Continuous-Time Kalman Filtering with Implicit Discrete Measurement Times

Shan Mohiuddin* and Mark L. Psiaki†
Cornell University, Ithaca, NY 14853-7501

A new approximation to a continuous-time Kalman filter has been developed, one that is useful for systems with implicitly defined measurement update times. This Kalman filter is applicable to radio navigation problems in which a state-dependent range delay must be subtracted from a measured reception time in order to compute the relevant Kalman filter measurement update time. The new filter can be implemented by incorporating three elements. The first element is a dense output numerical integration method that outputs a continuous description of the state over an interval. The second element is a finite dimensional approximation of the underlying continuous-time white process noise, for example, a finite order piecewise polynomial approximation. The third element is a new dynamic propagation/measurement update calculation that sensibly combines the implicit definition of the measurement update time with the dense output numerical integration scheme, the finite approximation of the process noise, and the statistical model of that approximation. After developing the necessary theory, the method is demonstrated in simulation for an example tracking problem.

I. Introduction

Many radio navigation tracking systems make measurements at known receiver clock times. In some cases, the measurements are made onboard the vehicle whose state is to be estimated. An example of this type of system is the tracking of a satellite orbit based on data from an onboard GPS receiver. In that scenario, the state update due to the measurements occurs at an explicitly defined time. If the receiver is remote from the vehicle whose state is to be estimated, however, the navigation filter becomes more complicated. Consider the case illustrated in Fig. 1 where a ground station measures the geometric range using a downlink signal. A particular measurement represents the length of the signal’s light path from the satellite’s position at the signal launch time to the ground station’s position at the signal reception time. The signal launch time in this scenario is unknown and must be determined iteratively during the Kalman filter’s measurement update step. The iteration solves the implicit equation that equates the propagation range $c\delta t$ to the distance between the known receiver location and the satellite’s location at the unknown signal launch time $t - \delta t$. A state prediction (and covariance) must be computed for each of those iterations. A naive strategy for producing this prediction is to numerically re-propagate the state estimate (and covariance) from time $t_k$ to each estimate of the $t - \delta t$ that is generated for each new iteration. This approach quickly becomes

*Graduate Student, Sibley School of Mechanical and Aerospace Engineering, AIAA Student Member.
†Professor, Sibley School of Mechanical and Aerospace Engineering, AIAA Associate Fellow.
computationally prohibitive, especially if the propagation algorithm evaluates detailed force models. A second problem with this approach is that in certain cases the integration step size may become so small that roundoff errors may become significant.

![Diagram](image)

**Figure 1.** A typical downlink measurement scenario. Scales are exaggerated.

A better approach is to employ a technique of state interpolation over the integration interval. Interpolation allows the integration step size to be determined independently of the measurement times by the user or by a step size control algorithm. The predicted state can be evaluated anywhere over the interval during the signal launch time iterations at the relatively cheap cost of a polynomial evaluation. The approach also has the benefit of completely decoupling the navigation filter’s state estimate times from the measurement times. Typical orbit determination Kalman filters use dense output strategies similar to the one illustrated in Fig. 2. The curved lines in this figure represent the state trajectory polynomials generated by a dense output numerical integration scheme. The state estimates are output (triangles) at the user-defined times, $t_k$, $t_{k+1}$, and $t_{k+2}$, by evaluating the most recent dense output polynomial. When a measurement (filled circle) is encountered, the filter exploits its dense output representation of the predicted state to iteratively determine the measurement update time. In Fig. 2, the multiple empty squares, which represent state polynomial evaluations at candidate signal launch times, indicate the need to determine the time of transmission.
via an iterative solution to the implicit equation. Each filled square indicates a converged solution, which occurs at a time prior to the filled circle measurement due to the subtraction of the signal propagation time. Once a converged state prediction is found, a new state estimate is generated (empty circle), and the numerical integration routine is restarted to generate a new trajectory polynomial. The process is repeated.

Figure 2. A strategy for applying an extended Kalman filter to multiple, iterated measurements over a given integration interval.

Although this approach is widely implemented, it is not free of problems, particularly when the measurement density is high. First, and rather obviously, the restarting of the integration algorithm at each measurement time wastes computation by overlapping the integration intervals. A second, and more subtle, problem involves the Kalman filter’s statistical model of the process noise and becomes important if the user is interested in obtaining realistic estimates of the process noise, as in a smoother. It is common in sampled data filtering to assume a constant random process noise between measurement sample times. In the scheme depicted in Fig. 2, however, this would result in process noise discontinuities within a dense output numerical integration interval. This situation is depicted in the zero-order hold process noise time history in the bottom panel of the figure. Such discontinuities would fundamentally violate the assumptions of the dense output numerical scheme.

This paper develops a different approach, illustrated in Fig. 3, in which multiple measurements taken over
a given integration interval are processed all at once in a manner consistent with estimating the parameters of a process noise model that is continuous over the same interval. In the new approach, the output times of the filter coincide with the integration periods; the a posteriori state estimates are calculated only at the ends of the integration intervals. Dense output state polynomials are used to generate state predictions during the measurement update signal launch time iterations. The key to making this approach work is to derive a sensible process noise model that maintains continuity and smoothness over an interval while approximating the statistics of white noise as best as possible. A continuous-time polynomial description of the process noise over each integration interval is proposed, as depicted in the bottom panel of Fig. 3. The polynomial coefficients are modeled as discrete-time variables drawn from zero-mean white-noise sequences in a way that optimally approximates continuous-time white noise. This update procedure causes discontinuous transitions only at the state output times $t_k, t_{k+1}$, etc., and it also involves the estimation of the process noise polynomial coefficients in a partial smoothing step.

![Figure 3](image-url)

**Figure 3.** A modified strategy for applying an extended Kalman filter to multiple, iterated measurements over a given integration interval.

This paper makes three contributions. First, a nonlinear state trajectory model is developed that includes the polynomial process noise model, and derivations are presented for all of the partial derivatives that relate the final state and measurements to the initial state and process noise polynomial coefficients. The linearized
system is then assembled into a square-root information filter form, which is convenient for performing the state update while simultaneously estimating the process noise coefficients. The second contribution is the development of an optimized statistical model of the polynomial process noise coefficients. This optimization causes the piecewise polynomial process noise to approximate continuous-time white noise as closely as possible. The third contribution uses the first two developments to design an extended Kalman filter that rationally handles implicitly defined measurement effectiveness times. Although this filtering technique is applicable to a variety of estimation problems, tracking problems with radio navigation measurements are used as examples throughout the paper. The validity of the new approach is tested in truth-model simulations using a simple tracking problem.

The remainder of the paper is divided into four major sections. Section II presents models for the state dynamics, the process noise, and the measurements, linearizes the resulting models, and discusses dense output numerical integration. Section II also incorporates the models into dynamic propagation and state update equations that deal with implicit measurement update times. Section III presents the statistical description of the piecewise polynomial process noise model and an optimization that causes it to approximate white noise. Section VI presents the tracking problem truth-model simulations and the results. Section V presents the conclusions.

II. Modeling and Linearization with Sampled Measurements, Implicit Update Times, and Dense Output Numerical Integration

A nonlinear state trajectory model that treats the process noise as a continuous-time polynomial over an integration interval is presented in this section. The relationships between the observables and the independent variables are linearized to obtain the simplified expressions that are needed to solve the nonlinear filtering problem using the principles of the extended Kalman filter. The resulting expressions are assembled into a non-standard form of a square-root information filter, one that performs its dynamic propagation and measurement update simultaneously.

A. Dynamics Modeling and Linearization with Dense Output

The trajectory model is expressed by the following nonlinear, continuous-time, ordinary differential equation with forcing:

\[ \dot{x}(t) = f[t, x(t), w(t)] \]  

(1)

where \( x(t) \) is the system’s state vector at time \( t \), \( w(t) \) is the process noise vector, and the function \( f \) models the system’s dynamics and how the process noise influences them. Since the signal launch times must
be solved for iteratively during the filter’s measurement update step, it is desirable to have a polynomial
description of the state trajectory over the integration interval from time $t_k$ to $t_{k+1}$. A generic $N^{th}$-order
model of such a polynomial takes the form

$$x(t) = \sum_{i=0}^{N} \tilde{x}_i^k P_i(t) \quad \text{for} \quad t_k \leq t < t_{k+1} \quad (2)$$

where $\tilde{x}_i^k$ is the $i^{th}$ polynomial coefficient vector that is valid over the $k^{th}$ integration interval, and the
function $P_i(t)$ is the $i^{th}$ component of a generic polynomial defined over the interval $t_k \leq t < t_{k+1}$. It should
be noted that there are many methods for developing such polynomials, including dense output Runge-Kutta
methods and multi-step Adams-Bashforth methods. Regardless of the method, an important consideration is
that the resulting polynomials satisfy the order condition of the numerical method. Runge-Kutta methods,
which do not require the storage of states or function evaluations prior to the beginning of the current
integration interval, are the most appropriate methods for use in sequential estimation and are considered
exclusively in the remainder of the paper. For a more thorough discussion of continuous output numerical
integration algorithms, please see Refs. 1 and 4.

