# Fast Robust Methods for Singular State-Space Models

Jonathan Jonker [1,a], Aleksandr Aravkin [b], James Burke [a], Gianluigi Pillonetto [d], Sarah Webster [c].

[a] *Department of Mathematics, University of Washington*

[b] *Department of Applied Mathematics, University of Washington*

[c] *Applied Physics Lab, University of Washington*

[d] *Department of Information Engineering, University of Padova*

**Abstract**

State-space models are used in a wide range of time series analysis applications. Kalman filtering and smoothing are work-horse algorithms in these settings. While classic algorithms assume Gaussian errors to simplify estimation, recent advances use a broad range of optimization formulations to allow outlier-robust estimation, as well as constraints to capture prior information.

Here we develop methods on state-space models where either transition or error covariances may be singular. These models frequently arise in navigation (e.g. for 'colored noise' models or deterministic integrals) and are ubiquitous in auto-correlated time series models such as ARMA. We reformulate all state-space models (singular as well as nonsingluar) as constrained convex optimization problems, and develop an efficient algorithm for this reformulation. The convergence rate is *locally linear*, with constants that do not depend on the conditioning of the problem.

Numerical comparisons show that the new approach outperforms competing approaches for *nonsingular* models, including state of the art interior point (IP) methods. IP methods converge at superlinear rates; we expect them to dominate. However, the steep rate of the proposed approach (independent of problem conditioning) combined with cheap iterations wins against IP in a run-time comparison. This suggests that the proposed approach can be a *default choice* for estimating state space models outside of the Gaussian context for singular and nonsingular models. To highlight the capabilities of the new framework, we focus on navigation applications that use singular process covariance models. The methods have been implemented in an open source Python code.

## 1 Introduction

The linear state space model is widely used in tracking and navigation [9], control [1], signal processing [2], and other time series [15,24]. The model assumes linear relationships between latent states with noisy observations:

$$\begin{aligned} x_1 &= x_0 + w_1 \\ x_k &= G_k x_{k-1} + w_k, \quad k = 2, \ldots, N \\ y_k &= H_k x_k + v_k, \quad k = 1, \ldots, N, \end{aligned} \tag{1}$$

where $x_0$ is a given initial state estimate, $x_1, \ldots, x_N$ are unknown latent states with known linear process models $G_k$, and $y_1, \ldots, y_N$ are observations obtained using known linear models $H_k$.

The errors $w_k$ and $v_k$ are assumed to be mutually independent random variables with known covariances $Q_k$ and $R_k$. In tracking and navigation, the end goal is the estimation of the latent states $\{x_k\}$. In autocorrelated time series models (e.g. Holt-Winters c.f. [15], ARMA c.f. [24]), estimating the state is a necessary step to estimating additional parameters on which $G_k$, $H_k$, $Q_k$ and $R_k$ may depend. In both settings, estimating the state sequence $\{x_k\}$ efficiently is essential.

**Singular Covariances**. We are particularly interested in models where $Q_k$ and $R_k$ may be singular. These models arise in all settings where state-space formulations are used. The classic Kalman filter [17] and RTS smoother [22] assume that $w_k, v_k$ are Gaussian, and find the minimum variance estimates of the state, conditioned on the observations [2]. More generally, the RTS smoother finds the linear minimum variance estimator. This procedure is well defined for singular covariances $Q_k$ and $R_k$, and the smoother can be derived as a se-

*Email addresses:* jonkerjo@uw.edu (Jonathan Jonker), saravkin@uw.edu (Aleksandr Aravkin), jvburke@uw.edu (James Burke), giapi@dei.unipd.it (Gianluigi Pillonetto), swebster@apl.washington.edu (Sarah Webster).

[1] The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

quence of least squares projections [3]. However, when the noise is not Gaussian (e.g. in the presence of outliers), these estimates are not satisfactory; and far better estimates can be obtained through a maximum a posteriori (MAP) estimator [4]. Implementing a general MAP estimator for singular covariances requires a new approach.

**General Kalman Smoothing**. Classic Gaussian formulations fail when outliers are present in the data, are unable to track abrupt state changes, and cannot incorporate side information through constraints. To develop effective approaches in these cases, generalized Kalman smoothing formulations have been proposed in the last few years, see [4] and the references within. The conditional mean is no longer tractable to compute these estimates, and *maximum likelihood* (ML) formulations are much more natural. The general form of Kalman smoothing considered in [4] is given by

$$\min_{x\in X} \sum_{i=0}^{n} \rho_1(Q_k^{-1/2}(x_k - G_k x_{k-1})) + \rho_2(R_k^{-1/2}(y_k - H_k x_k)), \tag{2}$$

where $\rho_1, \rho_2$ are convex penalties, and $x \in X$ is a set of state-space constraints. In the nonsingular Gaussian case, approach (2) reduces to the classic problem solvable by RTS or Mayne-Fraser algorithms [4]. Additional extensions include models with nonlinear inequality constraints [11] and robust nonlinear models [5,7].

**Contribution**. We develop a new reformulation to extend (2) to singular covariance models $Q_k$ and $R_k$, and implement a primal-dual algorithm using Douglas-Rachford splitting (DRS) to solve this reformulation.

