Description of Noise Covariance Model

Luca Iten

February 6, 2024

This document describes the implementations of class *CovModel*.

1 Model Description

The class *CovModel* implements an iterative method to estimate the (potentially evolving) covariance matrices of applied zero-mean Gaussian noise. In its heart, dedicated NUV priors are constructed to calculate the exact Gaussian message passes through normal nodes of general dimension. This is a generalization of the idea presented in [1] from the scalar to the more general D dimensional case.

Next to implementing this novel Gaussian message passing method through normal nodes, class *CovModel* also provides the framework to calculate the MAP estimate of the covariance matrices in question. Thereby, these covariance matrices can either be modelled 'Constant' or 'PWC' (i.e., piecewise constant). Later case builds on class *PWCModel*, handling the MAP estimates of J_i, $i \in \{1, \dots, N\}$ efficiently (method is explained in the documentation of *PWCModel*). A factor graph visualizing this general concept is shown in Figure 1.

2 Explanation of Implementations

This Section describes the implementations of class *CovModel*. In particular, the exact messages through normal nodes of general dimension D are derived. The subsequent processing of these messages to calculate the MAP estimates of J_i is not discussed in detail, as it is handled by other classes (either by class *ConstModel* or *PWCModel*, both extensively explained in their respective documentations). Note that the naming and direction of the messages shown in Figure 1 correspond with the following descriptions and the actual implementations. However, they may differ from conventions used in other models!

2.1 Initializing an Object of *CovModel*

When initializing an Object of class *CovModel*, the number of observations N, their dimension D, and the covariance evolution type 'evolType' $\in$ ['pwc', 'constant'] must be specified. Later states which kind of model should be used

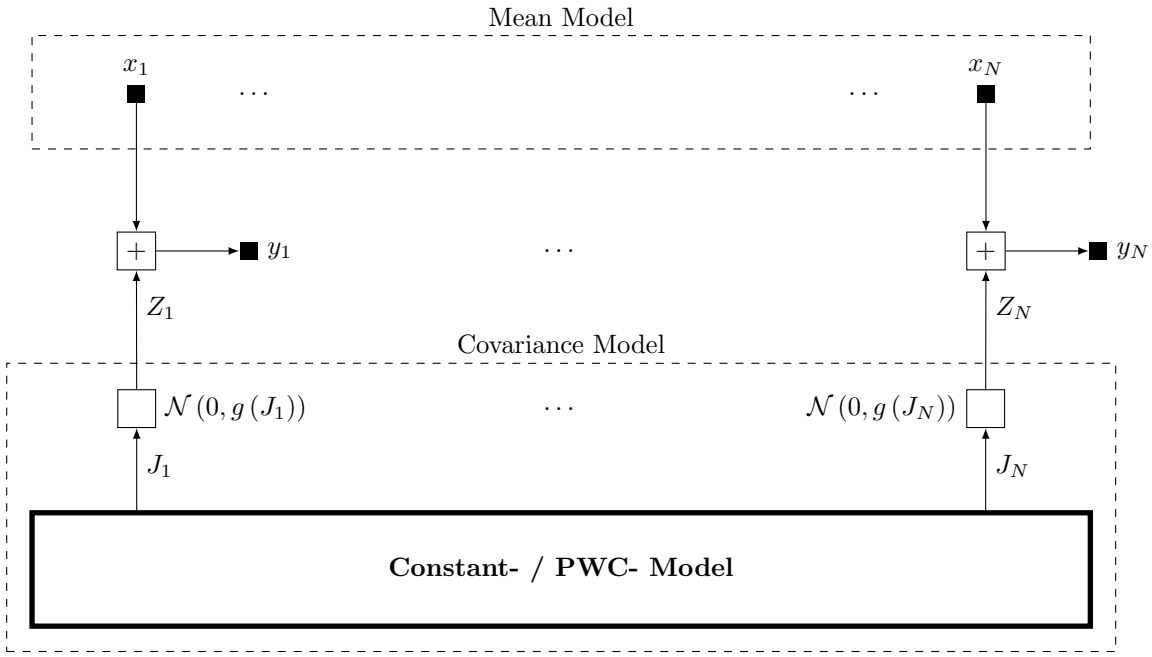

Figure 1: Factor graph of noise covariance matrix estimation model.

to calculate the MAP estimates of J_i, either a PWC model (implemented by class *PWCModel*) or a constant model (class *ConstModel*). The corresponding model is initialized too.

