CRICOS 00213J

ENN586: Introduction to nonlinear control.[†]

Jason J. Ford[‡]

VERSION: 0.2. April 2023.

Abstract

These notes provide a first look at some nonlinear control concepts. These are sequence decision problems where the aim is to determine a sequence of actions for a dynamic system (our typical context is robotic systems such as aerial, maritime or ground robotic platforms moving in the physical world). This is typically in the context of sequential determining actions on the basis of sequentially arriving measurements or observations information.

This material for these weeks of ENN586:

- Beyond the nominal control design
- A side journey: control design as optimisation and MPC
- Adaptive control
- Lyapunov and stability

1 Beyond the nominal control design

Feedback control has a long, largely hidden, history. "Nature discovered feedback long ago. It created feedback mechanisms and exploited them at all levels, which are central to homeostasis and life. As a

[†]©Queensland University of Technology, 2023. ENN586: Decision and Control. School of Electrical Engineering and Robotics, Queensland University of Technology.

[‡]Please report errors within this document to j2.ford@qut.edu.au

technology, control dates back at least two millennia." [2]. However, a golden age for control occurred in the post-World War 2 period and saw the progressive development of many foundational control concepts such as [1, 2]: stability, feedback, model-based design, optimal control, dynamic programming, frequency domain control design, state-space model control design, dynamic games, Kalman filtering, LQG, stochastic systems, adaptive control, and robust control.

Since the 2000s, key technology drivers for control system engineering have included [2]:

- Advances in computing and communication networks,
- the intersection of computing and control disciplines,
- the desire to achieve increasing levels of autonomy, and
- the increasing complexity of systems under control.

In the context of these technology drivers, control is a systems discipline aiming to create **reliably autonomy exhibiting increasingly sophisticated behaviour in a complex world that may be changing and imprecisely known**.

Since 2000, much of the autonomy developments have been empowered by increasing computation capabilities which have allowed increasingly difficult problems to be approached using powerful numeric techniques. These powerful **numeric techniques** have emerged under a variety of names such as numeric dynamic programming, adaptive dynamic programming, neurodynamic programming, Markov chain approximation approaches, partially observable Markov decision process (POMDP), Q-learning, reinforcement learning, and many more. These techniques share the core belief that complex situations can be effectively approximated by model representations with sufficiently high dimensions; representations whose computational feasibility has improved with the availability of more powerful computational resources. Importantly, application of these numerically powered autonomy techniques **also benefit from the rich foundational concepts developed during control's golden age.** For these reasons, future autonomy advancements will likely occur at the intersection between control system engineering, computer science, robotics, and (now) machine and deep learning.

1.1 Motivations to move beyond the nominal design

We first approach control system design assuming that the system is well known (that is, we have a nominal model that is correct and known with certainty). We then conducted a nominal control design on this basis. Unfortunately, there might exist a range of effects that might not be included in the nominal model or design. Examples of effects potentially not considered in the nominal design include:

- Sensor errors such as random noises, biases, etc.
- Un-modelled effects that impact the system's dynamics, such as wind, pressure, friction, etc.
- Un-modelled features in either the world or the system's dynamics.

It will be useful to group these effects into [8, Ch. 1]

- disturbances which have a direct impact on the inputs, outputs or internal variables of a control system, (such as sensor noises, wind effects) and
- structural disturbances which have an impact on interdependent relationships between inputs, output or internal variables of a control system (such as errors in the dynamic model used).

There are three lenses often given to motivate moving beyond the nominal design:

- Desire for **disturbance rejection**.
- Desire to **adapt to changes** in the nominal model.
- Desire to manage **model uncertainty** in the nominal model.

1.2 Disturbance rejection

Disturbance signals represent unwanted system inputs that influence the control system's response. Unmeasured external disturbances, plant model uncertainty and unmeasured plant model variations all degrade the performance of control systems.

A key objective in control system design is to include an ability to reject or minimise the impact of disturbances on a system's ability to achieve its desired purpose. A control system's actions to minimise the effect of disturbances are called **disturbance rejection**.

Some disturbances can be handled by estimating them and then directly cancelling their impact (e.g. a bias in a sensor measurement might be corrected by determining the bias amount and then correcting the measurement by the bias amount). However, many disturbances cannot be handled this way (because they cannot be measured) and need to be managed by smart design of the control loop architecture to minimise the impact of such disturbances (that is, reduce the system's sensitivity to disturbances).

Three common approaches for disturbance rejection are: feedback control, feedforward control (with feedback trim) and cascade (or nested) control.

1.2.1 Feedback control

The purpose of the feedback in a control system is to use information about the system's current departure from desired performance, to counter the impact of direct disturbances, and to adjust the system towards the desired performance.

Feedback control offers some inherent disturbance rejection due to its use of the system's current response in the calculation of the control signal. In fact, feedback can be designed to balance closed-loop system performance and the system's sensitivity to disturbances [4, Ch. 8 and 9]. However, there are limitations to how much disturbance rejection can be achieved through simple feedback design. For better disturbance rejection performance we may want to consider more advanced disturbance rejection techniques that become available via smart control loop architecture structure choice [4, Ch. 10].

