6. Model-Based Controller Synthesis

The first half of this book has been primarily concerned with characterizing Volterra models and identifying such models from input/output plant data. Beginning with this chapter, and for the next four chapters, we shift attention to the problem of how these models are used for designing effective controllers.

From a certain perspective, process models in general, and Volterra models in particular, may be viewed as a "direct" mapping of u, the process inputs, to y, the process outputs—"direct" in the sense that given any specific set of inputs u, the corresponding response observed in the process outputs y is obtainable ***directly*** from the process model. This is in contrast to the problem of utilizing this model for control computation, which involves the "inverse" problem of determining the process inputs u required to obtain a given specified set of process outputs, say y^*. It is in this context that model-based control is sometimes considered as a model-inversion problem. Depending on how this "inverse" problem is solved, one finds two distinct approaches to model-based control system design (see Ogunnaike and Ray (1994, chapter 19)): (i) the ***direct synthesis*** approach in which, once the desired process outputs are specified, the process model is used directly to synthesize—***explicitly***—the controller that will cause the actual process outputs to match the specifications exactly; (ii) the ***optimization*** approach (popularly known as ***Model Predictive Control*** (MPC)) in which the desired process output behavior is specified in the form of an objective function and the process model is used to construct—***implicitly***—the controller required to minimize (or maximize) this objective, typically subject to operating constraints. The first approach will be discussed in its basic form in this chapter and in a more advanced form in Chapter 7; the second approach will be discussed in Chapter 8. This chapter will emphasize the role of the model inverse in the synthesis of model-based controllers; Chapter 7 will focus on some of the more advanced issues related to utilizing different variations of this inverse for control systems design and implementation, particularly in the presence of constraints.

The main point of this chapter is that the Volterra model possesses a unique (i.e. partitioned) structure that can be exploited in the synthesis of nonlinear controllers; and our discussion concentrates on the fundamentals: the basic characteristics of the partitioned model, its inverse, and how to use

this inverse for controller design. Many practical issues related to the actual implementation of Volterra model-based controllers are considered more fully in subsequent chapters.

6.1 Introduction

6.1.1 General concepts of nonlinear model-based control

Consider a process whose dynamic behavior is represented by the *nonlinear* input/output correlation implied by the model:

$$\hat{y} = \mathbf{P}[u],$$

where $\mathbf{P}$ is a general nonlinear operator that maps the input u into the (predicted) output, or response, $\hat{y}$. If y represents the actual measurement of the plant output, then the model error obtained as:

$$e = y - \hat{y}$$

enables us to write:

$$y = \mathbf{P}[u] + e$$

as the relationship between the plant input and the actual plant output.

Given y^* as the desired trajectory for the actual plant output y to follow, the control action u that satisfies the following objective:

$$\min_u \phi = ||y^* - y|| \tag{6.1}$$

is easily obtained *explicitly* as:

$$u = \mathbf{P}^{-1}[y^* - e],$$

provided the inverse of the operator $\mathbf{P}$ *exists*. If y^* is chosen as y_{set} (the *setpoint* for y), then:

$$u = \mathbf{P}^{-1}[y_{set} - e]. \tag{6.2}$$

Note here that nominal behavior is concerned with the case $e = 0$, for which Equation (6.2) implies an *open-loop* control policy; feedback appears only in the presence of modeling error and unmeasured disturbances ($e \neq 0$).

Such a synthesis technique raises several theoretical issues that are best confronted later when we consider a *specific* form for the nonlinear operator $\mathbf{P}$ and in Chapter 7; in the meantime we note in general that:

1. $\mathbf{P}^{-1}$ may not exist; and when it does, it may not be realizable;
2. even when $\mathbf{P}^{-1}$ exists (and is realizable), it is clear that Equation (6.2) results in so-called "perfect control," with all the attendant robustness and implementation problems;

3. the model error e is composed of an essentially inseparable combination of unmodelled dynamics, unmeasured disturbances, and noise components; responding to this signal directly as implied in Equation (6.2) will result in serious robustness problems.

