Yaghoobi, Fatemeh; Abdulsamad, Hany; Särkkä, Simo

# A Recursive Newton Method for Smoothing in Nonlinear State Space Models

*Published in:*
31st European Signal Processing Conference, EUSIPCO 2023 - Proceedings

*DOI:*
10.23919/EUSIPCO58844.2023.10290119

Published: 01/01/2023

*Document Version*
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

*Please cite the original version:*
Yaghoobi, F., Abdulsamad, H., & Särkkä, S. (2023). A Recursive Newton Method for Smoothing in Nonlinear State Space Models. In *31st European Signal Processing Conference, EUSIPCO 2023 - Proceedings* (pp. 1758-1762). (European Signal Processing Conference). European Signal Processing Conference (EUSIPCO). https://doi.org/10.23919/EUSIPCO58844.2023.10290119

# A Recursive Newton Method for Smoothing in Nonlinear State Space Models

Fatemeh Yaghoobi*, Hany Abdulsamad†, Simo Särkkä
Department of Electrical Engineering and Automation, Aalto University, Finland
{fatemeh.yaghoobi, hany.abdulsamad, simo.sarkka}@aalto.fi

***Abstract*—In this paper, we use the optimization formulation of nonlinear Kalman filtering and smoothing problems to develop second-order variants of iterated Kalman smoother (IKS) methods. We show that Newton's method corresponds to a recursion over affine smoothing problems on a modified state-space model augmented by a pseudo measurement. The first and second derivatives required in this approach can be efficiently computed with widely available automatic differentiation tools. Furthermore, we show how to incorporate line-search and trust-region strategies into the proposed second-order IKS algorithm in order to regularize updates between iterations. Finally, we provide numerical examples to demonstrate the method's efficiency in terms of runtime compared to its batch counterpart.**

***Index Terms*—Newton's method, state-space model, iterated Kalman filter and smoother, line search, trust region.**

## I. Introduction

State estimation problem in nonlinear state-space models (SSMs) plays an important role in various areas of applications such as in control theory, signal processing, and robotics [1]–[3]. In this paper, we are interested in solving state estimation problems in SSMs of the form

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}, \quad \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k, \tag{1}$$

$\mathbf{x}_k \in \mathbb{R}^d$ is the state at time step $k$, $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement at the same time step, $\mathbf{f}(.)$ is the state transition function, and $\mathbf{h}(.)$ is the observation function. Furthermore, $\mathbf{q}_k$ and $\mathbf{r}_k$ are the process and measurement noises, assumed to be Gaussian with zero mean and covariance matrices $\mathbf{Q}$ and $\mathbf{R}$, respectively. The prior distribution of the state at $k = 0$ is Gaussian with known mean $\mathbf{m}_0$ and covariance $\mathbf{P}_0$.

The smoothing problem (see, e.g., [1]) amounts to computing the estimate of the state $\mathbf{x}_k$ given a batch of measurements $\mathbf{y}_1, \ldots, \mathbf{y}_N$, where $k \in \{0, \ldots, N\}$. The Kalman filter [4] and Rauch–Tung–Striebel (RTS) smoother [5] for linear SSM and their extension for nonlinear systems (see, e.g., [1], [2], [6]–[11]) provide powerful recursive solutions which are optimal in the minimum mean squared error (MMSE) sense.

On the other hand, the smoothing problem can be viewed in an optimization framework (see, e.g., [8], [12]), where the aim is to find the maximum a posteriori (MAP) trajectory estimate, that is, the trajectory $\mathbf{x}^*_{0:N}$ which maximizes $p(\mathbf{x}_{0:N} \mid \mathbf{y}_{1:N})$.

This work was funded by the Academy of Finland* and the Finnish Center for Artificial Intelligence (FCAI)†.

For the SSM of the form (1), the MAP estimate is the minimizer of the negative log-posterior

$$\mathbf{x}^*_{0:N} = \operatorname*{arg\,min}_{\mathbf{x}_{0:N}} \; L(\mathbf{x}_{0:N}), \tag{2}$$

where the negative log-posterior is given by

$$L(\mathbf{x}_{0:N}) = \frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} + \frac{1}{2}\sum_{k=1}^{N}\|\mathbf{x}_k - \mathbf{f}(\mathbf{x}_{k-1})\|^2_{\mathbf{Q}^{-1}} + \frac{1}{2}\sum_{k=1}^{N}\|\mathbf{y}_k - \mathbf{h}(\mathbf{x}_k)\|^2_{\mathbf{R}^{-1}}, \quad \text{with } \|\mathbf{x}\|^2_{\mathbf{A}} \coloneqq \mathbf{x}^\top \mathbf{A}\mathbf{x}. \tag{3}$$

Viewing the state estimation problem from an optimization standpoint enables us to employ several optimization techniques [13]. One widely-used example in the filtering and smoothing literature is the Gauss–Newton (GN) method [14]–[16], which has a close relationship with iterated extended Kalman filtering and smoothing methods [8], [17]. In particular, for nonlinear SSM with additive Gaussian noise, Bell [8] proved that the GN-method is equivalent to the iterated extended Kalman smoother (IEKS), a recursive method with less computational complexity than batch GN-methods. Recently, Särkkä & Svensson [12] developed line-search and Levenberg-Marquart extensions of the IEKS method.

