Bayesian Non-negative Matrix Factorization

Greta Gašparac

February 4, 2021

Regardless of the domain, a vast amount of data is represented in 2D matrices relating to two different entity types. User-review datasets used for recommendation systems, drug-target interactions in bioinformatics, person-images in visual recognition, document-term matrices in natural language processing, to list but a few. A popular approach used to analyse this type of datasets are matrix factorization methods. The main idea behind these methods is to decompose the original matrix into two or more smaller matrices, such that their product approximates the observed entries. The lower-dimensional representation can help us reveal patterns in data, which allow for easier interpretation and better understanding.

In this handout we focus on nonnegative matrix factorization (NMF), which imposes a nonnegativity constraint on the factor matrices. We introduce the reader to the classic NMF method and two approaches to NMF in the Bayesian framework – NMF Gibbs sampler and the iterated conditional modes algorithm (ICM). In the Appendix we also provide a short refresher on Bayesian statistics, where the reader can get the background knowledge needed to understand the Bayesian NMF methods.

Contents

1 *Non-negative matrix factorization*

Non-negative matrix factorization (NMF) belongs to the family of linear dimensionality reduction techniques based on matrix factorization. Earliest formulations of this method can be found in Paatero and Tapper [1], however it was Lee and Seung [2] who popularised it. NMF approximates our original matrix $\boldsymbol{X}$ with a low-rank matrix approximation, such that $\boldsymbol{X} \approx \boldsymbol{WH}$, where $\boldsymbol{W}$ and $\boldsymbol{H}$ are element-wise non-negative matrices. Data is often non-negative in nature, so the non-negativity constraint not only limits our search space, but also provides more intuitive factors, which are easier to interpret and are often inherent to the problem. This is demonstrated in Example 1, where we can also see an illustrative comparison between NMF and another method that gives us a lower-dimensional representation of our data in latent space – principal component analysis (PCA)[1].

[1] PCA constrains the columns of $\boldsymbol{W}$ to be orthonormal and the rows of $\boldsymbol{H}$ to be ortogonal to each other.

Example 1. The following example is borrowed from Lee and Seung [2]. NMF and PCA are applied to a database of facial images. Figure 1 shows that both methods learn to represent a face as a linear combination of basis images, however, in a different way. The basis images for PCA are 'eigenfaces', which resemble distorted faces from the database, while the NMF basis images are localized features that correspond better with a part-based representation in our brains.

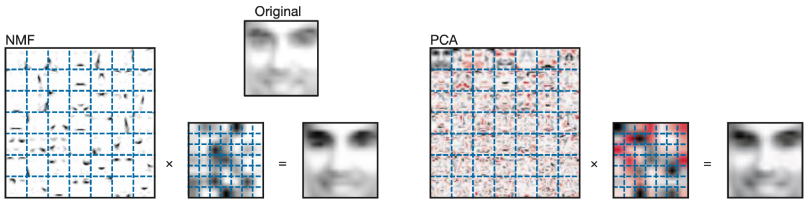

Figure 1: Non-negative matrix factorization (NMF) learns a parts-based representation of faces, whereas principal components analysis (PCA) learns holistic representations. Each method has learned a set of $r = 49$ basis images. Positive values are illustrated with black pixels and negative values with red pixels. A particular instance of a face, marked as *Original*, is approximately represented by a linear superposition of basis images.

1.1 *What is NMF?*

Let $\boldsymbol{X} \in \mathbb{R}_+^{n \times p}$ be a matrix we want to approximate. We reduce the p original dimensions to r (create a rank r approximation[2]) and approximate $\boldsymbol{X}$ with the product of non-negative matrices $\boldsymbol{W} \in \mathbb{R}_+^{n \times r}$ and $\boldsymbol{H} \in \mathbb{R}_+^{r \times p}$. Figure 2 shows a schematic representation.

[2] The rank r of the factorization is generally chosen, so that $(n+m)r < nm$.

With factor matrices we obtain a lower dimensional latent representation of our data. The columns of $\boldsymbol{W}$ are called *basis elements*. The columns of $\boldsymbol{H}$ are called *encodings*, they give coordinates of data points from the original matrix in the basis $\boldsymbol{W}$ and can be interpreted as instructions for the reconstruction of the matrix $\boldsymbol{X}$ using basis elements of $\boldsymbol{W}$.