The standard approach to this problem is the introduction of a setpoint "filter" defined by:

$$y^* = \mathbf{F}[y_{set}]$$

and a systematic "error estimator" $\mathbf{E}$, which produces $\tilde{e}$ (given y, $\mathbf{P}$ and u), and a resulting $\tilde{y}$, according to:

$$\tilde{e} = \mathbf{E}[y, \mathbf{P}, u]$$

$$\tilde{y} = \mathbf{P}[u] + \tilde{e}.$$

By modifying the objective in Equation (6.1) using $\tilde{y}$ in place of y, the result in Equation (6.2) is modified to read:

$$u = \mathbf{P}^{-1}\left[\mathbf{F}[y_{set}] - \tilde{e}\right] \tag{6.3}$$

In principle, the two operators $\mathbf{E}$ and $\mathbf{F}$ may be chosen ***appropriately*** to address the problems of realizability, robustness and implementation raised by the presence of the operator inverse in Equation (6.2). Thus conceptually, the controller synthesis problem as posed here involves the construction of $\mathbf{P}^{-1}$ given $\mathbf{P}$, and the choice of $\mathbf{E}$ and $\mathbf{F}$. The resulting control system is represented in block diagrammatic form in Figure 6.1, where $\mathbf{Q}$ denotes the controller, here given by $\mathbf{Q} = \mathbf{P}^{-1}$.

The choice of $\mathbf{E}$ and $\mathbf{F}$ and how each is implementated clearly depend on the nature of $\mathbf{P}$ and its inverse; and no useful, general statement can be made either about conditions for the existence of the inverse of the *general* nonlinear operator $\mathbf{P}$, or about the procedure for its construction and implementation, without considering some special cases. Nevertheless, we note that if $\mathbf{P}$ were a purely linear operator $\mathbf{L}$, then fairly general statements can be made about the nature, construction and implementation of its inverse. Furthermore, if $\mathbf{E}$ is chosen simply as a ***linear filter*** operating on the error e, and this filter is chosen in particular to be the same as the setpoint filter $\mathbf{F}$, i.e.

$$\tilde{e} = \mathbf{F}[e],$$

and if this $\mathbf{F}$ is chosen to have as one of its properties:

$$\lim_{t\to\infty} \mathbf{F}[x(t)] = \lim_{t\to\infty} x(t), \tag{6.4}$$

then Equation (6.3) becomes:

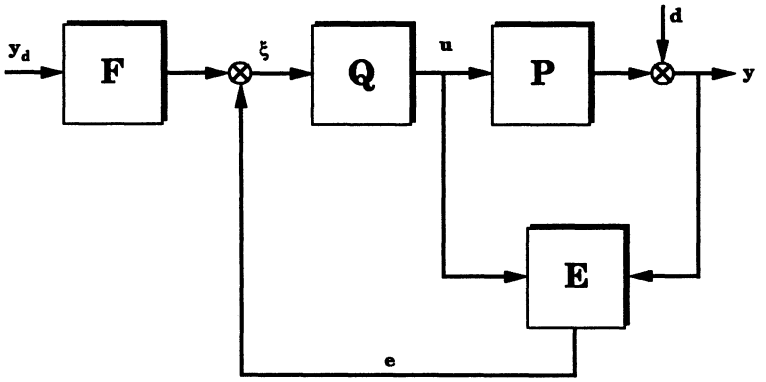

Fig. 6.1. General model-based control structure

$$u = \mathbf{P}^{-1}\mathbf{F}[y_{set} - e]$$

which is immediately recognizable as the *standard* IMC strategy (see Frank (1974) and García and Morari (1982)). The issues involved in implementing such a control scheme in practice is discussed in greater detail in Chapter 7.

As will be discussed more fully in Chapter 8, it is often quite advantageous *not* to seek an explicit solution to the problem posed in Equation (6.1); rather a computational scheme for obtaining an implicit solution via numerical optimization is to be preferred. The advantages go beyond merely avoiding the potential problems raised above concerning the explicit inverse of the operator **P**. Such a computational approach provides additional flexibility, the most important of which is that process constraints can be incorporated naturally. Yet, even this alternative approach, popularly known as MPC, can be represented as in Figure 6.1. The nature and form of the "estimator" **E** and the controller **Q** within this optimization framework will be discussed in detail in Chapter 8 specifically for the case when **P** is a Volterra model.

