3.3 run unit-tests and sanity-checks
   3.4 identify bottlenecks, find customized approximations and refinements



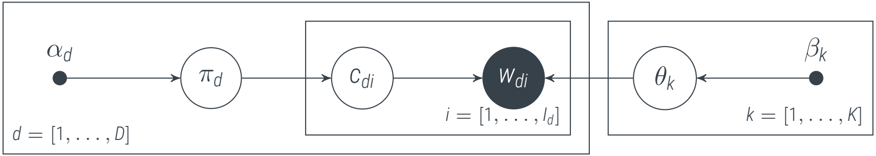
$$
\begin{aligned}
p(C, \Pi, \Theta \mid W) &= \frac{p(C, \Pi, \Theta, W)}{\iiint p(C, \Pi, \Theta, W)\, dC\, d\Pi, d\Theta} \\
&= \frac{p(W \mid C, \Pi, \Theta) \cdot p(C, \Pi, \Theta)}{\iiint p(C, \Pi, \Theta, W)\, dC\, d\Pi, d\Theta}
\end{aligned}
$$

Building an algorithm

A more careful look at the model



$$
\begin{aligned}
p(C, \Pi, \Theta, W) &= p(\Pi \mid \alpha) \cdot p(C \mid \Pi) \cdot p(\Theta \mid \beta) \cdot p(W \mid C, \Theta) \\
&= \left(\prod_{d=1}^{D} p(\boldsymbol{\pi}_d \mid \boldsymbol{\alpha}_d)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} p(c_{di} \mid \pi_d)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} p(w_{di} \mid c_{di}, \Theta)\right) \cdot \left(\prod_{k=1}^{K} p(\boldsymbol{\theta}_k \mid \boldsymbol{\beta}_k)\right) \\
&= \left(\prod_{d=1}^{D} \mathcal{D}(\boldsymbol{\pi}_d; \boldsymbol{\alpha}_d)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \pi_{dk}^{c_{dik}}\right)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \theta_{kw_{di}}^{c_{dik}}\right)\right) \cdot \left(\prod_{k=1}^{K} \mathcal{D}(\boldsymbol{\theta}_k; \boldsymbol{\beta}_k)\right) \\
&= \left(\prod_{d=1}^{D} \left(\frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk}-1}\right)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \pi_{dk}^{c_{dik}}\right)\right) \cdot \dots
\end{aligned}
$$



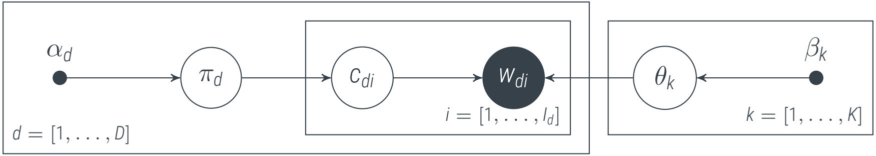
$$p(\Pi \mid \alpha) \cdot p(C \mid \Pi) = \left(\prod_{d=1}^{D} \left(\frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk}-1} \right) \right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \pi_{dk}^{c_{dik}} \right) \right)$$

Useful notation: $n_{dkv} = \#\{i : w_{di} = v, c_{dik} = 1\}$. Write $n_{dk:} := [n_{dk1}, \dots, n_{dkV}]$ and $n_{dk\cdot} = \sum_v n_{dkv}$, etc.

$$= \left(\prod_{d=1}^{D} \frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk}-1+n_{dk\cdot}} \right)$$