Furthermore, the dimension of J_i (denoted by $f(D)$)is calculate and saved. Note that it only depends on the dimension of the given observation, as it describes the number of non-zero elements in an upper-diagonal matrix of dimension D. The reasoning behind this is explained in Subsection 2.2. $f(D)$ is calculated as

$$f(D) = \frac{D(D+1)}{2}\,. \tag{1}$$

Finally, the initializing function of *CovModel* defines the initial mean values of J_i in such a way that the corresponding recovered noise covariance matrices would all be identity matrices. The uncertainty of this initial guess is initialized to be high.

2.2 Message Passes Through Normal Nodes

This Subsection extensively describes how class *CovModel* handles Gaussian message passes through normal nodes (i.e., how messages from Z_i to J_i evolve in Figure 1). These descriptions include the derivation of exact messages through normal nodes in general dimension D, therefore generalizing the ideas presented in [1] from scalar to higher dimensions. Understanding these derivations is not crucial to understand the implementations of class *CovModel*. Still, they are presented here for completeness and to clarify possible questions.

In the following, a manageable factor graph representation of normal nodes is derived in 2.2.1 and 2.2.2. These explanations motivate the definition of J_i, as it is described in 2.2.3. Finally, 2.2.4 states the resulting expressions for the means and covariance messages through general normal nodes. Note that the time indices $i \in \{1, \dots, N\}$ are omitted in all these derivations. Apart from that, the used naming in descriptions and mathematical expressions matches the names and directions of messages shown in Figure 1.

It is again pointed out that the following described ideas (basing the implementations of *CovModel*) are not fundamentally new, but a generalization of the work presented in [1].

2.2.1 Multiplicative Factor Graph Representation of Normal Nodes

In the factor graph shown in Figure 1, Z is a multi-dimensional normally distributed random vector of zero mean and covariance matrix $g(J) = \Sigma$. It is well known that Z can be expressed by

$$Z = AU, \quad A \in \mathbb{R}^{D\times D}, \quad Z, U \in \mathbb{R}^D\,, \tag{2}$$

where A is the Cholesky factor of the covariance of Z and U is i.i.d. with unit variance, i.e.,

$$\Sigma = AA^\intercal \tag{3}$$

$$U \sim \mathcal{N}(0, \mathbf{I}_D)\ . \tag{4}$$

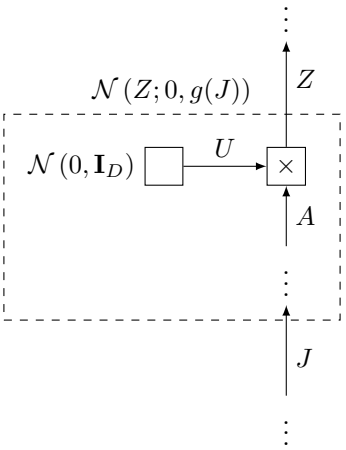

Figure 2: Factor Graph of normal node.

Due to the nature of the Cholesky decomposition, A is an upper-triangular matrix and, therefore, has at most $f(D) = \frac{D(D+1)}{2}$ non-zero elements. Furthermore, Σ is positive (semi-) definite, causing A to be positive (semi-) definite too.

Having in mind these facts, the expression in Equation (2) can be represented by the factor graph in Figure 2. Note that the vertical dots between A and J subsume some unknown transformation $\tilde{g} : \mathbb{R}^{f(D)} \to \mathbb{R}^{D \times D}$, the detailed nature of which is not important at this point.

2.2.2 Alternative Representation

Performing naive Gaussian message passing in Figure 2 is obviously not possible, as the product of two normally distributed RVs itself is not normally distributed. However, [1] describes a neat trick that can be adapted to this problem.

Following this idea, the subsequent observation is made

$$\mathcal{N}(Z; 0, g(J)) = \int_{-\infty}^{\infty} \mathcal{N}(U; 0, \mathbf{I}_D)\, \delta(AU - Z)\, \mathrm{d}U \tag{5}$$

$$= \frac{1}{|\det(A)|} \int_{-\infty}^{\infty} \mathcal{N}(U; 0, \mathbf{I}_D)\, \delta\left(U - A^{-1}Z\right) \mathrm{d}U \tag{6}$$

$$= |\det\left(A^{-1}\right)| \int_{-\infty}^{\infty} \mathcal{N}(U; 0, \mathbf{I}_D)\, \delta\left(U - A^{-1}Z\right) \mathrm{d}U\,. \tag{7}$$

Here, $\delta(\cdot)$ denotes the Dirac-delta function. The equality in (6) follows Proposition 2.3 in [2]. In words, line (7) states that the distribution of Z can be expressed as the product of its argument with A^{-1}, scaled by $|\det\left(A^{-1}\right)|$, where the joint distribution of the elements of A^{-1} is Gaussian. A corresponding factor graph of this point of view is depicted in Figure 3.