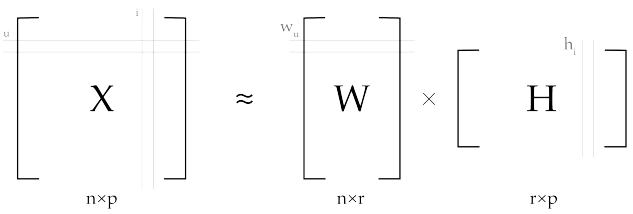

Figure 2: Schematic representation of NMF. Element x_{ui} can be approximated by multiplying the u-th row of $\boldsymbol{W}$ and the i-th column of $\boldsymbol{H}$.

Example 1 (continued). The columns of $\boldsymbol{W}$ are basis images. Columns of $\boldsymbol{H}$ are one-to-one correspondences with a face in $\boldsymbol{X}$. These encodings consist of the coefficients by which a face is represented with a linear combination of basis images. We can see from Figure 1 that both matrices are sparse. The basis images are sparse because they focus on localized features. The variability of a whole face is generated by combining these different parts. Even though all parts are used by at least one face in the database, any given face does not use all the available parts, which results in a sparse image encoding.

What makes a good latent representation? A first key aspect of linear dimensionality reduction methods is the choice of the measure to asses the quality of the approximation. The most widely used measure is the Frobenius norm of the error[3]:

$$||\boldsymbol{X} - \boldsymbol{WH}||_F^2 = \sum_{i,j} (\boldsymbol{X} - \boldsymbol{WH})_{i,j}^2.$$

This brings us to an optimization problem, which can be for a rank r factorization written as:

$$\min_{\boldsymbol{W} \in \mathbb{R}^{n \times r},\ \boldsymbol{H} \in \mathbb{R}^{r \times p}} ||\boldsymbol{X} - \boldsymbol{WH}||_F^2 \quad \text{such that} \quad \boldsymbol{W} \geq 0, \boldsymbol{H} \geq 0.$$

Note that if our results fit extremely well to the data, there is a good chance that the method overfitted to the noise in the data. Therefore base models are often extended by adding additional penalty terms to the cost function, for example L_1 sparsity norm for the factor matrices.

[3] Frobenius norm assumes Gaussian noise, other norms may be used in practice depending on the distribution.

1.2 Problems

Gillis [3] points out 3 most obvious issues that arise when we want to use NMF in practice:

NMF IS NP-HARD. The stated optimization problem is non-convex: it can be convex in $\boldsymbol{W}$ or $\boldsymbol{H}$, but not in both[4]. In practice we solve the problem using heuristic approaches.

NMF IS ILL-POSED. We have no guarantee there exists a single best unique decomposition and the resulting matrices can depend heavily on our initialization. Any matrix $\mathbf{Q}$ satisfying $\mathbf{WQ} \geq 0$ and $\mathbf{Q}^{-1}\boldsymbol{H} \geq 0$ generates a factorization, equivalent to $\boldsymbol{WH}$. If $\mathbf{Q}$ is a monomial matrix[5], this amounts to the scaling and permutation of the rank-one factors $\boldsymbol{W}(:,k)\boldsymbol{H}(k,:)$ for $1 \leq k \leq r$) and this is not an issue in practice. If $\mathbf{Q}$ is a non-monomial matrix satisfying the above conditions, equivalent factorizations generate different interpretations, which poses a problem. This issue is usually tackled by adding proper regularization terms in the objective function.

[4] Let $n = p = 1$. Our problem translates to scalars: $\min(x - wh)^2 = \min(x^2 - 2xwh + w^2h^2) = \min f_x$. We compute the Hessian:

$$\nabla^2 f_x(w,h) = \begin{bmatrix} 2h^2 & 4wh - 2x \\ 4wh - 2x & 2w^2 \end{bmatrix}.$$

The Hessian is not positive semidefinite for all x, w, $h \geq 0$ (example: $x = h = 1$, $w = 2$).

[5] *Monomial matrix* (also known as the *generalized permutation matrix*) is a matrix with the same pattern as the permutation matrix with one key difference - nonzero entry can be any nonzero value, not just 1.

