Prepared by Tom Kerr for AIAA/BMDO in July 2001 in Williamsburg, VA.

Comparison of EKF and Batch Least Squares Approaches to UEWR Target Tracking

Thomas H. Kerr, Ph.D., Consultant
TeK Associates
Lexington, MA 02420

Haywood Satz, Technical Staff
Raytheon Company
Bedford, MA 01730

Abstract

Tracking accuracy is compared using two alternative approaches for handling the radar target tracking of exo-atmospheric objects in ballistic flight (representative of satellites and reentry vehicles). The received radar returns, measuring range and angle (in radar sine space), are nonlinear observations of the target moving along an expected elliptical path, as viewed from the Cartesian Coordinates erected within the face plane of the observing UHF phased array radar used for Early Warning so that range-Doppler ambiguity can be appropriately compensated.

The two practical nonlinear filtering algorithms being investigated here are distinguished by the drastically different way the associated sequence of noisy measurements are processed. The first approach, using the well-known Extended Kalman Filter (EKF), processes measurements one-at-a-time as an in-place, sequential recursive estimator of the target's instantaneous position and velocity states as a function of time. The second approach processes the entire collection of measurements available over the time epoch of interest en masse using the Batch Least Squares (BLS) technique to yield a time history estimate of the target's entire trajectory in one fell swoop.

The BLS algorithm generally exhibits dramatically smaller estimation errors and more realistic truthful covariances obtainable on-line than does the EKF. The CPU loading of these alternative tracking approaches is gauged and discussed here as well as the operational benefits and drawbacks of each. Both are useful in different roles within the UEWR mission. An update is offered on the utility of Squareroot filters in this type of application.

1. Introduction

The fundamental estimation accuracy that can be achieved by an optimal, ideal sequential tracking estimator, as gauged in terms of Cramer-Rao Lower Bounds (CRLB) [1], was applied to the UEWR mission and extensively quantified and catalogued for a collection of 30+ trajectories of significance, as representatively reported in [3]. An overview of the multitudinous cross-considerations that arise for UEWR were presented in [4], including a summary of the status of a number of tracking related initiatives pursued during the earlier HAVE GOLD phase that ended in 1998 prior to the more up to date studies reported on here, as now evaluated with a more refined more evolved DoD sanctioned simulator.

A comparison is made here of radar target tracking performance between two different approaches to approximate nonlinear filtering of the noisy range and angle measurements of exo-atmospheric objects in ballistic motion. Elliptical motion is exhibited by satellites and by reentry vehicles (rv's). In both, object position and velocity are governed by the nonlinear dynamics of body motion in a central force inverse squared gravity field (Sec. 2.2). Since we additionally include considerations of the second harmonic J_2 for realism to account for the oblateness of the earth (Sec. 2.3), its presence induces two more characteristic motions known as the "regression of nodes'' and the "rotation of Apsides'' [5]-[7]. The radar range measurements are modeled as nonlinear observations of the target motions, as viewed from Earth-fixed Cartesian coordinates [6] erected in the face plane of the UEWR in order to compensate the range-Doppler ambiguity (Sec. 2.4). Afterwards, the motion is related back to the Earth Centered Inertial (ECI) frame [8] to simplify differencing the ground truth from the estimates in tallying the estimation errors incurred (where simplification is due to absence of Coriolis cross-product terms in this frame).

Both algorithms use the same 1^{st}-variation approximations of the nonlinear observations and error dynamics of object motion in common (as the two Jacobians explicitly defined, respectively, in Secs. 2.2 and 2.4). With these approximations in effect, the EKF then implements the familiar linear-minimum variance (Kalman) estimation formulas. The Raytheon BLS examined here, in contrast to other regression algorithms previously formulated for exactly this same type of early warning radar defense application (such as the 1981 Lincoln Laboratory whole value maximum likelihood least squares batch formulation [further pursued by Mission Research Corp. for HAVE GOLD in 1997] and the more recent whole-value batch formulation [9] which doesn't directly model the earth's oblateness), is unique in that it is an incremental

differential formulation and also optimizes its least-squares search over a functional variation, where each candidate function is indexed as a potential target state-vector-path over the given tracking time interval. The result of Raytheon having such a novel well-formulated BLS algorithm is that it provides estimates of generally superior accuracy than those availed by these other whole-value least squares formulations and all are generally better than using just an EKF (Sec. 5).

Results were obtained for both algorithms performing in a comprehensive mission simulator developed by XonTech Inc., where radar system errors are represented in detail. These representations include radar-cross-section and scan angle models, signal-to-noise computations, and radar front-end noise models. For realism, evaluations on one platform (designated to be the Test Driver/Software Algorithm Testbed [TD/SAT]) also included a fluctuating target model and realistic antenna nulls. The BLS algorithm was seen to converge faster and predict more accurate statistical bounds than the EKF in most situations. For the same length of data record, the accuracy of the BLS estimates is better (i.e., tighter and closer to true) than those available from the EKF for the same track at the same point in time but the drawback (Sec. 4) is that it possesses a greater CPU burden that grows with the length of target data of the same ID that BLS is processing. The utility of both in different roles within the UEWR mission is summarized in Sec. 6

2. Representation of the linearized target dynamics and measurement errors

2.1 Notation for UEWR target track estimation via EKF and/or via BLS:

Target tracking time epoch (or time interval of interest from initial to final):	$[t_o, t_f]$
Target position, target velocity, vector of target model dynamics:	$\mathbf{p}, \mathbf{v}, \mathbf{f}(\mathbf{x})$
Target state vector representation (3 position, 3 velocity components):	$\mathbf{x}^T(t) = [\mathbf{p}^T \mid \mathbf{v}^T] = [x_{1,\ldots,}x_6]$
Earth's rotation, gravity, its J_2 harmonic (effect due to earth's oblateness):	$\Omega, \mathbf{g}, \mathbf{a}_2$
Initial condition for target state estimate:	$\hat{\mathbf{x}}(t_o)$
Initial condition for target state estimate, after correction, denoted by:	$[\hat{\mathbf{x}}(t_o)]^+$
(at just an instant later after the corrective offset has been added) as:	$[\hat{\mathbf{x}}(t_o)]^+ = \hat{\mathbf{x}}(t_o) + \delta\mathbf{x}(t_o)$
Identification number provided within a multitarget tracking context:	ID or Target ID
Measurements of r, x/r, y/r (in face coordinates):	$\mathbf{z}(t) = [z_r(t), z_u(t), z_v(t)]^T$
Collection of all radar measurements available for a unique target ID:	$\mathbf{Z}^m = \{\mathbf{z}(t_o), \mathbf{z}(t_1), \ldots, \mathbf{z}(t_m)\}$
State estimate conditioned on prior measurements received up to t_m:	$\hat{\mathbf{x}}(t) = E\{\mathbf{x}(t) \mid \mathbf{Z}^m\}$
Estimation error:	$\delta\mathbf{x}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$
Error transition matrix (i.e., $\delta\mathbf{x}(t) = \Phi(t, t_o)\delta\mathbf{x}(t_o)$):	$\Phi(t, t_o)$
Earth Centered ECR-to-radar Face coordinate rotation:	$[C_e^f(t)] = [\mathbf{c}_1(t) \mid \mathbf{c}_2(t) \mid \mathbf{c}_3(t)]^T$
Observation model (as arises in measurements $\mathbf{z}(t) = \mathbf{h}(\mathbf{x}(t)) + \text{noise}$):	$\mathbf{h}(\mathbf{x}(t))$
Both Jacobians, as 1st derivatives of $\mathbf{h}(\mathbf{x}(t))$ and $\mathbf{f}(\mathbf{x}(t))$, respectively, wrt x:	$H(\mathbf{x}(t)) = [\partial\mathbf{h}(x)/\partial\mathbf{x}]\mid_{\hat{x}(t)}$
[where Jacobian is evaluated about the currently available estimate $\hat{\mathbf{x}}(t)$].	$F(\mathbf{x}(t)) = [\partial\mathbf{f}(x)/\partial\mathbf{x}]\mid_{\hat{x}(t)}$
Reciprocal of 1-σ (standard deviation) of measurement error:	$[\sigma_z(t)]^{-1}$
Range-to-target magnitude, range-to-target unit vector:	$r(t), \mathbf{u}(t) = \mathbf{r}(t)/r(t)$
Range/Doppler coupling coefficient:	(Tf_o/β)
(T is signal duration or pulse width, f_o is LFM center frequency, β is LFM pulse bandwidth)	
Threshold of Least Square Error (LSE) residual variation:	ε
Important intermediate BLS arrays:	$\begin{cases} T_{rk}(j) = t \text{ (current time tag)} \\ \bar{X}_{rk}(:,j) = \bar{X} \text{ (current time BLS estimate)} \\ \Phi_{rk}(:,:,j) = \phi \text{ (current transition matrix)} \end{cases}$
Parameter representing the maximum j index for the above three arrays (i.e., number of points to be included within Runge-Kutta integration):	RKMAX
Parameter representing maximum number of iterations allowed (for each call to the BLS algorithm)	LMAX
Abbreviation of State Variables: S.V.	

