Hierarchical mixture of linear regressions for multivariate spectroscopic calibration
An application for NIR calibration

Chenhao Cui and Tom Fearn

Department of Statistical Science,
University College London,London, WC1E 6BT, U.K.

Email:chenhao.cui.14@ucl.ac.uk; Tel:+44 747 838 3032

Abstract

This paper investigates the use of the hierarchical mixture of linear regressions (HMLR) and variational inference for multivariate spectroscopic calibration. The performance of HMLR is compared to the classical methods: partial least squares regression (PLSR), and PLS embedded locally weighted regression (LWR) on three different NIR datasets, including a publicly accessible one. In these tests, HMLR outperformed the other two benchmark methods. Compared to LWR, HMLR is parametric, which makes it interpretable and easy to use. In addition, HMLR provides a novel calibration scheme to build a two-tier PLS regression model automatically. This is especially useful when the investigated constituent covers a large range.

Keywords: Near-Infrared spectroscopy; Multivariate calibration; Partial least squares regressions; Hierarchical mixture of linear regression; Expectation maximization; Variational inference

1 Background

Partial least squares regression (PLSR) [1] and principal component regression (PCR) [2] have been applied to NIR calibration for decades. The beauty of these methods is that noisy and mutually dependent NIR spectra are first dimensionally reduced to a few factors, which ensures the following multiple linear regression (MLR) [3] process is properly regularized. However, the limitations of these regression schemes are also obvious: both the dimension reduction and the regression are linear, which means the final derived model is a linear combination of all input variables. This means neither of them can tackle non-linear effects. Lack of non-linearity is critical especially when the target constituent covers a large range. When a single linear model is forced to fit over a wide range, significant biases often appear in the two tails.

Non-linear regression methods were introduced to save prediction models from such heavy biases. Popular examples are support vector machine (SVM) [4] and Gaussian process regression (GPR) [5]. Some recent studies have proven these methods can be used to improve prediction accuracy [6] [7] [8]. Very similar to kernel-based regression schemes, local linear regression methods such as locally weighted regression (LWR) were also proposed to give accurate and unbiased prediction across the range [9]. These methods indeed break the limitations of linear regression. However, they are not always practically applicable. First of all, to build such kinds of models, a large data set is required. The methods fail if there are not adequate training instances in the neighborhood of the unknown observation. Secondly, using these methods for out-of-sample prediction requires information of the whole training set, including the spectra (or equivalently, the gram matrix in the kernel methods) and corresponding reference values. The size of the model grows with the size of the training set (i.e., these methods are all non-parametric). Even if the storage of the system is not a limitation, computational cost for making a prediction might be an intolerable issue. Computational cost also makes it extremely difficult to implement these methods in a high-speed on-line system. Finally, these models are complicated for training. Training an LWR, for example, requires prior knowledge of the number of neighbors, distance measurement, and weight function. Even can be tuned by cross-validation, these hyper-parameters are sensitive to the training set variation, which makes them less robust and stable than PLSR and more difficult to scale.

Another good solution is to build a two-tier PLSR model. In this kind of scheme, the training set will be divided into two groups: high or low in target value. Two PLSR models are fitted on each subset separately. This regression method can also remove the bias in the two tails. However, it is unclear how to split the data set, and how to choose the correct component model for making the prediction. Improper segmentation (in the calibration) and selection of model (in the prediction) will affect prediction accuracy, especially for the samples near the segmentation boundary

The method proposed in this study, hierarchical mixture of linear regressions (HMLR) [10] solves these issues. The calibration method assumes there are two or more underlying linear models behind the dataset. By using the expectation maximisation (EM) and the variational inference, the algorithm will find the most sensible component models automatically through an iterative optimisation process. A set of gating functions will also be trained simultaneously to assign individual observation into the correct component model. In the end, the trained model contains several independent linear models (e.g. PLSR or PCR) for different subsets of the training set and a set of gating functions. Compared to GPR and LWR, it is entirely parametric, which makes it possible to interpret and easy to use. It is also simple for training, only the number of PLS factors and spectral preprocessing methods need to be determined.

2 Methodology

2.1 Dimension reduction

For all calibration methods, the target is to train a regression model from a set of N training instances $\{\boldsymbol{x_n}, y_n\}$, where $\boldsymbol{x_n}$ is the original NIR spectrum and y_n is the lab measurement of the target constituent. Since NIR spectra are mostly high dimensional and mutually dependent, dimension reduction is applied to the original space. Partial least squares (PLS) and principal component analysis (PCA) are the most popular data shrinkage methods [11] [12]. In this research PLS with NIPALS decomposition was used to transform original space $\boldsymbol{x_n}$ into a selection of latent variables $\boldsymbol{\phi_n}$ [13]. PLSR is a result of direct MLR of $\boldsymbol{\phi_n}$ on y_n. HMLR and the other benchmark method LWR were also learned on the transformed training set $\{\boldsymbol{\phi_n}, y_n\}$.

2.2 Hierarchical mixture of linear regression

2.2.1 Overview of the graphical model

The structure of HMLR can be illustrated by fig.1. Red nodes in the graph represent a set of sequential gating functions, and black nodes are component linear regression models. When the new observation $\boldsymbol{x_{n+1}}$ is obtained, it flows through the decision tree, each gating node $\boldsymbol{v_i}$ can decide whether the corresponding component model M_i should be used to predict the observation.

There are two different prediction strategies: hard split and soft mix. Hard split means one and only one end model will be used to make the prediction. In the soft mix scheme, gating nodes are probabilistic. Outputs of gating nodes are not binary decisions, but a set of probabilities of using the corresponding component models. In the end, predictions from all component models will be weighted averaged out by the likelihood. In this study, soft mix of the component models was used to make the prediction.

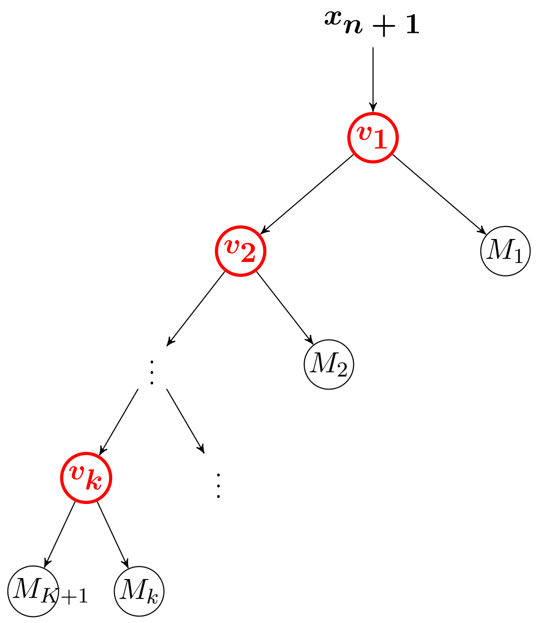


Figure 1: Nested structure for integrating linear regression models. Red nodes: gating nodes, determining weights of component linear models; Black nodes: component models, each end model is an independent linear expert

The critical part is how to learn the gating functions $\boldsymbol{V}$ and component linear models $\boldsymbol{M}$ from a given training set. For a total number of N training instances $\{\boldsymbol{\phi_n}, y_n\}$, assume there are $K+1$ underlying linear regression functions $\boldsymbol{w_i}, i = 1, 2, \ldots, K+1$. Gating functions $\boldsymbol{V}$ indicate the correct component model (or the weight of each component model) for input $\boldsymbol{\phi_n}$. A labeling variable $\boldsymbol{z_n}$ is assigned to each of the training instances. In this study, $\boldsymbol{z_n}$ is binary in training (i.e., either 0 or 1 depending on whether the corresponding end model is used). $\boldsymbol{z_n}$ is a vector of length $K+1$, corresponding to $K+1$ component models. The goal of the whole training process is to find a set of parameters of $\{\boldsymbol{V}, \boldsymbol{W}\}$ that can explain the training set. It is helpful to put appropriate regularization on these parameters to avoid over-fit.

Here Bayesian directed acyclic graph (DAG) [14] can be introduced to describe dependency of different parameters and variables in the calibration process. The DAG is shown in fig.2. The rectangular box represents N training instances. The connection from the black dot X_n to ϕ_n is deterministic transformation, which is a PLS data shrinkage process. $\{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$ are parameters of the regularization on coefficients $\{\boldsymbol{W}, \boldsymbol{V}\}$. $\boldsymbol{\tau}$ is a vector of length of $K+1$, indicates the precision of reference for each Bayesian linear regression.

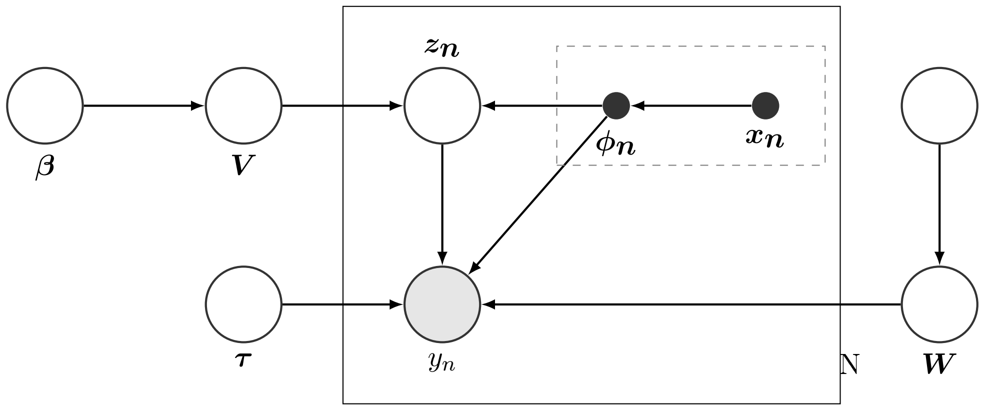


Figure 2: Directed graphical model for mixture of linear experts.

