CHAPTER III

THEORY

3.1 Adaptive Control

Many dynamic systems to be controlled have constraint or slowly-varying uncertain parameters. The basic idea in adaptive control is to estimate the uncertain plant parameters (or, equivalently, the corresponding controller parameters) on-line based on the measured system signals, and use the estimated parameters in the control input computation. An adaptive control system can thus be regarded as a control system with on-line parameter estimation.

Research in adaptive control started in the early 1950's in connection with the design of autopilots for high-performance aircraft, which operate at a wide range of speeds and altitudes and thus experience large parameter variations. Adaptive control was proposed as a way of automatically adjusting the controller parameters in the face of changing aircraft dynamics. But interest in the subject soon diminished due to the lack of insights and the crash of a test flight. It is only in the last decade that a coherent theory of adaptive control has been developed, using various tools from nonlinear control theory. These theoretical advances, together with the availability of cheap computation, have lead to many practical applications, in areas such as robotic manipulation, aircraft and rocket control, chemical processes, power systems, ship steering, and bioengineering.

3.1.1 Basic Concepts in Adaptive Control

In some control tasks, such as those in robot manipulation, the systems to be controlled have parameter uncertainty at the beginning of the control operation. Unless such parameter uncertainty is gradually reduced on-line by an adaptation or estimation mechanism, it may cause inaccuracy or instability for the control systems.

In many other tasks, such as those in power systems, the system dynamics may have well known dynamics at the beginning, but experience unpredictable parameter variations as the control operation goes on. Without continuous "redesign" of the controller, the initially appropriate controller design may not be able to control the changing plant well. Generally, the basic objective of adaptive control is to maintain consistent performance of a system in the presence of uncertainty or unknown variation in plant parameters. Since such parameter uncertainty or variation occurs in many practical problems, adaptive control is useful in many industrial contexts. These include:

- **Robot manipulation:** Robots have to manipulate loads of various sizes, weights, and mass distributions. It is very restrictive to assume that the inertial parameters of the loads are well known before a robot picks them up and moves them away. If controllers with constant gains are used and the load parameters are not accurately known, robot motion can be either inaccurate or unstable. Adaptive control, on the other hand, allows robots to move loads of unknown parameters with high speed and high accuracy.

- **Ship steering:** On long courses, ships are usually put under automatic steering. However, the dynamic characteristics of a ship strongly depend on many uncertain parameters, such as water depth, ship loading, and wind and wave conditions. Adaptive control can be used to achieve good control performance under varying operating conditions, As well as to avoid energy loss due to excessive rudder motion.

- **Aircraft control:** The dynamic behavior of an aircraft depends on its altitude, speed, and configuration. The ratio of variations of parameter can lie between 10 to 50 in a given flight. As mentioned earlier, adaptive control was originally developed to achieve consistent aircraft performance over a large flight envelope.

- **Process control:** Models for metallurgical and chemical processes are usually complex and also hard to obtain. The parameters characterizing the processes vary from batch to batch. Furthermore, the working conditions are usually time-varying (e.g., reactor characteristics vary during the reactor's life, the raw materials

entering the process are never exactly the same, atmospheric and climatic conditions also tend to change). In fact, process control is one of the most important and active application areas of adaptive control.

Adaptive control has also been applied to other areas, such as power systems and biomedical engineering. Most adaptive control applications are aimed at handling inevitable parameter variation or parameter uncertainty. However, in some applications, particularly in process control, where hundreds of control loops may be present in a given system, adaptive control is also used to reduce the number of design parameters to be manually tuned, thus yielding an increase in engineering efficiency and practicality.

To gain insights about the behavior of the adaptive control systems and also to avoid mathematical difficulties, the unknown plant parameters are assumed constant in analyzing the adaptive control designs. In practice, the adaptive control systems are often used to handle time-varying unknown parameters. In order for the analysis results to be applicable to these practical cases, the time-varying plant parameters must vary considerably slower than the parameter adaptation. Fortunately, this is often satisfied in practice. Note that fast parameter variations may also indicate that the modeling is inadequate and that the dynamics causing the parameter changes should be additionally modeled.

3.1.2 Adaptive Control Strategies

An adaptive controller differs from an ordinary controller in that the controller parameters are variable, and there is a mechanism for adjusting these parameters online based on signals in the system. There are two main approaches for constructing adaptive controllers. One is the so-called model-reference adaptive control method, and the other is the so-called self-tuning method.