The proposed process noise model over a particular integration interval takes a similar form:

$$w_k(t) = \sum_{j=0}^{M} \tilde{w}_j^k P_j(t) \quad \text{for} \quad t_k \leq t < t_{k+1} \quad (3)$$

where $\tilde{w}_j^k$ is the $j^{th}$ polynomial coefficient vector that is valid over the $k^{th}$ integration interval, and the
function $P_j(t)$ is the $j^{th}$ component of an $M^{th}$-order generic polynomial. The choice of the particular form of
polynomial used in Eq. (3) will almost certainly differ from one used for the state propagation, as discussed
in the next section. The polynomial coefficient vectors are assumed to be constant over the integration
interval and assumed to be drawn from white-noise sequences, which implies that $E[(\tilde{w}_j^k)(\tilde{w}_l^j)^T] = 0$ for all
$ji$ pairs when $k \neq l$, i.e., the process noise from one interval is uncorrelated with the process noise of all
other intervals.

Before continuing with the derivation of the model, it is important to point out the functional relationship
between the state and the process noise coefficients explicitly. Although the $\tilde{w}_j^k$ coefficients cannot be exactly
known a priori, the derivation of the filter equations requires the development of a model of how they affect
the state trajectory over the integration interval. With this in mind, the system’s state has the following
functional dependence:

$$x(t; x_k, t_k, t_{k+1}, \tilde{w}_k^0, \tilde{w}_k^1, \ldots, \tilde{w}_k^M) \quad (4)$$

This dependence is central to the derivation of the linearized expressions presented later in this section. An
algorithm for the explicit computation of this function can be developed by using the the dynamics model in Eq. (1), the process noise model in Eq. (3), and any dense output numerical integration method. The appendix presents one appropriate numerical method, a fifth-order dense output Runge-Kutta method. In order to simplify the notation, the set $\Sigma_k = \{x_k, t_k, t_{k+1}, \tilde{w}_k^0, \tilde{w}_k^1, \ldots, \tilde{w}_k^M\}$ is used to denote the interval-related dependencies in the subsequent expressions.

In order to produce state and process noise coefficient estimates, a filter needs a description of how changes in the initial state and process noise coefficients influence the state at any other time. This information is provided in the form of partial derivatives of the function in Eq. (4) with respect the initial state and process noise coefficients. These relationships are derived in the next paragraphs.

Given the dynamics model in Eq. (1), the state trajectory over the interval from $t_k$ to $t_{k+1}$ depends on the initial state and on the process noise coefficients over the interval. Any changes in those values will result in changes in the state at time $t_{k+1}$. Consider first how changes in the initial state influence its final state. The partial derivative of the state at time $t_{k+1}$ with respect to its state at time $t_k$ describes this relationship and defines the state transition matrix as

$$\Phi_k = \frac{\partial x(t_{k+1}; \Sigma_k)}{\partial x_k} \quad (5)$$

For a general nonlinear model, no analytical expression for this matrix exists. It is computed by numerically integrating the following matrix differential equation from time $t_k$ to $t_{k+1}$:

$$\dot{\Phi}(t, t_k) = A(t)\Phi(t, t_k) \quad (6)$$

with the initial condition $\Phi(t_k, t_k) = I$. In Eq. (6), the matrix $A(t)$ is defined as the partial derivative of the dynamics model with respect to the state:

$$A(t) = \frac{\partial f[t, x(t, \Sigma_k), w(t)]}{\partial x(t, \Sigma_k)} \quad (7)$$

This partial derivative can, in most cases, be evaluated analytically, although in practice it is often evaluated using simplified models or finite difference methods. It should be noted that numerical integration of Eq. (6) can be performed at the same time as the numerical integration of the state vector and that dense output Runge-Kutta polynomials can be generated for the elements of the state transition matrix.

Now consider how changes in the process noise polynomial coefficients influence the state at time $t_{k+1}$. This sensitivity is expressed in the following process noise coefficient influence matrix for the $j^{th}$ process
noise coefficient:
\[ \Gamma^j_k = \frac{\partial x(t_{k+1}; \Sigma_k)}{\partial \tilde{w}_{j,k}} \]  

(8)

As with the state transition matrix, an analytic expression for this matrix normally does not exist. It is computed by numerically integrating the following matrix differential equation from time \( t_k \) to \( t_{k+1} \):

\[ \dot{\Gamma}^j(t, t_k) = A(t)\Gamma^j(t, t_k) + D(t) \]  

(9)

with the initial condition \( \Gamma^j(t_k, t_k) = 0 \). In Eq. (9), the matrix \( D(t) \) is defined as the partial derivative of the model with respect to the process noise coefficients:

\[ D(t) = \frac{\partial f[t, x(t; \Sigma_k), w(t)]}{\partial \tilde{w}_{j,k}} \]  

(10)

In order to assemble these relationships into the linearized trajectory model, a vector of stacked process noise polynomial coefficients is defined as

\[ \tilde{w}_k = \begin{bmatrix} \tilde{w}_k^0 \\ \vdots \\ \tilde{w}_k^M \end{bmatrix} \]  

(11)

and the associated process noise coefficient influence matrix is defined as

\[ \Gamma_k = [\Gamma^0_k \ \Gamma^1_k \ \ldots \ \Gamma^M_k] \]  

(12)

Defining a state deviation vector as \( \Delta x_k = x(t_k; \Sigma_k) - \hat{x}(t_k; \Sigma_k) \), where \( \hat{x}(t_k; \Sigma_k) \) is the best estimate of the state at time \( t_k \), and assuming that \( E[\tilde{w}_k] = 0 \), the linearized trajectory model takes the form

\[ \Delta x_{k+1} = \Phi_k \Delta x_k + \Gamma_k \tilde{w}_k \]  

(13)

Note that with dense output polynomials for the elements of \( \Phi(t, t_k) \) and \( \Gamma(t, t_k) = [\Gamma^0(t, t_k) \ \Gamma^1(t, t_k) \ \ldots \ \Gamma^M(t, t_k)] \)

(14)

it is also possible to write the state deviation equation at any point in the interval as a function of \( x_k \) and \( \tilde{w}_k \), i.e.,

\[ \Delta x(t) = \Phi(t, t_k)\Delta x_k + \Gamma(t, t_k)\tilde{w}_k \]  

(15)
B. Measurement Modeling with Implicit Measurement Times

A general, nonlinear, implicitly constrained measurement model takes the form

\[ y_{k(q)} = h_{k(q)}[t_{k(q)}, \tilde{t}_{k(q)}, x(\tilde{t}_{k(q)}; \Sigma_k)] + v_{k(q)} \tag{16} \]

subject to the constraint

\[ 0 = g_{k(q)}[t_{k(q)}, \tilde{t}_{k(q)}, x(\tilde{t}_{k(q)}; \Sigma_k)] \tag{17} \]

where \( \tilde{t}_{k(q)} = t_{k(q)} - \delta t_{k(q)} \) with \( t_{k(q)} \) being a known signal reception time and \( \delta t_{k(q)} \) being an unknown transmission delay. In Eq. (16), \( y_{k(q)} \) is a discrete-time vector measurement that contains information about the state at the signal launch time, \( \tilde{t}_{k(q)} \). The subscript \( k(q) \) indicates the \( q^{th} \) measurement that was received on the \( k^{th} \) interval, i.e., the interval from \( t_k \) to \( t_{k+1} \). The function \( h_{k(q)} \) models the nonlinear relationship between the state and the measurement vector. Equation (16) includes a measurement noise vector, \( v_{k(q)} \), which is assumed to be drawn from a zero-mean white-noise sequence and to be uncorrelated to the process noise. In Eq. (17), the function \( g_{k(q)} \) constrains the speed of light multiplied by the propagation time, \( c\delta t_{k(q)} \), to equal the range between the radio navigation receiver’s position at reception time, \( t_{k(q)} \), and the vehicle at the signal launch time, \( \tilde{t}_{k(q)} \). The launch time, which is defined by Eq. (17) as an implicit function of \( \Sigma_k \), is determined by solving Eq. (17) using any appropriate numerical equation solving technique.

For multiple measurements on a given interval, i.e., for \( q = 1, 2, \ldots \), there are multiple versions of Eq. (17) that must be solved for multiple transmission times, \( \tilde{t}_{k(q)} \). Once the signal launch times have been determined, these multiple measurements can be stacked into a single large measurement vector that applies for the entire interval.