Newton's method has received less attention as an optimization method to solve smoothing problems due to the effort associated with computing second-order derivatives. However, the availability of automatic differentiation tools has eliminated the need for manual computation, making Newton's method attractive for smoothing problems. Although the application of Newton's method to filtering and smoothing has been mentioned in literature [18]–[20], the full Newton version of the IKS is yet to be realized.

The contribution of this paper is to develop the Newton formulation of iterated Kalman smoothers while leveraging automatic differentiation tools to compute the derivatives and Hessians. We also present robust modifications of the proposed method that incorporate line-search and trust-region schemes into the recursive structure.

This paper is structured as follows: Section II presents Newton's method for the MAP problem in batch and recursive form. Section III presents line-search and trust-region strategies to enhance the robustness of iterative Newton updates. Section IV analyzes the efficiency of the proposed recursive methods in the sense of runtime on a numerical example.

## II. Newton Iterated Kalman Smoother

Assuming a SSM of the form (1) and the objective specified in Equation (3), our aim, in this section, is to use Newton's optimization technique to minimize the objective function and develop the corresponding batch solution. Subsequently, we present a recursive alternative analogous to the IKS to improve computational efficiency.

### A. Batch Newton Optimization

The batch solution for smoothing follows the standard iterative optimization framework without specifically leveraging the underlying temporal structure of the problem. Accordingly, we can implement Newton's method as a generic second-order optimization of Equation (3) with respect to a decision variable $\mathbf{x}_{0:N}$ with a dimension $d_N = d \times N$.

At every iteration $i$, Newton's method approximates a twice differentiable objective $L(\mathbf{x}_{0:N})$ up to the second order in the neighborhood of a nominal trajectory $\hat{\mathbf{x}}_{0:N}^{(i)}$

$$\begin{aligned} L(\mathbf{x}_{0:N}) \approx L(\hat{\mathbf{x}}_{0:N}^{(i)}) + \nabla L^\top(\hat{\mathbf{x}}_{0:N}^{(i)})(\mathbf{x}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)}) \\ + \frac{1}{2}(\mathbf{x}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)})^\top \nabla^2 L(\hat{\mathbf{x}}_{0:N}^{(i)})(\mathbf{x}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)}), \end{aligned} \tag{4}$$

where $\nabla L(.)$ and $\nabla^2 L(.)$ denote the gradient and the Hessian of $L(.)$, respectively. Using this quadratic approximation, we get the Newton update rule

$$\hat{\mathbf{x}}_{0:N}^{(i+1)} = \hat{\mathbf{x}}_{0:N}^{(i)} - (\nabla^2 L(\hat{\mathbf{x}}_{0:N}^{(i)}) + \lambda\, \mathbf{I}_{d_N})^{-1}\, \nabla L(\hat{\mathbf{x}}_{0:N}^{(i)}). \tag{5}$$

Note that we have included a diagonal regularization term $\lambda\, \mathbf{I}_{d_N}$, with $\lambda \geq 0$, to ensure a positive-definite Hessian and a valid descent direction.

Despite the convenience of automatic differentiation frameworks that readily deliver $\nabla L(.)$ and $\nabla^2 L(.)$, the computational effort associated with the Newton update in Equation (5) is still a major issue. The Hessian $\nabla^2 L(.)$ is of dimensions $d_N \times d_N$, and its inversion leads to a worst-case computational complexity $\mathcal{O}(N^3 d^3)$, which scales poorly both in the state dimension and the trajectory length.

In the following, we will rely on the quadratic approximation in Equation (4). However, by taking advantage of the temporal structure of the state-space model, we will construct a modified affine state-space model and derive a recursive algorithm akin to the iterated Kalman smoother, leading to a considerable reduction in computational complexity.

### B. Recursive Newton Optimization

Constructing the modified state-space model requires analyzing the first- and second-order approximations of the individual terms in Equation (3). We start by considering the approximation of the transition dynamics term. For convenience, we define

$$S(\mathbf{x}_{0:N}) \coloneqq \sum_{k=1}^{N} S_k(\mathbf{x}_k, \mathbf{x}_{k-1}) = \sum_{k=1}^{N} \|\mathbf{x}_k - \mathbf{f}(\mathbf{x}_{k-1})\|^2_{\mathbf{Q}^{-1}},$$

and expand it around the current nominal trajectory $\hat{\mathbf{x}}_{0:N}^{(i)}$. For a simplified notation, we drop the iteration index $i$

$$\begin{aligned} S(\mathbf{x}_{0:N}) \approx \frac{1}{2}\delta\mathbf{x}_{0:N}^\top\, \nabla^2 S(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} \\ + \nabla S^\top(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} + S(\hat{\mathbf{x}}_{0:N}), \end{aligned} \tag{6}$$

where $\delta\mathbf{x}_{0:N} = \mathbf{x}_{0:N} - \hat{\mathbf{x}}_{0:N}$ and

$$\begin{aligned} \nabla S^\top(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} = 2\sum_{k=1}^{N} (\hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_{k-1}))^\top \mathbf{Q}^{-1}\, \delta\mathbf{x}_k \\ - 2\sum_{k=1}^{N} (\hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_{k-1}))^\top \mathbf{Q}^{-1} \mathbf{F}_{\mathbf{x}}(\hat{\mathbf{x}}_{k-1})\, \delta\mathbf{x}_{k-1}, \end{aligned} \tag{7}$$