We analyze the DRS for the singular reformulation, and show that its convergence rate does not depend on the conditioning of the system. Even when the model is nonsingular, the new approach is much faster than first-order and second-order methods for (2) for synthetic examples. The advantage increases as the models become more ill-conditioned; however the *local* linear rate means that initialization becomes very important.

The paper proceeds as follows. In Section 2 we discuss prior approaches to singular models. In Section 3, we develop a constrained reformulation of (2), building on early work of [21] for singular least squares. In Section 4, we show how to efficiently optimize a wide range of singular smoothing problems using DRS. The algorithm we use has a *local linear rate of convergence* for any piecewise linear-quadratic penalties $\rho_1, \rho_2$ in (2), and each iteration is efficiently and stably computed by exploiting dynamic problem structure. We compare the new algorithm to first-order methods, L-BFGS, and IP-solve, a toolbox specifically developed for PLQ Kalman smoothing (for nonsingular formulations).

## 2 Related Work

Several approaches in the literature deal with singular models. We give a brief description and references for each. To ground the discussion, consider tracking a particle moving along a smooth path in space, where state comprises velocity and position. Singular models arise naturally in this situation. We can model velocity as subject to error, and position as a deterministic integral:

$$\begin{aligned} x_{k+1} &= x_k + \Delta t \dot{x}_k \\ \dot{x}_{k+1} &= \dot{x}_k + \epsilon_k. \end{aligned} \tag{3}$$

Here, the process covariance matrix $Q_k$ has rank one.

**Using the original Kalman filter.** In the linear Gaussian setting, the original Kalman filter does not require $Q$ and $R$ to be invertible. Applying the Kalman filter (and RTS smoother) will return the minimum variance estimate for singular innovation/measurement errors [2]. This result does not extend to the general context of (2), with robust losses, sudden changes, and constraints.

**Changing the model.** A common approach in the standard setting is to modify the model to make $Q_k, R_k$ nonsingular. Treating (3) as a discretization of a stochastic differential equation (SDE), many authors opt for a nonsingular error model [8,10,16,20]

$$Q_k = \begin{bmatrix} \Delta t_k & \Delta t_k^2/2 \\ \Delta t_k^2/2 & \Delta t_k^3/3 \end{bmatrix},$$

derived by computing the variance of a discretized process noise term. The approach has limitations for navigation models with high-dimensional states driven by low-dimensional errors. The low-dimensional error structure should simplify estimation, but instead this approach introduces full-dimensional and ill-conditioned $Q_k$. In addition, making $Q_k$ nonsingular is antithetical to state-space formulations for models such as ARMA, which use singularity to enforce auto-regressive constraints.

**Pseudo-inverse with orthogonality constraints.** The formulation that is closest to ours is that of [19], which replaces the inverse of $Q_k$ by its pseudo-inverse, and adds orthogonality constraints. With potentially singular $Q_k$ and $R_k$, the maximum likelihood estimate for the Gaussian/LS model can be formulated as

$$\begin{aligned} \min_x \sum_k ||Q_k^{\dagger/2}(x_k - G_k x_{k-1})||^2 + ||R_k^{\dagger/2}(y_k - H_k x_k)||^2 \\ \text{s.t. } Q_k^{\perp}(x_k - G_k x_{k-1}) = 0, \quad R_k^{\perp}(y_k - H_k x_k) = 0 \\ \text{for all } k = 1, \ldots, N, \end{aligned} \tag{4}$$

see [4, Appendix A]. This requires computing both the pseudo-inverse and orthogonality constraints.

**Constrained reformulation.** The reformulation we choose was first used by Paige [21]. Given the singular least squares problem

$$\min_x \|Q^{\dagger/2}(Ax - b)\|^2 \quad \text{s.t.} \quad Q^{\perp}(Ax - b) = 0,$$

we can instead write it as

$$\min_{x,u} \|u\|^2 \quad \text{s.t.} \quad \mathrm{Q}^{1/2}u = Ax - b. \tag{5}$$

It is easy to see (4) and (5) are equivalent; the latter is more elegant, and only requires computing a root of $Q$, rather than using both $Q$ and $Q^\dagger$. When $Q$ is invertible, we can eliminate $u$ from both formulations and reduce to a least squares problem in $x$. Splitting the affine constraint from the original penalty has theoretical and practical advantages for general Kalman smoothing, as shown in the next sections.

## 3 General Singular Kalman Smoothing

Following the ideas proposed by [19], we introduce variables $u_k$ for the normalized process innovations and $t_k$ for the normalized residuals. We also introduce a penalty $\rho_3$ for the states. In the examples we consider, $\rho_3$ is an indicator function for the known feasible regions $X_k$:

$$\rho_3(x_k) = \begin{cases} 0 & x_k \in X_k \\ \infty & x_k \notin X_k \end{cases}.$$

The reformulated singular Kalman smoothing problem is given by

$$\begin{aligned} \min_{u,t,x} & \sum_{k=1}^{N} \rho_1(u_k) + \rho_2(t_k) + \rho_3(x_k) \\ \text{s.t.} \quad & Q_k^{1/2}u_k = G_k x_{k-1} - x_k \\ & R_k^{1/2}t_k = y_k - H_k x_k \end{aligned}. \tag{6}$$

This problem is equivalent to (2) when $Q_k$ and $R_k$ are nonsingular. For singular models, (6) requires only that roots $Q^{1/2}$ and $R^{1/2}$ are available.

