Int. J. Communications, Network and System Sciences, **2012, 5, 609-623**
http://dx.doi.org/10.4236/ijcns.2012.529071 Published Online September 2012 (http://www.SciRP.org/journal/ijcns)

Adaptation in Stochastic Dynamic Systems—Survey and New Results III: Robust LQ Regulator Modification

Innokentiy V. Semushin
School of Mathematics and Information Technology, Ulyanovsk State University, Ulyanovsk, Russia
Email: kentvsem@gmail.com

Received July 7, 2012; revised August 3, 2012; accepted August 14, 2012

ABSTRACT

The paper is intended to provide algorithmic and computational support for solving the frequently encountered linear-quadratic regulator (LQR) problems based on receding-horizon control methodology which is most applicable for adaptive and predictive control where Riccati iterations rather than solution of Algebraic Riccati Equations are needed. By extending the most efficient computational methods of LQG estimation to the LQR problems, some new algorithms are formulated and rigorously substantiated to prevent Riccati iterations divergence when cycled in computer implementation. Specifically developed for robust LQR implementation are the two-stage Riccati scalarized iteration algorithms belonging to one of three classes: 1) Potter style (square-root); 2) Bierman style (LDL^T); and 3) Kailath style (array) algorithms. They are based on scalarization, factorization and orthogonalization techniques, which allow more reliable LQR computations. Algorithmic templates offer customization flexibility, together with the utmost brevity, to both users and application programmers, and to ensure the independence of a specific computer language.

Keywords: Adaptive Control; Factorization; Least Squares; Linear Systems; LQG Estimator; LQ Regulator; Orthogonalization; Receding Horizon Control; Scalarization

1. Introduction

A thorough insight into the history of Automatic Control Systems theory gained by reading volumes such as the Systems and Control Encyclopedia [1] convinces us that ACS theory as a model-based science has passed through the three epochs of its development (**Figure 1**). The contemporary epoch III is the epoch of uncertainty system optimization. It has grown into two mutually complementary branches: adaptability and robustness. The latter percepts the uncertainty as a nuisance factor not to be identified but only compensated in a rough manner that leads to Fault Tolerant Control. On the contrary, the first branch brings three problems to be solved: 1) quickest Change Point Detection or more generally, Model Classification; 2) reliable Model Identification; and 3) adequate System Modification. In Gibson's view [2], these three functions are the determinant attributes of each adaptive system.

In accordance with this view, adaptability is realized as interoperability of the three units called *Modifier*/*Identifier*/*Classifier*, *MIC* for short [3,4]. In the corresponding **Figure 2**, Classifier detects Data Source parameter change points in order to give the well-timed Start for Identifier. Identifier seeks to estimate the Data Source parameters θ whose new unknown values $\theta^{\dagger}$ may result from the change, each change is considered as a system fault. While Identifier is implementing this feature, Classifier seeks to detect the point when the parameter estimates $\hat{\theta}$ have approached their reasonable (near-to-optimum) value $\hat{\theta}^{\star}$ in order to give the well-timed Stop for Identifier. At this point, Modifier begins to perform Compensator modification formally viewed as assigning the final value $\hat{\theta}^{\star}$ to the Compensator's parameter $\overline{\theta}$.

Different solutions for Classifier have been proposed, from the very simple [5] to sophisticated ones based on changepoint detection methods recently surveyed in [6] and developed in [7].

An abundance of solutions available for Identifier can be conventionally aggregated into the five functionally distinguishable categories [3]:

(C1) Bayesian Adaptive Model approach [8,9] has led to the modern multiple model adaptive estimation (MMAE) method based on a bank of parallel Kalman filters (KF), each KF designed to correspond a particular fault status of the system [10].

(C2) Extended (Augmented) Adaptive Model approach uses the Kalman Filtering to predict the state variables of the system and also to estimate its constant parameters. In so doing, the state vector is augmented by

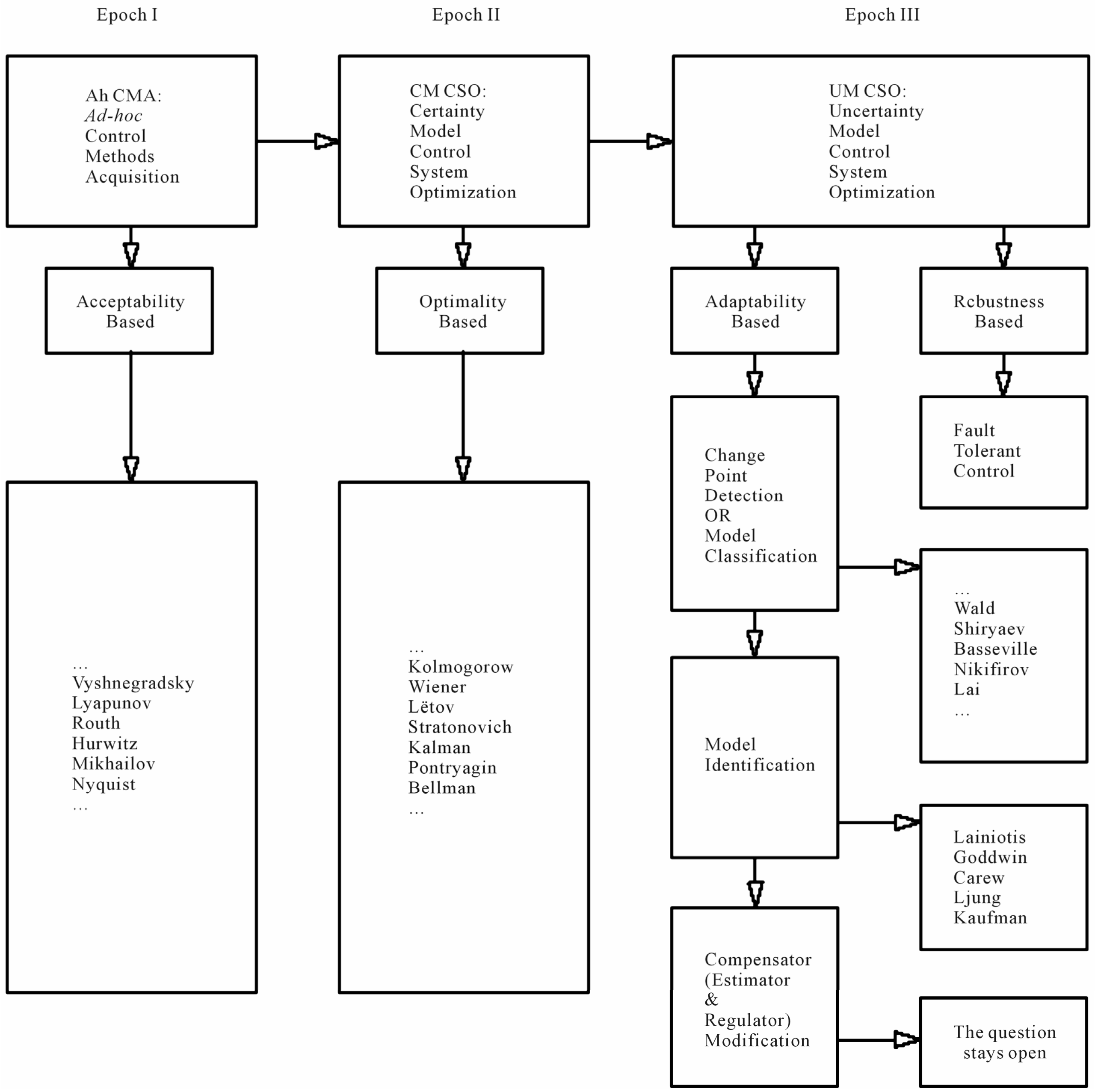

Figure 1. Three epochs of automatic control systems history.

the unknown parameters [11]. The augmented dynamic equations become nonlinear. If linearized, they lead to the Extended Kalman Filter (EKF) [12,13]. The EKF implements a Kalman filter for a system dynamics that results from the linearization of the original non-linear augmented filter dynamics around the preceding state estimates.

(C3) Analytical Relations based Adaptive Model approach uses the analytical relations between the optimal system equations and the Data Source (DS) statistics. They are used with the current estimates of unknown DS statistics (parameters) substituted for the exact (unknown) values. Usually, this requires a convergent numerical method to be developed [14]. The most distinctive feature of this approach is that it provides no feedback on the system performance index (the adaptation loop is open).

(C4) Performance Index based Adaptive Model approach provides a feedback on the system performance index. Two key features determine this category: 1) a performance index (PI) must be available in order to be used as a tool for system optimization in practice, not only in theory, and 2) numerical optimization methods must be applicable in order to select from them the best

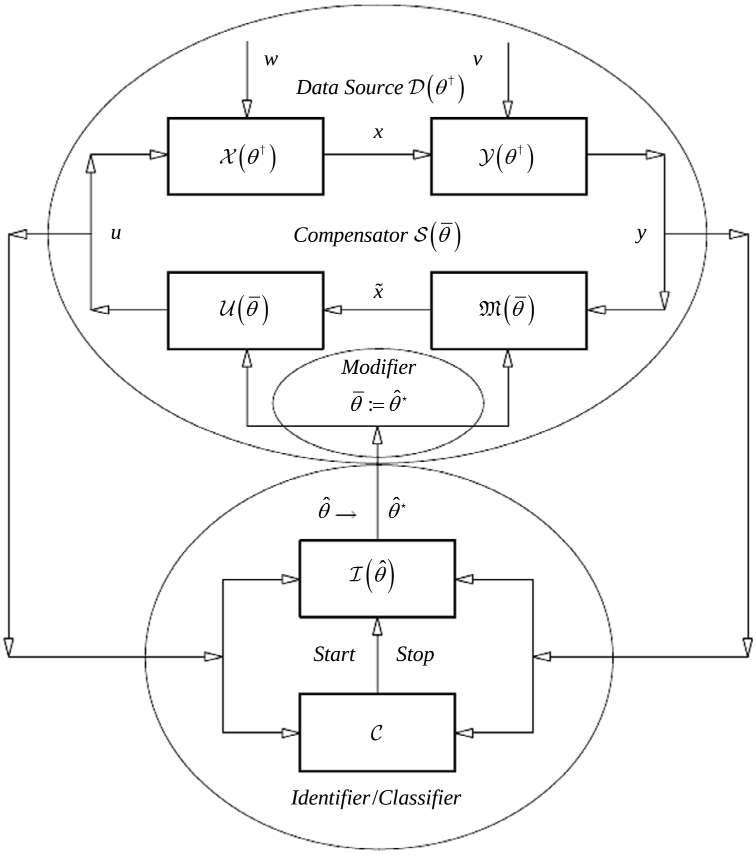
w
v
Data Source $\mathcal{D}(\theta^{\dagger})$
$\mathcal{X}(\theta^{\dagger})$
x
$\mathcal{Y}(\theta^{\dagger})$
u
Compensator $\mathcal{S}(\bar{\theta})$
y
$\mathcal{U}(\bar{\theta})$
$\tilde{x}$
$\mathfrak{M}(\bar{\theta})$
Modifier
$\bar{\theta} := \hat{\theta}^{*}$
$\hat{\theta} \longrightarrow$
$\hat{\theta}^{*}$
$\mathcal{I}(\hat{\theta})$
Start
Stop
$\mathcal{C}$
Identifier/Classifier