Given this measurement model with the implicit measurement times, the extended Kalman filter needs Jacobian partial derivatives of the model in order to carry out its measurement update calculation. These include partial derivatives of \( h_{k(q)} \) with respect to \( x_k \) and \( \tilde{w}_k \). The correct formulas must properly account for the dependence of \( x(t; \Sigma_k) \) on \( x_k \) and \( \tilde{w}_k \) and for the dependence of \( \tilde{t}_{k(q)} \) on these quantities. Starting with the partial derivative with respect to \( x_k \), all the possible dependency paths from the initial state to the measurements are accounted for through the application of the chain rule. The following expression is derived:

\[
H_{x_k(\tilde{t}_{k(q)})} = \frac{\partial h_{k(q)}[t_{k(q)}, \tilde{t}_{k(q)}, x(t_{k(q)}; \Sigma_k)]}{\partial x(t_{k(q)}; \Sigma_k)} \left[ \frac{\partial x(t_{k(q)}; \Sigma_k)}{\partial x_k} + \frac{\partial x(\tilde{t}_{k(q)}; \Sigma_k)}{\partial x_k} \right] \frac{\partial \tilde{t}_{k(q)}}{\partial x_k} + \frac{\partial h_{k(q)}[t_{k(q)}, \tilde{t}_{k(q)}, x(t_{k(q)}; \Sigma_k)]}{\partial \tilde{t}_{k(q)}} \frac{\partial \tilde{t}_{k(q)}}{\partial x_k} \tag{18}
\]

The first partial derivative on the right hand side of Eq. (18) indicates how changes in the state at the signal
launch time, \( \bar{t}_{k(q)} \), influence the measurement. An analytical expression for this partial derivative is easy to derive for most measurement models. The state at the signal launch time, in turn, is influenced by changes in the initial state, \( \mathbf{x}_k \), in two different ways, which are indicated by the bracketed partial derivatives. First, changes in the initial state influence the state at time \( \bar{t}_{k(q)} \) directly through the partial derivative \( \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \mathbf{x}_k} \), which is recognized as the state transition matrix from time \( t_k \) to \( \bar{t}_{k(q)} \). This partial derivative may be obtained by evaluating the Runge-Kutta method’s dense output polynomials. The second dependency path is through the signal launch time variable. Changes in the signal launch time influence the state at the launch time directly through partial derivative \( \frac{\partial \mathbf{x}(t_{k(q)}; \Sigma_k)}{\partial \bar{t}_{k(q)}} \), which is recognized as the time derivative of the state.* The signal launch time is influenced by changes in the initial state as expressed by the partial derivative \( \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \). The measurement may also be influenced directly by changes in the signal launch time. That dependence is represented in the first partial derivative outside the brackets, \( \frac{\partial \mathbf{x}(t_{k(q)}; \Sigma_k)}{\partial \bar{t}_{k(q)}} \) \( k \). The last term in Eq. (18), \( \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \), represents, again, the dependence of the signal launch time on the changes in the initial state.

In order to evaluate this last term, \( \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \), one takes the partial derivative of the constraint in Eq. (17) with respect to \( \mathbf{x}_k \). Using the chain rule, this partial derivative is

\[
0 = \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)} \left[ \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \mathbf{x}_k} + \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \bar{t}_{k(q)}} \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \right] + \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \bar{t}_{k(q)}} \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \tag{19}
\]

The partial derivatives in Eq. (19) indicate that changes in the initial state, \( \mathbf{x}_k \), influence the constraint in the same way as they influence the measurement. Note the similarity between Eq. (19) and Eq. (18). The important point when considering Eq. (19) is that the partial derivative, \( \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \), enters linearly, and the solution to this linear equation is

\[
\frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} = \left[ \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)} \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \mathbf{x}_k} + \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \bar{t}_{k(q)}} \right]^{-1} \left[ \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \bar{t}_{k(q)}} \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \right] \tag{20}
\]

It includes a combination of partial derivatives that can be evaluated analytically, e.g., \( \frac{\partial g_k(q)[\bar{t}_{k(q)}, \bar{t}_{k(q)}, \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)]}{\partial \bar{t}_{k(q)}} \frac{\partial \bar{t}_{k(q)}}{\partial \mathbf{x}_k} \), and partial derivatives that have already been discussed, e.g., the state transition matrix from time \( t_k \) to \( \bar{t}_{k(q)} \), \( \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \mathbf{x}_k} \), and the time derivative of the state at time \( \bar{t}_{k(q)} \), \( \frac{\partial \mathbf{x}(\bar{t}_{k(q)}; \Sigma_k)}{\partial \bar{t}_{k(q)}} \).

Next, the partial derivative of a measurement with respect to the process noise polynomial coefficients is derived in a completely analogous way. The results for the measurement’s dependence on the \( j \)th polynomial

---

*This time derivative normally comes from the dynamics equation in Eq. (1), though there might be reason to use the explicit time derivative of the dense output polynomial in Eq. (2). These will be close to each other by the construction of the dense output numerical scheme, but probably not equal.
coefficient are presented in the next two equations:

\[
H^j_{\hat{w}_k(q)} = \frac{\partial h_k(q)[t_k(q), \hat{t}_k(q), \hat{x}(\hat{t}_k(q); \Sigma_k)]}{\partial x(t_k(q); \Sigma_k)} \left[ \frac{\partial x(t_k(q); \Sigma_k)}{\partial \hat{w}_k^j} \right] + \frac{\partial h_k(q)[t_k(q), \hat{t}_k(q), \hat{x}(\hat{t}_k(q); \Sigma_k)]}{\partial \hat{t}_k(q)} \frac{\partial \hat{t}_k(q)}{\partial \hat{w}_k^j}
\]

(21)

\[
\frac{\partial \hat{t}_k(q)}{\partial \hat{w}_k^j} = \left[ -\frac{\partial g_k(q)[t_k(q), \hat{t}_k(q), \hat{x}(\hat{t}_k(q); \Sigma_k)]}{\partial \hat{x}(t_k(q); \Sigma_k)} \frac{\partial \hat{x}(t_k(q); \Sigma_k)}{\partial \hat{w}_k^j} + \frac{\partial g_k(q)[t_k(q), \hat{t}_k(q), \hat{x}(\hat{t}_k(q); \Sigma_k)]}{\partial \hat{t}_k(q)} \frac{\partial \hat{t}_k(q)}{\partial \hat{w}_k^j} \right]
\]

(22)

Equations (21) and (22) are evaluated for each of the \( M + 1 \) process noise polynomial coefficients, and the results are assembled into the form

\[
H_{\hat{w}_k(q)} = [H^0_{\hat{w}_k(q)} \ H^1_{\hat{w}_k(q)} \ \ldots \ H^M_{\hat{w}_k(q)}]
\]

(23)

An observed-minus-modeled measurement vector is defined as \( \Delta y_k(q) = y_k(q) - h_k(q)[t_k(q), \hat{t}_k(q), \hat{x}(\hat{t}_k(q); \Sigma_k)] \), and the linearized measurement model is defined as

\[
\Delta y_k(q) = H_{\hat{x}_k(q)} \Delta x_k + H_{\hat{w}_k(q)} \tilde{w}_k + v_k(q)
\]

(24)

Stacking all of the measurements \( q = 1, 2, \ldots, N_m \) for the sample interval from \( t_k \) to \( t_{k+1} \) results in the following definitions:

\[
\Delta y_k = \begin{bmatrix} \Delta y_{k(1)} \\ \Delta y_{k(2)} \\ \vdots \\ \Delta y_{k(N_m)} \end{bmatrix}
\]

(25)

\[
H_{\hat{x}_k} = \begin{bmatrix} H_{\hat{x}_k(1)} \\ H_{\hat{x}_k(2)} \\ \vdots \\ H_{\hat{x}_k(N_m)} \end{bmatrix}
\]

(26)

\[
H_{\hat{w}_k} = \begin{bmatrix} H_{\hat{w}_k(1)} \\ H_{\hat{w}_k(2)} \\ \vdots \\ H_{\hat{w}_k(N_m)} \end{bmatrix}
\]

(27)
and the final linearized measurement equation:

$$\Delta y_k = H_{x_k} \Delta x_k + H_{\tilde{w}_k} \tilde{w}_k + v_k$$  \hspace{1cm} (28)

**C. Combined SRIF Dynamics Propagation and Measurement Update**

In order to solve the linearized estimation problem with a square-root information filter, in a generalization of the method in Ref. 3, the following information equations are constructed:

$$z_{x_k} = R_{xx_k} \Delta x_k + v_{x_k}$$  \hspace{1cm} (29)

and

$$z_{\tilde{w}_k} = R_{\tilde{w}\tilde{w}_k} \tilde{w}_k + v_{\tilde{w}_k}$$  \hspace{1cm} (30)

where the noise terms, $v_{x_k}$ and $v_{\tilde{w}_k}$, are drawn from independent zero-mean, unit-variance white-noise sequences. The matrices, $R_{xx_k}$ and $R_{\tilde{w}\tilde{w}_k}$, are defined as the inverse square roots of the a priori state and process noise coefficient covariances, e.g., $R_{xx_k}^{-1} R_{xx_k}^{-T} = P_{xx_k}$ and $R_{\tilde{w}\tilde{w}_k}^{-1} R_{\tilde{w}\tilde{w}_k}^{-T} = Q_k$. The information equations are then combined with the measurements in the vector-matrix equation

$$\begin{bmatrix} z_{\tilde{w}_k} \\ z_{x_k} \\ \Delta y_k \end{bmatrix} = \begin{bmatrix} R_{\tilde{w}\tilde{w}_k} & 0 \\ 0 & R_{xx_k} \\ H_{\tilde{w}_k} & H_{x_k} \end{bmatrix} \begin{bmatrix} \tilde{w}_k \\ \Delta x_k \end{bmatrix} + \begin{bmatrix} v_{\tilde{w}_k} \\ v_{x_k} \\ v_k \end{bmatrix}$$  \hspace{1cm} (31)