2.2 S.V. Modeling of Target Error Dynamics

The dynamic model governing ballistic, exo-atmospheric target motion, is:

$$d\mathbf{p}/dt = \mathbf{v}, \quad (1)$$

$$d\mathbf{v}/dt = \mathbf{g} - \Omega x(\Omega x\mathbf{p}) - 2\Omega x\mathbf{v}, \quad (2)$$

and dynamics of the whole value filter state, $\mathbf{x}^T(t) = [\mathbf{p}^T\ \mathbf{v}^T]$, and its associated estimation errors,

$$\delta\mathbf{x}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t) \quad (3)$$

are defined as follows: $\mathbf{x}^T(t) \equiv [\mathbf{p}^T \mid \mathbf{v}^T]$, $\hat{\mathbf{x}}(t) \equiv E\{\mathbf{x}|\mathbf{Z}^m\}$,

$$\delta\mathbf{x}(t) \equiv \mathbf{x}(t) - \hat{\mathbf{x}}(t) = \Phi(t, t_o)\delta\mathbf{x}(t_o), \quad (4)$$

$d\hat{\mathbf{x}}/dt = \mathbf{f}(\hat{\mathbf{x}}, t; t_o)$, with initial condition $\hat{\mathbf{x}}(t_o) = \hat{\mathbf{x}}_o$, (5)

$d\Phi/dt = [\partial\mathbf{f}(\hat{\mathbf{x}})/\partial\mathbf{x}]\Phi$, with initial condition $\Phi(t_o) = \mathbf{I}_n$. (6)

The time histories of $\hat{\mathbf{x}}(t)$ and $\Phi(t, t_o)$ are obtained by simultaneous Runge-Kutta solution of their differential equations (Eqs. 1 and 2) for some choice of integration step-size over the tracking interval, $[t_o, t_m]$. Interpolation approximations of those time histories are used in the synthesis of the regression arrays defined in Sec. 3. Initial state error, $\delta\mathbf{x}(t_o)$, is estimated by least-squares (Sec. 3), and the prior estimate of initial state, $\hat{\mathbf{x}}(t_o)$, is updated with a new estimate of initial state error after each BLS iteration as:

$$[\hat{\mathbf{x}}(t_o)]^+ = \hat{\mathbf{x}}(t_o) + \delta\mathbf{x}(t_o) \quad (7)$$

2.3 Introducing the effect of gravity harmonic J_2 (due to earth's oblateness)

Within these equations, the ECR acceleration **a** is to be recomputed at each new time step as:

$\rho \equiv$ vector from earth geocenter to radar face center,

$$\mathbf{r}_o = \rho + \mathbf{r} + \mathbf{v} \cdot \left(\frac{\Delta T}{2}\right), \quad (8)$$

$\mu \equiv$ gravitational constant $= 3.986005 \times 10^{14}\ m^3/s^2$,

$$\mathbf{a}_g = \frac{-\mu}{r_o^3}\mathbf{r}_o, \text{ (inverse square term)}, \quad (9)$$

$$\mathbf{a}_{cent} = -\Omega \times (\Omega \times \mathbf{r}_o) \text{ (centrifugal term)}, \quad (10)$$

$$\mathbf{a}_{cor} = -2\Omega \times \mathbf{v} \text{ (Coriolis term)}, \quad (11)$$

$J_2 \equiv$ second harmonic of geo-potential (a constant)

$= 1.08263 \times 10^{-3}$,

$r_e \equiv$ geocentric (equatorial) earth radius = 6378.135 km,

$L \equiv$ earth latitude,

$\mathbf{N} \equiv$ unit vector from geo-center to North pole, expressed in radar face Cartesian coordinates with the origin at the earth's center $= \Omega/|\Omega|$,

$\sin^2 L = \dfrac{(\mathbf{r}_o^T\mathbf{N})^2}{r_o^2}$, (relationship between latitude and parameters) (12)

$\mathbf{a}_2 \equiv$ acceleration due to second spherical harmonic of gravity [7]

$$= \frac{\mu J_2}{r_o}\left(\frac{r_e}{r_o}\right)^2\left[\left(\frac{15(\mathbf{r}_o^T\mathbf{N})^2}{2r_o^4} - \frac{3}{2r_o^2}\right)\mathbf{r}_o - \frac{3(\mathbf{r}_o^T\mathbf{N})}{r_o^2}\mathbf{N}\right]$$

(cf. [40, Eq. 7]) (13)

$$\mathbf{a} = \mathbf{a}_g + \mathbf{a}_{cent} + \mathbf{a}_{cor} + \mathbf{a}_2. \quad (14)$$

The 3-dimensional vector ρ is constant for a specific radar face.

2.4 UEWR as a data measurement sensor (with range-Doppler coupling compensation)

The measurement model of the form:

$$\mathbf{z}(t) \equiv [z_r, z_u, z_v]^T = \mathbf{h}(\mathbf{x}), \quad (15)$$

to be explicitly defined below, is used to first predict measurements and then to form the measurement residual. The appropriate expression to be used by the radar to partially compensate the matched filter tuned to the transmitted Linear Frequency Modulated radar "chirp" (internal to the radar receiver) using the range/Doppler cross-coupling coefficient term, (Tf_o/β), is defined from:

$\mathbf{p}_r \equiv$ radar site position relative to Earth's center in ECR,

$\mathbf{r} = \mathbf{p} - \mathbf{p}_r$ (true range vector to object); (16)

$\mathbf{u} = \mathbf{r}/r$ (unit vector), (17)

$\dot{r}_{PD} \equiv$ range-rate from range-Doppler pre-distortion compensation signal processor,

$t_i \equiv i^{th}$ measurement time-tag,

$z_r(t_i) \equiv$ measured range,

$z_u(t_i) \equiv$ horizontal projection of measured range in radar face,

$z_v(t_i) \equiv$ vertical projection of measured range in radar face,

$\hat{\mathbf{z}}(t_i) \equiv [\hat{z}_r, \hat{z}_u, \hat{z}_v]^T = \mathbf{h}(\hat{\mathbf{x}}(t_i))$ (predicted mea-surements); (18)

$\hat{\mathbf{u}} = \hat{\mathbf{r}}/\hat{r}$ (19)

$\hat{z}_r(t_i) = \hat{r} + [Tf_o/\beta](\hat{\mathbf{v}}^T\hat{\mathbf{u}} - \dot{r}_{PD})$ (20)

$$\hat{z}_u(t_i) = [\mathbf{c}_1(t_i)]^T \hat{\mathbf{u}} \quad (21)$$

$$\hat{z}_v(t_i) = [\mathbf{c}_2(t_i)]^T \hat{\mathbf{u}} \quad (22)$$

$$\delta\mathbf{z}(t_i) = \mathbf{z}(t_i) - \hat{\mathbf{z}}(t_i) = \mathbf{z}(t_i) - \mathbf{h}(\hat{\mathbf{x}}(t_i)) \quad \text{(residuals)}. \quad (23)$$

In summary, the measurements, as functions of the state, are modeled here according to [15] in terms of radar location, $\mathbf{p}_r$, target range, $\mathbf{r}$, rows, $[\mathbf{c}_i(t_i)]^T$, of the ECR-to-Face coordinate rotation, $[C_e^f(t_i)]$, and the range/Doppler cross-coupling quotient (Tf_o/β) to be:

$$z_r = r + (Tf_o/\beta)\mathbf{v}^T\mathbf{u} \quad (24)$$

$$z_u = [\mathbf{c}_1(t_i)]^T(\mathbf{r}/r) = [\mathbf{c}_1(t_i)]^T\mathbf{u} \quad (25)$$

$$z_v = [\mathbf{c}_2(t_i)]^T(\mathbf{r}/r) = [\mathbf{c}_2(t_i)]^T\mathbf{u} \quad (26)$$

which, in vector notation using $\mathbf{z} \equiv [z_r, z_u, z_v]^T$, completes the detailed specification of Eq. 15.