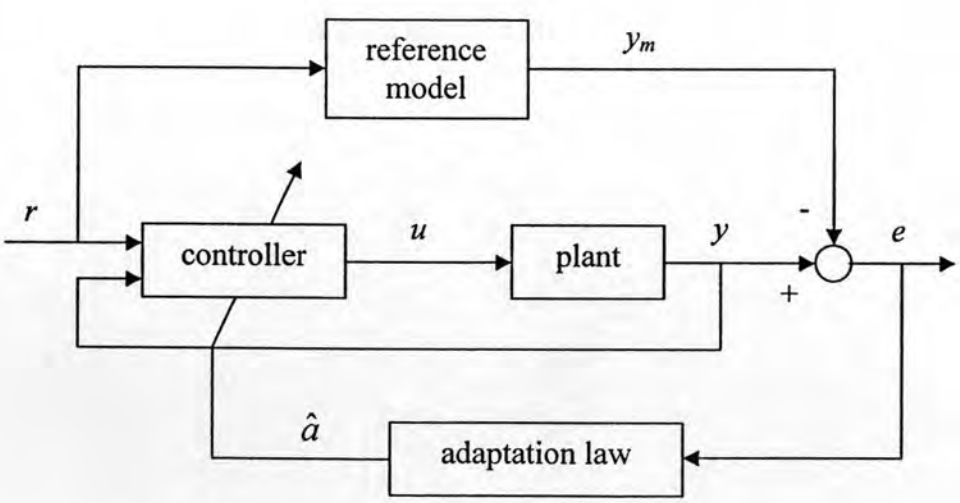
3.1.2.1 Model-reference adaptive control (MRAC)

Generally, a model-reference adaptive control system can be schematically represented by figure 3.1. It is composed of four parts: a plant containing unknown parameters, a reference model for compactly specifying the desired output of the

control system, a feedback control law containing adjustable parameters, and an adaptation mechanism for updating the adjustable parameters.

The plant is assumed to have a know structure, although the parameters are unknown. For linear plants, this means that the number of poles and the number of zeros are assumed to be known, but that the locations of these poles and zeros are not. For nonlinear plants, this implies that the structure of the dynamic equations is known, but that some parameters are not.

A reference model is used to specify the ideal response of the adaptive control system to the external command. Intuitively, it provides the ideal plant response which the adaptation mechanism should seek in adjusting the parameters. The choice of the reference model is part of the adaptive control system design. This choice has to satisfy two requirements. On the one hand, it should reflect the performance specification in the control tasks, such as rise time, settling time, overshoot or frequency domain characteristics. On the other hand, this ideal behavior should be achievable for the adaptive control system, i.e., there are some inherent constraints on the structure of the reference model (e.g., its order and relative degree) given the assumed structure of the plant model.



$\hat{a}$--- estimated parameters

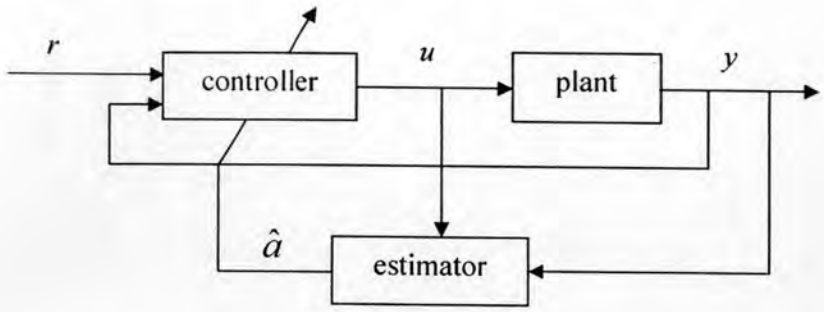
Figure 3.1 A model reference adaptive control

The controller is usually parameterized by number of adjustable parameters (implying that one may obtain a family of controllers by assigning various values to the adjustable parameters). The controller should have perfect tracking capacity in order to allow the possibility of tracking convergence. That is, when the plant parameters are exactly known, the corresponding controller parameters should make the plant output identical to that of the reference model. When the plant parameters are not known, the adaptation mechanism will adjust the controller parameters so that perfect tracking is asymptotically achieved. If the control law is linear in terms of the adjustable parameters, it is said to be linearly parameterized. Existing adaptive control designs normally require linear parameterization of the controller in order to obtain adaptation mechanisms with guaranteed stability and tracking convergence.

The adaptation mechanism is used to adjust the parameters in the control law. In MRAC systems, the adaptation law searches for parameters such that the response of the plant under adaptive control becomes the same as that of the reference model, i.e., the objective of the adaptation is make the tracking error converge to zero. Clearly, the main difference from conventional control is to synthesize an adaptation mechanism which will guarantee that the control system remains stable and the tracking error converges to zero as the parameters are varies. Many formalisms in nonlinear control can be used to this end, such as Lyapunov theory, hyperstability theory, and passivity theory. Although the application of one formalism may be more convenient than that of another, the result are often equivalent.

