Description of Constant-Level-Fitting Model

Luca Iten

January 30, 2024

This document describes the implementations of class *CLFModel*.

1 Model Description

The class *CLFModel* implements a Constant Level Fitting (CLF) model, where the maximum number of (assumed) levels M can be specified. This finite number of levels makes it fundamentally different from a standard PWC model, as the exact same level is potentially visited multiple times, depending on the given data.

A factor graph of a model achieving this desired behaviour is shown in Figure 1. It consists of M constant levels that are connected through a model selector as implemented in class *ModelSelector*. It is given N observations y_i, $i \in \{1, \dots, N\}$ of general dimension D. Generally, Gaussian message passing can not be performed in the depicted factor graph due to the multiplication nodes (because the product of two normally distributed RVs is no longer Gaussian). To tackle this issue, class *CLFModel* implements an iterative approach, where either the model selector vectors S_i or the levels $X_{i,m}$ are assumed to be known and fixed, while the estimation of the other quantities are improved. This causes the message passes through the multiplication notes to become simple scalar / vector multiplications, which can easily be handled by Gaussian message passing. Furthermore, it makes the factor graph loop-free.

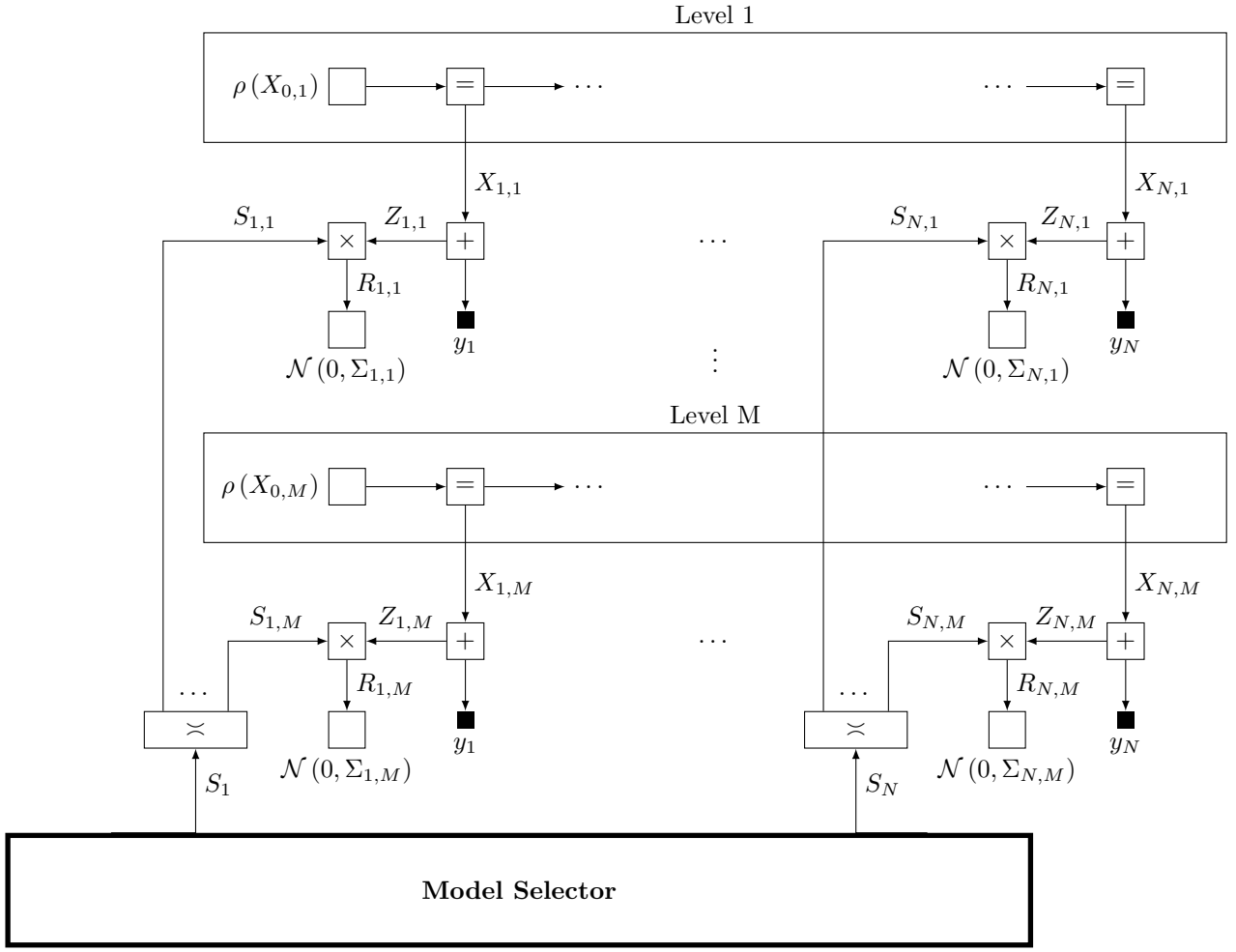

Figure 1: Factor graph of the CLF model.

2 Explanation of Implementations

In this Section the implementations of class *CLFModel* are explained. In particular, two different approaches to estimate the M levels for given estimations of S_i are discussed as well as initialization and scheduling of the whole method. Note that the naming and directions of the messages in Figure 1, the following explanations, and the actual implementations are all consistent. However, they can differ from the conventions used in other investigated models.

2.1 Initializing an Object of *CLFModel*

When initializing an object of class *CLFModel*, the number of observations N, their dimensions D, and the number of levels M have to be specified. Note that latter is the maximum number of levels that are used. Therefore, M should be chosen "high enough" depending on an initial guess, implicitly incorporating prior knowledge about the given observations.

