Modified Bryson-Frazier Smoothing and Hyperparameter Learning for Temporal Gaussian Process Regression

Tom Colemont — *Leuven Gravity Institute, Department of Electrical Engineering (STADIUS), KU Leuven, Belgium*
Brecht Evens — *Leuven Gravity Institute, Department of Electrical Engineering (STADIUS), KU Leuven, Belgium*
Tjonnie G.-F. Li — *Leuven Gravity Institute, Department of Physics & Astronomy, KU Leuven, Belgium*
Frederik De Ceuster — *Leuven Gravity Institute, Department of Physics & Astronomy, KU Leuven, Belgium*

Abstract

One-dimensional Gaussian processes with stationary, integrable kernel functions admit exact or arbitrarily accurate state-space representations, enabling linear-time inference through Kalman filtering and Rauch-Tung-Striebel (RTS) smoothing. However, the RTS smoother requires inversion of predicted state covariance matrices, which can become ill-conditioned and may therefore lead to numerical instabilities. In this work, we revisit the modified Bryson-Frazier (MBF) smoother as an alternative to the RTS smoother for Gaussian process regression in its state-space representation. In addition to reducing computational cost and memory requirements, the MBF smoother computes the same posterior distributions as the RTS smoother while avoiding the problematic covariance matrix inversion and the associated numerical instabilities. Furthermore, we demonstrate that the intermediate quantities computed by the MBF smoother can be reused to compute gradients of the negative log marginal likelihood, enabling kernel hyperparameter learning with minimal additional cost. Together, these results establish the MBF smoother as a unified and numerically robust approach to inference and kernel hyperparameter learning for one-dimensional Gaussian process regression.

Proceedings of the 2nd International Conference on Probabilistic Numerics (ProbNum) 2026, Lappeenranta, Finland. PMLR Volume 341.

1 Introduction

Gaussian processes (GPs) provide flexible and nonparametric models for regression, and play a central role in probabilistic numerics (PN). However, standard GP inference has a cubic complexity in the number of data points, limiting scalability (Williams and Rasmussen, 2006). A common strategy to overcome this limitation is to leverage the class of Markovian kernel functions. For such kernels, GP regression can be reformulated as inference in an equivalent linear Gaussian state-space model, enabling linear-time inference through Bayesian filtering and smoothing (Hartikainen and Särkkä, 2010).

Since these state-space models are linear and Gaussian, the filtering and smoothing distributions admit closed-form recursions: the Kalman filter and Rauch-Tung-Striebel (RTS) smoother, respectively (Kalman, 1960; Rauch et al., 1965). Although exact finite-dimensional state-space representations are available only for Markovian kernels, any one-dimensional GP with a stationary, integrable kernel can be approximated arbitrarily well by such models (Loper et al., 2021). The combination of linear-time inference and broad kernel applicability has made this approach widely used in PN, particularly in probabilistic ODE solvers (Tronarp et al., 2019).

Despite reducing the computational complexity in the number of data points from cubic to linear, Kalman filtering and RTS smoothing remain cubic in the state-space dimension. This dependence on the state-space dimension becomes problematic in high-dimensional settings, such as spatio-temporal regression. In response, computation-aware Kalman filters and smoothers have been developed that approximate both the filtering and smoothing distributions while tracking the induced approximation errors (Pförtner et al., 2025).

A second, distinct limitation concerns numerical stability. The Kalman filter and, in particular, the RTS smoother may suffer from numerical instabilities when state covariance matrices become ill-conditioned or effectively singular. Such regimes arise naturally when PN methods aim to compute highly accurate solutions, where the posterior uncertainty may become very small. This is especially problematic for RTS smoothing, which requires inversion of the predicted state covariance matrix. When this matrix is ill-conditioned or singular, the RTS recursions may become unstable and break down. Coordinate transformations and square-root formulations can mitigate these issues, but incur additional computational cost (Kraemer and Hennig, 2024).

In this paper, we revisit a classical alternative to the RTS smoother from control and estimation theory: the modified Bryson-Frazier (MBF) smoother. Originally proposed by Bierman (1973), this reformulation of the RTS smoother avoids the problematic inversion of the predicted covariance matrix, while additionally reducing computational and memory requirements. Closely related smoothing formulations have appeared under different names, including the *disturbance smoother* of Kohn and Ansley (1989) and, more recently, the *inversion-free RTS smoother* in Pförtner et al. (2025) in the context of computation-aware methods, where its favorable numerical properties were also noted.

This paper makes the following contributions:

1. We present the MBF smoother as a numerically stable alternative to the RTS smoother for state-space inference in PN, retaining linear complexity.
2. We provide a unified dynamic-programming view of Kalman filtering, RTS, and MBF smoothing, showing that the MBF recursions arise as the adjoint equations of the same optimization problem.
3. We show that the MBF variables can be reused to compute gradients of the negative log marginal likelihood, enabling efficient hyperparameter optimization with minimal additional cost.

Together, these results establish the MBF smoother as a unified and numerically robust approach to inference and hyperparameter optimization for one-dimensional GP regression. Although we focus on regression for clarity, all results naturally extend to probabilistic numerical methods based on state-space inference.

2 Inference in State-Space Models

State-space methods provide an efficient computational framework for performing inference in GP models with Markovian structure. This section summarizes the conventional filtering and smoothing algorithms, and discusses their numerical aspects.

As shown by Hartikainen and Särkkä (2010); Loper et al. (2021), any one-dimensional GP with a stationary, integrable kernel can be represented either exactly or approximated arbitrarily well by a linear Gaussian state-space model,

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \tag{1}$$
$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_k \tag{2}$$

where $\mathbf{x}_k \in \mathbb{R}^d$ denotes the latent state at time step k, $\mathbf{y}_k \in \mathbb{R}^m$ the measurement, and $\mathbf{q}_k$ and $\mathbf{r}_k$ the process and measurement noise, respectively,

$$\mathbf{q}_k \sim \mathcal{N}(0, \mathbf{Q}_k) \tag{3}$$
$$\mathbf{r}_k \sim \mathcal{N}(0, \mathbf{R}_k) \tag{4}$$

with covariance matrices $\mathbf{Q}_k \in \mathbb{R}^{d\times d}$ and $\mathbf{R}_k \in \mathbb{R}^{m\times m}$. Together with the transition matrix $\mathbf{A}_k \in \mathbb{R}^{d\times d}$ and measurement matrix $\mathbf{H}_k \in \mathbb{R}^{m\times d}$, these quantities define the state-space model. This equivalent representation reformulates GP regression as a state estimation problem, which can be solved efficiently through Kalman filtering and RTS smoothing (Särkkä and Svensson, 2023).

2.1 Kalman Filtering

For linear Gaussian state-space models, Bayesian filtering admits a closed-form solution, known as the Kalman filter (Kalman, 1960). The filter starts from an initial distribution $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{m}_0, \mathbf{P}_0)$, where $\mathbf{P}_0$ is the stationary prior covariance. Assuming that $\mathbf{x}_0$, $\mathbf{q}_k$ and $\mathbf{r}_k$ are mutually independent, it proceeds in two steps:

Predict: the predicted distribution of the latent state is recursively determined as:

$$\mathbf{x}_k \mid \mathbf{y}_{1:k-1} \sim \mathcal{N}(\mathbf{m}_k^-, \mathbf{P}_k^-) \tag{5}$$
$$\mathbf{m}_k^- = \mathbf{A}_{k-1}\mathbf{m}_{k-1} \tag{6}$$
$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^\top + \mathbf{Q}_{k-1} \tag{7}$$

Filter: the filtered distribution of the latent state incorporates a new measurement and reads:

$$\mathbf{x}_k \mid \mathbf{y}_{1:k} \sim \mathcal{N}(\mathbf{m}_k, \mathbf{P}_k) \tag{8}$$
$$\mathbf{z}_k = \mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^- \tag{9}$$
$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k \tag{10}$$
$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top\mathbf{S}_k^{-1} \tag{11}$$
$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k\mathbf{z}_k \tag{12}$$
$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^- \tag{13}$$

If no measurement is available at time step k, the update step is omitted and the filtered distribution coincides with the predicted distribution, effectively extrapolating the state estimate based on prior information. In conclusion, the Kalman filter computes the distribution of the latent state at time step k, conditioned on all measurements up to and including that time step.

2.2 Rauch-Tung-Striebel Smoothing

Bayesian smoothing determines the smoothing distribution of the latent state at time step k conditioned on all measurements up to and including a final time step T. For linear Gaussian state-space models, it admits a closed-form solution, known as the RTS smoother (Rauch et al., 1965). Starting from the final filtered state $\mathbf{x}_T \sim \mathcal{N}(\mathbf{m}_T, \mathbf{P}_T)$, a backward pass from $k = T-1 \ldots 0$ uses the predicted and filtered distributions to compute the smoothing distributions as:

$$\mathbf{x}_k \mid \mathbf{y}_{1:T} \sim \mathcal{N}(\mathbf{m}_k^s, \mathbf{P}_k^s) \tag{14}$$
$$\mathbf{J}_k = \mathbf{P}_k\mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1} \tag{15}$$
$$\mathbf{m}_k^s = \mathbf{m}_k + \mathbf{J}_k(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \tag{16}$$
$$\mathbf{P}_k^s = \mathbf{P}_k + \mathbf{J}_k(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\mathbf{J}_k^\top \tag{17}$$

2.3 Numerical Considerations

Although Kalman filtering and RTS smoothing provide exact inference for linear Gaussian state-space models, their practical implementation may suffer from numerical instabilities.

A first source of numerical instability arises in the covariance update equations of both the Kalman filter and the RTS smoother. In their standard form, these updates involve repeated subtractions of positive definite matrices. Due to the accumulation of numerical errors, the resulting matrices may lose positive definiteness.

In addition, the standard Kalman filter covariance update does not enforce symmetry by construction. To mitigate these issues, the covariance updates can be reformulated in their so-called Joseph form (Kraemer and Hennig, 2024):

$$\mathbf{B}_k = \mathbf{I} - \mathbf{K}_k\mathbf{H}_k \tag{18}$$
$$\mathbf{G}_k = \mathbf{I} - \mathbf{J}_k\mathbf{A}_k \tag{19}$$
$$\mathbf{P}_k = \mathbf{B}_k\mathbf{P}_k^-\mathbf{B}_k^\top + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^\top \tag{20}$$
$$\mathbf{P}_k^s = \mathbf{G}_k\mathbf{P}_k\mathbf{G}_k^\top + \mathbf{J}_k\mathbf{Q}_k\mathbf{J}_k^\top + \mathbf{J}_k\mathbf{P}_{k+1}^s\mathbf{J}_k^\top \tag{21}$$

These forms avoid the subtraction of covariance matrices and preserve symmetry by construction, but increase the computational cost. As shown in Appendix E, this increase is modest for the Kalman filter but becomes significant for the RTS smoother. Moreover, the symmetric Joseph form makes it straightforward to derive its square-root implementations, which further improve numerical properties at the expense of additional computational cost. In this work, square-root implementations are not studied further.