2.2.2 Loss function: complete data log-likelihood

As introduced, the goal of training is to find a set of parameters that can explain the training set. The complete data log-likelihood (CDLL) is introduced as the loss function. According to the graphical model in fig.2, CDLL can be described by the following equation:

$$p(\{\boldsymbol{\phi_n}, \boldsymbol{z_n}, y_n\}_{n=1,\ldots,N}, \boldsymbol{W}, \boldsymbol{V}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\tau}) = \prod_{n}^{N} \{p(y_n|\boldsymbol{W}, \boldsymbol{z_n}, \boldsymbol{\phi_n}, \boldsymbol{\tau}) p(\boldsymbol{z_n}|\boldsymbol{V}, \boldsymbol{\phi_n})\} \\ \times p(\boldsymbol{V}|\boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) p(\boldsymbol{\beta}) p(\boldsymbol{\tau}) \tag{1}$$

where $p(y_n|\boldsymbol{z_n}, \boldsymbol{\phi_n}, \boldsymbol{\tau}, \boldsymbol{W})$ is a Gaussian distribution centered at the prediction mean with a precision vector of $\boldsymbol{\tau}$. $p(\boldsymbol{z_n}|\boldsymbol{V}, \boldsymbol{\phi_n})$ is a Sigmoid distribution, which realizes the soft gating scheme. The following equations formulate the conditional probability:

$$p(y_n|\boldsymbol{z_n}, \boldsymbol{\phi_n}, \boldsymbol{\tau}, \boldsymbol{W}) = \prod_{i=1}^{K+1} \mathcal{N}(y_n|\boldsymbol{w_i}^T \boldsymbol{\phi_n}, \tau_i^{-1}\boldsymbol{I})^{z_{ni}} \tag{2}$$

$$
\begin{aligned}
p(\boldsymbol{z_n}|\boldsymbol{V},\boldsymbol{\phi_n}) &= \prod_{i=1}^{K}\{\sigma(\boldsymbol{v_i^T}\boldsymbol{\phi_n})^{z_{ni}}[1-\sigma(\boldsymbol{v_i^T}\boldsymbol{\phi_n})]^{1-z_{ni}}\}^{t(i,\boldsymbol{z_n})} \\
&= \prod_{i=1}^{K}\{\exp(z_{ni}\boldsymbol{v_i^T}\boldsymbol{\phi_n})\sigma(-\boldsymbol{v_i^T}\boldsymbol{\phi_n})\}^{t(i,\boldsymbol{z_n})}
\end{aligned} \tag{3}
$$

The logistic function $\sigma(x)$ is defined as:

$$
\sigma(x) = \frac{1}{1+\exp(-x)} \tag{4}
$$

Here a binary gating indicator function $t(i, z_n)$ is introduced. $t(i, z_n)$ indicates whether the i^{th} node is active. For example, if gating node 1 determines that the first end model is the reached, then the rest of the regression tree will not be activated. $t(i, z_n)$ can be formulated as:

$$
t(i,\boldsymbol{z_n}) = \begin{cases} 1 & i = 1 \\ 1-\sum_{j=1}^{j<i} z_{nj} & i \geq 2 \end{cases} \tag{5}
$$

Notice that eq.3 just gives the distribution of z_{n1} to z_{nk}. Since one and only one digit of $\boldsymbol{z_n}$ is 1, the following condition should be satisfied:

$$
p(z_{n(K+1)} = 1) = \prod_{i=1}^{K} p(z_{ni} = 0) \tag{6}
$$

The distributions of the regression coefficients and the gating coefficients are all Gaussian, and their precisions all have Gamma distributions:

$$
p(\boldsymbol{V}|\boldsymbol{\beta}) = \prod_{i=1}^{K}\mathcal{N}(\boldsymbol{v_i}|0,\beta_i^{-1}\boldsymbol{I}) \tag{7}
$$

$$
p(\boldsymbol{W}|\boldsymbol{\alpha}) = \prod_{i=1}^{K+1}\mathcal{N}(\boldsymbol{w_i}|\boldsymbol{0},\alpha_i^{-1}\boldsymbol{I}) \tag{8}
$$

$$
p(\boldsymbol{\alpha}) = \prod_{i=1}^{K+1} Gam(\alpha_i|a_\alpha,b_\alpha) \tag{9}
$$

$$
p(\boldsymbol{\beta}) = \prod_{i=1}^{K} Gam(\beta_i|a_\beta,b_\beta) \tag{10}
$$

$$
p(\boldsymbol{\tau}) = \prod_{i=1}^{K} Gam(\tau_i|a_\tau,b_\tau) \tag{11}
$$

Now the target is to find a set of $\boldsymbol{Z},\boldsymbol{V},\boldsymbol{W},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\tau}$ to maximize CDLL defined by eq.1, given a fixed training set $\{\boldsymbol{\phi_n}, y_n\}$. The optimisation can be achieved by using the EM algorithm and variational inference.

2.2.3 Expectation maximization and variational inference

Expectation maximization (EM) algorithm [15] is often used to find maximum likelihood estimates of the parameters in a statistical model. It runs iteratively to optimize the parameters and latent variables in the model, where the latent variables are often unobserved. As a result, in EM algorithm the model is segmented into three different parts: observed data set ($\mathcal{X}$), latent variables ($\mathcal{Y}$) and the parameters to inference (θ). EM algorithm has two steps: E-step and M-step. In E-step, latent variables ($\mathcal{Y}$) are estimated according to observed data ($\mathcal{X}$) and fixed parameter set (θ), i.e. maximize the free energy term [16] w.r.t the latent variables $\mathcal{F}(q,\theta)$:

$$\mathcal{F}(q,\theta) = \int q(\mathcal{Y}) \log \frac{p(\mathcal{Y},\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \tag{12}$$

Where $q(\mathcal{Y})$ represents any possible distribution of the latent variables. This is proven to be equivalent to minimizing the Kullback-Leibler divergence term [16] $KL[q(\mathcal{Y})||p(\mathcal{Y}|\mathcal{X},\theta)]$. In the M-step, the latent variables are fixed at the results from E-step as if they are not hidden, and then the likelihood is maximized w.r.t the parameters. E-step and M-step are iterated until a self-consistent result is obtained.

However, in this study, EM algorithm cannot be directly applied to optimize the model. The main spoiler is the fact that the logistic function defined by eq.4 does not belong to the exponential family, which means in E-step, the integral of marginal likelihood is not analytically tractable. At this point, variational inference is introduced to find a proxy solution [17]. The logistic sigmoid function has a lower bound:

$$\sigma(x) \geq \sigma(\epsilon) \exp\{(x-\epsilon)/2 - \lambda(\epsilon)(x^2 - \epsilon^2)\} \tag{13}$$

where

$$\lambda(\epsilon) = \frac{1}{2\epsilon}\left[\sigma(\epsilon) - \frac{1}{2}\right] \tag{14}$$

ϵ is a new parameter(variational parameter) introduced, which also needs to be optimized in the training process.

In the variation E-step, the logistic function is substituted by its lower bound, which is from the exponential family. M-step is unchanged. Theoretically, variational inference is not ensured to bring up the likelihood in every iteration, especially when the lower bound proxy is not very tight. In this study, it is not a severe problem. This will be proved later in the result.

In this study, the variational inference was configured as follows: the model was first segmented into three parts:observed data set: $\mathcal{X} = \{\boldsymbol{\phi_n}, y_n\}$; latent variables: $\mathcal{Y} = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{W}, \boldsymbol{V}, \boldsymbol{Z}\}$; variational parameter: $\theta = \{\boldsymbol{\epsilon}\}$. All the variables in the graphical model were taken as latent variables and the additional parameter $\boldsymbol{\epsilon}$ was referred as the parameter to inference. EM algorithm was then invoked:

Factored variational E-step: since there are 6 latent variables, they are partitioned into disjoint sets $\mathcal{Y}_s$ as follows:

$$q(\mathcal{Y}) = \prod_{s=1}^{6} q_s(\mathcal{Y}_s) \tag{15}$$

In correspondence, the E-step is also partitioned into iterations: maximizes free energy $\mathcal{F}(q,\theta)$ w.r.t $q_s(\mathcal{Y}_s)$ with the other latent variable $q_{j\neq s}(\mathcal{Y}_{j\neq s})$ and the parameter θ fixed, which can be described as:

$$q_s^{new}(\mathcal{Y}_s) := \underset{q_s(Y_s)}{argmax} \mathcal{F}(q_s(\mathcal{Y}_s) \prod_{j\neq s} q_j(\mathcal{Y}_j), \theta^{old}) \tag{16}$$

The result of the E-step is:

$$q_s(\mathcal{Y}_s) \propto \exp\langle \log p(\mathcal{X}, \mathcal{Y}|\theta^{old})\rangle_{\prod_{j\neq s} q_j(\mathcal{Y}_j)} \tag{17}$$

where notation <> represents the expectation. Notice here it might be a bit difficult to explicitly show the distribution of $q_s(\mathcal{Y}_s)$, but it is not necessary. The only necessary result is its expectation. Besides, the above formulation only depends on the sufficient statistics of $q_s(\mathcal{Y}_s)$, which significantly facilitates the computational process.

