5
Adaptive and Model Predictive Control

Contents

In this chapter, we discuss some of the core control system design methodologies within the context of our approximation in value space framework. In particular, in the next two sections, we will discuss problems with unknown or changing problem parameters, and briefly review some of the principal types of adaptive control methods. We will then focus on schemes that are based on on-line replanning, including the use of rollout. The idea here is to use an approximation in value space scheme/Newton step in place of a full reoptimization of the controller, in response to the changed system parameters; we have noted this possibility in Chapter 1. Subsequently, in Sections 5.3 and 5.4, we will discuss the model predictive control methodology, and its connections with approximation in value space, Newton's method, adaptive control, and the attendant stability issues.

5.1 SYSTEMS WITH UNKNOWN PARAMETERS - ROBUST AND PID CONTROL

Our discussion so far dealt with problems with a known and unchanging mathematical model, i.e., one where the system equation, cost function, control constraints, and probability distributions of disturbances are perfectly known. The mathematical model may be available through explicit mathematical formulas and assumptions, or through a computer program that can emulate all of the mathematical operations involved in the model, including Monte Carlo simulation for the calculation of expected values. From our point of view, it makes no difference whether the mathematical model is available through closed form mathematical expressions or through a computer simulator: the methods that we discuss are valid either way, only their suitability for a given problem may be affected by the availability of mathematical formulas.

In practice, however, it is common that the system involves parameters that are either not known exactly or may change over time. In such cases it is important to design controllers that take the parameter changes into account. The methodology for doing so is generally known as *adaptive control*, an intricate and multifaceted subject, with many and diverse applications, and a long history.†

We should note also that unknown problem environments are an integral part of the artificial intelligence view of RL. In particular, to quote

† The difficulties of designing adaptive controllers are often underestimated. Among others, they complicate the balance between off-line training and on-line play, which we discussed in Chapter 1 in connection to AlphaZero. It is worth keeping in mind that as much as learning to play high quality chess is a great challenge, the rules of play are stable and do not change unpredictably in the middle of a game! Problems with changing system parameters can be far more challenging!

from the book by Sutton and Barto [SuB18], "learning from interaction with the environment is a foundational idea underlying nearly all theories of learning and intelligence." The idea of interaction with the environment is typically connected with the idea of exploring the environment to identify its characteristics. In control theory this is often viewed as part of the *system identification* methodology, which aims to construct mathematical models of dynamic systems by using data. The system identification process is often combined with the control process to deal with unknown or changing problem parameters. This is one of the most challenging areas of stochastic optimal and suboptimal control, and has been studied extensively since the early 1960s.

Robust and PID Control

Given a controller design that has been obtained assuming a nominal DP problem model, one possibility is to simply ignore changes in problem parameters. We may then try to design a controller that is adequate for the entire range of the changing parameters. This is sometimes called a *robust controller*. A robust controller makes no effort to keep track of changing problem parameters. It is just designed so that it is resilient to parameter changes, and in practice, it often tends to be biased towards addressing the worst case.

An important and time-honored robust control approach for continuous-state problems is the *PID (Proportional-Integral-Derivative) controller*; see e.g., the books by Aström and Hagglund [AsH95], [AsH06]. In particular, PID control aims to maintain the output of a single-input single-output dynamic system around a set point or to follow a given trajectory, as the system parameters change within a relatively broad range. In its simplest form, the PID controller is parametrized by three scalar parameters, which may be determined by a variety of methods, some of them manual/heuristic. PID control is used widely and with success, although its range of application is mainly restricted to relatively simple, single-input and single-output continuous-state control systems.

Combined System Identification and Control

In robust control schemes, such as PID control, no attempt is made to maintain a mathematical model and to track unknown model parameters as they change. Alternatively we may introduce into the controller a mechanism for measuring or estimating the unknown or changing system parameters, and make suitable control adaptations in response.†

† In the adaptive control literature, schemes that involve parameter estimation are sometimes called *indirect*, while schemes that do not involve parameter estimation (like PID control) are called *direct*. To quote from the book by Aström

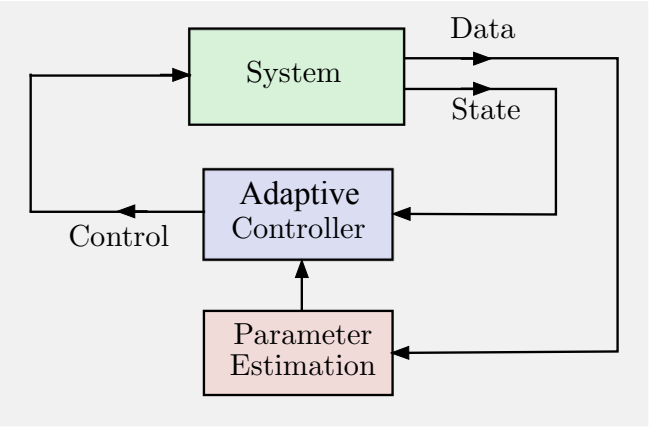

Figure 5.1.1 Schematic illustration of concurrent parameter estimation and system control. The system parameters are estimated on-line and the estimates are passed on to the controller whenever this is desirable (e.g., after the estimates change substantially). This structure is also known as indirect adaptive control.