6.1.2 The partitioned nonlinear model

indexpartitioned nonlinear model Consider the case in which the nonlinear operator **P** can be partitioned into a linear and nonlinear portion as follows:

$$\mathbf{P} = \mathbf{L} + \mathbf{N}, \tag{6.5}$$

then if the inverse of the *linear portion* exists, a simple rearrangement yields:

$$\mathbf{P} = \mathbf{L}(\mathbf{I} + \mathbf{L}^{-1}\mathbf{N}),$$

from where we obtain:

$$\mathbf{P}^{-1} = (\mathbf{I} + \mathbf{L}^{-1}\mathbf{N})^{-1}\mathbf{L}^{-1} \tag{6.6}$$

and the existence of such an inverse depends on the existence of the two inverses indicated above. We may now observe the following:

1. it is fairly straightforward to comment on the existence of the inverse of a general linear operator $\mathbf{L}$; and
2. while it is less straightforward to comment in general on the existence of $(\mathbf{I}+\mathbf{L}^{-1}\mathbf{N})^{-1}$, we may note that the norm of the composite operator $\mathbf{L}^{-1}\mathbf{N}$ (provided it exists)—clearly a measure of the "strength" of the nonlinearity of $\mathbf{P}$ relative to its linear component—will have a significant influence on the existence of this operator inverse;
3. more importantly, however, if these two inverses exist, then the required nonlinear operator inverse $\mathbf{P}^{-1}$ can be constructed and implemented as shown in Figure 6.2: *requiring only the inverse of the linear portion.*

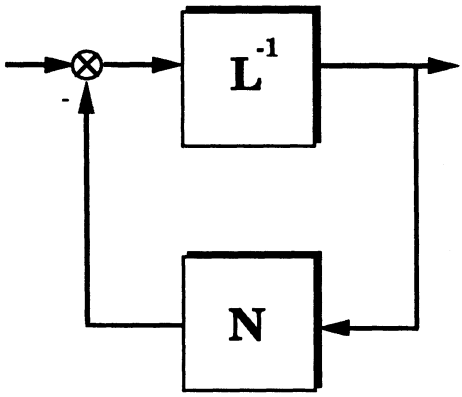

Fig. 6.2. Realization of partitioned nonlinear model inverse

The advantages accruing if the nonlinear operator admits of the representation in Equation (6.5) are now clear: the nature of the operator inverse and the conditions for its existence are easier to determine; its construction and its implementation depend in a remarkably straightforward manner only on the linear subsystem inverse. We now note that the Volterra model not only naturally occurs in the partitioned form in Equation (6.5), but its linear portion is precisely what conventional MPC is based upon. The rest of this chapter is devoted to exploiting these facts for laying the foundation of a scheme to extend conventional linear model-based control schemes to nonlinear systems on the basis of such model inversions.

6.2 Volterra model-based controller synthesis

6.2.1 Basic results

Linear systems. For the linear system modeled by:

$$y = \mathbf{P}_1[u] \tag{6.7}$$

the *basic* controller synthesis result is:

$$u = \mathbf{P}_1^{-1}[y_{set} - e],$$

As noted previously, this may be modified as needed using the "estimator" $\mathbf{E}$, and the "filter" $\mathbf{F}$, but fundamentally this controller is based on $\mathbf{P}_1^{-1}$, the linear model inverse.

Nonlinear systems. Of the class of Volterra models, the simplest nonlinear extension of the linear convolution model in Equation (6.7) is:

$$y = (\mathbf{P}_1 + \mathbf{P}_2)[u]$$

the second-order Volterra model, where $\mathbf{P}_1$ is the linear part and $\mathbf{P}_2$ is the quadratic part. From the result in the Section 6.1.2 (see Equation (6.6)), the controller synthesis procedure yields, in this case

$$u = (\mathbf{I} + \mathbf{P}_1^{-1}\mathbf{P}_2)^{-1}\mathbf{P}_1^{-1}[\xi], \tag{6.8}$$

where:

$$\xi = y_{set} - e, \tag{6.9}$$

The important point to note here is the fact that such a controller can in fact be implemented as shown in Figure 6.2: the "main controller" still remains the linear model inverse, but it is now augmented with the auxiliary loop incorporating the quadratic nonlinear operator $\mathbf{P}_2$.

It is now a straightforward exercise to show that in the general case, with

$$\mathbf{P} = \mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \ldots$$

the corresponding controller expression is:

$$\mathbf{Q} = \mathbf{P}^{-1} = (\mathbf{I} + \mathbf{P}_1^{-1}\mathbf{P}_2 + \mathbf{P}_1^{-1}\mathbf{P}_3 + \ldots)^{-1}\mathbf{P}_1^{-1} \tag{6.10}$$

and the corresponding block diagram will now have the residual nonlinear operators $(\mathbf{P}_2 + \mathbf{P}_3 + \ldots)$ in the auxiliary loop.