$$\begin{aligned} \frac{1}{2}\delta\mathbf{x}_{0:N}^\top\, \nabla^2 S(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} = \sum_{k=1}^{N} \delta\mathbf{x}_k^\top \mathbf{Q}^{-1} \delta\mathbf{x}_k \\ - 2\sum_{k=1}^{N} \delta\mathbf{x}_k^\top \mathbf{F}_{\mathbf{x}}^\top(\hat{\mathbf{x}}_{k-1})\, \mathbf{Q}^{-1} \delta\mathbf{x}_{k-1} \\ + \sum_{k=1}^{N} \delta\mathbf{x}_{k-1}^\top \mathbf{F}_{\mathbf{x}}^\top(\hat{\mathbf{x}}_{k-1})\, \mathbf{Q}^{-1} \mathbf{F}_{\mathbf{x}}(\hat{\mathbf{x}}_{k-1})\, \delta\mathbf{x}_{k-1} \\ - \sum_{k=1}^{N} \delta\mathbf{x}_{k-1}^\top \mathbf{F}_{\mathbf{xx}}^\top(\hat{\mathbf{x}}_{k-1}) \cdot \mathbf{Q}^{-1}(\hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_{k-1}))\, \delta\mathbf{x}_{k-1}, \end{aligned} \tag{8}$$

where $\mathbf{F}_{\mathbf{x}}(.)$ is the Jacobian and $\mathbf{F}_{\mathbf{xx}}(.)$ is a third-rank Hessian tensor of the transition function $\mathbf{f}(.)$. The notation $(M \cdot v)$ refers to a tensor dot product so that $(M \cdot v)_{ij} = \sum_k M_{ijk} v_k$.

Plugging Equations (7) and (8) into Equation (6) and applying simple algebraic manipulations, we arrive at the following decomposition of the quadratic expansion in Equation (6)

$$\begin{aligned} S(\mathbf{x}_{0:N}) \approx \sum_{k=1}^{N} \|\mathbf{x}_k - \mathbf{F}_{k-1}\, \mathbf{x}_{k-1} - \mathbf{b}_{k-1}\|^2_{\mathbf{Q}^{-1}} \\ + \sum_{k=1}^{N} \|\hat{\mathbf{x}}_{k-1} - \mathbf{x}_{k-1}\|^2_{\mathbf{\Psi}_{k-1}}, \end{aligned} \tag{9}$$

where

$$\begin{aligned} \mathbf{F}_{k-1} &= \mathbf{F}_{\mathbf{x}}(\hat{\mathbf{x}}_{k-1}), \\ \mathbf{b}_{k-1} &= \mathbf{f}(\hat{\mathbf{x}}_{k-1}) - \mathbf{F}_{\mathbf{x}}(\hat{\mathbf{x}}_{k-1})\, \hat{\mathbf{x}}_{k-1}, \\ \mathbf{\Psi}_{k-1} &= -\mathbf{F}_{\mathbf{xx}}^\top(\hat{\mathbf{x}}_{k-1}) \cdot \mathbf{Q}^{-1}(\hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_{k-1})). \end{aligned}$$

A similar second-order expansion can be carried out for the observation model term in Equation (3). Again, for convenience, we define the following

$$G(\mathbf{x}_{0:N}) \coloneqq \sum_{k=1}^{N} G_k(\mathbf{x}_k) = \sum_{k=1}^{N} \|\mathbf{y}_k - \mathbf{h}(\mathbf{x}_k)\|^2_{\mathbf{R}^{-1}},$$

and expand it to the second order around $\hat{\mathbf{x}}_{0:N}$

$$\begin{aligned} G(\mathbf{x}_{0:N}) \approx \frac{1}{2}\delta\mathbf{x}_{0:N}^\top\, \nabla^2 G(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} \\ + \nabla G^\top(\hat{\mathbf{x}}_{0:N})\, \delta\mathbf{x}_{0:N} + G(\hat{\mathbf{x}}_{0:N}), \end{aligned} \tag{10}$$

where the linear and quadratic terms are

$$\nabla G^\top(\hat{\mathbf{x}}_{0:N})\,\delta\mathbf{x}_{0:N} = -2\sum_{k=1}^{N}(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k))^\top \mathbf{R}^{-1}\mathbf{H}_{\mathbf{x}}(\hat{\mathbf{x}}_k)\,\delta\mathbf{x}_k,$$

$$\frac{1}{2}\delta\mathbf{x}_{0:N}^\top \nabla^2 G(\hat{\mathbf{x}}_{0:N})\,\delta\mathbf{x}_{0:N} = \sum_{k=1}^{N}\delta\mathbf{x}_k^\top \mathbf{H}_{\mathbf{x}}^\top(\hat{\mathbf{x}}_k)\,\mathbf{R}^{-1}\mathbf{H}_{\mathbf{x}}(\hat{\mathbf{x}}_k)\delta\mathbf{x}_k - \sum_{k=1}^{N}\delta\mathbf{x}_k^\top \mathbf{H}_{\mathbf{xx}}^\top(\hat{\mathbf{x}}_k)\cdot\mathbf{R}^{-1}(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k))\,\delta\mathbf{x}_k.$$