Let us note here that updating problem parameters need not require an elaborate algorithm. In many cases the set of problem parameters may take a known finite set of values (for example each set of parameter values may correspond to a distinct maneuver of a vehicle, motion of a robotic arm, flying regime of an aircraft, etc). Once the control scheme detects a change in problem parameters, it can incorporate the change into the approximation in value space scheme, and in the case of policy rollout, it may switch to a corresponding predesigned base policy.

In what follows in this chapter (including our discussion of MPC in Section 5.3), we will assume that there is a mechanism to learn (perhaps imperfectly and by some unspecified procedure) the model of the system as it evolves over time. We will loosely refer to this learning process with the classical name *system identification*, but we will not go into specific identification methods, keeping in mind that such methods could be imprecise and challenging, but could also be fast and simple, depending on the problem at hand.

An apparently reasonable scheme is to separate the control process into two phases, a *system identification phase* and a *control phase*. In the first phase the unknown parameters are estimated, while the control takes no account of the interim results of estimation. The final parameter

and Wittenmark [AsW08], "indirect methods are those in which the estimated parameters are used to calculate required controller parameters" (see Fig. 5.1.1). The methods subsequently described in this section, and the rollout-based adaptive control methods discussed in the next section should be viewed as indirect.

estimates from the first phase are then used to implement an optimal or suboptimal policy in the second phase.

This alternation of estimation and control phases may be repeated several times during the system's operation in order to take into account subsequent changes of the parameters. Note that it is not necessary to introduce a hard separation between the identification and the control phases. They may be going on simultaneously, with new parameter estimates being generated in the background, and introduced into the control process, whenever this is thought to be desirable; see Fig. 5.1.1.

One drawback of this approach is that it is not always easy to determine when to terminate one phase and start the other. A second difficulty, of a more fundamental nature, is that the control process may make some of the unknown parameters invisible to the estimation process. This is known as the problem of *parameter identifiability*, which is discussed in the context of adaptive control in several sources. On-line parameter estimation algorithms, which address among others the issue of identifiability, have been discussed extensively in the control theory literature, but the corresponding methodology is complex and beyond our scope in this book. However, assuming that we can make the estimation phase work somehow, we are free to reoptimize the controller using the newly estimated parameters, in a form of on-line replanning process.

Unfortunately, there is still another difficulty with this type of on-line replanning: it may be hard to recompute an optimal or near-optimal policy on-line, using a newly identified system model. In particular, it may be impossible to use time-consuming and/or data-intensive methods that involve for example the training of a neural network, or discrete/integer control constraints. A simpler possibility is to use rollout, which we discuss in the next section.

5.2 APPROXIMATION IN VALUE SPACE, ROLLOUT, AND ADAPTIVE CONTROL

We will now consider an approach for dealing with unknown or changing parameters, which is based on rollout and on-line replanning. We have already noted this approach in Chapter 1, where we stressed the importance of fast on-line policy improvement.

Let us assume that some problem parameters change over time, while the controller estimates the changes on-line, perhaps after a suitable delay for data collection. The method by which the problem parameters are recalculated or become known is immaterial for the purposes of the following discussion. It may involve a limited form of parameter estimation, whereby the unknown parameters are "tracked" by data collection over a few time stages, with due attention paid to issues of parameter identifiability; or it may involve new features of the control environment, such as a changing number of servers and/or tasks in a service system.

We thus assume away/ignore the detailed issues of parameter estimation, and focus on revising the controller by on-line replanning based on the newly obtained parameters. This revision may be based on any suboptimal method, but rollout with some base policy is particularly attractive. The base policy may be either a fixed robust controller (such as some form of PID control) or it may be updated over time (in the background, on the basis of some unspecified rationale), in which case the rollout policy will be revised both in response to the changed base policy and in response to the changing parameters.

Here the advantage of rollout is that it is simple, reliable, and relatively fast. In particular, it does not require a complicated training procedure, based for example on the use of neural networks or other approximation architectures, so *no new policy is explicitly computed in response to the parameter changes*. Instead the available controls at the current state are compared through a one-step or multistep minimization, with cost function approximation provided by the base policy (cf. Fig. 5.2.1).

Another issue to consider is the stability and robustness properties of the rollout policy. In this connection, it can be generally proved, under mild conditions, that *if the base policy is stable within a range of parameter values, the same is true for the rollout policy*; this can also be inferred from Fig. 3.4.3. Related ideas have a long history in the control theory literature; see Beard [Bea95], Beard, Saridis, and Wen [BSW99], Jiang and Jiang [JiJ17], Kalise, Kundu, Kunisch [KKK20], Pang and Jiang [PaJ21].

The principal requirement for using rollout in an adaptive control context is that the rollout control computation should be fast enough to be performed between stages. In this connection, we note that accelerated/truncated or simplified versions of rollout, as well as parallel computation, can be used to meet this time constraint.

Generally, adaptive control by rollout and on-line replanning makes sense in situations where the calculation of the rollout controls for a given set of problem parameters is faster and/or more convenient than the calculation of the optimal controls for the same set of parameter values. These problems include cases involving nonlinear systems and/or difficult (e.g., integer) constraints.

The following example illustrates on-line replanning with the use of rollout in the context of the simple one-dimensional linear quadratic problem that we discussed earlier. The purpose of the example is to show analytically how rollout with a base policy that is optimal for a nominal set of problem parameters works well when the parameters change from their nominal values. This property is not practically useful in linear quadratic problems because when the parameter change, it is possible to calculate the new optimal policy in closed form, but it is indicative of the performance robustness of rollout in other contexts; for example linear quadratic problems with constraints.