In Section 6.2.2 we derive the result in Equation (6.8) using expansions of Volterra series operators (and their inverses). Once the equivalence is established between the two approaches, we will use the Volterra series expansion approach to derive the operator $\mathbf{Q}$ for broad classes of dynamical systems using generalized inverses.

6.2.2 The "standard" approach

An alternative approach to deriving the controller in Equation (6.8) is outlined below. Consider a controller operator $\mathbf{Q}$ (to be determined) represented by:

$$u = \mathbf{Q}[\xi]$$

with ξ as previously defined in Equation (6.9). Substituting these expressions for $\mathbf{Q}$ and ξ into Equation (6.1) yields the control problem:

$$\min_{Q} \phi = ||\xi - \mathbf{P}[\mathbf{Q}[\xi]]||.$$

This objective will clearly be minimized if the following relationship holds:

$$\mathbf{P}[\mathbf{Q}[\xi]] = \xi. \tag{6.11}$$

If we define the nonlinear operator $\mathbf{H}$ as:

$$\mathbf{H} = \mathbf{P} * \mathbf{Q},$$

where $*$ denotes the composition operation, then the operator $\mathbf{Q}$ that will satisfy Equation (6.11) precisely must clearly be the inverse of $\mathbf{P}$ (provided that it exists), in which case $\mathbf{H}$ will be the identity operator $\mathbf{I}$. We now present a procedure for computing the operator $\mathbf{Q}$ given the process model operator $\mathbf{P}$. In following such a procedure, however, as noted in Schetzen (1980, chapter 7), one should keep in mind that not all nonlinear operators $\mathbf{P}$ possess such an inverse; also, many possess an inverse only for a restricted range of input amplitude. One must therefore be careful to ensure that the range of the operator $\mathbf{Q}$ is contained in the domain of the operator $\mathbf{P}$. This will be assumed throughout the following discussion. For further discussion of the various implications of this statement, see, for example, Schetzen (1980, chapter 7).

To obtain an explicit expression for $\mathbf{Q}$, using the operator notation defined above, with the plant represented by the operator:

$$\mathbf{P} = \mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \dots$$

the controller operator $\mathbf{Q}$ is also set up as an infinite sequence of operators:

$$\mathbf{Q} = \mathbf{Q}_1 + \mathbf{Q}_2 + \mathbf{Q}_3 + \dots \tag{6.12}$$

with:

$$\mathbf{Q}_1[\xi] = \sum_{i=1}^{N} v(i)\ \xi(k-i)$$

$$\mathbf{Q}_2[\xi] = \sum_{i=1}^{N}\sum_{j=1}^{N} w(i,j)\ \xi(k-i)\xi(k-j)$$

$$\mathbf{Q}_3[\xi] = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N} \psi(i,j,l)\ \xi(k-i)\xi(k-j)\xi(k-l)$$

in the discrete case, with the corresponding integral expressions in the continuous case.

Thus the controller synthesis problem requires solving the equation

$$\mathbf{I} = \mathbf{P} * \mathbf{Q} = (\mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \ldots) * (\mathbf{Q}_1 + \mathbf{Q}_2 + \mathbf{Q}_3 + \ldots) \tag{6.13}$$

for the controller $\mathbf{Q}$. By expanding the composition of the operators indicated in Equation (6.13) and rearranging and grouping the results according to operator order and equating like terms, it can be shown that the controller operator will be given explicitly by:

$$\begin{aligned}
\mathbf{Q}_1 &= \mathbf{P}_1^{-1} \\
\mathbf{Q}_2 &= -\mathbf{P}_1^{-1} * \mathbf{P}_2 * \mathbf{Q}_1 \\
\mathbf{Q}_3 &= -\mathbf{P}_1^{-1} * (\mathbf{P}_2 * (\mathbf{Q}_1 + \mathbf{Q}_2) - \mathbf{P}_2 * \mathbf{Q}_1 - \mathbf{P}_2 * \mathbf{Q}_2 + \mathbf{P}_3 * \mathbf{Q}_1) \\
&\vdots = \vdots
\end{aligned}$$

Here, $*$ indicates a composition of the operators concerned. Note that this controller is an infinite sequence of operators, that the first term in the sequence is the linear model inverse, and that this is the only inverse operator present in the subsequent terms.