It is assumed that the measurements are scaled and transformed appropriately such that the measurement noise, $v_k$, is drawn from a zero-mean, unit-variance white-noise sequence. The linearized trajectory model in Eq. (13) is used to eliminate $\Delta x_k$ from Eq. (32) in the following vector-matrix equation:

$$\begin{bmatrix} z_{\tilde{w}_k} \\ z_{x_k} \\ \Delta y_k \end{bmatrix} = \begin{bmatrix} R_{\tilde{w}\tilde{w}_k} & 0 \\ -R_{xx_k} \Phi_k^{-1} \Gamma_k & R_{xx_k} \Phi_k^{-1} \\ [H_{\tilde{w}_k} - H_{x_k} \Phi_k^{-1} \Gamma_k] & H_{x_k} \Phi_k^{-1} \end{bmatrix} \begin{bmatrix} \tilde{w}_k \\ \Delta x_{k+1} \end{bmatrix} + \begin{bmatrix} v_{\tilde{w}_k} \\ v_{x_k} \\ v_k \end{bmatrix}$$  \hspace{1cm} (32)

Next, the propagation and measurement update steps are performed simultaneously by orthogonal/upper triangular (QR) factorization according to standard square-root information filtering techniques, resulting
in the following a posteriori system:

\[
\begin{bmatrix}
\hat{z}_{\omega_k} \\
\hat{z}_{x_{k+1}} \\
\z_{res}
\end{bmatrix} =
\begin{bmatrix}
\hat{R}_{\omega\omega_k} & \hat{R}_{\omega x_{k+1}} & 0 \\
0 & \hat{R}_{xx_{k+1}} & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\hat{w}_k \\
\Delta x_{k+1} \\
v_{res}
\end{bmatrix}
+ \begin{bmatrix}
\hat{v}_{\omega_k} \\
\hat{v}_{x_{k+1}} \\
v_{res}
\end{bmatrix}
\] (33)

This equation may be used to generate the a posteriori state estimate, the process noise polynomial coefficient estimates, and the associated covariances according to the following formulas:

\[
\Delta \hat{x}_{k+1} = \hat{R}^{-1}_{xx_{k+1}} \hat{z}_{x_{k+1}}
\] (34)

\[
\hat{x}_{k+1} = x(t_{k+1}; x_k, t_k, t_{k+1}, \tilde{w}_k = 0) + \Delta \hat{x}_{k+1}
\]

\[
P_{xx_{k+1}} = \hat{R}^{-1}_{xx_{k+1}} \hat{R}^{-T}_{xx_{k+1}}
\]

\[
\hat{w}_k = \hat{R}^{-1}_{\omega\omega_k} \begin{bmatrix} z_{\omega_k+1} - \hat{R}_{\omega x_{k+1}} \Delta \hat{x}_{k+1} \end{bmatrix}
\]

An important point has been glossed over in the preceding. In the process noise coefficient information equation, Eq. (30), the matrix \(R_{\tilde{w}\tilde{w}}\) that meets the requirement that the noise sequence \(v_{\tilde{w}_k}\) is zero-mean and unit-variance was assumed to be given. How is that matrix determined? Also, how does one choose \(R_{\tilde{w}\tilde{w}}\) such that the continuous-time polynomial process noise is as white as possible over the integration interval? These questions are answered in the next section.

III. Optimal Approximation to Continuous-Time White Noise Via Random Piecewise Polynomials

This section extends the definition of the polynomial process noise model in Eq. (3) for all integration intervals and develops the model’s statistics. That development is guided by the basic assumption of the Kalman filter that the system’s state is a Markov process, i.e., that all of the information necessary to propagate the state is summed up in the state at a given time. This assumption relies on the complete unpredictability of the process noise due to its whiteness. If the process noise were autocorrelated, i.e., colored, the states prior to the given time would be needed to predict future process noise values, and thus future state values in some fashion. Therefore, in order to satisfy the Markov process assumption, the system must be driven by white noise, which is usually also assumed to be zero-mean.\(^5\) The statistical behavior of a generic continuous-time Gaussian white-noise process is completely characterized by its first

\(^5\)If this whiteness condition is not satisfied, the model can be augmented with a shaping filter that is driven by white process noise. If the process noise has non-zero, but known, mean, that value can be treated as a deterministic input.
two moments, i.e., its mean and covariance, as follows:

\[ E[n(t)] = 0 \]  
\[ E[n(t)n^T(\tau)] = Q\delta(t - \tau) \]

where \( Q \) is the (possibly time-varying) process noise intensity matrix, and where \( \delta(t - \tau) \) is the Dirac delta function. The goal of this section is to develop the statistics of the proposed polynomial process noise model such that they approximate as closely as possible, under certain constraints, the standard white noise statistics.

**A. Analysis and Design of Polynomial White Noise Approximation**

The proposed process noise model is defined for all time by repeating Eq. (3) over all possible integration intervals, which are indicated by the index \( k \). The resulting infinite sum defines a piecewise polynomial process noise model as follows:

\[ w(t) = \sum_{k=-\infty}^{\infty} \sum_{j=0}^{M} \bar{w}_j P_j \left( \frac{2}{\Delta t} |t - t_k| - 1 \right) \]

where for the \( j^{th} \) piecewise polynomial element

\[ P_j(\eta) = \begin{cases} 
C_j(\eta) & \text{if } -1 \leq \eta < 1 \\
0 & \text{otherwise}
\end{cases} \]

and where \( C_j(\eta) \) is some appropriate polynomial in \( \eta \), such as a Chebyshev polynomial. Notice that the polynomial \( P_j(\eta) \) is non-zero only when the normalized argument, \( \eta \), is on the interval from \(-1\) to \(1\). 

Since this model includes discrete-time polynomial coefficients and a continuous-time description of process noise over each integration interval, a combined continuous/discrete-time approach is used to analyze its statistics with the goal of developing an analysis-based statistical design that causes Eq. (39) to approximate white noise.

Note that \( \Delta t \) is the length of a single integration interval. The following analysis assumes that \( \Delta t = t_{k+1} - t_k \) is constant; that is, it is independent of \( k \). This assumption simplifies the statistical analysis, but probably could be relaxed. The timeline illustrated in Fig. 4 shows the integration intervals for the range of \( k \) from -1 to 2. Note the points \(-\frac{\Delta t}{2}, \frac{\Delta t}{2}, \frac{3\Delta t}{2}, \text{ and } \frac{5\Delta t}{2}\) are the midpoints of the intervals over which the four sets of polynomials apply.
The key to developing the statistics of the piecewise polynomial process noise model lies in the modeling of the statistics of the polynomial's discrete-time coefficients. It is assumed that the coefficients are drawn from a zero-mean white-noise sequence. The mean for the $j^{th}$ polynomial element coefficient is, thus,

$$E[\tilde{w}_k^j] = 0 \quad (41)$$

and the covariance for the $ji^{th}$ pair is

$$E[\tilde{w}_k^j(\tilde{w}_i^l)^T] = \alpha_{ji}Q\delta_{kl} \quad (42)$$

Here, $\alpha_{ji}$ is a scaling factor, $Q$ is the continuous-time process noise intensity matrix, assumed constant in order to simplify the analysis, and $\delta_{kl}$ is the Kronecker delta function. The combined term $\alpha_{ji}Q$ represents the discrete-time process noise covariance for the $j^{th}$ coefficient pair. The goal of this section is to design the $\alpha_{ji}$ scaling factors based on an analysis of how they affect the fidelity with which Eq. (39) approximates continuous-time white noise.

This analysis starts by considering the mean and autocorrelation of the model in Eq. (39). The mean is

$$E[w(t)] = 0 \quad (43)$$

which follows from the assumption that the polynomial coefficients are drawn from a zero-mean white-noise sequence. The autocorrelation is

$$E[w(t)w^T(\tau)] = E \left[ \sum_{k=-\infty}^{\infty} \sum_{j=0}^{M} \tilde{w}_k^j P_j \left( \frac{2}{\Delta t}[t - t_k] - 1 \right) \right] \left[ \sum_{l=-\infty}^{\infty} \sum_{i=0}^{M} \tilde{w}_i^l P_i \left( \frac{2}{\Delta t}[\tau - t_k] - 1 \right) \right]^T \quad (44)$$

$$= Q \sum_{j,i=0}^{M} \alpha_{ji}R_{ji}(t - \tau) \quad (45)$$
where
\begin{equation}
R_{ji}(t-\tau) = E\left[ \sum_{k=-\infty}^{\infty} P_j \left( \frac{2}{\Delta t} [t-t_k] - 1 \right) P_i \left( \frac{2}{\Delta t} [\tau-t_k] - 1 \right) \right] \tag{46}
\end{equation}

\begin{equation}
= \frac{1}{2} \int_{\gamma_{min}}^{\gamma_{max}} C_j \left( \frac{2}{\Delta t} [t-\tau] + \gamma \right) C_i(\gamma) d\gamma \tag{47}
\end{equation}

and where
\begin{equation}
\gamma_{min} = -\frac{2}{\Delta t} \text{min}[0,t-\tau] - 1 \tag{48}
\end{equation}
\begin{equation}
\gamma_{max} = -\frac{2}{\Delta t} \text{max}[0,t-\tau] + 1 \tag{49}
\end{equation}

The derivation of Eq. (47) also uses the principle of ergodicity by which the expectation value in Eq. (46) can be computed using a time average. This time average has been computed by the change of variables 
\begin{equation}
t_k = \frac{\Delta t}{2} (\gamma + 1) \end{equation}
and by recognizing that the piecewise polynomials in Eq. (46) go to zero when their arguments have absolute values greater than one. As a consequence, \( R_{ji}(t-\tau) \) is zero if \( |t-\tau| \geq \Delta t \), consistent with the fact that \( \gamma_{min} \leq \gamma_{max} \) only if \( |t-\tau| \leq \Delta t \). Thus, the integral in Eq. (47) only needs to be evaluated when \( \gamma_{min} < \gamma_{max} \).