The matrix $\mathbf{H}_{\mathbf{x}}(.)$ is the Jacobian and $\mathbf{H}_{\mathbf{xx}}(.)$ is a third-rank Hessian tensor of the observation function $\mathbf{h}(.)$. Similarly, by rearranging these terms, we can construct a specific decomposition of the quadratic expansion in Equation (10)

$$G(\mathbf{x}_{0:N}) \approx \sum_{k=1}^{N}\|\mathbf{y}_k - \mathbf{H}_k\,\mathbf{x}_k - \mathbf{c}_k\|^2_{\mathbf{R}^{-1}} + \sum_{k=1}^{N}\|\hat{\mathbf{x}}_k - \mathbf{x}_k\|^2_{\mathbf{\Gamma}_k}, \quad (11)$$

where

$$\mathbf{H}_k = \mathbf{H}_{\mathbf{x}}(\hat{\mathbf{x}}_k),$$
$$\mathbf{c}_k = \mathbf{h}(\hat{\mathbf{x}}_k) - \mathbf{H}_{\mathbf{x}}(\hat{\mathbf{x}}_k)\,\hat{\mathbf{x}}_k,$$
$$\mathbf{\Gamma}_k = -\mathbf{H}_{\mathbf{xx}}^\top(\hat{\mathbf{x}}_k)\cdot\mathbf{R}^{-1}(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k)).$$

We can now take the second-order terms of the transition and observation functions in Equations (9) and (11) and plug them back into the objective in Equation (3) which leads to the overall (regularized) second-order approximation

$$\begin{aligned}\tilde{L}(\mathbf{x}_{0:N}) = {} & \frac{1}{2}\|\mathbf{x}_0 - \mathbf{m}_0\|^2_{\mathbf{P}_0^{-1}} + \frac{1}{2}\|\mathbf{x}_0 - \hat{\mathbf{x}}_0\|^2_{\mathbf{\Phi}_0^{-1}} \\ & + \frac{1}{2}\sum_{k=1}^{N}\|\hat{\mathbf{x}}_k - \mathbf{x}_k\|^2_{\mathbf{\Phi}_k^{-1}} + \frac{1}{2}\sum_{k=1}^{N}\|\mathbf{y}_k - \mathbf{H}_k\,\mathbf{x}_k - \mathbf{c}_k\|^2_{\mathbf{R}^{-1}} \\ & + \frac{1}{2}\sum_{k=1}^{N}\|\mathbf{x}_k - \mathbf{F}_{k-1}\,\mathbf{x}_{k-1} - \mathbf{b}_{k-1}\|^2_{\mathbf{Q}^{-1}}, \end{aligned} \quad (12)$$

where

$$\mathbf{\Phi}_0 = (\mathbf{\Psi}_0 + \lambda\,\mathbf{I}_d)^{-1},$$
$$\mathbf{\Phi}_k = (\mathbf{\Psi}_k + \mathbf{\Gamma}_k + \lambda\,\mathbf{I}_d)^{-1},$$
$$\mathbf{\Phi}_N = (\mathbf{\Gamma}_N + \lambda\,\mathbf{I}_d)^{-1}.$$

The result in Equation (12) indicates that the second-order approximation of $L(.)$ can be viewed as a first-order approximation of the functions $\mathbf{f}$ and $\mathbf{h}$, augmented by an affine pseudo observation model, in which the expansion point $\hat{\mathbf{x}}_k$ acts as a pseudo measurement of the state $\mathbf{x}_k$. This interpretation of (12) corresponds to the *modified* state-space model of the form

$$\begin{aligned} \mathbf{x}_k &\approx \mathbf{F}_{k-1}\,\mathbf{x}_{k-1} + \mathbf{b}_{k-1} + \mathbf{q}_k, & \mathbf{q}_k &\sim \mathcal{N}(0, \mathbf{Q}), \\ \mathbf{y}_k &\approx \mathbf{H}_k\,\mathbf{x}_k + \mathbf{c}_k + \mathbf{r}_k, & \mathbf{r}_k &\sim \mathcal{N}(0, \mathbf{R}), \\ \hat{\mathbf{x}}_k &\approx \mathbf{x}_k + \mathbf{e}_k, & \mathbf{e}_k &\sim \mathcal{N}(0, \mathbf{\Phi}_k), \end{aligned}$$

with a *modified* prior distribution $\mathbf{x}_0 \sim \mathcal{N}(\tau_0, \mathbf{\Omega}_0)$

$$\mathbf{\Omega}_0 = (\mathbf{P}_0^{-1} + \mathbf{\Phi}_0^{-1})^{-1},$$
$$\tau_0 = (\mathbf{P}_0^{-1} + \mathbf{\Phi}_0^{-1})^{-1}(\mathbf{P}_0^{-1}\mathbf{m}_0 + \mathbf{\Phi}_0^{-1}\hat{\mathbf{x}}_0).$$