Computing first variations piecewise with respect to first the position $\mathbf{p}$ and then the velocity $\mathbf{v}$, the measurement residual sensitivities to both these components of the state are found, respectively, to be of the following form:

$$\partial z_r/\partial\mathbf{p} = \partial r/\partial\mathbf{p} + (Tf_o/\beta)\mathbf{v}^T\partial\mathbf{u}/\partial\mathbf{p} = \mathbf{u}^T + (Tf_o/\beta)\mathbf{v}^T(\mathbf{I}_3 - \mathbf{u}\mathbf{u}^T)/r \quad (27)$$

$$\partial z_u/\partial\mathbf{p} = [\mathbf{c}_1]^T\partial\mathbf{u}/\partial\mathbf{p} = [\mathbf{c}_1]^T(\mathbf{I}_3 - \mathbf{u}\mathbf{u}^T)/r = ([\mathbf{c}_1]^T - z_u\mathbf{u}^T)/r \quad (28)$$

$$\partial z_v/\partial\mathbf{p} = [\mathbf{c}_2]^T\partial\mathbf{u}/\partial\mathbf{p} = [\mathbf{c}_2]^T(\mathbf{I}_3 - \mathbf{u}\mathbf{u}^T)/r = ([\mathbf{c}_2]^T - z_v\mathbf{u}^T)/r \quad (29)$$

and

$$\partial z_r/\partial\mathbf{v} = \partial\{(Tf_o/\beta)\mathbf{v}^T\mathbf{u}\}/\partial\mathbf{v} = (Tf_o/\beta)\mathbf{u}^T \quad (30)$$

$$\partial z_u/\partial\mathbf{v} = \mathbf{0}^T \quad (31)$$

$$\partial z_v/\partial\mathbf{v} = \mathbf{0}^T. \quad (32)$$

Both the above components can be combined into a single composite $\partial\mathbf{z}/\partial\mathbf{x} = [\partial\mathbf{h}(\mathbf{x})/\partial\mathbf{x}] = \mathrm{H}(\mathbf{x})$, which yields the final linear observation sensitivity matrix (measurement Jacobian) as the goal:

$$\mathrm{H}(\mathbf{x}) \equiv \left[\ \partial\mathbf{h}(\mathbf{x})/\partial\mathrm{x}\ \right] = \frac{\partial\mathbf{z}}{\partial\mathbf{x}} = \begin{bmatrix} \partial z_r/\partial\mathbf{x} \\ \partial z_u/\partial\mathbf{x} \\ \partial z_v/\partial\mathbf{x} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{u}^T + (Tf_0/\beta)\mathbf{v}^T(\mathbf{I}_3 - \mathbf{u}\mathbf{u}^T)/r & (Tf_0/\beta)\mathbf{u}^T \\ (\mathbf{c}_1^T(t) - z_u\mathbf{u}^T)/r & \mathbf{0}^T \\ (\mathbf{c}_2^T(t) - z_v\mathbf{u}^T)/r & \mathbf{0}^T \end{bmatrix}. \quad (33)$$

3. Estimate of BLS Implementation Needs

A brief high level summary of the Raytheon Batch Least Squares (BLS) algorithm, along with salient aspects of its internal structure that are exploited within BLS, is offered now since it may be less familiar than the structure of a standard EKF that appears fairly frequently in the literature for this type of application [9]-[15], [22], [25]-[29], [36]-[41]. However, least squares algorithms have been investigated before [31]-[33] but never before compared so closely to an EKF. (For other precedents, see chap. 15 of Example Book associated with [16].) Differences between measurements and their predictions, $\mathbf{z} - \hat{\mathbf{z}}$, at the same time point t_i constitute the measurement residuals, which exhibit the following structure:

$$\mathbf{z}(t_i) - \hat{\mathbf{z}}(t_i) = \mathbf{z}(t_i) - \hat{\mathbf{h}}(\mathbf{x}(t_i)) \quad (34)$$

$$\cong \mathbf{z}(t_i) - \mathbf{h}(\hat{\mathbf{x}}(t_i)) - \{\partial\mathbf{h}(\hat{\mathbf{x}}(t_i))/\partial\mathbf{x})\}[\mathbf{x}(t_i) - \hat{\mathbf{x}}(t_i)] \quad (35)$$

$$= \mathbf{z}(t_i) - \mathbf{h}(\hat{\mathbf{x}}(t_i)) - \mathrm{H}(\hat{\mathbf{x}}(t_i))\delta\hat{\mathbf{x}}(t) \quad (36)$$

$$= \delta\mathbf{z}(t_i) - \mathrm{H}(\hat{\mathbf{x}}(t_i))\Phi(t_i, t_o)\delta\hat{\mathbf{x}}(t_o). \quad (37)$$

The chronological sequence of reciprocal 1σ-weighted or normalized set of coefficient matrices are aggregated or adjoined by placing each beneath its predecessor for each of the times, t_i, for which measurements are available. The resulting composite rectangular coefficient array, $\mathbf{A}$(3mx6), has the internal structure (cf., Eq. 63 of [2]) summarized here on the right as:

$$\mathbf{A} \equiv \begin{bmatrix} [\sigma_z(t_0)]^{-1}\,\mathbf{H}(\hat{\mathbf{x}}(t_0))\Phi(t_0) \\ [\sigma_z(t_1)]^{-1}\,\mathbf{H}(\hat{\mathbf{x}}(t_1))\Phi(t_1) \\ \vdots \\ [\sigma_z(t_m)]^{-1}\,\mathbf{H}(\hat{\mathbf{x}}(t_m))\Phi(t_m) \end{bmatrix}. \quad (38)$$

and the corresponding aggregate column array $\mathbf{Z}$(3mx1) is likewise formed by adjoining the correspondingly normalized (reciprocal 1-σ-weighted) set of measurement residuals, each (3x1) grouping adjoined beneath its predecessor according to the same ascending time tags, t_i, as:

$$\delta\mathbf{Z} \equiv \begin{bmatrix} [\sigma_z(t_0)]^{-1}\,\delta\mathbf{z}(t_0) \\ [\sigma_z(t_1)]^{-1}\,\delta\mathbf{z}(t_1) \\ \vdots \\ [\sigma_z(t_m)]^{-1}\,\delta\mathbf{z}(t_m) \end{bmatrix} \quad (39)$$

where, accompanying the measurements as input to the BLS algorithm, we also have the measurement noise standard-deviations, $\sigma_r(t_i)$, $\sigma_u(t_i)$, and $\sigma_v(t_i)$, which define the appropriate normalizing or scaling factors that were already utilized above, as shown.

The residual model introduced above is used in least-squares iterations as the criterion for halting the iterations, where convergence of $|\mathbf{z} - \hat{\mathbf{z}}|$ to a minimum (denoted $|\delta\mathbf{Z}|$) is sought as $\delta\mathbf{x}(t_o)$ approaches zero. The specific rule for either stopping or continuing the iterations is based on a test for convergence, by gauging how the magnitude of the square of the residual differences is proceeding as compared to its immediate

history one iteration back: If this is the first pass through the regression loop, then set $|\delta\mathbf{Z}|^2_{last} = |\delta\mathbf{Z}|^2$ and continue iterating; else if $|\,|\delta\mathbf{Z}|^2_{last} - |\delta\mathbf{Z}|^2\,| \geq \varepsilon$, then set $|\delta\mathbf{Z}|^2_{last} = |\delta\mathbf{Z}|^2$ and continue iterating; else if $|\,|\delta\mathbf{Z}|^2_{last} - |\delta\mathbf{Z}|^2\,| < \varepsilon$, then calculate BLS covariance and output it and the current estimate and stop.

3.1 The B matrix input to BLS

The target object's fundamental physical data, captured as the measurements received from the radar, serve as inputs upon which the BLS algorithm operates. For convenience, this input data is encapsulated as an Aggregate Batch Measurement Matrix, B, being (9 x m) with entries as explicitly defined below for each enumerated row designator:

1: $t_k \equiv$ time (seconds) [must be monotonically non-decreasing from column 1 to column m];

2: $t_{rd} \equiv$range-Doppler coupling time (in sec.) used as compensation for this observed radar effect;

3: $I_f \equiv$ index of radar face (being either 1, 2, or maybe 3, depending on the radar site);

4: r $\equiv$ range measurement (km) from target to radar face at sample time, t_k;

5: u $\equiv$ direction cosine measurement (dimensionless) from target to radar face at sample time, t_k;

6: v $\equiv$ direction cosine measurement (dimensionless) from target to radar face at sample time, t_k;

7: $\sigma_r \equiv$ standard deviation (km) of range measurement at sample time, t_k;

8: $\sigma_u \equiv$ standard deviation (dimensionless) of u-direction cosine measurement at sample time, t_k;

9: $\sigma_v \equiv$ standard deviation (dimensionless) of v-direction cosine measurement at sample time, t_k, where, in the above, the three quantities (r(t), u(t), v(t)) are measured according to the standard radar convention in sine space and where there are exactly m measurements available that constitute the columns of this B matrix.

3.2 Overview of BLS's processing flow

The criteria of optimality is the minimization of a cost function consisting of the sum of the squares of the normalized errors (gauged in terms of the entire collection of measurement-error-weighted residuals of the form $|[\sigma_z(t)]^{-1}\delta\mathbf{z}(t)|$ using the reciprocal standard deviations: $[\sigma_z]^{-1} = \mathrm{diag}[1/\sigma_r, 1/\sigma_u, 1/\sigma_v]$ as the normalizing weightings). The error is calculated between results obtained using the propagated target trajectory (obtained as an intermediate calculation evolving from an initial state estimate $\hat{\mathbf{x}}(t_o)$) and the aggregate of measurement data received $\{\mathbf{z}(t_o), \mathbf{z}(t_1),..., \mathbf{z}(t_m)\}$, consisting of all the associated individual radar return realizations collected or logged over the entire specified tracking interval $[t_o, t_m]$ for a single identified target ID.