**Structure-preserving Reformulation.** We now rewrite (6) into a more compact form. Define

$$\begin{aligned} D_i &= \begin{pmatrix} Q_i^{1/2} & 0 & I \\ 0 & R_i^{1/2} & H_i \end{pmatrix} \text{ for } i = 1, \dots N, \\ B_j &= \begin{pmatrix} 0 & 0 & -G_{j+1} \\ 0 & 0 & 0 \end{pmatrix}, \text{ for } j = 1, \dots, N-1, \end{aligned} \tag{7}$$

and let

$$A = \begin{pmatrix} D_1 & 0 & \dots & 0 \\ B_1 & D_2 & 0 & \vdots \\ 0 & \ddots & \ddots & 0 \\ 0 & 0 & B_{N-1} & D_N \end{pmatrix}. \tag{8}$$

Define also

$$\begin{aligned} z^T &= \begin{pmatrix} u_1^T & t_1^T & x_1^T & \dots u_N^T & t_N^T & x_N^T \end{pmatrix} \\ \hat{w}^T &= \begin{pmatrix} x_0^T & y_1^T & 0 & y_2^T & \dots & 0 & y_N^T \end{pmatrix}. \end{aligned} \tag{9}$$

Now we can write (6) compactly as

$$\begin{aligned} &\min_z \quad \rho(z) \quad \text{s.t. } Az = \hat{w}, \\ &\rho(z) = \sum_{k=1}^{N} \rho_1(u_k) + \rho_2(t_k) + \rho_3(x_k). \end{aligned} \tag{10}$$

The order of the blocks in $z$ is chosen so that the constraint matrix $A$ in (8) lower block bi-diagonal.
The constraint $Az = \hat{w}$ raises a natural question: when is a singular Kalman smoothing model solvable? Clearly we want $\hat{w} \in \mathrm{Ran}(A)$, but we want this condition to hold for any realization of the data $\hat{w}$, so we want to know when $A$ is surjective. We can characterize this condition precisely in terms of a simple conditions on the individual blocks $R_i, Q_i, H_i$.

**Theorem 3.1 (Surjectivity of $A$)** *The following are equivalent.*

*(1) $A$ is surjective.*
*(2) Each block $D_i$ is surjective.*
*(3)* $\mathrm{null}\left(\begin{bmatrix} Q_i^{1/2} & 0 \\ 0 & R_i^{1/2} \end{bmatrix}\right) \subset \mathrm{Ran}\left(\begin{bmatrix} I \\ H_i \end{bmatrix}\right)$ *for all $i$.*
*(4) $R_i + H_i \left(I - (Q_i + I)^{-1}\right) H_i^T$ is invertible for all $i$.*

The proof is given in the Appendix.

**Remark 3.2** *In practice it is not strictly necessary for $A$ to be surjective, in fact if $y_i \in \mathrm{Ran}(H_i)$ for all $i$, then $\hat{w} \in \mathrm{Ran}(A)$. It is also worth noting that if the data is not in the range of the measurement maps, a sparse penalty can be used instead and the same convergence rates would hold.*

## 4 Douglas-Rachford Splitting for Problem (10)

Consider problem (10) as a sum of two functions, $\rho + g$, with $\rho$ as in (10) and $g$ the indicator function of the affine constraint $Az = \hat{w}$:

$$g(z) = \begin{cases} 0 & Az = \hat{w} \\ \infty & Az \neq \hat{w} \end{cases}. \tag{11}$$

Douglas-Rachford splitting (DRS) is a classic algorithm for this problem. For a convex function $f$, define the proximity operator (see e.g. [12]) as

$$\mathrm{prox}_{\alpha f}(\zeta) = \arg\min_x \frac{1}{2\alpha}\|\zeta - x\|^2 + f(x).$$

The DRS algorithm for (10) is detailed in Algorithm 1. For more on splitting methods and their convergence rates see the survey [13].

**Algorithm 1** Douglas-Rachford Splitting (DRS)

**Require:** Initialize at any $z^0$, $\zeta^0$.
1: **loop**
2: $z^k = \text{prox}_{\tau g}(z^{k-1} - \tau\zeta^{k-1})$
3: $\zeta^k = \text{prox}_{\sigma\rho^*}(\zeta^{k-1} + \sigma(2z^k - z^{k-1}))$
**return** $z^k$

Implementing DRS in our case requires computing two proximity operators at each iteration. One proximity operator is $\text{prox}_{\rho^*}$, where $\rho^*$ denotes the *convex conjugate*:

$$\rho^*(y) := \sup_x \langle y, x\rangle - \rho(x).$$

The prox of a function is related to the prox of its conjugate by Moreau's decomposition:

$$\text{prox}_\rho(x) + \text{prox}_{\rho^*}(x) = x.$$

Thus it suffices to compute $\text{prox}_\rho$. The function $\rho$ captures all user-supplied models, including losses used process and measurement transitions, as well as penalties or constraints on the state, $\rho_1, \rho_2$ and $\rho_3$. The proximity operators of these individual elements must be provided; then $\text{prox}_\rho$ is a stack of these input functions. Proximity operators for many common functions are easily available [12], and we include a small library with our implementation [2].