A second and more fundamental source of instability arises from the matrix inversions required by the RTS smoother. While filtering only requires the innovation covariance $\mathbf{S}_k$ to be non-singular, RTS smoothing additionally requires the predicted state covariance matrix $\mathbf{P}_{k+1}^-$ to be non-singular. This requirement may be violated in practice when PN methods operate in high-accuracy regimes where the filtered covariance matrix $\mathbf{P}_k$ becomes negligible, such that the predicted covariance matrix $\mathbf{P}_{k+1}^- = \mathbf{A}_k\mathbf{P}_k\mathbf{A}_k^\top + \mathbf{Q}_k$ is dominated by the process noise covariance $\mathbf{Q}_k$. For small time steps, this matrix may become ill-conditioned, potentially leading to numerical instabilities or breakdown of the RTS equations (Kraemer and Hennig, 2024).

3 Inference in State-Space Models through Dynamic Programming

This section formulates an optimization view of Bayesian smoothing in linear Gaussian state-space models to derive an alternative smoothing algorithm that mitigates the numerical issues of the RTS smoother. Because the smoothing posterior is Gaussian, it is fully characterized by a quadratic negative log-posterior: its minimizer gives the posterior mean trajectory, while its curvature determines the smoothing covariances. Writing smoothing as a maximum-a-posteriori (MAP) estimation problem reveals a dynamic-programming structure (Särkkä and Svensson, 2023). The Kalman filter corresponds to the forward elimination of past states, while the backward pass can be interpreted in two equivalent ways. One interpretation leads to the classical RTS smoother through backward substitution; the other leads to the MBF smoother through adjoint equations.

Although this viewpoint is well known (see e.g. Cox, 1964), it provides a unifying framework for interpreting different smoothing algorithms. In particular, it shows that the MBF variables are not merely an algebraic reformulation of RTS quantities, but arise as adjoint variables of the same optimization problem. This interpretation will be central in Section 5 where these quantities are reused for gradient-based hyperparameter learning. Full derivations are deferred to Appendix B.

3.1 Maximum-A-Posteriori Estimation

Consider the posterior distribution over the state trajectory. The corresponding MAP estimator is given by:

$$\mathbf{x}_{0:T}^{\text{MAP}} = \arg\max_{\mathbf{x}_{0:T}} p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}) \tag{22}$$
$$= \arg\max_{\mathbf{x}_{0:T}} p(\mathbf{x}_0) \prod_{k=1}^{T} p(\mathbf{y}_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \tag{23}$$

Equivalently, this can be written as the minimization of the negative log-posterior (Särkkä and Svensson, 2023):

$$\mathbf{x}_{0:T}^{\text{MAP}} = \arg\min_{\mathbf{x}_{0:T}} L(\mathbf{x}_{0:T}) \tag{24}$$

where $L(\mathbf{x}_{0:T})$ represents the loss function, defined as:

$$L(\mathbf{x}_{0:T}) = -\log\left[p(\mathbf{x}_0)\prod_{k=1}^{T} p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})\right] \tag{25}$$

$$\begin{aligned} &= -\log p(\mathbf{x}_0) - \sum_{k=1}^{T}\log p(\mathbf{y}_k|\mathbf{x}_k) \\ &\quad - \sum_{k=1}^{T}\log p(\mathbf{x}_k|\mathbf{x}_{k-1}) \end{aligned} \tag{26}$$

Under the linear Gaussian state-space model specified in Section 2, the loss function becomes:

$$\begin{aligned} L(\mathbf{x}_{0:T}) = C &+ \frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} \\ &+ \frac{1}{2}\sum_{k=1}^{\top}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k\|^2_{\mathbf{R}_k^{-1}} \\ &+ \frac{1}{2}\sum_{k=1}^{\top}\|\mathbf{x}_k - \mathbf{A}_{k-1}\mathbf{x}_{k-1}\|^2_{\mathbf{Q}_{k-1}^{-1}} \end{aligned} \tag{27}$$

where C is a constant independent of the state trajectory. This is a sparse quadratic optimization problem, where the sparsity is induced by the Markovian structure of the state-space model. As a result, the optimal trajectory and associated Gaussian marginals can be computed efficiently using dynamic programming.

3.2 Forward Dynamic Programming

Dynamic programming solves the MAP estimation problem by decomposing it into a sequence of local subproblems. In the forward pass, past states are recursively eliminated while the cost is accumulated. This procedure recovers the predicted and filtered means and covariances as determined by the Kalman filter.

For the first time step, one minimizes the following loss function with respect to $\mathbf{x}_0$ as a function of $\mathbf{x}_1$,

$$\min_{\mathbf{x}_0} \frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} + \frac{1}{2}\|\mathbf{x}_1 - \mathbf{A}_0\mathbf{x}_0\|^2_{\mathbf{Q}_0^{-1}} \tag{28}$$

The resulting minimizer expresses $\mathbf{x}_0$ conditionally as a function of $\mathbf{x}_1$, which we denote as the conditional mean $\mathbf{m}_0^c(\mathbf{x}_1)$. For subsequent steps $k = 1 \dots T-1$, a similar elimination step results in the conditional means $\mathbf{m}_k^c(\mathbf{x}_{k+1})$ by minimizing:

$$\min_{\mathbf{x}_k} \frac{1}{2}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k\|^2_{\mathbf{R}_k^{-1}} + \frac{1}{2}\|\mathbf{x}_{k+1} - \mathbf{A}_k\mathbf{x}_k\|^2_{\mathbf{Q}_k^{-1}} + \frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k^-\|^2_{[\mathbf{P}_k^-]^{-1}} \tag{29}$$

At the final time step $k = T$, the optimization problem becomes:

$$\min_{\mathbf{x}_T} \frac{1}{2}\|\mathbf{y}_T - \mathbf{H}_T\mathbf{x}_T\|^2_{\mathbf{R}_T^{-1}} + \frac{1}{2}\|\mathbf{x}_T - \mathbf{m}_T^-\|^2_{[\mathbf{P}_T^-]^{-1}} \tag{30}$$

This uniquely determines the final filtered mean and covariance, which coincide with the final smoothing mean and covariance.

As shown in Appendix B, this forward elimination procedure recovers the Kalman filtering equations. Hence, it produces the predicted and filtered means and covariances at all time steps. In addition, it provides conditional relations expressing each optimal state $\mathbf{x}_k$ in terms of the next state $\mathbf{x}_{k+1}$, denoted as $\mathbf{m}_k^c(\mathbf{x}_{k+1})$, for $k = 0 \dots T-1$. Finally, the minimal value of the loss function can also be determined and reads:

$$\min_{\mathbf{x}_{0:T}} L(\mathbf{x}_{0:T}) = C + \frac{1}{2}\sum_{k=1}^{\top} \|\mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-\|^2_{\mathbf{S}_k^{-1}} \tag{31}$$

3.3 Backward Substitution

At the end of the forward dynamic-programming pass, the final state $\mathbf{x}_T$ has been determined and each preceding state $\mathbf{x}_k$ is expressed conditionally in terms of the next one $\mathbf{x}_{k+1}$ through the conditional mean $\mathbf{m}_k^c(\mathbf{x}_{k+1})$, for $k = 0 \dots T-1$. The optimal state trajectory can hence be recovered by recursively substituting these conditional relations backwards from $k = T-1$ to $k = 0$.

As shown in Appendix B, this backward substitution procedure recovers the RTS smoothing equations. Thus, RTS smoothing can be interpreted as solving the sparse MAP problem by forward elimination followed by backward substitution.

3.4 Adjoint Equations

An alternative but equivalent backward pass can be obtained by considering sensitivities of the optimal cost with respect to the intermediate quantities produced during the forward pass. The forward pass yields both the minimal cost and the predicted and filtered distributions on which this cost depends. The derivatives of this optimal cost with respect to these intermediate quantities can hence be propagated backwards in time.

Rather than propagating the smoothed means and covariances directly, this approach propagates adjoint variables. These adjoints measure how changes in the intermediate predictive and filtering quantities affect the optimal value of the cost function. This viewpoint is closely related to adjoint methods, Lagrangian duality, and backpropagation. Since the cost function is quadratic, its gradients are linear and contain sufficient information to reconstruct the smoothed state trajectory through the optimality conditions, as demonstrated in Appendix B.

As further shown in Appendix B, this adjoint-based backward pass recovers the MBF smoothing equations, which will be detailed in Section 4. This interpretation provides a derivation that is independent of the RTS recursions and additionally establishes a direct connection between smoothing and gradient computation. This connection will be leveraged in Section 5 to optimize kernel hyperparameters.

4 Modified Bryson-Frazier Smoother

This section details the MBF smoothing equations, summarizing their algorithmic form following the qualitative and quantitative derivations in Section 3 and Appendix B, respectively.

4.1 Modified Bryson-Frazier Smoothing

The MBF smoother reformulates the backward smoothing pass in terms of adjoint variables rather than propagating the smoothed means and covariances directly. Specifically, it propagates two sets of variables, $(\boldsymbol{\lambda}_k^+, \boldsymbol{\Lambda}_k^+)$ and $(\boldsymbol{\lambda}_k^-, \boldsymbol{\Lambda}_k^-)$, in a backward pass similar to the RTS smoother. The former are associated with the filtered state at time k, while the latter are associated with the predicted state at time k. Starting from $\boldsymbol{\lambda}_T^+ = 0$ and $\boldsymbol{\Lambda}_T^+ = 0$, the backward pass proceeds for $k = T \dots 1$. First, an update step computes

$$\boldsymbol{\lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top \mathbf{S}_k^{-1}\mathbf{z}_k \tag{32}$$

$$\boldsymbol{\Lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\Lambda}_k^+ (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) + \mathbf{H}_k^\top \mathbf{S}_k^{-1}\mathbf{H}_k \tag{33}$$

which is followed by a prediction step:

$$\boldsymbol{\lambda}_{k-1}^+ = \mathbf{A}_{k-1}^\top \boldsymbol{\lambda}_k^- \tag{34}$$

$$\boldsymbol{\Lambda}_{k-1}^+ = \mathbf{A}_{k-1}^\top \boldsymbol{\Lambda}_k^- \mathbf{A}_{k-1} \tag{35}$$

The smoothed variables can then be recovered, either from $(\boldsymbol{\lambda}_k^+, \boldsymbol{\Lambda}_k^+)$, using,

$$\mathbf{m}_k^s = \mathbf{m}_k - \mathbf{P}_k\boldsymbol{\lambda}_k^+ \tag{36}$$

$$\mathbf{P}_k^s = \mathbf{P}_k - \mathbf{P}_k\boldsymbol{\Lambda}_k^+\mathbf{P}_k \tag{37}$$

or from $(\boldsymbol{\lambda}_k^-, \boldsymbol{\Lambda}_k^-)$, using,

$$\mathbf{m}_k^s = \mathbf{m}_k^- - \mathbf{P}_k^-\boldsymbol{\lambda}_k^- \tag{38}$$

$$\mathbf{P}_k^s = \mathbf{P}_k^- - \mathbf{P}_k^-\boldsymbol{\Lambda}_k^-\mathbf{P}_k^- \tag{39}$$

Both retrieval formulas result in the same smoothed means and covariances as the RTS smoother whenever the latter is well-defined. Indeed, Bierman (1973) originally derived the MBF algorithm as a reformulation of the RTS smoother in continuous time. For completeness, a self-contained derivation of the MBF smoother from the RTS recursions in discrete-time is provided in Appendix A.