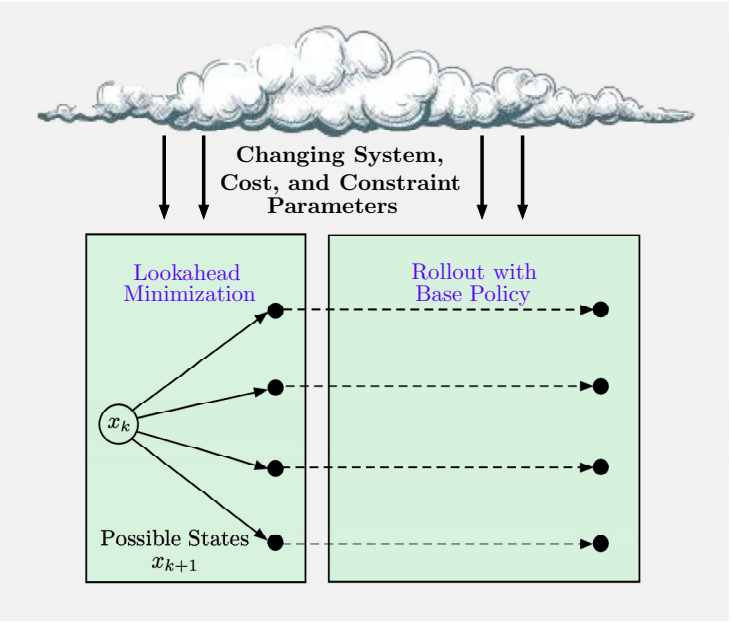

Figure 5.2.1 Schematic illustration of adaptive control by on-line replanning based on rollout. One-step lookahead minimization is followed by simulation with the base policy, which stays fixed. The system, cost, and constraint parameters are changing over time, and the most recent estimates of their values are incorporated into the lookahead minimization and rollout operations. Truncated rollout with multistep lookahead minimization and terminal cost approximation is also possible. The base policy may also be revised based on various criteria. For the discussion of this section, we may assume that all the changing parameter information is provided by some computation and sensor "cloud" that is beyond our control.

Example 5.2.1 (On-Line Replanning for Linear Quadratic Problems Based on Rollout)

Consider a deterministic undiscounted infinite horizon linear quadratic problem involving the linear system

$$x_{k+1} = x_k + bu_k,$$

and the quadratic cost function

$$\lim_{N\to\infty} \sum_{k=0}^{N-1} (x_k^2 + ru_k^2).$$

This is the one-dimensional problem of the preceding section for the special case where $a = 1$ and $q = 1$. The optimal cost function is given by

$$J^*(x) = K^* x^2,$$

where K^* is the unique positive solution of the Riccati equation

$$K = \frac{rK}{r + b^2K} + 1. \tag{5.1}$$

The optimal policy has the form

$$\mu^*(x) = L^*x, \tag{5.2}$$

where

$$L^* = -\frac{bK^*}{r + b^2K^*}. \tag{5.3}$$

As an example, consider the optimal policy that corresponds to the nominal problem parameters $b = 2$ and $r = 0.5$: this is the policy (5.2)-(5.3), with K computed as the positive solution of the quadratic Riccati Eq. (5.1) for $b = 2$ and $r = 0.5$. For these nominal parameter values, we have

$$K^* = \frac{2 + \sqrt{6}}{4} \approx 1.11.$$

From Eq. (5.3) we then also obtain

$$L^* = -\frac{2 + \sqrt{6}}{5 + 2\sqrt{6}}. \tag{5.4}$$

We will now consider changes of the values of b and r while keeping L constant to the preceding value, and we will compare the quadratic cost coefficients of the following three cost functions as b and r vary:

(a) The optimal cost function K^*x^2, where K^* is given by the positive solution of the Riccati Eq. (5.1).

(b) The cost function K_Lx^2 that corresponds to the base policy

$$\mu_L(x) = Lx,$$

where L is given by Eq. (5.4). Here, we have (cf. Section 4.1)

$$K_L = \frac{1 + rL^2}{1 - (1 + bL)^2}. \tag{5.5}$$

(c) The cost function $\tilde{K}_Lx^2$ that corresponds to the rollout policy

$$\tilde{\mu}_L(x) = \tilde{L}x,$$

obtained by using the policy μ_L as base policy. Using the formulas derived earlier, we have [cf. Eq. (5.5)]

$$\tilde{L} = -\frac{bK_L}{r + b^2K_L},$$

and (cf. Section 4.1)

$$\tilde{K}_L = \frac{1 + r\tilde{L}^2}{1 - (1 + b\tilde{L})^2}.$$

Figure 5.2.2 shows the coefficients K^*, K_L, and $\tilde{K}_L$ for a range of values of r and b. We have

$$K^* \leq \tilde{K}_L \leq K_L.$$

The difference $K_L - K^*$ is indicative of the robustness of the policy μ_L, i.e., the performance loss incurred by ignoring the changes in the values of b and r, and continuing to use the policy μ_L, which is optimal for the nominal values $b = 2$ and $r = 0.5$, but suboptimal for other values of b and r. The difference $\tilde{K}_L - K^*$ is indicative of the performance loss due to using online replanning by rollout rather than using optimal replanning. Finally, the difference $K_L - \tilde{K}_L$ is indicative of the performance improvement due to online replanning using rollout rather than keeping the policy μ_L unchanged.