Both the candidate initial hypothesized $\hat{\mathbf{x}}(t_o)$ and its associated error correction $\delta\mathbf{x}(t_o)$ are to then be further processed and refined to determine the least-squares estimate of this combined initial state vector after being provided all these measurements. The associated underlying regression iterations are to be tested for the simultaneous convergence of the estimation residual evaluation $|\delta\mathbf{Z}|^2$ as defined just prior to Sec. 3.1. In order to obtain final results, a fundamental linearized system of algebraic matrix equations (posed as a nominal plus offset error formulation) is to be iteratively solved using a Householder transformation as the crux of this BLS approach:

$$[\delta\mathbf{Z}(t_i)] = [\mathbf{A}(t_i)]\delta\mathbf{x}(t_o). \qquad (40)$$

This linearized system is to be solved yet again on each BLS iteration for the indicated initial state estimate correction $\delta\mathbf{x}(t_o)$, where available aggregate quantities portrayed above summarize the fundamental underlying relationship of Eq. 37 by using standard familiar regression array notation: $\mathbf{A}(t_i) \equiv [\sigma_z]^{-1}[\partial\mathbf{h}(\hat{\mathbf{x}}(t_i))/\partial\mathbf{x}]\Phi(t_i)$ and likewise denoting the entire collection of measurements as $\delta\mathbf{Z}(t_i) \equiv [\sigma_z]^{-1}[\mathbf{z}(t_i) - \mathbf{h}(\hat{\mathbf{x}}(t_i))]$, respectively, as already defined in Eqs. 38 and 39. The two step Householder transformation-based solution procedure consists of: (1) first forming an upper triangular matrix as an intermediate result [by systematically annihilating all elements in the columns below the principal diagonal of the normalized matrix of measurement derived data]; (2) then, via standard "back substitution", solving for the unknown state estimate $\delta\mathbf{x}(t_o)$ [corresponding to providing a refined initial condition estimate at the starting time via Eq. 7].

The result of the above optimization may be summarized as an initial state vector "whole value" estimate availed at the starting time for the time epoch of interest. This, in turn, is propagated to the end time of interest along with an associated covariance of estimation error as the primary outputs. After all refining intermediate iterations cease, the LSE error covariance that emerges, denoted below as $[\mathbf{A}_H{}^T\mathbf{A}_H]^{-1}$, associated with the final estimate $\hat{\mathbf{x}}(t_m)$ is computed as output error covariance: $\mathrm{Cov}_{BLS} = [\mathbf{A}_H{}^T(t)\,\mathbf{A}_H(t)]^{-1}$ and

$$\Phi(t_f, t_o)\cdot\mathrm{Cov}_{BLS}\cdot\Phi^T(t_f, t_o) \text{ or, equivalently, as: } E\{(\delta\hat{\mathbf{x}})(\delta\hat{\mathbf{x}})^T\} = \Phi(t_m, t_o)[\mathbf{A}^T(t_m)\mathbf{A}(t_m)]^{-1}\,\Phi^T(t_m, t_o). \qquad (41)$$

(The number of sample points availed as radar return measurements depends on actual acceleration magnitudes experienced by the target and the length of the time interval over which it is viewed, so array sizes associated with BLS are not known precisely in advance. However, they can be upper bounded using the parameter RKMAX.)

3.3 Summary of the sequence of BLS operations in this joint evaluation versus that of EKF

1. Prior to any track estimation, initialize EKF with a'priori estimate $\hat{\mathbf{x}}(t_o)$.
2. Save EKF update of prior estimate, $[\hat{\mathbf{x}}(t_o)]^+$, and all subsequent measurements, $\mathbf{Z}^m$.
3. Request Batch Filter execution at $t=t_m$ and input initial estimate $[\hat{\mathbf{x}}(t_o)]^+$and measurements $\mathbf{Z}^m$.
4. Define LSE model, $\delta\mathbf{Z}(t) = [\sigma_z]^{-1}[\partial\mathbf{h}(\hat{\mathbf{x}})/\partial\mathbf{x})]\Phi(t, t_o)\delta\mathbf{x}(t_o) = \mathbf{A}(t)\delta\mathbf{x}(t_o)$.
5. Use successive Least Square iterations, each with an update of the initial state error $\delta\mathbf{x}(t_o)$, with revised initial state as $[\hat{\mathbf{x}}(t_o)]^+ = [\hat{\mathbf{x}}(t_o)]^+ + \delta\mathbf{x}(t_o)$.
6. Iterate over Steps 3 to 5 above until convergence is achieved or until LMAX is exceeded.
7. Upon satisfying the stopping criterion, then output number of iterations, indicate how looping terminated, and output answers as final estimate: $\hat{\mathbf{x}}(t_f)$ and corresponding error covariance.

The interpolation step discussed above was replaced during the later timing studies of Sec. 4 (performed on a desktop PC) with use of more frequent invocation of Runge-Kutta integrations.

Either the prior direct EKF output information (dynamically propagated back to the start of the interval over which UEWR measurement processing is to be initiated as a "cold start") or indirect prior EKF outputs (obtained externally as part of the prior solution to the Lambert problem [30], which corroborates with earlier target sightings from SBIRS [43], [44]) are used to provide a better more accurate initial estimate designated below as an "initial seed" for the batch algorithm (thus equipping the BLS with an initial full estimated trajectory upon which it will operate to improve this estimate). However, to obtain all the improvement being sought requires performing a number of internal optimizing iterations as already called for within the algorithm.

4. Estimate of BLS Implementation Needs

A gauge of anticipated mission CPU time was made using the Fortran intrinsic function CPU_TIME to measure the latency or computational delay in running the BLS algorithm on a desktop PC using Compaq Digital Visual Fortran ver.6 (an F90 Fortran compiler), Windows NT-4 operating system with service pack SP-3, 128M RAM memory, and a 266 MHz clock. The test scenario consisted of a single target of constant radar-cross-section (with antenna scan-loss present) at a starting range of about 2500 km with a minimum signal-to-noise ratio of 10 dB but increasing to nominally be a SNR of ~13 dB. Use of a non-fluctuating target in this manner gives a better idea of the actual algorithm timing under full worst case loading when no measurements are missed or dropped.

The normalized time interval expended on the PC, per iteration, per measurement, was approximately 0.00035 seconds. It is estimated that for the anticipated planned (April 2000) hardware, the embedded version of this same algorithm should execute several (at least 3 to 4) times faster than the desktop version. The numbers quoted here were obtained for just one Monte-Carlo run for one scenario (D13.1).

A parallelized version could be even faster but that is a different architecture. Only sequential Von Neumann implementations were considered here. A parallelized version of the Householder transformation, at the heart of BLS, may be found in [42]. Lincoln Laboratory investigated this systolic array version of a parallel Householder algorithm in the late 1980's for radar signal processing but later converted to implement a cordic algorithm based on Givens rotations.

A parallel processing Fortran reference [16] indicates that interpolation benefits from implementing on parallel processors by greatly speeding up the attainment of the goal. Unfortunately, the clean lines of the original BLS algorithm were altered to replace interpolation with more frequent integrations between the available measurements that are logged. This version of BLS, written up for the MAS SRS spec, presumes only a sequential Von Neumann implementation since we could not accurately guess how a multi-threaded version would be partitioned across processors and performance also depends on target platform capabilities, its OS, and on compiler settings.

Another possible variation on the original BLS is to use SVD in place of the Householder's transformation when solving the fundamental system of linear equations at the heart of the BLS algorithm. When possible, a "Householder transformation" should be used over SVD in well-behaved situations since Householder constitutes a lesser computational burden (being a voltage or squareroot method) than that of SVD (which is a power method, as revealed in [13, App. B]). However, the vagaries of real data in some applications may warrant use of SVD for robustness.

An estimate of the algorithm-allocated memory for its internally defined variables is about 600 8-byte words or about 5,000 bytes. If an object is to be tracked over an interval of 2 minutes at a frequency of 1 Hz, then the required memory allocation could be as much as

(120)(5,000) or 600 K bytes (assuming no measurements are missed or dropped).

BLS processes all the available measurements en masse and requires several iterations (albeit a small number, nominally 2) to do so. (The number of BLS iterations previously encountered for the interpolated version ranged as high as 8 but was nominally 3.) The number of measurements processed from this segment of the D13.1 scenario as a test case was not extremely large. Other scenarios can have much longer lengths of data to contend with. Intermediate arrays of transition matrices depicted in the symbol notation of Sec. 2.1, as well as regression arrays such as those in Eqs. 38 and 39, as well as the fundamental (9 by m) Aggregate Batch Measurement Matrix B defined in Sec. 3.1 for BLS must be accounted for in 8 byte double precision for these variable arrays for each track ID in process, where m is the number of measurements available for each track ID. As m increases, so must these intermediate arrays increase within the BLS algorithm. Memory for all but the above (9 x m) BLS B-matrix array can be released afterwards and re-allocated where needed. The actual BLS program code itself remains as a known fixed size.

Regarding sensitivity to the radar data that it is operating on, an EKF will follow any radar measurements it receives while the BLS tries to fit all the received data en masse to its internal model. The BLS internal model assumes that all the data be exclusively from a ballistic regime and that it be homogeneously pure by not being tainted with mis-associations from other tracks (which otherwise degrade BLS).