1.2.2 Feedforward control with feedback trim

If we can measure the disturbance, then we can inject a control signal that cancels its impact. The control signal injected for this purpose is called a **feedforward input**. Conceptually, you "invert" the system output to work out the required feedforward signal to inject at the input. You can then add an additional feedback "trim" term so that resulting dynamic response tracks the trajectory (i.e., reduces any error between the system's response to the calculated feedforward input and desired response).

Transfer function example of feedforward with feedback trim for disturbance rejection: Consider transfer function

$$X(s) = P(s)U(s) + D(s),$$

where $P(s)$ is the system, $U(s)$ is the input you can design, $D(s)$ is the disturbance and $X(s)$ is the output you want to control. Let $U(s) = U_{fb}(s) + U_{ff}(s)$ be our feedforward with feedback trim control where $U_{fb}(s)$, $U_{ff}(s)$ are feedback and feedforward controls, respectively. In a feedforward approach, we design $U_{ff}(s)$ (this is the "invert" step) so that

$$P(s)U_{ff}(s) = -D(s),$$

or as close $-D(s)$ as we are able (noting perfect "inversion" might not be possible). Then

$$\begin{aligned} X(s) &= P(s)U(s) + D(s) \\ &= P(s)U_{fb}(s) + P(s)U_{ff}(s) + D(s) \\ &= P(s)U_{fb}(s) - D(s) + D(s) \\ &= P(s)U_{fb}(s), \end{aligned}$$

and the disturbance is gone. We are then free to design $U_{fb}(s)$ to achieve out desired response, and note that $U_{fb}(s)$ might be helpful in handling any imperfections in the "invert" step.

□

Unfortunately, feedforward control requires a measurement of the disturbance and hence is not possible in situations where it is infeasible to measure the disturbance, e.g. wind disturbances. Additionally, the "invert" step can be sensitive to both measurement and model errors.

Another common use of feedforward control is to achieve precise **reference tracking** of a time-varying signal that you require the control system's response to follow (in this situation, **you feedforward the control signal required to track a known reference, rather than to reject a measured disturbance**). This is particularly useful for vehicle systems (which have limited control authority). For example, to achieve a car turn manoeuvre, you can feedforward the main part of the turn command, and then use small feedback corrections about the main signal (to correct slippage etc. errors) so that the car perfectly tracks the turning curve.

State space model example of feedforward with feedback trim for reference tracking: Consider the state space model

$$x(k+1) = Ax(k) + Bu(k)$$

where A, B are the usual state space model matrices, $x(k)$ is the state process that you want to control, and $u(k)$ is the input you can design. Assume that you want your system to track a provided reference signal $r(k)$. Let $u(k) = u_{fb}(k) + u_{ff}(k)$ be our feedforward with feedback trim control where $u_{fb}(k)$, $u_{ff}(k)$ are feedback and feedforward controls, respectively. In a feedforward approach, we design $u_{ff}(k)$ (this is the "invert" step) so that for all k

$$r(k+1) = Ar(k) + Bu_{ff}(k)$$

or tracks as close as we can to $r(k)$ (perfect tracking may not be possible). Let us write $x(k) = r(k) + \delta x(k)$ Then we can write

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ &= Ax(k) + B(u_{fb}(k) + u_{ff}(k)) \\ r(k+1) + \delta x(k+1) &= A(r(k) + \delta x(k)) + B(u_{fb}(k) + u_{ff}(k)) \\ \delta x(k+1) &= A\delta x(k) + Bu_{fb}(k) \end{aligned}$$

where, in the second last step, we have used that $r(k+1) = Ar(k) + Bu_{ff}(k)$ (and can be removed from both sides of the equation). Hence, we can then design a feedback control $u_{fb}(k)$ so that $\delta x(k+1)$ goes to zero and $x(k)$ goes to $r(k)$ (and this type of feedback action is called feedback "trim"). □

The presence of indirect disturbances or model errors may require a control system feedback mechanism to be supplemented with additional layers of design or mechanisms.

1.2.3 Cascade control

Cascade control is a control loop architecture involving multiple nested loops (that is, loops inside other loops where the inner loops are faster in the sense of having smaller settling times). One common situation where cascade control is possible is in systems having rate quantities that are measurable (this is often the situation in physical systems under motion, as a simple by-product of Newton's laws). Vehicle systems are one example, but many other physical systems and non-physical situations are suitable for cascade control architectures.

Cascade control can be understood as a control loop architecture where the **outer loop commands appropriate set points (or target point) as reference signals for the inner loop**. The inner loop quickly controls the system to this set point. Cascade control's ability to reject disturbances comes from the **inner loop's ability to quickly measure and correct any undesirable system response** away from these inner set points, noting that undesirable behaviours that **would only slowly become "apparent" in the outer loop's signal** (i.e., after the undesirable effect slowly building up through integration of the undesirable rate signal).