A key property of these expressions for the controller $\mathbf{Q}$ as the Volterra operator inverse is the fact that the calculation of successive terms is completely decoupled. In other words, a linear operator inversion is all that is required initially, and subsequent terms depend only upon the same linear inverse, previous terms in the inverse operator series, and terms from the original plant operator series. However, there are three important issues that this approach does not address:

1. Since $\mathbf{Q}$ is obtained by this technique as an infinite sequence of operators, it is pertinent to contemplate the issue of convergence. Under what conditions will the sequence in Equation (6.12) converge?
2. And even when the sequence can be shown to converge, how is the controller to be implemented in practice, particularly when the rate of convergence is slow?
3. What happens when the inverse of $\mathbf{P}_1$ either does not exist, or is unrealizable?

The procedure we have introduced in Section 6.1.2 resolves the first two issues because:

1. it provides an exact, analytical expression for the controller $\mathbf{Q}$ that was represented as an infinite sequence using the alternative approach;
2. as shown previously, it results in a controller that can be implemented as shown in the block diagram in Figure 6.2.

This analytic representation for the controller, therefore, has the dual advantage that it does not depend on the properties of an infinite series, and its implementation is straightforward and intuitive.

The third issue is resolved using generalized inverses, as we now discuss.

6.2.3 Controller synthesis using generalized inverses

In recognition of the fact that there will be situations in which $\mathbf{P}_1$, the linear (degree one) operator, either has no inverse, or its inverse is undesirable for any number of reasons, we now proceed to develop explicit expressions for the controller in terms of appropriate generalized inverses.

Problem formulation and solution. In finding the *exact* inverse of $\mathbf{P}$ and using it as the controller $\mathbf{Q}$, the strategy presented in Section 6.2.2 was to solve the series of equations resulting from Equation (6.13). For our current purposes, we now reformulate the problem as follows: since model prediction using $\mathbf{P}$ and control action using $\mathbf{Q}$ are respectively determined according to:

$$y = \mathbf{P}[u]$$

$$u = \mathbf{Q}[\xi],$$

then from the definition of each operator, we have first that:

$$y = y^{(1)} + y^{(2)} + y^{(3)} + \dots , \tag{6.14}$$

where $y^{(i)}$ is the degree i contribution to the overall signal y, arising from:

$$y^{(i)} = \mathbf{P}_i[u]. \tag{6.15}$$

Similarly, we also have:

$$u = u^{(1)} + u^{(2)} + u^{(3)} + \dots , \tag{6.16}$$

where $u^{(i)}$ is the degree i contribution to the overall control action signal u, arising from:

$$u^{(i)} = \mathbf{Q}_i[\xi].$$

Finally, observe from Equations (6.14) and (6.15) that:

$$y = \mathbf{P}_1[u] + \mathbf{P}_2[u] + \mathbf{P}_3[u] + \dots ,$$

but from Equation (6.16) u itself is the indicated infinite sequence. We now introduce the notation:

$$\delta y^{(ij)} = \mathbf{P}_i[u^{(j)}],$$

with the first index denoting the degree of the *operator* and the second index the contributing degree of the component of the signal being operated upon. Thus, for example, $\delta y^{(12)}$ is that portion of $y^{(1)}$, the degree one contribution to y, due to $u^{(2)}$, the degree two component of the input signal u: it is obtained by operating on $u^{(2)}$ by $\mathbf{P}_1$.

Let r be the residual signal, the difference between the plant/controller operator composition $\mathbf{H}$ acting on ξ and the desired value of this composition (i.e. ξ itself):

$$r = \mathbf{H}[\xi] - \xi. \tag{6.17}$$

Now let r be defined in the same manner as y and u in Equations (6.14) and (6.16), so that:

$$r = r^{(1)} + r^{(2)} + r^{(3)} + \ldots .$$

That is, $r^{(i)}$ is given by

$$r^{(1)} = \mathbf{H}_1[\xi] - \xi$$

and

$$r^{(j)} = \mathbf{H}_j[\xi]$$

for $j = 2,3, \ldots$, where $\mathbf{H}_j$ is the j^{th} order term in the expansion of the composition $\mathbf{H}$ defined in Equation (6.13). Observe now that obtaining the *exact inverse* of $\mathbf{P}$ (demanding that $\mathbf{H} = \mathbf{I}$) is equivalent to requiring

$$r^{(i)} = 0 \tag{6.18}$$

for all $i = 1, 2, 3, \ldots$. As shown previously, this requires the existence of $\mathbf{P}_1^{-1}$. It is possible to relax this requirement and, instead of Equation (6.18), require only that the norm $\|r^{(i)}\|$ be minimized.