The second proximity operator is $\text{prox}_g$, which is independent of user choice for process, measurement, and prior models:

$$\begin{aligned}\text{prox}_g(\eta) &= \arg\min_{Az=\hat{w}} \frac{1}{2}\|\eta - z\|^2 \\ &= (I - A^\dagger A)\eta + z_p,\end{aligned}$$

where $z_p$ is any particular solution to $Az = \hat{w}$. In practice, we can apply the operator quickly by solving a simple quadratic with affine constraints, with optimality conditions given by

$$\begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} z \\ \nu \end{bmatrix} = \begin{bmatrix} \eta \\ \hat{w} \end{bmatrix}.$$

There are many ways to solve this system. We opt to reduce the problem to solving a block tridiagonal system:

$$\begin{bmatrix} I & A^T \\ 0 & AA^T \end{bmatrix} \begin{bmatrix} z \\ \nu \end{bmatrix} = \begin{bmatrix} \eta \\ A\eta - \hat{w} \end{bmatrix}$$

[2] https://github.com/UW-AMO/KalmanJulia.

We solve $AA^T\nu = A\eta - \hat{w}$, then back-substitute to get the optimal $z$. The system $AA^T$ does not change over iterations; only the right hand side changes. We can therefore compute a single factorization, then use it in each iteration. Since $A$ is block bidiagonal (8), $AA^T$ is block tridiagonal; when $A$ is surjective, $AA^T$ is nonsingular, and we can find a lower block diagonal Cholesky factorization $L$ with $LL^T = AA^T$:

$$AA^T = \begin{bmatrix} a_1 & b_1^T & & \\ b_1 & a_2 & b_2^T & \\ & b_2 & a_3 & b_3^T \\ & & b_3 & a_4 \end{bmatrix}, \quad L = \begin{bmatrix} c_1 & & & \\ d_1 & c_2 & & \\ & d_2 & c_3 & \\ & & d_3 & c_4 \end{bmatrix} \tag{12}$$

The factorization is obtained by a simple variant of the block-forward Thomas algorithm, with $O(n^3N)$ arithmetic operations required. Once $L$ has been precomputed, we need only $O(n^2N)$ arithmetic operations to solve $LL^T\nu = Ac - \hat{w}$ for any right hand side. This is the same complexity as that of a matrix-vector multiply with $A$.

**Local Linear Rate.** When $\rho$ is piecewise linear-quadratic [23,6], the DRS algorithm converges locally linearly to a solution, see Figure 4. More precisely, there is a real number $R > 0$ such that if $||\eta^K - \eta^*|| < R$ then there is a constant $\kappa$ with $0 < \kappa < 1$ such that for all $k > K$,

$$\|\eta^{k+1} - \eta^*\| < \kappa\|\eta^k - \eta^*\|,$$

where $\eta = \begin{bmatrix} z & \zeta \end{bmatrix}^T$, is the primal and dual pair.

**Theorem 4.1** *Algorithm 1 converges with a locally linear rate.*

*Proof:* Following the proof technique of [18, Theorem 5], Algorithm 1 has a local linear convergence rate if the following two conditions are satisfied:

(1) Algorithm 1 can be written as the action of a nonlinear operator satisfying a regularity property (see Lemma 4.2).
(2) The functions $g, \rho$ are *subdiffererentially metrically subregular* [3].

We show that these conditions hold for Algorithm 1. Define

$$Dx \mapsto \begin{bmatrix} \partial g(z) \\ \partial\rho^*(\zeta) \end{bmatrix}, \quad M = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \quad H = \begin{bmatrix} \frac{1}{\tau}I & 0 \\ -2I & \frac{1}{\sigma}I \end{bmatrix}.$$

[3] A mapping $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ is called *metrically subregular* at $\bar{x}$ for $\bar{y}$ if $(\bar{x}, \bar{y}) \in \text{graph } F$ and there exists $\eta \in [0, \infty)$, neighborhoods $\mathcal{U}$ of $\bar{x}$, and $\mathcal{Y}$ of $\bar{y}$ such that

$$d(x, F^{-1}\bar{y}) \leq \eta d(\bar{y}, Fx \cap \mathcal{Y}), \quad \forall x \in \mathcal{U}$$

Define the nonlinear operator $T$ by

$$T = (H + D)^{-1}(H - M). \tag{13}$$

$T$ captures the iteration in Algorithm 1, which can be written as $\eta^k = T\eta^{k-1}$, for $\eta = \left[z^T, \zeta^T\right]^T$. Then we have the following lemma.