Pros of cascade control

- Often, the inner design loop can manage all the nonlinearities. That is, all the difficult dynamics and related control design activities occur in one location.
- If able to locate all the challenging nonlinearities in the inner loop, the outer loops are then transferable to other hardware etc. This increases the portability of solution approaches to different platforms (that is, only the inner loops need to be redesigned).
- Often cascade control is a relatively simple approach to achieve practical robust solutions (compared to complex single loop designs).

Cons of cascade control

- Requires measurements of signals in each loop (that is, potentially involves additional sensor costs).
- Involves a sequence of multiple control designs, one for each loop; however, this design sequence is often easier to achieve than a single complex loop design.
- There is a possibility of (undesirable) dynamic coupling between different levels in the nested loops (this might be resolved by slowing the outer loops, at the cost of a slower overall system response).

1.2.4 Summary of disturbance rejection

Beyond the basic disturbance-rejected capability of simple feedback control, the control loop architecture structure is a design choice in itself. Disturbance rejection via control loop architecture design is important in autonomous systems (disturbances are very common and vehicle systems are typically under-actuated). Cascade control and feedforward control structures are two common approaches for disturbance rejection. In autonomous systems, we often have position control as our objective, but control the vehicle via force quantities (i.e., two derivatives between them), and hence we have the possibility of inner and outer loops in a cascade control architecture. Further, we often have time-varying references to track (e.g., want to follow a curve on a road), and hence use feedforward control (with feedback trim).

Hence, cascade, feedforward and feedback control components are attractive for use in autonomous vehicle systems. We will next learn about some of the specific control structures suitable for vehicle systems.

1.3 Adapt to changes

Control has its roots in frequency domain models of the system to be controlled [11, Ch 1]. The limitations of frequency descriptions led to nominal control designs assuming a time-invariant environment. However, we would often like the control systems we build to operate in environments or situations which might vary over time. If the variations are significant enough it might not be possible to find a frequency domain model-based controller that works across the range of variations.

The challenges are not limited to control designs arising from frequency domain models.

In other model-based paradigms, the same challenge arises whenever there is a desire for a control design able to operate in the face of slow but significantly sized within-model class variations. That is, consider designing a model-based controller in the presence of slow parameter variation within their model class. Again, it might not be possible to find a single parameter that leads to a nominal control design that works well across the range of possible parameter variations.

More issues arise if we allow for the possibility of rapid within-model class variations or for out-of-model class variations.

One approach in the situation of a time-varying environment is to attempt to design **a control that adapts to changes** in the environment experienced. This design approach is called adaptive control, which we will say more about later.

1.4 Model Uncertainty

We first approach control system design assuming that the system is well known (that is, we have a nominal model that is correct and known with certainty). We then conducted a nominal control design on this basis, and (largely) ignored the possibility of model error or uncertainty. Unfortunately, a model might be in error in a manner such that a nominal control design (based on an incorrect model of the system) makes the real system behave in a bad way (e.g. to be unstable).

Two paradigms for addressing model uncertainty or model error are adaptive control and robust control.

In an **adaptive control** design approach to handling uncertainty (with similar but with subtly different motivations to the "adapt to changes" of the previous section), we use measurements of the system to estimate or learn a better model, and then conduct control design on the basis of the estimated (learnt) model. Adaptive design approaches become harder if the system characteristics are changing over time, as the estimated model and controller design may need to adapt or learn over time to the changing system characteristics.

In a **robust control** design approach, we seek a control design that achieves reasonable performance over a range of possible models (a range that we believe contains the true system). That is, we accept that we don't have a perfect model of the system, but seek a design that achieves reasonable performance on the unknown true system as long as the system is within some range.

Both adaptive and robust design approaches have their merits; however, **in practice it is safest to allow for the possibility of model error**, and hence all practical designs should give some consideration of their robustness to modelling error.

Robust control is a rich area, but beyond what we will be able to cover in ENN586.

2 A side journey: control design as optimisation and MPC

We will briefly provide a bit more information about the landscape of concepts in decision and control relevant to robotics and AI.

2.1 Non-linear optimal control

In ENN586 we have largely focused on non-linear control from a stability perspective. An alternative perspective is to consider non-linear control from the perspective of an optimisation problem, in which the desired behaviour is encoded in performance criteria that problem a metric of design suitability. For example, we might consider some non-linear dynamics

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

and we seek a control sequence $u_{[1,T]} = \{u_1, u_2, \dots, u_T\}$ that minimised the finite horizon cost

$$J(u_{[1,T]}) = \sum_{k=1}^{T} g(x_k, u_k)$$

where $g(\cdot, \cdot) \geq 0$ is termed the running cost and represents the cost of being at x_k, u_k at time k (an example might be an energy like cost $g(x_k, u_k) = x_k^2 + u_k^2$ when we prefer x_k and u_k be close to zero). We can also incorporate any constraint we might have on u_k and x_k into our optimisation problem, similar to the optimisation problem extensions that we considered in EGH431.