It is important to note that the required minimization is on each individual degree i contributor to the residual signal, not the composite residual signal itself. What is required here is equivalent to minimizing the *operator norm* of the difference between corresponding degree i operators, $\mathbf{I}_i$ and $\mathbf{H}_i$, for each i. This clearly gives rise to a much stronger result, since it guarantees that each degree i operator is as "close" as possible to the ideal operator of corresponding degree, and not merely that the overall operator $\mathbf{H}$ is as "close" to $\mathbf{I}$ as possible.

Remarks

There are some subtle issues of mathematical rigor associated with the solution of the minimization problems we have posed.

1. First, there is the issue of the precise definition of the domain and range of the yet undetermined, obviously nonlinear, operator $\mathbf{Q}$ that we seek to derive via optimization. Discussions of this kind for the Volterra operator $\mathbf{P}$ employed for the plant model may be found, for example in Rugh (1981, section 1.5). The converse discussion, in which it is recognized that in determining $\mathbf{Q}$ as the inverse of $\mathbf{P}$, the domain of the latter is the range of the former, and vice versa, may be found in Schetzen (1980, chapter 7). It is relatively easy to show that the same arguments hold in this case, since what we are deriving is, in fact, a less stringent inverse of $\mathbf{P}$.
2. A more serious issue has to do with the convergence of $\mathbf{Q}$. It is clear from the earlier discussion that, in general, a stable, causal $\mathbf{Q}$ does not exist as an exact inverse of $\mathbf{P}$ if $\mathbf{P}_1^{-1}$ is not stable or causal. However,

as we will soon show, a stable, causal $\mathbf{Q}$, which will depend on a generalized inverse for $\mathbf{P}_1$, may be obtained under these circumstances. Such a $\mathbf{Q}$ will of course be a pseudo-inverse of $\mathbf{P}$, but by choosing the criteria for pseudo-inversion carefully (see Ben-Israel and Greville (1974)) we can guarantee stability and causality for $\mathbf{Q}$. Nevertheless, even in this case, the resulting operator $\mathbf{Q}$, expressed as the Volterra series in Equation (6.12), may converge only under a limited range of the amplitude of its input ξ. Further, as indicated in Schetzen (1980, section 7.5), this range is not easy to determine in general. These facts notwithstanding, we may make an appeal to the earlier discussion and employ the "filter" $\mathbf{F}$ and the estimator $\mathbf{E}$ in the feedback loop as a means of ensuring that the amplitude of the feedback signal ξ is sufficiently modified for the purposes of achieving convergence for $\mathbf{Q}$.

3. The notion of "inversion" that is used here is fairly general and can be relaxed, from a numerical perspective (Economou and Morari, 1985) or from a performance perspective, to yield robust controlled behavior. For instance, the analytical solution to the standard unconstrained MPC problem contains an explicit weighting term that relaxes the explicit model inversion as a direct consequence of incorporating a penalty on the manipulated variable. For a nice summary of this notion of relaxed model inversion, see Hernández (1992).
4. Finally, even though we will soon show that the solution of the specific optimization problem posed above *technically* involves only the linear operator $\mathbf{P}_1$ explicitly, it is, nevertheless, also clear from the Equation (6.17) that at the foundation of the entire exercise is the notion of the norm of nonlinear operators (for $i > 1$). Although the notion of the norm of a linear operator is straightforward and well known, it is not as obvious or as straightforward with nonlinear operators. It is clearly outside the intended scope of this book to provide even a cursory discussion of the precise mathematical foundations underlying the operations indicated in Equation (6.17) and which make it possible to derive from it the results we will soon present. It is sufficient to note that most of the pertinent issues are taken up with enough detail, for example, in Berger (1977, chapter 2). The specific issues regarding the spaces of nonlinear operators, as well as those regarding the existence of solutions to nonlinear operator equations in Banach spaces (such as the kind we have posed in Equation (6.17)) are considered in detail in Martin (1976, chapters 3 and 4). The interested reader is encouraged to consult these references.