Figure 2. Adaptive stochastic control system structure: $\mathcal{X}$ stands for Plant; $\mathcal{Y}$ for Sensor; $\mathcal{U}$ for Regulator; $\mathfrak{M}$ for Estimator.

one as the system parameters estimator able to minimize the adopted PI. They are the focus of particularly intense scrutiny, and among them most theoretically substantiated and practically developed is Ljungian Minimum Prediction Error (MPE) method [15,16]. A less known method belonging to the same category is the Auxiliary Performance Index (API) method which renders possible least squares fitting the adaptive model state to the pure albeit hidden DS state, not to the measured (incomplete and noisy) output [3,4].

(C5) Characteristic Matching Adaptive Model approach is usually based on introducing into model equations a fictitious noise with its root-mean-square (RMS) adjustable [17,18] for a better fitting of the model. This approach is adjacent to robust system control [19].

The above generalized categories are not necessarily pure, *i.e.* they can be met in combinations as it is the case of (C2) + (C5) in [20] or (C1) + (C2) in [21]. More to it, the same five categories hold for Classifier solutions, e.g. [22-24].

Given Classifier running as a "hot-line alert system", a next successive adaptation period can emerge spontaneously as a model identification phase followed by a compensator (Estimator & Regulator) modification phase (cf. **Figures 1** and **2**). For the basic **Figure 2**, it is claimed that in automatic control, a regulator is a device which has the function of maintaining a designated characteristic of the plant behaviour by transforming the estimates of plant internal states into the control inputs applied (through an actuator) to the plant. A state estimator is typically a computer-implemented mathematical model that models a real system composed of the plant and a sensor in order to provide an estimate of plant's internal state, given measurements of the input and output of the real system, which is called *Data Source* in **Figure 2**.

The present paper in intended to provide algorithmic and computational perspective for answering questions on how to perform Regulator Modification phase in adaptive control systems (the lower right side block in **Figure 1**). It is absolutely understandable that this phase should be based on the regulator design methods, so that the very regulator modification can be treated as a re-design procedure. In this respect, we restrict ourselves to the case of discrete LQG control problem, that is, the control problem with *Linear* discrete-time plant and sensor models, and *Quadratic* performance index, and *Gaussian* random disturbances.

Fundamental to the discrete LQG control problem are two matrix nonlinear Riccati Difference Equations (RDE), which are dual to each other.

The *forward* RDE is used in the synthesis of LQG-estimator (LQGE, that is, Kalman Filter, KF), and the *backward* RDE for the design of Linear-Quadratic Regulator (LQR). KF and LQR are designed independently of each other (the *separation principle* [25]). The latter is cascaded with the first, thus closing the feedback loop (the *compesator* in **Figure 2**). LQR for the LQG-control law is identical to its counterpart for the LQ deterministic control law (the *certainty-equivalence principle*, [26], p. 228).

Textbooks on LQG-control contain these well-known theoretical facts. However, not every one covers the computational aspects of RDE. Often, classic textbooks confine themselves to giving references about **`care`** or **`dare`** functions of MATLAB® [27,28] thus meaning algebraic Riccati equations (ARE) [29] which can be either continuous-time ARE (**`care`**), or discrete-time ARE (**`dare`**). Solution of ARE is the critical task for stabilizing the compensator design. For discrete-time systems, the solution to DARE coincides with the steady-state solution of the RDE approached as the control horizon tends to infinity [30]. Generalized Riccati theory is the key tool to robust control [31].

Many textbooks on Automatic Control Systems contain sometimes not only the presentation or derivation of RDE, but also a summary of numerical solutions of ARE as well. For example, [32] says (Section 11.5) that citation of published works on solutions and features of ARE (as of 1986) may amount to a book of its own. It mentions an iterative method and considers, in some more detail, the iterative solution and eigenvalue methods as well.

However, the main source of information on Riccati equations is the vast research literature [33]. Great attention paid to ARE arises from the fact that the direct RDE iterations, even if they are taken in the robustified form (Joseph style), do not exhibit fast convergence to the steady-state positive-definite solution.

The study and development of computational methods for ARE have evolved vigorously for many years. Of the great many publications, we mention only a few: [34-37] for the LQ-regulator solution and [38-44] for the LQG-estimator design. Based on these methods, solvers for ARE have been implemented in such software packages as *Maple* [45], *Mathematica* [46], MATLAB [47-49] and in computer libraries as BLAS (level I-III), EISPACK and LINPACK, as well as in their successor LAPACK [50,51]. Many Riccati solvers are written in FORTRAN, and also in Python [52]. The number of publications on ARE solvers has continued to grow [53-55].

Efficient use of existing methods within the software packages and libraries is mostly meant for off line applications owing to their high comptutational cost. For instance, at each Kleinmann iteration step [56], the computationally expensive Lyapunov equation has to be solved ([26], pp. 34-121). Newton's method [57] requires solving a Lyapunov equation in the main step ([54], p. 6). Schur method [34], the most popular one amongst the eigenvalue methods, also needs considerable computation efforts and additional details to make this approach work satisfactorily.

All these methods are intended for solving ARE, and so their ultimate end is to find a stabilizable regulator solution [58]. However, sometimes there is no need for solving ARE. Such a category of problems includes the Model Predictive Control, or MPC [26,59] exploiting the idea of *finite* receding horizon control (RHC) [26]. In finite RHC, the attainment of the steady-state Riccati solution is not the case due to the very sense of words "finite horizon" and "system adaptation" as can be seen from the generalized adaptive stochastic control system structure (in **Figure 2**, reproduced from [4]). This explains why we do not consider the above surveyed methods of solving ARE advisable for regulator modification (re-design) in the adaptive control structure of **Figure 2**.

In this paper, we consider the duality relations between the two RDE, that are at the heart of LQGE on the one part, and LQR on the other part, to secure further advancement in the algorithms for Linear-Quadratic Regulator Optimization [51]. In doing so, we expect the computational methods, which have been derived for the LQG-Estimator implementation over the preceding decades and recently surveyed in [60], to be successfully extended to the LQR re-design where a stepwise solution for the (backward) RDE, rather than ARE, is of primary importance.

In Section 2 we formulate Problem 1 of determining the LQG control law for the system composed of the plant and sensor both linearly modeled and subjected to additive Gaussian white noises [26,61].

Section 3 describes Problem 2 of LQG receding horizon control in line with [26].

Solutions to the above two problems are presented in Section 4 in order to compare both forward and backward RDEs and then to move to a single Riccati iteration (aiming at the backward RDE) which is given in Section 5 in an intermediate abstract notation.

In Section 6, we split the single Riccati iteration into two consequtive stages in order to construct two separate computational procedures called "**`Riciup`** ≡ Riccati Instant Update" and "**`Rictup`** ≡ Riccati Temporal Update," which we use as the starting point for their numerical robustification.

Section 7 presents the scalarization of **`Riciup`**. By this equivalent transformation, we have prepared both stages for the three robust modification styles: Potter style in Section 8, Bierman style in Section 9 and Kailath style in Section 10.

Section 11 provides a brief look at the typical applications of the results discussed in this paper, together with a characterization of related challenges.

The paper closes with the concluding remarks about the novelty of the new algorithmic insights.

2. LQG Control Problem

The overall system model includes: an *n*-dimentional stochastic discrete-time plant state equation

$$\begin{aligned} &x(t_{i+1}) = \Phi(t_{i+1}, t_i)\,x(t_i) + B(t_i)\,u(t_i) + w(t_i),\\ &i = 0, 1, \cdots; \quad x(t_0) \sim \mathcal{N}(\bar{x}_0, P_0) \end{aligned} \tag{1}$$

and a *m*-dimentional measurement (sensor) equation

$$y(t_i) = H(t_i)\,x(t_i) + v(t_i), \; i = 1, 2, \cdots, \tag{2}$$

where $\{w(t_0), w(t_1), \cdots\}$ and $\{v(t_1), v(t_2), \cdots\}$ are two mutually independent noise sequences of independent Gaussian (normal) zero mean random vectors representing the state disturbance *w* and the measurement error *v*, characterized by covariance matrices $Q(t_i) \geq 0$ (positive semi-definite) and $R(t_i) > 0$ (positive definite) correspondingly and independent of Gaussian initial state $x(t_0)$ of mean $\bar{x}_0$ and covariance matrix P_0. Control input $u_{[0,N]} \equiv \{u(t_k)\}_{k=0}^{N}$ is assumed to be sought as a sequence $\{u(t_0), u(t_1), \cdots, u(t_N)\}$ of *r*-dimentional control vectors $u(t_k)$ which are applied to (1) to minimize the mean square performance index, PI (the expected cost) on a finite horizon of *N* time steps:

$$\begin{aligned} &J\left(t_0, x(t_0), u_{[0,N]}\right)\\ &\equiv E\left\{\sum_{k=0}^{N}\left(\left\|x(t_k)\right\|^2_{\Psi(t_k)} + \left\|u(t_k)\right\|^2_{\Sigma(t_k)}\right) + \left\|x(t_{N+1})\right\|^2_{\Psi_f}\right\}. \end{aligned} \tag{3}$$

where the equivalence symbol reads “is equal by definition to”. It is assumed that any nonzero input $u(t_k)$ within the prediction horizon incurres a cost. This amounts to assuming that each weighting matrix $\Sigma(t_k)$, which is symmetric, is positive definite (PD): $\Sigma(t_k) = \Sigma(t_k)^T > 0$. Further, it is assumed that the instantaneous cost at time t_k (the term within parentheses in (3)) is nonnegative. Together with the above PD condition, this is equivalent to the semi-positive definiteness of each symmetric matrix $\Psi(t_k) = \Psi(t_k)^T$: $\Psi(t_k) \geq 0$. The terminal (or final) cost $\left\|x(t_{N+1})\right\|^2_{\Psi_f}$ is also assumed nonnegative, hence $\Psi_f = \Psi_f^T \geq 0$ ($f \equiv final$).