Timing above was performed for a situation that was already known to converge. The real world is not so accommodating as to allow us to know the situation beforehand. Conservatism would dictate use of the worst-case number of iterations, LMAX, that can be incurred as a multiplier of these per measurement timing estimates. While the in-place EKF processes measurements one-at-a-time so CPU-time-per-a-measurement is a valid criteria for an EKF, the BLS does not process measurements one-at-a-time so timing estimates portrayed only this way are, at best, an intermediate approximation for BLS. For BLS, it is all or nothing in terms of measurement processing so CPU time should be interpreted for the whole measurement set that is being processed to be reasonable. BLS uses "all the measurements available to it all the time" when it is invoked.

The appropriate final CPU timing number that is unraveled here now decomposes naturally into two cases. The first case is the simplest and corresponds to when BLS is merely called once (as in the UEWR MAS SRS). There is an upper bound worse case for this situation of 25 x m x (0.00035 seconds). In nominally benign situations (the prevalent case), the above timing expression holds with 2 replacing 25 as the appropriate pre-multiplying factor. These two expressions immediately above with alternate prefixes, can be used to bracket the actual CPU time from above and below. Use of both bounds together in this way is more conservative for performing predictions because the user does not have to know what the actual situation is beforehand regarding number of iterations to be incurred for BLS to converge for an upcoming trial.

The second case involves a slightly more complex expression and corresponds to when BLS is called repeatedly at a known, fixed periodic rate. In this situation too, there is an upper bound worse case as:

$$25\cdot(0.00035)\cdot\left\{\left(\sum_{i=1}^{\left[\frac{m}{r}\right]} i\right)\cdot r\right\}=$$

$$25\cdot(0.00035)\cdot\left\{\left[\frac{m}{r}\right]\left(\left[\frac{m}{r}\right]+1\right)\cdot\frac{r}{2}\right\}. \qquad (42)$$

In the preceding expression, m is as defined in Sec. 3.1 and r is the period at which BLS is automatically invoked (at the desktop PC, or within TD/SAT or in later simulators if implemented the same way) and the brackets in the upper limit of the finite summation and within the expression on the right hand side (where the prior sum is simplified) denotes the smallest integer portion of the resulting division indicated to be performed within the brackets. Here LMAX = 25.

To see how this expression was obtained, first consider the case for measurements being processed by BLS at a periodic rate where BLS is invoked every 10 measurements (where at each invocation, all the measurements logged since the beginning for this object ID are reprocessed by BLS). The CPU loading looks like the following dot diagram representing batches of measurements processed by the BLS algorithm each time it is invoked:

.....
....
....
....

For the first 40 data measurement points, where BLS was invoked after 10, after 20, after 30, and after 40, the total number of data points processed after 40 is 10 + 20 + 30 + 40 = 100. This is 100 times the measured individual per measurement CPU times reported above. At the 47th measurement, the remainder now processed

is nominally no more than at 40 since the big burden of BLS processing is not invoked until at 50.

Tallying the above dot diagram yields: $\left\{\sum_{i=1}^{\left\lceil \frac{47}{10} \right\rceil} i\right\} \cdot 10 = 100$. A useful formula is

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2}. \qquad (43)$$

Using this in the above CPU loading factor for a periodically invoked BLS yields a loading of:

$$\left\{\sum_{i=1}^{\left\lceil \frac{47}{10} \right\rceil} i\right\} \cdot 10 = \frac{\left\lceil \frac{47}{10} \right\rceil \left(\left\lceil \frac{47}{10} \right\rceil + 1 \right)}{2} 10 = 100$$

which generalizes to:

$$\left\{\sum_{i=1}^{\left\lceil \frac{k}{c} \right\rceil} i\right\} \cdot c = \frac{\left\lceil \frac{k}{c} \right\rceil \left(\left\lceil \frac{k}{c} \right\rceil + 1 \right)}{2} \cdot c. \qquad (44)$$

The per measurement normalization for BLS utilized above appears to be appropriate and consistent with numerical analysis theory for this main potential bottleneck apparently facing BLS. As already mentioned above, the main problems being solved at the heart of each BLS iteration is the solution of a system of linear equations (the array of regression equations). Recall that this is the crux or fundamental kernel and the Householder transformation is used to solve it (as the algorithm of least computational complexity, which accomplishes the task at hand). Parallel implementations should be no slower than these estimates for a Von Neumann machine and parallel multi-threaded implementations may be considerably faster.

Operations Counts are available for a perfectly implemented *sequential* version of the Householder transformation from page 148 of [35]. The operations count incurred in applying back-substitution as Householder's transformation is being applied to solve the linear problem [ignoring for the moment any considerations related to obtaining the BLS covariance]), is: O(mn) flops, where n = 6 and m is the total number of measurements. For this aspect, averaging by dividing the previous expression by m to obtain a per measurement normalization yields the constant n= 6. The CPU burden is merely linear in the number of measurements and consistent with the above criteria selected of "per measurement evaluations". Since we also need the explicit upper triangular matrix in order to calculate the BLS covariance matrix (but don't need an explicit representation of the matrix transformation that gets us there), the numerical complexity of a Householder transformation in this case is greater for this aspect but still only linear in the measurements and still consistent with the criteria selected for conveying CPU time above. Also from page 148 of [35], the expression for the Householder operations counts in this case of providing an explicit U matrix is: $(n^2 m - n^3/3)$ flops, where n = 6 and m is the total number of measurements. This operations count goes as m (the dominant power) and again just grows linearly with m. Averaging by dividing the previous expression by m to obtain an expected per measurement normalization yields a constant based on this numerical analysis theory. A similar invocation of a Householder transformation per a measurement depicted on page 252 of [13] also obtained a constant that is a cubic in the remaining fixed variable, being n^3. All are consistent.

5. Representative EKF vs. BLS Performance in UEWR Mission Simulator (TD/SAT)

The graph of Figs. 1 and 2 show the estimation error time histories of both the Batch (BLS) and RVCC EKF filter [10] tracking a single RV within the D13.1 scenario of multiple objects in exo-atmospheric free-fall-trajectories. The radar sensitivity in this simulation was such that the tracking signal-to-noise ratio was a minimum of 10 dB at a range of about 2500 km.

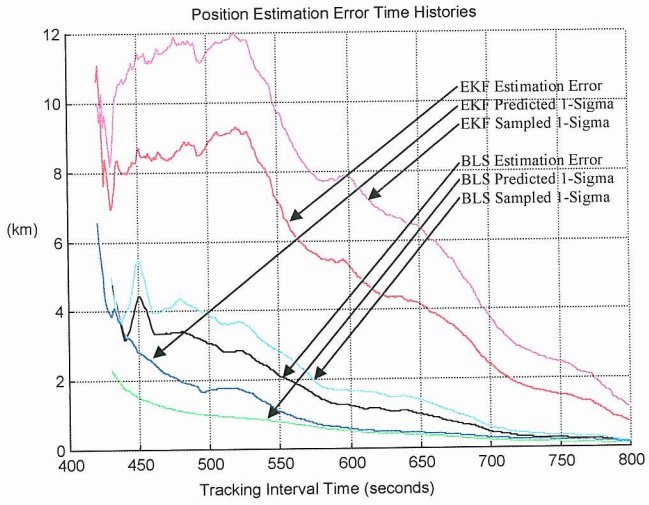

Figure 1: Estimated Position Error Time Histories for Batch and EKF Algorithms

Batch algorithm estimation of radar object tracking was compared with EKF tracking of the same object. Comparisons were made in a desktop simulation of an object in exo-atmospheric free-fall. The radar was modeled by the range equation with constant RCS and pulse-width.

Observations were represented by models of monopulse range and angle measurement errors in terms of SNR. Comparisons were also made in a more comprehensive mission simulator which included Earth's mass distri-

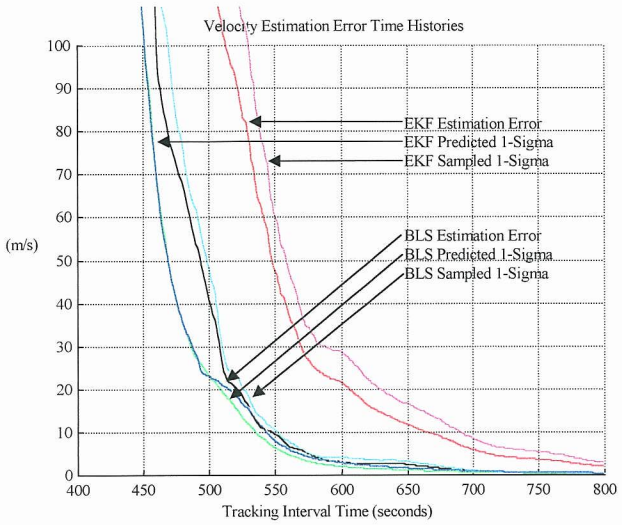

Figure 2: Estimated Velocity Error Time Histories for Batch and EKF Algorithms

bution and rotation, fluctuating targets, antenna gain patterns, RF noise, resource allocation and scheduling, and multiple-target detections-to-tracks association. The mission simulator generated a 100 Monte-Carlo sequence of tracking interval time histories to lend further support to the ten-trial desktop results (more came later).