CHOICE OF r. The choice of factorization rank r is an important, yet non-trivial decision we need to make in matrix factorisation. We are looking for meaningful factors while keeping in mind the dimensionality trade-off – if the value r is too high, we will overfit, if it is too low, we will not be able to find the right patterns. Some popular approaches include trial and error (try different values and pick the one that performs best for the problem at hand), estimation using the singular value decomposition (SVD) (look at the decay of the singular values of the input data matrix), and help from experts (use domain knowledge for the problem at hand).

1.3 NMF algorithms

Most of the NMF algorithms use a two-block coordinate descent scheme, that is, they optimize alternatively over one of the two factors, while keeping the other fixed. The subproblem in one factor is convex. This translates to a non-negative least squares problem (NNLS): for example, when $\boldsymbol{H}$ is fixed, we are solving $\min_{\boldsymbol{W}\geq 0} ||\boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}||_F^2$. Many algorithms exist to solve the NNLS problem[6] and NMF algorithms based on two-block coordinate descent mainly differ by which NNLS algorithm is used. The general scheme for the two-block coordinate descent is shown in Algorithm 1.

[6] Such as alternating least squares, alternating nonnegative least squares, hierarchical alternating least squares. Lee and Seung [2] proposed the following multiplicative update rule:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} \circ \frac{\boldsymbol{X}\boldsymbol{H}^T}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^T},$$
$$\boldsymbol{H} \leftarrow \boldsymbol{H} \circ \frac{\boldsymbol{W}^T\boldsymbol{X}}{\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{H}},$$

where all operations are element-wise.

Another thing to consider for the NMF implementation is also the initialization of matrices $\boldsymbol{W}$ and $\boldsymbol{H}$. They can be initialized randomly, however, there are also alternative strategies designed to give better initial estimates in the hope of converging faster to a good solution:

- Use some clustering method. Columns of $\boldsymbol{W}$ are the clustering means of the top r clusters, $\boldsymbol{H}$ is a scaling of the cluster indicator matrix (which elements belong to which cluster).
- Use SVD to find the best rank-r approximation and use the result to initialize $\boldsymbol{W}$ and $\boldsymbol{H}$.
- Initialize $\boldsymbol{W}$ with r columns of $\boldsymbol{X}$.

Once we initialize the matrices we iteratively update them until the stopping criteria is reached. As is often the practice with these kind of algorithms, the stopping criteria can be based on the value of the objective function, on the difference between consecutive iterates etc. The chosen criteria can also be combined with either a maximum number of iterations or a time limit.

Algorithm 1: Two-Block Coordinate Descent

Initialize matrices $\boldsymbol{W}^{(0)} \geq 0$ and $\boldsymbol{H}^{(0)} \geq 0$.
for *i = 1, 2, ..., stopping criteria* **do**
    $\boldsymbol{W}^{(i)} =$ update$(\boldsymbol{X}, \boldsymbol{H}^{(i-1)}, \boldsymbol{W}^{(i-1)})$
    $\boldsymbol{H}^{(i)} =$ update$(\boldsymbol{X}, \boldsymbol{H}^{(i-1)}, \boldsymbol{W}^{(i)})$

1.4 Applications

We have already suggested in the beginning that NMF proves to be useful in many fields. We have already shown an image analysis example and to satisfy the reader's curiosity we list some other interesting applications:

- Financial data mining. Drakakis et al. [4] showed how NMF can help identify underlying trends from the stock market data. The results can guide the investors to diversify their money into different clusters.

- Pattern detection in bioinformatics. Brunet et al. [5] applied NMF to cluster genes and demonstrated how the method can recover meaningful biological information, analogous to facial features we described in Example 1.

- Natural language processing. Researchers proposed the use of NMF in document clustering [6; 7]. NMF is also used for other unsupervised tasks, such as topic modeling [8].

- Environmetrics. This is the field where it all started, when Paatero and Tapper [1] initially presented the method under the name *positive matrix factorization* and offered a list of potential applications using environmental data. An interesting example is the air pollution source apportionment, where the main goal is to understand the patterns of particle pollution and its sources in order to propose sensible strategies for preserving our environment [9].

2 Bayesian Non-negative Matrix Factorization

The nonnegativity itself is an informative assumption, however, we have seen in Section 1 that NMF leaves some open questions partly unresolved. The classic NMF method yields no uncertainty estimates for its parameters, which could provide additional information. In this section we discuss the method in the Bayesian framework.

