Description of Return-To-Base Model

Luca Iten

January 30, 2024

This document describes the implementations of class *RTBModel*.

1 Model Description

The class *RTBModel* can be seen as an extended PWC model (implemented by class *PWCModel*) with a tunable preference to a known base level. In other words, it infers the means of N given data points y_i of dimension D, where y_i is assumed to originate either from an unknown PWC model or a perfectly known model. Naturally, the perfectly known model could correspond to some constant base level (for example the all-zero level). This general way of modelling the data also motivates the name Return-To-Base (RTB) model.

Building on already described methods, the class *RTBModel* consists of an PWC model, a (perfectly known) base model, and a model selector. A factor graph visualizing the interplay between these building blocks is shown in Figure 1. Note the similarities of it structure with the factor graphs used to describe the model selector (shown in documentation of class *ModelSelector*) and the CLF model (class *CLFModel*).

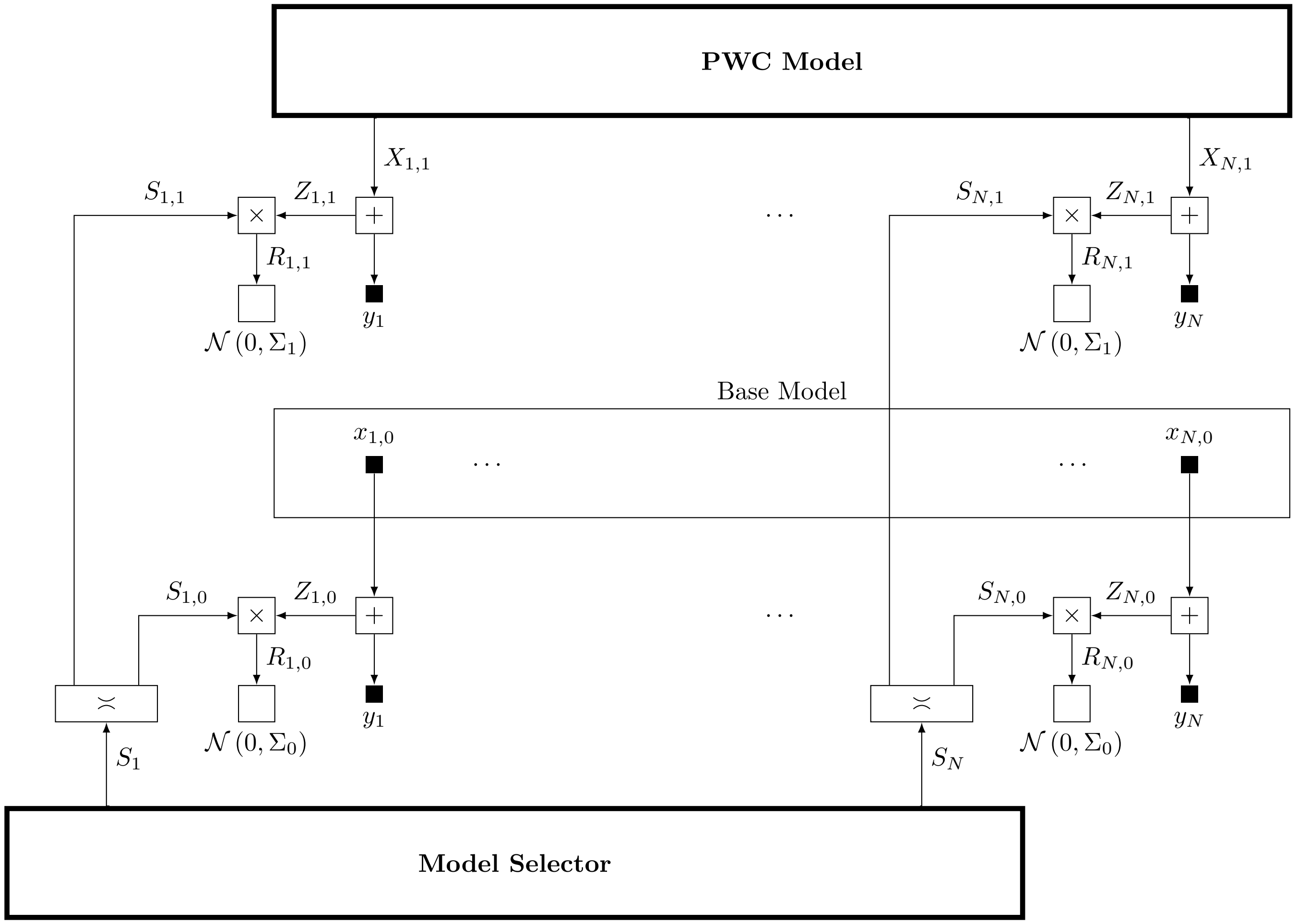

Figure 1: Factor graph of the RTB model.

2 Explanation of Implementations

This Section describes how a PWC output with a preference to a specified base level is fitted to the given observations in class *RTBModel*. Due to the structural similarities of the depicted model in Figure 1 to the one discussed in the documentation of class *CLFModel*, some aspects of the implementations are omitted in this document to avoid repetitions of the same concepts. Furthermore, note that the naming and directions of the messages in Figure 1, the implementations, and the following explanations are all consistent.

2.1 Initializing an Object of *RTBModel*

When initializing an object of class *RTBModel*, the number of observations N along with their dimension D must be specified. Furthermore, the outputs of the base model must be specified. If only one data point (of dimension D)is given, a constant base level is assumed. Finally, the assumed constant observation noise per model must be specified by their respective covariance matrices. This last specification can be seen as a tuning factor, as the scaling of these covariance matrices heavily influences the model selection process. To further explain this behaviour, think of the case where the observed data should be explained by the specified base model. Without any further prior knowledge (i.e., tuning), this subset of observations could be explainable by both models equally well, as the PWC model can fit the data arbitrarily well. Due to the probabilistic nature of the model selector however, the base model can be given a (slight) preference by upscaling its associated observation noise covariance matrix. In other words, higher assumed observation noise variances allow the model to "explain" higher deviations from the expected mean with higher probability, causing the algorithm to prefer the this model in the discussed scenario!

Optionally, it is further possible to specify the initial values of $X_{i,m}$. This can be used to incorporate further knowledge about the general model.

2.2 Scheduling

Scheduling is handled very similar to the implementations of class *CLFModel*. However, it is noteworthy that in the implementations of *RTBModel* both, the improvement of S_i as well as the improvement of $X_{i,1}$ are performed by IRLS (in *CLFModel*, the level estimation did not require any iterations). This obviously impacts the complexity of the algorithm. Furthermore, it poses another "tuning choice", as the performance is indeed affected by the maximum number of IRLS iterations per outer iteration (i.e., iterations between the two improvement steps). Extensive simulations showed that the model selector generally needs less iterations than the PWC model, often preventing premature conversion.