**Lemma 4.2** *Suppose that $\tau, \sigma < 1$. Then*

$$||T\eta - \eta||^2_{H-M} \leq \langle \eta^* - \eta, (H - M)(T\eta - \eta)\rangle$$

*where $\eta^*$ is such that $0 \in (D + M)\eta^*$.*

The proof is given in the Appendix.

## 5 Navigation Models

Autonomous navigation requires high-fidelity tracking using occasional GPS and frequent depth/height, gyro-compass, and linear acceleration data. Gyro, compass, and linear acceleration are readily available from inertial measurement units (IMUs).

In this section, we develop a simple kinematic model that is trivially applicable to any vehicle, and particularly useful for applications such as survey work where accelerations are heavily damped and autonomous vehicles often travel in long straight lines. When the attitude is known or changing slowly, the model can be linearized effectively and the situation simplifies considerably.

**Linear Singular Navigation Model.** For a vehicle that is well-instrumented in attitude, the uncertainty in position (and the x-y states in particular) is typically orders of magnitude larger than the uncertainty in attitude. We simplify the vehicle process model to track only position states $(x, y, z)$, while assuming that the attitude states $(r, p, h)$ are directly available from the most recent sensor measurements. To make the model linear, the position and its derivatives are referenced to the local-level frame.

To incorporate linear acceleration measurements from an inertial measurement unit (IMU), we must track both linear velocities and linear acceleration in the state vector. This leads to the augmented state

$$x_s = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \ddot{x}, \ddot{y}, \ddot{z}]^\top. \tag{14}$$

The linear kinematic process model is given by

$$\dot{x}_s = \underbrace{\begin{bmatrix} 0 & I & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{bmatrix}}_{F_s} x_s + \underbrace{\begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}}_{G_s} w_s, \tag{15}$$

where $w_s \sim \mathcal{N}(0, Q_s)$ is zero-mean Gaussian noise.

The linear process model (15) is usually discretized using a Taylor series:

$$x_{s_{k+1}} = F_{s_k} x_{s_k} + w_{s_k} \tag{16}$$

$$F_{s_k} = e^{F_s T} = I + F_s T + \frac{1}{2!}F_s^2 T^2 + \cancelto{0}{\frac{1}{3!}F_s^3 T^3} + \cdots$$

$$= \begin{bmatrix} I & IT & \frac{1}{2}IT^2 \\ 0 & I & IT \\ 0 & 0 & I \end{bmatrix}$$

where the higher order terms are *identically zero* because of the structure of $F_s$, resulting in a simple closed-form solution for $F_{s_k}$. The discretized process noise

$$w_{s_k} = \int_0^T e^{F_s(T-\tau)} G_s w_s(\tau) d\tau, \tag{17}$$

is a zero-mean Gaussian, with covariance given by

$$Q_{s_k} = \begin{bmatrix} \frac{1}{3}T^3 & \frac{1}{2}T^2 & 0 \\ \frac{1}{2}T^2 & T & 0 \\ 0 & 0 & 0 \end{bmatrix} Q_s, \tag{18}$$

Model (18) forces the acceleration to be 0 across the entire model because the lower right corner is set to 0. As a result, the initialized track can be biased away from the data by a fixed velocity, obtained by finding the slope from the most recent position data.

Instead, we model the covariance as if the error were the next term in the Taylor series approximation, a technique suggested by [8]. More precisely we set covariance to be the outer product, $\Gamma^T\Gamma$ where

$$\Gamma = \begin{bmatrix} \frac{1}{3!}IT^3 & \frac{1}{2!}IT^2 & IT \end{bmatrix}$$

This leads to a rank 3 covariance for a $9 \times 9$ matrix for a model that comprises position, velocity, and acceleration in 3D space.

**Measurement Models for the IMU.** Any navigation system that relies on an IMU needs occasional measurements that inform the position (e.g. GPS), otherwise the error in position estimates grows without bound. We are given these data from a separate source, sampled at a lower update rate than that of the IMU. For any $s$ where such data is available, we have the measurement model

$$H_s = \begin{bmatrix} I_{3\times 3} & 0_{3\times 6} \\ 0_{3\times 6} & R(\varphi) \end{bmatrix}, \quad z_s = \begin{bmatrix} b^\top & \ddot{x}_{\text{meas}} & \ddot{y}_{\text{meas}} & \ddot{z}_{\text{meas}} \end{bmatrix}^\top.$$

with $R(\varphi)$ a known coordinate transform based on instrumented roll, pitch, and yaw. If there is no position