Consistent with the form of \( R_{ji}(t-\tau) \), the right-hand side of Eq. (47) depends only on \( t-\tau \), which enters both through its integrand and through its limits of integration. This dependence ensures that the right-hand side of Eq. (45) has a form similar to the desired form on the right-hand side of Eq. (38). If the summation on the right-hand side of Eq. (45) were equal to a Dirac delta function, then this process noise model would be white noise.

With the proposed process noise model’s autocorrelation in its final form, it is now possible to develop a method for choosing the scaling factors \( \alpha_{ji} \) for all of the coefficient pairs. The goal is to design \( \alpha_{ji} \) scaling factors such that the summation in Eq. (45) approximates the Dirac delta function in Eq. (38) as closely as possible. Notice that cross- and autocorrelation functions between the polynomial elements are completely determined by the specific polynomials used, and that for each set of polynomials, a corresponding set of tuning parameters can be chosen such that the polynomial process noise approximates white noise as closely as possible. Before those scaling factors are chosen, however, there is another constraint to consider that only becomes clear when the joint statistics of all of the polynomial coefficients are considered.

Using the large vector of stacked polynomial coefficients defined in Eq. (11), and recalling that these
values are assumed to be zero-mean, the process noise coefficient covariance matrix is

\[
Q_k = E[\tilde{w}_k \tilde{w}_k^T] = \begin{bmatrix}
\alpha_{00}Q & \alpha_{10}Q & \ldots & \alpha_{M0}Q \\
\alpha_{10}Q & \alpha_{11}Q & & \\
& \ddots & \ddots & \\
\alpha_{M0}Q & & \ldots & \alpha_{MM}Q \\
\end{bmatrix}
\]  

(50)

This matrix is constrained to be symmetric and positive semidefinite. It can be easily shown that this constraint requires the symmetric matrix

\[
A_\alpha = \begin{bmatrix}
\alpha_{00} & \alpha_{10} & \ldots & \alpha_{M0} \\
\alpha_{10} & \alpha_{11} & & \\
& \ddots & \ddots & \\
\alpha_{M0} & & \ldots & \alpha_{MM} \\
\end{bmatrix}
\]  

(51)

to be positive semi-definite as well.

The choice of the scaling factors \(\alpha_{ji}\) can be posed as a nonlinear, constrained optimization problem. Several strategies have been considered to pose and solve sensible problems, including methods based on semidefinite programming and methods based on standard nonlinear optimization techniques combined with factorization techniques that ensure the positive semidefiniteness of \(A_\alpha\). For simplicity, however, this study has limited the optimization to the diagonal elements of the \(A_\alpha\) matrix, setting all off-diagonal elements to be zero. This approach, which reduces the optimization to a quadratic programming problem, admits a simple form of the positive semidefinite constraint, but still provides the optimization enough flexibility to choose \(\alpha_{jj}\) coefficients that yield a good approximation to white noise. A further study may reveal that optimizing over all of the elements in \(A_\alpha\) is worthwhile.

The posing of a sensible quadratic programming problems begins by defining a new function that is the sum of the product of the scaling factors and the polynomial autocorrelation functions:

\[
B(\tau) = \sum_{j=0}^{M} \alpha_{jj} R_{jj}(\tau) 
\]  

(52)

If the autocorrelation functions \(R_{jj}\) for all \(j\) are uniformly sampled \(N_s\) times over the interval from \(-\Delta t\) to
\( \Delta t \), a sampled version of \( B(\tau) \) can be constructed. It takes the following vector form:

\[
\mathbf{b} = \begin{bmatrix} B(\tau_1) & B(\tau_2) & \cdots & B(\tau_{N_s}) \end{bmatrix}^T = R_s \alpha
\]  

(53)

(54)

where the following definitions apply:

\[
\alpha = \begin{bmatrix} \alpha_{00} \\
\alpha_{11} \\
\vdots \\
\alpha_{MM} \end{bmatrix}
\]  

(55)

is an \((M + 1) \times 1\) vector and

\[
R_s = \begin{bmatrix} R_{00}(\tau_1) & R_{11}(\tau_1) & \cdots & R_{MM}(\tau_1) \\
R_{00}(\tau_2) & R_{11}(\tau_2) & \cdots & R_{MM}(\tau_2) \\
\vdots & \vdots & \ddots & \vdots \\
R_{00}(\tau_{N_s}) & R_{11}(\tau_{N_s}) & \cdots & R_{MM}(\tau_{N_s}) \end{bmatrix}
\]  

(56)

is an \(N_s \times (M + 1)\) matrix. In Eqs. (53) and (56), the \(i^{\text{th}}\) sample time is defined as

\[
\tau_i = \frac{2\Delta t}{N_s} (i - \frac{1}{2}) - \Delta t \quad \text{for} \quad i = 1, 2, \ldots, N_s.
\]

The vector of diagonal scaling factors \( \alpha \) is chosen by solving the following quadratic programming problem: minimize the quadratic cost function

\[
J(\alpha) = \frac{1}{2} \mathbf{b}^T W \mathbf{b} = \frac{1}{2} \alpha^T R_s^T W R_s \alpha
\]  

(57)

(58)

subject to a linear inequality constraint and a linear equality constraint, both of which will be discussed below. The term \( W \) is a positive semidefinite diagonal weighing matrix, the choice of which will also be discussed in the next section. The goal of the cost function is to keep \( B(\tau_i) \) as small as possible for \( \tau_i \) values different from zero, consistent with the form of the Dirac delta function.

The vector of diagonal scaling factors \( \alpha \) is chosen by solving the following quadratic programming problem: minimize the quadratic cost function

\[
J(\alpha) = \frac{1}{2} \mathbf{b}^T W \mathbf{b} = \frac{1}{2} \alpha^T R_s^T W R_s \alpha
\]  

(57)

(58)

subject to a linear inequality constraint and a linear equality constraint, both of which will be discussed below. The term \( W \) is a positive semidefinite diagonal weighing matrix, the choice of which will also be discussed in the next section. The goal of the cost function is to keep \( B(\tau_i) \) as small as possible for \( \tau_i \) values different from zero, consistent with the form of the Dirac delta function.

The first constraint is the following inequality:

\[
\alpha \geq 0
\]  

(59)

which ensures that the resulting \( A_\alpha \) matrix will satisfy the positive simidefinite condition. The second
constraint is motivated by the definition of the Dirac delta function, specifically that it integrates to one. If the autocorrelation function of the polynomial process noise is intended to approximate white noise, then the summation in Eq. (45) should approximate a Dirac delta function and thus also integrate to one. In the case of a continuous polynomial, that constraint takes the form

\[ \int_{-\Delta t}^{\Delta t} \sum_{j=0}^{M} \alpha_{jj} R_{jj}(\tau) d\tau = 1 \]  \hspace{1cm} (60)

The finite sum approximation of the integral in Eq. (60) takes the form

\[ c^T b = c^T R_s \alpha = 1 \]  \hspace{1cm} (61)

where

\[ c = \Delta \tau \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \]  \hspace{1cm} (62)

In the last equation, \( \Delta \tau = \tau_{i+1} - \tau_i = \frac{2M}{N_s} \) is the sample interval used to construct \( b \) in Eq. (53).

Note that in the actual calculations, numerical approximations of \( R_{jj} \) have been used to construct the elements of the \( R_s \) matrix in Eq. (56). These approximations use numerical integration to evaluate the right-hand side of Eq. (47).

**B. Example Development of a Piecewise Polynomial Approximation of White Noise**

This subsection designs and analyzes an example piecewise polynomial noise model. It also analyzes the effect of the number of polynomials \( M + 1 \) on the white noise approximation accuracy.

The example design is verified by analyzing a simulated realization of the piecewise polynomial noise model. The design involve choosing a set of polynomials \( C_j(\eta) \) and the number of polynomials \( M + 1 \). For the present example, and throughout the remainder of this paper, Chebyshev polynomials have been used. These polynomials, which are easily generated using recursive formulas, have favorable cross-correlation properties and predictable error properties over the interval from \(-1\) to \(1\).\(^6\) The example number of polynomials has been chosen to be \( M + 1 = 11 \).