You will encounter the non-linear optimal control problem in ENN582 Reinforcement Learning and Optimal Control, and we will not say more here other than report that there are whole fields of effort to computationally efficiently solve this sort of problem.

2.2 Model Predictive Control

One particularly useful design approach that is sitting somewhere between the above-mentioned optimisation-based approaches and stability-based approaches are a group of approaches termed Model Predictive Control (MPC). (It is also conceptually informative to consider these to be receded horizon approaches).

The basic idea of these approaches is to at each time instant from your current state value solve a finite horizon optimal control problem but only apply the first step of the optimal solution. Then at the next time instant solve a new finite horizon optimal control problem for your new state value, and solve a finite horizon optimal problem on a new horizon shift 1 step ahead.

That is, at time $k \geq 0$, we seek the control sequence $u_{[k,T+k]}$ that minimses the receding horizon cost

$$J(u_{[k,T+k]}) = \sum_{\ell=1}^{T} g(x_{k+\ell}, u_{k+\ell}).$$

In our controller, we applied the first step of the optimal solution u_k^*. We then move forward to time $\bar{k} = k + 1$ and seek the control sequence $u_{[\bar{k},T+\bar{k}]}$ that minimises the receding horizon cost

$$J(u_{[\bar{k},T+\bar{k}]}) = \sum_{\ell=1}^{T} g(x_{\bar{k}+1}, u_{\bar{k}+\ell}),$$

and applying the optimal $u_{\bar{k}}^*$ that results, so on at future time steps, after updating $\bar{k} = \bar{k} + 1$.

In addition to seeming a sensibility idea, and exploiting any of the computational tools we already have for finite horizon optimal control problems, it can be shown under some mild assumptions, which leads to a stabilising control solution.

Importantly, because *we can incorporate constraints on u_k and x_k in MPC we can consider control problems with constraints that are difficult to explicitly consider with the stability-based approaches highlighted in the rest of these ENN586 notes.* A key downside of MPC is the optimisation tools needed within MPC implementations may not scale well with dimensional and nonlinearity.

We leave further implementation aspects of MPC to the interested reader. We will next focus our attention on a brief introduction to adaptive control.

3 Adaptive Control

The system identification ideas presented in the "ENN586: Estimation and decision in a sequential setting" notes will be assumed pre-requisite knowledge for this adaptive control section. Here we introduce the concept of "adaptive control" before discussing some adaptive control systems.

3.1 What is Adaptive control?

The following introduction to adaptive control partially follows the presentation in chapter 1 of [8].

Adaptive control provides techniques for automatically adjusting in real time the parameters of a controller to achieve or maintain the desired level of performance.

The usual approach to control system design is represented as the top 3 elements in Figure 1. We require the following sorts of information:

- The desired performance
- Know the plant model, and
- Have access to a suitable controller design process

This allows us to determine, before implementation controller parameters of our controller. During operation, the reference signal can be modified to change the output, but the controller parameters remain constant and unaffected by the current input u and output y.

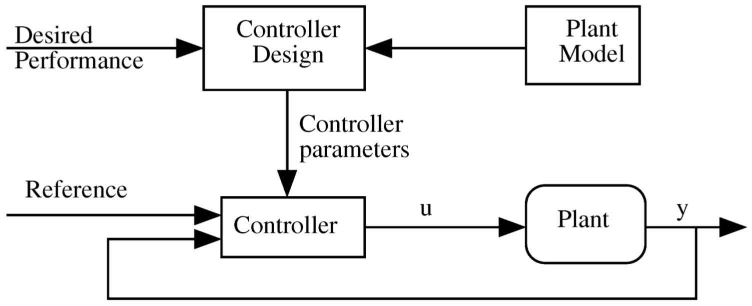

Figure 1: Principle of control design. Image credit: [8].

An adaptive controller, see Figure 2, can be considered a modification of the usual approach to control system design in which there is an adaption scheme in which controller parameters can be adjusted by the current input u and output y.

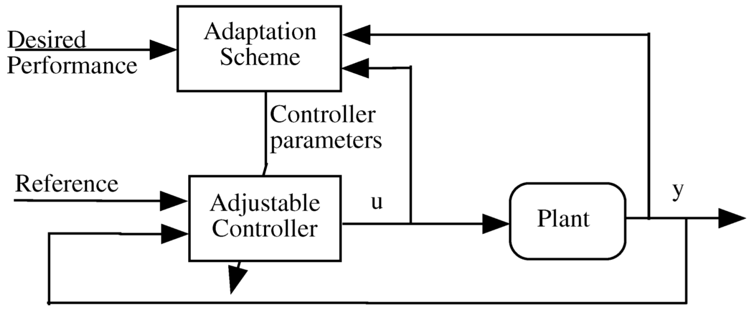

Figure 2: An adaptive control system. Image credit: [8].