M-Step the M-step is unchanged, Free energy $\mathcal{F}(q,\theta)$ was optimized w.r.t the parameter set $\theta = \{\boldsymbol{\epsilon}\}$, with fixed latent variable distributions (results from the E-Step). This can be described by:

$$\theta^{new} = \underset{\theta}{argmax} \mathcal{F}(q^{new}(\mathcal{Y}), \theta) \tag{18}$$

which is equivalent [16] to

$$\theta^{new} = \underset{\theta}{argmax} \langle \log p(\mathcal{X}, \mathcal{Y}|\theta)\rangle_{q^{new}(\mathcal{Y})} \tag{19}$$

Similarly, it only depends on the sufficient statistics of θ.

All variables in the graphical model were optimized iteratively until a self-consistent result is obtained, i.e., either the CDLL or the parameters themselves converge. Due to the limitation of space, derivations of all the formulations are attached in Appendix A.

2.2.4 Making the prediciton

In this study, a soft mixture of the component models was used to make the predictions. For an unknown observation, each component model produced a prediction, and the outputs of the gating functions indicate the weights for each model. Predictions from $K+1$ component models were then weighted averaged out to give a single prediction. For all the datasets tested in this study, it has been observed that when 2 component models were used, the best prediction accuracy can be achieved. As a result, K was set to 1 for all the HMLR calibrations.

2.3 Locally weighted regression

Locally weighted regression (LWR) is not the primary focus of this study, but just a benchmark method. In LWR, the whole transformed training set $\{\boldsymbol{\phi_n}, y_n\}$ is recorded as a searching library for the model. When making predictions on new observations, raw spectra are first preprocessed and then PLS transformed. Several nearest neighbors to the new observation in the training set are picked up to build a local linear regression [9] [18]. Then the local linear regression is used to predict the unknown observation. The calibration and prediction process are repeated for every individual sample in the test set. In this study, cross-validation was used to decide the number of neighbors in the model, with a Euclidean distance measurement and distance-based weighting mechanism.

2.4 Comparing prediction methods

Suppose two different prediction methods have been calibrated to predict target values $\boldsymbol{y}$ from spectral data $\boldsymbol{X}$. Usually, biases and root mean squared errors of predictions (RMSEP) of the prediction errors are compared to evaluate the performance of the two methods. However, since the errors in the lab measurements for the test samples contribute to both sets of predictions errors, biases and standard deviations are correlated. The following calculation finds the actual difference on the biases and the true ratio of the RMSEP [19] [20]:

Let e_i donate the prediction error for the i^{th} sample in the test set, the bias, standard deviation and RMSEP can be written by:

$$m = \frac{1}{n}\sum_{i=1}^{n} e_i \tag{20}$$

$$s = \sqrt{\frac{\sum_{i=1}^{n}(e_i - m)^2}{n-1}} \tag{21}$$

$$RMSEP = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n}} \tag{22}$$

the difference between the means $m_1 - m_2$ of the two methods has a standard deviation of:

$$s_d = \sqrt{\frac{\sum_{i=1}^{n}(d_i - \bar{d})^2}{n \times (n-1)}} \tag{23}$$

where $d_i = e_{i1} - e_{i2}$ is the difference on errors of the two methods for the i^{th} sample. The 95% T confidence interval in biases is :

$$((m_1 - m_2) - t_{n-1.0.025} \times s_d, (m_1 - m_2) + t_{n-1.0.025} \times s_d) \tag{24}$$

where $t_{n-1.0.025}$ is the upper 2.5% point of a T distribution on $n-1$ degrees of freedom.

Normally the critical comparison is between the RMSEP (or standard deviations). Similarly, the lower and upper limits of a 95% confidence interval for the ratio of the RMSEP is given by:

$$(\frac{RMSEP_1}{RMSEP_2} \times \frac{1}{L}, \frac{RMSEP_1}{RMSEP_2} \times L) \tag{25}$$

where L is defined by:

$$L = \sqrt{K + \sqrt{(K^2 - 1)}} \tag{26}$$

$$K = 1 + \frac{2(1-r^2)t_{n-2.0.025}^2}{n-2} \tag{27}$$

and r is the correlation coefficient between the two sets of prediction errors, $t_{n-2.0.025}$ is the upper 2.5% point of a T distribution on $n-2$ degrees of freedom.

3 Dataset

Dataset1 Dataset1 is on ash contents of some wheat flour samples. Products come from different origins and consist of multiple varieties. The training set contains 5007 pairs of NIR spectra and corresponding lab ash measurements. The test set has a sample size of 1980. Spectral range is from $850nm$ to $1650nm$, with an interval of $5nm$. Raw spectra were processed with second-order detrend followed by SNV. 7-factor PLS was applied to the training set to reduce the number of attributes in X.

Dataset2 Dataset2 contains NIR spectra and corresponding moisture contents of wheat flour. Samples also have different origins and varieties. The spectral range is from $850nm$ to $1650nm$. Spectral resolution is $5nm$. The training set contains 700 samples; the test set has 388 instances. Raw spectra were preprocessed by Savitzky-Golay filter (2 side points, 2^{nd} order derivative on 2^{nd} order polynomial fitting) followed by SNV. Preprocessed spectra in the training set were then reduced to 2 PLS factors for calibration.

Dataset3 Dataset3 is a publicly dataset [21] [22].The training set contains 415 NIR spectra and corresponding protein concentration of single wheat kernels from different locations in Denmark, while the test set contains 108 samples.Raw spectra were preprocessed by second-order detrend followed by SNV. 6 PLS factors were used to reduce the dimensionality before calibration.

Dataset 1 and dataset 2 are original to this paper. The datasets were collected and organized by Bühler AG. Origins of the samples span the globe. Wheat flour was completely homogenised before measured. Multiple measurements were repeated on the same sample under different environmental conditions. The training and the test datasets were collected separately on different samples with the same spectroscopic and lab measurement system. There is no apparent outlier on training or test sets (i.e. mislabelling or very noisy spectrum). The training sets and the test sets have very similar structures. The only difference is that they were collected from separate measurements, on different samples. The training and the test datasets are genuinely independent, for all the calibration methods the models are solely supervised by the training set. Any overfitting effect on the training set be be spotted when evaluated on the test set, except the factors introduced by the spectroscopic and lab system.

For all the datasets, preprocessing methods and numbers of PLS factors were picked up by cross-validation on the training set, with a target of minimizing the root mean squared error of cross-validation (RMSECV) on PLSR. This guarantees optimized results from PLSR.

The primary study and analysis are focused on dataset1 because it has the most significant size. Results from dataset2 and dataset3 will be briefly discussed and illustrated, to provide insights into how HMLR performs on different commodities and constituents.

4 Software

All the simulations were done in Python 3.6.1. Matrix arithmetic was realized with numpy package [23] and partial least squares decomposition was achieved with scikit-learn package [24]. The HMLR part was coded from scratch by the authors.

5 Results and analysis

For all datasets, the raw NIR spectra were preprocessed and then dimensionally reduced by PLS as described above. PLS scores were then passed to MLR, HMLR, and LWR.

In the HMLR model, two submodels were trained on the training set. The initial values of $\{\boldsymbol{Z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\epsilon}\}$ were randomly assigned, then the parameters were updated in order of $\boldsymbol{V} \rightarrow \boldsymbol{W} \rightarrow \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\tau}\} \rightarrow \boldsymbol{Z} \rightarrow \boldsymbol{\epsilon}$. The last step (update on $\boldsymbol{\epsilon}$) corresponds to the M-step, and the others are the factorized variational E-step in the variational inference algorithm. Notice that $\boldsymbol{\alpha}, \boldsymbol{\beta}$ should take small initial values (0.1 for example) and a, b in eq.9, eq.10 and eq.11 should be very small numbers (10^{-4} for example) to make regularization on the regression coefficients weak. In the HMLR model, the raw data were already regularized by PLS shrinkage. It is not necessary to penalize too much on the coefficients (In fact, there is no penalty on the regression coefficients in PLSR, but they are still robust).

The parameter update process was iterated 100 times for each calibration. It is recommended to check whether some optimisation indicators (i.e., CDLL, RMSEC, etc.) are consistent after the optimisation. Since the optimisation started at some random values, it is possible the final result is just a local optimum, so the whole optimisation process was repeated at several random start values (in this study 80 random initialisation were used) to ensure a global optimum for each calibration.

5.1 Ash calibration

5.1.1 Monitoring the training process

The main uncertainty of the HMLR method is that variational inference is only an approximation to the log-likelihood. In theory after a whole EM iteration, the log-likelihood may not increase. It is recommended to monitor the optimisation process, especially the changes in the CDLL after each update, to make sure the optimisation is in the correct direction. In addition to the CDLL, fitting ability of the model is also of interest. It is essential that after each update fitting power of the model is also increased (otherwise the improvement on CDLL only comes from the regularization terms), to reject some over-regularized models. Referring to eq.1, only $p(y_n|\boldsymbol{W}, \boldsymbol{z_n}, \boldsymbol{\phi_n}, \boldsymbol{\tau})$ and $p(\boldsymbol{z_n}|\boldsymbol{V}, \boldsymbol{\phi_n})$ are relevant to the fitting, the logarithm of the sums of products of these two terms can be taken as the "fitting score" of the model. Fig.3 shows a typical case of the changes in the CDLL and the fitting scores after each update.