3.1.2.2 Self-tuning Controllers (STC)

In non-adaptive control design (e.g., pole placement), one computes the parameters of the controllers from those of the plant. If the plant parameters are not known, it is intuitively reasonable to replace them by their estimates values, as provided by a parameter estimator. A controller thus obtained by coupling a controller with an online (recursive) parameter estimator is called a self-tuning controller. Figure 3.2 illustrates the schematic structure of such an adaptive controller. Thus, a self tuning controller is a controller which performs simultaneous identification of the unknown plant.



$\hat{a}$--- estimated parameters

Figure 3.2 Self-tuning Controller

The operation of a self-tuning controller is as follows: at each time instant, the estimator sends to the controller a set of estimated plant parameters (a in figure 3.2), which is computed based on the past plant input u and output y; the computer finds the corresponding controller parameters, and then computes a control input u based on the controller parameters and measures signals; this control input u causes a new plant output to be generated, and the whole cycle of parameter and input updates is repeated. Note that the controller parameters are computed from the estimates of the plant parameters as if they were the true plant parameters. This idea is often called the certainty equivalence principle.

Parameter estimation can be understood simply as the process of finding a set of parameters that fits the available input-output data from a plant. This is different from parameter adaptation in MRAC systems, where the parameters are adjusted so that the tracking errors converge to zero. For linear plants, many techniques are available to estimate the unknown parameters of the plant. The most popular one is the least-squares method and its extensions. There are also many control techniques for linear plants, such as pole-placement, PID, LQR (linear quadratic control), minimum variance control, or H^{∞} designs. By coupling different control and estimation schemes, one can obtain a variety of self-tuning regulators. The self-tuning method can also be applied to some nonlinear system without any conceptual difference.

In the basic approach to self-tuning control, one estimates the plant parameters and then computes the controller parameters. Such a scheme is often called *indirect adaptive control*, because of the need to translate the estimates parameters into controller parameters. It is possible to eliminate this part of the computation. To do this, one notes that the control law parameters and plant parameters related to each other for a specific control method. This implies that we may reparameterize the plant model using controller parameters (which are also unknown, of course), and then use standard estimation techniques on such a model. Since no translation is needed in this scheme, it is called a *direct adaptive control* scheme. In MRAC systems, one can similarly consider direct and indirect ways of updating the controller parameters.

In self-tuning control, estimator design and controller design are separated. The estimation law (using y and u) is independent of the choice of the control law, unlike in MRAC design where the parameter adaptation law is affected by the choice of the control law (it is also interesting to note that, in self-tuning control, saturation of control input has no direct consequence on the convergence of parameter estimation). While this implies flexibility in design and simplicity in concept, the analysis of the convergence and stability of the self-tuning control system is usually more complicated.

3.1.2.3 Relations between MRAC and ST methods

As described above, MRAC control and ST control arise from different perspectives, with the parameters in MRAC systems being updated so as to minimize the tracking error between the plant output and reference model output, and the parameters in ST systems being updated so as to minimize the data-fitting error in input-output measurements. However, there are strong relations between the two design methodologies. Comparing figure 3.1 and figure 3.2, we note that the two kinds of systems both have an inner loop for control and outer loop for parameter estimation. From a theoretical point of view, it can actually be shown that MRAC and ST controllers can be put under a unified framework.

The two methods can be quite different in terms of analysis and implementation. Compared with MRAC controllers, ST controllers are more flexible

because of the possibility of coupling various controllers with various estimators (i.e., the separation of control and estimation). However, the stability and convergence of self-tuning controllers are generally quite difficult to guarantee, often requiring the signals in the system to be sufficiently rich so that the estimated parameters converge to the true parameters. If the signals are not very rich (for example, if the reference signal is zero or a constant), the estimated parameters may not be close to the true parameters, and the stability and convergence of the resulting control system may not be guaranteed. In this situation, one must either introduce perturbation signals in the input, or somehow modify the control law. In MRAC systems, however, the stability and tracking error convergence are usually guaranteed regardless of the richness of the signals.

Historically, the MRAC method was developed from optimal control of deterministic servomechanisms, while the ST method evolved from the study of stochastic regulation problems. MRAC systems have usually been considered in continuous-time form, and ST regulators in discrete time-form. In recent years, discrete-time version of MRAC controllers and continuous versions of ST controllers have also been developed.

3.1.3 The Design of Adaptive Controllers

In conventional (non-adaptive) control design, a controller structure (e.g., pole placement) is chosen first, and the parameters of the controller are then computed based on the known parameters of the plant. In adaptive control, the major difference is that the plant parameters are unknown, so that the controller parameters have to be provided by an adaptation law. As a result, the adaptive control design is more involved, with the additional needs of choosing an adaptation law and proving the stability of the system with adaptation.