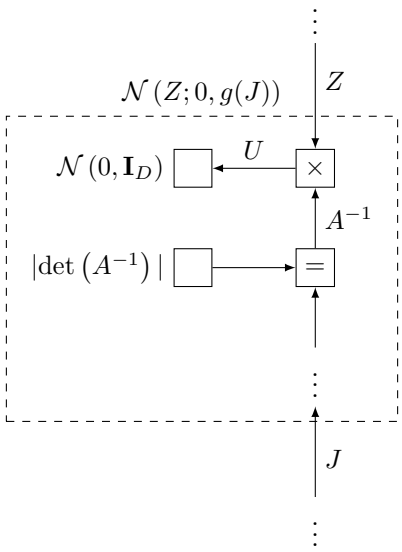

Figure 3: Modified Factor Graph of normal node.

At this point, a few things can be stated. Firstly, for perfectly known values of Z, Gaussian message passing through the multiplication node in Figure 3 is possible. Secondly, because A is an upper-triangular matrix, A^{-1} is an upper-triangular matrix too. Therefore, its determinant is equal to the product of its diagonal elements. This results in the following simplification

$$|\det(A^{-1})| = \prod_{d=1}^{D} |\tilde{a}_{dd}|, \tag{8}$$

where $\tilde{a}_{k\ell}$ denotes the elements of A^{-1}. Thirdly, there exists an NUV prior representation for absolute values. In conclusion, Gaussian message passing in Figure 3 is possible!

2.2.3 Definition of J

From Figure 3 it can be seen that J should somehow describe the elements of A^{-1} (or implicitly those of A as previously stated). A^{-1} is an upper-diagonal matrix of dimension $D \times D$, whose non-zero elements are jointly normal distributed. In the following, these elements are are denoted as

$$A^{-1} = \begin{bmatrix} \tilde{a}_{11} & \tilde{a}_{12} & \dots & \tilde{a}_{1D} \\ 0 & \tilde{a}_{22} & \dots & \tilde{a}_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{a}_{DD} \end{bmatrix}. \tag{9}$$

To "capture" all this information, J must have exactly $f(D) = D(D+1)/2$. In the implementations of class *CovModel*, it is defined as

$$J \triangleq \left[\tilde{a}_{11} \;\; \tilde{a}_{22} \;\; \ldots \;\; \tilde{a}_{\mathrm{DD}} \middle| \tilde{a}_{12} \;\; \tilde{a}_{23} \;\; \ldots \;\; \tilde{a}_{\mathrm{D\text{-}1D}} \middle| \ldots \middle| \tilde{a}_{\mathrm{1D}} \right]^{\mathsf{T}} . \tag{10}$$

I.e., J is constructed by reading the increasingly offset diagonal vectors of A^{-1}. This direct correspondence to the elements of A^{-1} also motivates the name of J, which is the *Vectorised Inverse Cholesky Factor* (VICF). Naturally, the ordering of the elements of J is arbitrary. However, this particular choice makes its implementation quite easy.

Given this definition of J, the following matrix $\tilde{Z}$ is constructed

$$\tilde{Z} \triangleq \left[\begin{array}{ccccc|ccccc|c|c} z_1 & 0 & \ldots & 0 & 0 & z_2 & 0 & \ldots & 0 & & z_{\mathrm{D}} \\ 0 & z_2 & \ldots & 0 & 0 & 0 & z_3 & \ldots & 0 & & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ 0 & 0 & \ldots & z_{\mathrm{D\text{-}1}} & 0 & 0 & 0 & \ldots & z_{\mathrm{D}} & & 0 \\ 0 & 0 & \ldots & 0 & z_{\mathrm{D}} & 0 & 0 & \ldots & 0 & & 0 \end{array}\right] \tag{11}$$

$$= \left[\tilde{Z}_1 \quad \tilde{Z}_2 \quad \ldots \quad \tilde{Z}_{\mathrm{D}} \right] \in \mathbb{R}^{D \times D(D+1)/2} , \tag{12}$$

where

$$\tilde{Z}_d \triangleq \left[\begin{array}{cccc} z_{\mathrm{d}} & 0 & \ldots & 0 \\ 0 & z_{\mathrm{d}+1} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & z_{\mathrm{D}} \\ \hline 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & 0 \end{array}\right] \in \mathbb{R}^{D \times D-i+1} . \tag{13}$$