Next to these standard configurations, the following optional adjustments can be made. A priori known levels can be specified in 'knownLevels' (for example a known "base" level at zero). This sets the priors $\rho\left(X_{0,m}\right)$, $m \in \{1, \ldots, M\}$ shown in Figure 1 to the corresponding values with very low uncertainty. The remaining priors (i.e., those of the levels that are not specified) are all set to zero-mean with very high uncertainty. Furthermore, the assumed observation noise per level can be specified by the respective covariance matrices in 'knownSigmas'. If nothing is given, i.i.d. noise with a variance of 1 is assumed.

2.2 Scheduling

When looking at the factor graph shown in Figure 1, two issues become apparent. Firstly, the factor graph is not loop-free, meaning that simple message passing algorithms are not guaranteed to find any solution. Secondly, the depicted multiplication nodes multiply two Gaussian quantities, meaning that their results $R_{i,m}$ are not normally distributed. Fortunately, both of these issues can be resolved with the following trick. Instead of solving the whole problem in one go, class *CLFModel* implements an iterative approach, alternately improving the model selector vectors S_i, $i \in \{1, \ldots, N\}$ and the levels X_m, $m \in \{1, \ldots, M\}$, while the other quantities are assumed to be fixed and perfectly known. For the improvement of S_i, this results exactly in the scenario discussed in the documentation of *ModelSelector*. For the improvement of X_m, the resulting factor graph is shown in Figure 2. How the corresponding levels are estimated in this second scenario is explained in Subsection 2.3.

Based on this fundamental idea, the estimation in class *CLFModel* is implemented as follows. Initially, the model selector is initialized without any prior knowledge (as described in the documentation of *ModelSelector*). The priors on the levels are initialized as described in Subsection 2.1. Next, the estimates of S_i and X_m are iteratively improved, starting by improving the model selection vector. Note that the estimation of S_i is based on IRLS, which is performed for

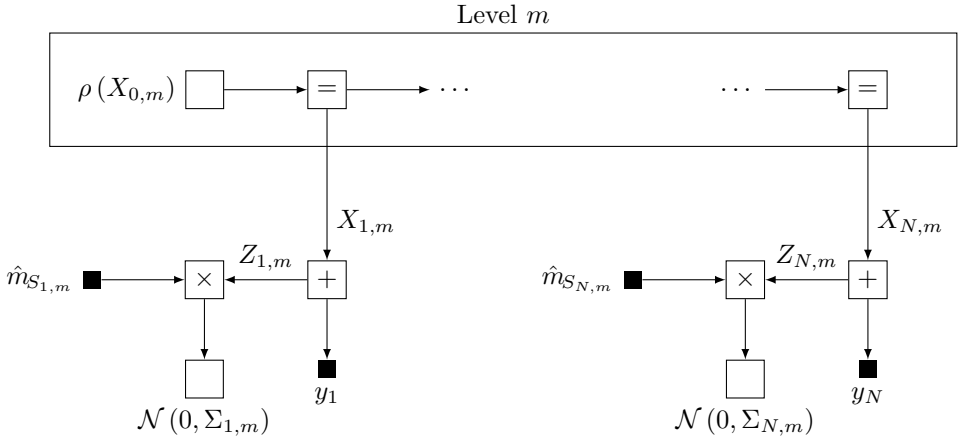

Figure 2: Factor graph of level estimation for fixed model selector vectors S_i.