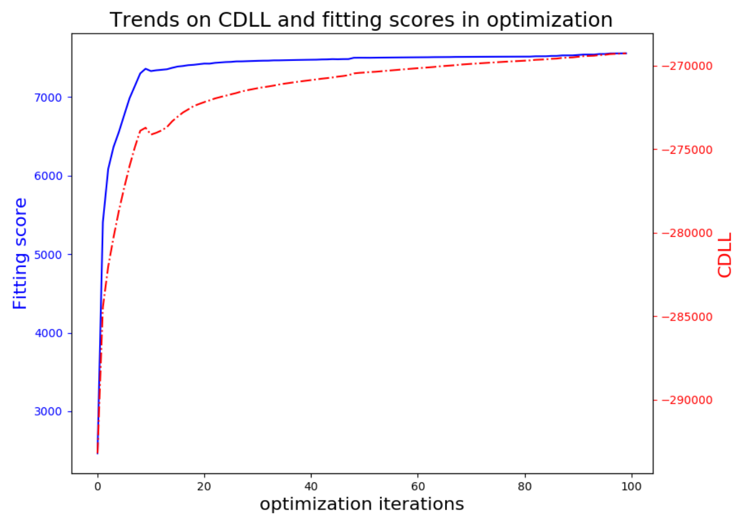


Figure 3: Trends on CDLL and fitting scores after each whole EM iteration. Blue solid line: trend on fitting scores. Red dotted line: CDLL

It can be observed that the overall trends on both fitting scores and CDLL were increased during the optimisation. Fitting score converged after around 20 iterations, while CDLL took more than 80 iterations to reach self-consistency.

5.1.2 Interpretation of the model

As explained, every single sample in the training set was assigned a label $\boldsymbol{z_n}$, which indicated the clustering of the training samples. This labeling variable was also optimized automatically by the variation inference algorithm. It is useful to see have the automatic segmentation is done by the algorithm. Fig.4 shows the distribution of ash content in the two component models trained by variation inference.

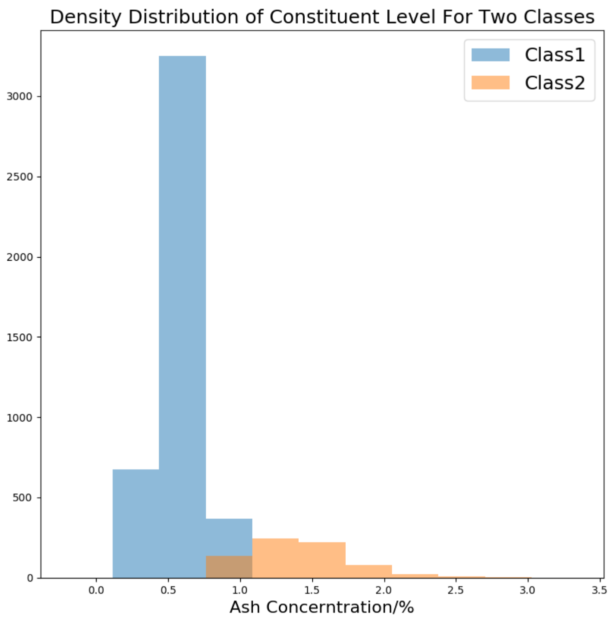


Figure 4: Composition of the training samples for the two component models.

It can be observed that class 1 (blue face colour) comprises the training samples low in ash content, whereas class 2 (orange face color) consists of the ones high in ash content. There is a small overlap, but the two clusters of the training samples mainly represent two clusters of the reference values.

In the end, the HMLR model is equivalent to a two-tier PLSR model, according to the ash content. However, the attraction of this approach is this segmentation process was done automatically by the EM and variational inference optimisation, which means it also meet the optimum requirements of the data likelihood. Besides, a gating function was obtained simultaneously, optimizing the segmentation alongside with the component regressions. Refer to eq.3, the outputs of the logistic gating function represent the probabilities of the two models, which makes it clear how to average out the prediction results from the two models when making the predictions.

It is also useful to plot the regression coefficients. Fig.5 shows the regression coefficients from the PLSR (red dash-dot line) and the two component models of the HMLR (green line and the blue dashed line). The trends of the three coefficients are very similar, except the signal strengths at some peaks vary slightly. The coefficients from PLSR are closer to that of the component model 1 in HMLR. The result is expected because most of the training samples were assigned to the component model 1.

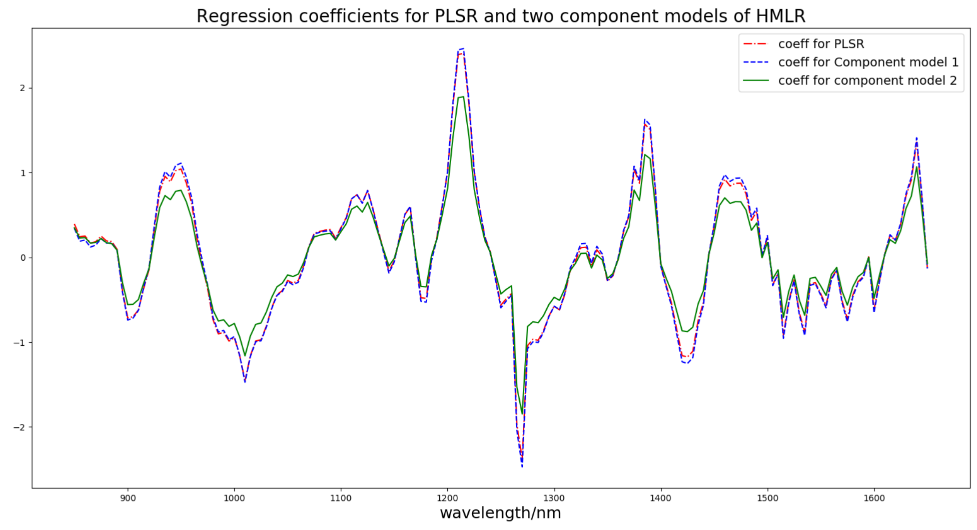


Figure 5: Regression coefficients for the PLSR, the two component models of HMLR.

Fig.6 shows the coefficients for the gating function. The trend is very similar to those in fig.5. In fact, the segmentation was based on the ash contents so it is not surprising that the gating function expresses the similar message to the ash concentration.

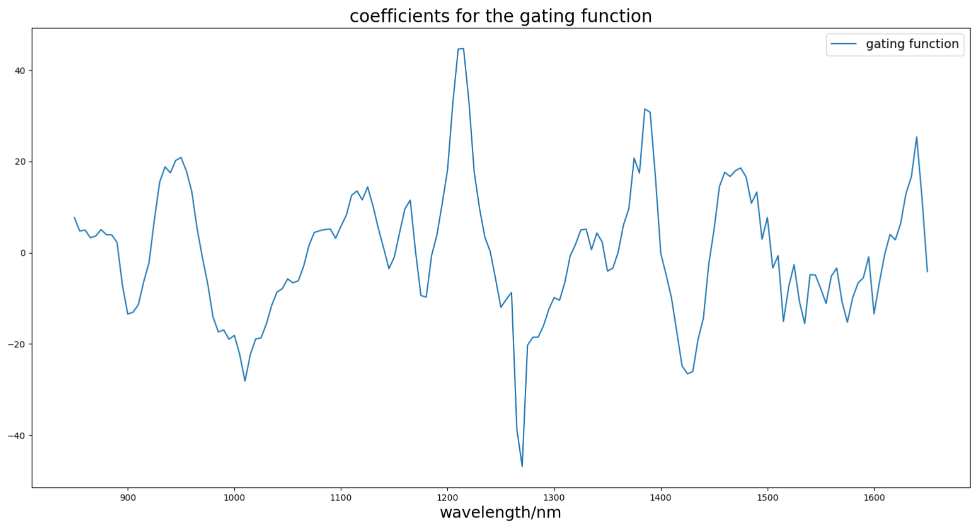


Figure 6: Coefficients for the gating function

It may be confusing why the segmentation is based on the concentration of the target constituent. First of all, from a mathematical standpoint, the initial values of the free parameters in the model were randomly assigned. The target of variation inference is to maximise data log-likelihood. There is no chance that any prior preference on segmentation can be applied to the training process. Considering that the sample for this study, wheat flour, are thoroughly homogenised before measured, the signal from the same sample is strong and highly reproducible. So the segmentation is based on the intrinsic correlation between input spectrum $\boldsymbol{X}$ and target property $\boldsymbol{y}$. The correlation is slightly different when $\boldsymbol{y}$ varies. The effect is found, by the EM and variational inference, to be the most influential criteria for segmentation. Later in another dataset, it can be observed how segmentation is changed on whole grain samples.

5.1.3 Interpretation of τ

Referring to the graphical model described in Fig.2, $\boldsymbol{\tau}$ is the precision on $\boldsymbol{y}$, which means the precision of the reference measurement. This value has physical meaning behind it, since it is determined by the accuracy of lab measurement. When training the model, $\boldsymbol{\tau}$ was first assigned randomly, then automatically optimized until self-consistent. It is essential to check whether the final value of $\boldsymbol{\tau}$ is close to the reference value of lab measurement. If $\boldsymbol{\tau}$ is too high, then the model is very likely over-fitted; if it is too small, then the model is under-fitted.

Since there were two component models, separate τ values were tuned for each of them. Results showed that $\tau_1 = 313$, $\tau_2 = 172$. The equivalent variances on reference $\boldsymbol{y}$ are 0.003 and 0.006. These values coincide with the expectations based on lab measurement. Besides, τ_2 is smaller than τ_1, which also makes sense because the error of lab measurement increases with the ash concentration.