**Algorithm 1** One Iteration of the (Regularized) Newton-IKS

**input:** Nominal trajectory $\hat{\mathbf{x}}_{0:N}^{(i)}$, measurements $\mathbf{y}_{1:N}$, Jacobians at nominal: $\mathbf{F}_{0:N-1}$, $\mathbf{H}_{1:N}$, offsets at nominal: $\mathbf{b}_{0:N-1}, \mathbf{c}_{1:N}$, covariances at nominal: $\mathbf{Q}, \mathbf{R}, \mathbf{\Phi}_{1:N}$, prior at nominal: $\tau_0$, $\mathbf{\Omega}_0$, and optional regularization $\lambda$
**output:** Smoothed trajectory $\hat{\mathbf{x}}_{0:N}$
**procedure** NEWTON-IKS($\hat{\mathbf{x}}_{0:N}^{(i)}, \lambda$):
  Set $\mathbf{x}_0^f \leftarrow \tau_0(\lambda)$, $\mathbf{P}_0^f \leftarrow \mathbf{\Omega}_0(\lambda)$ ▷ Initialize
  **for** $k \leftarrow 1$ **to** $N$ **do**
    $\mathbf{x}_k^p \leftarrow \mathbf{F}_{k-1}\,\mathbf{x}_{k-1}^f + \mathbf{b}_{k-1}$ ▷ Prediction
    $\mathbf{P}_k^p \leftarrow \mathbf{F}_{k-1}\mathbf{P}_{k-1}^f\mathbf{F}_{k-1}^\top + \mathbf{Q}$
    $\mu_k \leftarrow \mathbf{H}_k\,\mathbf{x}_k^p + \mathbf{c}_k$
    $\mathbf{\Sigma}_k \leftarrow \mathbf{H}_k\,\mathbf{P}_k^p\,\mathbf{H}_k^\top + \mathbf{R}$
    $\mathbf{K}_k \leftarrow \mathbf{P}_k^p\,\mathbf{H}_k^\top\mathbf{\Sigma}_k^{-1}$
    $\mathbf{x}_k^y \leftarrow \mathbf{x}_k^p + \mathbf{K}_k(\mathbf{y}_k - \mu_k)$ ▷ Measure. Update
    $\mathbf{P}_k^y \leftarrow \mathbf{P}_k^p - \mathbf{K}_k\mathbf{\Sigma}_k\mathbf{K}_k^\top$
    $\mathbf{\Delta}_k \leftarrow \mathbf{P}_k^y + \mathbf{\Phi}_k(\lambda)$
    $\mathbf{U}_k \leftarrow \mathbf{P}_k^y\mathbf{\Delta}_k^{-1}$
    $\mathbf{x}_k^f \leftarrow \mathbf{x}_k^y + \mathbf{U}_k(\hat{\mathbf{x}}_k^{(i)} - \mathbf{x}_k^y)$ ▷ Pseudo Update
    $\mathbf{P}_k^f \leftarrow \mathbf{P}_k^y - \mathbf{U}_k\mathbf{\Delta}_k\mathbf{U}_k^\top$
  **end for**
  Set $\hat{\mathbf{x}}_N \leftarrow \mathbf{x}_N^f$ and $\mathbf{P}_N \leftarrow \mathbf{P}_N^f$
  **for** $k \leftarrow N-1$ **to** $0$ **do**
    $\mathbf{G}_k \leftarrow \mathbf{P}_k^f\,\mathbf{F}_k^\top\,(\mathbf{P}_{k+1}^p)^{-1}$
    $\hat{\mathbf{x}}_k \leftarrow \mathbf{x}_k^f + \mathbf{G}_k(\hat{\mathbf{x}}_{k+1} - \mathbf{x}_{k+1}^p)$ ▷ Smoothing
    $\mathbf{P}_k \leftarrow \mathbf{P}_k^f + \mathbf{G}_k(\mathbf{P}_{k+1} - \mathbf{P}_{k+1}^p)\mathbf{G}_k^\top$
  **end for**
**end procedure**

Note that we have again included a diagonal term $\lambda\,\mathbf{I}_d$ equivalent to that in Section II-A. In this modified state-space model, $\lambda\,\mathbf{I}_d$ can be interpreted as regularization of the pseudo observation model to guarantee a positive-definite covariance and well-defined Gaussian noise. The significance of this regularization will become clear in the upcoming section.

Given this modified affine state-space model, we can iteratively minimize the approximate objective in Equation (12) by implementing a recursive RTS smoother [5] that incorporates the pseudo measurements and dramatically lowers the computational complexity to $\mathcal{O}(Nd^3)$. Algorithm 1 summarizes a single iteration of a Newton iterated Kalman smoother (Newton-IKS). For more details on smoothing algorithms for affine state space models, we refer to [1].

## III. Implementation Strategies of Newton Iterated Kalman Smoothers

In the upcoming sections, we describe two algorithms for a robust implementation of the Newton iterated Kalman smoother. The line-search and trust-region strategies that we incorporate into the Newton-IKS are realizations of fundamental principles in optimization for scaling and regularizing the

update of an iterate $\hat{\mathbf{x}}_{0:N}^{(i)}$ along a direction $\mathbf{p}^{(i)}$ to guarantee a consistent reduction of the objective [13].