The final part of the design is to determine the vector of scaling factors defined in Eq. (55). Recall that this is done by solving the quadratic programming problem presented in the previous subsection. In order to solve that problem, several parameters must be set. The number of samples is chosen to be \( N_s = 201 \),
and the weighing matrix $W$ is chosen to be a modified version of the $N_s \times N_s$ identity matrix. Like the identity matrix, $W$ has ones along its diagonal, but the center 15 percent of its diagonal elements are set to zero. This heuristically derived weighting penalizes in Eq. (57) any deviation from zero towards the ends of the polynomial noise’s autocorrelation, but allows deviations towards the middle. Unweighting the middle affords the optimization enough flexibility to satisfy the constraints in Eqs. (59) and (60) in a manner that helps the autocorrelation approximate the Dirac delta function. The resulting sets of scaling factors for all $M$ values from 0 to 10 are presented in Tables 1 and 2 for the case $\Delta t = 1$. If $\Delta t$ is different from 1, then the $\alpha$ values in Tables 1 and 2 must be rescaled by the factor $\frac{1}{\Delta t_{new}}$.

The realization is constructed by generating random polynomial coefficients according the statistics defined in Eqs. (41) and (42) for a large number of integration intervals. The resulting piecewise continuous noise is analyzed in order to show that its statistics agree with the theory. The following steps have been followed to create a scalar piecewise polynomial noise realization:

1. Choose the continuous-time noise intensity $Q$ in Eq. (42). For simplicity, this scalar is set to unity.

2. Generate random polynomial coefficients according to the statistics defined in Eqs. (41) and (42) using $Q$ from step 1 and $\alpha$ from Table 2. For each interval from $t_k$ to $t_{k+1}$, $M + 1$ independent scalar coefficients are obtained for this scalar noise example.

3. Densely sample the corresponding realization of the piecewise polynomial noise model. The resulting sequence is a densely sampled version of $w(t)$, which is plotted over 1,000 piecewise polynomial intervals, each of unit length, in Fig. 5. Figure 6, which shows a close up of the first three intervals, reveals the piecewise polynomial structure. Note that the sample interval $\Delta t = 1$ has been used in this example.

An analysis has been used to verify that the statistics of their realization match the theory that has been used to design it. In particular, the autocorrelation of the densely sampled version of $w(t)$ has been calculated and compared to the theoretical autocorrelation predicted by Eq. (45) for the $\alpha_{jj}$ coefficients in Tables 1 and 2. The simulated data’s autocorrelation and the theoretical autocorrelation are both shown in Fig. 7. Both curves integrate to one over the interval from $-\Delta t (= -1)$ to $+\Delta t (= +1)$. The agreement between the curves indicates that the simulated piecewise polynomial noise behaves as expected over a large number of integration intervals.

In addition to analyzing the statistics of the simulated data, it is important to demonstrate the polynomial noise’s ability to approximate white noise. This behavior is shown in Fig. 8, which plots autocorrelation functions for noise models with piecewise polynomials of increasing order, ranging from 0 to 15. When the piecewise polynomial noise model is realized at $M = 0$, the result is the familiar zero-order hold, or piecewise constant, noise. Its autocorrelation, the low triangular-shaped dashed curve, is the poorest approximation
Table 1. Polynomial noise coefficient covariance scaling vectors, $\alpha$, for $M = 0, 1, \ldots, 5$.

<table>
<thead>
<tr>
<th>$j$</th>
<th>$M = 0$</th>
<th>$M = 1$</th>
<th>$M = 2$</th>
<th>$M = 3$</th>
<th>$M = 4$</th>
<th>$M = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td>0.80763605</td>
<td>1.00000000</td>
<td>0.81233857</td>
<td>0.77186939</td>
</tr>
<tr>
<td>1</td>
<td>0.35760408</td>
<td>0.00000000</td>
<td>0.00000000</td>
<td>0.19385607</td>
<td>0.00000000</td>
<td>2.13783371</td>
</tr>
<tr>
<td>2</td>
<td>1.80266096</td>
<td>0.00000000</td>
<td>1.68713208</td>
<td>2.13783371</td>
<td>0.00000000</td>
<td>1.2366553</td>
</tr>
<tr>
<td>3</td>
<td>2.26233225</td>
<td>0.00000000</td>
<td>1.2366553</td>
<td>1.2366553</td>
<td>0.00000000</td>
<td>2.16598011</td>
</tr>
<tr>
<td>4</td>
<td>2.03418489</td>
<td>0.00000000</td>
<td>0.60884398</td>
<td>2.16598011</td>
<td>0.00000000</td>
<td>0.84909729</td>
</tr>
<tr>
<td>5</td>
<td>1.97400828</td>
<td>0.00000000</td>
<td>1.61190861</td>
<td>0.84909729</td>
<td>0.00000000</td>
<td>1.5050771</td>
</tr>
<tr>
<td>6</td>
<td>1.813372</td>
<td>0.00000000</td>
<td>1.40826626</td>
<td>1.5050771</td>
<td>0.00000000</td>
<td>1.28861941</td>
</tr>
<tr>
<td>7</td>
<td>1.54757888</td>
<td>0.00000000</td>
<td>1.31627568</td>
<td>1.28861941</td>
<td>0.00000000</td>
<td>1.08351061</td>
</tr>
</tbody>
</table>

Table 2. Polynomial noise coefficient covariance scaling vectors, $\alpha$, for $M = 6, 7, \ldots, 10$.

<table>
<thead>
<tr>
<th>$j$</th>
<th>$M = 6$</th>
<th>$M = 7$</th>
<th>$M = 8$</th>
<th>$M = 9$</th>
<th>$M = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.80835445</td>
<td>0.71434311</td>
<td>0.81179258</td>
<td>0.78044527</td>
<td>0.81610341</td>
</tr>
<tr>
<td>1</td>
<td>0.93181255</td>
<td>0.77392921</td>
<td>1.40863993</td>
<td>1.3515895</td>
<td>1.6006409</td>
</tr>
<tr>
<td>2</td>
<td>1.7247258</td>
<td>2.63136217</td>
<td>1.69586744</td>
<td>2.00404106</td>
<td>1.65584545</td>
</tr>
<tr>
<td>3</td>
<td>0.76752142</td>
<td>0.56510364</td>
<td>1.10715424</td>
<td>1.0907825</td>
<td>1.24788453</td>
</tr>
<tr>
<td>4</td>
<td>1.74449507</td>
<td>1.29677485</td>
<td>1.65692286</td>
<td>1.42710078</td>
<td>1.66362196</td>
</tr>
<tr>
<td>5</td>
<td>0.00000000</td>
<td>1.50903074</td>
<td>0.60884398</td>
<td>1.32314487</td>
<td>0.84909729</td>
</tr>
<tr>
<td>6</td>
<td>1.97400828</td>
<td>0.00000000</td>
<td>1.61190861</td>
<td>0.65538133</td>
<td>1.5050771</td>
</tr>
<tr>
<td>7</td>
<td>1.813372</td>
<td>0.00000000</td>
<td>1.40826626</td>
<td>0.42558707</td>
<td>1.28861941</td>
</tr>
<tr>
<td>8</td>
<td>1.54757888</td>
<td>0.00000000</td>
<td>1.31627568</td>
<td>0.00000000</td>
<td>1.08351061</td>
</tr>
</tbody>
</table>
Figure 5. The time history of simulated polynomial noise over 1,000 piecewise polynomial intervals.

Figure 6. The time history of simulated polynomial noise over 3 piecewise polynomial intervals.
Figure 7. Theoretical and simulated piecewise polynomial noise autocorrelation.

to continuous-time white noise of the four shown in Fig. 8, recalling that the target autocorrelation is the Dirac delta function. When the model is realized at $M = 15$, the resulting noise is a better approximation of continuous-time white noise, as is evident in its autocorrelation function. It looks much narrower, like the desired delta function. This behavior demonstrates that as the order of the piecewise polynomial noise increases, it better approximates the behavior of continuous-time white noise. Note that each curve of Fig. 8 required the solution to the new quadratic program for the $\alpha_{jj}$ values. In the case of $M = 0$, $\alpha_{00} = \frac{1}{\Delta t}$, as dictated by Eq. (60). This demonstrates that the present developments represent a generalization of the standard approximation of continuous-time white noise via a simple zero-order hold model, which is the $M = 0$ model.

Note that the intended use of this process noise model is in the context of a sampled data Kalman filter that uses dense output Runge-Kutta numerical integration, as is discussed in the appendix. For a given order, the Runge-Kutta method has a limited ability to accurately approximate the effects of rapid, high-order dynamic variations. This fact imposes a limit on the order of the polynomials used in the piecewise polynomial process noise model. A simple analysis suggests that the order of the polynomials, $M$, should be less than or equal to the order of the Runge-Kutta method minus one. It is conjectured that using too large a value for $M$ for a given Runge-Kutta order does no more harm than the following: useless extra calculations are carried out by the Kalman filter in a vain attempt to simulate white-noise bandwidth that
Figure 8. Autocorrelation for polynomial noise realizations of different order.

lies outside of the sensitive range of the Runge-Kutta numerical integration.

IV. Demonstration of the Kalman Filter in a Simulation

The Kalman filter described in Section II that processes implicitly constrained measurements and that incorporates the piecewise polynomial process noise model described in Section III is demonstrated in this section by solving a simple radio navigation tracking problem using data from a truth-model simulation. One goal of the simulation is to test how well the piecewise polynomial process noise theory approximates the true continuous-time process noise statistics. The statistics of the square-root information filter (SRIF) residuals, $z_{res}$ in Eq. (33), are used to investigate this issue. A second goal of the simulation is to test how well the filter performs as the order of the piecewise polynomial process noise model is increased from the standard zero-order hold model and as the number of measurements in a given Kalman filter interval is increased. For this test, the sum of the squares (SOS) of the SRIF residuals will be used as a performance indicator.