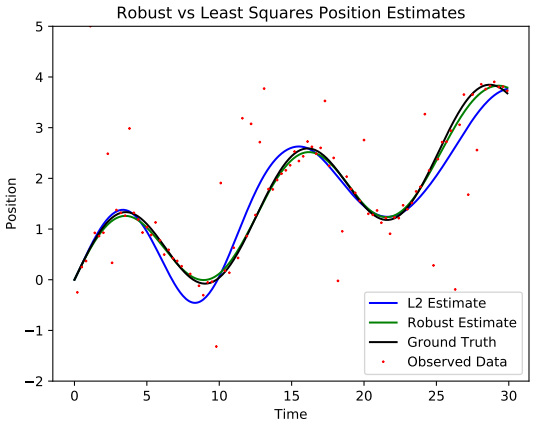


Fig. 1. Position estimate for both robust and standard penalties

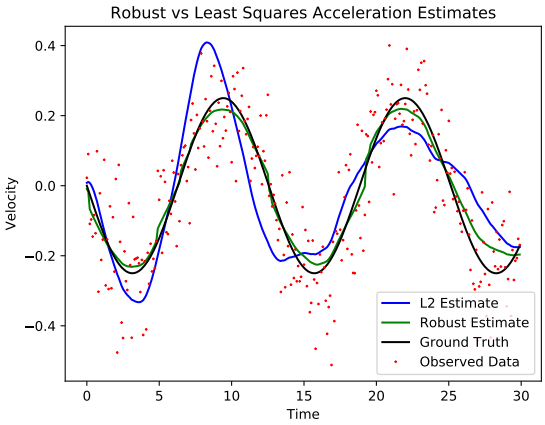


Fig. 2. Acceleration estimate for both robust and standard penalties

data measured at time $s$ then we use the model

$$H_s = \begin{bmatrix} 0_{3\times 3} & 0_{3\times 6} \\ 0_{3\times 6} & R(\varphi) \end{bmatrix}, \quad z_s = \begin{bmatrix} 0 & \ddot{x}_{\text{meas}} & \ddot{y}_{\text{meas}} & \ddot{z}_{\text{meas}} \end{bmatrix}^\top.$$

The covariance used for measurement data depends on whether there was position data available:

$$R_s = \begin{bmatrix} 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & r_s I_{3\times 3} \end{bmatrix} \quad \text{or} \quad R_s = \begin{bmatrix} U_s & 0_{3\times 3} \\ 0_{3\times 3} & r_s I_{3\times 3} \end{bmatrix}$$

where the top $3 \times 3$ block is either 0 (position not available) or $U_s$, a diagonal matrix reflecting position uncertainty (position is available). The scalar $r_s$ models uncertainty in IMU measurements.

**Synthetic Tracking Example.** This example is implemented using an open source Python codebase that implements the general singular smoother [4]. To keep the presentation brief, we use the models described above to track a particle moving along the path

$$y(t) = \sin(t/2) + t/10$$

with $\Delta t = 0.1$. At every time point we observe acceleration and every 1 second observe position, both with small Gaussian errors ($\mathcal{N}(0,1)$). With probability $p = 0.3$, the observed position is further corrupted by large errors (outliers) by drawing from a $\mathcal{N}(0,4)$ distribution. We test the model with both standard and robust penalties. The position estimates plotted against observed data are shown in Figure 1 and acceleration estimates along with data are shown in Figure 2.

When the data is corrupted, the results for the (singular) robust formulation using the huber loss [14,4] are much closer to the synthetic ground truth than the solutions of the singular least squares model.

[4] https://github.com/jonkerjo/Singular_Kalman_Python

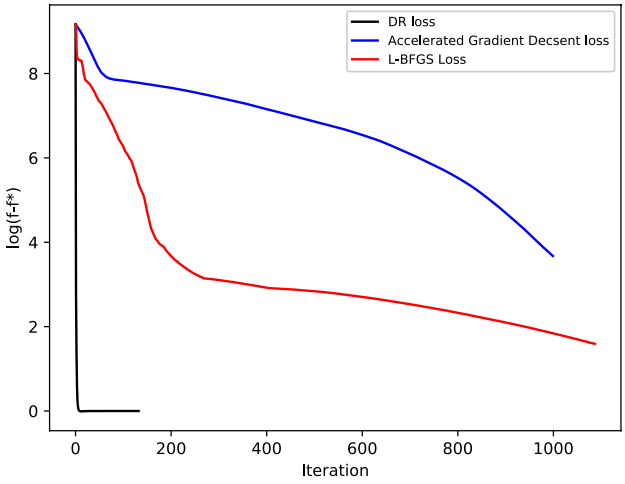


Fig. 3. Objective vs. iteration counts of Algorithm 1 for (10) (black), vs. accelerated gradient descent (AGD) (blue) and L-BFGS (red) for (2). Both $\rho_1$ and $\rho_2$ are Huber losses, with $\rho_3 \equiv 0$, $N = 200, n = 2$ and $Q, R$ nonsingular, so (10) and (2) are equivalent. All iterations require $O(n^2 N)$ operations. DRS splitting is much faster than methods with linear convergence rates and similar iteration complexity.

**Comparison on Smooth Nonsingular Problems.** If the covariances, $Q, R$ are non-singular and the penalties $\rho_{1,2}$ are $\mathcal{C}^1$-smooth, then the Kalman smoothing problem can be written as a smooth convex problem. In this case the same reformulation will work and Algorithm 1 will still give a local linear rate. However more common algorithms such as gradient descent and L-BFGS can also be applied. We compare the performance of these three algorithms to track a particle moving along a smooth path with $N = 200$ and $n = 2$. We use nonsingular $Q_k$, and Huber penalty functions.