In words, $\tilde{Z}$ is defined as D concatenated matrices $\tilde{Z}_{\mathrm{d}}$, $d \in \{1, \ldots, D\}$, where $\tilde{Z}_{\mathrm{d}}$ is defined as a diagonal matrix stacked on an all zero matrix. The elements of the respective diagonal matrix are the last $D+1-d$ elements of the observed zero-mean Gaussian noise Z in Figure 3. Note that with these definitions, the following equality holds

$$U = A^{-1}Z = \tilde{Z}J . \tag{14}$$

Finally, the factor graph in Figure 3 can be adapted to the one shown in Figure 4. Note that the absolute value of the determinant of A^{-1} has been simplified according to Equation (8). Furthermore, a positivity constraint is added to de-emphasize the trivial all-zero solution (following the argumentation made in [1]). This constraint can be added as $\tilde{a}_{dd}$, $d \in \{1, \ldots, D\}$ correspond to the diagonal elements of A^{-1}, which are known to be positive anyway.

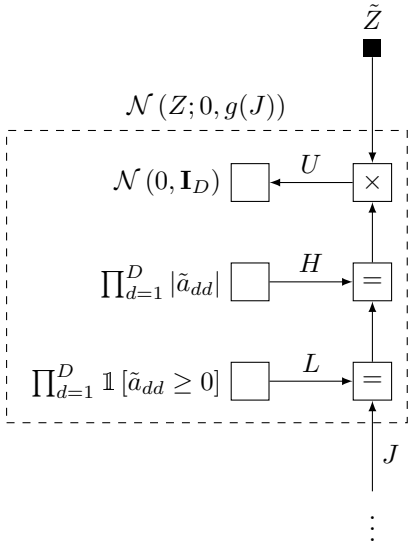

Figure 4: Factor graph representation of normal node, based on which class *CovModel* is implemented.

2.2.4 Resulting Messages Generated by the Normal Node NUV Prior

Everything that has been discussed in Subsection 2.2 up to this point lead to the factor graph shown in Figure 4. Note that this is an exact representation of normal nodes in general dimensions, where the "generated" noise Z is observed and used to construct $\tilde{Z}$. Furthermore, it has been stated that all remaining quantities are jointly normal distributed, leading to the conclusion that Gaussian message passing can be used to infer the values of J. In the following, the corresponding backward messages through J are derived. Note that these derivations build on previously described NUV priors, implementing the absolute-value and positivity constraints shown in Figure 4. The resulting messages can then be seen as being generated by a "Normal Node" NUV prior.

In Figure 4, the two NUV constraints generating the messages trough H and L are only applied to the first D elements of J. The corresponding values per dimension can be calculated as

$$\overrightarrow{\xi}_{H_d} = 0 \tag{15}$$

$$\overrightarrow{W}_{H_d} = -\frac{1}{\hat{m}_{J_d}^2} \tag{16}$$

$$\overrightarrow{\xi}_{L_d} = \beta_\ell \tag{17}$$

$$\overrightarrow{W}_{L_d} = \frac{\beta_\ell}{|\hat{m}_{J_d}|}, \tag{18}$$

where the subscript $\cdot_d$ denotes the d-th element and takes values $d \in [1, \ldots, D]$.

The symbols $\hat{m}_J$ denote the posterior mean estimates of J from the previous iteration. β_ℓ can be seen as a tuning factor, determining how "strong" the positivity prior is. Furthermore, the backward messages through the multiplication node are

$$\overleftarrow{\xi}_{J'} = \mathbf{0} \tag{19}$$

$$\overleftarrow{W}_{J'} = \tilde{Z}^\intercal \tilde{Z}\,, \tag{20}$$

where $\mathbf{0}$ denotes an all-zero vector.