4.2 Numerical Considerations

From a numerical perspective, the MBF smoother has several favorable properties in terms of numerical stability. First, the adjoint variables propagated by the MBF smoother do not involve differences of covariance matrices, which eliminates the risk of losing positive definiteness through the accumulation of round-off errors. Although covariance differences reappear in the reconstruction of the smoothed covariances, they are not themselves propagated through the backward recursion, preventing the accumulation of such errors. Moreover, these remaining subtractions can be eliminated using Householder transformations and square-root formulations (Gibbs, 2011). Second, the MBF smoother removes the additional non-singularity requirement imposed by the RTS smoother on the predicted state covariance matrix. Indeed, the smoothing pass only requires the same matrix inversions as the Kalman filter, namely those involving the innovation covariance matrices $\mathbf{S}_k$. Finally, the MBF smoother also reduces computational and memory requirements as compared to the RTS smoother, as quantified in Appendix E, which provides a detailed flop-count and memory analysis of the algorithms discussed in this paper.

5 Hyperparameter Optimization

Kernel functions in Gaussian processes typically depend on unknown parameters, often referred to as hyperparameters. A common approach to estimate these is to maximize the marginal likelihood, also known as the evidence (Williams and Rasmussen, 2006). In standard GP regression, evaluating and differentiating the marginal likelihood scales cubically in the number of data points. In the state-space formulation, however, both quantities can be computed with linear scaling in the number of time points (Särkkä and Svensson, 2023; Parellier et al., 2023). This section shows that the MBF smoother naturally provides part of the quantities required for gradient-based hyperparameter optimization. The remaining quantities can be determined through an auxiliary backward recursion.

Consider the negative log marginal likelihood (NLL):

$$J_{1:T}^{\mathrm{NLL}} = -\log p(\mathbf{y}_{1:T} \mid \theta) \tag{40}$$

where θ denotes a hyperparameter. Due to the Markovian structure, it can be expressed, up to a constant, using the prediction error decomposition as (Schweppe, 1965):

$$J_{1:T}^{\mathrm{NLL}} = \frac{1}{2}\sum_{k=1}^{T}\left(\|\mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-\|^2_{\mathbf{S}_k^{-1}} + \log\det(\mathbf{S}_k)\right) \tag{41}$$

We denote the data-fit term by:

$$J_{1:T} = \frac{1}{2}\sum_{k=1}^{T}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-\|^2_{\mathbf{S}_k^{-1}} \tag{42}$$

For regression, the innovation covariance $\mathbf{S}_k$ does not depend on the predicted or filtered means $\mathbf{m}_k^-$ and $\mathbf{m}_k$. As a result, the derivatives of the NLL with respect to these means coincide with those of $J_{1:T}$:

$$\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k^-} = \frac{\partial J_{k:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k^-} = \frac{\partial J_{k:T}}{\partial \mathbf{m}_k^-} = \frac{\partial J_{1:T}}{\partial \mathbf{m}_k^-} \tag{43}$$

$$\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k} = \frac{\partial J_{k+1:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k} = \frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k} = \frac{\partial J_{1:T}}{\partial \mathbf{m}_k} \tag{44}$$

5.1 Adjoint View on MBF Smoothing

As shown in Appendix B, the MBF smoother can be derived as an adjoint method applied to the MAP objective. Through the equivalence above, it can hence be shown that, for regression, these adjoint quantities coincide with the derivatives of the NLL with respect to the predicted and filtered means:

$$\boldsymbol{\lambda}_k^- = \frac{\partial J_{k:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k^-} \qquad \boldsymbol{\lambda}_k^+ = \frac{\partial J_{k+1:T}^{\mathrm{NLL}}}{\partial \mathbf{m}_k} \tag{45}$$

$$\boldsymbol{\Lambda}_k^- = \frac{\partial^2 J_{k:T}^{\mathrm{NLL}}}{\partial (\mathbf{m}_k^-)^2} \qquad \boldsymbol{\Lambda}_k^+ = \frac{\partial^2 J_{k+1:T}^{\mathrm{NLL}}}{\partial (\mathbf{m}_k)^2} \tag{46}$$

Thus, this establishes the result that the MBF smoother computes the first- and second-order derivatives of the NLL with respect to the intermediate mean quantities produced by the Kalman filter.

5.2 Gradients with Respect to Covariances

To obtain gradients of the NLL with respect to kernel hyperparameters, derivatives with respect to the predicted and filtered covariance matrices are also required. These derivatives are not directly provided by the MBF smoothing equations, but can be computed through an auxiliary backward recurrence. Define:

$$\mathbf{T}_k^- = \frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_k^-} \tag{47}$$

$$\mathbf{T}_k^+ = \frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_k} \tag{48}$$

and correction factors, $(\widetilde{\boldsymbol{\Lambda}}_k^-, \widetilde{\boldsymbol{\Lambda}}_k^+)$, such that the covariance sensitivities admit the following decomposition:

$$\mathbf{T}_k^- = \boldsymbol{\Lambda}_k^- + \widetilde{\boldsymbol{\Lambda}}_k^- \tag{49}$$

$$\mathbf{T}_k^+ = \boldsymbol{\Lambda}_k^+ + \widetilde{\boldsymbol{\Lambda}}_k^+ \tag{50}$$

As detailed in Appendix C, the correction terms can be computed through an auxiliary backward recurrence. Starting from $\widetilde{\boldsymbol{\Lambda}}_T^+ = \mathbf{0}$, for $k = T, \ldots, 1$, it reads:

$$\mathbf{l}_k = \frac{1}{2}\boldsymbol{\lambda}_k^+\mathbf{z}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k \tag{51}$$

$$\begin{aligned}\widetilde{\boldsymbol{\Lambda}}_k^- = {} & (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top\widetilde{\boldsymbol{\Lambda}}_k^+(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) \\ & + (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top(\mathbf{l}_k + \mathbf{l}_k^\top)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) \\ & - \frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k - \frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k\mathbf{z}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k\end{aligned} \tag{52}$$

$$\widetilde{\boldsymbol{\Lambda}}_{k-1}^+ = \mathbf{A}_{k-1}^\top\widetilde{\boldsymbol{\Lambda}}_k^-\mathbf{A}_{k-1} \tag{53}$$

Together with the MBF variables, these correction terms provide the full derivatives of the NLL with respect to the predicted and filtered covariance matrices.

5.3 Kernel Hyperparameter Optimization

Having access to derivatives with respect to both means and covariances enables efficient computation of gradients of the NLL with respect to the hyperparameter θ. Assuming the dependence on θ enters through the state-space matrices $\mathbf{A}_k$ and $\mathbf{Q}_k$, and the initial covariance $\mathbf{P}_0$, the chain rule gives,

$$\frac{\mathrm{d}J^{\mathrm{NLL}}_{1:T}}{\mathrm{d}\theta} = \sum_{k=0}^{T-1} \left(\left\langle \frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{A}_k}, \frac{\mathrm{d}\mathbf{A}_k}{\mathrm{d}\theta} \right\rangle + \left\langle \frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{Q}_k}, \frac{\mathrm{d}\mathbf{Q}_k}{\mathrm{d}\theta} \right\rangle \right) + \left\langle \frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{P}_0}, \frac{\mathrm{d}\mathbf{P}_0}{\mathrm{d}\theta} \right\rangle \tag{54}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product. Each term can be expanded further; the corresponding derivations are deferred to Appendix D. This leads to the following expression for the gradient of the NLL with respect to a hyperparameter θ:

$$\frac{\mathrm{d}J^{\mathrm{NLL}}_{1:T}}{\mathrm{d}\theta} = \sum_{k=0}^{T-1} \left\langle (\boldsymbol{\lambda}^-_{k+1}\mathbf{m}_k^\top + 2\mathbf{T}^-_{k+1}\mathbf{A}_k\mathbf{P}_k), \frac{\mathrm{d}\mathbf{A}_k}{\mathrm{d}\theta} \right\rangle + \sum_{k=0}^{T-1} \left\langle \mathbf{T}^-_{k+1}, \frac{\mathrm{d}\mathbf{Q}_k}{\mathrm{d}\theta} \right\rangle + \left\langle \mathbf{T}^+_0, \frac{\mathrm{d}\mathbf{P}_0}{\mathrm{d}\theta} \right\rangle \tag{55}$$

This expression enables gradient-based hyperparameter optimization using quantities obtained from the Kalman filter, the MBF smoother, and the additional covariance-correction recursion.

6 Numerical Experiments

We demonstrate the MBF smoother through three numerical experiments that highlight its correctness, practical usefulness, and robustness. First, we show that GP regression can be performed using Kalman filtering and MBF smoothing, yielding results identical to conventional RTS smoothing in a well-conditioned setting. Second, we illustrate how the intermediate quantities propagated by the MBF smoother enable efficient kernel hyperparameter optimization. Finally, we show that MBF smoothing remains applicable in singular state-space models where RTS smoothing breaks down. The code for the numerical experiments is available online.[1]

6.1 Experimental Set-Up

We generate a latent function by drawing from a Gaussian process with a Matérn kernel with smoothness parameter $\nu = 3/2$,

$$k(t, t') = \sigma_f^2(1 + \lambda|t - t'|)\exp(-\lambda|t - t'|) \tag{56}$$

where $\sigma_f = 1$ is the output scale and $\lambda = 1$, corresponding to the length scale $\ell = \sqrt{3}/\lambda = \sqrt{3}$ (Särkkä and Solin, 2019). In total, 5 000 data points are generated over a time-interval of 200 seconds, of which 200 data points are uniformly selected as training data.

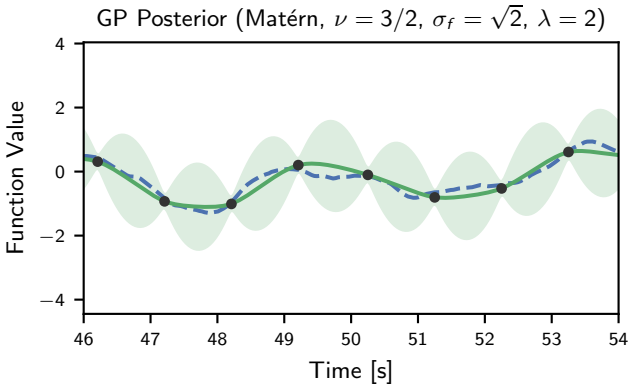

Figure 1: Segment of the latent function (dashed blue), observed at uniformly sampled time instances (black). The GP posterior with mean (solid green) and 2σ-interval (light green) are shown for a Matérn-3/2 kernel with misspecified hyperparameters, resulting in overly large posterior uncertainty.