5.1.4 Comparison of the result

In this test, the root-mean-square errors of prediction (RMSEP) on the test sets for the three models are PLSR-0.083%; LWR-0.084%; HMLR-0.070%. LWR was trained on 13 neighbours (chosen by cross-validation on the training set). The performance of LWR and PLSR are very close. The error obtained by HMLR is 16% smaller than the PLSR. To prove whether the difference is significant, a test for paired predictions was utilized to calculate the true ratio of RMSEP for the two models. Detailed calculation was introduced in section 2.4. The test indicated that the 95% confidence intervals for the true ratio of RMSEP on ash is $\frac{PLSR}{HMLR} \propto (1.167, 1.273)$. Since this interval does not include 1, there is strong evidence that error of PLSR is significantly larger than the error of HMLR.

Fig.7 shows the prediction results from PLSR and HMLR on the test set on the same scale. It is a bit messy on the left side, however, for the samples with an ash content above 1% is is quite apparent that the predictions of HMLR are closer to the reference value.

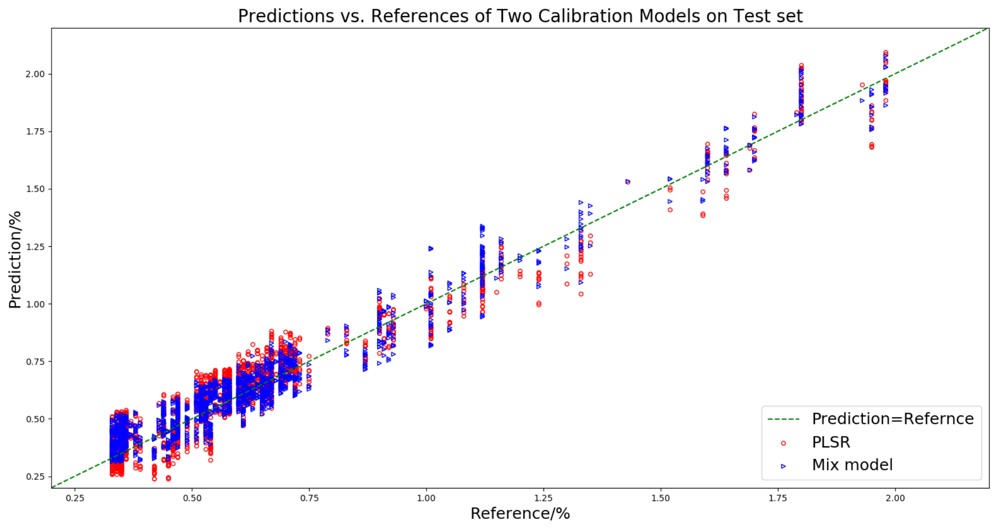
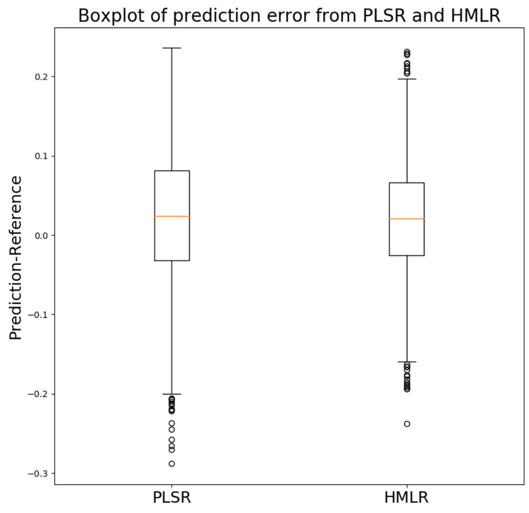
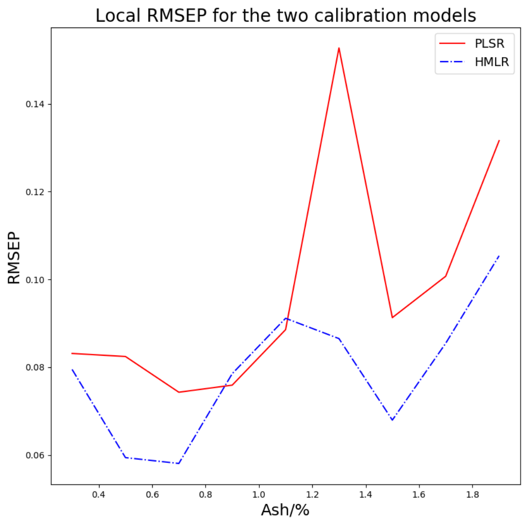


Figure 7: Prediction vs. reference on dataset1.Results from two calibration methods.

Some results from the scatter plots in fig.7 are summarized in fig.8. Fig.8(a) is a boxplot of the prediction errors from the PLSR and the HMLR. Median, the first and the third quantile and the inner fences are indicated. It is clear the HMLR outperformed the PLSR in the sense of the global accuracy. Fig.8(b) illustrates the performance from a standpoint of regional accuracy. The validation set was split into nine local regions with a bin width of 0.2%. The red dashed line is the accuracy curve for the HMLR, and the solid blue line is for the PLSR. In most parts of the population, RMSEP from the HMLR is smaller than that of the PLSR, except for the range 0.8% $\sim$ 1.2%, where the two methods have very similar performances. The field is also where the training set got segmented in the HMLR model. So for the samples with ambiguous labels (i.e., the output of the gating function is close to 0.5 for both the component models), there is no improvement using the HMLR methods. When there is a dominating component model for the prediction, using HMLR can bring significant improvement.



(a) Boxplot of the global errors.

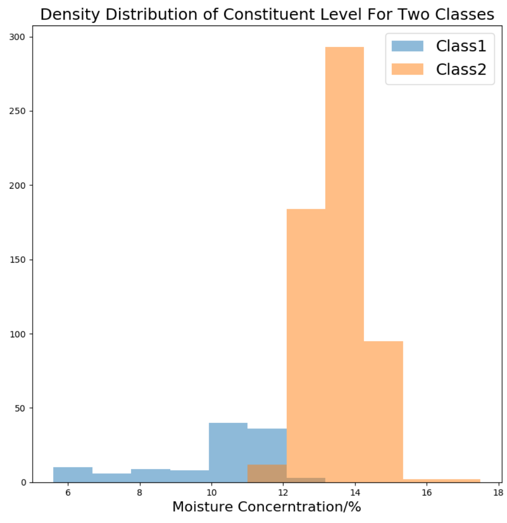


(b) Local RMSEP for the two models.

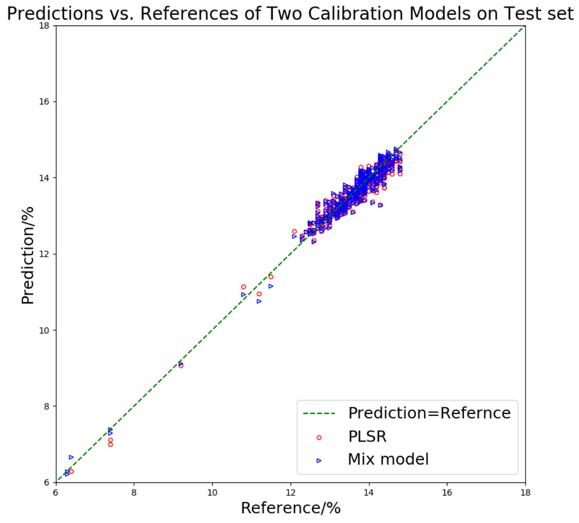
Figure 8: Comparison of the prediction accuracies of the two ash calibration models

5.2 Moisture and protein calibrations

Results on dataset2 are plotted in fig.9. Dataset 2 is on moisture of wheat flour. From the results it can be observed, similar to the ash calibration, the data was automatically segmented into two subsets according to the concentration on moisture content. The boundary of the two classes is at around 12%. Fig.9 (b) shows the predictions on the test set from the two models. Unfortunately, the test samples below 12% are not numerous enough to show the difference between the two models in this region. The RMSEPs of the three models on the test sets are PLSR:0.228%; LWR:0.222% (8 neighbours); HMLR: 0.212%. There is an improvement in the prediction accuracy by using HMLR, but it is not significant. In general, calibration on moisture is not challenging. PLSR can produce an entirely satisfactory result (refer to fig.9, predictions from PLSR on extreme samples were not heavily biased or distorted).



(a) Dataset segmentation on the training set



(b) Predictions vs. reference from the two models.

Figure 9: Segmentation and predictions on dataset2

Dateset2 is not a very challenging dataset. Moisture has a strong correlation with NIR spectrum, especially for powder samples where signal is robust and reproducible. Results of the test indicate that even for simple datasets like moisture in flour, HMLR does not overfit on the training set. PLS shrinkage + CDLL is proven to be an appropriate, well-regularised training process. Hence HMLR can be used as a routine calibration strategy.

Results from dataset3 are plotted in fig.10. It can be found there is more overlap in the two subsets. The consequence is partly because the distribution of the protein in wheat kernels is more symmetric than ash or moisture, i.e., the distribution profile has no outstanding tails. The other reason is that the gating in this calibration is not entirely based on the protein concentration. Fig.10 shows the regression coefficients from the PLSR and the gating coefficients of the HMLR. It can be seen the two curves partially overlap, mainly on the right-hand side of the spectrum. On the left-hand side, the gating and the regression functions show different messages. Whole grain samples have more complicated features than powder products, and there might be other segmentation criteria of the samples. For example, the presentation of the grain (crease side or the opposite side), color, shape and surface conditions.