Accordingly, the resulting backward messages through J (i.e., the messages generated by the normal node NUV prior) become

$$\overleftarrow{\xi}_J = \sum_{d=1}^{D} C_d \cdot \beta_\ell \tag{21}$$

$$\overleftarrow{W}_J = \tilde{Z}^\intercal \tilde{Z} + \sum_{d=1}^{D} C_d C_d^\intercal \cdot \frac{\beta_\ell \left|\hat{m}_{J_d}\right| - 1}{\hat{m}_{J_d}^2}\,, \tag{22}$$

where C_d is a vector of dimension $D(D+1)/2$, whose values are all zero except for its d-th value begin 1.

2.3 Estimation of J_i

Subsection 2.2 describes how class *CovModel* calculates the backward messages through J_i in Figure 1. It is also described that these messages are based on NUV priors. Therefore, estimating the means and covariance matrices of J_i requires multiple iterations of IRLS.

In the implementations of class *CovModel*, the maximum number of iterations can be specified when calling the estimation function. In each iteration, first the backward messages through J_i according to Equations (21) and (22) are calculated. Note that these messages depend on the current mean estimates of J. Afterwards, the calculated dual mean and precision matrix messages are passed to the evolution model. This evolution model is either PWC or constant, depending on how the object of *CovModel* is initialized. The respective model then updates the mean and covariance matrix estimates of J_i. During this update step, the relative change of the mean estimate is compared against a threshold. If this change falls below the threshold, the algorithm terminates.

2.4 Recovering Σ_i from J_i

According to Equations (9) and (10), A_i^{-1} and J_i are defined as

$$A_i^{-1} = \begin{bmatrix} \tilde{a}_{i,11} & \tilde{a}_{i,12} & \dots & \tilde{a}_{i,1D} \\ 0 & \tilde{a}_{i,22} & \dots & \tilde{a}_{i,2D} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{a}_{i,DD} \end{bmatrix}\,, \tag{23}$$

and

$$J_i = \left[\, \tilde{a}_{i,11} \;\; \tilde{a}_{i,22} \;\; \ldots \;\; \tilde{a}_{i,\mathrm{DD}} \middle| \tilde{a}_{i,12} \;\; \tilde{a}_{i,23} \;\; \ldots \;\; \tilde{a}_{i,\mathrm{D\text{-}1D}} \middle| \ldots \middle| \tilde{a}_{i,1\mathrm{D}} \, \right]^{\mathsf{T}} , \tag{24}$$

respectively (including time indices $i \in \{1, \ldots, N\}$).

Therefore, given the mean estimates of J_i as $\hat{m}_{J_i}$, estimates of $\hat{A}_i^{-1}$ can be constructed as

$$\hat{A}_i^{-1} = \begin{bmatrix} \hat{m}_{J_{i,1}} & \hat{m}_{J_{i,D+1}} & \ldots & \hat{m}_{J_{i,D(D+1)/2}} \\ 0 & \hat{m}_{J_{i,2}} & \ldots & \hat{m}_{J_{i,D(D+1)/2-1}} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \hat{m}_{J_{i,D}} \end{bmatrix} . \tag{25}$$

With these reconstructed of $\hat{A}_i^{-1}$, the estimated noise covariance matrices $\hat{\Sigma}_i$ can be calculated as

$$\hat{\Sigma}_i = \hat{A}_i \hat{A}_i^{\mathsf{T}} \tag{26}$$

where $\hat{A}_i$ is recovered as

$$\hat{A}_i = \left(\hat{A}_i^{-1} \right)^{-1} . \tag{27}$$

In Equation (26), fact that A_i is a Cholesky factor of Σ_i has been used (i.e., Equation (3)).

Note that the inversion in Equation (27) is computationally cheap, as $\hat{A}_i^{-1}$ is an upper-triangular matrix. However, this inversion can be omitted entirely if, instead of the noise covariance matrices, its precision matrices are estimated. In this case, $\hat{\Sigma}_i^{-1}$ can be directly calculated from $\hat{A}_i^{-1}$ as

$$\hat{\Sigma}_i^{-1} = \left(\hat{A}_i^{-1} \right)^{\mathsf{T}} \hat{A}_i^{-1} . \tag{28}$$

Class *CovModel* provides dedicated functions to calculate both, the noise covariance and precision matrix estimates given the current mean estimates of J_i.

References

[1] H.-A. Loeliger, "On nup priors and gaussian message passing," 2023.

[2] L. Zhang, "Dirac delta function of matrix argument," 2018.