Note that Fig. 5.2.2 illustrates the behavior of the error ratio

$$\frac{\tilde{J} - J^*}{J - J^*},$$

where for a given initial state, $\tilde{J}$ is the rollout performance, J^* is the optimal performance, and J is the base policy performance. This ratio approaches 0 as $J - J^*$ becomes smaller because of the superlinear/quadratic convergence rate of Newton's method that underlies the rollout algorithm.

5.3 APPROXIMATION IN VALUE SPACE, ROLLOUT, AND MODEL PREDICTIVE CONTROL

In this section, we briefly discuss the MPC methodology, with a view towards its connection with approximation in value space and the rollout algorithm. We will focus on the undiscounted infinite horizon deterministic problem, which involves the system

$$x_{k+1} = f(x_k, u_k),$$

whose state x_k and control u_k are finite-dimensional vectors. The cost per stage is assumed nonnegative

$$g(x_k, u_k) \geq 0, \qquad \text{for all } (x_k, u_k),$$

(e.g., a positive definite quadratic cost). There are control constraints $u_k \in U(x_k)$, and to simplify the following discussion, we will initially consider no state constraints. We assume that the system can be kept at the origin at zero cost, i.e.,

$$f(0, \overline{u}_k) = 0, \quad g(0, \overline{u}_k) = 0 \qquad \text{for some control } \overline{u}_k \in U(0).$$

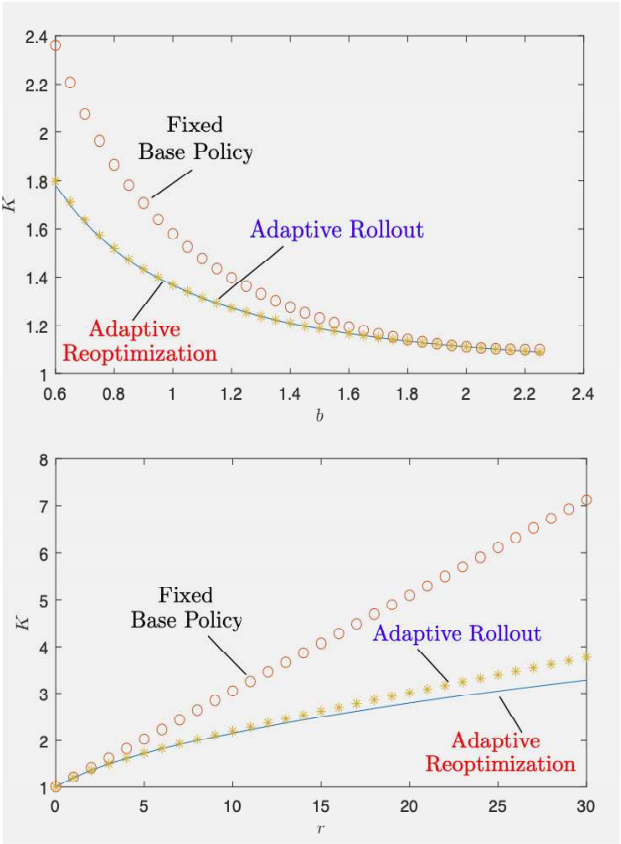

Figure 5.2.2 Illustration of control by rollout under changing problem parameters. The quadratic cost coefficients K^* (optimal, denoted by solid line), K_L (base policy, denoted by circles), and $\tilde{K}_L$ (rollout policy, denoted by asterisks) are shown for the two cases where $r = 0.5$ and b varies, and $b = 2$ and r varies. The value of L is fixed at the value that is optimal for $b = 2$ and $r = 0.5$ [cf. Eq. (5.4)]. The rollout policy performance is very close to optimal, even when the base policy is far from optimal.

Note that, as the figure illustrates, we have

$$\lim_{J \to J^*} \frac{\tilde{J} - J^*}{J - J^*} = 0,$$

where for a given initial state, $\tilde{J}$ is the rollout performance, J^* is the optimal performance, and J is the base policy performance. This is a consequence of the superlinear/quadratic convergence rate of Newton's method that underlies rollout, and guarantees that the rollout performance approaches the optimal much faster than the base policy performance does.

For a given initial state x_0, we want to obtain a sequence $\{u_0, u_1, \ldots\}$ that satisfies the control constraints, while minimizing the total cost.

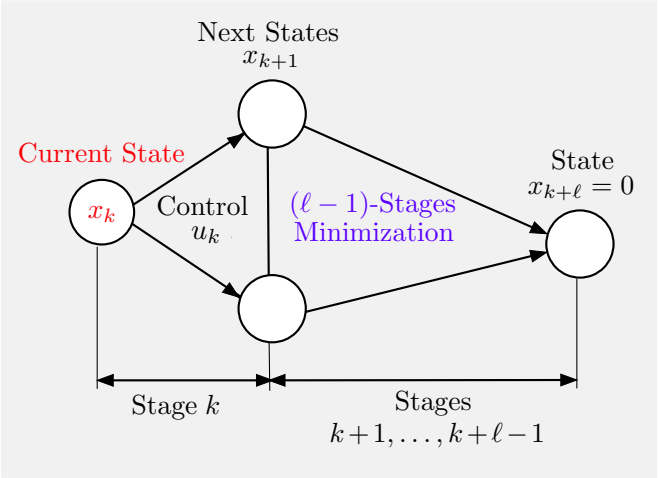

Figure 5.3.1 Illustration of the problem solved by a classical form of MPC at state x_k. We minimize the cost function over the next ℓ stages while imposing the requirement that $x_{k+\ell}=0$. We then apply the first control of the optimizing sequence. In the context of rollout, the minimization over u_k is the one-step lookahead, while the minimization over $u_{k+1},\ldots,u_{k+\ell-1}$ that drives $x_{k+\ell}$ to 0 is the base heuristic.