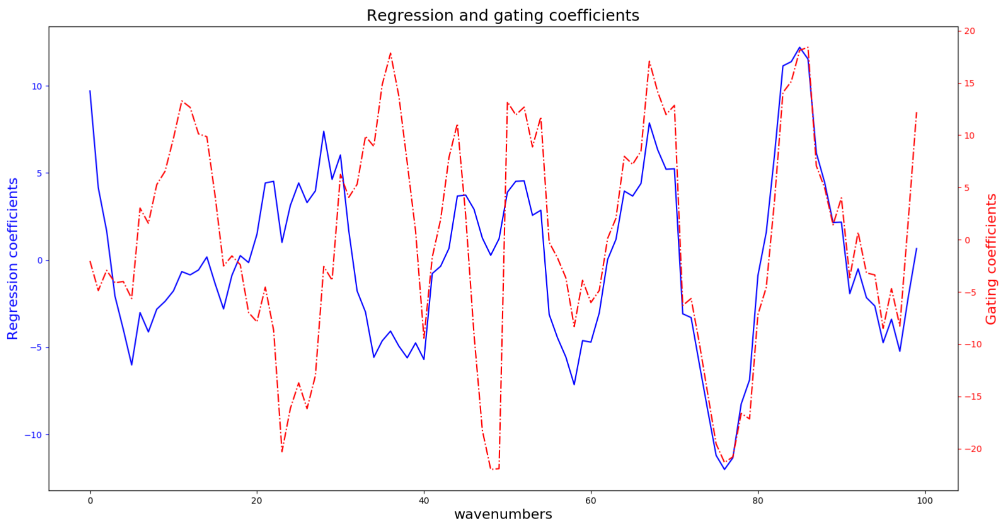
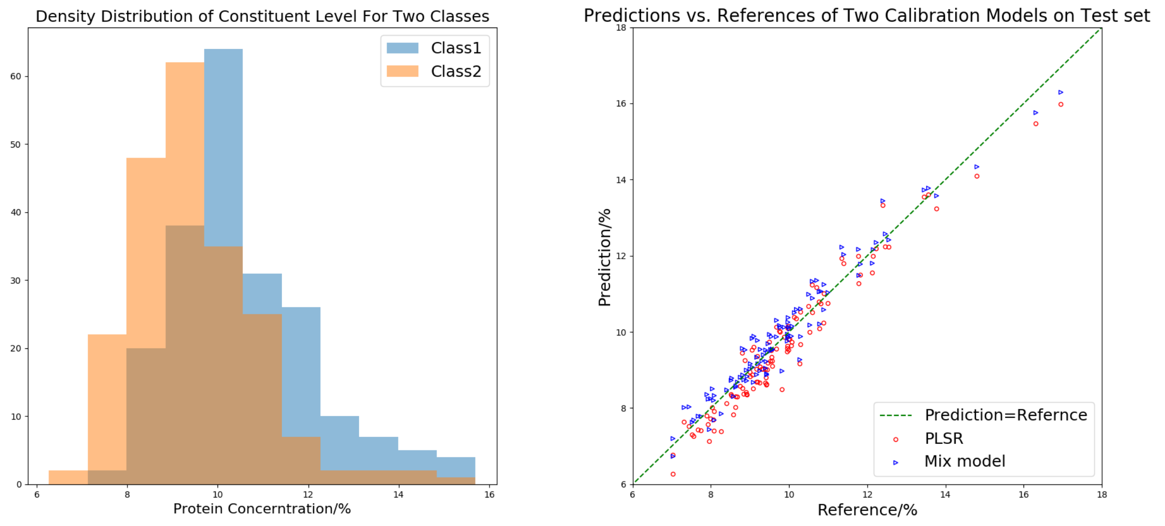


Figure 10: PLSR coefficients (6 factors) and gating function coefficients for protein calibration.

The RMSEPs of the three models on the test sets are PLSR:0.468%; LWR:0.753% (4 neighbors); HMLR:%0.386. From fig.11 (b) it can be seen that the improvement is across the range.



(a) Dataset segmentation on the training set **(b)** Predictions vs. reference from the two models.

Figure 11: Segmentation and predictions on dataset3

Residual vs reference is plotted in fig.12. It can be observed that the bias of the HMLR model is smaller than the PLSR model. The PLSR model has a global bias of -0.240, and that of the HMLR model is 0.008. The method introduced in section 2.4 is used to find out the T confidence interval for the biases of the two models. Result indicates 95% confidence interval for the actual difference in biases is $(-0.0267, -0.0229)$. Since this interval does not include 1, there is strong evidence to prove that the PLSR model has a significantly larger negative bias than the HMLR model.

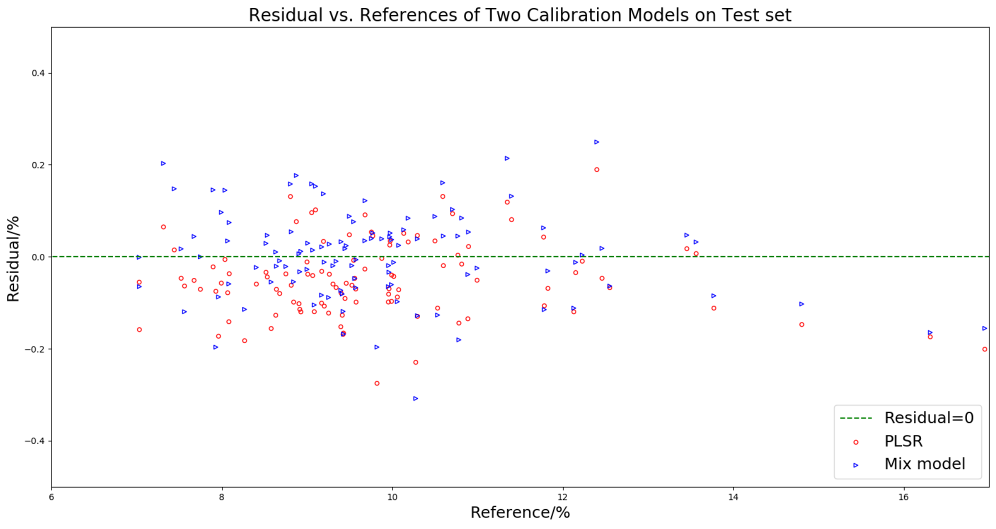


Figure 12: Residual vs. reference on dataset3.

6 Conclusion

In this research, a new calibration method for NIR spectroscopy, hierarchical mixture of linear regressions, was introduced. The technique automatically searches for a few component PLSR models on the training set, along with a set of gating functions to determine the presence of the component models when making the predictions. Compared to the traditional scheme of ensemble modelling, HMLR uses the EM algorithm with variational inference, which makes it entirely automated. No manual data segmentation is needed for the training. Besides, it keeps the characteristics of a parametric model. Similar to PLSR, it is interpretable and very compact. In practical, it is possible to understand the model and to monitor the optimisation process. It can be implemented efficiently on in-line measurement systems.

The introduced method was tested on predicting three different constituents — ash, moisture and protein, on wheat flour or whole wheat grains. HMLR showed superiority over PLSR and LWR on prediction accuracy. In the other experiments not reported here, HMLR outperform PLSR in most cases, while sometimes it has similar performance with PLSR. It is recommended to use this calibration scheme as a routine method, especially when an interpretable, compact, unbiased and accurate model is desired.

Acknowledgements

The authors wish to express their gratitude to Bühler AG for supplying high-quality large datasets for the study. They also want to thank Juste Hahne and Martin Heine for demon-

stration and analysis of the datasets.

References

[1] H. Wold, “Partial least squares,” *Encyclopedia of statistical sciences*, 1985.

[2] I. T. Jolliffe, “A note on the use of principal components in regression,” *Applied Statistics*, pp. 300–303, 1982.

[3] G. Grégoire, “Multiple linear regression,” *European Astronomical Society Publications Series*, vol. 66, pp. 45–72, 2014.

[4] J. A. Suykens, T. Van Gestel, and J. De Brabanter, *Least squares support vector machines.* World Scientific, 2002.

[5] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*, vol. 1. MIT press Cambridge, 2006.

[6] Q. Chen, J. Zhao, C. Fang, and D. Wang, “Feasibility study on identification of green, black and oolong teas using near-infrared reflectance spectroscopy based on support vector machine (svm),” *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 66, no. 3, pp. 568–574, 2007.

[7] C. Cui and T. Fearn, “Comparison of partial least squares regression, least squares support vector machines, and gaussian process regression for a near infrared calibration,” *Journal of Near Infrared Spectroscopy*, vol. 25, no. 1, pp. 5–14, 2017.

[8] J. Pierna, V. Baeten, A. M. Renier, R. Cogdill, and P. Dardenne, “Combination of support vector machines (svm) and near-infrared (nir) imaging spectroscopy for the detection of meat and bone meal (mbm) in compound feeds,” *Journal of Chemometrics*, vol. 18, no. 7-8, pp. 341–349, 2004.

[9] T. Naes, T. Isaksson, and B. Kowalski, “Locally weighted regression and scatter correction for near-infrared reflectance data,” *Analytical Chemistry*, vol. 62, no. 7, pp. 664–673, 1990.

[10] P. Estevez and R. Nakano, “Hierarchical mixture of experts and max-min propagation neural networks,” in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 1, pp. 651–656, IEEE, 1995.

[11] S. Wold, H. Martens, and H. Wold, “The multivariate calibration problem in chemistry solved by the pls method,” *Matrix pencils*, pp. 286–293, 1983.

[12] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[13] J. A. Wegelin *et al.*, “A survey of partial least squares (pls) methods, with emphasis on the two-block case,” *University of Washington, Department of Statistics, Tech. Rep*, 2000.

[14] F. V. Jensen, *An introduction to Bayesian networks*, vol. 210. UCL press London, 1996.

[15] C. M. Bishop, *Pattern recognition and machine learning.* springer, 2006.

[16] M. Sahani, "Lecture notes distributed in probabilistic and unsupervised learning. available from http://www.gatsby.ucl.ac.uk/teaching/courses," 2016.