The basic configuration or mechanism of adaptive control, see Figure 3, might be a model of the required control information listed above. The adaption scheme might have an internal performance measurement to compare with desired performance and an adaptation mechanism for changing the controller.

Through this lens, adaptive control can be viewed as a hierarchical system with 2 levels:

- Level 1: Conventional feedback control
- Level 2: Adaptation loop.

The *Fundamental Hypothesis in Adaptive Control* is [8]: “For any possible values of plant model parameters, there is a controller with a fixed structure and complexity such that the specified performances can be achieved with appropriate values of the controller parameters”. Typically, the task is to look for a good solution based on some assumed prior knowledge “In other words, an adaptive controller is not a ”black box” which can solve a control problem in real time without an initial knowledge about the plant to be controlled.”

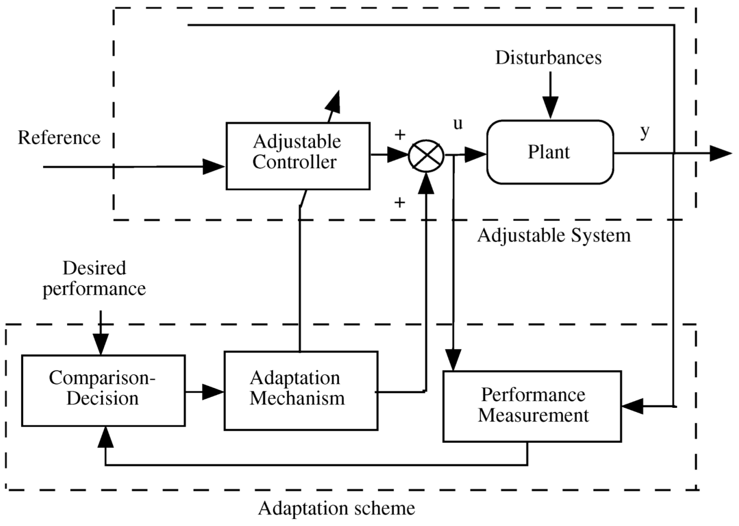

Figure 3: Configuration of an adaptive control system. Image credit: [8].

3.2 Basic Adaptive control schemes

3.2.1 First case: Gain scheduling

As discussed in [10, Section 1.2,6], perhaps the first adaptive control idea might be to design a set of separate controllers and then switch between the designed controller based on a measurement of the environment (also known as open loop control), see Figure 4.

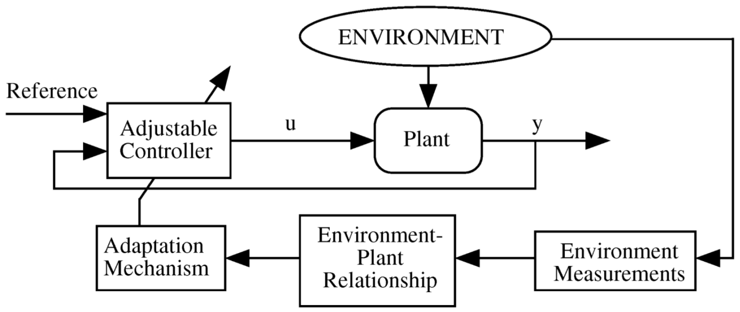

Figure 4: Open loop adaptive control. Image credit: [8].

This is the basic premise of an approach extensively used in aircraft. Let us consider design for N operating points (flight conditions at different altitudes, flight speeds, and air densities expected to be experienced during different stages of flight). We can enumerate them by $i = 1, 2, \ldots, N$ for each of the different (linearised) models of the aircraft dynamics at these different operating points. For each of those operating points (flight conditions), we can do a (linear) design before the flight (parameterised by controller gains θ_i). During flight, we can switch between our pre-designed controller gains based on our measurement of the environment (current flight condition altitude, flight speed, air density).

Gain scheduling has the advantage that the controller gain can quickly be switched, and no control design computation is required during flight. A key disadvantage is that the adjustment mechanism does not have any feedback about the actual flight performance resulting from the gains currently being used (in the sense that there is no feedback from the aircraft state).

3.2.2 Indirect and Direct Adaptive control

There are two adaptation mechanism types corresponding to indirect and direct adaptive control [10].

In indirect adaptive control, see Figure 5, the adaptation mechanism uses information from the sequence of input commands u and output y to estimate a plant model that can be used instead of the true plant model shown at the top 3 elements of Figure 1 (and the adjustable controller is essentially the controller used in the bottom half of Figure 1 with parameters that can change). This process of estimating a plant model is called **identification)**, and we provided details and material about this model identification step in the companion estimation and filtering notes.

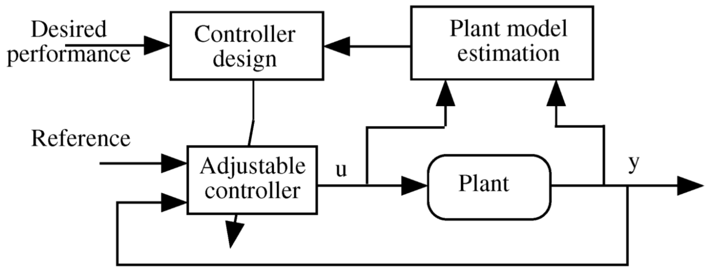

Figure 5: Indirect adaptive control. Image credit: [8].