The NMF problem can be rewritten as $\boldsymbol{X} = \boldsymbol{W}\boldsymbol{H} + \boldsymbol{E}$, where $\boldsymbol{X} \in \mathbb{R}_+^{n \times p}$ is the data matrix that is factorized with $\boldsymbol{W} \in \mathbb{R}_+^{n \times r}$ and $\boldsymbol{H} \in \mathbb{R}_+^{r \times p}$, and $\boldsymbol{E} \in \mathbb{R}^{n \times p}$ is a residual matrix. Bayesian matrix factorization approaches use a likelihood distribution to capture the noise in the data and place priors over factor matrices[7]. Using Bayes' theorem the problem can be stated in the following way:

$$p(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\theta} | \boldsymbol{X}) \propto p(\boldsymbol{X} | \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\theta}) p(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\theta}),$$

where we use $\boldsymbol{\theta}$ to denote potential hyperparameters in prior and likelihood distributions. Inference is often performed using Gibbs sampling or variational Bayesian inference. We present two methods, proposed by Schmidt et al. [10]: a NMF Gibbs sampler and a maximum-a-posteriori algorithm called iterated conditional modes (ICM). The former yields posterior draws of the parameters, while the latter outputs a maximum-a-posteriori (MAP) estimate. Both use a normal likelihood and exponential priors.

[7] Note the similarity with the non-probabilistic method: the likelihood acts as a cost function and the priors are additional penalty terms over the factor matrices. They can induce sparsity or constrain the values.

2.1 NMF Gibbs sampler

Schmidt et al. [10] argue that a normal likelihood and exponential priors are suitable for a wide range of problems and allow for an efficient Gibbs sampling procedure. We therefore assume that the residuals $E_{n,p}$ are normally distributed with mean 0 and variance σ^2 and that $\boldsymbol{W}$ and $\boldsymbol{H}$ are independently exponentially distributed with scales $\alpha_{n,r}$ and $\beta_{r,p}$. We choose an inverse gamma distribution with shape k and scale θ as prior for the σ^2 (the noise variance)[8].

By multiplying the likelihood and the priors we get the posterior, which can be maximized to yield an estimate of the two factor matrices. However, we are interested in estimating the marginal density of the factors. Computing the marginals by integrating the posterior is difficult and tedious, but since we can calculate the full conditional posterior distributions, we exploit the benefits of Gibbs sampling.

We need to be able to draw samples from the following distributions:

$$p(W_{n,r} | \boldsymbol{X}, \boldsymbol{W}_{\setminus(n,r)}, \boldsymbol{H}, \sigma^2), \quad p(H_{r,p} | \boldsymbol{X}, \boldsymbol{W}, \boldsymbol{H}_{\setminus(r,p)}, \sigma^2), \quad p(\sigma^2 | \boldsymbol{X}, \boldsymbol{W}, \boldsymbol{H}),$$

[8] Likelihood:

$$p(\boldsymbol{X} | \boldsymbol{W}, \boldsymbol{H}, \sigma^2) = \prod_{n,p} \mathcal{N}(\boldsymbol{X}_{n,p}; (\boldsymbol{W}\boldsymbol{H})_{n,p}, \sigma^2),$$

priors:

$$p(\boldsymbol{W}) = \prod_{n,r} \mathcal{E}(\boldsymbol{W}_{n,r}; \alpha_{n,r}),$$
$$p(\boldsymbol{H}) = \prod_{r,p} \mathcal{E}(\boldsymbol{H}_{r,p}; \beta_{r,p}),$$
$$p(\sigma^2) = \mathcal{G}^{-1}(\sigma^2; k, \theta).$$

where $\boldsymbol{W}_{\setminus(n,r)}$ denotes all elements of $\boldsymbol{W}$ except $W_{n,r}$ and same logic also applies to $\boldsymbol{H}_{\setminus(r,p)}$. Using Bayes' rule[9] we can see that the conditional densities of $\boldsymbol{W}$ and $\boldsymbol{H}$ are proportional to a one sided truncated[10] normal distribution with lower bound $a = 0$[11]. The conditional density of σ^2 is an inverse-gamma distribution. The posterior can be approximated by sequentially sampling from these conditional densities.