Following Hartikainen and Särkkä (2010); Jordán et al. (2021), this GP can be represented in discrete-time as a linear Gaussian state-space model:

$$\Delta t_k := t_{k+1} - t_k \tag{57}$$
$$\phi_k := \exp(-\lambda \Delta t_k) \tag{58}$$
$$\mathbf{A}_k = \phi_k \begin{bmatrix} 1 + \lambda\Delta t_k & \Delta t_k \\ -\lambda^2 \Delta t_k & 1 - \lambda \Delta t_k \end{bmatrix} \tag{59}$$
$$[\mathbf{Q}_k]_{11} = \sigma_f^2(1 - \phi_k^2(1 + 2\lambda\Delta t_k + 2(\lambda\Delta t_k)^2)) \tag{60}$$
$$[\mathbf{Q}_k]_{12} = [\mathbf{Q}_k]_{21} = 2\sigma_f^2 \Delta t_k^2 \lambda^3 \phi_k^2 \tag{61}$$
$$[\mathbf{Q}_k]_{22} = \sigma_f^2 \lambda^2 (1 - \phi_k^2(1 - 2\lambda\Delta t_k + 2(\lambda\Delta t_k)^2)) \tag{62}$$
$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \qquad \mathbf{R}_k = \sigma_n^2 = 10^{-2} \tag{63}$$

The initial distribution is given by

$$\mathbf{x}_0 \sim \mathcal{N}\left(0,\ \sigma_f^2 \begin{bmatrix} 1 & 0 \\ 0 & \lambda^2 \end{bmatrix}\right) \tag{64}$$

where $\mathbf{P}_0$ is the stationary prior covariance. We use this set-up to study the MBF smoother in three settings.

6.2 Gaussian Process Regression

Fig. 1 shows the GP posterior obtained using a Matérn-3/2 kernel with deliberately misspecified hyperparameters, namely $\sigma_f = \sqrt{2}$ and $\lambda = 2$. These differ from the values used to generate the latent function and therefore lead to an intentionally misspecified posterior. In this well-conditioned setting, all state-space covariance matrices remain regular, so both the RTS and MBF smoothers are applicable. As expected, the MBF-smoothed and RTS-smoothed means coincide up to machine precision and match the GP posterior computed through a Cholesky factorization of the full covariance matrix (Williams and Rasmussen, 2006). This experiment verifies that the MBF smoother is an exact drop-in replacement for RTS smoothing when the state-space model is non-singular.

[1] github.com/TomColemont/ProbNum26-MBF-Smoothing

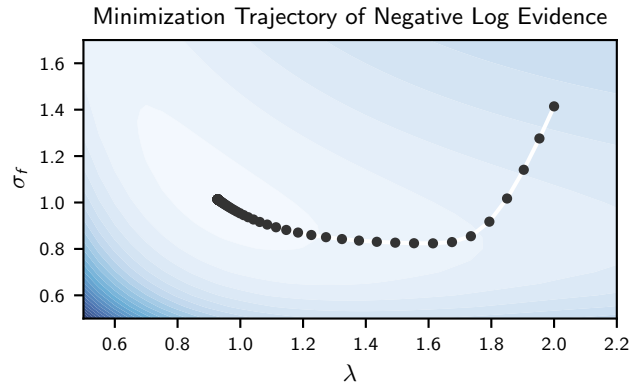

Figure 2: Minimization trajectory of the negative log evidence as a function of the kernel hyperparameters σ_f and λ. The gradients are computed directly from MBF intermediate quantities. Lighter colors indicate lower values of the cost function. Gradient descent converges to $\hat{\sigma}_f = 1.04$ and $\hat{\lambda} = 0.93$, close to the true hyperparameters.

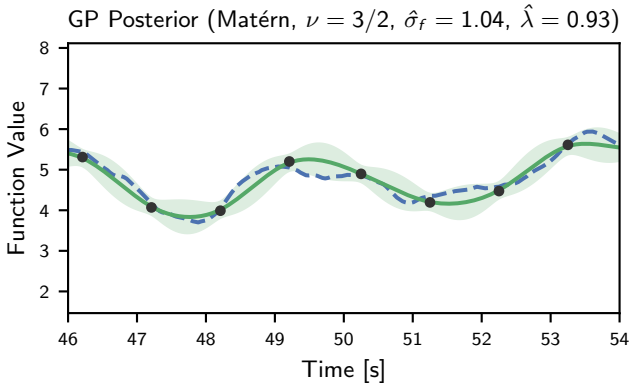

Figure 3: GP posterior with mean (solid green) and 2σ (light green) after augmenting the state-space model with a deterministic bias term, making both the state and process noise covariances singular. The MBF smoother remains stable and matches the corresponding GP posterior, while the RTS smoother fails due to the singular predicted covariance matrix.

6.3 Hyperparameter Optimization

As described in Section 5, the quantities propagated by the MBF smoother can be reused to compute the gradients of the negative log marginal likelihood, or equivalently negative log evidence, with respect to kernel hyperparameters. Figure 2 illustrates the minimization trajectory in the two-dimensional hyperparameter space (σ_f, λ) when minimizing the negative log evidence using gradient descent with a learning rate of $1.5 \cdot 10^{-3}$. Other first-order optimizers could be used as well. Lighter colors indicate lower values of the objective function. The resulting hyperparameter estimates are $\hat{\sigma}_f = 1.04$ and $\hat{\lambda} = 0.93$, which closely match the true hyperparameters of the generative model. This experiment demonstrates that the MBF smoother provides the quantities needed for gradient-based kernel hyperparameter optimization.

6.4 Singular State-Space Models

Finally, we consider a state-space model in which both the state and process noise covariance matrices become singular. Such situations arise naturally when augmenting the state vector with deterministic components, such as a constant bias term. We add a bias of 5 units to the latent function, and include this as a deterministic bias in the state-space model. This introduces zeros in the final row and column of both $\mathbf{P}_k$ and $\mathbf{Q}_k$. Figure 3 shows the resulting GP posterior when the estimated hyperparameters from Section 6.3 are used.

In this singular setting, the RTS smoother is no longer applicable because it requires inversion of the predicted state covariance matrix. In contrast, the MBF smoother remains well-defined and produces a smoothed trajectory that agrees with the GP posterior up to machine precision. This illustrates the MBF smoother's robustness in regimes where the RTS smoother breaks down.

7 Discussion

The results of this work demonstrate that the modified Bryson-Frazier (MBF) smoother provides a principled alternative to the Rauch-Tung-Striebel (RTS) smoother for Gaussian process regression in its state-space representation. In well-conditioned settings, MBF smoothing is fully equivalent to RTS smoothing and yields the same posterior distributions as standard Gaussian process regression. Its main advantage appears in numerically challenging regimes: by avoiding inversion of the predicted state covariance matrix, MBF smoothing remains applicable when this matrix becomes ill-conditioned or singular. This is particularly relevant in high-accuracy regimes, where covariance matrices may become nearly singular, and in constrained or augmented state-space models, as illustrated in Section 6.4.

Beyond numerical robustness, MBF smoothing also offers computational advantages. Although both RTS and MBF smoothing are cubic in the state dimension, the latter has lower leading-order computational cost and reduced memory requirements, as quantified in Appendix E. Moreover, unlike the RTS smoother, it does not propagate covariance differences backwards, avoiding round-off error accumulation that can lead to loss of positive definiteness. The remaining covariance subtractions during the reconstruction can be avoided using square-root implementations (Gibbs, 2011).

A useful consequence of the MBF smoothing formulation is that smoothed means and covariances need not be reconstructed at every step of the backward pass. Instead, the algorithm propagates adjoint variables, from which smoothed quantities can be recovered only when needed. This is especially relevant in probabilistic ODE solvers, where small internal time steps may be required in stiff regions even when the solution is only requested at a coarser set of output times.

The adjoint interpretation also connects MBF smoothing to kernel hyperparameter optimization. The MBF backward pass naturally provides derivatives of the negative log marginal likelihood with respect to the predicted and filtered means. Together with an additional covariance-correction recursion, this yields the sensitivities required for hyperparameter optimization, allowing smoothing and hyperparameter learning to be treated within a single state-space framework.

This work has focused on one-dimensional Gaussian process regression with stationary, integrable kernels, for which exact or arbitrarily accurate state-space representations are available. Natural extensions include spatio-temporal Gaussian processes, where larger state dimensions may make the computational advantages of MBF smoothing more pronounced (Särkkä et al., 2013), and probabilistic ODE solvers. While MBF smoothing applies directly to the latter class of solvers, MBF-based hyperparameter optimization requires further adaptation for EK1-based solvers, where the innovation covariance depends on the predicted mean. Combining this with iterated extended smoothing schemes is especially interesting, as these methods naturally involve multiple forward-backward passes necessary for gradient-based hyperparameter optimization.

8 Conclusion

We have revisited the modified Bryson-Frazier smoother as a numerically robust and computationally efficient alternative to RTS smoothing for Gaussian process regression in its state-space representation. The MBF smoother computes the same posterior distributions as the RTS smoother whenever both are well-defined, while avoiding the inversion of predicted state covariance matrices that can make RTS smoothing unstable or infeasible. Furthermore, the MBF backward variables admit an adjoint interpretation, allowing them to be reused for gradient-based kernel hyperparameter optimization with minimal additional cost.

Overall, the MBF smoother provides a unified route to stable smoothing and hyperparameter learning in Gaussian processes in their state-space representation, making it a useful building block for probabilistic numerical methods based on Bayesian filtering and smoothing.

Acknowledgments

This research was partially supported by the Research Foundation – Flanders (FWO; Grant No. I000725N, I002123N, and 1178926N).

References

G. J. Bierman. Fixed interval smoothing with discrete measurements. *International Journal of Control*, 18 (1):65–75, 1973.

H. Cox. On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on automatic control*, 9(1):5–12, 1964.

R. G. Gibbs. Square root modified Bryson–Frazier smoother. *IEEE Transactions on Automatic Control*, 56(2):452–456, 2011.

J. Hartikainen and S. Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384, 2010.

A. Jordán, S. Eyheramendy, and J. Buchner. State-space representation of Matérn and damped simple harmonic oscillator Gaussian processes. *arXiv preprint arXiv:2109.10685*, 2021.

R. E. Kalman. A new approach to linear filtering and prediction problems. 1960.

R. Kohn and C. F. Ansley. A fast algorithm for signal extraction, influence and cross-validation in state space models. *Biometrika*, pages 65–79, 1989.

N. Kraemer and P. Hennig. Stable implementation of probabilistic ODE solvers. *Journal of Machine Learning Research*, 25(111):1–29, 2024.

J. Loper, D. Blei, J. P. Cunningham, and L. Paninski. A general linear-time inference method for Gaussian processes on one dimension. *Journal of Machine Learning Research*, 22(234):1–36, 2021.