In direct adaptive control, the adaption mechanism bypasses the process shown at the top 3 elements of Figure 1 and changes the controller directly, without relying on having a plant model. Hence, the adaption step in direct adaptive control is different from the model identification step used in indirect adaptive control and requires analysis via tools such as those we cover in Section 4

3.2.3 Model reference adaptive control

Model reference adaptive control (MRAC) is a common approach to adaptive control. The main structure of MRA is illustrated in Figure 6. The reference model represents the desired trajectory or response of the system, y_M to a reference input. The response tracking error, $y - y_m$, and plant output, y, are used to drive the adaptation mechanism to adjust the controller.

Typically, we select the reference model which is both realistically achievable by the plant, but also a

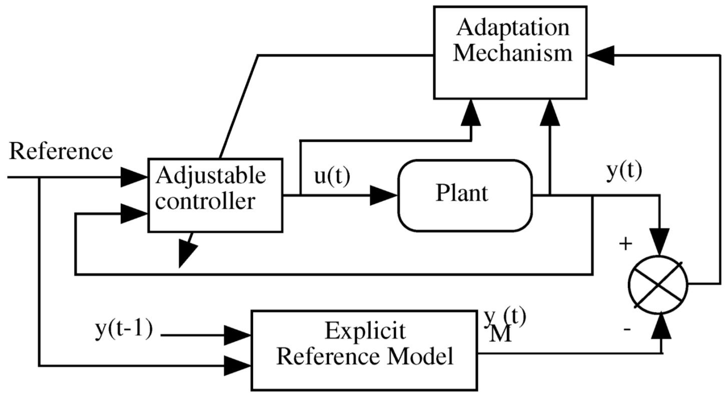

Figure 6: Model Reference Adaptive Control (MRAC). Image credit: [8].

desirable response.

There are direct and indirect versions of MRAC, with the difference being related to whether estimating controller parameter direction or indirectly via plant parameters.

3.2.4 An MRAC example

To illustrate a few of these ideas we now present a simple example of MRAC [10, Ch. 6]:

The unknown plant: Consider a simple first-order linear system with an unknown pole

$$\dot{x} = ax + u, \ x(0) = x_0$$

where $x \in R$ is the state, $a \in R$ is the unknown pole location, $u \in R$ is the control input to be designed.

The model reference model: Consider a simple reference linear system with our desired response

$$\dot{x}_m = -a_m x_m, \ x_m(0) = x_{m0}$$

for some (known) $a_m > 0$. Hence, a_m is select to be desired pole location. If we know a we could use control of the form $u = -k^* x$ where $k^* = a + a_m$ (this would lead to the closed-loop system of $\dot{x} = ax + u = -a_m x$). In terms of the proposed MRAC we will consider k^* to be the (unknown) target value of adaption, which would lead to the model reference behaviour (i.e. pole at a_m) in the closed loop system.

The model reference adaptive controller: Our model reference controller has an update law for gain k of the form:

$$\dot{k} = \gamma(x - x_m)x$$

with $\gamma > 0$ and the adaptive controller

$$u = -k(t)x.$$

Foreshadowing: a stability analysis approach: In a later section of notes, we will introduce the Lyapunov stability analysis technique. The analysis here may not make sense until we cover later material, but by using a Lyapunov stability analysis technique the stability of the above-proposed model reference adaptive control can be established.

Let $e_1 \overset{\text{def}}{=\!=} x - x_m$ and $\widetilde{k} \overset{\text{def}}{=\!=} k - k^*$, as the error in state value and error in gain respectively. We note $\dot{e}_1 = (a-k)x - a_m x_m = -a_m e_1 - \widetilde{k}x$ and $\dot{\widetilde{k}} = \dot{k} = \gamma e_1 x$.

To understand the stability properties of the proposed MRAC we can then defined the proposed candidate Lyapunov function:

$$V \overset{\text{def}}{=\!=} \frac{e_1^2}{2} + \frac{\widetilde{k}^2}{2\gamma}.$$

Then to see that this V is actually a valid Lyapunov function we note $V \geq 0$ and that $\dot{V} \leq 0$ as

$$\dot{V} = e_1\dot{e}_1 + \frac{\widetilde{k}\dot{\widetilde{k}}}{\gamma} = -a_m e_1^2 - e_1\widetilde{k}x + e_1\widetilde{k}x = -a_m e_1^2.$$

where we have selected $a_m > 0$. Hence, this Lyapunov function establishes that the proposed MRAC stability the unknown plant.