Building an algorithm

A more careful look at the model

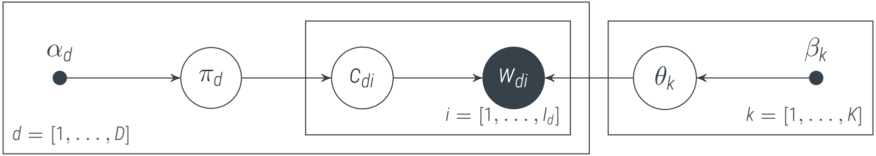


$$
\begin{aligned}
p(C, \Pi, \Theta, W) &= p(\Pi \mid \alpha) \cdot p(C \mid \Pi) \cdot p(\Theta \mid \beta) \cdot p(W \mid C, \Theta) \\
&= \left(\prod_{d=1}^{D} p(\boldsymbol{\pi}_d \mid \boldsymbol{\alpha}_d)\right) \cdot \left(\prod_{d=1}^{D}\prod_{i=1}^{I_d} p(c_{di} \mid \pi_d)\right) \cdot \left(\prod_{d=1}^{D}\prod_{i=1}^{I_d} p(w_{di} \mid c_{di}, \Theta)\right) \cdot \left(\prod_{k=1}^{K} p(\boldsymbol{\theta}_k \mid \boldsymbol{\beta}_k)\right) \\
&= \left(\prod_{d=1}^{D} \mathcal{D}(\boldsymbol{\pi}_d; \boldsymbol{\alpha}_d)\right) \cdot \left(\prod_{d=1}^{D}\prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \pi_{dk}^{c_{dik}}\right)\right) \cdot \left(\prod_{d=1}^{D}\prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \theta_{kw_{di}}^{c_{dik}}\right)\right) \cdot \left(\prod_{k=1}^{K} \mathcal{D}(\boldsymbol{\theta}_k; \boldsymbol{\beta}_k)\right) \\
&= \left(\prod_{d=1}^{D} \frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk}-1+n_{dk\cdot}}\right) \cdot \left(\prod_{k=1}^{K} \frac{\Gamma(\sum_v \beta_{kv})}{\prod_v \Gamma(\beta_{kv})} \prod_{v=1}^{V} \theta_{kv}^{\beta_{kv}-1+n_{\cdot kv}}\right)
\end{aligned}
$$

Useful notation: $n_{dkv} = \#\{i : w_{di} = v, c_{dik} = 1\}$. Write $n_{dk:} := [n_{dk1}, \ldots, n_{dkV}]$ and $n_{dk\cdot} = \sum_v n_{dkv}$, etc.

Building an algorithm

A more careful look at the model



$$p(C, \Pi, \Theta, W) = \left(\prod_{d=1}^{D} \frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk}-1+n_{dk\cdot}} \right) \cdot \left(\prod_{k=1}^{K} \frac{\Gamma(\sum_v \beta_{kv})}{\prod_v \Gamma(\beta_{kv})} \prod_{v=1}^{V} \theta_{kv}^{\beta_{kv}-1+n_{\cdot kv}} \right)$$

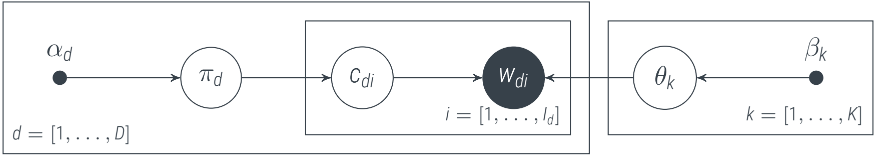
- If we had Π, Θ (which we don't), then the posterior $p(C \mid \Theta, \Pi, W)$ would be easy:

$$p(C \mid \Theta, \Pi, W) = \frac{p(W, C, \Theta, \Pi)}{\sum_C p(W, C, \Theta, \Pi)} = \prod_{d=1}^{D} \prod_{i=1}^{I_d} \frac{\prod_{k=1}^{K} (\pi_{dk} \theta_{kw_{di}})^{c_{dik}}}{\sum_{k'} (\pi_{dk'} \theta_{k' w_{di}})}$$

- note that this conditional independence can easily be read off from the above graph!

Building an algorithm

A more careful look at the model

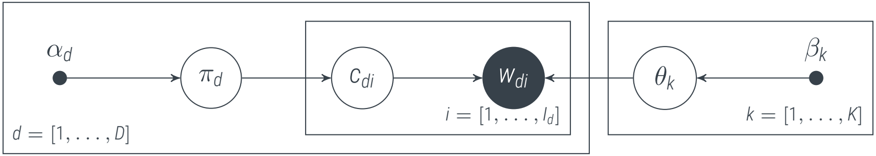


$$p(C, \Pi, \Theta, W) = \left(\prod_{d=1}^{D} \mathcal{D}(\boldsymbol{\pi}_d; \boldsymbol{\alpha}_d)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \pi_{dk}^{c_{dik}}\right)\right) \cdot \left(\prod_{d=1}^{D} \prod_{i=1}^{I_d} \left(\prod_{k=1}^{K} \theta_{kw_{di}}^{c_{dik}}\right)\right) \cdot \left(\prod_{k=1}^{K} \mathcal{D}(\boldsymbol{\theta}_k; \boldsymbol{\beta}_k)\right)$$

- If we had Π, Θ (which we don't), then the posterior $p(C \mid \Theta, \Pi, W)$ would be easy:

$$p(C \mid \Theta, \Pi, W) = \frac{p(W, C, \Theta, \Pi)}{\sum_C p(W, C, \Theta, \Pi)} = \prod_{d=1}^{D} \prod_{i=1}^{I_d} \frac{\prod_{k=1}^{K} (\pi_{dk} \theta_{kw_{di}})^{c_{dik}}}{\sum_{k'} (\pi_{dk'} \theta_{k' w_{di}})}$$

- note that this conditional independence can easily be read off from the above graph!