### A. Recursive Newton Method with Line Search

The procedure of line search assumes the existence of a direction $\mathbf{p}^{(i)}$ at a current iterate $\mathbf{x}_{0:N}^{(i)}$ and proposes an updated iterate $\mathbf{x}_{0:N}^{(i+1)}$. The distance taken along the direction $\mathbf{p}^{(i)}$ is scaled by a step size $\alpha > 0$ in a way that guarantees a reduction of the objective function

$$\hat{\mathbf{x}}_{0:N}^{(i+1)} = \hat{\mathbf{x}}_{0:N}^{(i)} + \alpha\, \mathbf{p}^{(i)}. \tag{13}$$

In our case, the Newton-IKS from Section II-B indirectly supplies the search direction of the smoothed trajectory via $\mathbf{p}^{(i)} = \hat{\mathbf{x}}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)}$, where $\hat{\mathbf{x}}_{0:N}$ is the output of Algorithm 1 given the current iterate $\hat{\mathbf{x}}_{0:N}^{(i)}$ as a nominal trajectory.

However, the direction that the Newton-IKS delivers may not be a valid search direction as the Hessian of the objective function may not be positive-definite. To overcome this challenge, we propose a simple approach that increases the diagonal regularization factor $\lambda$ until the *expected* cost reduction is positive $\tilde{L}(\hat{\mathbf{x}}_{0:N}^{(i)}) - \tilde{L}(\hat{\mathbf{x}}_{0:N}) > 0$ where $\tilde{L}(.)$ is the (regularized) second-order approximation in Equation (12), which corresponds to a descent direction.

Given a descent direction $\mathbf{p}^{(i)}$, various approaches are available for choosing $\alpha$ exactly or approximately [13]. We choose to apply a backtracking line-search scheme to find a step size $\alpha$ such that $L(\hat{\mathbf{x}}_{0:N}^{(i)} + \alpha\, \mathbf{p}^{(i)}) < L(\hat{\mathbf{x}}_{0:N}^{(i)})$, where $L(.)$ is the *original* nonlinear objective in Equation (3). Algorithm 2 provides an overview of a Newton-IKS algorithm with an approximate line-search strategy.

### B. Recursive Newton Method with a Trust Region

While line-search techniques optimize the step size along a pre-defined search direction, trust-region methods intervene and directly modify the search direction based on an approximate model of the nonlinear objective in a region around the current iterate. The size of this region implies the relative trust of the local approximation and simultaneously influences both the update direction and the step size.

In the case of the Newton-IKS, we implement a trust-region technique akin to a Levenberg-Marquardt algorithm [21]. This approach directly controls the regularization in Equation (12) to modify the search direction based on the quality of the local approximation. The quality is measured by the ratio of the *actual* cost difference to the *expected* cost difference given a nominal trajectory $\hat{\mathbf{x}}_{0:N}^{(i)}$ and a candidate solution $\hat{\mathbf{x}}_{0:N}$

$$\rho = \frac{\Delta L}{\Delta \tilde{L}} = \frac{L(\hat{\mathbf{x}}_{0:N}^{(i)}) - L(\hat{\mathbf{x}}_{0:N})}{\tilde{L}(\hat{\mathbf{x}}_{0:N}^{(i)}) - \tilde{L}(\hat{\mathbf{x}}_{0:N})}.$$

An update is accepted when $\rho > 0$, implying that the current approximation is close to the true underlying objective around the current iterate, and the trust region is enlarged accordingly by reducing $\lambda$. When $\rho \leq 0$, the update is rejected, and the region is tightened by increasing $\lambda$. Algorithm 3 provides an overview of the Newton-IKS with a trust-region strategy.