This is a classical problem in control system design, known as the *regulation problem*, where the aim is to keep the state of the system near the origin (or more generally some desired set point), in the face of disturbances and/or parameter changes. In an important variant of the problem, there are additional state constraints of the form $x_k \in X$, and there arises the issue of maintaining the state within X, not just at the present time but also in future times. We will address this issue later in this section.

The Classical Form of MPC - View as a Rollout Algorithm

We will first focus on a classical form of the MPC algorithm, proposed in the form given here by Keerthi and Gilbert [KeG88]. In this algorithm, at each encountered state x_k, we apply a control $\tilde{u}_k$ that is computed as follows; see Fig. 5.3.1:

(a) We solve an ℓ-stage optimal control problem involving the same cost function and the requirement that the state after ℓ steps is driven to 0, i.e., $x_{k+\ell}=0$. This is the problem

$$\min_{u_t,\, t=k,\ldots,k+\ell-1} \sum_{t=k}^{k+\ell-1} g(x_t,u_t), \tag{5.6}$$

subject to the system equation constraints

$$x_{t+1}=f(x_t,u_t), \qquad t=k,\ldots,k+\ell-1, \tag{5.7}$$

the control constraints

$$u_t \in U(x_t), \qquad t = k, \ldots, k+\ell-1, \tag{5.8}$$

and the terminal state constraint

$$x_{k+\ell} = 0. \tag{5.9}$$

Here ℓ is an integer with $\ell > 1$, which is chosen in some largely empirical way.

(b) If $\{\tilde{u}_k, \ldots, \tilde{u}_{k+\ell-1}\}$ is the optimal control sequence of this problem, we apply $\tilde{u}_k$ and we discard the other controls $\tilde{u}_{k+1}, \ldots, \tilde{u}_{k+\ell-1}$.

(c) At the next stage, we repeat this process, once the next state x_{k+1} is revealed.

To make the connection of the preceding MPC algorithm with rollout, we note that *the one-step lookahead function $\tilde{J}$ implicitly used by MPC [cf. Eq. (5.6)] is the cost function of a certain stable base policy*. This is the policy that drives to 0 the state after $\ell - 1$ stages (*not ℓ stages*) and keeps the state at 0 thereafter, while observing the state and control constraints, and minimizing the associated $(\ell-1)$-stages cost. This rollout view of MPC was first discussed in the author's paper [Ber05]. It is useful for making a connection with the approximate DP/RL, rollout, and its interpretation in terms of Newton's method. In particular, an important consequence is that *the MPC policy is stable*, since rollout with a stable base policy yields a stable policy, as we have discussed in Section 3.2.

We may also equivalently view the preceding MPC algorithm as rollout with $\bar{\ell}$-step lookahead, where $1 < \bar{\ell} < \ell$, with the base policy that drives to 0 the state after $\ell - \bar{\ell}$ stages and keeps the state at 0 thereafter. This suggests variations of MPC that involve truncated rollout with terminal cost function approximation, which we will discuss shortly.

Note also that when faced with changing problem parameters, it is natural to consider on-line replanning as per our earlier discussion. In particular, once new estimates of system and/or cost function parameters become available, MPC can adapt accordingly by introducing the new parameter estimates into the ℓ-stage optimization problem in (a) above.

5.4 TERMINAL COST APPROXIMATION - STABILITY ISSUES

In a common variant of MPC, the requirement of driving the system state to 0 in ℓ steps in the ℓ-stage MPC problem (5.6), is replaced by a nonnegative terminal cost $G(x_{k+\ell})$. Thus at state x_k, we solve the problem

$$\min_{u_t,\, t=k,\ldots,k+\ell-1} \left[G(x_{k+\ell}) + \sum_{t=k}^{k+\ell-1} g(x_t, u_t) \right], \tag{5.10}$$

instead of problem (5.6) where we require that $x_{k+\ell} = 0$. This variant can be viewed as rollout with one-step lookahead, and a base policy, which at state x_{k+1} applies the first control $\tilde{u}_{k+1}$ of the sequence $\{\tilde{u}_{k+1}, \ldots, \tilde{u}_{k+\ell-1}\}$ that minimizes

$$G(x_{k+\ell}) + \sum_{t=k+1}^{k+\ell-1} g(x_t, u_t).$$

It can also be viewed outside the context of rollout, as approximation in value space with ℓ-step lookahead minimization and terminal cost approximation given by G. Thus the preceding MPC controller may have cost function that is much closer to J^* than G is. This is due to the superlinear/quadratic convergence rate of Newton's method that underlies approximation in value space, as we have discussed in Chapter 3.