Remark 1. Expectation operator $E\{\cdot\}$ in (3) is defined w.r.t. probability measures induced by $x(t_0) \sim \mathcal{N}(\bar{x}_0, P_0)$ and $w(t_i) \sim \mathcal{N}(0, Q(t_i))$, $v(t_i) \sim \mathcal{N}(0, R(t_i))$.

Remark 2. Metric in (3) is chosen to be elliptic:

$$\left\|x(t_k)\right\|^2_{\Psi(t_k)} \equiv x(t_k)^T\,\Psi(t_k)\,x(t_k)$$

and so on. By means of it, one can regulate the importance of any summand in criterion (3). For example, the more costly is a single (the *j*-th) control input $u_j(t_k)$ within the prediction control horizon (PCH, $0 \leq k \leq N$), the greater should be its weight defined as the *j*-th diagonal element of matrix $\Sigma(t_k)$ in comparison with others.

Selecting $\Psi(t_k)$, $\Sigma(t_k)$, Ψ_f as identity matrices brings us back to classical spherical distance measures. In signal processing, one needs sometimes to emphasize specific directions/dimentional components where statistical facts are more relevant. This is called ICA (Independent Component Analysis) [62].

Remark 3. The length of PCH, *N* in (3), approaching infinity ($N \to \infty$) is not a judicious choice for adaptive systems. Infinitely large *N* would mean the intention to attain a steady-state mode of control, when no unforeseen model changes are considered anymore. This is in deep contradiction with the very sense of adaptation. Thus, *N* must be finite. A question arises: what finite value of *N* can be selected? Obviously, it depends on the mechanism which the unforeseen changes are subject to. By assumption, these changes should not occur very frequently compared with the control system transition time in order let the adaptor keep up with the dynamics of changes. If the changes mechanism operates as an independent actor, it is reasonable to take *N* equal or greater than the expected time interval between the neighbouring model change points.

Remark 4. Weighting matrices $\Psi(t_k)$ and $\Sigma(t_k)$ can be time-dependent within the PCH. There are many ways to do so. A reasonable one would be a matrix ordering: both weighting matrix sequences are chosen in decreasing order (Version 1)

$$\begin{aligned} &\Psi(t_0) \geq \Psi(t_1) \geq \cdots \geq \Psi(t_N)\\ &\Sigma(t_0) \geq \Sigma(t_1) \geq \cdots \geq \Sigma(t_N) \end{aligned} \tag{4}$$

or in increasing order (Version 2)

$$\begin{aligned} &\Psi(t_0) \leq \Psi(t_1) \leq \cdots \leq \Psi(t_N)\\ &\Sigma(t_0) \leq \Sigma(t_1) \leq \cdots \leq \Sigma(t_N) \end{aligned} \tag{5}$$

where the matrix inequality is interpreted as the standard positive matrix inequality: $\Psi(t_0) \geq \Psi(t_1)$, meaning $\Psi(t_0) - \Psi(t_1) \geq 0$ and so on.

Remark 5. In adaptive systems, our knowledge of how the matrices describing models (1), (2) behave is getting more and more vague in course of time within the

PCH. Common sense seems to tell us that this fact gives us reasons for prefering the decreasing version (4). In this case, the instantaneous cost at a farther time t_k as part of penalty criterion (3), will be less than at preceding times. Working with decreasing sequences (4) can be also justified on grounds of RHC. As described in Section 3, all the control inputs except for the first one, which are found to minimize the overal RHC cost, are discarded anyway.

On the other hand, the choice of (4) can mean underestimating the risk of farther erroneous states $x(t_k)$ and wrong controls $u(t_k)$ caused just by our vague knowledge of how the matrices describing models (1), (2) will behave in future. To mitigate the risk, one should prefer the increasing version (5).

Thus, it becomes clear that the cost behaviour of control, especially of RHC, is a serious issue deserving a special study and experimenting which is planned to be made beyond the scope of this paper.

The standard LQG control problem (P1) is stated as follows.

Problem P1 Consider (1) and (2) as the *linear* models of a plant and a sensor subject to *Gaussian* excitations *w* and *v*. Define the *quadratic* performance index (3) with the symmetric matrices $\Sigma(t_k) > 0$, $\Psi(t_k) \geq 0$ and $\Psi_f \geq 0$. Find an optimal *physically feasible* control input $u^\star = u^\star_{[0,N]}$ to the plant (1), initialized from the event $(t_0, x(t_0))$, minimizing the performance index (3).

Remark 6. The notion "physically feasible" applied to control inputs here and below infers cause-and-effect relationships between control inputs $\{u(t_i)\}$ and system outputs $\{y(t_i)\}$: the first can not appear before the latter.

3. Receding Horizon (LQG) Control

RHC was introduced by French engineer Richalet and his colleagues in 1978 [59] to relax the computational difficulties of steady-state control [26]. In the LQG framework, the RHC problem (P2) looks as follows.

Problem P2 Consider (1) and (2) as the linear models of a plant and a sensor subject to Gaussian noise inputs *w* and *v*.

RHC Procedure: At time t_i, define the *quadratic* PI

$$J\left(t_i, x(t_i), u_{[i,i+N]}\right) \equiv E\left\{\sum_{k=i}^{i+N}\left(\left\|x(t_k)\right\|^2_{\Psi(t_k)} + \left\|u(t_k)\right\|^2_{\Sigma(t_k)}\right) + \left\|x(t_{i+N+1})\right\|^2_{\Psi_f}\right\} \quad (6)$$

for symmetric matrices $\Sigma(t_k) > 0$, $\Psi(t_k) \geq 0$ and $\Psi_f \geq 0$. At t_i, find an optimal physically feasible sequence $u^\star_{[i,i+N]} \equiv \left\{u^\star(t_i), u^\star(t_{i+1}), \cdots, u^\star(t_{i+N})\right\}$ starting with the event $(t_i, x(t_i))$, minimizing the performance index (6). Apply to the plant (1) the initial control vector $u^\star(t_i)$ of the optimal sequence whose subsequent *N* vectors are discarded. Repeat the procedure at time t_{i+1} to select $u^\star(t_{i+1})$ from the next optimal sequence $u^\star_{[i+1,i+1+N]} \equiv \left\{u^\star(t_{i+1}), u^\star(t_{i+2}), \cdots, u^\star(t_{i+1+N})\right\}$.

Although so defined RH LQG control can be hardly considered "optimal" in a rigor sense, it has attractive features [26] thus prompting suggestions that the RHC be used as the basis in the adaptive control structure (of **Figure 2**) for robust regulator computations as done in the sections that follow.

4. Riccati-Based Solution

From the comparison of **P1** and **P2** statements, in order to obtain criterion (6) from criterion (3) one should advance the (zero-indexed) event $(t_0, x(t_0))$, which is initial for the whole control sequence in **P1**, *i* steps: $t_0 \to t_{i+0} = t_i$, and so $u^\star_{[0,N]} \to u^\star_{[i+0,i+N]} = u^\star_{[i,i+N]}$. Therefore all the subsequent results concerning regulation problems will be formulated for **P1**. They can be shifted in parallel *i* steps ahead to obtain the corresponding correct result for **P2**.

Theorem 1 [26]. Optimal LQG-control law is decomposed into two independent series-connected parts 4.1 and 4.2:

4.1. Optimal (Kalman) Filter, KF Equations

1) For $i = 0, 1, \cdots, N$ the KF computes the extrapolated estimates $\hat{x}(t^-_{i+1})$ for $x(t_{i+1})$. They are obtained through the temporal update, from t_i to t_{i+1}, of the filtered estimates $\hat{x}(t^+_i)$ as

$$\hat{x}(t^-_{i+1}) = \Phi(t_{i+1}, t_i)\hat{x}(t^+_i) + B(t_i)u^\star(t_i) \quad (7)$$

with $\hat{x}(t^+_0) := \bar{x}_0 \equiv E\{x(t_0)\}$ and the *covariance matrices*

$$P(t^-_{i+1}) = Q(t_i) + \Phi(t_{i+1}, t_i)P(t^+_i)\Phi^T(t_{i+1}, t_i)$$

where

$$P(t^+_0) := P_0 \equiv E\left\{\left[x(t_0) - \bar{x}_0\right]\left[x(t_0) - \bar{x}_0\right]^T\right\}.$$

2) For $i = 1, 2, \cdots, N$ the KF computes the so-called filtered (that is measurement updated) estimates $\hat{x}(t^+_i)$. They are obtained through the measurement update using $z_i \equiv z(t_i)$ with covariances $R(t_i) > 0$ at t_i, as

$$\hat{x}(t^+_i) = \hat{x}(t^-_i) + K_f(t_i)\left[z_i - H(t_i)\hat{x}(t^-_i)\right] \quad (8)$$

with the filter gain ($_f \equiv$ filter)

$$K_f(t_i) = P(t^-_i)H^T(t_i)\left[R(t_i) + H(t_i)P(t_i)H^T(t_i)\right]^{-1}$$

and also the *filtered estimates covariance matrices*

$$P\left(t_i^+\right) = P\left(t_i^-\right) - K_f\left(t_i\right)H\left(t_i\right)P\left(t_i^-\right).$$