**Algorithm 2** Newton-IKS with Line Search

**input:** Initial trajectory $\hat{\mathbf{x}}_{0:N}^{(0)}$, measurements $\mathbf{y}_{1:N}$, Models, Hessians, and Jacobians: $\mathbf{f}, \mathbf{h}, \mathbf{F_x}, \mathbf{H_x}, \mathbf{F_{xx}}, \mathbf{H_{xx}}$, constants: $\mathbf{m}_0, \mathbf{P}_0, \mathbf{Q}, \mathbf{R}$, backtracking mult. $\beta \in (0, 1)$, backtracking iterations $M$, and overall iterations $N_i$
**output:** The MAP trajectory $\hat{\mathbf{x}}_{0:N}^{*}$
**for** $0 \leq i < N_i$ **do**
  $\hat{\mathbf{x}}_{0:N} \leftarrow$ NEWTON-IKS($\hat{\mathbf{x}}_{0:N}^{(i)}, \lambda = 0$)
  **if** $\tilde{L}(\hat{\mathbf{x}}_{0:N}^{(i)}) - \tilde{L}(\hat{\mathbf{x}}_{0:N}) > 0$ **then**
    $\mathbf{p}^{(i)} \leftarrow \hat{\mathbf{x}}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)}$ ▷ Descent direction
  **else**
    Set $\lambda \leftarrow 10^{-6}$
    $\hat{\mathbf{x}}_{0:N} \leftarrow$ NEWTON-IKS($\hat{\mathbf{x}}_{0:N}^{(i)}, \lambda$) ▷ Regularize
    **while** $\tilde{L}(\hat{\mathbf{x}}_{0:N}^{(i)}) - \tilde{L}(\hat{\mathbf{x}}_{0:N}) \leq 0$ **and** $\lambda \leq 10^{16}$ **do**
      $\lambda \leftarrow 10\,\lambda$
      $\hat{\mathbf{x}}_{0:N} \leftarrow$ NEWTON-IKS($\hat{\mathbf{x}}_{0:N}^{(i)}, \lambda$)
    **end while**
    $\mathbf{p}^{(i)} \leftarrow \hat{\mathbf{x}}_{0:N} - \hat{\mathbf{x}}_{0:N}^{(i)}$
  **end if**
  Set $\alpha \leftarrow 1$, $m \leftarrow 0$
  **while** $L(\hat{\mathbf{x}}_{0:N}^{(i)} + \alpha\, \mathbf{p}^{(i)}) \geq L(\hat{\mathbf{x}}_{0:N}^{(i)})$ **and** $m \leq M$ **do**
    $\alpha \leftarrow \beta\, \alpha$, $m \leftarrow m + 1$ ▷ Backtracking
  **end while**
  **if** $L(\hat{\mathbf{x}}_{0:N}^{(i)} + \alpha\, \mathbf{p}^{(i)}) < L(\hat{\mathbf{x}}_{0:N}^{(i)})$ **then**
    $\hat{\mathbf{x}}_{0:N}^{(i+1)} \leftarrow \hat{\mathbf{x}}_{0:N}^{(i)} + \alpha\, \mathbf{p}^{(i)}$ ▷ Accept step
  **else**
    $\hat{\mathbf{x}}_{0:N}^{(i+1)} \leftarrow \hat{\mathbf{x}}_{0:N}^{(i)}$ ▷ Reject step
  **end if**
**end for**

**Algorithm 3** Newton-IKS with a Trust Region

**input:** Initial trajectory $\hat{\mathbf{x}}_{0:N}^{(0)}$, measurements $\mathbf{y}_{1:N}$, Models, Hessians, and Jacobians: $\mathbf{f}, \mathbf{h}, \mathbf{F_x}, \mathbf{H_x}, \mathbf{F_{xx}}, \mathbf{H_{xx}}$, constants: $\mathbf{m}_0$, $\mathbf{P}_0$, $\mathbf{Q}$, $\mathbf{R}$, initial regularization $\lambda_0$, regularization mult. $\nu > 1$, and overall iterations $N_i$
**output:** The MAP trajectory $\hat{\mathbf{x}}_{0:N}^{*}$
Set $\lambda \leftarrow \lambda_0$ $\nu \leftarrow 2$
**for** $0 \leq i < N_i$ **do**
  $\hat{\mathbf{x}}_{0:N} \leftarrow$ NEWTON-IKS($\hat{\mathbf{x}}_{0:N}^{(i)}, \lambda$)
  $\Delta L \leftarrow L(\hat{\mathbf{x}}_{0:N}^{(i)}) - L(\hat{\mathbf{x}}_{0:N})$ ▷ Actual cost diff.
  $\Delta \tilde{L} \leftarrow \tilde{L}(\hat{\mathbf{x}}_{0:N}^{(i)}) - \tilde{L}(\hat{\mathbf{x}}_{0:N})$ ▷ Expected cost diff.
  $\rho \leftarrow \Delta L / \Delta \tilde{L}$
  **if** $\rho > 0$ **and** $\Delta \tilde{L} > 0$ **then**
    $\lambda \leftarrow \lambda \max\{\frac{1}{3}, 1 - (2\,\rho - 1)^3\}$, $\nu \leftarrow 2$
    $\hat{\mathbf{x}}_{0:N}^{(i+1)} \leftarrow \hat{\mathbf{x}}_{0:N}$ ▷ Accept step
  **else**
    $\lambda \leftarrow \nu\, \lambda$, $\nu \leftarrow 2\,\nu$
    $\hat{\mathbf{x}}_{0:N}^{(i+1)} \leftarrow \hat{\mathbf{x}}_{0:N}^{(i)}$ ▷ Reject step
  **end if**
**end for**

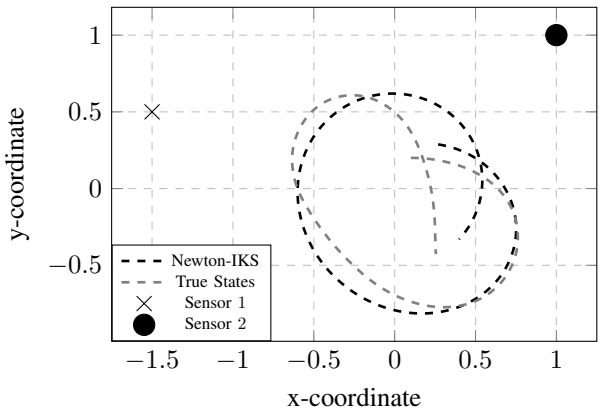


Fig. 1. Example of a smoothed trajectory obtained from a Newton-IKS with a trust-region method in the coordinated turn model.