As seen in Figure 3, Algorithm 1 for (10) converges

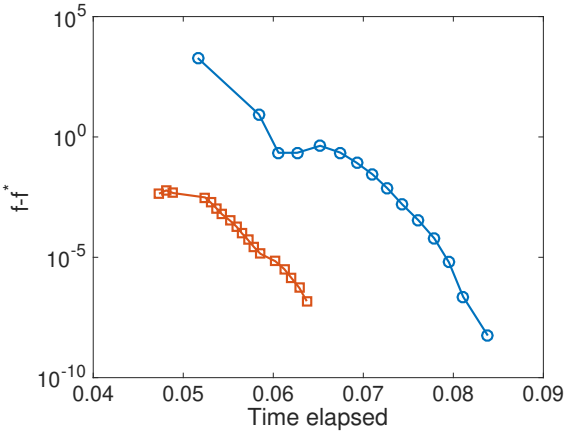


Fig. 4. Timed run of Algorithm 1 vs. IPsolve for the same setup as presented in Figure 3. At this scale, we see the locally linear convergence rate of the DRS. Even though IPsolve has superlinear rate, DRS wins because the slope of the rate is very steep, and each iteration is fast. By the time DRS is done, IPsolve has had time for only taken a few iterations.

far faster than either accelerated gradient descent or LBFGS method on the equivalent nonsingular smoothing formulation (2) . This is because its convergence rate does not depend on the condition number of the matrix $A$, so each iteration makes good progress while keeping the number of arithmetic operations in each iteration at $O(n^2N)$. This is the same order as for the matrix-vector multiply needed for a gradient evaluation, given that the sparse block tridiagonal matrix $AA^T$ is factorized at the start of the algorithm.

We also compare with the second-order interior point method, implemented in the IPsolve package [5] . Use-cases and performance of IPsolve for nonsingular Kalman smoothing is discussed in [4]. The results are shown in Figure 4, where IPsolve and DRS for the equivalent reformulation are compared for the nonsingular Huber model. Even though DRS has at best a linear rate, the constants are very good, as they do not depend on the conditioning of the Kalman smoothing problem. The other advantage is that DRS can use a pre-factorized matrix, while IPsolve has to solve a modified linear system every time; there is no simple strategy to pre-factor as with DRS.

The numerical experiments suggest that Algorithm 1 should be used regardless of whether $Q$ and $R$ are singular or not. In the next section, we focus on a rich class of singular noise models found in navigation.

## 6 Conclusion.

We propose a singular Kalman smoothing framework that can use singular covariance models for process and measurements, convex robust losses, and state-space constraints. The modeler can use any convex loss that has an implementable prox; in particular any piecewise linear-quadratic loss and simple polyhedral constraint can be used. Future work will consider real-time implementation, as well as extensions to nonlinear models. Numerical experiments illustrate that the local linear rate we have in theory requires a good initialization in practice. All experiments in the paper were initialized by propagating the state estimate forward; this worked far better than an arbitrary initialization (e.g. at the 0 vector). Smarter initialization can be developed for streaming/online contexts, where recent estimates play a key role in initializing smoothing subproblems.

[5] https://github.com/UW-AMO/IPsolve.

## Appendix

### 6.1 *Proof of Theorem 3.1*

Conditions $2, 3, 4$ are easily seen to be equivalent. To see that 2 and 3 are equivalent, note that the matrix

$$\begin{bmatrix} Q_i^{1/2} & 0 \\ 0 & R_i^{1/2} \end{bmatrix}$$

is symmetric, so its nullspace is perpendicular to its range. Therefore surjectivity of $D_i$ is equivalent to the condition that the range of $\begin{bmatrix} I \\ H_i \end{bmatrix}$ covers this nullspace.

To see the equivalence of 2 and 4, recall that $B$ is surjective if and only if $BB^T$ is invertible, so $D_i$ is surjective exactly when the matrix

$$\begin{bmatrix} Q_i + I & H_i^T \\ H_i & R_i + H_i H_i^T \end{bmatrix}$$

is invertible. $Q_i + I$ is always invertible, so invertibility of the block $2 \times 2$ matrix is equivalent to the invertibility of the Schur complement $R_i + H_i \left(I - (Q_i + I)^{-1}\right) H_i^T$. It remains to show that conditions 1 and 2 are equivalent. We proceed by induction on $N$. The base case is trivial, since for $N = 1$, $A = D_1$. For the inductive case, consider that for $N = k$ the result holds, and write the $N = k+1$ case as

$$\begin{bmatrix} A_k & 0 \\ [0 & B_k] & D_{k+1} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

and assume that $A_k$ is surjective. We then know that there exists $z_1$ that satisfies $A_k z_1 = w_1$. The second row can now be written explicitly as

$$D_{k+1} z_2 = w_2 + G_{k+1} x_k,$$

where $x_k$ is the last component of $z_1$. Thus $A_{k+1}$ is surjective exactly when $D_{k+1}$ is, as desired.

