Training sparse coding with maximum likelihood is intractable. Instead, the training alternates between encoding the data and training the decoder to better reconstruct the data given the encoding. This approach will be justified further as a principled approximation to maximum likelihood later, in section 19.3.

For models such as PCA, we have seen the use of a parametric encoder function that predicts $\boldsymbol{h}$ and consists only of multiplication by a weight matrix. The encoder that we use with sparse coding is not a parametric encoder. Instead, the encoder is an optimization algorithm, that solves an optimization problem in which we seek the single most likely code value:

$$\boldsymbol{h}^* = f(\boldsymbol{x}) = \arg\max_{\boldsymbol{h}} p(\boldsymbol{h} \mid \boldsymbol{x}). \tag{13.15}$$

When combined with equation 13.13 and equation 13.12, this yields the following optimization problem:

$$\arg\max_{\boldsymbol{h}} p(\boldsymbol{h} \mid \boldsymbol{x}) \tag{13.16}$$

$$= \arg\max_{\boldsymbol{h}} \log p(\boldsymbol{h} \mid \boldsymbol{x}) \tag{13.17}$$

$$= \arg\min_{\boldsymbol{h}} \lambda ||\boldsymbol{h}||_1 + \beta ||\boldsymbol{x} - \boldsymbol{W}\boldsymbol{h}||_2^2, \tag{13.18}$$

where we have dropped terms not depending on $\boldsymbol{h}$ and divided by positive scaling factors to simplify the equation.

Due to the imposition of an L^1 norm on $\boldsymbol{h}$, this procedure will yield a sparse $\boldsymbol{h}^*$ (See section 7.1.2).

To train the model rather than just perform inference, we alternate between minimization with respect to $\boldsymbol{h}$ and minimization with respect to $\boldsymbol{W}$. In this presentation, we treat β as a hyperparameter. Typically it is set to 1 because its role in this optimization problem is shared with λ and there is no need for both hyperparameters. In principle, we could also treat β as a parameter of the model and learn it. Our presentation here has discarded some terms that do not depend on $\boldsymbol{h}$ but do depend on β. To learn β, these terms must be included, or β will collapse to 0.

Not all approaches to sparse coding explicitly build a $p(\boldsymbol{h})$ and a $p(\boldsymbol{x} \mid \boldsymbol{h})$. Often we are just interested in learning a dictionary of features with activation values that will often be zero when extracted using this inference procedure.

If we sample $\boldsymbol{h}$ from a Laplace prior, it is in fact a zero probability event for an element of $\boldsymbol{h}$ to actually be zero. The generative model itself is not especially sparse, only the feature extractor is. Goodfellow *et al.* (2013d) describe approximate