$$p(C, \Pi, \Theta, W) = \left(\prod_{d=1}^{D} \frac{\Gamma(\sum_k \alpha_{dk})}{\prod_k \Gamma(\alpha_{dk})} \prod_{k=1}^{K} \pi_{dk}^{\alpha_{dk} - 1 + n_{dk\cdot}} \right) \cdot \left(\prod_{k=1}^{K} \frac{\Gamma(\sum_v \beta_{kv})}{\prod_v \Gamma(\beta_{kv})} \prod_{v=1}^{V} \theta_{kv}^{\beta_{kv} - 1 + n_{\cdot kv}} \right)$$

- If we had C (which we don't), then the posterior $p(\Theta, \Pi \mid C, W)$ would be easy:

$$p(\Theta, \Pi \mid C, W) = \frac{p(C, W, \Pi, \Theta)}{\int p(\Theta, \Pi, C, W) \, d\Theta \, d\Pi} = \frac{\left(\prod_d \mathcal{D}(\pi_d; \alpha_d) \left(\prod_k \pi_{dk}^{n_{dk\cdot}}\right)\right) \left(\prod_k \mathcal{D}(\theta_k; \beta_k) \left(\prod_v \theta_{kv}^{n_{\cdot kv}}\right)\right)}{p(C, W)}$$

$$= \left(\prod_d \mathcal{D}(\pi_d; \alpha_{d:} + n_{d:\cdot}) \right) \left(\prod_k \mathcal{D}(\theta_k; \beta_{k:} + n_{\cdot k:}) \right)$$

- note that this conditional independence **can not** easily be read off from the above graph!

The Toolbox

Framework:

$$\int p(x_1, x_2)\, dx_2 = p(x_1) \qquad p(x_1, x_2) = p(x_1 \mid x_2)p(x_2) \qquad p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)}$$

Modelling:

- graphical models
- Gaussian distributions
- (deep) learnt representations
- Kernels
- Markov Chains
- Exponential Families / Conjugate Priors
- Factor Graphs & Message Passing

Computation:

- Monte Carlo
- Linear algebra / Gaussian inference
- maximum likelihood / MAP
- Laplace approximations
-

Gibbs sampling:

- $x_t \leftarrow x_{t-1}; x_{ti} \sim p(x_{ti} \mid x_{t1}, x_{t2}, \dots, x_{t(i-1)}, x_{t(i+1)}, \dots)$
- a special case of Metropolis-Hastings:
  - $q(x' \mid x_t) = \delta(x'_{\setminus i} - x_{t,\setminus i}) p(x'_i \mid x_{t,\setminus i})$
  - $p(x') = p(x'_i \mid x'_{\setminus i}) p(x'_{\setminus i}) = p(x'_i \mid x_{t,\setminus i}) p(x_{t,\setminus i})$
  - acceptance rate: $a = \dfrac{p(x')}{p(x_t)} \cdot \dfrac{q(x_t \mid x')}{q(x' \mid x_t)} = \dfrac{p(x'_i \mid x_{t,\setminus i})}{p(x_{ti} \mid x_{t,\setminus i})} \cdot \dfrac{p(x_{ti} \mid x_{t,\setminus i})}{p(x'_i \mid x_{t,\setminus i})} = 1$

Markov Chain Monte Carlo Methods provide a relatively simple way to construct approximate posterior distributions. They are asymptotically exact. Compared to other approximate inference methods, like variational inference, they *tend* to be *easier to implement* but *harder to monitor*, and *may* also be more computationally expensive

$$\int f(x)p(x)\,dx \approx \frac{1}{S}\sum_{s=1}^{S} f(x_s) \quad \text{if} \quad x_s \sim p$$

Iterate between (recall $n_{dkv} = \#\{i : w_{di} = v, c_{dik} = 1\}$)

$$\Theta \sim p(\Theta \mid C, W) = \prod_k \mathcal{D}(\theta_k; \beta_{k:} + n_{\cdot k:})$$

$$\Pi \sim p(\Pi \mid C, W) = \prod_d \mathcal{D}(\pi_d; \alpha_{d:} + n_{d:\cdot})$$

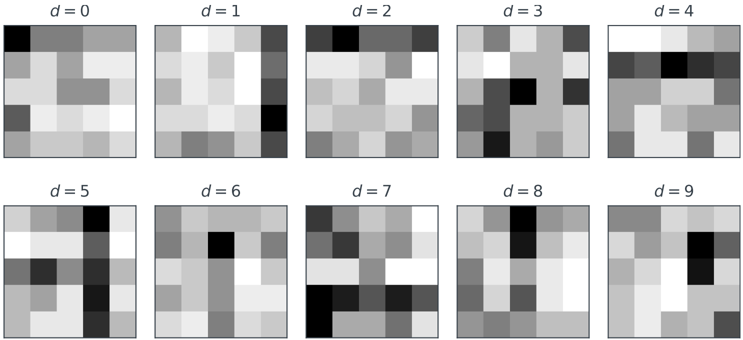
$$C \sim p(C \mid \Theta, \Pi, W) = \prod_{d=1}^{D} \prod_{i=1}^{I_d} \frac{\prod_{k=1}^{K} (\pi_{dk}\theta_{kw_{di}})^{c_{dik}}}{\sum_{k'} (\pi_{dk'}\theta_{k'w_{di}})}$$