a maximum of 'n_it_irls_s' iterations (or until convergence). The estimation of X_m does not involve any internal iterations. Finally, the algorithm converges if both, the change in the estimation of S_i and X_m, fall below the defined thresholds or after 'n_it_outer' iterative improvement steps.

2.3 Constant Level Estimation

This Subsection describes how the (unknown) constant levels are estimated for fixed values of S_i. A corresponding factor graph is shown in Figure 2. In the following, the two different approaches implemented in class *CLFModel* are described.

2.3.1 Direct Approach

By default, simple Gaussian message passing is performed in Figure 2 to find the MAP estimates of X_m. This method is used when 'levelEstType' = 'superPos' is selected.

In a first step, the backward messages through $X_{i,m}$ are calculated as

$$\overleftarrow{W}_{X_{i,m}} = \hat{m}_{S_{i,m}}^2 \Sigma_{i,m}^{-1} \tag{1}$$

$$\overleftarrow{\xi}_{X_{i,m}} = \overleftarrow{W}_{X_{i,m}} y_i \,. \tag{2}$$

From there, the MAP estimate of X_m is easily found as

$$\hat{V}_{X_m} = \left(\overrightarrow{W}_{X_{0,m}} + \sum_{i=1}^{N} \overleftarrow{W}_{X_{i,m}} \right)^{-1} \tag{3}$$

$$\hat{m}_{X_m} = \hat{V}_{X_m} \left(\overrightarrow{\xi}_{X_{0,m}} + \sum_{i=1}^{N} \overleftarrow{\xi}_{X_{i,m}} \right), \tag{4}$$

where $\overrightarrow{W}_{X_{0,m}}$ and $\overrightarrow{\xi}_{X_{0,m}}$ have been specified during initialization (as explained in 2.1), incorporating possible prior knowledge.

2.3.2 Selective Approach

In the previously described approach, the elements of S_i can be seen as weighting factors for the MAP estimation of X_m. This interpretation seems appropriate to the task considering that the model selector vector should converge to an all-$\{0,1\}$ solution, effectively grouping the set of observations into M clusters, each corresponding to one level. But what happens before S_i has converged? During this stage, the described approach (in some sense) calculates a weighted mean, causing the MAP estimates to be biased towards the over-all mean of all solutions! This effect eases over time, but the speed of convergence is very slow.

A straight-forward way to speed this process up is to change the "soft" weighting factors (i.e., the elements of S_i) into "hard" selectors. Thereby, the observations are first grouped into M groups, where the m-th group contains all observations with indices in

$$\mathcal{I}_m \triangleq \left\{ i \in \{1,\dots,N\} \mid \arg\max_{m'} S_{i,m'} = m \right\}, \quad m \in \{1,\dots,M\}. \tag{5}$$

According to this grouping, the estimated levels are calculated as

$$\hat{V}_{X_m} = \left(\overrightarrow{W}_{X_{0,m}} + \sum_{i \in \mathcal{I}_m} \Sigma_m^{-1} \right)^{-1} \tag{6}$$

$$= \left(\overrightarrow{W}_{X_{0,m}} + |\mathcal{I}_m| \cdot \Sigma_m^{-1} \right)^{-1} \tag{7}$$

$$\hat{m}_{X_m} = \hat{V}_{X_m} \left(\overrightarrow{\xi}_{X_{0,m}} + \sum_{i \in \mathcal{I}_m} \Sigma_m^{-1} y_i \right) \tag{8}$$

$$= \hat{V}_{X_m} \left(\overrightarrow{\xi}_{X_{0,m}} + \Sigma_m^{-1} \sum_{i \in \mathcal{I}_m} y_i \right). \tag{9}$$

As already described in the 'Direct Approach', $\overrightarrow{W}_{X_{0,m}}$ and $\overrightarrow{\xi}_{X_{0,m}}$ have been specified during initialization.