Algorithm 2 shows an efficient implementation. Since the elements in each column of $\boldsymbol{W}$ (row of $\boldsymbol{H}$) are conditionally independent, we can sample an entire column (row) simultaneously. The precomputed matrices $\boldsymbol{C}$, $\boldsymbol{D}$, $\boldsymbol{E}$ and $\boldsymbol{F}$ help us avoid computing large matrix products of size $n \times p$. Notation $\boldsymbol{W}_{:,\setminus j}$ denotes the submatrix of $\boldsymbol{W}$ that consists of all columns except for the j-th. Our implementation in `R` can be found in Supplementary materials.

Algorithm 2: NMF Gibbs sampler

Initialize $\boldsymbol{W}^{(0)}$ and $\boldsymbol{H}^{(0)}$.
for $i = 1{:}N$ **do**
  $\boldsymbol{W} \leftarrow \boldsymbol{W}^{(i-1)}$, $\boldsymbol{H} \leftarrow \boldsymbol{H}^{(i-1)}$
  $\boldsymbol{C} = \boldsymbol{H}\boldsymbol{H}^T$, $\boldsymbol{D} = \boldsymbol{X}\boldsymbol{H}^T$
  for $j = 1$ *to* r **do**
    $\boldsymbol{W}_{:,j} \leftarrow \mathcal{TN}(\boldsymbol{w}_j, \frac{\sigma^2}{C_{j,j}})$ // $\boldsymbol{w}_j = \frac{\boldsymbol{D}_{:,j} - \boldsymbol{W}_{:,\setminus j}\boldsymbol{C}_{\setminus j,j} - \boldsymbol{\alpha}_{:,j}\sigma^2}{C_{j,j}}$
  $\sigma^2 \leftarrow \mathcal{G}^{-1}(\frac{np}{2} + k + 1, \mathcal{X} + \theta + \xi)$
    // $\mathcal{X} = \frac{1}{2}\sum_{n,p} X_{n,p}^2$, $\xi = \frac{1}{2}\sum_{n,r} W_{n,r}(\boldsymbol{W}\boldsymbol{C} - 2\boldsymbol{D})_{n,r}$
  $\boldsymbol{E} = \boldsymbol{W}^T\boldsymbol{W}$, $\boldsymbol{F} = \boldsymbol{W}^T\boldsymbol{X}$
  for $j = 1$ *to* r **do**
    $\boldsymbol{H}_{j,:} \leftarrow \mathcal{TN}(\boldsymbol{h}_j, \frac{\sigma^2}{E_{j,j}})$ // $\boldsymbol{h}_j = \frac{\boldsymbol{F}_{j,:} - \boldsymbol{E}_{j,\setminus j}\boldsymbol{B}_{\setminus j,:} - \boldsymbol{\beta}_{j,:}\sigma^2}{E_{j,j}}$
  $\boldsymbol{W}^{(i)} \leftarrow \boldsymbol{W}$, $\boldsymbol{H}^{(i)} \leftarrow \boldsymbol{H}$
Output: $\{\boldsymbol{W}^{(i)}, \boldsymbol{H}^{(i)}\}_{i=1}^N$

2.2 Iterated conditional modes (ICM)

The NMF Gibbs sampler gives us the full posterior of our parameters. With the conditional densities in place, Schmidt et al. [10] also proposed a maximum-a-posteriori (MAP) algorithm called iterated conditional modes. Its underlying logic remains the same, but instead of drawing random samples from the conditionals, we take conditional mode for each parameter. After a number of interations the algorithm converges to a local maximum of the joint posterior density. Rather than a full posterior distribution, our output is a MAP point estimate. The whole process is shown in Algorithm 3, where P_+ denotes a function that sets negative elements of its argument to some small value ϵ[12].

[9] The derivation can be found in Brouwer et al. [11], Supplementary materials, Section 1.1.

[10] In the original paper the authors mistakenly defined it as a rectified normal distribution. Later they indicated that what they refer to as a rectified Gaussian is actually a truncated Gaussian.

[11] In our case this is a normal distribution with zero density below $x = 0$ and renormalised to integrate to 1:

$$\mathcal{TN}(x; \mu, \sigma^2) = \begin{cases} \frac{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{1 - \Phi\left(-\frac{\mu}{\sigma}\right)} & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

[12] The original proposed algorithm sets the negative elements to 0, however, Brouwer [12] argues that ICM often converges to solutions where multiple columns are all zeros, which leads to poor approximations. They address this issue by resetting zeros to a small positive value like 0.1.