The design of an adaptive controller usually involves the following three steps:

- choose a control law containing variable parameters
- choose an adaptation law for adjusting those parameters
- analyze the convergence properties of the resulting control system

When one uses the self-tuning approach for linear systems, the first two steps are quite straightforward, with inventories of control and adaptation (estimation) laws available. The difficultly lies in the analysis. When one uses MRAC design, the adaptive controller is usually found by trial and error. Sometimes, the three steps are coordinated by the use of an appropriate Lyapunov function, or using some symbolic construction tools such as the passivity formalism.

3.2 Generic Model Control (GMC)

Lee and Sullivan (1988) have generalized many of the model-based techniques into a generic structure called the generic model control, which allows the incorporation of nonlinear process models directly in the control algorithm. The diagram of generic model control is shown in figure 3.3.

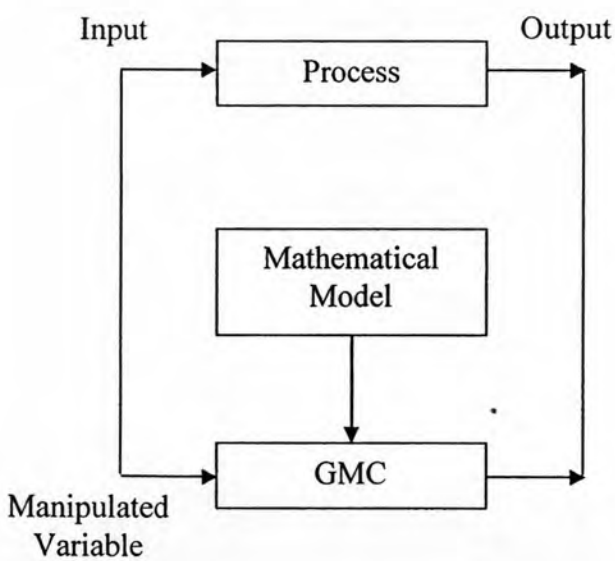


Figure 3.3 The generic model control diagram

Consider a process described by:

$$\dot{x} = f(x,u,t) \tag{3.1}$$

$$y = g(x) \tag{3.2}$$

where $\dot{x}$ is a state variable,

u is the manipulated input variable,

y is the output of the process model.

In general, f and g are nonlinear functions. From equations (3.1) and (3.2), $\dot{y}$ can be written as

$$\dot{y} = G_x f(x,u,t) \tag{3.3}$$

where

$$G_x = \frac{\partial g}{\partial x} \tag{3.4}$$

In a classical optimal control, the trajectory of y is usually compared against a nominal trajectory, $y^*(t)$, as a measure of system performance. As an alternative, consider the performance of the system to be such that:

$$(\dot{y})^*(t) = r^*(y) \tag{3.5}$$

where r^* represents some arbitrary function to be specified.

When the process is away from its desired steady state y^*, the rate of change of y, y^*, is selected to be such that the process moves towards steady state, i.e.

$$\dot{y} = K_1(t)(y^* - y) \tag{3.6}$$

where $K_1(t)$ is some diagonal matrix.

The process is selected to have zero offset, i.e.

$$\dot{y} = K_2(t)\int (y^* - y)dt \tag{3.7}$$

where $K_2(t)$ is some diagonal matrix.

$K_1(t)$ and $K_2(t)$ are constant with respect to time. Good control performance will be given by some combination of these objectives, i.e.

$$(\dot{y})^{*} = K_1(y^{*} - y) + K_2 \int (y^{*} - y)dt \tag{3.8}$$

It can be seen that by different choices of $K_1(t)$ and $K_2(t)$ the performance specification can be altered for each variable separately. One can use these values to select any "reasonable" desired response for the system. "Reasonable" implies that the parameters are chosen in relation to the system's natural dynamic response. How well the system matches this performance index is governed by how closely the chosen model matches the plant behavior.