4.2. Optimal Linear-Quadratic Regulator, LQR Equations

Minimum expected cost for completing the control process on the regulation horizon is provided by the following *LQR* ($_r \equiv$ regulator):

$$u^\star\left(t_i\right) = -G_r\left(t_i\right)\hat{x}\left(t_i^+\right), \quad i = 0,1,\cdots,N. \qquad (9)$$

Control function of stochastic *LQR*

$$u^\star\left[t_i,(\cdot)\right] = -G_r\left(t_i\right)(\cdot) \qquad (10)$$

is identical to the control function of deterministic *LQR*, and for matrix $G_r\left(t_i\right)$ in (10) the following algorithm holds:

$$\Pi\left(t_{N+1}\right) = \Psi_f, \qquad (11a)$$

$$\bar{A}\left(t_i\right) = \Sigma\left(t_i\right) + B^T\left(t_i\right)\Pi\left(t_{i+1}\right)B\left(t_i\right), \qquad (11b)$$

$$\begin{aligned}\Lambda\left(t_i\right) = {} & \Phi^T\left(t_{i+1},t_i\right)\Pi\left(t_{i+1}\right)B\left(t_i\right) \\ & \times \bar{A}^{-1}\left(t_i\right)B^T\left(t_i\right)\Pi\left(t_{i+1}\right)\Phi\left(t_{i+1},t_i\right),\end{aligned} \qquad (11c)$$

$$M\left(t_i\right) = \Phi^T\left(t_{i+1},t_i\right)\Pi\left(t_{i+1}\right)\Phi\left(t_{i+1},t_i\right) - \Lambda\left(t_i\right), \qquad (11d)$$

$$K_r\left(t_i\right) = \bar{A}^{-1}\left(t_i\right)B^T\left(t_i\right)\Pi\left(t_{i+1}\right), \qquad (11e)$$

$$G_r\left(t_i\right) = K_r\left(t_i\right)\Phi\left(t_{i+1},t_i\right), \qquad (11f)$$

$$\Pi\left(t_i\right) = \Psi\left(t_i\right) + M\left(t_i\right). \qquad (11g)$$

Remark 7. In (11), items (11b) to (11g) cycle for $i = N, N-1,\cdots,1,0$, although $\Pi\left(t_0\right)$ found at $i = 0$ by (11g) is out of use (end of computations).

Remark 8. Matrix $\Pi\left(t_i\right)$ in (11) satisfies the *backward* RDE

$$\begin{aligned}\Pi\left(t_i\right) = {} & \Psi\left(t_i\right) + \Phi^T\left(t_{i+1},t_i\right)\left\{\Pi\left(t_{i+1}\right)\right. \\ & -\Pi\left(t_{i+1}\right)B\left(t_i\right)\left[\Sigma\left(t_i\right) + B^T\left(t_i\right)\Pi\left(t_{i+1}\right)\right. \\ & \left.\left.\times B\left(t_i\right)\right]^{-1}B^T\left(t_i\right)\Pi\left(t_{i+1}\right)\right\}\Phi\left(t_{t+1},t_i\right), \\ & i = N, N-1,\cdots,1,0\end{aligned} \qquad (12)$$

with the final condition $\Pi\left(t_{N+1}\right) = \Psi_f$ at $i = N$ when the inverse-time computations start. Equation (12) is dual to the following forward RDE for matrix $P\left(t_j^-\right)$:

$$\begin{aligned}P\left(t_{j+1}^-\right) = {} & Q\left(t_j\right) + \Phi\left(t_{j+1},t_j\right)\left\{P\left(t_j^-\right)\right. \\ & -P\left(t_j^-\right)H^T\left(t_j\right)\left[R\left(t_j\right) + H\left(t_j\right)P\left(t_j^-\right)\right. \\ & \left.\left.\times H^T\left(t_j\right)\right]^{-1}H\left(t_j\right)P\left(t_j^-\right)\right\}\Phi^T\left(t_{j+1},t_j\right), \\ & j = 0,1,\cdots,N\end{aligned} \qquad (13)$$

with the initial condition $\left\{\cdot_{13}\right\} = P_0$ at $j = 0$ where $\left\{\cdot_{13}\right\} = P_0$ stands for the term within braces in (13).

5. Singly Taken Formal Riccati Iteration

Consider a single Riccati iteration (RI) as a formal procedure in abstract matrix notations including an arbitrary matrix *G* of compatible size ($\dim G = n \times s$):

$$\tilde{X}_- = V + A^T\left[\tilde{X} - \tilde{X}G\left(C + G^T\tilde{X}G\right)^{-1}G^T\tilde{X}\right]A, \qquad (14)$$

where $C \in \mathbb{R}^{s\times s}$, $C = C^T > 0$, $\tilde{X} > 0$ and $V \geq 0$.

As seen from Equations (12) and (13), iterations (14) are repeated for both KF and LQR with the assignment operation $\tilde{X} := \tilde{X}_-$ between the iterative repetitions. From here on, we omit the case of KF which is well-known and widely presented in literature [60] and direct our attention towards the LQR.

For the case of LQR, let us introduce the following correspondences between the formal and actual specifications:

$$\begin{aligned}&\tilde{X}_- \equiv \Pi\left(t_i\right),\ V \equiv \Psi\left(t_i\right),\ A \equiv \Phi\left(t_{i+1},t_i\right), \\ &\tilde{X} \equiv \Pi\left(t_{i+1}\right),\ G \equiv B\left(t_i\right),\ C \equiv \Sigma\left(t_i\right), \\ &K_r \equiv K_r\left(t_i\right),\ G_r \equiv G_r\left(t_i\right),\ V_f \equiv \Psi_f,\ s \equiv r.\end{aligned} \qquad (15)$$

Substituting (15) into (14) yields (12), *i.e.* backward RDE for $\Pi\left(t_i\right)$ in algorithm (11) with the terminal condition $\Pi\left(t_{N+1}\right) = \Psi_f$ taken at $i = N$. Considering formal symbols K_r and G_r in (15) leads to the equivalent form

$$K_r = \left(C + G^T\tilde{X}G\right)^{-1}G^T\tilde{X}, \qquad (16)$$

$$\tilde{X}_- = V + A^T\left[\tilde{X} - \tilde{X}GK_r\right]A, \qquad (17)$$

$$G_r = K_r A \qquad (18)$$

of procedure (14). Let us represent the computations (16), (17) and (18) as a procedure denoted by

$$\texttt{Ric}\left(C, G, \tilde{X}, V, A, s \middle| \tilde{X}_-, G_r\right)$$

with the assignment $\tilde{X} := V_f$ at $i = N$ and by cycling the procedure `Ric` for *i* = *N* down to 0 in such a way as to take input parameters in accordance with (15), we get the output parameters in accordance with (15).

Remark 9. In reality, the last statement is true only theoretically, that is in the absence of computer roundoff errors. Formula (17) constitutes a real danger for matrix $\tilde{X}_-$ to have lost its property of positive definiteness at the differencing in brackets. This, in particular, is the prime cause of `Ric`'s numerical instability able to diverge Riccati iterations when cycled in computer implementation.

6. Two-Stage Riccati Iterations

Just as the discrete KF naturally operates in two stages 1) and 2) (stated in Theorem 1, Subsection 4.1), a single Riccati iteration **`Ric`** (16), (17), (18) can be used in two consecutive stages shown schematically in **Figure 3**

$$\hat{X} = \tilde{X} - \tilde{X}GK_r, \qquad (19)$$

$$\tilde{X}_- = V + A^T\hat{X}A. \qquad (20)$$

We name them correspondingly as follows:

Stage I: $\boxed{\texttt{Riciup}\left(C, G, \tilde{X}, s \,\middle|\, \hat{X}, K_r\right)}$

Name: Riccati instant update: (16) ⇒ (19), and

Stage II: $\boxed{\texttt{Rictup}\left(V, A, \hat{X}, K_r, s \,\middle|\, \tilde{X}_-, G_r\right)}$

Name: Riccati temporal update: (20) ⇒ (18).

Remark 10. Notation such as (16) ⇒ (19) hereafter reads: "(16) computed and then (19) computed".

Lemma 1. For all positive definite matrices $\tilde{X}$ and C algorithm (16) ⇒ (19) of Stage I is equivalent to the following algorithm (21)

$$\hat{Z} = \tilde{Z} + GC^{-1}G^T \qquad (21)$$

in the sense that $\hat{Z}^{-1} = \hat{X}$ with any matrix $G \in \mathbb{R}^{n\times s}$ when $\tilde{Z}^{-1} = \tilde{X}$.

Proof. It can be found in [40], pp. 26-27.

7. Riccati Scalarized Instant Update

When $C > 0$ is a non-diagonal matrix, the square-root free Cholesky decomposition $C = L_C D_C L_C^T$ with the unit lower triangular matrix L_C and diagonal matrix $D_C = diag\ [d_1, d_2, \cdots, d_s]$, $\forall d_k > 0$, proves to be useful. With that purpose denote $Y \equiv GL_C^{-T}$, where Y^T is a solution to the lower triangular matrix equation $L_C Y^T = G^T$. Then instead of algorithm (16) ⇒ (19) for Stage I we obtain its equivalent

$$\hat{X} = \tilde{X} - \tilde{X}Y\left(D_C + Y^T\tilde{X}Y\right)^{-1}Y^T\tilde{X}. \qquad (22)$$

Considering matrix $Y = \left[y_1 \middle| y_2 \middle| \cdots \middle| y_s\right]$ columnwise permits one to use the following **Algorithms 1** and **2**.

Algorithm 1 (scalarized, direct).

1) Initialization: $X_0 := \tilde{X}$.

2) Scalarized (columnwise) input:
for $k = 1$ to s cycle

$$\begin{aligned} \bar{\alpha}_k &= d_k + y_k^T X_{k-1} y_k, \\ X_k &= X_{k-1} - X_{k-1} y_k \bar{\alpha}_k^{-1} y_k^T X_{k-1}. \end{aligned} \qquad (23)$$

3) Concluding assignment: $\hat{X} := X_s$.

Algorithm 2 (scalarized, inverse).