An important question is to choose the terminal cost approximation so that the resulting MPC controller is stable. Our discussion of Section 3.3 on the region of stability of approximation in value space schemes applies here. In particular, under the nonnegative cost assumption of this section, the MPC controller will be stable if $TG \leq G$ (using the abstract DP notation introduced in Chapter 3), or equivalently

$$(TG)(x) = \min_{u \in U(x)} \Big\{ g(x,u) + G\big(f(x,u)\big) \Big\} \leq G(x), \qquad \text{for all } x, \quad (5.11)$$

as noted in Section 3.2. This condition is sufficient for stability of the MPC controller, but it is not necessary. Figure 5.4.1 provides a graphical illustration. It shows that the condition $TG \leq G$ implies that $J^* \leq T^\ell G \leq T^{\ell-1} G$ for all $\ell \geq 1$ (the books [Ber12] and [Ber18a] provide mathematical proofs of this fact). This in turn implies that $T^\ell G$ lies within the region of the stability for all $\ell \geq 0$.

We also expect that as the length ℓ of the lookahead minimization is increased, the stability properties of the MPC controller are improved. In particular, *given $G \geq 0$, the resulting MPC controller is likely to be stable for ℓ sufficiently large*, since $T^\ell G$ ordinarily converges to J^*, which lies within the region of stability. Results of this type are known within the MPC framework under various conditions (see the papers by Mayne at al. [MRR00], Magni et al. [MDM01], the MPC book [RMD17], and the author's book [Ber20a], Section 3.1.2). Our discussion of stability in Sections 4.4 and 4.6 is also relevant within this context.

In another variant of MPC, in addition to the terminal cost function approximation G, we use truncated rollout, which involves running some stable base policy μ for a number of steps m; see Fig. 5.4.2. This is quite similar to standard truncated rollout, except that the computational solution of the lookahead minimization problem (5.10) may become complicated when the control space is infinite. As discussed in Section 3.3, *increasing the length of the truncated rollout enlarges the region of stability of the MPC controller*. The reason is that by increasing the length of the

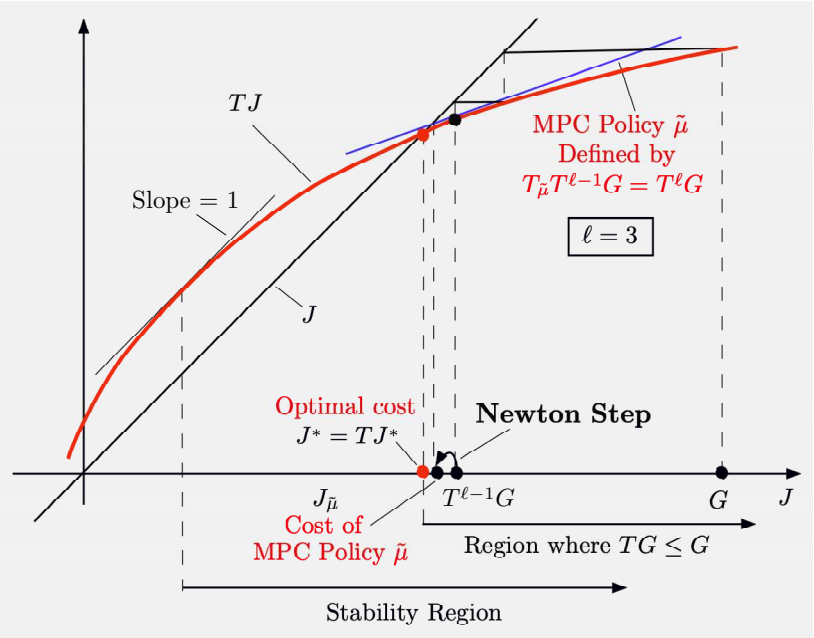

Figure 5.4.1 Illustration of the condition $TG \leq G$ or equivalently

$$(TG)(x) = \min_{u \in U(x)} \Big\{ g(x,u) + G\big(f(x,u)\big) \Big\} \leq G(x), \qquad \text{for all } x.$$

When satisfied by the terminal cost function approximation G, it guarantees the stability of the MPC policy $\tilde{\mu}$ with ℓ-step lookahead minimization, defined by

$$T_{\tilde{\mu}} T^{\ell-1} G = T^{\ell} G,$$

where for a generic policy μ, T_μ is defined (using the abstract DP notation of Chapter 3) by

$$(T_\mu J)(x) = g\big(x, \mu(x)\big) + J\Big(f\big(x, \mu(x)\big)\Big), \qquad \text{for all } x.$$

In this figure, $\ell = 3$.

truncated rollout, we push the start of the Newton step towards of the cost function J_μ of the stable policy, which lies within the region of stability since $TJ_\mu \leq T_\mu J_\mu = J_\mu$; see also the discussion on linear quadratic problems in Section 4.7. The base policy may also be used to address state constraints; see the papers by Rosolia and Borelli [RoB17], [RoB19], and the discussion in the author's RL book [Ber20a].