C. Parellier, A. Barrau, and S. Bonnabel. Speeding-up backpropagation of gradients through the Kalman filter via closed-form expressions. *IEEE Transactions on Automatic Control*, 68(12):8171–8177, 2023.

M. Pförtner, J. Wenger, J. Cockayne, and P. Hennig. Computation-aware Kalman filtering and smoothing. In *International Conference on Artificial Intelligence and Statistics*, pages 2071–2079. PMLR, 2025.

H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model predictive control: theory, computation, and design*. Nob Hill Publishing, Madison, 2nd ed. edition, 2017. ISBN 9780975937730.

S. Särkkä and A. Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

S. Särkkä and L. Svensson. *Bayesian filtering and smoothing*, volume 17. Cambridge university press, 2023.

S. Särkkä, A. Solin, and J. Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

F. Schweppe. Evaluation of likelihood functions for Gaussian signals. *IEEE transactions on Information Theory*, 11(1):61–70, 1965.

F. Tronarp, H. Kersting, S. Särkkä, and P. Hennig. Probabilistic solutions to ordinary differential equations as nonlinear Bayesian filtering: a new perspective. *Statistics and Computing*, 29(6):1297–1315, 2019.

C. K. Williams and C. E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT Press Cambridge, MA, 2006.

A MBF-RTS Smoother Equivalence

Bierman (1973) originally derived the MBF smoother from the RTS smoother recursions by relying on the underlying continuous-time differential equations. In contrast, this section provides a self-contained derivation of the discrete-time MBF smoother. We start by defining the MBF variables explicitly in terms of the predicted, filtered and smoothed quantities:

$$\boldsymbol{\lambda}_k^+ = \mathbf{P}_k^{-1}(\mathbf{m}_k - \mathbf{m}_k^s) \tag{65}$$
$$\boldsymbol{\lambda}_k^- = [\mathbf{P}_k^-]^{-1}(\mathbf{m}_k^- - \mathbf{m}_k^s) \tag{66}$$
$$\boldsymbol{\Lambda}_k^+ = \mathbf{P}_k^{-1}(\mathbf{P}_k - \mathbf{P}_k^s)\mathbf{P}_k^{-1} \tag{67}$$
$$\boldsymbol{\Lambda}_k^- = [\mathbf{P}_k^-]^{-1}(\mathbf{P}_k^- - \mathbf{P}_k^s)[\mathbf{P}_k^-]^{-1} \tag{68}$$

These definitions assume that the predicted and filtered state covariance matrices are non-singular. An alternative derivation that derives the MBF smoother without this assumption is provided by Gibbs (2011).

A.1 Derivation: Mean Recursions

After rearranging, the RTS smoothing equation for the mean can be expressed as:

$$\mathbf{m}_k^s - \mathbf{m}_k = \mathbf{J}_k(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \tag{69}$$
$$= \mathbf{P}_k\mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1}(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \tag{70}$$

Left-multiplication by $\mathbf{P}_k^{-1}$ yields:

$$\mathbf{P}_k^{-1}(\mathbf{m}_k^s - \mathbf{m}_k) = \mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1}(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \tag{71}$$
$$\boldsymbol{\lambda}_k^+ = \mathbf{A}_k^\top\boldsymbol{\lambda}_{k+1}^- \tag{72}$$

which recovers the MBF mean-prediction equation. To derive the update equation, we relate $\boldsymbol{\lambda}_k^+$ and $\boldsymbol{\lambda}_k^-$ as:

$$\mathbf{P}_k\boldsymbol{\lambda}_k^+ - \mathbf{P}_k^-\boldsymbol{\lambda}_k^- = \mathbf{m}_k - \mathbf{m}_k^- \tag{73}$$
$$= \mathbf{K}_k\mathbf{z}_k \tag{74}$$
$$= \mathbf{P}_k^-\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k \tag{75}$$

Left-multiplication by $[\mathbf{P}_k^-]^{-1}$ yields after rearranging:

$$\boldsymbol{\lambda}_k^- = [\mathbf{P}_k^-]^{-1}\mathbf{P}_k\boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k \tag{76}$$

Using the symmetry of covariance matrices, the first term can be re-expressed as:

$$[\mathbf{P}_k^-]^{-1}\mathbf{P}_k = [\mathbf{P}_k^-]^{-1}\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \tag{77}$$
$$= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \tag{78}$$

which recovers the MBF mean-update equation:

$$\boldsymbol{\lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top\boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k \tag{79}$$

A.2 Derivation: Covariance Recursions

After rearranging, the RTS smoothing equation for the covariance can be expressed as:

$$\mathbf{P}_k^s - \mathbf{P}_k = \mathbf{J}_k(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\mathbf{J}_k^\top \tag{80}$$
$$= \mathbf{P}_k\mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1}(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)[\mathbf{P}_{k+1}^-]^{-1}\mathbf{A}_k\mathbf{P}_k \tag{81}$$

Left- and right-multiplication by $\mathbf{P}_k^{-1}$ recovers the MBF covariance-prediction equation:

$$\boldsymbol{\Lambda}_k^+ = \mathbf{A}_k^\top\boldsymbol{\Lambda}_{k+1}^-\mathbf{A}_k \tag{82}$$

For the update equation, we relate $\boldsymbol{\Lambda}_k^+$ and $\boldsymbol{\Lambda}_k^-$ as:

$$\mathbf{P}_k\boldsymbol{\Lambda}_k^+\mathbf{P}_k - \mathbf{P}_k^-\boldsymbol{\Lambda}_k^-\mathbf{P}_k^- = \mathbf{P}_k - \mathbf{P}_k^- \tag{83}$$
$$= -\mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^- \tag{84}$$

Left- and right-multiplication by $[\mathbf{P}_k^-]^{-1}$ leads to:

$$[\mathbf{P}_k^-]^{-1}\mathbf{P}_k\boldsymbol{\Lambda}_k^+\mathbf{P}_k[\mathbf{P}_k^-]^{-1} - \boldsymbol{\Lambda}_k^- = -[\mathbf{P}_k^-]^{-1}\mathbf{K}_k\mathbf{H}_k \tag{85}$$
$$= -\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k \tag{86}$$

Using the identity in Eq. (78) recovers the MBF covariance-update equation.

$$\boldsymbol{\Lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top\boldsymbol{\Lambda}_k^+(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) + \mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k \tag{87}$$

B Unified Derivation: Kalman Filter, RTS and MBF Smoother

This Appendix subsequently derives the Kalman filter, RTS and MBF smoother through a unified dynamic programming framework.

B.1 Kalman Filtering: Forward Dynamic Programming

We aim to minimize the cost function in Eq. (27) through forward dynamic programming. Starting from the initial costs:

$$V_0^\star(\mathbf{x}_0) = \frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} \tag{88}$$
$$V_1^\star(\mathbf{x}_1) = \min_{\mathbf{x}_0}\{V_0^\star(\mathbf{x}_0) + \frac{1}{2}\|\mathbf{x}_1 - \mathbf{A}_0\mathbf{x}_0\|^2_{\mathbf{Q}_0^{-1}}\} \tag{89}$$

We can successively minimize the cost by considering the following problem for $k = 1 \ldots T-1$:

$$V_{k+1}^\star(\mathbf{x}_{k+1}) = \min_{\mathbf{x}_k}\{V_k^\star(\mathbf{x}_k) + \frac{1}{2}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k\|^2_{\mathbf{R}_k^{-1}} + \frac{1}{2}\|\mathbf{x}_{k+1} - \mathbf{A}_k\mathbf{x}_k\|^2_{\mathbf{Q}_k^{-1}}\} \tag{90}$$

To end, for $k = T$, the recursion becomes:

$$V_{T+1}^\star = \min_{\mathbf{x}_T}\{V_T^\star(\mathbf{x}_T) + \frac{1}{2}\|\mathbf{y}_T - \mathbf{H}_T\mathbf{x}_T\|^2_{\mathbf{R}_T^{-1}}\} \tag{91}$$

At the end of the forward pass, we both have the exact minimizer $\mathbf{x}_T^\star$ as well as the minimal cost $V_{T+1}^\star$. We will show that this forward dynamic programming perspective recovers the well-known Kalman filtering equations. To this end, we will exhaustively rely upon the following well-known identity for the sum of two quadratic functions.

Lemma B.1. *Let $\mathbf{x}, \bar{\mathbf{x}} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, and $\boldsymbol{\Gamma} \in \mathbb{R}^{m\times n}$. Suppose that $\mathbf{T}_1 \in \mathbb{R}^{n\times n}$ and $\mathbf{T}_2 \in \mathbb{R}^{m\times m}$ are symmetric positive definite matrices and define the function*

$$q(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \bar{\mathbf{x}}\|^2_{\mathbf{T}_1^{-1}} + \frac{1}{2}\|\boldsymbol{\Gamma}\mathbf{x} - \mathbf{b}\|^2_{\mathbf{T}_2^{-1}}. \tag{92}$$

Then, the following assertions hold.

1. $q(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^\star\|^2_{\mathbf{M}^{-1}} + \frac{1}{2}\|\mathbf{\Gamma}\bar{\mathbf{x}} - \mathbf{b}\|^2_{\mathbf{S}^{-1}}$, *where*

$$\mathbf{S} = \mathbf{\Gamma}\mathbf{T}_1\mathbf{\Gamma}^\top + \mathbf{T}_2, \quad \mathbf{L} = \mathbf{T}_1\mathbf{\Gamma}^\top\mathbf{S}^{-1},$$
$$\mathbf{M} = (\mathbf{I} - \mathbf{L}\mathbf{\Gamma})\mathbf{T}_1, \quad \mathbf{x}^\star = \bar{\mathbf{x}} + \mathbf{L}(\mathbf{b} - \mathbf{\Gamma}\bar{\mathbf{x}}).$$

2. $\mathbf{x}^\star = \arg\min_{\mathbf{x}\in\mathbb{R}^n} q(\mathbf{x})$ *and* $q(\mathbf{x}^\star) = \frac{1}{2}\|\mathbf{\Gamma}\bar{\mathbf{x}} - \mathbf{b}\|^2_{\mathbf{S}^{-1}}$.