[17] T. S. Jaakkola and M. I. Jordan, "Bayesian parameter estimation via variational methods," *Statistics and Computing*, vol. 10, no. 1, pp. 25–37, 2000.

[18] H. Martens and T. Naes, *Multivariate calibration.* John Wiley & Sons, 1992.

[19] T. Næs, T. Isaksson, T. Fearn, and T. Davies, *A user friendly guide to multivariate calibration and classification.* 2002.

[20] T. Fearn, "Comparing standard deviations," *NIR news*, vol. 7, no. 5, pp. 5–6, 1996.

[21] D. K. Pedersen, H. Martens, J. P. Nielsen, and S. B. Engelsen, "Near-infrared absorption and scattering separated by extended inverted signal correction (eisc): analysis of near-infrared transmittance spectra of single wheat seeds," *Applied spectroscopy*, vol. 56, no. 9, pp. 1206–1214, 2002.

[22] J. P. Nielsen, D. K. Pedersen, and L. Munck, "Development of nondestructive screening methods for single kernel characterization of wheat," *Cereal chemistry*, vol. 80, no. 3, pp. 274–280, 2003.

[23] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The numpy array: a structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

Appendix A Variational distribution for each parameter

Using eq.17 and eq.19, we can write out the expectation of all latent variables and parameter after each update.
Z:

$$\begin{aligned}\log q_{\boldsymbol{Z}}^*(\boldsymbol{Z}) =&\langle \log\{\prod_n p(y_n|\boldsymbol{\phi_n},\boldsymbol{z_n},\boldsymbol{W},\boldsymbol{\tau})p(\boldsymbol{z_n}|\boldsymbol{V},\boldsymbol{\phi_n})\}\rangle + const\\ =&\langle\sum_n \log p(y_n|\boldsymbol{\phi_n},\boldsymbol{W},\boldsymbol{z_n},\boldsymbol{\tau}) + \sum_n \log p(\boldsymbol{z_n}|\boldsymbol{V},\boldsymbol{\phi_n})\rangle + const\\ =&\langle\sum_n(\sum_{i=1}^{K+1} z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\boldsymbol{\phi_n})^2) + \sum_{i=1}^{K} t(i,\boldsymbol{z_n})z_{ni}\boldsymbol{v_i^T}\boldsymbol{\phi_n})\rangle + const\\ =&\langle\sum_n(\sum_{i=1}^{K} z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\boldsymbol{\phi_n})^2)\\ &+ z_{n(K+1)}(\frac{1}{2}\log\tau_{K+1} - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_{K+1}(y_n - \boldsymbol{w_{K+1}}\boldsymbol{\phi_n})^2) + \sum_{i=1}^{K} t(i,\boldsymbol{z_n})z_{ni}\boldsymbol{v_i^T}\boldsymbol{\phi_n})\rangle + const\end{aligned} \tag{28}$$

Where $t(i,\boldsymbol{z_n})$ is defined by eq.(5), and the following relationship is satisfied:

$$t(i,\boldsymbol{z_n}) = \begin{cases}1 & z_{ni}=1\\ 0 & z_{ni}=0\end{cases} \tag{29}$$

As a result we can replace $t(i,\boldsymbol{z_n})z_{ni}$ with z_{ni} and eliminate $t(i,\boldsymbol{z_n})$. eq.(20) can be rewritten as:

$$\begin{aligned}\log q_Z^*(Z) =&\langle\sum_n\{\sum_{i=1}^{K} z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\boldsymbol{\phi_n})^2 + \sum_{i=1}^{K} z_{ni}\boldsymbol{v_i^T}\boldsymbol{\phi_n})\\ &+ z_{n(K+1)}(\frac{1}{2}\log\tau_{K+1} - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_{K+1}(y_n - \boldsymbol{w_{K+1}}\boldsymbol{\phi_n})^2)\}\rangle + const\\ =&\langle\sum_n\{\sum_{i=1}^{K} z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\boldsymbol{\phi_n})^2 + \boldsymbol{v_i^T}\boldsymbol{\phi_n})\\ &+ z_{n(K+1)}(\frac{1}{2}\log\tau_{K+1} - \frac{1}{2}\tau_{K+1}(y_n - \boldsymbol{w_{K+1}}\boldsymbol{\phi_n})^2)\}\rangle + const\end{aligned} \tag{30}$$

$$z_{n(K+1)} = 1 - \sum_{i=1}^{K} z_{ni} \tag{31}$$

Eliminate all terms irrelevant with z_{ni}, further we have:

$$\begin{aligned}\log q_{z_{i\neq K+1}}^*(z_{i\neq K+1}) =&\langle\sum_n\sum_{i=1}^{K} z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\log\tau_{K+1} - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\boldsymbol{\phi_n})^2\\ &+ \frac{1}{2}\tau_{K+1}(y_n - \boldsymbol{w_{K+1}^T}\boldsymbol{\phi_n})^2 + t(i,\boldsymbol{z_n})\boldsymbol{v_i^T}\boldsymbol{\phi_n})\rangle + const\end{aligned} \tag{32}$$

It is quite difficult to write out the distribution of $\boldsymbol{z_n}$, but it is relatively easier to find out a set of binary varaibles $\boldsymbol{z_n}$ which can maximize $\log q_Z(Z)$. Set $z_{ni} = 1$ if and only if i maximize the target function:

$$\begin{aligned} g(i) =& \underset{i}{argmax}\{\frac{1}{2}\log\tau_i - \frac{1}{2}\log\tau_{K+1} - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i}^T\boldsymbol{\phi_n})^2 \\ &+ \frac{1}{2}\tau_{K+1}(y_n - \boldsymbol{w_{K+1}}^T\boldsymbol{\phi_n})^2 + t(i,\boldsymbol{z_n})\boldsymbol{v_i}^T\boldsymbol{\phi_n}\} \end{aligned} \tag{33}$$

If this maximum value is negative, then set $z_{n(K+1)} = 1$ and all other terms are zeros.
V:

$$\begin{aligned} \log q_V^*(V) =& \langle\log\{\prod_n p(\boldsymbol{z_n}|V,\boldsymbol{\phi_n})p(V|\beta)\}\rangle + const \\ =& \langle\sum_n\sum_{i=1}^K\{t(i,\boldsymbol{z_n})(z_{ni}\boldsymbol{v_i}^T\boldsymbol{\phi_n} + \log\sigma(-\boldsymbol{v_i}^T\boldsymbol{\phi_n}))\} + \sum_{i=1}^K\log p(\boldsymbol{v_i}|\beta)\rangle + const \end{aligned} \tag{34}$$

logistic sigmoid function has a lower bound:

$$\sigma(x) \geq \sigma(\epsilon)\exp\{(x-\epsilon)/2 - \lambda(\epsilon)(x^2-\epsilon^2)\} \tag{35}$$

where

$$\lambda(\epsilon) = \frac{1}{2\epsilon}\left[\sigma(\epsilon) - \frac{1}{2}\right] \tag{36}$$

Substitute $\sigma(-\boldsymbol{v_i}^T\boldsymbol{\phi_n})$ with its lower bound in eq.(13), we have:

$$\log q_V^*(V) \geq \langle\sum_n\sum_{i=1}^K t(i,\boldsymbol{z_n})\{(z_{ni} - \frac{1}{2})_{\boldsymbol{i}}\boldsymbol{v_i}^T\boldsymbol{\phi_n} - \boldsymbol{v_i}^T\lambda(\epsilon_{ni})\boldsymbol{\phi_n}\boldsymbol{\phi_n}^T\boldsymbol{v_i}\} - \frac{1}{2}\sum_{i=1}^K\boldsymbol{v_i}^T\beta\boldsymbol{v_i}\rangle + const \tag{37}$$

and then we can write down the lower bound of distribution for each $q_{\boldsymbol{v_i}}(\boldsymbol{v_i})$ as follows:

$$\begin{aligned} \log q_{\boldsymbol{v_i}}(\boldsymbol{v_i}) \geq & \langle\sum_n t(i,\boldsymbol{z_n})(z_{ni} - \frac{1}{2})\boldsymbol{v_i}^T\boldsymbol{\phi_n} - \\ & \boldsymbol{v_i}^T(\sum_n t(i,\boldsymbol{z_n})\lambda(\epsilon_{ni})(\boldsymbol{\phi_n}\boldsymbol{\phi_n}^T))\boldsymbol{v_i} - \boldsymbol{v_i}^T(\frac{\beta}{2})\boldsymbol{v_i}\rangle + const \end{aligned} \tag{38}$$

By completing the square, we can write down the lower bound of the posterior distribution of each $q_{\boldsymbol{v_i}}(\boldsymbol{v_i})$, which is a Gaussian distribution:

$$q_{\boldsymbol{v_i}}(\boldsymbol{v_i}) \sim \mathcal{N}(\boldsymbol{v_i}|\boldsymbol{m_{v_i}},\boldsymbol{S_{v_i}}) \tag{39}$$

where $\boldsymbol{m_{v_i}}, \boldsymbol{S_{v_i}}$ are given by:

$$\boldsymbol{m_{v_i}} = \boldsymbol{S_{v_i}}(\sum_n t(i,\boldsymbol{z_n})(z_{ni} - \frac{1}{2})\boldsymbol{\phi_n}) \tag{40}$$

$$\boldsymbol{S_{v_i}^{-1}} = \beta I + 2\sum_n t(i, \boldsymbol{z_n})\lambda(\epsilon_{ni})(\boldsymbol{\phi_n}\boldsymbol{\phi_n^T}) \tag{41}$$