Algorithm 3: ICM

Initialize $\boldsymbol{W}$ and $\boldsymbol{H}$.
while *stopping criteria not reached* **do**
  $\boldsymbol{C} = \boldsymbol{H}\boldsymbol{H}^T$, $\boldsymbol{D} = \boldsymbol{X}\boldsymbol{H}^T$
  for $j = 1$ *to* r **do**
    $\boldsymbol{W}_{:,j} \leftarrow P_+(w_j, \epsilon_1)$ // $\boldsymbol{w}_j = \frac{\boldsymbol{D}_{:,j} - \boldsymbol{W}_{:,\setminus j}\boldsymbol{C}_{\setminus j,j} - \boldsymbol{\alpha}_{:,j}\sigma^2}{C_{j,j}}$
  $\sigma^2 = \frac{\theta + \mathcal{X} + \xi}{\frac{np}{2} + k + 1}$ // $\mathcal{X} = \frac{1}{2}\sum_{n,p} X_{n,p}^2$, $\xi = \frac{1}{2}\sum_{n,r} W_{n,r}(\boldsymbol{W}\boldsymbol{C} - 2\boldsymbol{D})_{n,r}$
  $\boldsymbol{E} = \boldsymbol{W}^T\boldsymbol{W}$, $\boldsymbol{F} = \boldsymbol{W}^T\boldsymbol{X}$
  for $j = 1$ *to* r **do**
    $\boldsymbol{H}_{j,:} \leftarrow P_+(h_j, \epsilon_2)$ // $\boldsymbol{h}_j = \frac{\boldsymbol{F}_{j,:} - \boldsymbol{E}_{j,\setminus j}\boldsymbol{B}_{\setminus j,:} - \boldsymbol{\beta}_{j,:}\sigma^2}{E_{j,j}}$
Output: $\boldsymbol{W}, \boldsymbol{H}$

A An Overview of Bayesian Statistics and Gibbs Sampling

[13]

The main difference between the frequentist and the Bayesian inference is how we treat the parameters. Given some data y the frequentists want to find an unknown constant θ that would maximize the likelihood $p(y|\theta)$. Bayesian statisticians use the same models (that is, the same likelihoods $p(y|\theta)$), but the parameters θ are treated as random variables, which allows us to express uncertainty.

A.1 Bayesian Modeling

The cornerstone of Bayesian statistics is the Bayes' theorem:

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta} \propto p(y|\theta)p(\theta).$$

Let us explain each term:

- $p(y|\theta)$ denotes the *likelihood*, which is determined by our choice of model.
- $p(\theta)$ is the *prior distribution*. It expresses our uncertainty about the parameters before we see the data.
- $p(\theta|y)$ is referred to as the *posterior distribution*. It expresses our uncertainty about the parameters after we see the data and can be used to answer any probabilistic question regarding the parameters.
- $p(y)$ serves as a normalization factor, which is usually difficult to compute. However, since it is not a function of θ it does not effect the shape of the posterior, so we can often avoid computing it.

The Bayes' theorem tells us how our beliefs should change after seeing new information. See example in Figure 3.
Sometimes we can compute the posterior distribution analitically, but since most of the models do not have nice conjugate posteriors[14], more often than not we resort to MCMC methods. One of these methods is also Gibbs sampling.

A.2 Gibbs Sampling

Gibbs sampling is a specific case of the Metropolis-Hastings algorithm, where the proposals are always accepted[15]. However, historically speaking, Metropolis-Hastings was never very useful in practice. It was Gibbs sampling that truly revolutionized the field of Bayesian statistics in the 80s and the 90s as it was the basis for the first generation of Bayesian stats tools (BUGS, WinBUGS, JAGS).

[13] The contents of this section were inspired by prof. E. Štrumbelj's lecture notes and by Hoff [13].

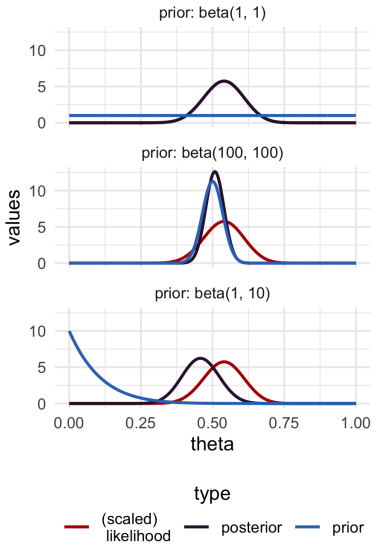

Figure 3: We model 50 coin tosses with a binomial-beta model. When we have no prior opinion (the prior distribution is flat) our updated belief is based on the data, we see that the posterior sits right on top of the likelihood. In the second example we can see the opposite: we strongly believe that the value of θ is around 0.5 and the likelihood seems more uncertain, so it has little impact on our updated beliefs. The third plot shows an example where our prior beliefs completely disagree with the data and are therefore shifted more towards what we observed.