Proof. Follows directly from the optimality conditions and the Woodbury matrix identity, see e.g. (Rawlings et al., 2017, Example 1.1). □

First, we will show for all $k = 1 \dots T$ that:

$$V_k^\star(\mathbf{x}_k) = \frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k^-\|^2_{[\mathbf{P}_k^-]^{-1}} + \underbrace{\frac{1}{2}\sum_{i=1}^{k-1}\|\mathbf{y}_i - \mathbf{H}_i\mathbf{m}_i^-\|^2_{\mathbf{S}_i^{-1}}}_{\equiv J_{1:k-1}} + \underbrace{\min_{\mathbf{x}_{k-1}}\left\{\frac{1}{2}\|\mathbf{x}_{k-1} - \mathbf{m}_{k-1}^c(\mathbf{x}_k)\|^2_{[\mathbf{P}_{k-1}^c]^{-1}}\right\}}_{=0} \tag{93}$$

up to constants, where:

$$\mathbf{m}_k^- = \mathbf{A}_{k-1}\mathbf{m}_{k-1} \tag{94}$$
$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^\top + \mathbf{Q}_{k-1} \tag{95}$$
$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k \tag{96}$$
$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top\mathbf{S}_k^{-1} \tag{97}$$
$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-) \tag{98}$$
$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^- \tag{99}$$
$$\mathbf{J}_k = \mathbf{P}_k\mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1} \tag{100}$$
$$\mathbf{m}_k^c(\mathbf{x}_{k+1}) = \mathbf{m}_k + \mathbf{J}_k(\mathbf{x}_{k+1} - \mathbf{m}_{k+1}^-) \tag{101}$$
$$\mathbf{P}_k^c = (\mathbf{I} - \mathbf{J}_k\mathbf{A}_k)\mathbf{P}_k = \mathbf{P}_k - \mathbf{J}_k\mathbf{P}_{k+1}^-\mathbf{J}_k^\top \tag{102}$$

where $\mathbf{m}_k^-$, $\mathbf{m}_k$ and $\mathbf{m}_k^c$ denote the predicted, filtered and conditional smoothed means, respectively. Consider the case where $k = 1$, for which:

$$V_1^\star(\mathbf{x}_1) = \min_{\mathbf{x}_0}\left\{\frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} + \frac{1}{2}\|\mathbf{x}_1 - \mathbf{A}_0\mathbf{x}_0\|^2_{\mathbf{Q}_0^{-1}}\right\} \tag{103}$$

Then, it follows directly from Lemma B.1 that:

$$V_1^\star(\mathbf{x}_1) = \frac{1}{2}\|\mathbf{x}_1 - \mathbf{m}_1^-\|^2_{[\mathbf{P}_1^-]^{-1}} + \min_{\mathbf{x}_0}\left\{\frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0^c(\mathbf{x}_1)\|^2_{[\mathbf{P}_0^c]^{-1}}\right\} \tag{104}$$

which establishes the claim for $k = 1$. Now, suppose the claim holds for some $k = 1, \dots, T-1$. Then, it follows from Eq. (90) that:

$$V_{k+1}^\star(\mathbf{x}_{k+1}) = J_{1:k-1} + \min_{\mathbf{x}_k}\{\frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k^-\|^2_{[\mathbf{P}_k^-]^{-1}} + \frac{1}{2}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k\|^2_{\mathbf{R}_k^{-1}} + \frac{1}{2}\|\mathbf{x}_{k+1} - \mathbf{A}_k\mathbf{x}_k\|^2_{\mathbf{Q}_k^{-1}}\} \tag{105}$$

Applying Lemma B.1 to the first two norms yields:

$$V_{k+1}^\star(\mathbf{x}_{k+1}) = \underbrace{J_{1:k-1} + \frac{1}{2}\|\mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-\|^2_{\mathbf{S}_k^{-1}}}_{\equiv J_{1:k}} + \min_{\mathbf{x}_k}\{\frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k\|^2_{\mathbf{P}_k^{-1}} + \frac{1}{2}\|\mathbf{x}_{k+1} - \mathbf{A}_k\mathbf{x}_k\|^2_{\mathbf{Q}_k^{-1}}\} \tag{106}$$

Hence, applying Lemma B.1, it follows that:

$$V_{k+1}^\star(\mathbf{x}_{k+1}) = \frac{1}{2}\|\mathbf{x}_{k+1} - \mathbf{m}_{k+1}^-\|^2_{[\mathbf{P}_{k+1}^-]^{-1}} + J_{1:k} + \min_{\mathbf{x}_k}\left\{\frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k^c(\mathbf{x}_{k+1})\|^2_{[\mathbf{P}_k^c]^{-1}}\right\} \tag{107}$$

Consequently, by induction, we have established that Eq. (93) holds for $k = 1 \dots T-1$. Consider the case where $k = T$, for which:

$$V_{T+1}^\star = J_{1:T-1} + \min_{\mathbf{x}_T}\{\frac{1}{2}\|\mathbf{x}_T - \mathbf{m}_T^-\|^2_{[\mathbf{P}_T^-]^{-1}} + \frac{1}{2}\|\mathbf{y}_T - \mathbf{H}_T\mathbf{x}_T\|^2_{\mathbf{R}_T^{-1}} \tag{108}$$

Then, applying Lemma B.1 leads to:

$$V_{T+1}^\star = J_{1:T-1} + \frac{1}{2}\|\mathbf{y}_T - \mathbf{H}_T\mathbf{m}_T^-\|^2_{\mathbf{S}_T^{-1}} + \min_{\mathbf{x}_T}\{\frac{1}{2}\|\mathbf{x}_T - \mathbf{m}_T\|^2_{\mathbf{P}_T^{-1}}\} = J_{1:T} \tag{109}$$

It is hence clear that the obtained minimal cost reads:

$$L_{\min} \equiv \min_{\mathbf{x}_{0:T}} L(\mathbf{x}_{0:T}) = V_{T+1}^\star + C = J_{1:T} + C \tag{110}$$

B.2 RTS Smoothing: Backsubstitution

The forward dynamic programming derivation provides not only the minimal cost but also the first and second moments of the predicted, filtered states and conditional smoothed states. This structure naturally induces a backward recursion that will recover the RTS smoother. Indeed, it has been shown in a recursive manner that

$$L(\mathbf{x}_{0:T}) = C + J_{1:T} + \frac{1}{2}\|\mathbf{x}_T - \mathbf{m}_T\|^2_{\mathbf{P}_T^{-1}} + \frac{1}{2}\sum_{k=0}^{T-1}\|\mathbf{x}_k - \mathbf{m}_k^c(\mathbf{x}_{k+1})\|^2_{[\mathbf{P}_k^c]^{-1}}$$

This decomposition shows that the posterior consists of a terminal quadratic term and a series of conditional quadratic terms coupling $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$. Interpreting L as a negative log-posterior, this directly corresponds to

$$p(\mathbf{x}_T \mid \mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{m}_T, \mathbf{P}_T) \tag{111}$$
$$p(\mathbf{x}_k \mid \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{m}_k^c(\mathbf{x}_{k+1}), \mathbf{P}_k^c) \tag{112}$$

Starting from $p(\mathbf{x}_T \mid y_{1:T})$, the marginals follow via the backward recursion

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k+1}, \mathbf{y}_{1:T})p(\mathbf{x}_{k+1} \mid \mathbf{y}_{1:T})\,\mathrm{d}\mathbf{x}_{k+1}, \tag{113}$$

which, using Gaussian identities, leads to

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{m}_k^s, \mathbf{P}_k^s), \quad \forall k = 0 \dots T-1,$$

where

$$\mathbf{m}_k^s = \mathbf{m}_k + \mathbf{J}_k(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \tag{114}$$
$$\mathbf{P}_k^s = \mathbf{P}_k^c + \mathbf{J}_k \mathbf{P}_{k+1}^s \mathbf{J}_k^\top \tag{115}$$
$$= \mathbf{P}_k + \mathbf{J}_k(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\mathbf{J}_k^\top \tag{116}$$

which recover the RTS smoothing equations.

B.3 MBF Smoothing: Adjoint Equations

As shown in Appendix A, the MBF smoother can be derived directly from the RTS smoother. Here, we consider an alternative derivation that does not start from the RTS smoothing equations. Instead, we interpret the MBF variables as adjoints of the dynamic programming objective introduced earlier.

After the forward dynamic programming pass, the minimum value of the cost function can be determined as:

$$L_{\text{min}} = \frac{1}{2}\sum_{i=1}^{\top} \|\mathbf{y}_i - \mathbf{H}_i \mathbf{m}_i^-\|^2_{\mathbf{S}_i^{-1}} + C \tag{117}$$

Ignoring the constant C, define the corresponding cost-to-go as:

$$J_{k:T} \equiv \frac{1}{2}\sum_{i=k}^{\top} \|\mathbf{y}_i - \mathbf{H}_i \mathbf{m}_i^-\|^2_{\mathbf{S}_i^{-1}} = J_{1:T} - J_{1:k-1} \tag{118}$$

Define the adjoint variables as the first derivatives of this cost with respect to the predicted and filtered means:

$$\boldsymbol{\lambda}_k^- = \frac{\partial J_{k:T}}{\partial \mathbf{m}_k^-} \tag{119}$$
$$\boldsymbol{\lambda}_k^+ = \frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k} \tag{120}$$

These definitions reflect that $\mathbf{m}_k^-$ affects the current and future terms, whereas $\mathbf{m}_k$ only affects future terms.

We first focus on the mean-prediction equation. Assuming $\boldsymbol{\lambda}_{k+1}^+$ and $\boldsymbol{\lambda}_{k+1}^-$ are known, then $\boldsymbol{\lambda}_k^+$ can be determined through the chain rule:

$$\underbrace{\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k}}_{=\boldsymbol{\lambda}_k^+} = \underbrace{\left(\frac{\partial \mathbf{m}_{k+1}^-}{\partial \mathbf{m}_k}\right)^\top}_{=\mathbf{A}_k^\top} \underbrace{\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_{k+1}^-}}_{=\boldsymbol{\lambda}_{k+1}^-} \tag{121}$$

which recovers the MBF mean-prediction equation. We now turn to the mean-update equation. To determine $\boldsymbol{\lambda}_k^-$, consider:

$$\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k^-} = \underbrace{\left(\frac{\partial \mathbf{m}_k}{\partial \mathbf{m}_k^-}\right)^\top}_{=(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^\top} \underbrace{\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k}}_{=\boldsymbol{\lambda}_k^+} \tag{122}$$

where the left-hand side can be expressed in terms of $\boldsymbol{\lambda}_k^-$ by rewriting the cost as:

$$J_{k+1:T} = J_{k:T} - \frac{1}{2}\|\mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^-\|^2_{\mathbf{S}_k^{-1}} \tag{123}$$

which after taking the first derivative results in:

$$\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k^-} = \underbrace{\frac{\partial J_{k:T}}{\partial \mathbf{m}_k^-}}_{=\boldsymbol{\lambda}_k^-} + \mathbf{H}_k^\top \mathbf{S}_k^{-1} \underbrace{(\mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^-)}_{=\mathbf{z}_k} \tag{124}$$

Combining both expressions recovers the MBF mean-update equation:

$$\boldsymbol{\lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top \mathbf{S}_k^{-1} \mathbf{z}_k \tag{125}$$

Together with the MBF mean-prediction equation, the first-order adjoint equations are therefore:

$$\boldsymbol{\lambda}_k^+ = \mathbf{A}_k^\top \boldsymbol{\lambda}_{k+1}^- \tag{126}$$
$$\boldsymbol{\lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top \mathbf{S}_k^{-1}(\mathbf{y}_k - \mathbf{H}_k \mathbf{m}_k^-) \tag{127}$$

which, by definition of $J_{k:T}$, is initialized from:

$$\boldsymbol{\lambda}_T^+ = 0 \tag{128}$$

The same argument applies to the second derivatives. Define the adjoint variables as:

$$\boldsymbol{\Lambda}_k^- = \frac{\partial^2 J_{k:T}}{\partial (\mathbf{m}_k^-)^2} \tag{129}$$
$$\boldsymbol{\Lambda}_k^+ = \frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_k)^2} \tag{130}$$

