Then the priors terms are:

$$\alpha \rho ||W||_1 + \alpha \rho ||H||_1 + \frac{\alpha(1-\rho)}{2} ||W||_{Fro}^2 + \frac{\alpha(1-\rho)}{2} ||H||_{Fro}^2$$

and the regularized objective function is:

$$\frac{1}{2}||X - WH||_{Fro}^2 + \alpha \rho ||W||_1 + \alpha \rho ||H||_1 + \frac{\alpha(1-\rho)}{2} ||W||_{Fro}^2 + \frac{\alpha(1-\rho)}{2} ||H||_{Fro}^2$$

`NMF` regularizes both W and H. The public function `non_negative_factorization` allows a finer control through the `regularization` attribute, and may regularize only W, only H, or both.

Examples:

- *Faces dataset decompositions*
- *Topic extraction with Non-negative Matrix Factorization and Latent Dirichlet Allocation*

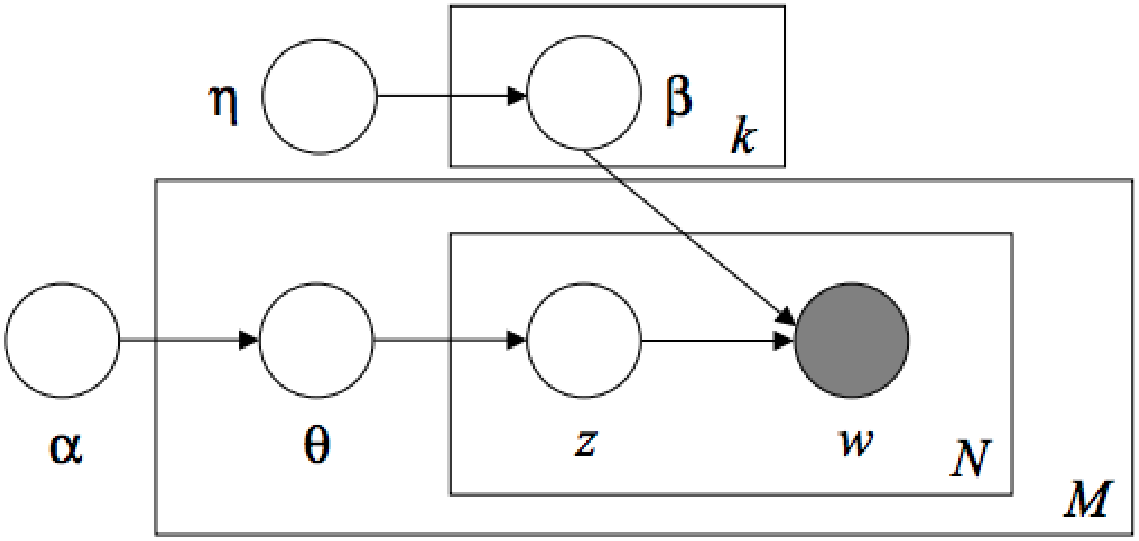
References:

- "Learning the parts of objects by non-negative matrix factorization" D. Lee, S. Seung, 1999
- "Non-negative Matrix Factorization with Sparseness Constraints" P. Hoyer, 2004
- "Projected gradient methods for non-negative matrix factorization" C.-J. Lin, 2007
- "SVD based initialization: A head start for nonnegative matrix factorization" C. Boutsidis, E. Gallopoulos, 2008
- "Fast local algorithms for large scale nonnegative matrix and tensor factorizations." A. Cichocki, P. Anh-Huy, 2009

Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation is a generative probabilistic model for collections of discrete dataset such as text corpora. It is also a topic model that is used for discovering abstract topics from a collection of documents.