Taking Laplace transform of the equation (3.8), transfer function of this equation becomes:

$$\frac{y}{y^{*}} = \frac{2\tau\xi s + 1}{\tau^2 s^2 + 2\tau\xi s + 1} \tag{3.9}$$

where

$$\tau = \frac{1}{\sqrt{K_2}} \qquad \text{and} \qquad \xi = \frac{K_1}{2\sqrt{K_2}}$$

This system does not yield the same response as a classical second-order system (Stephanopoulos, 1984). However, similar plots to the classical second-order response showing the normalized response of the system y/y^{*} vs. normalized time t/τ with ξ as a parameter can be produced and is shown in figure 3.4. The design procedure can be specified as follows:

1. Choose ξ from figure 3.4 to give desired shape of response,

2. Choose τ from figure 3.4 to give "appropriate" timing of response in relation to known or estimated plant speed of response,

3. Calculate K_1 and K_2 using the following equations:

$$K_1 = \frac{2\xi}{\tau} \tag{3.10}$$

$$K_2 = \frac{1}{\tau^2} \tag{3.11}$$

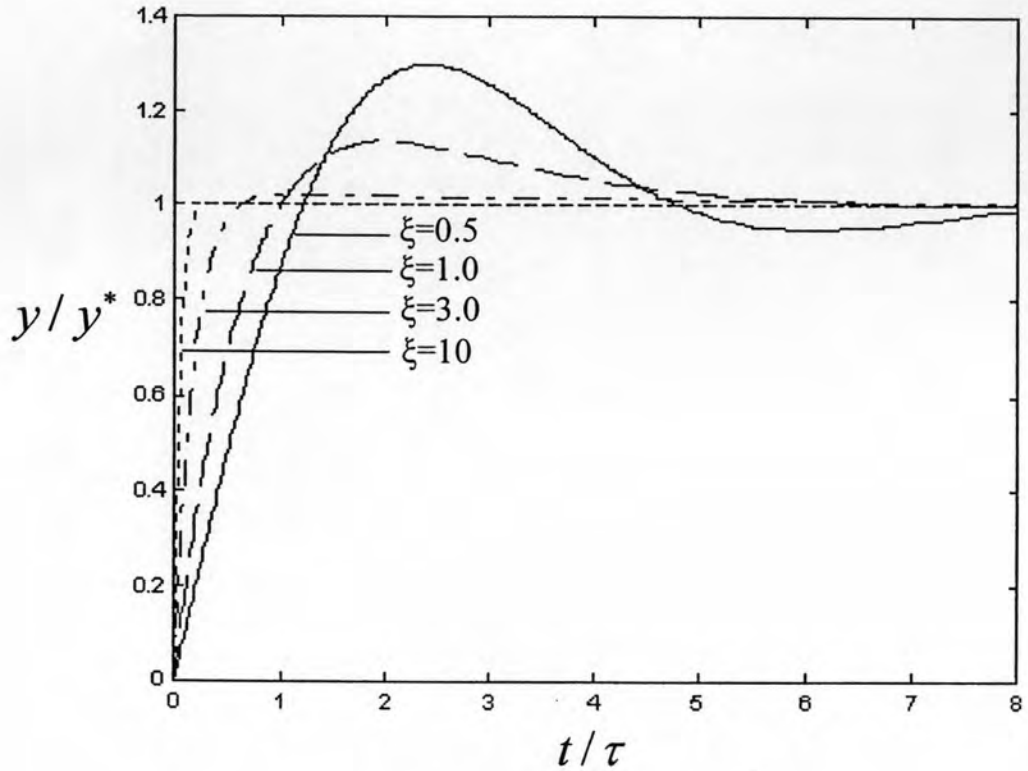


Figure 3.4 Generalized GMC profile specification

GMC has several advantages that make it a good framework for developing reactor controllers:

1. The process model appears directly in the control algorithm.

2. The process model does not need to be linearized before use, allowing for the inherent nonlinearity of exothermic batch reactor operation to be taken into account.

3. By design, GMC provides feedback control of the rate of change of the controlled variable. This suggests that the rate of temperature change, which as mentioned above is very important in heat-up operations, can be used directly as a control variable.

4. The relationship between feedforward and feedback control is explicitly stated in the GMC algorithm.

5. Finally and importantly, the GMC framework permits for developing a control algorithm that can be used for both heat-up and temperature maintenance and therefore eliminates the need for a switching criterion between different algorithms; this should result in a much more robust control strategy.

3.3 Kalman Filter

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) solution of the least-squares method. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the model system is unknown.

3.3.1 The discrete Kalman filter

The Kalman filter addresses the general problem of trying to estimate the state $x \in \Re^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{3.12}$$

with a measurement $z \in \Re^m$ that is

$$z_k = Hx_k + v_k \tag{3.13}$$

The random variables w_k and v_k represent the process and measurement noise (respectively) and assume to be independent (of each other), white, and with normal probability distributions

$$p(w) \sim N(0, Q) \tag{3.14}$$

$$p(v) \sim N(0, R) \tag{3.15}$$

In practice, the process noise covariance Q and measurement noise covariance R matrices might change with each time step or measurement, however here they are assumed to be constant.