We first focus on the covariance-prediction equation. Again, assuming $\boldsymbol{\Lambda}_{k+1}^+$ and $\boldsymbol{\Lambda}_{k+1}^-$ are known, then, $\boldsymbol{\Lambda}_k^+$ can be determined through the chain rule:

$$\underbrace{\frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_k)^2}}_{=\boldsymbol{\Lambda}_k^+} = \underbrace{\left(\frac{\partial \mathbf{m}_{k+1}^-}{\partial \mathbf{m}_k}\right)^\top}_{=\mathbf{A}_k^\top} \underbrace{\frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_{k+1}^-)^2}}_{=\boldsymbol{\Lambda}_{k+1}^-} \underbrace{\left(\frac{\partial \mathbf{m}_{k+1}^-}{\partial \mathbf{m}_k}\right)}_{=\mathbf{A}_k} \tag{131}$$

which recovers the MBF covariance-prediction equation. Focusing on the covariance-update equation to determine $\boldsymbol{\Lambda}_k^-$, consider:

$$\frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_k^-)^2} = \underbrace{\left(\frac{\partial \mathbf{m}_k}{\partial \mathbf{m}_k^-}\right)^\top}_{=(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^\top} \underbrace{\left(\frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_k)^2}\right)}_{=\boldsymbol{\Lambda}_k^+} \underbrace{\left(\frac{\partial \mathbf{m}_k}{\partial \mathbf{m}_k^-}\right)}_{=(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)} \tag{132}$$

where the left-hand side can be expressed in terms of $\boldsymbol{\Lambda}_k^-$ in a similar manner as above using Eq. (123):

$$\frac{\partial^2 J_{k+1:T}}{\partial (\mathbf{m}_k^-)^2} = \boldsymbol{\Lambda}_k^- - \mathbf{H}_k^\top \mathbf{S}_k^{-1} \mathbf{H}_k \tag{133}$$

Combining both expressions recovers the MBF covariance-update equation:

$$\boldsymbol{\Lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\Lambda}_k^+ (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) + \mathbf{H}_k^\top \mathbf{S}_k^{-1} \mathbf{H}_k \tag{134}$$

Together with the MBF covariance-prediction equation, the second-order adjoint recursions are therefore:

$$\boldsymbol{\Lambda}_k^+ = \mathbf{A}_k^\top \boldsymbol{\Lambda}_{k+1}^- \mathbf{A}_k \tag{135}$$
$$\boldsymbol{\Lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top \boldsymbol{\Lambda}_k^+ (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) + \mathbf{H}_k^\top \mathbf{S}_k^{-1} \mathbf{H}_k \tag{136}$$

which, by definition of $J_{k:T}$, is initialized from:

$$\boldsymbol{\Lambda}_T^+ = 0 \tag{137}$$

B.4 MBF Smoothing: Recovery of MAP Trajectory

It remains to show how the smoothed means and covariances can be recovered from the adjoint variables. After eliminating the states up to time k, the remaining cost can be written in the form of Eq. (106) as:

$$\widetilde{L}_{k:T}(\mathbf{x}_k) = J_{1:k} + \frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k\|^2_{\mathbf{P}_k^{-1}} + f_{k+1:T}(\mathbf{x}_k) \tag{138}$$

where $f_{k+1:T}(\mathbf{x}_k)$ collects all future cost terms expressed as a function of $\mathbf{x}_k$. Define the cost function as:

$$L_{k:T}(\mathbf{x}_k) = \frac{1}{2}\|\mathbf{x}_k - \mathbf{m}_k\|^2_{\mathbf{P}_k^{-1}} + f_{k+1:T}(\mathbf{x}_k) \tag{139}$$

Then it follows that:

$$J_{k+1:T} = \min_{\mathbf{x}_k} L_{k:T}(\mathbf{x}_k) \tag{140}$$

of which the minimizer is precisely the smoothed mean:

$$\mathbf{m}_k^s = \arg\min_{\mathbf{x}_k} L_{k:T}(\mathbf{x}_k) \tag{141}$$

Consider now the derivative of the optimal cost $J_{k+1:T}$ with respect to the filtered mean $\mathbf{m}_k$:

$$\frac{\partial J_{k+1:T}}{\partial \mathbf{m}_k} = \frac{\partial L_{k:T}}{\partial \mathbf{m}_k}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s} + \frac{\partial L_{k:T}}{\partial \mathbf{x}_k}\frac{\partial \mathbf{x}_k}{\partial \mathbf{m}_k}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s} \tag{142}$$

$$= \mathbf{P}_k^{-1}(\mathbf{m}_k - \mathbf{m}_k^s) \tag{143}$$

since the second term vanishes due to the first order optimality condition:

$$\frac{\partial L_{k:T}}{\partial \mathbf{x}_k}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s} = \mathbf{P}_k^{-1}(\mathbf{m}_k^s - \mathbf{m}_k) + \frac{\partial f_{k+1:T}}{\partial \mathbf{x}_k}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s} = 0. \tag{144}$$

Using the definition of $\boldsymbol{\lambda}_k^+$ from Eq. (120) directly results in its relation to the smoothed mean $\mathbf{m}_k^s$:

$$\boldsymbol{\lambda}_k^+ = \mathbf{P}_k^{-1}(\mathbf{m}_k - \mathbf{m}_k^s) \tag{145}$$

To recover the relation to the smoothed covariance, derive Eq. (145) with respect to the filtered mean $\mathbf{m}_k$:

$$\boldsymbol{\Lambda}_k^+ = \left(\mathbf{I} - \frac{\partial \mathbf{m}_k^s}{\partial \mathbf{m}_k}\right)^\top \mathbf{P}_k^{-1} \tag{146}$$

In order to obtain an expression for the partial derivative of the smoothed mean with respect to the filtered mean, derive the first-order optimality condition in Eq. (144) with respect to the filtered mean $\mathbf{m}_k$:

$$\mathbf{P}_k^{-1}\left(\frac{\partial \mathbf{m}_k^s}{\partial \mathbf{m}_k} - \mathbf{I}\right) + \frac{\partial^2 f_{k+1:T}}{\partial \mathbf{x}_k^2}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s}\frac{\partial \mathbf{m}_k^s}{\partial \mathbf{m}_k} = 0 \tag{147}$$

$$\left(\mathbf{P}_k^{-1} + \frac{\partial^2 f_{k+1:T}}{\partial \mathbf{x}_k^2}\bigg|_{\mathbf{x}_k=\mathbf{m}_k^s}\right)\frac{\partial \mathbf{m}_k^s}{\partial \mathbf{m}_k} = \mathbf{P}_k^{-1} \tag{148}$$

The matrix between parentheses is the Hessian of the posterior cost at the optimum and is therefore equal to $[\mathbf{P}_k^s]^{-1}$, hence:

$$\frac{\partial \mathbf{m}_k^s}{\partial \mathbf{m}_k} = \mathbf{P}_k^s\mathbf{P}_k^{-1} \tag{149}$$

which directly relates the smoothed covariance to the adjoint variable $\boldsymbol{\Lambda}_k^+$:

$$\boldsymbol{\Lambda}_k^+ = (\mathbf{I} - \mathbf{P}_k^s\mathbf{P}_k^{-1})^\top\mathbf{P}_k^{-1} \tag{150}$$

$$= \mathbf{P}_k^{-1}(\mathbf{P}_k - \mathbf{P}_k^s)\mathbf{P}_k^{-1} \tag{151}$$

A similar derivation can be done to determine the relation between the smoothed mean and covariance and the set $(\boldsymbol{\lambda}_k^-, \boldsymbol{\Lambda}_k^-)$.

C Derivation: Correction Equations

Parellier et al. (2023) derived backward recursions for differentiating scalar objectives through the Kalman filter. We use these results to obtain the gradients of the negative log marginal likelihood with respect to the predicted and filtered covariance matrices, which are not directly provided by the MBF smoothing equations. Note that our definition of the negative log evidence in Eq. (41) differs by a factor of 2 from the one presented in Parellier et al. (2023).

To simplify notation, define

$$\mathbf{B}_k = \mathbf{I} - \mathbf{K}_k\mathbf{H}_k \tag{152}$$

The corresponding covariance-gradient recursions are then given by (Parellier et al., 2023):

$$\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_k} = \mathbf{A}_k^\top\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_{k+1}^-}\mathbf{A}_k \tag{153}$$

$$\begin{aligned}\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_{k^-}} &= \mathbf{B}_k^\top\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \mathbf{P}_k}\mathbf{B}_k + \frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k \\ &+ \frac{1}{2}\mathbf{B}_k^\top\boldsymbol{\lambda}_k^+\mathbf{z}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{B}_k \\ &+ \frac{1}{2}\mathbf{B}_k^\top\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{z}_k\left(\boldsymbol{\lambda}_k^+\right)^\top\mathbf{B}_k \\ &- \frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k\mathbf{z}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k\end{aligned} \tag{154}$$

Substituting the decompositions in Eqs. (47) to (50) into these recursions separates the full covariance gradients into the MBF contribution and a remaining correction term. Using the MBF smoothing equations to cancel the terms involving $(\boldsymbol{\Lambda}_k^+, \boldsymbol{\Lambda}_k^-)$ then yields the correction recurrences in Eqs. (51) to (53) for $(\widetilde{\boldsymbol{\Lambda}}_k^+, \widetilde{\boldsymbol{\Lambda}}_k^-)$.

D Derivation: Gradient Expressions

This Appendix derives the individual terms appearing in Eqs. (54) to (55). For compactness, we use index notation and assume summation over repeated indices.