1) Initialization: $Z_0 := \tilde{Z} \equiv \tilde{X}^{-1}$.

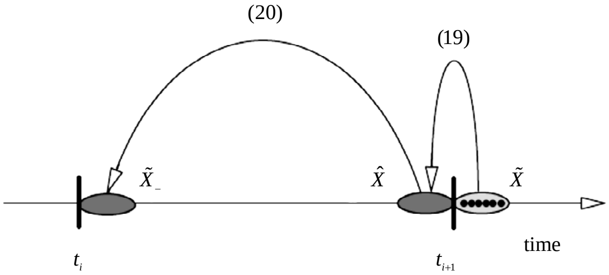

Figure 3. Backward Riccati recursion. Initialization *i* := *N*; $\tilde{X}$:= V_f; while (*i* ≥ 0) do {(19); (20); *i* := *i* − 1}.

2) Scalarized (columnwise) input:
for $k = 1$ to s cycle

$$Z_k = Z_{k-1} + y_k d_k^{-1} y_k^T. \qquad (24)$$

3) Concluding assignment: $\hat{Z} := Z_s$.

Lemma 2. Algorithms 1 and 2 are equivalent to each other.

Proof. All X_k and Z_k in (23), (24) are mutually inverse of each other by virtue of Lemma 1, and so, $\hat{Z} = \hat{X}^{-1}$.

Theorem 2 (*Verification of Algorithm* 1 *for Equation* (22)). Algorithm 1 is true, *i.e.*, it can be used instead of (22).

Proof. By reason of Lemma 1, equality (22) is equivalent to equality (21) taking into account transcriptions $G \to Y$, $C \to D$. Thereby (21) is as follows:

$$\begin{aligned} \hat{Z} &= \tilde{Z} + \left[y_1 \middle| \cdots \middle| y_s\right] \begin{bmatrix} d_1^{-1} & & 0 \\ & \ddots & \\ 0 & & d_s^{-1} \end{bmatrix} \begin{bmatrix} y_1^T \\ \vdots \\ y_s^T \end{bmatrix} \\ &= \tilde{Z} + \sum_{k=1}^{s} y_k d_k^{-1} y_k^T. \end{aligned}$$

Algorithm 2 gives the same value $\hat{Z}$. It is equivalent to algorithm 1 (by virtue of Lemma 2). Hence, algorithm 1 results in matrix $\hat{X}$, which is produced by procedure (16) ⇒ (19) (**`Riciup`**) and also by (22).

Let us introduce the scalarized procedure **`Ricsiup`**:

Stage I: $\boxed{\texttt{Ricsiup}\left(C, G, \tilde{X}, s \,\middle|\, \hat{X}, K_r\right)}$

Name: Riccati scalarized instant update
(instead of (16) ⇒ (19))

```
begin
    $C = L_C D_C L_C^T$        ⊙ obtain $L_C$, $D_C$
    $L_C Y^T = G^T$            ⊙ obtain $Y$
    $k := 1$                   ⊙ begin
    while $k \le s$ do         ⊙ cycle
    begin                      ⊙ continue
        $p^T := y_k^T \tilde{X}$   ⊙ a row
        $\bar{\alpha} := d_k + p^T y_k$                      ⊙ a scalar
        $\bar{K}_k := p^T / \bar{\alpha}$                    ⊙ a row
        $\hat{X} := \tilde{X} - \tilde{X} y_k \bar{K}_k$     ⊙ matrix $\hat{X}$
        $\tilde{X} := \hat{X}$                               ⊙ update $\tilde{X}$
        $k := k+1$                                           ⊙ increment
    end                                                      ⊙ finish
    $\bar{K}^T := \left[ \bar{K}_1^T \middle| \cdots \middle| \bar{K}_s^T \right]$   ⊙ collect $\bar{K}^T$
    $L_C^T K_r = \bar{K}$                                    ⊙ obtain $K_r$
end
```

Theorem 3. Algorithm **`Ricsiup`** is equivalent to algorithm **`Riciup`**.

Proof. In cycle **`while`**, there is implemented the proved algorithm 1, which forms $\hat{X}$. To complete the proof, it is sufficient to substitute $C = L_C D_C L_C^T$ into (16) and verify that $K_r = L_C^{-T} \bar{K}$ where the intermediate matrix $\bar{K} = \left(D_C + Y^T \tilde{X} Y\right)^{-1} Y^T \tilde{X}$ has been introduced.

Remark 11. In the transition from **`Riciup`** to **`Ricsiup`**, there is eliminated the operation of matrix inversion in formula (16). However, the origin of numerical instability (that is computation of $\hat{X}$ in cycle **`while`** which is a scalar (k-th) step of (19)) calls, as before, for further algebraically equivalent modifications.

8. Potter Style Modification

Applying Cholesky decomposition (for definiteness, the lower triangular one, as described, for instance, in [60]) to the symmetric matrices $\hat{X}$, $\tilde{X}$, $\tilde{X}_-$, V and V_f, we change to operations with their square roots (denoted by generic symbol S):

$$\begin{aligned}&\hat{X} \equiv \hat{S}\hat{S}^T,\ \tilde{X} \equiv \tilde{S}\tilde{S}^T,\ \tilde{X}_- \equiv \tilde{S}_-\tilde{S}_-^T,\\ &V \equiv S_V S_V^T,\ V_f \equiv S_f S_f^T.\end{aligned} \qquad (25)$$

Modified procedure **`srRicsiup`** operates with the square roots of (25) like it was first introduced in [63]:

Stage I: $\boxed{\mathtt{srRicsiup}\left(C, G, \tilde{S}, s \middle| \hat{S}, K_r\right)}$

Name: square-root Riccati scalarized instant update (instead of (16) $\Rightarrow$ (19)):

```
begin
    $C = L_C D_C L_C^T$                                      ⊙ obtain $L_C$, $D_C$
    $L_C Y^T = G^T$                                          ⊙ obtain $Y$
    $k := 1$                                                 ⊙ begin
    while $k \le s$ do                                       ⊙ cycle
    begin                                                    ⊙ continue
        $f := \tilde{S}^T y_k$                               ⊙ a column
        $\bar{\alpha} := d_k + f^T f$                        ⊙ a scalar
        $\gamma := \left(1 + \sqrt{d_k / \bar{\alpha}}\right)^{-1}$   ⊙ a scalar
        $\bar{K}_k := f^T \tilde{S}^T / \bar{\alpha}$        ⊙ a row
        $\hat{S} := \tilde{S} - \gamma \bar{K}_k^T f^T$      ⊙ matrix $\hat{S}$
        $\tilde{S} := \hat{S}$                               ⊙ update $\tilde{S}$
        $k := k+1$                                           ⊙ increment
    end                                                      ⊙ finish
    $\bar{K}^T := \left[ \bar{K}_1^T \middle| \cdots \middle| \bar{K}_s^T \right]$   ⊙ collect $\bar{K}^T$
    $L_C^T K_r = \bar{K}$                                    ⊙ obtain $K_r$
end
```

Correspondingly, we change from procedure **`Rictup`** to **`srRictup`** using orthogonal transformations:

Stage II: $\boxed{\mathtt{srRictup}\left(S_V, A, \hat{S}, K_r, s \middle| \tilde{S}_-, G_r\right)}$

Name: square-root Riccati temporal update orthogonalized (instead of (20) $\Rightarrow$ (18)):

$$\begin{bmatrix} \tilde{S}_-^T \\ 0 \end{bmatrix} = \mathcal{O} \begin{bmatrix} \hat{S}^T A \\ S_V^T \end{bmatrix}, \quad G_r = -K_r A \qquad (26)$$

where $\mathcal{O}$ is one of the orthogonal transformations (Hausholder or Givens or Gram-Schmidt) reducing matrix in the right-hand side of (26) to the upper triangular form.

Theorem 4. Algorithm **`srRicsiup`** is equivalent to algorithm **`Ricsiup`** and algorithm **`srRictup`** is equivalent to algorithm **`Rictup`**.

Proof. Selecting from (25) the proper substitution for matrix $\tilde{X}$ in algorithm **`Ricsiup`** and then factoring difference $\tilde{X} - \tilde{X} y_k \bar{K}_k$ (denoted $\hat{X}$ for each k in **`Ricsiup`**) into $\hat{S}\hat{S}^T$ with $\hat{S} \equiv \tilde{S}\left(I - \beta f f^T\right)$ yields

$$\forall f : I - f f^T / \bar{\alpha} = \left(I - \beta f f^T\right)^2.$$

This results in the quadratic equation with respect to β

$$\beta^2 - 2\left(\bar{\alpha} - d_k\right)^{-1} \beta + \bar{\alpha}^{-1}\left(\bar{\alpha} - d_k\right)^{-1} = 0.$$

From its two solutions one selects

$$\beta = \bar{\alpha}^{-1}\left(1 + \sqrt{d_k / \bar{\alpha}}\right)^{-1}$$

as being numerically stable, and introduces the intermediate notation $\gamma \equiv \bar{\alpha}\beta$. The first equation in (26) can be proved by premultiplying it by itself transposed. The product coincides with (20).

Remark 12. Some comparative insight into numerical stability of the above algorithm as well as of the two algorithmic modifications that follow in Sections 9 and 10, can be gained from [64], pp. 163-167 and pp. 198-201.

9. Bierman Style Modification

The algorithm to be presented here is conceptually the Bierman's algorithm originally developed for the *U-D* matrix decomposition used for the KF covariance factorizations. It was motivated by the work of Agee and Turner about the one-rank modification of the UDU^T Cholesky factorization [40] which we convert into the LDL^T formulation as follows.

Theorem 5 (*Agee-Turner PD Factorization Update*). Let

$$\bar{P} = \bar{L}\bar{D}\bar{L}^T = L\tilde{D}L^T + caa^T$$

where L is unit lower triangular, $\tilde{D} = diag\left[\tilde{d}_1, \cdots, \tilde{d}_n\right]$, c is a scalar, $a = \left[a_1, a_2, \cdots, a_n\right]^T$, and $n = \dim P$.

If $\bar{P}$ is PD (positive definite), then factors $\bar{L}$ (unit lower triangular) and $\bar{D} > 0$ (diagonal) can be calculated in the following algorithm:

1) Initialization: $c_1 = c$.