We now proceed to obtain the solution to the optimization problem.

Degree 1

In this case, since:

$$\mathbf{H}_1[\xi] = \mathbf{P}_1 * \mathbf{Q}_1[\xi]$$

and $\mathbf{I}_1[\xi] = \xi$, we have:

$$r^{(1)} = \mathbf{P}_1 * \mathbf{Q}_1[\xi] - \xi,$$

but by definition $\mathbf{Q}_1[\xi] = u^{(1)}$, so that the optimization problem becomes:

$$\|r^{(1)}\| = \|\mathbf{P}_1[u^{(1)}] - \xi\|. \tag{6.19}$$

If, in particular $\|.\|$ refers to the L_2 norm, then Equation (6.19) has the well-known solution:

$$u^{(1)} = \mathbf{P}_1^{\dagger}[\xi], \tag{6.20}$$

where the operator $\mathbf{P}_1^{\dagger}$ is the *left inverse* of $\mathbf{P}_1$ defined by:

$$\mathbf{P}_1^{\dagger}\mathbf{P}_1 = \mathbf{I}_1.$$

Of course, since, by definition, $u^{(1)} = \mathbf{Q}_1[\xi]$, it follows from Equation (6.20) that the linear portion of the $\mathbf{Q}$ operator is given by:

$$\mathbf{Q}_1 = \mathbf{P}_1^{\dagger}.$$

(Compare with $\mathbf{Q}_1 = \mathbf{P}_1^{-1}$ for the exact inverse controller.)

Degree 2

In this case,

$$\mathbf{H}_2[\xi] = \mathbf{P}_1 * \mathbf{Q}_2[\xi] + \mathbf{P}_2 * \mathbf{Q}_1[\xi]$$

or, equivalently,

$$\mathbf{H}_2[\xi] = \mathbf{P}_1[u^{(2)}] + \mathbf{P}_2[u^{(1)}]. \tag{6.21}$$

If we keep in mind that from Equation (6.20) $u^{(1)}$ is now a known quantity, and recall the definition of $\delta y^{(ij)}$, we observe immediately that the expression in Equation (6.21) becomes:

$$\mathbf{H}_2[\xi] = \mathbf{P}_1[u^{(2)}] + \delta y^{(21)};$$

and finally, since $\mathbf{I}_2[\xi] = 0$, we then have the second degree optimization problem as:

$$\|r^{(2)}\| = \|\mathbf{P}_1[u^{(2)}] + \delta y^{(21)}\|,$$

a problem entirely similar in structure to Equation (6.19), and involving only the linear operator $\mathbf{P}_1$, even though the input signal in question, $u^{(2)}$, is a second degree contribution. The solution is:

$$u^{(2)} = -\mathbf{P}_1^{\dagger}[\delta y^{(21)}]$$

with the same $\mathbf{P}_1^{\dagger}$ as in Equation (6.20).

Degree 3

Here the problem is somewhat more complicated, but the principles are essentially the same. In this case

$$\mathbf{H}_3[\xi] = \mathbf{P}_1 * \mathbf{Q}_3[\xi] + \mathbf{P}_3 * \mathbf{Q}_1[\xi] - \mathbf{P}_2 * \mathbf{Q}_1[\xi] - \mathbf{P}_2 * \mathbf{Q}_2[\xi] + \mathbf{P}_2 * [\mathbf{Q}_1[\xi] + \mathbf{Q}_2[\xi]]$$

or, equivalently,

$$\mathbf{H}_3[\xi] = \mathbf{P}_1[u^{(3)}] + \mathbf{P}_3[u^{(1)}] - \mathbf{P}_2[u^{(1)}] - \mathbf{P}_2[u^{(2)}] + \mathbf{P}_2[u^{(1)} + u^{(2)}]. \tag{6.22}$$