The n x n matrix A in the difference equation (3.12) relates the state at the previous time step $k-1$ to the state at the current step k, in the absence of either a driving function or process noise. Note that in practice A might change with each time step, but here it is assumed to be constant. The n x l matrix B relates the optional control input $u \in \Re^l$ to the state x. The m x n matrix H in the measurement equation (3.13) relates the state to the measurement z_k. In practice H might change with each time step or measurement, but here it is assumed to be constant.

3.3.1.1 The computational origins of the filter

Define $\hat{x}_k^- \in \Re^n$ to be *a priori* state estimate at step k given knowledge of the process prior to step k, and $\hat{x}_k^- \in \Re^n$ to be *a posteriori* state estimate at step k given measurement z_k. A *priori* and a *posteriori* estimate errors can be defined as

$$e_k^- \equiv x_k - \hat{x}_k^- \tag{3.16}$$

and $$e_k \equiv x_k - \hat{x}_k \tag{3.17}$$

The *a priori* estimate error covariance is then

$$P_k^- = E[e_k^- e_k^{-T}] \tag{3.18}$$

and the *a posteriori* estimate error covariance is

$$P_k = E[e_k e_k^T] \tag{3.19}$$

An *a posteriori* state estimate $\hat{x}_k$ is computed as a linear combination of an *a priori* estimate $\hat{x}_k^-$ and a weighted difference between an actual measurement z_k and a measurement prediction $H\hat{x}_k^-$ as shown below in equation (3.20). Some justification for equation (3.30) is given in "The Probabilistic Origins of the Filter" found below.

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \tag{3.20}$$

The difference $(z_k - H\hat{x}_k^-)$ in equation (3.20) is called the measurement *innovation*, or the *residual*. The residual reflects the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. A residual of zero means that the two are in complete agreement.

The n x m matrix K in equation (3.20) is chosen to be the *gain* or *blending factor* that minimizes the *a posteriori* error covariance equation (3.19). This minimization can be accomplished by first substituting equation (3.20) into the above definition for e_k, substituting that into equation (3.19), performing the indicated expectations, taking the derivative of the trace of the result with respect to K, setting that result equal to zero, and then solving for K. One form of the resulting K that minimizes equation (3.19) is given by:

$$\begin{aligned} K_k &= P_k^- H^T (HP_k^- H^T + R)^{-1} \\ &= \frac{P_k^- H^T}{HP_k^- H^T + R} \end{aligned} \tag{3.21}$$

From equation (3.21) as the measurement error covariance R approaches zero, the gain K weights the residual more heavily.

$$\lim_{R_k \to 0} K_k = H^{-1} \tag{3.22}$$

Another way of thinking about the weighting by K is that as the measurement error covariance R approaches zero, the actual measurement z_k is trusted more and more, while the predicted measurement $H\hat{x}_k^-$ is trusted less and less. On the other hand, as the a priori estimate error covariance P_k^- approaches zero the actual measurement z_k is trusted less and less, while the predicted measurement $H\hat{x}_k^-$ is trusted more and more.

3.1.1.2 The discrete Kalman filter algorithm

The Kalman filter estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of

(noisy) measurements. As such, the equations for the Kalman filter fall into two groups: *time update* equations and *measurement update* equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the *a priori* estimates for the next time step. The measurement update equations are responsible for the feedback – i.e. for incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate.

The specific equations for the time and measurement updates are presented below:

Time Update ("Predict) equations:

- Project the state ahead

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$$

- Project the error covariance ahead

$$P_k^- = A\,P_{k-1}A^T + Q$$

Measurement Update ("Correct") equations

- Compute the Kalman gain

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

- Update estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

- Update the error covariance

$$P_k = (I - K_k H)P_k^-$$

The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations. Indeed the final estimation algorithm resembles that of a predictor-corrector algorithm for solving numerical problems as shown in figure 3.5