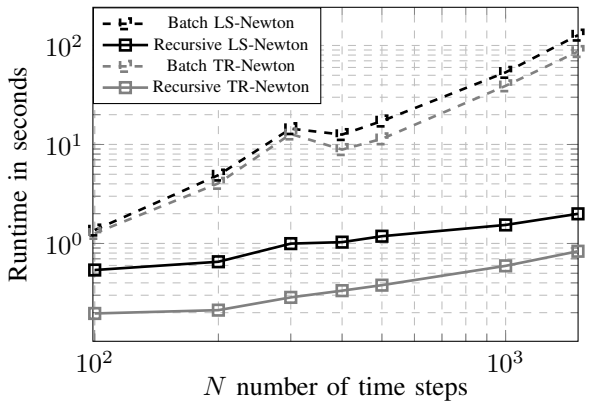


Fig. 2. Runtime comparison of the batch Newton method against the recursive trust-region (TR) and line-search (LS) Newton algorithms.

## IV. Experimental Results

In this section, we assess the performance of the proposed approaches using a simulated coordinated turn model example with bearings-only measurements [12], [15], [19]. The system has a 5-dimensional state vector $\mathbf{x} = [p_x, p_y, \dot{p}_x, \dot{p}_y, \omega]^\top$ which describes the $x-y$ position, the $x-y$ velocity, and the turn rate of the target. The bearing is measured by two sensors located at known positions. Figure 1 depicts an example true trajectory, an estimated trajectory using a trust-region Newton-IKS, and the locations of the two sensors.

In addition to our recursive algorithms, we implement the equivalent batch optimization techniques as presented in [13] and use the same hyperparameters in the line-search and trust-region variants. We focus on comparing the computational complexity of the recursive and batch techniques. We rely on JAX [22] for automatic differentiation.

In this study, we investigate trajectories of different lengths, ranging from $N = 100$ to $N = 1500$, and report the average runtime (over 20 runs) of running 30 overall iterations of the iterated batch and recursive Newton methods. The average runtime as a function of trajectory length is illustrated in Figure 2. As expected, the computational performance of the recursive Newton algorithms is superior to their batch counterpart in terms of runtime. An open-source implementation is available at https://github.com/hanyas/second-order-smoothers.

## V. Conclusion

We presented a computationally efficient realization of Newton's method for smoothing in nonlinear state-space models with additive noise. We leveraged automatic differentiation tools to compute the required first- and second-order derivatives with minimal effort and formulated a corresponding affine state-space model with augmented pseudo measurements. We showed that this modified SSM form enables the implementation of a recursive, computationally favorable Kalman smoothing algorithm equivalent to a Newton step. Furthermore, We proposed line-search and trust-region extensions of the proposed method to ensure the convergence to a local optimum. Finally, we empirically validated the efficiency of our recursive Newton method against standard batch solutions.

## References


[1] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[2] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley, 2001.

[3] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1995.

[4] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME journal of Basic Engineering*, 1960.

[5] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, 1965.

[6] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[7] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *Transactions on Automatic Control*, 2000.

[8] B. M. Bell, "The iterated Kalman smoother as a Gauss–Newton method," *SIAM Journal on Optimization*, 1994.

[9] Á. F. García-Fernández, L. Svensson, M. Morelande, and S. Särkkä, "Posterior linearization filter: Principles and implementation using sigma points," *IEEE Transactions on Signal Processing*, 2015.

[10] Á. F. García-Fernández, L. Svensson, and S. Särkkä, "Iterated posterior linearization smoother," *IEEE Transactions on Automatic Control*, 2017.

[11] F. Tronarp, A. F. García-Fernández, and S. Särkkä, "Iterative filtering and smoothing in nonlinear and non-Gaussian systems using conditional moments," *IEEE Signal Processing Letters*, 2018.

[12] S. Särkkä and L. Svensson, "Levenberg-Marquardt and line-search extended Kalman smoothers," in *International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020.

[13] S. Wright and J. Nocedal, *Numerical Optimization*. Springer, 1999.

[14] Á. F. García-Fernández and L. Svensson, "Gaussian MAP filtering using Kalman optimization," *Transactions on Automatic Control*, 2014.

[15] M. Fatemi, L. Svensson, L. Hammarstrand, and M. Morelande, "A study of MAP estimation techniques for nonlinear filtering," in *International Conference on Information Fusion*. IEEE, 2012.

[16] H. Moriyama, N. Yamashita, and M. Fukushima, "The incremental Gauss-Newton algorithm with adaptive stepsize rule," *Computational Optimization and Applications*, 2003.

[17] B. M. Bell and F. W. Cathey, "The iterated Kalman filter update as a Gauss–Newton method," *Transactions on Automatic Control*, 1993.

[18] J. Humpherys, P. Redd, and J. West, "A fresh look at the Kalman filter," *SIAM Review*, 2012.

[19] M. A. Skoglund, G. Hendeby, and D. Axehill, "Extended Kalman filter modifications based on an optimization view point," in *International Conference on Information Fusion*. IEEE, 2015.

[20] Y. Ollivier, "The extended Kalman filter is a natural gradient descent in trajectory space," *arXiv preprint arXiv:1901.00696*, 2019.

[21] K. Madsen and H. B. Nielsen, *Introduction to Optimization and Data Fitting*, 2008.

[22] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, and S. Wanderman-Milne, "JAX: Composable transformations of Python+NumPy programs," http://github.com/google/jax, 2018.