D.1 Derivative with respect to $\mathbf{A}_k$

By the chain rule,

$$\frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial [\mathbf{A}_k]_{ij}} = \frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \left[\mathbf{m}_{k+1}^-\right]_p}\frac{\partial \left[\mathbf{m}_{k+1}^-\right]_p}{\partial [\mathbf{A}_k]_{ij}} \tag{155}$$

$$+ \frac{\partial J_{1:T}^{\mathrm{NLL}}}{\partial \left[\mathbf{P}_{k+1}^-\right]_{rs}}\frac{\partial \left[\mathbf{P}_{k+1}^-\right]_{rs}}{\partial [\mathbf{A}_k]_{ij}} \tag{156}$$

The first term follows from:

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{m}^-_{k+1}\right]_p} = \frac{\partial J^{\mathrm{NLL}}_{k+1:T}}{\partial \left[\mathbf{m}^-_{k+1}\right]_p} \tag{157}$$

$$= [\boldsymbol{\lambda}^-_{k+1}]_p \tag{158}$$

$$\frac{\partial \left[\mathbf{m}^-_{k+1}\right]_p}{\partial [\mathbf{A}_k]_{ij}} = \frac{\partial \left[\mathbf{A}_k\right]_{pq} \left[\mathbf{m}_k\right]_q}{\partial [\mathbf{A}_k]_{ij}} \tag{159}$$

$$= \delta_{ip}\delta_{jq} \left[\mathbf{m}_k\right]_q \tag{160}$$

$$= \delta_{ip} \left[\mathbf{m}_k\right]_j \tag{161}$$

Hence,

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{m}^-_{k+1}\right]_p} \frac{\partial \left[\mathbf{m}^-_{k+1}\right]_p}{\partial [\mathbf{A}_k]_{ij}} = [\boldsymbol{\lambda}^-_{k+1}]_p \delta_{ip} \left[\mathbf{m}_k\right]_j \tag{162}$$

$$= [\boldsymbol{\lambda}^-_{k+1} \mathbf{m}_k^\top]_{ij} \tag{163}$$

Considering the second term:

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} = \frac{\partial J^{\mathrm{NLL}}_{k+1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} \tag{164}$$

$$= [\mathbf{T}^-_{k+1}]_{rs} \tag{165}$$

$$\frac{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}}{\partial [\mathbf{A}_k]_{ij}} = \frac{\partial ([\mathbf{A}_k]_{rt}[\mathbf{P}_k]_{tu}[\mathbf{A}_k]_{su} + [\mathbf{Q}_k]_{rs})}{\partial [\mathbf{A}_k]_{ij}} \tag{166}$$

$$= \delta_{ri}[\mathbf{P}_k \mathbf{A}_k^\top]_{js} + \delta_{si}[\mathbf{A}_k \mathbf{P}_k]_{rj} \tag{167}$$

Hence, this simplifies to:

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} \frac{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}}{\partial [\mathbf{A}_k]_{ij}} = [\mathbf{T}^-_{k+1}]_{rs}\delta_{ri}[\mathbf{P}_k \mathbf{A}_k^\top]_{js} \tag{168}$$

$$+ [\mathbf{T}^-_{k+1}]_{rs}\delta_{si}[\mathbf{A}_k \mathbf{P}_k]_{rj} \tag{169}$$

$$= 2[\mathbf{T}^-_{k+1} \mathbf{A}_k \mathbf{P}_k]_{ij} \tag{170}$$

Therefore,

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{A}_k} = \boldsymbol{\lambda}^-_{k+1} \mathbf{m}_k^\top + 2\mathbf{T}^-_{k+1} \mathbf{A}_k \mathbf{P}_k \tag{171}$$

D.2 Derivative with respect to $\mathbf{Q}_k$

Again by the chain rule,

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial [\mathbf{Q}_k]_{ij}} = \frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{m}^-_{k+1}\right]_p} \frac{\partial \left[\mathbf{m}^-_{k+1}\right]_p}{\partial [\mathbf{Q}_k]_{ij}} \tag{172}$$

$$+ \frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} \frac{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}}{\partial [\mathbf{Q}_k]_{ij}} \tag{173}$$

The predicted mean $\mathbf{m}^-_{k+1}$ does not depend on the process noise covariance matrix $\mathbf{Q}_k$, so the first term vanishes. For the second term:

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} = [\mathbf{T}^-_{k+1}]_{rs} \tag{174}$$

$$\frac{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}}{\partial [\mathbf{Q}_k]_{ij}} = \frac{\partial ([\mathbf{A}_k]_{rt}[\mathbf{P}_k]_{tu}[\mathbf{A}_k]_{su} + [\mathbf{Q}_k]_{rs})}{\partial [\mathbf{Q}_k]_{ij}} \tag{175}$$

$$= \delta_{ri}\delta_{sj} \tag{176}$$

Hence this simplifies to:

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}} \frac{\partial \left[\mathbf{P}^-_{k+1}\right]_{rs}}{\partial [\mathbf{Q}_k]_{ij}} = [\mathbf{T}^-_{k+1}]_{ij} \tag{177}$$

Therefore,

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{Q}_k} = \mathbf{T}^-_{k+1} \tag{178}$$

D.3 Derivative with respect to $\mathbf{P}_0$

Finally, by definition of $\mathbf{T}^+_0$,

$$\frac{\partial J^{\mathrm{NLL}}_{1:T}}{\partial \mathbf{P}_0} = \mathbf{T}^+_0 \tag{179}$$

D.4 Derivative with respect to $\mathbf{R}_k$

In this work, we do not consider the derivative of the negative log marginal likelihood w.r.t. the measurement noise covariance $\mathbf{R}_k$. For an expression, we refer to Eq. (26) in Section III.C of (Parellier et al., 2023).

E Numerical Analysis

This Appendix summarizes the computational and memory requirements of the filtering, smoothing, and hyperparameter learning algorithms considered in this work. Tables 1 to 6 report the corresponding floating-point operation (flop) counts, where a flop represents a single operation (e.g. addition, multiplication). Symmetry is exploited whenever applicable, and we assume $m < d$ when choosing operation ordering.

The smoothing methods also differ in their memory requirements. The following analysis assumes that symmetric matrices are stored using only their upper or lower triangular entries. The standard RTS smoother stores the predicted and filtered means and covariances at every step, as well as the transition matrices, resulting in a total memory requirement of:

$$T(2d^2 + 3d) \tag{180}$$

expressed in the number of stored scalar entries. The Joseph-form RTS smoother additionally requires the process noise covariance matrices, Kalman gains, and observation matrices, leading to:

$$T\frac{1}{2}(5d^2 + 7d + 4md) \tag{181}$$

The MBF smoother stores the transition matrices, Kalman gains, observation matrices, Cholesky factors of the innovation covariance, and measurement residuals. If the smoothed solution is reconstructed only at T_{sol} out of a total of T time steps, the total memory requirement is:

$$T(d^2 + 2dm + \frac{1}{2}m^2 + \frac{3}{2}m) + T_{\mathrm{sol}}\frac{1}{2}(d^2 + 3d) \tag{182}$$

Table 1: Flop counts of the Kalman predictor equations.

Equation	Flop Count
$\mathbf{m}_k^- = \mathbf{A}_{k-1}\mathbf{m}_{k-1}$	$2d^2 - d$
$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^\top + \mathbf{Q}_{k-1}$	$3d^3$

Table 2: Flop counts of the Kalman filtering equations.

Equation	Flop Count
$\mathbf{z}_k = \mathbf{y}_k - \mathbf{H}_k\mathbf{m}_k^-$	$2dm$
$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k$	$2d^2m + dm^2$
$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top\mathbf{S}_k^{-1}$	$2d^2m + d(2m^2 - m) + \frac{1}{3}m^3 + \frac{1}{2}m^2 + \frac{1}{6}m$
$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k\mathbf{z}_k$	$2dm$
$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-$	$3d^2m$
Joseph form	$7d^2m + 2dm^2$

Table 3: Flop counts of the RTS smoothing equations.

Equation	Flop Count
$\mathbf{J}_k = \mathbf{P}_k\mathbf{A}_k^\top[\mathbf{P}_{k+1}^-]^{-1}$	$\frac{13}{3}d^3 - \frac{1}{2}d^2 + \frac{1}{6}d$
$\mathbf{m}_k^s = \mathbf{m}_k + \mathbf{J}_k(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-)$	$2d^2 + d$
$\mathbf{P}_k^s = \mathbf{P}_k + \mathbf{J}_k(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\mathbf{J}_k^\top$	$3d^3 + \frac{1}{2}d^2 + \frac{1}{2}d$
Joseph form	$7d^3 + \frac{5}{2}d^2 + \frac{3}{2}d$

Table 4: Flop counts of the MBF smoothing equations.

Equation	Flop Count
$\boldsymbol{\lambda}_{k-1}^+ = \mathbf{A}_{k-1}^\top\boldsymbol{\lambda}_k^-$	$2d^2 - d$
$\boldsymbol{\Lambda}_{k-1}^+ = \mathbf{A}_{k-1}^\top\boldsymbol{\Lambda}_k^-\mathbf{A}_{k-1}$	$3d^3 - \frac{1}{2}d^2 - \frac{1}{2}d$
$\boldsymbol{\lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top\boldsymbol{\lambda}_k^+ - \mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k$	$6dm + 2m^2 - m$
$\boldsymbol{\Lambda}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top\boldsymbol{\Lambda}_k^+(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) + \mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k$	$d^2(7m+\frac{1}{2}) + d(4m^2+2m+\frac{1}{2})$
$\mathbf{m}_k^s = \mathbf{m}_k - \mathbf{P}_k\boldsymbol{\lambda}_k^+$	$2d^2$
$\mathbf{P}_k^s = \mathbf{P}_k - \mathbf{P}_k\boldsymbol{\Lambda}_k^+\mathbf{P}_k$	$3d^3$

Table 5: Flop counts of the MBF correction equations.

Equation	Flop Count
$\mathbf{l}_k = \frac{1}{2}\boldsymbol{\lambda}_k^+\mathbf{z}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k$	$d(2m^2 + 3m - 1) + \frac{1}{3}m^3 + \frac{1}{2}m^2 + \frac{1}{6}m$
$\widetilde{\boldsymbol{\Lambda}}_k^- = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top(\widetilde{\boldsymbol{\Lambda}}_k^+ + \mathbf{l}_k + \mathbf{l}_k^\top)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) - \frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k$ $-\frac{1}{2}\mathbf{H}_k^\top\mathbf{S}_k^{-1}\mathbf{z}_k\mathbf{z}_k^\top\mathbf{S}_k^{-1}\mathbf{H}_k$	$d^2(7m + 4) + d(2m^2 + 3m + 2) + 2m^2$
$\widetilde{\boldsymbol{\Lambda}}_{k-1}^+ = \mathbf{A}_{k-1}^\top\widetilde{\boldsymbol{\Lambda}}_k^-\mathbf{A}_{k-1}$	$3d^3 - \frac{1}{2}d^2 - \frac{1}{2}d$

Table 6: Flop counts of the gradient components of the negative log marginal likelihood.

Equation	Flop Count
$\sum_{k=0}^{T-1}\left\langle(\boldsymbol{\lambda}_{k+1}^-\mathbf{m}_k^\top + 2\mathbf{T}_{k+1}^-\mathbf{A}_k\mathbf{P}_k), \frac{\mathrm{d}\mathbf{A}_k}{\mathrm{d}\theta}\right\rangle$	$T(3d^3 + \frac{5}{2}d^2 + \frac{1}{2}d)$
$\sum_{k=0}^{T-1}\left\langle\mathbf{T}_{k+1}^-, \frac{\mathrm{d}\mathbf{Q}_k}{\mathrm{d}\theta}\right\rangle$	$T(d^2 + d)$
$\left\langle\mathbf{T}_0^+, \frac{\mathrm{d}\mathbf{P}_0}{\mathrm{d}\theta}\right\rangle$	$d^2 + d$