2) Computation: for $i = 1$ to $n-1$ cycle:

a) $\bar{d}_i = \tilde{d}_i + c_i a_i^2$;

b) for $k = i+1$ to n cycle:

i) $\underline{a_k} := a_k - a_i l_{ki}$;

ii) $\bar{l}_{ki} = l_{ki} + c_i a_i a_k / \bar{d}_i$; ⊙ *In matrices L, nontrivial entries exist only below their unit diagonal.*

c) $c_{i+1} = c_i \tilde{d}_i / \bar{d}_i$.

3) Concluding assignment: $\bar{d}_n = \tilde{d}_n + c_n a_n^2$.

Proof. The algorithm is validated by representing $x^T \bar{P} x$ as a sum of complete squares with substitution of the equation $\bar{P} = P + caa^T$, $P = L\tilde{D}L^T$ in this quadratic form. Details can be found in [40] or [60].

We apply Bierman's algorithm to LQR design in the context of Remark 11, thereby presenting another modification of procedure **`Ricsiup`** named here **`ldRisciup`** that avoids potentially unstable numerical differencing. What is required for that is conversion of the aforesaid *UD* Bierman's algorithm into its *LD* analogue and writing it in terms of LQR. In doing this, we obtain the following result.

Theorem 6 (*Bierman style* **`ldRisciup`** *algorithm*). Let $\tilde{X} := \tilde{X} - \tilde{X} y_k \bar{K}_k$ written as $\hat{X} := \tilde{X} - \tilde{X} y_k \bar{K}_k$ followed by $\tilde{X} := \hat{X}$ in procedure **`Ricsiup`** be using factorizations $\hat{X} = \hat{L}\hat{D}\hat{L}^T$ and $\tilde{X} = \tilde{L}\tilde{D}\tilde{L}^T$ where any and all L are unit lower triangular and any and all D positive diagonal. Then **`Ricsiup`** is equivalent to the following procedure.

```
Stage I: $\boxed{\texttt{ldRicsiup}\left(C, G, \tilde{L}, \tilde{D}, s \,\middle|\, \hat{L}, \hat{D}, K_r\right)}$
Name: L-D Riccati scalarized instant update
      (instead of (16) ⇒ (19)):
begin
    $C = L_C D_C L_C^T$                         ⊙ obtain $L_C$, $D_C$
    $L_C Y^T = G^T$                             ⊙ obtain $Y$
    $k := 1$                                    ⊙ begin
    while $k \le s$ do                          ⊙ cycle
    begin                                       ⊙ continue
        $f = [f_1, \cdots, f_n]^T := \tilde{L}^T y_k$
        $v = [v_1, \cdots, v_n]^T := \tilde{D} f$
        $\alpha' := d_k$
        $\bar{K}_k := [0|\cdots|0|v_n]$
    for $i = n$ down to 1 do
      begin
        $\alpha := \alpha' + v_i f_i$
        $\gamma := 1/\alpha$
        $\hat{d}_i := \alpha' \gamma \tilde{d}_i$
        $\lambda := -\gamma f_i$
        for $j = i+1$ to $n$ do
          begin
            $\hat{l}_{ji} := \tilde{l}_{ji} + \lambda \bar{K}_{j,k}^T$
            $\bar{K}_{j,k}^T := \bar{K}_{j,k}^T + \tilde{l}_{ji} v_i$
          end
          $\alpha' := \alpha$
      end
    $\tilde{L} := \hat{L}$; $\tilde{D} := \hat{D}$             ⊙ update $\tilde{L}$ and $\tilde{D}$
    $k := k+1$                                  ⊙ increment
    end                                         ⊙ finish
    $\bar{K}^T := [\bar{K}_1^T|\cdots|\bar{K}_s^T]$             ⊙ collect $\bar{K}^T$
    $L_C^T K_r = \bar{K}$                         ⊙ obtain $K_r$
end
```

Remark 13. Recall that y_k is the *k*-th column of matrix *Y* and d_k is the *k*-th diagonal element of matrix *D*, both introduced in Section 7; $\bar{K}_{j,k}^T$ is the *j*-th element of column $\bar{K}_k^T$ that exists within each repetition of cycle **`while`**.

Proof. Given in [60] similarly to the *UD*-version of [40]. □

Forming the matrices $\{\hat{L}, \hat{D}\}$ from $\{\tilde{L}, \tilde{D}\}$ is illustrated schematically by **Figure 4**. It shows that: 1) this computation is columnwise starting from the last column and moving backwards; 2) the diagonal positions are used to store elements of *D* because the predetermined unit diagonals of both $\hat{L}$ and $\tilde{L}$ need no storing; (3) output data $\{\hat{L}, \hat{D}\}$ can supersede $\{\tilde{L}, \tilde{D}\}$ in the same array; and (4) the upper triangular part of the array is zero and so may not be stored thus saving memory.

We now turn to the *LQ* implementation of Stage II in the form of a new procedure **`ldRictup`** which is to be equivalent to **`Rictup`**.

At entry to **`ldRictup`**, we have two pairs of factors:

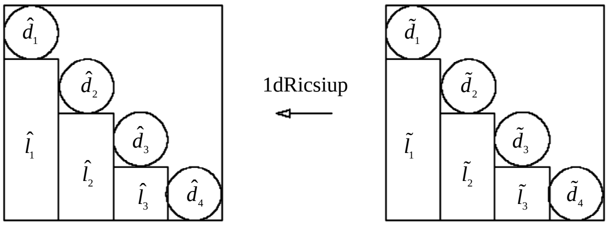

Figure 4. Forming of matrices $\{\hat{L}, \hat{D}\}$.

1) $\{\hat L, \hat D\}$ instead of $\hat X = \hat L \hat D \hat L^T$, and

2) $\{L_V, D_V\}$ instead of $V = L_V D_V L_V^T$ in (20).

Using them, re-write (20) as

$$\tilde X_- = \underbrace{\left[A^T \hat L \middle| L_V\right]}_{W} \underbrace{\begin{bmatrix}\hat D & 0\\ 0 & D_V\end{bmatrix}}_{D} \underbrace{\begin{bmatrix}\hat L^T A\\ L_V^T\end{bmatrix}}_{W^T} \qquad (27)$$

The problem of Stage II sounds as follows: Given are factors W and D for which (27) holds, find factors $\bar L$ and $\bar D$, $\bar L$ unit lower triangular and $\bar D$ positive diagonal, such that for matrix $\tilde X_-$ to be represented in the factored form $\tilde X_- = \tilde L_- \tilde D_- \tilde L_-^T$ the following factorization holds: $\tilde X_- = \bar L \bar D \bar L^T$. In other words, we seek to have an algorithm yielding the pair $\{\bar L, \bar D\}$ so as to immediately get results: $\tilde L_- \equiv \bar L$ and $\tilde D_- \equiv \bar D$. So, in equation $WDW^T = \bar L \bar D \bar L^T$, the left hand side is given and the right hand side is what we wish to find. This is exactly what is known as Weighted Gram-Schmidt Orthogonalization (WG-SO). It is presented in [40] (pp. 125-126) in the *UD*-version. For our needs, we convert it into the *LD*-version as follows.

Lemma 3. Let $\{w_1, \cdots, w_n\}$ be a linear independent set of (column) *M*-vectors, $M \ge n$, and let $D_1, \cdots, D_M$ be positive scalars in a diagonal matrix $D = diag\ [D_1, \cdots, D_M]$. If $\{v_1, \cdots, v_n\}$ are defined by the following algorithm, then none of the *v*'s are zero and $v_k^T D v_j = 0$ for $k \ne j$:

MG-SO: $\boxed{\mathtt{1dMG\text{-}SOrt}\left(W, D \middle| \bar L, \bar D\right)}$

Name: *L-D* Modified Weighted Gram-Schmidt Orthogonalization:

```
begin
  for $k = 1$ to $n$ do
      $v_k := w_k$
  for $k = 1$ to $n$ do
    begin
        $\bar D_k := v_k^T D v_k$
        for $j = k+1$ to $n$ do
          begin
              $\bar L_{jk} := \left(v_j^T D v_k\right) / \bar D_k$
              $v_j := v_j - \bar L_{jk} v_k$
          end
    end
end
```

Proof. Can be obtained by a straightforward calculation.

Remark 14. The above procedure is called modified because it works columnwise (**Figure 5**).

Finally for the case of $V \equiv \Psi(t_i) > 0$, we obtain

Stage II: $\boxed{\mathtt{1dRictup}\left(L_V, D_V, A, \hat L, \hat D, K_r, s \middle| \tilde L_-, \tilde D_-, G_r\right)}$

Name: *L-D* Riccati temporal update orthogonalized (instead of (20) ⇒ (18)):

Compute $W^T \equiv \left[w_1 | \ldots | w_n\right] = \begin{bmatrix}\hat L^T A\\ L_V^T\end{bmatrix}$

(with $M = 2n$).

Compute $D \equiv diag\ [D_1, \cdots, D_M] = \begin{bmatrix}\hat D & 0\\ 0 & D_V\end{bmatrix}$.

Call **ldMG-SOrt** $\left(W, D \middle| \tilde L_-, \tilde D_-\right)$.

Compute $G_r = -K_r A$.

10. Kailath Style Modification

There exists another a comparatively new class of algorithms in Kalman filtering (LQG estimation) area [65], the so-called *array algorithms*. They alleviate some computational problems associated with Riccati iterations by using the well-known *QR*-decomposition in numerical linear algebra with an appropriate orthogonal matrix *Q* where *R* is upper triangular (*R* indicates here the right corner of a matrix). Below, we show how to adapt such algorithms for LQR implementations, and we refer to them as *Kailath style* paying thus a tribute to works by Kailath and co-authors [43]. Starting out again from Remark 11, we choose now (from several alternatives recently serveyed in [60]) a *square-root array* modification.