The example problem involves a vehicle that transmits a radio beacon to two stationary receivers, as illustrated in Fig. 9. The vehicle is allowed to move along a straight line with three states (position, velocity, and acceleration) according to a continuous-time, triple-integrator dynamics model. The acceleration is
driven by a white-noise process. Additionally, the transmitter clock error is modeled as a two-state random process in which both its phase and frequency errors are driven by white-noise processes. The clock model is taken from Ref. 7. The resulting linear, continuous-time system is given in the following matrix-vector equation:

\[
\begin{bmatrix}
\dot{r}(t) \\
\dot{v}(t) \\
\dot{a}(t) \\
c\dot{\delta t}_{clk}(t) \\
c\dot{\delta f}_{clk}(t)
\end{bmatrix}
= 
\begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
r(t) \\
v(t) \\
a(t) \\
c\delta t_{clk}(t) \\
c\delta f_{clk}(t)
\end{bmatrix}
+ 
\begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
w_a(t) \\
w_t(t) \\
w_f(t)
\end{bmatrix}
\]  
(63)

where \( \delta t_{clk}(t) \) is the transmitter clock phase error, \( \delta f_{clk}(t) \) is the transmitter clock frequency error, \( c \) is the speed of light, and \( w_a(t), w_t(t), \) and \( w_f(t) \) are three uncorrelated zero-mean Gaussian white-noise processes. They have the following autocorrelations:

\[
E[w_a(t)w_a(\tau)] = q_{da}\delta(t - \tau) 
\]  
(64)

\[
E[w_t(t)w_t(\tau)] = c^2h_0\delta(t - \tau) 
\]  
(65)

\[
E[w_f(t)w_f(\tau)] = c^2h_{-2}2\pi^2\delta(t - \tau) 
\]  
(66)

where \( q_{da} \) is the white-noise intensity for the acceleration perturbation, and where \( h_0 \) and \( h_{-2} \) are Allen variance parameters for the clock model.

The truth states at the signal launch times are generated using the equivalent discrete-time model driven by white-noise sequences that have the same effect as the continuous-time random processes in Eq. (63).
Time histories of the true position and transmitter clock phase error are combined with the known receiver locations to generate noisy pseudorange observables according to the following model, which assumes perfect receiver clocks:

\[ P_A(t_{A_{k(q)}}) = r(\bar{t}_{k(q)}) - r_A - c\delta_{clk}(\bar{t}_{k(q)}) + \nu_A(t_{A_{k(q)}}) \] (67)

\[ P_B(t_{B_{k(p)}}) = r_B - r(\bar{t}_{k(p)}) - c\delta_{clk}(\bar{t}_{k(p)}) + \nu_B(t_{B_{k(p)}}) \] (68)

For receiver A, \( P_A(t_{A_{k(q)}}) \) is the \( q \)th pseudorange measured on the \( k \)th filtering interval at the known signal reception time \( t_{A_{k(q)}} \), \( r(\bar{t}_{k(q)}) \) is the transmitter’s position at the unknown signal launch time \( \bar{t}_{k(q)} \), \( c\delta_{clk}(\bar{t}_{k(q)}) \) is the range-equivalent transmitter clock phase error, and \( \nu_A(t_{A_{k(q)}}) \) is white Gaussian measurement noise. Similar definitions apply to receiver B.

For a particular scenario, the transmitter’s position time history is plotted in Fig. 10. Note that the launch times, \( \bar{t}_{k(q)} \) and \( \bar{t}_{k(p)} \), are implicitly defined by the following light time equations:

\[ 0 = g_{A_{k(q)}}[t_{A_{k(q)}}, \bar{t}_{k(q)}, r(\bar{t}_{k(q)})] = r(\bar{t}_{k(q)}) - r_A - c(t_{A_{k(q)}} - \bar{t}_{k(q)}) \] (69)

\[ 0 = g_{B_{k(p)}}[t_{B_{k(p)}}, \bar{t}_{k(p)}, r(\bar{t}_{k(p)})] = r_B - r(\bar{t}_{k(p)}) - c(t_{B_{k(p)}} - \bar{t}_{k(p)}) \] (70)

Given the dynamics model in Eq. (63), the continuous-time statistics in Eqs. (64) through (66), the sampled measurement model in Eqs. (67) and (68), and the implicit transmission time constraints in Eqs. (69) and (70), it is straightforward to develop a truth-model simulation and to design a Kalman filter based on this paper’s principles. This has been done. The remainder of this section discusses the results of applying the new filter to the corresponding simulated data.

The SRIF residual statistics offer a good indicator of the fidelity with which the piecewise polynomial process noise model approximates the underlying continuous-time random processes used to generate the truth states. If the filter’s model is reasonable, then the SRIF residuals will be samples from a zero-mean white-noise distribution. Equivalently, the SOS of the SRIF residuals at each filtering step should be drawn from a \( \chi^2_n \) distribution with \( n = N_m \) degrees of freedom, where \( N_m \) is the number of measurements processed during each filtering interval. Figure 11 shows a time history of the SRIF residuals for an example case. The mean of the residuals over simulation time is -0.0010, and the variance is 1.0402, both indicating a close
match to the expected statistics. The autocorrelation function associated with this plot has been computed, and it closely resembles the required Dirac delta function form.

Next, three different simulation scenarios, each with a different number of measurements on each filtering interval, $N_m$, have been processed by three different filters, each with a different polynomial process noise model order, $M$. Figure 12 shows the normalized SOS of the SRIF residuals for each scenario and each filter. The groups of bars indicate filters of a particular polynomial order, i.e., groups of bars for $M = 0, 2,$ and $4$. Each bar in a particular group indicates the normalized SOS of the SRIF residuals over a simulation time. The SOS values are divided by the total number of measurements processed in the scenario in order to normalize them.

There are several important aspects to this plot. First, it is obvious that the magnitude of the normalized SOS increases for a given process noise polynomial order as the number of measurements is increased, i.e., the bars in each group tend to get larger as the number of measurements is increased. This makes sense since the normalized SOS is a measure of how well the model fits the data. As the amount of data increases, finding a good fit becomes more challenging, and the normalized SOS of the residuals increases. At the same time, the accuracy of the estimate typically improves due to the extra information that the additional data provides. This claim is supported by looking at the filter’s computed steady-state estimation error standard deviations in Table 3. Those standard deviations exhibit notable improvement as the measurement density
is increased.

Table 3. For $M = 4$, variations of computed estimation error standard deviations with the number of measurements per integration interval, $N_m$.

<table>
<thead>
<tr>
<th>$N_m$</th>
<th>$\sigma_r$ (m)</th>
<th>$\sigma_v$ (m/s)</th>
<th>$\sigma_a$ (m/s$^2$)</th>
<th>$\sigma_{\delta t_{\text{clk}}}$ (m)</th>
<th>$\sigma_{\delta f_{\text{clk}}}$ (m/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5.3209e-01</td>
<td>1.4214e-03</td>
<td>1.5859e-06</td>
<td>7.9075e-00</td>
<td>6.9407e-02</td>
</tr>
<tr>
<td>6</td>
<td>4.7947e-01</td>
<td>1.2809e-03</td>
<td>1.4291e-06</td>
<td>7.1255e-00</td>
<td>6.2543e-02</td>
</tr>
<tr>
<td>10</td>
<td>1.6423e-01</td>
<td>1.7157e-04</td>
<td>7.9398e-08</td>
<td>6.9355e-01</td>
<td>5.1747e-03</td>
</tr>
</tbody>
</table>

A better measure of the filters’ performance is to look at their capacity to fit a particular amount of data as the polynomial order is increased. This behavior is revealed by noting how bars associated with a particular number of measurements change height as the polynomial order is increased. For example, the relative height of the first bar in each group indicates how the polynomial order changes the filter’s ability to fit the data given $N_m = 2$. For all such comparisons, the filters’ capacity to fit the data increases as the polynomial order increases. The trend is most pronounced, however, when the measurement density is highest. For a more direct comparison, Fig. 13 reorganizes the groupings by the number of measurements rather than the order of the polynomial model and the color coding of the bars is changed to reflect the order of the polynomial. Thus, each grouping of three bars shows how the fit improves as the polynomial
process noise model order $M$ increases for a fixed value of $N_m$. The downward trend in each grouping as $M$ goes from 0 to 4 is the important feature of this plot. The downward trend is most pronounced for $N_m = 10$ measurements per integration interval, as expected. This trend suggests that the additional fit parameters offered by the higher order piecewise polynomial process noise models provides the filter increased capability to fit dense data over a given filtering interval.