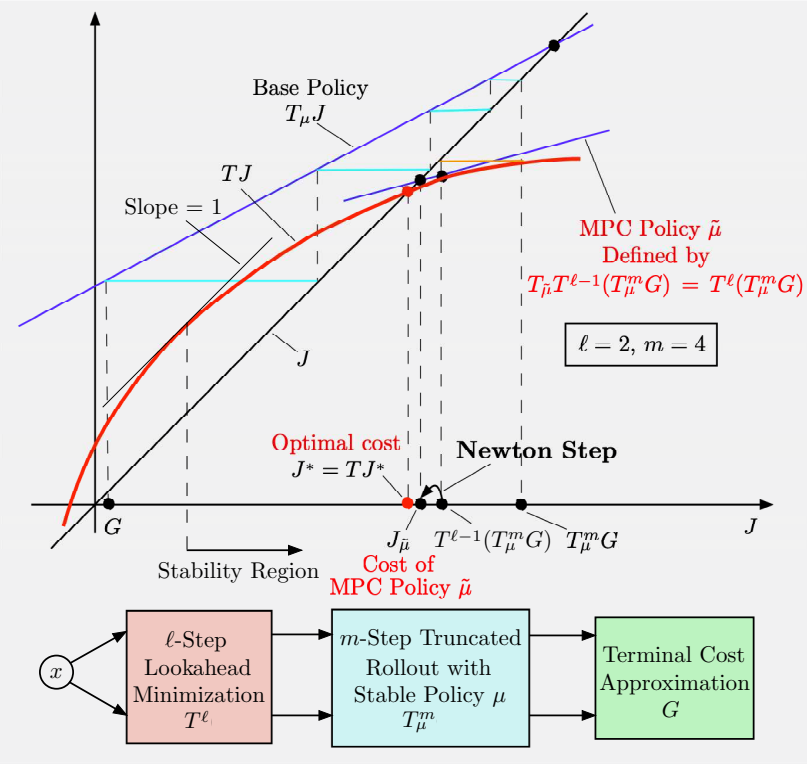

Figure 5.4.2 An MPC scheme with ℓ-step lookahead minimization, m-step truncated rollout with a stable base policy μ, and a terminal cost function approximation G, together with its interpretation as a Newton step. In this figure, $\ell = 2$ and $m = 4$. As m increases, $T_\mu^m G$ moves closer to J_μ, which lies within the region of stability.

A Rollout Variant of MPC with Multiple Terminal States and Base Policies

In another variation of MPC, proposed in the paper by Li et al. [LJM21], instead of driving the state to 0 at the end of ℓ steps, we consider multiple terminal system states at the end of the ℓ-step horizon, as well as the use of multiple base policies for rollout. In particular, in this scheme we have a finite set of states $\mathcal{X}$ and a finite set of stable base policies $\mathcal{M}$, and we assume that we have computed off-line the cost function values $J_\mu(x)$ for all $x \in \mathcal{X}$ and $\mu \in \mathcal{M}$. At state x_k, to compute the MPC control $\tilde{u}_k$, we solve for each $x \in \mathcal{X}$ a problem that is the same as the problem (5.6)-(5.9), which is solved by the classical form of MPC, except that the terminal state $x_{k+\ell}$ is equal to x instead of $x_{k+\ell} = 0$. This is the problem

$$\min_{u_t,\, t=k,\ldots,k+\ell-1} \sum_{t=k}^{k+\ell-1} g(x_t, u_t), \tag{5.12}$$

subject to the system equation constraints

$$x_{t+1} = f(x_t, u_t), \qquad t = k, \ldots, k+\ell-1, \tag{5.13}$$

the control constraints

$$u_t \in U(x_t), \qquad t = k, \ldots, k+\ell-1, \tag{5.14}$$

and the terminal state constraint

$$x_{k+\ell} = x. \tag{5.15}$$

Let $V(x_k; x)$ be the optimal value of this problem. Having computed $V(x_k; x)$ for all $x \in \mathcal{X}$, we compare all values

$$V(x_k; x) + J_\mu(x), \qquad x \in \mathcal{X},\ \mu \in \mathcal{M},$$

and find the pair $(\overline{x}, \overline{\mu})$ that yields the minimal value of $V(x_k; x) + J_\mu(x)$. We then define the MPC control $\tilde{u}_k$ to be the control u_k that attains the minimum in the corresponding problem (5.12)-(5.15) with $x = \overline{x}$.

Thus, in this variant of MPC we solve multiple problems of the type that is solved in the classical form of MPC, for multiple values of the terminal state $x_{k+\ell}$, and we then compute the MPC control based on the "best" terminal state $x \in \mathcal{X}$, assuming that the "best" base policy $\overline{\mu}$ will be used after state $k+\ell$. It is possible to show, under appropriate conditions,† that the cost function $J_{\tilde{\mu}}$ of the MPC policy $\tilde{\mu}$, which applies $\tilde{\mu}(x_k) = \tilde{u}_k$ as described above, has the cost improvement property

$$J_{\tilde{\mu}}(x) \leq J_\mu(x), \qquad \text{for all } x \in \mathcal{X},\ \mu \in \mathcal{M}; \tag{5.16}$$

see [LJM21]. Moreover, based on this property and the assumption that the base policies $\mu \in \mathcal{M}$ are stable, it follows that the MPC policy $\tilde{\mu}$ thus obtained is also stable.

The preceding variation can also be used for systems with arbitrary state and control spaces, continuous as well as discrete. It is also well-suited for addressing state constraints, provided the base policies are designed to satisfy these constraints. In this case, the state constraints are included in the constraints of the ℓ-step problems (5.12)-(5.15). We refer to the paper [LJM21], which provides supporting analysis, extensions to the case where X is an infinite set, as well as computational results involving several types of problems, with both discrete and continuous state and control spaces.