Note the Lyapunov function form involves terms quadratic in error quantities. This quadratic form leads to an interpretation of Lyapunov functions as providing a description of energy error which is always decreasing if the controller is stabilising. This quadratic error structure is commonly seen in the stability analysis of linear systems, but will not always appear in analysis problems with non-linear dynamics. Also, note how within the $\dot{V}$ algebraic steps a term arising from the $\widetilde{k}^2$ derivative exactly cancels a term arising from the e_1^2 derivative. We will see later other examples of where a similar cancelling mechanism also appears.

3.3 When does adaptive control work?

The performance of an adaptive control approach depends on both the Level 1 (Conventional feedback control) and Level 2 (Adaptation loop) design aspects. A typical design process might proceed by understanding how to successfully achieved Level 1 design under (unrealistic) assumed knowledge of the plant. Hence, the eventual performance of the adaptive control becomes dependent on the stability properties of Level 2 (Adaptation loop).

Some basic methods used for adaptive laws are [10]:

- Sensitivity methods,
- Gradient methods and least squares methods, and

- Positivity and Lyapunov design methods.

As Lyapunov methods provide strong stability analysis methods we will spend some time discussing them in the next section, but the first methods are worth discussing first as they can provide initial construction tools.

3.3.1 Sensitivity methods

First proposed in the 1960s, sensitivity methods are adaption approaches where the adaption law updates parameters in a direction that minimises some performance function.

For example, if the control objective is for plant output y to track a model references y_m, then a possible performance function based on track error $e_1(\theta_c) = y(\theta_c) - y_m$, where θ_c is some controller parameter to tune, might be

$$J(\theta) = \frac{e_1(\theta_c)^2}{2}.$$

A simple method for updated θ_c might be the gradient search

$$\dot{\theta}_c = -\gamma \nabla J(\theta) = -\gamma e_1(\theta_c) \nabla e_1(\theta_c)$$

where $\gamma > 0$ is the adaption gain and

$$\nabla e_1(\theta_c) \overset{\text{def}}{=\!=} \left[\begin{array}{cccc} \frac{\partial e_1}{\partial \theta_1} & \frac{\partial e_1}{\partial \theta_2} & \dots & \frac{\partial e_1}{\partial \theta_n} \end{array} \right] = \nabla y(\theta_c)$$

is the gradient of e_1 (and in this case also y) with respect to

$$\theta_c = \left[\begin{array}{cccc} \theta_1 & \theta_2 & \dots & \theta_n \end{array} \right].$$

As $\nabla y(\theta_c)$ would require knowledge of the plant, during online implementation approximations to these sensitive functions are typically used (the need for approximation is a drawback).

Here is a small example.

The unknown plant: Consider a simple second-order linear system with an unknown pole

$$\ddot{y} = -a_1 \dot{y} - a_2 y + u,$$

where $y \in R$ is the state, $a_i \in R$ is the unknown pole location, $u \in R$ is the control input to be designed. We assume y and $\dot{y}$ are available for measurement and use in control.

The model reference model: Consider the example model system with our desired response

$$\ddot{y}_m = -2\dot{y}_m - y_m + r,$$

where $r \in R$ is a reference.

The perfect knowledge controller: The control law

$$u = \theta_1^* \dot{y} + \theta_2^* y + r$$

where $\theta_1^* = a_1 - 2$ and $\theta_2^* = a_1 - 1$ will achieve perfect tracking.

The adaptive controller with sensitivity adaption Instead, we use

$$u = \theta_1 \dot{y} + \theta_2 y + r$$

where we seek θ_1 and θ_2 via an adaption via the (MIT) rule:

$$\dot{\theta}_1 = -\gamma_1 e_1 \frac{\partial y}{\partial \theta_1},\ \dot{\theta}_1 = -\gamma_2 e_1 \frac{\partial y}{\partial \theta_2}.$$

However, we will need approximations to these gradients.

The stability properties of sensitivity methods? Whilst sensitivity methods might perform reasonably for small values of γ [10] then stability properties can be difficult to prove (which leads to the need for the analysis tools provided by Lyapunov which we discuss in the next section).

3.3.2 Gradient methods and least squares methods

As just mentioned, one of the main drawbacks of sensitivity methods was that the sensitivity information could not be known directly and approximations are required. One way to overcome this limitation is to propose cost functions that lead to sensitivity functions that are directly available. One such class of cost functions are those related to parameter estimation error (convex functions are selected).

Here is a small example of adjusted adaptive control for a previous plant using least squares methods.

For the example (unknown plant, model reference model and adaptive controller structure) given in the above sensitivity method subsection consider the following adaption law approach.