[14] **Definition (Conjugate)** *A class P of prior distributions for θ is called conjugate for a sampling model $p(y|\theta)$ if*

$$p(\theta) \in P \Rightarrow p(\theta|y) \in P.$$

[15] Let x_i denote the i-th variable, let x_{-i} denote the set of all variables except x_i, let x_i^* be the proposed value. The proposal distribution q for Gibbs sampling are the full conditional distributions: $q = p(x_i^*|x_{-i})$. The acceptance rate α then equals to 1:

$$\alpha = \frac{p(x_i^*, x_{-i})p(x_i|x_{-i})}{p(x_i, x_{-i})p(x_i^*|x_{-i})} =$$
$$= \frac{p(x_i^*|x_{-i})p(x_{-i})p(x_i|x_{-i})}{p(x_i|x_{-i})p(x_{-i})p(x_i^*|x_{-i})} = 1.$$

Before we describe this sampling procedure, let's take a look at an interesting property of the full conditional distributions, which stands behind the main idea of Gibbs sampling. It turns out[16] that a joint distribution $p(x_1, x_2, ..., x_n)$ can be characterized by its complete set of full conditional distributions: $p(x_1|x_2, x_3, ..., x_n)$, $p(x_2|x_1, x_3, ..., x_n)$, ..., $p(x_n|x_1, x_2, ..., x_{n-1})$. In other words, in order to sample from the joint distribution it is enough to know how to sample from the full conditional distributions and this is exactly what Gibbs sampling does.

[16] This is a simplification of the Hammersley-Clifford theorem [14].

We use Gibbs sampling when the joint posterior distribution $p(\theta_1, \theta_2, ..., \theta_n)$ is difficult to sample from directly and full conditional distributions of the parameters are known. In practice it is relatively easy to work with the full conditionals, since we usually partition the parameter space in a way that gives us univariate conditionals and we know lots of algorithms for sampling from these distributions. Algorithm 4 shows a generic Gibbs sampler. We can notice a similar idea to the block coordinate descent algorithm – both algorithms fix all variables except for the one being updated, and take turns updating individual variables.

Algorithm 4: Gibbs sampler

Initialize $\boldsymbol{x}^{(0)}$.
for $i = 1{:}N$ **do**
  $x_1^{(i)} \sim p(x_1|x_2^{(i-1)}, x_3^{(i-1)}, ..., x_n^{(i-1)})$
  $x_2^{(i)} \sim p(x_2|x_1^{(i)}, x_3^{(i-1)}, ..., x_n^{(i-1)})$
  $\vdots$
  $x_n^{(i)} \sim p(x_n|x_1^{(i)}, x_2^{(i)}, ..., x_{n-1}^{(i)})$
Output: $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$

Example 2. Suppose we want to sample from a bivariate normal distribution $\boldsymbol{X} = \binom{x_A}{x_B} \sim N\left(\binom{\mu_A}{\mu_B}, \begin{pmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{pmatrix}\right)$ and all we know is how to sample from a univariate normal distribution. Since we know that the conditional distributions in this case are univariate normal[17], we can use Gibbs sampling to simulate a Markov chain that will converge to a bivariate normal. Let $\mu_A = \mu_B = 0$, $\Sigma_{AA} = \Sigma_{BB} = 1$ and $\Sigma_{AB} = \Sigma_{BA} = 0.5$. We implement the algorithm in `R` (see code below).

[17] $X_A|X_B \sim N(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})$

```
> set.seed(0)
> s <- 1000
> mu_A <- rep(0, s)
> mu_B <- rep(0, s)
> for(i in 2:s) {
+   mu_A[i] <- rnorm(1, 0.5 * mu_B[i-1], 0.75)
+   mu_B[i] <- rnorm(1, 0.5 * mu_A[i], 0.75)
+ }
```

Figure 4 compares 1000 Gibbs samples to 1000 samples from function `MASS::mvrnorm`. We can see that the results are similar. Figure 5 shows that we converge to the stationary distribution even if our starting position is lousy. In practice we usually discard the first m samples in order for the Markov chain to eliminate the effects of the initial state and stabilize. This is called the *adaptation/burn-in period*.