Since, by now, both $u^{(1)}$ and $u^{(2)}$ are known quantities, we may simplify Equation (6.22) by defining:

$$\delta\xi^{(3)} = \delta y^{(31)} - \delta y^{(21)} - \delta y^{(22)} + \delta y^{(2S_{12})}$$

with $\delta y^{(2S_{12})}$ defined as:

$$\delta y^{(2S_{12})} = \mathbf{P}_2[u^{(1)} + u^{(2)}]$$

and the others as previously defined; the result is that Equation (6.22) becomes:

$$\mathbf{H}_3[\xi] = \mathbf{P}_1[u^{(3)}] + \delta\xi^{(3)}$$

and again, since $\mathbf{I}_3[\xi] = 0$, we have the third degree optimization problem as:

$$\|r^{(3)}\| = \|\mathbf{P}_1[u^{(3)}] + \delta\xi^{(3)}\|.$$

Observe again that this is a problem entirely similar in structure to Equation (6.19), involving again only the linear operator $\mathbf{P}_1$. (The input signal in question, $u^{(3)}$, of course, is a third degree contribution.) The solution again is:

$$u^{(3)} = -\mathbf{P}_1^{\dagger}[\delta\xi^{(3)}]$$

with the same $\mathbf{P}_1^{\dagger}$ in the degrees one and two problems.

It is now possible to generalize: the degree i problem will always have the form:

$$\|r^{(i)}\| = \|\mathbf{P}_1[u^{(i)}] + \delta\xi^{(i)}\|,$$

where $\delta\xi^{(i)}$ consists only of known quantities dependent on $u^{(i-1)}$, and other components of lower degree up to and including $u^{(1)}$.

We now observe, once again, that as previously obtained under the assumption that $\mathbf{P}_1^{-1}$ exists, the controller implied above is an infinite sequence, with each term involving only $\mathbf{P}_1^{\dagger}$ this time. It can be shown that the corresponding analytical expression for this infinite sequence controller is precisely as in Equations (6.8) and (6.10) but now with $\mathbf{P}_1^{-1}$ replaced with $\mathbf{P}_1^{\dagger}$.

Implications for constrained discrete MPC. The fact that the problem of generating the controller $\mathbf{Q}$ can be thus posed as an optimization problem has some significant implications for controller implementation within the explicit MPC framework. Observe that the norms to be minimized could be weighted; and the weights could in fact be made dependent on the operator degree; constraints could also be introduced (in the multivariable case) if

needed. Furthermore, the entire controller could be implemented as in Figure 6.3, with $\mathbf{P}_1^{-1}$ replaced by the linear MPC controller, noting the crucial point that the auxiliary loop endows the otherwise well known linear MPC controller with nonlinear characteristics by providing a form of "correction" for the effect of nonlinearities.

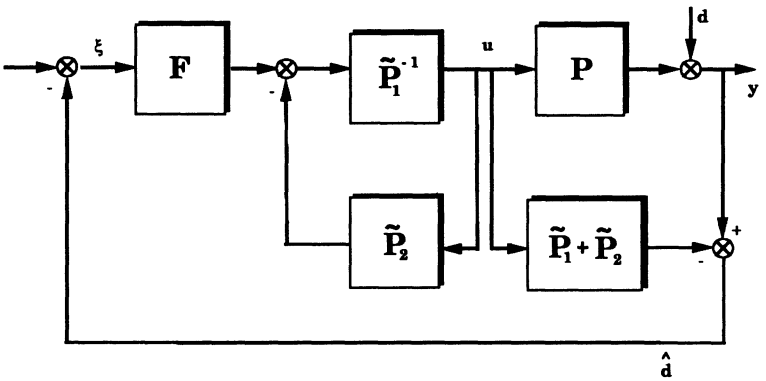

Fig. 6.3. Structure for second-order Volterra controller

These and other related issues are significant enough in their own rights that they warrant a full-scale discussion, which we shall defer to Chapter 8.

6.3 Summary

In this chapter, the first of four chapters devoted to nonlinear controller design and implementation, we have presented a systematic procedure for model-based controller synthesis that exploits the unique structure of Volterra models. The role of the model inverse in the synthesis of model-based controllers was central to these discussions; and we saw that the particular structure of Volterra models as partitioned nonlinear models lent itself nicely to a design procedure that rests on the inversion of only the linear portion of the model. The controller can then be implemented as a linear controller augmented with an auxiliary loop incorporating corrections for the effect of nonlinearities. Thus, the computational requirements for this procedure are minimal: (i) a second-order Volterra model, and (ii) a linear controller derived from the inverse (or pseudo-inverse) of the linear portion of the process model.

Many issues of practical importance (for example constraints, and other more advanced problems) were not discussed here; these are reserved for Chapter 7. Also, model predictive control of nonlinear processes using the Volterra model has been deferred until Chapter 8. After discussing these important topics, Chapter 9 presents several case studies that illustrate the

design and performance of the entire collection of nonlinear model-based controllers.