Theorem 7 (*Kailath style* **asrRisciup** *algorithm*). Let $\tilde X := \tilde X - \tilde X y_k \bar K_k$ written as $\hat X := \tilde X - \tilde X y_k \bar K_k$ followed by $\tilde X := \hat X$ in procedure **Ricsiup** be using factorizations like in (25), that is $\hat X = \hat S \hat S^T$ and $\tilde X = \tilde S \tilde S^T$

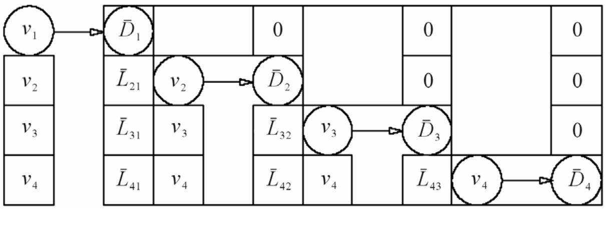

Figure 5. Modified WG-SO procedure, *LD*-version.

with both S lower triangular. Then **Ricsiup** is equivalent to the following procedure.

```
Stage I: $\boxed{\texttt{asrRicsiup}\left(C,G,\tilde{S},s\,\middle|\,\hat{S},K_r\right)}$
Name: array square-root Riccati scalarized instant
      update (instead of (16) $\Rightarrow$ (19)):
begin
    $C = L_C D_C L_C^T$                 ⊙ obtain $L_C$ , $D_C$
    $L_C Y^T = G^T$                     ⊙ obtain $Y$
    $k := 1$                            ⊙ begin
    while $k \le s$ do                  ⊙ cycle
      begin                             ⊙ continue
        $\sigma := \sqrt{d_k}$
        $f = [\, f_1, \cdots, f_n \,]^T := \tilde{S}^T y_k$
        $\left[\begin{array}{c|c} \delta & \bar{K}_k\delta \\ \hline 0 & \hat{S}^T \end{array}\right] := Q_k \left[\begin{array}{c|c} \sigma & 0 \\ \hline f & \tilde{S}^T \end{array}\right]$     (T)
        $\tilde{S} := \hat{S}$          ⊙ update $\tilde{S}$
        $k := k+1$                      ⊙ increment
      end                               ⊙ finish
    $\bar{K}^T := \left[ \bar{K}_1^T \middle| \cdots \middle| \bar{K}_s^T \right]$     ⊙ collect $\bar{K}^T$
    $L_C^T K_r = \bar{K}$               ⊙ obtain $K_r$
end
```

Proof. Assignment operator (T) in the above algorithm contains two arrays: pre-array B (on the right) and post-array A (on the left). At each cycle, let the latter be obtained from the first in the upper triangular form by means of an appropriate orthogomal transformation Q_k (it may be Hausholder reflections or Givens rotations): $A = Q_k B$. Since $Q_k^T Q_k = I$, we have $B^T B = A^T A$ and via straightforward calculations, we are done.

Stage II in this modification coincides with Stage II in the Potter style modification, that is expression (26).

11. Applications Challenges

Possible applications of adaptation capability of stochastic systems are numerous and can be found in almost every field of modern engineering. While considering applicability of the above results, one should select the cases that seem to fit perfectly in the pattern of **Figure 2**. In this pattern, the very necessity for adaptation is considered as a factual constraint the occurrence of which in time is comparatively rare resulting from an abrupt fault against the long lasting nominal mode of system operation. This can be exemplified by the development and implementation of a high integrity navigation system based on the combined use of an inertial measurement unit aided by different outer sources of data [20,66] some of which are fault-susceptible or working in an accident-prone situation.

Famous industrial/technological study cases to be brought forward as applications to theoretical/computational work are those on advanced MPC collected in [59].

Overall, we have to admit that practical problems are much more challenging than theoretical ones. One barrier to overcome is the nonlinearity of the original system (Data Source) models. The traditional remedy for this is to invoke a linearized *perturbation model* or equation of the first variation about a nominal (or reference) solution to the nonlinear model on the assumption that such a solution is known (as in [67]) or delivered by an External (more precise) Data Source. In the strict sense, equation (1) has been written yet in the form of perturbation model, as it can be seen from criterion (3). The latter case, combining the use of the Global Positioning System (GPS acting as an external data source) and an inertial measurement unit for vehicle applications, can be viewed as a modeling technique for online estimation of the error between the reference model and the real dynamics [68].

Another challenge to be considered is a set of constraints representing the physical limitations of the process variables as is the case in MPC and optimization for papermaking machines [69].

12. Concluding Remarks

The emphasis in this paper has been on the robust linear quadratic regulator computations where the single Riccati iteration algorithm is an integral part and where seeking a steady-state Riccati solution (Algebraic Riccari Equation) does not apply.

Main novelty of the results is technical: we have shown that linear algebra methods of input scalarization, matrix factorization and array orthogonalization earlier known for the robustified linear quadratic estimators due to [40, 63,65] and many other works, now are successfully extended to the robust LQR computation problems including LQ Regulator Modification phase in the adaptive control systems. The new algorithmic LQ regulator formulations based on these methods enhance LQR numeric robustness and generate a productive perspective for further investigations into the regulator modification (redesign) methods within the structure of adaptive control.

Further research is encouraged into the advancement of new insights about the numerics of LQR/ARE/DARE procedures, thus leading to Adaptive Control System CAD that is expected to include all three ACS phases—*Modifier/Identifier/Classifier*.

13. Acknowledgements

The author would like to express his gratitude to an anonymous reviewer for a number of suggestions that helped to improve the style of this paper.

REFERENCES

[1] M. G. Singh, "Systems and Control Encyclopedia: Theory, Technology, Applications," Pergamon Press, Inc., Elmsford, 1986.

[2] J. E. Gibson, "Nonlinear Automatic Control: Chapter 11," McGrow Hill, New York, 1962.

[3] I. V. Semushin, "Adaptation in Stochastic Dynamic Systems—Survey and New Results I," *International Journal of Communications, Network and System Sciences*, Vol. 4, No. 1, 2011, pp. 17-23. doi:10.4236/ijcns.2011.41002

[4] I. V. Semushin, "Adaptation in Stochastic Dynamic Systems—Survey and New Results II," *International Journal of Communications, Network and System Sciences*, Vol. 4, No. 4, 2011, pp. 266-285. doi:10.4236/ijcns.2011.44032

[5] I. V. Semushin and S. A. Ponyrko, "On the Choice of Start-Stop Algorithm while Minimizing the Square Mean Performance Index," *Autometria, Siberian Division of the USSR Academy of Sciences*, No. 2, 1973, pp. 68-74.

[6] T. L. Lai, "Sequential Changepoint Detection in Quality Control and Dynamical Systems," *Journal of the Royal Statistical Society*, *Series B* (*Methodological*), Vol. 57, No. 4, 1995, pp. 613-658.

[7] T. L. Lai and H. P. Xing, "Sequential Change-Point Detection When the Pre- and Post-Change Parameters Are Unknown," Technical Report No. 2009-5, Department of Statistics, Stanford University, Stanford, 2009.

[8] D. G. Lainiotis, "Partitioning: A Unified Framework for Adaptive Systems, I. Estimation," *Proceedings of the IEEE*, Vol. 64, No. 8, 1976, pp. 1126-1143. doi:10.1109/PROC.1976.10284

[9] D. G. Lainiotis, "Partitioning: A Unified Framework for Adaptive Systems, II. Control," *Proceedings of the IEEE*, Vol. 64, No. 8, 1976, pp. 1182-1197. doi:10.1109/PROC.1976.10289

[10] M. Athans and C.-B. Chang, "Adaptive Estimation and Parameter Identification Using Multiple Model Estimation Algorithm," Technical Note, Lincoln Lab, MIT, Lexington, 1976.

[11] G. C. Goodwin and K. S. Sin, "Adaptive Filtering Prediction and Control," Prentice Hall, Englewood Cliffs, 1984.

[12] H. W. Sorenson, "Kalman Filtering: Theory and Application," IEEE Press, New York, 1985.

[13] J. K. Uhlmann, "Algorithms for Multiple Target Tracking," *American Scientist*, Vol. 80, No. 2, 1992, pp. 128-141.

[14] B. Carew and P. Belanger, "Identification of Optimaum Filter Steady-State Gain for Systems with Unknown Noise Covariances," *IEEE Transactions on Automatic Control*, Vol. 18, No. 6, 1973, pp. 582-587. doi:10.1109/TAC.1973.1100420

[15] L. Ljung, "System Identification—Theory for the User," Prentice Hall, Englewood Cliffs, 1987.

[16] P. E. Caines, "Linear Stochastic Systems," John Wiley & Sons, Inc., New York, 1987.

[17] H. Kaufman and D. Beaulier, "Adaptive Parameter Identification," *IEEE Transactions on Automatic Control*, Vol. 17, No. 5, 1972, pp. 729-731. doi:10.1109/TAC.1972.1100111

[18] T. Yoshimura and T. Soeda, "A Technique for Compensating the Filter Performance by Fictitious Noise," *ASME Journal of Dynanmic Systems, Measurement, and Control*, Vol. 100, 1978.

[19] H. S. Zhang, D. D. Zhang, W. Wang and L. H. Xie, "Robust Filtering by Fictitious Noises," *Proceedings of the* 42*nd IEEE Conference on Decision and Control*, Maui, 9-12 December 2003, pp. 1280-1284.

[20] I. V. Semoushin, "Identifying Parameters of Linear Stochastic Differential Equations from Incomplete Noisy Measurements," In: Y.-C. Hon, M. Yamamoto, J. Cheng and J.-Y. Lee, Eds., *Recent Developments in Theories & Numerics*, World Scientific, 2003, pp. 281-290.

[21] C. Barrios, H. Himberg, Y. Motai and A. Sadek, "Multiple Model Framework of Adaptive Extended Kalman Filtering for Predicting Vehicle Location," *Proceedings of the* 2006 *IEEE Intelligent Transportation Systems Conference*, Toronto, 17-20 September 2006, pp. 1053-1059.

[22] D. Rupp, G. Ducard, E. Shafai and H. P. Geering, "Extended Multiple Model Adaptive Estimation for the Detection of Sensor and Actuator Faults," *Proceedings of the* 44*th IEEE Conference on Decision and Control, and the European Control Conference* 2005, Seville, 12-15 December 2005, pp. 3079-3084.