Output of a Gibbs sampler is a Markov chain: a dependent sequence of vectors: $\theta^{(1)}, \theta^{(2)}, ..., \theta^{(s)}$, where the Markov property $P(\theta_n|\theta_{n-1}, ..., \theta_1) = P(\theta_n|\theta_{n-1})$ holds. After a reasonable amount of iterations we expect our chain to converge (i.e. reach the stationary distribution, which is our target distribution). However, there is no guarantee, so we can employ some of the well-known techniques to recognize potential problems: traceplot, autocorrelation, effective sample size.

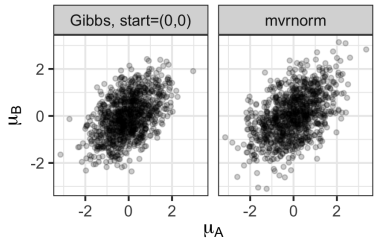

Figure 4: 1000 samples from the bivariate normal distribution, comparison of Gibbs samples (right) and samples obtained with function `MASS::mvrnorm`.

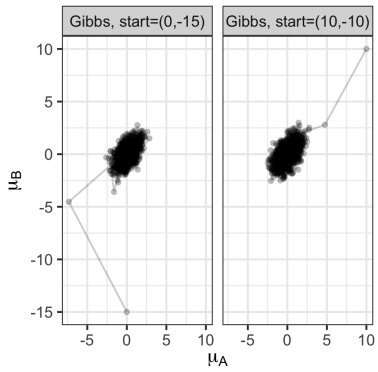

Figure 5: 1000 Gibbs samples from the bivariate normal distribution. We can see that we converge to the stationary distribution even when the starting points are lousy.

References

[1] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[2] Daniel Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91, 11 1999. doi: 10.1038/44565.

[3] Nicolas Gillis. The why and how of nonnegative matrix factorization. *Regularization, Optimization, Kernels, and Support Vector Machines*, 12, 01 2014.

[4] Konstantinos Drakakis, Scott Rickard, Ruairí de Fréin, and Andrzej Cichocki. Analysis of financial data using non-negative matrix factorization. *International Mathematical Forum*, 3, 01 2008. doi: 10.12988/imf.

[5] Jean-Philippe Brunet, Pablo Tamayo, Todd R. Golub, and Jill P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences*, 101(12):4164–4169, 2004. ISSN 0027-8424.

[6] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. SIGIR '03, page 267–273. Association for Computing Machinery, 2003. ISBN 1581136463. doi: 10.1145/860435.860485.

[7] V.Paul Pauca, Farial Shahnaz, Michael Berry, and Robert Plemmons. Text mining using non-negative matrix factorizations. 04 2004. doi: 10.1137/1.9781611972740.45.

[8] J. Choo, C. Lee, C. K. Reddy, and H. Park. Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1992–2001, 2013. doi: 10.1109/TVCG.2013.212.

[9] Alexander Thiem, U. Schlink, X. Pan, Min Hu, A. Peters, A. Wiedensohler, S. Breitner, J. Cyrys, B. Wehner, Carolin Rösch, and U. Franck. Using non-negative matrix factorization for the identification of daily patterns of particulate air pollution in Beijing during 2004-2008. *Atmospheric Chemistry and Physics*, 12:13015–13052, 2012.

[10] Mikkel N. Schmidt, Ole Winther, and Lars Kai Hansen. Bayesian non-negative matrix factorization. In Tülay Adali, Christian Jutten, João Marcos Travassos Romano, and Allan Kardec Barros, editors, *Independent Component Analysis and Signal Separation*. Springer Berlin Heidelberg, 2009.

[11] Thomas Brouwer, Jes Frellsen, and Pietro Lio. Fast Bayesian Non-Negative Matrix Factorisation and Tri-Factorisation. 10 2016.

[12] Thomas Alexander Brouwer. *Bayesian matrix factorisation: inference, priors, and data integration*. PhD thesis, University of Cambridge, 2017.

[13] Peter D. Hoff. *A First Course in Bayesian Statistical Methods*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 0387922997.

[14] J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. 1971.