We mention that the idea of using multiple base policies to evaluate the available controls at a given state, and selecting the control that yields the least cost, has been known since the original proposal of the paper [BTW97]. The main result for such schemes is a cost improvement property, whereby the rollout policy outperforms simultaneously all the base policies; cf. Eq. (5.16). This property is also discussed in Sections 6.3 and 6.4, as well as the books [Ber17a], [Ber19a], [Ber20a].

† These conditions include that for every $x \in \mathcal{X}$, we have $f\big(x, \mu(x)\big) \in \mathcal{X}$ for some $\mu \in \mathcal{X}$, which plays the same role as the assumption that the origin is cost free and absorbing in the classical form of MPC.

Stochastic MPC by Certainty Equivalence

Let us mention that while in this section we have focused on deterministic problems, there are variants of MPC, which include the treatment of uncertainty. The books and papers cited earlier contain several ideas along these lines; see for example the books by Kouvaritakis and Cannon [KoC16], Rawlings, Mayne, and Diehl [RMD17], and the survey by Mesbah [Mes16].

In this connection it is also worth mentioning the *certainty equivalence approach* that we discussed briefly in Section 3.2. As noted in that section, upon reaching state x_k we may perform the MPC calculations after replacing the uncertain quantities $w_{k+1}, w_{k+2}, \ldots$ with deterministic quantities $\overline{w}_{k+1}, \overline{w}_{k+2}, \ldots$, *while allowing for the stochastic character of the disturbance* w_k *of just the current stage* k. This MPC calculation is not much more difficult that the one for deterministic problems, while still implementing a Newton step for solving the associated Bellman equation; see the discussion of Section 3.2, and also Section 2.5.3 of the RL book [Ber19a].

State Constraints, Target Tubes, and Off-Line Training

Our discussion so far has skirted a major issue in MPC, which is that there may be additional state constraints of the form $x_k \in X$, for all k, where X is some subset of the true state space. Indeed much of the original work on MPC was motivated by control problems with state constraints, imposed by the physics of the problem, which could not be handled effectively with the nice unconstrained framework of the linear quadratic problem that we have discussed in Chapter 4.

The treatment of state constraints is connected to the theory of reachability of target tubes, first formulated and studied by the author in his Ph.D. thesis [Ber71], and subsequent papers [BeR71], [Ber72]; see the books [Ber17a], [Ber19a], [Ber20a] for a discussion that is consistent with the viewpoint of this section. A target tube is a subset $\tilde{X}$ of the state constraint set X, within which the state can be kept indefinitely with feasible control choices, assuming that the initial state belongs to $\tilde{X}$. In other words, the problem (5.10) may not be feasible for every $x_k \in X$, once the constraint $x_t \in X$ for all $t = k+1, k+2, \ldots$, is added in problem (5.10). However, a suitable target tube is one specified by a subset $\tilde{X} \subset X$ such that the problem (5.10) is feasible under the constraint $x_t \in \tilde{X}$ for all $t = k+1, k+2, \ldots$, provided $x_k \in \tilde{X}$.

There are several ways to compute sets $\tilde{X}$ with this property, for which we refer to the aforementioned author's work and the MPC literature; see e.g., the book by Rawlings, Mayne, and Diehl [RMD17], and the survey by Mayne [May14]. The important point here is that the computation of a target tube must be done off-line with one of several available algorithmic

approaches, so it becomes part of the off-line training (in addition to the terminal cost function G).

Given an off-line training process, which provides a target tube constraint $x_k \in \tilde{X}$ for all k, a terminal cost function G, and possibly one or more base policies for truncated rollout, MPC becomes an on-line play algorithm for which our earlier discussion applies. Note, however, that in an indirect adaptive control context, where a model is estimated on-line as it is changing, it may be difficult to recompute on-line a target tube that can be used to enforce the state constraints of the problem, particularly if the states constraints change themselves as part of the changing problem data. This is a problem-dependent issue that deserves further attention.

5.5 NOTES AND SOURCES

The literature for PID control is extensive and includes the books by Aström and Hagglund [AsH95], [AsH06]. For detailed accounts of adaptive control, we refer to the books by Aström and Wittenmark [AsW08], Bodson [Bod20], Goodwin and Sin [GoS84], Ioannou and Sun [IoS96], Jiang and Jiang [JiJ17], Krstic, Kanellakopoulos, and Kokotovic [KKK95], Kokotovic [Kok91], Kumar and Varaiya [KuV86], Liu, et al. [LWW17], Lavretsky and Wise [LaW13], Narendra and Annaswamy [NaA12], Sastry and Bodson [SaB11], Slotine and Li [SlL91], and Vrabie, Vamvoudakis, and Lewis [VVL13].

The literature on MPC is voluminous, and has grown over time to include problem and algorithm variations and extensions. For detailed accounts, we refer to the textbooks by Maciejowski [Mac02], Goodwin, Seron, and De Dona [GSD06], Camacho and Bordons [CaB07], Kouvaritakis and Cannon [KoC16], Borrelli, Bemporad, and Morari [BBM17], and Rawlings, Mayne, and Diehl [RMD17].

Deterministic optimal control with infinite state and control spaces can exhibit unusual/pathological behavior. For the case of nonnegative cost per stage, an analysis of the exact value and policy iteration algorithms, including convergence issues and counterexamples, is given in the author's paper [Ber17b] and abstract DP book [Ber22a]. The case of nonpositive cost per stage has been addressed in classical analyses, beginning with the work of Blackwell [Bla65]; see also [Str66], [BeS78], [YuB15].