V. Conclusions

This paper has presented a new approximation to a continuous-time sampled-data Kalman filter that is appropriate for systems with implicitly defined measurement times. The method is useful for any system in which such measurements arise, including many radio navigation and tracking problems. The filter uniquely combines dense output numerical propagation with a finite dimensional approximation of the underlying continuous-time white process noise in a joint propagation/measurement update step that can process multiple, implicitly constrained measurements. The filter’s models have been presented and linearizations of those models have been derived. The data processing that produces a posteriori state and process noise parameter estimates has been explained. A new statistical model for the finite dimensional approximation to white process noise has been developed and demonstrated in simulation, and the filter has been tested on
Figure 13. The relative effect of the order of the piecewise polynomial process noise, \( m \), for three different numbers of measurements per integration interval, \( N_m \).

A simple radio navigation tracking problem by using a truth-model simulation.

The underlying continuous-time white noise is approximated with a piecewise polynomial that is a randomly weighted sum of basis polynomial functions over any given numerical integration interval. In its simplest form, it is equivalent to a zero-order-hold discrete-time white-noise model. The random polynomial weighting coefficients are treated as discrete-time random variables that are constant for a particular filtering interval and that are drawn from a zero-mean white-noise sequence. The covariances for these variables are chosen to be scaled versions of the continuous-time noise intensity. The scaling factors, which are chosen such the piecewise polynomial process noise optimally approximates the behavior of continuous-time white noise, have been obtained by solving a quadratic programming problem. They are presented for models that use Chebyshev polynomials basis functions with orders \( M = 0, 1, \ldots, 10 \). Simulated piecewise polynomial process noise has been analyzed, and its statistics agree closely with the statistics predicted by the theory. A theoretical analysis has demonstrated that piecewise polynomials of higher order better approximate continuous-time white noise.

A truth-model simulation based on a simple radio navigation tracking problem has tested the performance of the new Kalman filter. An analysis of the SRIF residual statistics has shown that the new Kalman filter yields a statistically reasonable approximation to the original problem. Further, the simulations have shown
that, for a given measurement density, filters with higher-order polynomial process noise models provide better fits to the data. This conclusion has been reached by analyzing the sums of the squares of the SRIF residuals for scenarios with varying measurement density and polynomial order.

Appendix: A Candidate Dense Output Runge-Kutta Method

Runge-Kutta methods are a class of numerical integration routines that only require an initial state and a dynamics model to propagate a state over an interval. Such methods, which are referred to as single-step methods, are particularly well suited for use in sequential estimation. They propagate a system’s state by evaluating its rates according to a dynamics model at several intermediate points over the interval and by combining those function evaluations with the initial state. The result is an approximation of the state at the end of the interval. Consider an example $N_{rk}$-step explicit Runge-Kutta method defined by the following two equations:

\[ \xi_i = f[t_k + c_i \Delta t, x(t_k) + \Delta t \sum_{j=1}^{i-1} a_{ij} \xi_j] \quad \text{for} \quad i = 1, 2, \ldots, N_{rk} \tag{71} \]

\[ x(t_k + \Delta t) \approx x(t_k) + \Delta t \sum_{i=1}^{N_{rk}} b_i \xi_i \tag{72} \]

In Eq. (71), $\xi_i$ is the $i^{th}$ function evaluation at the intermediate time $t_k + c_i \Delta t$ and at the intermediate state approximation $x(t_k) + \Delta t \sum_{j=1}^{i-1} a_{ij} \xi_j$. The specific choices of the intermediate times and state approximations are defined by the coefficients $c_i$ and $a_{ij}$, respectively. In Eq (72), the final state approximation equals the initial state plus a weighted sum of the function evaluations. The weighting is defined by the parameters $b_i$ and the step size. The particular choices of the parameters in these equation, which define the specific Runge-Kutta method and ensure its level of accuracy, are unimportant to this development and are not discussed here. For a more thorough discussion, please refer to Refs. 1 and 4. The important point here is that the method’s only output is the final state approximation.

In some instances, the user not only requires an approximate state at the end of the integration interval but also at intermediate points over the interval. Numerically propagating to intermediate points, however, may be computationally expensive or even numerically ineffectual if the intermediate step size becomes very small. Dense output Runge-Kutta methods, sometimes referred to as continuous methods, offer a solution to this problem. In addition to outputting the final state approximation, these methods output the function evaluation values, $\xi_i$. When these values are scaled by the step size and by specially designed normalized polynomial coefficients, they represent coefficients for polynomials that describe the state’s evolution continuously on the integration interval. The normalized coefficients are designed in conjunction with the method’s other parameters such that the resulting polynomials satisfy the method’s order condition over the
entire integration interval. Importantly, the continuous approximation of the state is obtained cheaply, i.e.,
computing the necessary coefficients requires none or few additional function evaluations. For example, in
addition to the original $N_{rk}$ function evaluations, the dense output version of the method in Eqs. (71) and
(72) may require additional function evaluations of the form

$$\xi_i = f[t_k + c_i \Delta t, x(t_k) + \Delta t \sum_{j=1}^{i-1} a_{ij} \xi_j] \quad \text{for} \quad i = N_{rk} + 1, N_{rk} + 2, \ldots, N_{rk} + P_{rk}$$

These values are used in the following formula to approximate the state at any intermediate point on the
integration interval:

$$x(t_k + \theta \Delta t) \approx x(t_k) + \Delta t \sum_{i=1}^{N_{rk}+P_{rk}} b_i^*(\theta) \xi_i$$

Here, $\theta$ is a continuous variable between 0 and 1, and $b_i^*(\theta)$ is the $i^{th}$ normalized polynomial in $\theta$, which take
the form

$$b_i^*(\theta) = \sum_{j=1}^{Nd} d_{ij} \theta^j \quad \text{for} \quad i = 1, 2, \ldots, N_{rk} + P_{rk}$$

An important consideration is how these polynomials are used in the extended Kalman filter presented in
this paper. The filter not only requires a continuous approximation of the system’s state over each integration
interval, but also of the state transition matrix, a continuous version of Eq. (5), i.e.,

$$\Phi(t, t_k) = \frac{\partial x(t; \Sigma_k)}{\partial x_k}$$

and the process noise coefficient influence matrix, Eq. (14). One way to evaluate these matrices is to
analytically differentiate the dense output Runge-Kutta method’s polynomials with respect to the initial
state, $x_k$, and the polynomial process noise coefficients, $\tilde{w}_k$. Another method, the one used in this study,
recognizes that the desired matrices are the solutions to the initial value problems defined in Eqs. (6) and
(9) and thus may be approximated by the same dense output Runge-Kutta method used to propagate the
system’s state. Such an approach makes sense computationally since many or all of the most expensive
calculations, i.e., the function evaluations, are computed for the state propagation. As it turns out, the two
methods can be shown to be equivalent if the solutions to the differential equations for $x(t)$, $\Phi(t, t_k)$, and
$\Gamma_j(t, t_k)$ for $j = 0, 1, \ldots, M$ are computed in one unified set of dense output Runge-Kutta calculations.

The specific dense output Runge-Kutta method used in this study is from Ref. 8. It is a six-stage, fifth-
order explicit method that requires a seventh stage for the dense output calculations. The coefficients of
this method are presented in standard form in Table 4. Its normalized polynomial interpolant in $\theta$ takes the
Table 4. Coefficients for a dense output Runge-Kutta numerical integration method.

<table>
<thead>
<tr>
<th>i</th>
<th>c_i</th>
<th>a_{ij}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1/5</td>
<td>1/5 0</td>
</tr>
<tr>
<td>3</td>
<td>3/10</td>
<td>3/40 9/40 0</td>
</tr>
<tr>
<td>4</td>
<td>4/5</td>
<td>44/45 -56/15 32/9 0</td>
</tr>
<tr>
<td>5</td>
<td>8/9</td>
<td>19372/6561 -25360/2187 64448/6561 -212/729 0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>9017/3168 -355/33 46732/5247 49/176 -5103/18656 0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>35/384 0 500/1113 125/192 -2187/6784 11/84</td>
</tr>
</tbody>
</table>

following form:

\[ b^*_1(\theta) = \theta^2(3 - 2\theta)b_1 + \theta(\theta - 1)^2 - 5\theta^2(\theta - 1)^2(2558722523 - 31403016\theta)/11282082432 \]  
\[ b^*_2(\theta) = 0 \]  
\[ b^*_3(\theta) = \theta^2(3 - 2\theta)b_3 + 100\theta^2(\theta - 1)^2(88272551 - 15701508\theta)/32700410799 \]  
\[ b^*_4(\theta) = \theta^2(3 - 2\theta)b_4 - 25\theta^2(\theta - 1)^2(443332067 - 31403016\theta)/1880347072 \]  
\[ b^*_5(\theta) = \theta^2(3 - 2\theta)b_5 + 32805\theta^2(\theta - 1)^2(23143187 - 3489224\theta)/199316789632 \]  
\[ b^*_6(\theta) = \theta^2(3 - 2\theta)b_6 - 55\theta^2(\theta - 1)^2(29972135 - 7076736\theta)/822651844 \]  
\[ b^*_7(\theta) = \theta^2(\theta - 1) + 10\theta^2(\theta - 1)^2(7414447 - 829305\theta)/29380423 \]

Notice how the coefficients of these polynomials ensure that when \( \theta = 0 \), \( b^*_j = 0 \) for all \( j \), and that when \( \theta = 1 \), \( b^*_j = b_j \) for all \( j \), as is expected for consistency between the standard and dense output.

References