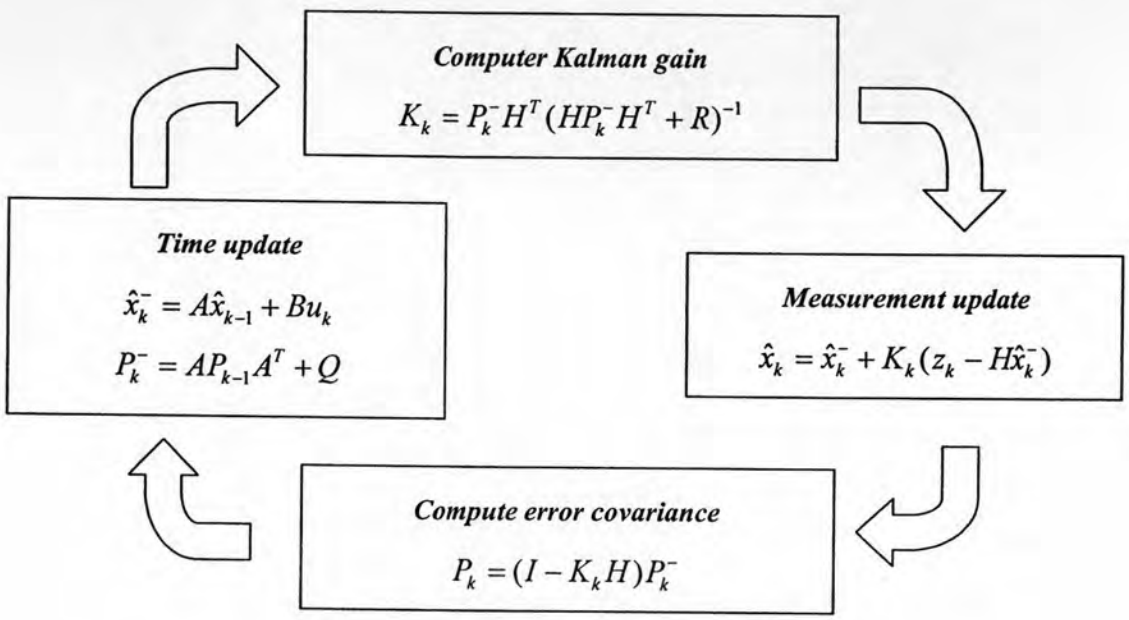


Figure 3.5 The discrete Kalman filter loop

3.3.1.3 Filter parameters and tuning

In the actual implementation of the filter, the measurement noise covariance R is usually measured prior to operation of the filter. Measuring the measurement error covariance R is generally practical and is supposed to be able to measure the process anyway (while operating the filter).

The determination of the process noise covariance Q is generally more difficult because it does not have the ability to directly observe the estimating process. Sometimes a relatively simple (poor) process model can produce acceptable results if one injects enough uncertainty into the process via the selection of Q. Certainly in this case one would hope that the process measurements are reliable.

The tuning of the parameters Q and R is usually performed off-line, frequently with the help of another (distinct) Kalman filter in a process generally referred to as system identification. Under conditions where Q and R are in fact constant, both the estimation error covariance P_k and the Kalman gain K_k will stabilize quickly and then remain constant. If this is the case, these parameters can be pre-computed by either running the filter off-line or for example by determining the steady-state value of P_k.

3.3.2 The extended Kalman filter (EKF)

As described above, the Kalman filter addresses the general problem of trying to estimate the state $x \in \Re^n$ of a discrete-time controlled process that is governed by the linear stochastic different equation. Some of the most interesting and successful applications of Kalman filtering have been such the process to be estimated and (or) the measurement relationship to the process is nonlinear. A Kalman filter that linearizes about the current mean and covariance is referred to as an extended Kalman filter or EKF.

Assuming the process has a state vector $x \in \Re^n$, but the process is now governed by the nonlinear stochastic difference equation

$$x_k = f(x_{k-1}, U_k, w_{k-1}) \tag{3.23}$$

With a measurement $z \in \Re^m$

$$z_k = h(x_k, v_k) \tag{3.24}$$

where the random variables w_k and v_k represent the process and measurement noise. In this case the nonlinear function f in the difference equation (3.23) relates the state at the previous time step $k-1$ to the state at the current time step k. It includes as parameters any driving function u_k and the zero-mean process noise w_k. The nonlinear function h in the measurement equation (3.24) relates the state x_k to the measurement z_k.

In practice of course one does not know the in individual values of the noise w_k and v_k at each time step. However, one can approximate the state and measurement vector as:

$$\tilde{x}_k = f(\tilde{x}_{k-1}, u_k, 0) \tag{3.25}$$

and $$\tilde{z}_k = h(\tilde{x}_k, 0) \tag{3.26}$$

where $\tilde{x}_k$ is some *a posteriori* estimate of the state (from a previous time step *k*).

To estimate a process with nonlinear difference and measurement relationships, begin by writing new governing equations that linearize and estimate equation (3.25) and (3.26),

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + Ww_{k-1} \tag{3.27}$$

$$z_k \approx \hat{z}_k + H(x_k - \tilde{x}_k) + Vv_k \tag{3.28}$$

where x_k and z_k are the actual state and measurement vectors,

$\tilde{x}_k$ and $\tilde{z}_k$ are the approximate state and measurement vectors from equation (3.25) and (3.26),