- This is *comparably* easy to implement because there are libraries for sampling from Dirichlet's, and discrete sampling is trivial. All we have to keep around are the counts n (which are sparse!) and Θ, Π (which are comparably small). Thanks to factorization, much can also be done in parallel!
- Unfortunately, this sampling scheme is relatively slow to move out of initialization, because z depends strongly on θ, π and vice versa.
- properly vectorizing the code is important for speed

A visual example

building a toy model

adapted from T. L. Griffiths & M. Steyvers, *Finding scientific topics*, PNAS **101**, 5228–5235

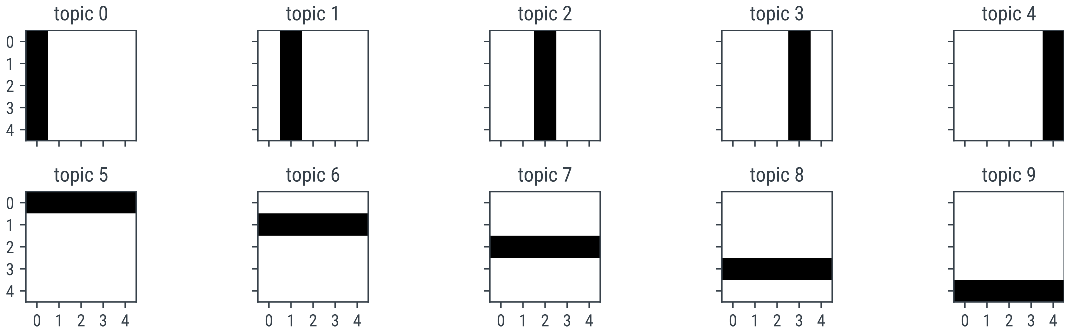


- $K = 10, V = 25, N = 100, D = 1000$
- $p(\Pi) = \prod_d \mathcal{D}(\pi_d; \alpha = 1)$

A visual example

building a toy model

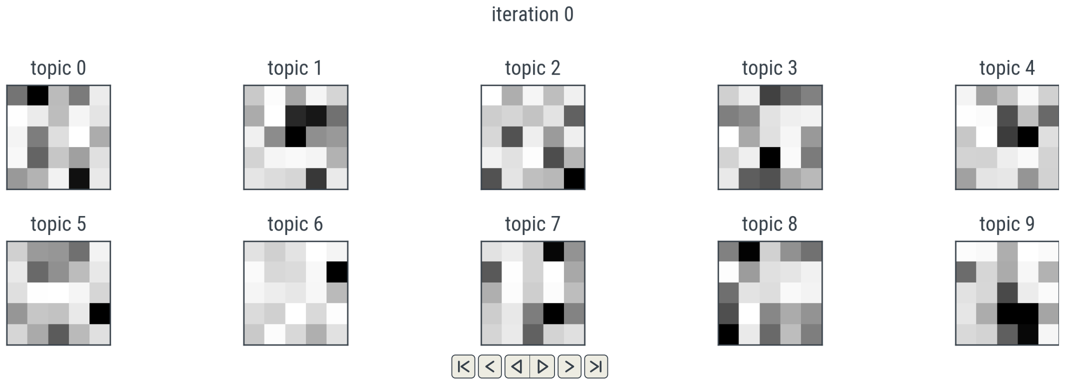
adapted from T. L. Griffiths & M. Steyvers, *Finding scientific topics*, PNAS **101**, 5228–5235



- $K = 10, V = 25, N = 100, D = 1000$
- $p(\Pi) = \prod_d \mathcal{D}(\pi_d; \alpha = 1)$

It works!

your homework this week



EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Designing a Probabilistic Machine Learning Model

1. get the **data**
   1.1 try to collect as much meta-data as possible
   1.2 take a close look at the data
2. build the **model**
   2.1 identify quantities and datastructures; assign names
   2.2 design a generative process (graphical model)
   2.3 assign (conditional) distributions to factors/arrows (use exponential families!)
3. design the **algorithm**
   3.1 consider conditional independence
   3.2 try standard methods for early experiments
   3.3 run unit-tests and sanity-checks
   3.4 identify bottlenecks, find customized approximations and refinements
4. **Test** the Setup
5. Revisit the Model and try to **improve** it, using **creativity**