Both the desktop and the mission simulators showed that, in comparison with EKF, the Batch algorithm converged faster, more accurately, and closer to its own self-assessed 1-sigma value.

6. Benefits of BLS for the UEWR Mission and Summary of BLS Characteristics

The superior state vector estimation accuracy of the Batch algorithm, in particular the more precise velocity estimate, is a direct benefit to the UEWR portion of the mission because it:

- enables earlier launch of the interceptor merely by satisfying accuracy guidelines sooner and provides more accurate in-flight-target-update (IFTU) for the kinetic vehicle;
- provides better support for phase-ambiguity-resolution and contextual feature discrimination;
- estimates better orbital elements for space object identification, and results in better satellite vs. missile discrimination;
- Offers better launch and impact accuracy for legacy early warning functions;
- is potentially more robust in the ionospheric scintillation, range, and Doppler error environment (see caveat below);
- should save 6 dB of radar energy in comparison with conventional EKF object tracking.

Regarding the first item above, the more accurate (non-optimistic) on-line prediction of 1-sigma BLS bounds helps properly constrain the region that the interceptor needs to search for target acquisition. Use of an optimistic bound in this role would result in limiting search to too small a volume of space and therefore risk missing the target although supporting theoretical numerical calculations would falsely assure success (because they expect the available 1-sigma to be trustworthy, which it is not in general for an EKF).

The EKF immediately avails outputted estimates in a more timely fashion and will follow any measurement data that it is provided with. The EKF is appropriate to use with the data association algorithm for multi-target tracking (MTT) because it is a fixed lesser CPU burden. On the other hand, the BLS algorithm provides more accurate estimates with a higher fidelity (more trustworthy) on-line computed covariance accompanying its estimates for the same data segment length. Estimation errors of about 1 km are predicted from the on-line computation of 1-sigma bounds while the actual value is at the level of 8 km. The Batch algorithm on-line calculation predicts error bounds of a similar magnitude, however, BLS pays off by actually realizing errors of 1.5 km which are in the same vicinity.

However, BLS use incurs a larger computational burden and more associated senescence (computational delay time that is not fixed) than exhibited or needed by an EKF (which has a delay time for computed output that is *fixed* and *known* [12, Sec. 7.6]). The *known* but *not fixed* BLS total processing demands always grow with the amount of measurement data collected for the particular track ID.

As stated above, BLS provides greater accuracy and a better (more trustworthy) on-line 1-sigma gauge of what the accuracy is but is more sensitive and may not converge (i.e., may not produce any useable output if LMAX is exceeded because no ballistic curve could be fit to it) when the measurement data record is tainted.

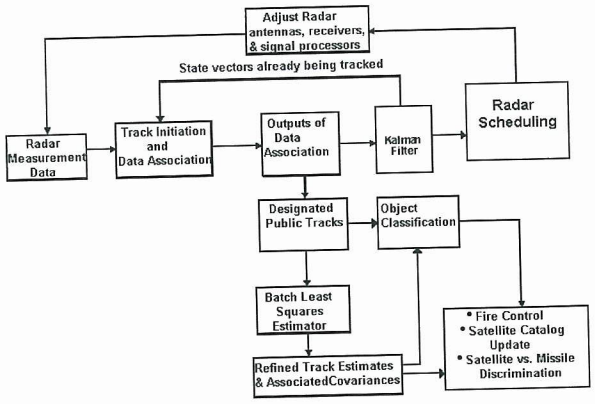

Figure 3: Simplified High Level Overview of the UEWR Processing Architecture

The worrisome tainting may be either by mis-associations, or by failure to correctly prune away the boosting segment, or by later thrusting if it comprises a significant portion of the data record, or by the effects of ionospheric scintillation. BLS expects data which it assumes matches its internal model; otherwise, it may fail to converge. This is less of a problem the longer the measurement data segment is that BLS is provided with to operate on, as long as the dominant regime it represents is ballistic (and ionospheric scintillation errors have been approximately compensated for, as planned, or either are not dominant). The good news is that this better situation for BLS use will occur naturally if BLS is only applied to more mature tracks of interest (which corresponds to those with a longer data collection record).

There is still room for improvement of the EFK itself either via inclusion of more terms in the approximating Taylor series of the measurements [23], known as the Hessian; or by including a few additional iterations (2 or 3) of the measurement linearization [22]. Both these strategies should improve the accuracy of the measurement linearization with but a slight increase in the CPU burden. Another approach involving the Hessian for handling measurement structures like this involving a direct measurement of range [39] is actually adaptive. Other options are to use different degrees of decoupling in the initial covariance (and Kalman gain as a consequence) or to pursue exquisite analytic variations of EKF and its creative generalizations offered in [45]. Other more challenging AOT filter issues arise [25] when escort jammers accompany RV's.

Appendix A: Updated Considerations Regarding Bierman's Squareroot Filtering

Squareroot filtering is a convenient and practical contrivance used to obtain an effective double precision implementation of a Kalman filter without having to actually resort to explicit implementation in double precision but merely by use of an alternate implementation (in single precision) of the factors of the covariance matrix being propagated forward in time. The so-called Bierman's form or U-D-U^T form of squareroot filtering [13] (which propagates U and a diagonal D) had historically proved to be the best formulation up until the late 1990's in that it is a numerically stable implementation (an important consideration for long duration on-line run times) and has the smallest number of required operations and does not call for the implementation of explicit scalar squareroots (as earlier squareroot implementations did). Earlier versions of squareroot filtering constituted much larger computational burdens than conventional Kalman filtering; however, the Bierman implementation was no worst a computational burden than the standard KF implementation but offers an effective doubling in the precision availed (and with guarantees of numerical stability [which a standard Kalman Filter implementation lacks]). For more detail, see Chap. 7 (and, in particular, the comparisons of Table 7.1) on p. 403 of [46]. Other important considerations in U-D-U^T squareroot filltering are addressed in [47], which shows how to rigorously handle vector measurements in U-D-U^T filters. Ref. [48] discusses how to capitalize on certain efficiencies that arise in Navigation applications involving GPS as an external navaid for an Inertial Navigation System (INS). Ref. [49] demonstrates how to incorporate handling of manual updates (known by military pilots and navigators as "MARK ON TOP" as a procedure that uses landmarks of known location to update position in an aircraft's internal INS filter at the more or less "precise moment" that the aircraft flies over the landmark). Ref. [50] was one of the first designs that clearly demonstrated the details of how to accommodate a U-D-U^T filter formulation within a NAVSTAR GPS application.

Bierman's U-D-U^T Squareroot filter formulation was preceded by or evolved from other efforts at deriving squareroot filter formulations by J. Potter (1964), J. F. Bellantoni and K. W. Dodge (1967), A. Andrews (1968), P. Dyer and S. McReynolds (1969), P. G. Kaminski and A. E. Bryson and S. I. Schmidt (1971), W. S. Agee and R. H. Turner (1972), and N. Carlson (1973). Bierman's formulation (1974, '75, '77) had originally proved to be the best of the lot for embedded architectures where explicit scalar square root extraction was a much more time consuming algorithm until computer architectures surfaced in the late 1990's where this was no longer the case and now scalar squareroot calculation is about the same as a floating point

multiply thus now favoring Carlson's [51] over Bierman's formulation. There are also other recent contenders and logical extensions [52]-[55].

For situations where the discrete-time dynamic state variable system model is of the form:

$$x(k+1) = A\,x(k) + F\,w(k) + B\,u(k), \text{ with initial condition: } x(0) = x_0 \quad (45)$$

with (optional) deterministic control input (i.e., exogenous input) being present and the discrete-time sensor data measurement observation model is of the form:

$$z(k) = C\,x(k) + G\,v(k), \quad (46)$$

where w(k) and v(k) are independent Gaussian white noises (GWN) with intensity variances of Q and R, respectively. For the purpose of further reducing the adverse effect of round-off error accumulation and to avoid explicit calculation of the matrix inverse within the Kalman filter by using a degenerate scalar form that is, instead, only a division, it is frequently desired to update squareroot filters using only one-scalar-measurement-component-at-a-time, but the standard procedure for doing so is only valid if R is diagonal (corresponding to uncorrelated measurement noise) and G is the identity matrix. In the more general case where both of these conditions fail to be met yet the user still wants to update the filter one scalar measurement component at a time, the following simple (possibly time-varying) transformation can be applied to achieve the desired structure for single-component-at-a-time updating. Merely form $[G(k)\,R(k)\,G^T(k)]$ and decompose it via a Choleski decomposition into $[G(k)\,R(k)\,G^T(k)] = W(k)\,W^T(k)$, where W(k) is lower triangular, then just pre-multiply the entire measurement equation above to obtain $z_1(k) \equiv W^{-1}(k)z(k) = [W^{-1}(k)C(k)]\,x(k) + [W^{-1}(k)G(k)]\,v(k)$, and we have $[W^{-1}(k)G(k)\,R(k)\,G^T W^{-1}(k)] = W^{-1}(k)W(k)W^T(k)W^T(k) = I_{mxm}$, where I_{mxm} is the mxm identity matix. The original Kalman filter, described recursively by the following eqn., driven by the measurement z(k) and control:

$$\hat{\mathbf{x}}(k+1|k) = \Phi(k+1,k)\,[I - K(k)\,C]\,\hat{\mathbf{x}}(k|k-1) + \Phi(k+1,k)\,K(k)\,z(k) + B\,u(k) \quad (47)$$

and with on-line propagate covariance of estimation error equation being:

$$P(k|k) = [I - K(k)\,C]\,P(k|k-1)\,[I - K(k)\,C]^T + K(k)\,G\,R_3\,G^T\,K(k)^T \quad (48)$$

and with on-line update covariance of estimation error equation being:

$$P(k|k-1) = \Phi(k,k-1)\,P(k-1|k-1)\,\Phi^T(k,k-1) + F\,Q_3(k)\,F^T, \text{ (then k=k+1)} \quad (49)$$

with the standard discrete-time Kalman gain being:

$$K(k) = P(k)\,C^T\,[C\,P(k|k-1)\,C^T + G\,R_3\,G^T]^{-1}. \quad (50)$$

The above four equations are now modified for one-component-at-a-time filtering as the following equivalent Kalman filter driven by the transformed measurement $z_1(k)$ and same deterministic control u(k), respectively, as:

$$\hat{\mathbf{x}}(k+1|k) = \Phi(k+1,k)\,[I - K'(k)\,W^{-1}\,C]\,\hat{\mathbf{x}}(k|k-1) + \Phi(k+1,k)\,K'(k)\,z_1(k) + B\,u(k) \quad (51)$$

and

$$P(k|k) = [I - K'(k)\,W^{-1}\,C]\,P(k|k-1)\,[I - K'(k)\,W^{-1}C]^T + K'(k)\,K'^T(k) \quad (52)$$

and

$$P(k|k-1) = \Phi(k,k-1)\,P(k-1|k-1)\,\Phi^T(k,k-1) + F\,Q_3(k)\,F^T, \text{ (then k=k+1)} \quad (53)$$

with the new discrete-time Kalman gain being:

$$K'(k) = P(k)\,C^T W^{-T}\,[W^{-1}C\,P(k|k-1)\,C^T W^{-T} + I]^{-1}. \quad (54)$$

Hint: If noises w(k) and v(k) are present but matrices F and/or G are not apparent in the defining system and measurement models; then, obviously, $F = I_{nxn}$ and $G = I_{mxm}$.

Again, the new wrinkle of the late 1990's in Square Root Filtering is that within new processor chips, the algorithm for performing explicit scalar squareroots is no longer an iterative mechanization but now is just as fast as multiplication or addition operations. The prior motivation to use a particular version of Square Root Filtering, based on operation counts that penalized explicit computation of scalar squareroots is no longer viable for implementation processors that calculate the squareroot this new way. Motivation still exists to use a Square Root Filtering structure for real-time implementations with long run times because these squareroot formulations are still the only numerically stable implementation of a Kalman filter. Its use avoids such contrivances as inserting stops to prevent any main diagonal terms of the covariance matrix from eventually becoming negative as a consequence of adverse effects of accumulated round-off errors within the more straight forward implementation of the easy-to-read conventional Kalman filter (although possibly "Stabilized" by adding the transposed covariance to itself and dividing by 2). Every navigation application should be using a Square Root Filter formulation in the case of long run times. However, target tracking for strategic missiles may not require such stringent mechanizations because main targets of interest (RVs) don't persist for long enough time intervals to warrant Square Root Filter use. New targets trigger new filter starts. Implementation needs for radar tracking of persistent cooperative FAA targets can be a different story. The longer "Control Segment" tracking intervals for GPS satellite ephimerii drift definitely do use Square Root Filter formulations. Evidently, whether or not to use Bierman's U-D-U^T or Carlson's squareroot filter formulation in a particular application should be decided on a case-by-case basis.

The usual benefits touted for the use of a U-D is that (1) it is numerically stable (it mitigates the build-up of round-off errors), while other formulations devoid of square root filtering are not numerically stable; (2) it

exposes problematic effects early on as evidenced by algorithmically examining the principal diagonals of the propagated matrix D at each time step, and (3) it effectively doubles the effective precision of the on-line computed covariances and the associated estimator's gain as a consequence (although this last reason is usually the primary reason for using a U-D formulation, it may not be necessary in some applications where the register size is already adequate). The CPU burden for a U-D-U square root filter is no greater than that of an ordinary Kalman filter but cross-checking proper implementation is a little more challenging because it's output estimates and covariances are usually compared to those of a non-square root filter for the short term (when both should be identical) but only the U-D square root formulation will be adequate for the long term of longer missions and frequent measurements being obtained. Finally, there is even a parallel processing implementation of it available [56].

Acknowledgements

We credit Peter Bancroft's (Raytheon) pioneering in originating and first implementing this BLS and Frederick Daum's (Raytheon) prior precedents with BLS in recognizing its utility in Early Warning Radar tracking beyond just maintaining a satellite catalog and in initially evaluating it for UEWR. The later software instantiations of the BLS and EKF algorithms run on a desktop PC by Haywood S. Satz (Raytheon) were refined from earlier versions of the PC code converted from the Unix version by Pham Thong (Raytheon); however TeK Associates participated jointly in debugging the original Unix version [26]-[29] run by Daniel Pulido (General Dynamics), in specifying and interpreting the trials depicted here, in helping to quantify the computer burden incurred, and in writing the corresponding sections of the software specification MAS SRS. Thanks go to Roger Reed (Raytheon) and William M. Stonestreet (Raytheon) and his predecessor Ronald Hettich (Raytheon) for expediting and overseeing our work.

REFERENCES

1. Kerr, T. H., "Status of CR-Like Lower bounds for Nonlinear Filtering," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-25, No. 5, pp. 590-601, Sept. 1989 (Author's reply in Vol. AES-26, No. 5, pp. 896-898, Sept. 1990).
2. Kerr, T. H., "A New Multivariate Cramer-Rao Inequality for Parameter Estimation (Application: Input Probing Function Specification)," *Proceedings of IEEE Conference on Decision and Control*, Phoenix, AZ, pp. 97-103, December 1974.
3. Kerr, T. H., "Developing Cramer-Rao Lower Bounds to Gauge the Effectiveness of UEWR Target Tracking Filters," *Proceedings of AIAA/ BMDO Technology Readiness Conference and Exhibit*, Colorado Springs, CO, 3-7 August 1998.
4. Kerr, T. H., *UEWR Design Notebook-Section 2.3: Track Analysis*, TeK Associates, Lexington, MA, (for XonTech, Hartwell Rd, Lexington, MA), XonTech Report No. D744-10300, 29 March 1999.
5. Brown, C. D., *Spacecraft Mission Design*, AIAA Education Series, NY, 1992.
6. Bate, R. R.,, Mueller, D. D., White, J. E., *Fundamentals of Astrodynamics*, Dover Publications, NY, 1971.
7. Battin, R. H., *An Introduction to the Mathematics and Methods of Astrodynamics*, AIAA Education Series, NY, 1987.
8. Olson, D. K., "Converting Earth-Centered, Earth-Fixed Coordinates to Geodetic Coordinates," *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 32, No. 1, pp. 473-475, Jan. 1996.
9. Hough, M. E., "Improved Performance of Recursive Tracking Filters using Batch Initialization and Process Noise Adaptation," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 5, pp. 675-681, Sept.-Oct. 1999.
10. Wishner, R. P., Larson, R. E., and Athans, M., "Status of Radar Tracking Algorithms," *Proceedings of Symposium on Nonlinear Estimation Theory and Its Applications*, San Diego, CA, pp. 32-54, 1970.
11. Chang, C. B., and Tabaczynski, J. A., "Application of State Estimation to Target Tracking," *IEEE Transactions on Automatic Control*, Vol. AC-29, No. 2, pp. 98-109, February 1984.
12. Grewal, M. S., and Andrews, A. P., *Kalman Filtering: Theory and Practice*, Prentice-Hall, Upper Saddle River, NJ, 1993.
13. Bierman, G. J., *Factorization Methods for Discrete Sequential Estimation*, Academic Press, 1977.
14. Daum, F. E., Fitzgerald, R. J., "Decoupled Kalman Filters for Phased Array Radar Tracking," *IEEE Transactions on Automatic Control,* Vol. AC-28, No. 3, pp. 264-283, March 1983.
15. Fitzgerald, R. J., "Effects of Range-Doppler Coupling on Chirp Radar Tracking Accuracy," *IEEE Transactions on Aerospace and Electronic Systems,* Vol. AES-10, No. 4, pp. 528-532, July 1974.
16. Press, W. H., Teukolsky, S. A., et al, *Numerical Recipes in Fortran 90: The Art of Parallel Scientific Computing*, Vol. 2, 2nd Edition, Cambridge University Press, NY, 1999.