The graphical model of LDA is a three-level Bayesian model:



When modeling text corpora, the model assumes the following generative process for a corpus with D documents and K topics:

1. For each topic k, draw $\beta_k \sim Dirichlet(\eta),\; k = 1...K$
2. For each document d, draw $\theta_d \sim Dirichlet(\alpha),\; d = 1...D$
3. For each word i in document d:

1. Draw a topic index $z_{di} \sim Multinomial(\theta_d)$
2. Draw the observed word $w_{ij} \sim Multinomial(beta_{z_{di}}.)$

For parameter estimation, the posterior distribution is:

$$p(z, \theta, \beta|w, \alpha, \eta) = \frac{p(z, \theta, \beta|\alpha, \eta)}{p(w|\alpha, \eta)}$$

Since the posterior is intractable, variational Bayesian method uses a simpler distribution $q(z, \theta, \beta|\lambda, \phi, \gamma)$ to approximate it, and those variational parameters λ, ϕ, γ are optimized to maximize the Evidence Lower Bound (ELBO):

$$log\,P(w|\alpha, \eta) \geq L(w, \phi, \gamma, \lambda) \overset{\triangle}{=} E_q[log\,p(w, z, \theta, \beta|\alpha, \eta)] - E_q[log\,q(z, \theta, \beta)]$$

Maximizing ELBO is equivalent to minimizing the Kullback-Leibler(KL) divergence between $q(z, \theta, \beta)$ and the true posterior $p(z, \theta, \beta|w, \alpha, \eta)$.

`LatentDirichletAllocation` implements online variational Bayes algorithm and supports both online and batch update method. While batch method updates variational variables after each full pass through the data, online method updates variational variables from mini-batch data points.

Note: Although online method is guaranteed to converge to a local optimum point, the quality of the optimum point and the speed of convergence may depend on mini-batch size and attributes related to learning rate setting.

When `LatentDirichletAllocation` is applied on a “document-term” matrix, the matrix will be decomposed into a “topic-term” matrix and a “document-topic” matrix. While “topic-term” matrix is stored as `components_` in the model, “document-topic” matrix can be calculated from `transform` method.

`LatentDirichletAllocation` also implements `partial_fit` method. This is used when data can be fetched sequentially.

Examples:

- *Topic extraction with Non-negative Matrix Factorization and Latent Dirichlet Allocation*

References:

- "Latent Dirichlet Allocation" D. Blei, A. Ng, M. Jordan, 2003
- "Online Learning for Latent Dirichlet Allocation" M. Hoffman, D. Blei, F. Bach, 2010
- "Stochastic Variational Inference" M. Hoffman, D. Blei, C. Wang, J. Paisley, 2013

3.2.6 Covariance estimation

Many statistical problems require at some point the estimation of a population's covariance matrix, which can be seen as an estimation of data set scatter plot shape. Most of the time, such an estimation has to be done on a sample whose properties (size, structure, homogeneity) has a large influence on the estimation's quality. The *sklearn.covariance* package aims at providing tools affording an accurate estimation of a population's covariance matrix under various settings.

We assume that the observations are independent and identically distributed (i.i.d.).

Empirical covariance

The covariance matrix of a data set is known to be well approximated with the classical *maximum likelihood estimator* (or "empirical covariance"), provided the number of observations is large enough compared to the number of features (the variables describing the observations). More precisely, the Maximum Likelihood Estimator of a sample is an unbiased estimator of the corresponding population covariance matrix.

The empirical covariance matrix of a sample can be computed using the `empirical_covariance` function of the package, or by fitting an `EmpiricalCovariance` object to the data sample with the `EmpiricalCovariance.fit` method. Be careful that depending whether the data are centered or not, the result will be different, so one may want to use the `assume_centered` parameter accurately. More precisely if one uses `assume_centered=False`, then the test set is supposed to have the same mean vector as the training set. If not so, both should be centered by the user, and `assume_centered=True` should be used.

Examples:

- See *Shrinkage covariance estimation: LedoitWolf vs OAS and max-likelihood* for an example on how to fit an `EmpiricalCovariance` object to data.

Shrunk Covariance

Basic shrinkage

Despite being an unbiased estimator of the covariance matrix, the Maximum Likelihood Estimator is not a good estimator of the eigenvalues of the covariance matrix, so the precision matrix obtained from its inversion is not accurate.