$\hat{x}_k$ is an *a posteriori* estimate of the state at step k,

A is the Jacobian matrix of partial derivatives of f with respect to x,

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{k-1}, u_k, 0)$$

W is the Jacobian matrix of partial derivatives of f with respect to w,

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}_{k-1}, u_k, 0)$$

H is the Jacobian matrix of partial derivatives of h with respect to x,

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0)$$

V is the Jacobian matrix of partial derivatives of h with respect to v,

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(\tilde{x}_k, 0)$$

Define a new notation for the prediction error

$$\tilde{e}_{x_k} \equiv x_k - \tilde{x}_k \tag{3.29}$$

and the measurement residual

$$\tilde{e}_{z_k} \equiv z_k - \tilde{z}_k \tag{3.30}$$

Using equation (3.29) and (3.30) an error process can be written as:

$$\tilde{e}_{x_k} \approx A(x_{k-1} - \hat{x}_{k-1}) + \varepsilon_k \tag{3.31}$$

$$\tilde{e}_{x_k} \approx H\tilde{e}_{x_k} + \eta_k \tag{3.32}$$

where ε_k and η_k represent new independent random variables having zero mean and covariance matrices WQW^T and VRV^T.

The *a posteriori* state estimates for the original nonlinear process can be obtained by:

$$\hat{x}_k = \tilde{x}_k + \hat{e}_k \tag{3.33}$$

The random variables of equation (3.31) and (3.32) have approximately the following probability distributions:

$$p(\tilde{e}_{x_k}) \sim N(0, E[\tilde{e}_{x_k}\tilde{e}_{x_k}^T])$$
$$p(\varepsilon_k) \sim N(0, WQ_kW^T)$$
$$p(\eta_k) \sim N(0, VR_kV^T)$$

Given these approximations and letting the predicted value of $\hat{e}_k$ be zero, the Kalman filter equation used to estimate $\hat{e}_k$ is

$$\hat{e}_k = K_k \tilde{e}_{z_k} \tag{3.34}$$

Substituting (3.34) back into (3.33) and making use of (3.30) gives:

$$\begin{aligned} \hat{x}_k &= \tilde{x}_k + K_k \tilde{e}_{z_k} \\ &= \tilde{x}_k + K_k (z_k - \tilde{z}_k) \end{aligned} \tag{3.35}$$

The specific equations for the time and measurement updates are presented below:

Time Update ("Predict) equations:

- Project the state ahead

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k, 0)$$

- Project the error covariance ahead

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$$

Measurement Update ("Correct") equations

- Compute the Kalman gain

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$$

- Update estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - h(\hat{x}_k^-, 0))$$

- Update the error covariance

$$P_k = (I - K_k H_k) P_k^-$$

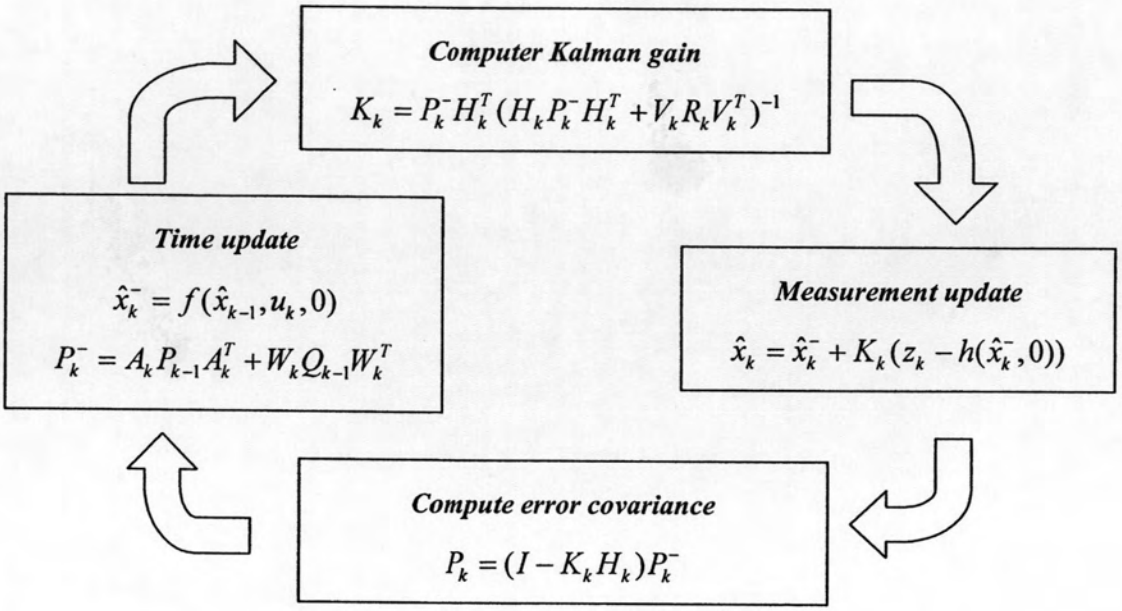


Figure 3.6 The extended Kalman filter loop