*6.2 Proof of Lemma 4.2.*

*Proof: This proof is similar to that of [18] but included in full detail here to emphasize independence from the conditioning of the system.*

*As $D$ is monotone we have*

$$\langle \eta^* - T\eta, D\eta^* - DT\eta \rangle \geq 0$$

*as $0 \in (D+M)\eta^*$ this implies*

$$\langle \eta^* - T\eta, -M\eta^* - DT\eta \rangle \geq 0$$

*Now $DT\eta = DT\eta + HT\eta - HT\eta = (H-M)\eta - HT\eta$. Thus*

$$\begin{aligned} 0 &\leq \langle \eta^* - T\eta, -M\eta^* + HT\eta - (H-M)\eta \rangle \\ &= \langle \eta^* - T\eta, -M(\eta^* - \eta) + H(T\eta - \eta) \rangle \\ &= \langle \eta^* - \eta, -M(\eta^* - \eta) + H(T\eta - \eta) \rangle \\ &\quad + \langle \eta - T\eta, -M(\eta^* - \eta) + H(T\eta - \eta) \rangle \end{aligned}$$

*By definition of $M$ we have*

$$\langle M\eta, \eta \rangle = 0$$

*for any $\eta$. Therefore*

$$\begin{aligned} 0 &\leq \langle \eta^* - \eta, H(T\eta - \eta) \rangle + \langle \eta - T\eta, -M(\eta^* - \eta) \rangle \\ &\quad + \langle \eta - T\eta, H(T\eta - \eta) \rangle - \langle \eta - T\eta, M(T\eta - \eta) \rangle \\ &= \langle \eta^* - \eta, H(T\eta - \eta) \rangle + \langle \eta - T\eta, -M(\eta^* - \eta) \rangle - ||T\eta - \eta||^2_{H-M} \\ &= \langle \eta^* - \eta, H(T\eta - \eta) \rangle + \langle M(\eta - T\eta), \eta^* - \eta \rangle - ||T\eta - \eta||^2_{H-M} \\ &= \langle \eta^* - \eta, (H-M)(T\eta - \eta) \rangle - ||T\eta - \eta||^2_{H-M} \end{aligned}$$

## References


[1] B. D. Anderson and J. B. Moore. *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[2] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice Hall, 1979.

[3] C. F. Ansley and R. Kohn. A geometric derivation of the fixed interval smoothing algorithm. *Biometrika*, 69:486–487, 1982.

[4] A. Aravkin, J. V. Burke, L. Ljung, A. Lozano, and G. Pillonetto. Generalized kalman smoothing: Modeling and algorithms. *Automatica*, 86:63–86, 2017.

[5] A. Y. Aravkin, B. M. Bell, J. V. Burke, and G. Pillonetto. An $\ell$_1-laplace robust kalman smoother. *IEEE Transactions on Automatic Control*, 56(12):2898–2911, 2011.

[6] A. Y. Aravkin, J. V. Burke, and G. Pillonetto. Sparse/robust estimation and kalman smoothing with nonsmooth log-concave densities: Modeling, computation, and theory. *Journal of Machine Learning Research*, 14:2689–2728, 2013.

[7] A. Y. Aravkin, J. V. Burke, and G. Pillonetto. Robust and trend-following student's t kalman smoothers. *SIAM Journal on Control and Optimization*, 52(5):2891–2916, 2014.

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

[9] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation*. John Wiley & Sons, Inc., New York, 2001.

[10] B. M. Bell, J. V. Burke, and G. Pillonetto. An inequality constrained nonlinear Kalman-Bucy smoother by interior point likelihood maximization. *Automatica*, 45(1):25–33, Jan. 2008.

[11] B. M. Bell, J. V. Burke, and G. Pillonetto. An inequality constrained nonlinear kalman–bucy smoother by interior point likelihood maximization. *Automatica*, 45(1):25–33, 2009.

[12] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[13] D. Davis and W. Yin. Convergence rate analysis of several splitting schemes. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 115–163. Springer, 2016.

[14] P. J. Huber. Robust statistics. In *International Encyclopedia of Statistical Science*, pages 1248–1251. Springer, 2011.

[15] R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose. A state space framework for automatic forecasting using exponential smoothing methods. *International Journal of Forecasting*, 18(3):439–454, 2002.

[16] A. Jazwinski. *Stochastic Processes and Filtering Theory*. Dover Publications, Inc, 1970.

[17] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the AMSE - Journal of Basic Engineering*, 82(D):35–45, 1960.

[18] P. Latafat, N. Freris, and P. Patrinos. A new randomized block-coordinate primal-dual proximal algorithm for distributed optimization. *arXiv preprint arXiv:1706.02882*, 2017.

[19] H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Smoothed state estimates under abrupt changes using sum-of-norms regularization. *Automatica*, 48:595–605, 2012.

[20] B. Oksendal. *Stochastic Differential Equations*. Springer, sixth edition, 2005.

[21] C. Paige. Computer solution and perturbation analysis of generalized linear least squares problems. *Mathematics of Computation*, 33:171–183, 1979.

[22] H. E. Rauch, F. Tung, and C. T. Striebel. Maximum Likelihood estimates of linear dynamic systems. *AIAA J.*, 3(8):1145–1150, 1965.

[23] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*, volume 317. Springer, 1998.

[24] R. S. Tsay. *Analysis of financial time series*, volume 543. John Wiley & Sons, 2005.