17. Kerr, T. H., "Computational Techniques for the Matrix Pseudoinverse in Minimum Variance Reduced-Order Filtering and Control," in *Control and Dynamic Systems-Advances in Theory and Applications*, Vol. XXVIII: Advances in Algorithms and computational Techniques for Dynamic Control Systems, Part 1 of 3, C. T. Leondes (Ed.), Academic Press, 1988.
18. Kerr, T. H., "Multichannel AR Modeling for the Active Decoy (U)," MIT Lincoln Laboratory Report No. PA-499, Lexington, MA, March 1987 (Secret).
19. Kerr, T. H., "Multichannel Shaping Filter Formulations for Vector Random Process Modeling Using Matrix Spectral Factorization," MIT Lincoln Laboratory Report No. PA-500, Lexington, MA, 27 March 1989 (BMO limited distribution).
20. Kerr, T. H., "Emulating Random Process Target Statistics (using MSF)," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-30, No. 2, pp. 556-577, April 1994.
21. Kerr, T. H., "On Duality Between Failure Detection and Radar/Optical Maneuver Detection," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-25, No. 4, pp. 581-583, July 1989.
22. Kerr, T. H., "Streamlining Measurement Iteration for EKF Target Tracking," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-27, No. 2, March 1991 (minor correction appears in Nov. 1991 issue).
23. Satz, H. S., Cox, D. B., and Beard, R. L., "GPS Inertial Attitude Estimation Via Carrier Accumulated Phase Measurements," *Navigation: Journal of the Institute of Navigation*, Vol. 38, No. 3, pp. 273-284, Fall 1991.
24. Kerr, T. H., "A Critical Perspective on Some Aspects of GPS Development and Use," *Proceedings of 16th Digital Avionics System Conference,* Vol. II, pp. 9.4-9 to 9.4-20, Irvine, CA, 26-30 Oct. 1997.
25. Kerr, T. H., "Assessing and Improving the Status of Existing Angle-Only Tracking (AOT) Results," *Proc. of the Inter-national Conf. on Signal Processing Applications & Technology (ICSPAT)*, Boston, MA, pp. 1574-1587, 24-26 Oct. 1995.
26. Kerr, T. H., "Rationale for Monte-Carlo Simulator Design to Support Multichannel Spectral Estimation and/or Kalman Filter Performance Testing and Software Validation & Verification Using Closed-Form Test Cases," MIT Lincoln Laboratory Report No. PA-512, Lexington, MA, 22 December 1989 (BSD [previously BMO] limited distribution).
27. Kerr, T. H., "A Constructive Use of Idempotent Matrices to Validate Linear Systems Analysis Software," *IEEE Trans. on Aerospace and Electronic Systems*, Vol. AES-26, No. 6, pp. 935-952, November 1990 minor correction in Nov. 1991 issue.
28. Kerr, T. H., "Numerical Approximations and Other Structural Issues in Practical Implementations of Kalman Filtering," a chapter in *Approximate Kalman Filtering*, edited by Guanrong Chen, 1993.
29. Kerr, T. H., and Satz, H., S., "Applications of Some Explicit Formulas for the Matrix Exponential in Linear Systems Soft-ware Validation," *Proceedings of 16th Digital Avionics System Conf.*, Vol. I, pp. 1.4-9 to 1.4-20, Irvine, CA, 26-30 Oct. '97
30. Kerr, T. H., "Considerations in whether to use Marquardt Nonlinear Least Squares vs. Lambert Algorithm for NMD Cue Track Initiation (TI) calculations," TeK Associates, Lexington, MA, (for Raytheon, Sudbury, MA), 27 September 2000.
31. Fang, B. T., "A Nonlinear Counterexample for Batch and Extended Sequential Estimation Algorithms," *IEEE Transactions on Automatic Control*, Vol. 21, No. 1, pp. 138-139, Feb. 1976.
32. Fang, B. T., "Iterated batch least-squares and extended sequential estimation: convergence to different estimates," *IEEE TAC*, pp. 138-139, Feb. 1976.
33. Fang, B. T., "Linear weighted least-squares estimator: proof of minimum variance property," *IEEE TAC*, pp. 765-766, Dec. 1969.
34. Vetter, W. J., "Linear estimation with a priori information: minimum variance and least-squares criteria," *IEEE TAC*, pp. 265-266, June 1971.
35. Golub, G. H., and Van Loan, C. F., *Matrix Computations*, John Hopkins University Press, Baltimore, MD, 1983.
36. Miller, K. S., and Leskiw, D. M., "Nonlinear Estimation with Radar Observations," *IEEE Transactions on Aerospace and Electronic Sys-tems*, Vol. AES-18, No. 2, pp. 192-200, March 1982.
37. Mehra, R. K., "A Comparison of Several Nonlinear Filters for Reentry Vehicle Tracking," *IEEE Transactions on Automatic Control*, Vol. AC-16, No. 4, pp. 307-319, August 1971.
38. Liang, D. F., "Exact and Approximate Nonlinear Estimation Techniques," in *Advances in the Techniques and Technology in the Application of Nonlinear Filters and Kalman Filters*, edited by C. T. Leondes, NATO Advisory Group for Aero-space Research and Development, AGARDograph No. 256, Noordhoff International Publishing, Lieden, pp. 2-1 to 2-21, 1981.

39. Widnall, W. S., "Enlarging the Region of Convergence of Kalman Filter Employing Range Measurements," *AIAA Journal*, Vol. 11, No. 3, pp. 283-287, March 1973.
40. Souris, G. M., Chen, G., Wang, J., "Tracking an Incoming Ballistic Missile Using an Extended Interval Kalman Filter," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-33, No. 1, pp. 232-240, January 1997.
41. Chen, G., Wang, J., Shieh, L. S., "Interval Kalman Filtering," *IEEE Trans. on AES*, Vol..33, No. 1, pp. 250-259, Jan. 1997.
42. Rader, C. M., Steinhardt, A. O., "Hyperbolic Householder Transformation," *IEEE Trans. on Signal Processing*, Vol. 34, No. 6, pp. 1589-1502, Dec. 1986.
43. Danis, N. J., "Space-Based Tactical Ballistic Missile Launch Parameter Estimation," *IEEE Trans. On Aerospace and Electronic Systems*, Vol. AES-29, No. 2, April 1993.
44. Yeddanapudi, M., Bar-Shalom, Y., "Trajectory Prediction for Ballistic Missiles Based on Boost Phase LOS Measurements," *Proceedings of SPIE: Signal and Data Processing of Small Targets 1997*, O. E. Drummond (Ed.), Vol. 3163, pp. 316-328, San Diego, CA, 29-31 July 1997.
45. Gura, I. A., "Extension of Linear Estimation Techniques to Nonlinear Problems," *The Journal of Astronautical Sciences*, Vol. XV, No. 4, pp. 194-205, July/August 1968.
46. Maybeck, P. S., *Stochastic Models, Estimation, and Control*, Vol. 1, Academic Press, NY, 1979.
47. Brown, A., and Bowles, W. M., "Measurement Updating Using the U-D Factorization Method When the Measurement Noise is Correlated," *Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON)*, Dayton, OH, pp. 344-348, 17-19 May 1983.
48. Lupash, L. O., "Comments on 'Efficient Time Propagation of U-D Covariance Factors'," *IEEE Trans. on Automatic Control*, Vol. AC-28, No. 11, pp. 1061-062, June 1983.
49. Lupash, L. O., "Case of Updating the Factorized Covariance Matrix," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 1, pp. 221-222, Jan.-Feb. 1994.
50. Upadhyay, T. N., and Damonlakis, J. N., "Sequential Piecewise Recursive Filter for GPS Low-Dynamics Navigation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-16, No. 4, pp. 481-491, July 1980.
51. Carlson, N. A., "Fast Triangular Formulation of the Squareroot Filter," *AIAA Journal*, Vol. 11, No. 9, pp. 1259-1265, 1973.
52. Boncelet, C. G., Dickinson, B. W., "An Extension to the Kalman Filter," *IEEE Trans. on Automatic Control*, Vol. 32, No. 2, pp. 176-179, Feb. 1987.
53. Oshman, Y., "Gain-free Information Filtering using the Spectral Decomposition," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 12, pp. 681-690, 1989.
54. Park, P., Kailath, T., "New Square-Root Algorithms for Kalman Filtering," *IEEE Trans. on Automatic Control*, Vol. 40, No. 5, pp. 895-900, May 1995.
55. Campbell, L. A., *TRACE Trajectory Analysis and Orbit Determination Program*, Vol. XIII: Square Root Information Filtering and Smoothing, USAF Space Division Report No. SSD-TR-91-07, AD-A234957, 15 March 1991.
56. Itzkowitz, H. R., Baheti, R. S., "Demonstration of Square Root Kalman Filter on WARP Parallel Computer," *Proceedings of American Control Conference*, Pittsburgh, PA, June 1989.