$\boldsymbol{W}$:

$$\begin{aligned}\log q_{\boldsymbol{w_i}}(\boldsymbol{w_i}) &= \langle \log \prod_n \{p(y_n|\boldsymbol{w_i}, z_{ni}, \boldsymbol{\phi_n}, \tau_i)\} p(\boldsymbol{w_i}|\alpha)\rangle + const \\ &= \langle \sum_n z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\log 2\pi - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T}\phi)^2) - \frac{1}{2}\boldsymbol{w_i^T}\alpha_i\boldsymbol{w_i}\rangle + const \\ &= \langle \sum_n z_{ni}\tau_i y_n \boldsymbol{w_i^T}\boldsymbol{\phi_n} - \frac{1}{2}\boldsymbol{w_i^T}(\tau_i\sum_n z_{ni}\boldsymbol{\phi_n}\boldsymbol{\phi_n^T})\boldsymbol{w_i}\rangle + const\end{aligned} \tag{42}$$

Similarly, $\boldsymbol{w_i}$ follows a normal distribution:

$$q^*_{\boldsymbol{w_i}}(\boldsymbol{w_i}) \sim \mathcal{N}(\boldsymbol{w_i}|\boldsymbol{m_{w_i}}, \boldsymbol{S_{w_i}}) \tag{43}$$

where $\boldsymbol{m_{w_i}}, \boldsymbol{S_{w_i}}$ are given by:

$$\boldsymbol{m_{w_i}} = \boldsymbol{S_{w_i}}\tau_i\sum_n z_{ni}y_n\boldsymbol{\phi_n} \tag{44}$$

$$\boldsymbol{S_{w_i}^{-1}} = \alpha_i I + \tau_i\sum_n z_{ni}\boldsymbol{\phi_n}\boldsymbol{\phi_n^T} \tag{45}$$

$\boldsymbol{\alpha}$:

$$\begin{aligned}\log q^*_{\alpha_i}(\alpha_i) =& \langle \log p(\boldsymbol{w_i}|\alpha_i)p(\alpha_i)\rangle + const \\ =& \langle \log(\sqrt{\frac{\alpha_i}{2\pi}}\exp(-\frac{1}{2}\boldsymbol{w_i^T}\alpha_i\boldsymbol{w_i})\cdot\frac{1}{\Gamma(\alpha_i)}b^a\alpha^{a-1}\exp(-b\alpha_i))\rangle + const \\ =& \langle\frac{1}{2}\log\alpha_i - \frac{1}{2}\boldsymbol{w_i^T}\alpha_i\boldsymbol{w_i} - \log\Gamma(\alpha_i) + (a-1)\log\alpha_i - b^{\alpha_i}\rangle + const \\ =& -(b + \frac{1}{2}\langle\boldsymbol{w_i^T}\boldsymbol{w_i}\rangle)\alpha_i + (a + \frac{1}{2} - 1)\log\alpha_i - \log\Gamma(\alpha_i) + const\end{aligned} \tag{46}$$

where $\langle\boldsymbol{w_i^T}\boldsymbol{w_i}\rangle$ can be further presented by:

$$\begin{aligned}\langle\boldsymbol{w_i^T}\boldsymbol{w_i}\rangle =& tr(\langle\boldsymbol{w_i^T}\boldsymbol{w_i}\rangle) \\ =& \langle tr(\boldsymbol{w_i^T}\boldsymbol{w_i})\rangle \\ =& tr(\boldsymbol{S_{w_i}} + \boldsymbol{m_{w_i}^T}\boldsymbol{m_{w_i}})\end{aligned} \tag{47}$$

and α_i follows a Gamma distribution:

$$\alpha_i \sim Gam(\alpha_i|a + \frac{1}{2}, b + \frac{1}{2}\langle\boldsymbol{w_i^T}\boldsymbol{w_i}\rangle) \tag{48}$$

and the expectation of α_i is :

$$\langle\alpha_i\rangle^* = \frac{a + \frac{1}{2}}{b + \frac{1}{2}tr(\boldsymbol{S_{w_i}} + \boldsymbol{m_{w_i}^T}\boldsymbol{m_{w_i}})} \tag{49}$$

$\boldsymbol{\tau}$:

$$\begin{aligned}\log q^*_{\tau_i}(\tau_i) =& \langle\sum_n \log p(y_n|\boldsymbol{\phi_n}, \boldsymbol{w_i}, \tau_i, \boldsymbol{z_n})p(\tau)\rangle + const\\ =&\langle\sum_n z_{ni}(\frac{1}{2}\log\tau_i - \frac{1}{2}\tau_i(y_n - \boldsymbol{w_i^T\phi_n})^2)\\ &-\log\Gamma(\tau_i) + a\log b + (a-1)\log\tau_i - b\tau\rangle + const\\ =&\langle(a+\frac{1}{2}\sum_n -1)\log\tau_i - \log\Gamma(\tau_i) - (b+\frac{1}{2}\sum_n z_{ni}(y_n - \boldsymbol{w_i^T\phi_n})^2\tau_i)\rangle + const\end{aligned} \tag{50}$$

τ_i follows a Gamma distribution:

$$\tau_i \sim Gam(\tau_i|a + \frac{1}{2}\sum_n z_{ni}, b + \frac{1}{2}\sum_n z_{ni}(y_n - \boldsymbol{w_i^T\phi_n})^2) \tag{51}$$

and the expectation of τ_i is:

$$\langle\tau_i\rangle^* = \frac{a + \frac{1}{2}\sum_n z_{ni}}{b + \frac{1}{2}\sum_n z_{ni}(y_n - \boldsymbol{w_i^T\phi_n})^2} \tag{52}$$

$\boldsymbol{\beta}$:

$$\begin{aligned}\log q^*_{\beta_i}(\beta_i) =& \langle\log p(\boldsymbol{v_i}|\beta_i)p(\beta_i)\rangle + const\\ =&\langle\log(\sqrt{\frac{\beta_i}{2\pi}}\exp(-\frac{1}{2}\boldsymbol{v_i^T}\beta_i\boldsymbol{v_i})\cdot\frac{1}{\Gamma(\beta_i)}b^a\beta^{a-1}\exp(-b\beta_i))\rangle + const\\ =&\langle\frac{1}{2}\log\beta_i - \frac{1}{2}\boldsymbol{v_i^T}\beta_i\boldsymbol{v_i} - \log\Gamma(\beta_i) + (a-1)\log\beta_i - b^{\beta_i}\rangle + const\\ =& -(b+\frac{1}{2}\langle\boldsymbol{v_i^T v_i}\rangle)\beta_i + (a+\frac{1}{2}-1)\log\beta_i - \log\Gamma(\beta_i) + const\end{aligned} \tag{53}$$

Distribution and the expectation of β_i can be presented by:

$$\beta_i \sim Gam(\beta_i|a+\frac{1}{2}, b+\frac{1}{2}\langle\boldsymbol{v_i^T v_i}\rangle) \tag{54}$$

$$\langle\beta_i\rangle^* = \frac{a+\frac{1}{2}}{b+\frac{1}{2}tr(\boldsymbol{S_{v_i}} + \boldsymbol{m_{v_i}^T m_{v_i}})} \tag{55}$$

$\boldsymbol{\epsilon}$:
$\boldsymbol{\epsilon}$ is optimized by M-Step in the variational framework. Refer to conclusion in eq.(19), it requires the expectation of posterior distribution $p(\mathcal{X}, \mathcal{Y}|\theta)$ with all latent variable fixed. It only depends on the sufficient statistics of $\theta = \{\boldsymbol{\epsilon}\}$, $\boldsymbol{\epsilon}$ after the update can be presented by:

$$
\begin{aligned}
\boldsymbol{\epsilon} =& \underset{\boldsymbol{\epsilon}}{argmax} \langle p(\mathcal{X}, \mathcal{Y}|\theta)\rangle \\
=& \underset{\boldsymbol{\epsilon}}{argmax} \langle \log \prod_{i=1}^{K} t(i, z_{ni}) * \sigma(-\boldsymbol{v_i}\boldsymbol{\Phi})\rangle \\
=& \underset{\boldsymbol{\epsilon}}{argmax} \langle \sum_{n} t(i, z_{ni})(\log \sigma(\epsilon_{ni}) - \frac{1}{2}\epsilon_{ni} - \lambda(\epsilon_{ni})(\boldsymbol{\phi_n^T}\langle \boldsymbol{v_i v_i^T}\rangle - \epsilon_{ni}^2))\rangle
\end{aligned} \tag{56}
$$

set the derivative of above equation w.r.t $\boldsymbol{\epsilon}$ to 0, we have:

$$
\frac{\partial}{\partial \epsilon_{ni}} = t(i, z_{ni})(\frac{1}{\sigma(\epsilon_{ni})}\sigma^{'}(\epsilon_{ni}) - \frac{1}{2} + \sigma(\epsilon_{ni}) - \frac{1}{2} - \lambda^{'}(\epsilon_{ni})(\boldsymbol{\phi_n^T}\langle \boldsymbol{v_i v_i^T}\rangle \boldsymbol{\phi_n} - \epsilon_{ni}^2)) \equiv 0 \tag{57}
$$

we have:

$$
\begin{aligned}
\epsilon_{ni}^2 =& \boldsymbol{\phi_n^T}\langle \boldsymbol{v_i v_i^T}\rangle \boldsymbol{\phi_n} \\
=& \boldsymbol{\phi_n^T}(\boldsymbol{S_{v_i}} + \boldsymbol{m_{v_i} m_{v_i}^T})\boldsymbol{\phi_n}
\end{aligned} \tag{58}
$$