[23] I. Semoushin, J. Tsyganova and M. Kulikova, "Fault Point Detection with the Bank of Competitive Kalman Filters," *Lecture Notes in Computer Science*, Vol. 2658, Part 2, 2003, pp. 417-426.

[24] P. Eide and P. Maybeck, "An MMAE Failure Detection System for the F-16," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 32, No. 3, 1996, pp. 1125-1136. doi:10.1109/7.532271

[25] H. S. Witsenhausen, "Separation of Estimation and Control for Discrete Time Systems," *Proceedings of the IEEE*, Vol. 59, No. 11, 1971, pp. 1557-1566. doi:10.1109/PROC.1971.8488

[26] E. Mosca, "Optimal, Predictive and Adaptive Control," Prentice Hall, Englewood Cliffs, 1994.

[27] S. E. Dushin, N. S. Zotov, D. Kh. Imaev, N. N. Kuzmin and V. B. Yakovlev, "Theory of Automatic Control," 3rd Edition, Vysshaya Shkola, Moscow, 2009.

[28] A. G. Aleksandrov, "Optimal and Adaptive Systems," Vysshaya Shkola, Moscow, 1989.

[29] P. Lancaster and L. Rodman, "Algebraic Riccati Equations," Oxford University Press, Inc., New York, 1995.

[30] J. C. Willems, "Least Squares Stationary Optimal Control and the Algebraic Riccati Equation," *IEEE Transactions on Automatic Control*, Vol. 16, No. 6, 1971, pp. 621-634. doi:10.1109/TAC.1971.1099831

[31] V. Ionescu, C. Oara, M. D. Weiss and M. Weiss, "Generalized Riccati Theory and Robust Control. A Popov Func-

tion Approach," John Wiley & Sons, Inc., New York, 1999.

[32] B. C. Kuo, "Digital Control Systems," Holt, Rinehart and Winston, Inc., New York, 1980.

[33] "Riccati Solvers at ScienceDirect: 485 Articles Found," 2012. http://www.sciencedirect.com/

[34] A. J. Laub, "A Schur Method for Solving Algebraic Riccati Equations," *IEEE Transactions on Automatic Control*, Vol. 24, No. 6, 1979, pp. 913-921. doi:10.1109/TAC.1979.1102178

[35] T. Pappas, A. J. Laub and N. R. Sandell Jr., "On the Numerical Solution of the Discrete-Time Algebraic Riccati Equation," *IEEE Transactions on Automatic Control*, Vol. 25, No. 4, 1980, pp. 631-641. doi:10.1109/TAC.1980.1102434

[36] W. F. Arnold III and A. J. Laub, "Generalized Eigen-Problem Algorithms and Software for Algebraic Riccati Equations," *Proceedings of the IEEE*, Vol. 72, No. 12, 1984, pp. 1746-1754. doi:10.1109/PROC.1984.13083

[37] F. A. Aliyev, B. A. Bordyug and B. V. Larin, "Orthogonal Projections and Solution of Algebraic Riccati Equations," *USSR Computational Mathematics and Mathematical Physics*, Vol. 29, No. 3, 1989, pp. 104-108. doi:10.1016/0041-5553(89)90154-7

[38] P. G. Kaminski, A. E. Bryson Jr. and S. F. Schmidt, "Discrete Square Root Filtering: A Survey of Current Techniques," *IEEE Transactions on Automatic Control*, Vol. 16, No. 6, 1971, pp. 727-736. doi:10.1109/TAC.1971.1099816

[39] D. G. Lainiotis, "Discrete Riccati Equation Solutions: Partitioned Algorithms," *IEEE Transactions on Automatic Control*, Vol. 20, No. 4, 1975, pp. 555-556. doi:10.1109/TAC.1975.1101010

[40] G. J. Bierman, "Factorization Methods for Discrete Sequential Estimation," Academic Press, New York, San Francisco, London, 1977.

[41] D. G. Lainiotis, N. D. Assimakis and S. K. Katsikas, "A New Computationally Effective Algorithm for Solving the Discrete Riccati Equation," *Journal of Mathematical Analysis and Applications*, Vol. 186, No. 3, 1994, pp. 868-895. doi:10.1006/jmaa.1994.1338

[42] N. D. Assimakis, D. G. Lainiotis, S. K. Katsikas and F. L. Sanida, "A Survey of Recursive Algorithms for the Solution of the Discrete Time Riccati Equation," *Nonlinear Analysis*, *Theory*, *Methods & Applications*, Vol. 30, No. 4, 1997, pp. 2409-2420.

[43] T. Kailath, A. H. Sayed and B. Hassibi, "Linear Estimation," Prentice Hall, Englewood Cliffs, 2000.

[44] N. Assimakis, S. Roulis and D. Lainiotis, "Recursive Solutions of the Discrete Time Riccati Equation," *Neural*, *Parallel & Scientific Computations*, Vol. 11, No. 3, 2003, pp. 343-350.

[45] "New Features in Maple 15: Algebraic Riccati Equation Solvers," 2012.
http://www.maplesoft.com/products/maple/newfeatures/algebraicriccati.aspx

[46] "Wolfram Mathematica 8: Control System Professional Documentation. 10.3 Riccati Equations," 2012.
http://reference.wolfram.com/legacy/applications/control/OptimalControlSystemsDesign/10.3.html

[47] "MATLAB Toolboxes of SLICOT (Subroutine Library in Systems and Control Theory)," 2012.
http://www.slicot.org/index.php?site=home

[48] "SLICOT Basic Systems and Control Toolbox," 2012.
http://www.slicot.org/index.php?site=slbasic

[49] P. Benner and V. Sima, "Solving Algebraic Riccati Equations with SLICOT," *CD-ROM Proceedings of the* 11*th Mediterranean Conference on Control and Automation MED*'03," Rhodes, 18-20 June 2003, Invited Session IV01, Paper IV01-01.

[50] V. Sima, "Computational Experience in Solving Algebraic Riccati Equations," *Proceedings of the* 44*th IEEE conference on Decision and Control*, *and European Control Conference* 2005," Seville, 12-15 December 2005, pp. 7982-7987.

[51] V. Sima, "Algorithms for Linear-Quadratic Optimization (Vol. 200 of Series Pure and Applied Mathematics: A series of Monographs and Textbooks)," Chapman and Hall/CRC, London, 1996.

[52] V. Armstrong, "Updated Discrete Algebraic Riccati Equation Solver in Python," 2010.
http://jeff.rainbow-100.com/?p=141

[53] W. H. Kwon, Y. S. Moon and S. C. Ahn, "Bounds in Algebraic Riccati and Lyapunov Equations: A Survey and Some New Results," *International Journal of Control*, Vol. 64, No. 3, 1996, pp. 377-389. doi:10.1080/00207179608921634

[54] A. Bunse-Gerstner, "Computational Solution of the Algebraic Riccati Equation," 2012.
http://www.math.uni-bremen.de/zetem/numerik/Published/Report982.ps.gz

[55] H. Dai and Z.-Z. Bai. "On Eigenvalue Bounds and Iteration Methods for Discrete Algebraic Riccati Equations," *Journal of Computational Mathematics*, Vol. 29, No. 3, 2011, pp. 341-366.
http://www.jcm.ac.cn/EN/10.4208/jcm.1010-m3258
http://www.jcm.ac.cn/EN/Y2011/V29/I3/341

[56] D. L. Kleinman, "On an Iterative Technique for Riccati Equation Computations," *IEEE Transactions on Automatic Control*, Vol. 13, No. 1, 1968, pp. 114-115. doi:10.1109/TAC.1968.1098829

[57] S. J. Hammarling, "Newton's Method for Solving Algebraic Riccati Equation," NPL Report, DITC 12/82, 1982.

[58] G. Kreisselmeier, "Stabilization of Linear Systems by Constant Output Feedback Using the Riccati Equation," *IEEE Transactions on Automatic Control*, Vol. 20, No. 4, 1975, pp. 556-557. doi:10.1109/TAC.1975.1101013

[59] T. Zheng, "Advanced Model Predictive Control," InTech, Rijeka, 2011.

[60] I. V. Semushin, "Computational Methods of Algebra and Estimation," UlSTU, Ulyanovsk, 2011.
http://venec.ulstu.ru/lib/disk/2012/Semuwin.pdf

[61] P. S. Maybeck, "Stochastic Models, Estimation, and Control, Vol. 3," Academic Press, Inc., New York, London,

Paris, San Diego, San Francisco, São Paulo, Sydney, Tokyo, Toronto, 1982.

[62] P. Comon, “Independent Component Analysis, a New Concept?” *Signal Processing*, Vol. 36. No. 3, 1994, pp. 287-314.

[63] J. E. Potter and R. G. Stern, “Statistical Filtering of Space Navigation Measurements,” *Proceedings of* 1963 *AIAA Guidance and Control Conference*, New York, 12-14 August 1963, 19 p.

[64] I. V. Semushin, “Adaptive Systems of Filtering, Control and Fault Detection,” USU, Ulyanovsk, 2011.

[65] P. G. Park and T. Kailath, “New Square-Root Algorithms for Kalman Filtering,” *IEEE Transactions on Automatic Control*, Vol. 40, No. 5, 1995, pp. 895-899. doi:10.1109/9.384225

[66] P. S. Maybeck, “Stochastic Models, Estimation, and Control, Vol. 1,” Academic Press, Inc., New York, San Francisco, London, 1979, Chapter 6.

[67] D. Song, J. T. Qi, J. D. Han and G. J. Liu, “Predictive Control for Active Model and Its Applications on Unmanned Helicopters,” InTech, Rijeka, 2011. doi:10.5772/17716

[68] A. Fakharian, T. Gustafsson and M. Mehrfam, “Adaptive Kalman Filtering Based Navigation: An IMU/GPS Integration Approach,” *Proceedings of the* 8*th IEEE International Conference on Networking, Sensing and Control*, Delft, 11-13 April 2011, pp. 181-185.

[69] D. Chu, M. Forbes, J. Backstrom, C. Gheorghe and S. Chu, “Model Predictive Control and Optimization for Papermaking Processes,” InTech, Rijeka, 2011.