Plant and estimated plant response It is useful to re-express the plant in terms of controller parameters as follows

$$\ddot{y} = \theta_1^* \dot{y}_f + \theta_2^* y_f + u_f,$$

in terms of the filter signals

$$\dot{y}_f = -\frac{1}{s^2 + 2s + 1}\dot{y},\ y_f = -\frac{1}{s^2 + 2s + 1}y,\ \text{ and } u_f = -\frac{1}{s^2 + 2s + 1}u.$$

We can also then write, for known θ_1 and θ_2 the estimated response

$$\widehat{y} = \theta_1 \dot{y}_f + \theta_2 y_f + u_f.$$

Adaption rule We can now consider the estimation error

$$\epsilon_1(\theta_1, \theta_2) \overset{\text{def}}{=} y - \widehat{y} = y - \theta_1 \dot{y}_f - \theta_2 y_f - u_f.$$

If we select the adaption cost function

$$J(\theta_1, \theta_2) = \frac{\epsilon_1(\theta_c)^2}{2}.$$

where minimising gradients leads to the implementable adaptive laws

$$\dot{\theta}_1 = -\gamma_1 \frac{\partial J}{\partial \theta_1} = \gamma_1 \epsilon_1 \dot{y}_f$$

for some $\gamma_1 > 0$, and

$$\dot{\theta}_2 = -\gamma_2 \frac{\partial J}{\partial \theta_2} = \gamma_2 \epsilon_1 y_f$$

for some $\gamma_2 > 0$.

This is the simplest estimation cost function example, but this approach is amiable to a large range of cost functions and associated minimisation methods.

The stability of Gradient methods and least squares methods? Although the required sensitivity functions can now be calculated it still remains difficult to establish stability properties which again leads to the need for the analysis tools provided by Lyapunov which we discuss in the next section

4 Lyapunov Methods for Stability analysis

4.1 An intuitive explanation of Lyapunov stability analysis: Perhaps all you need to know

Most control systems are non-linear but establishing the stability of a general non-linear system is hard. One of the few general analysis approaches is called a Lyapunov Method. Intuitively, these methods are based on two aspects:

- Establishing the displacement from a desired stability point (the equilibrium) can be described by a function that has energy-like characteristics (for example, positive except at desired point). This is called a (candidate) Lyapunov function.
- Establishing this function has everywhere a negative derivative with respect to time. If both aspects are established it is called a Lyapunov function.

An intuitive explanation is that you are looking for a bowl shape, where the bottom of the bowl is the desired stability point.

Establishing the 2nd aspect is the application of calculus. Presentations of Lyapunov stability analysis can look opaque but essentially these methods are grounded in multi-variable or vector calculus, rather than more specialised maths. This means the concepts are reasonably accessible.

4.2 Introduction into Lyapunov

Lyapunov methods are a key tool in the analysis of linear and nonlinear systems. The main reason we are interested in the analysis of systems is for the purpose of understanding the properties of systems under control. You will already be familiar with some of the techniques of Lyapunov but may not know them that way.

Lyapunov methods are typically proposed in a continuous time setting but they have natural analogs in discrete-time settings. We will present a brief introduction to the main themes of Lyapunov methods in a continuous time setting before providing discrete-time equivalents, where they exist.

Before reading below, you might enjoy this top-level description of the same technical ideas [22, 23].

4.2.1 Background concepts

Let R_+ denote the set of non-negative reals (that is ≥ 0) A commonly used continuous-time model of a nonlinear system is

$$\dot{x}(t) = f_c(t, x(t), u(t)) \tag{4.1}$$

where $t \in R_+$ denotes a time variable, $x(t) \in R^n$ denotes a time-varying state process and $u(t) \in R^m$ is a time-varying input process.

If $u(t)$ is a design state feedback control input we are often interested in the system, where the control has already been incorporated:

$$\dot{x}(t) = f(t, x(t)) \tag{4.2}$$

We will let $s(t, t_0, x_0)$ denote solutions of (4.2) from initial conditions t_0, x_0 in the sense of satisfing

$$\frac{d}{dt} s(t, t_0, x_0) = f(t, s(t, t_0, x_0))$$

for all $t \geq t_0$ and $s(t_0, t_0, x_0) = x_0$.

4.2.2 Some vector space concepts

We let the notation $||.||$ denote the vector norm operation. Examples of norms include the p norms, $p \in [1, \infty]$:

$$||x||_p = \left(\sum_{i=1}^{n} |x^i|^p \right)^{1./p}$$

where x^i is the ith element of the vector $x \in R^N$. The 3 values $p = 1, 2$ and ∞ are the most commonly used.

For a given norm, we define a ball of radius r, centred at the origin, as

$$B_r = \{x \in R^n : ||x|| \le r\}.$$

4.2.3 Some function concepts

We will need a small number of concepts to write down the Lyapunov results.

Definition 1 *A function $\phi(\cdot) : R_+ \to R_+$ is of* **class** $\mathcal{K}$ *if it is continuous, strictly increasing and $\phi(0) = 0$. Further, if $\phi(\cdot)$ is of class $\mathcal{K}$ and also unbounded then it is also of* **class** $\mathcal{K}_\infty$*. Alternative a function $\phi(\cdot) : R_+ \to R_+$ is of* **class** $\mathcal{L}$ *if it is continuous on $[0, \infty)$, strictly decreasing, $\phi(0) < \infty$ and $\phi(r) \to 0$ and $r \to \infty$. Finally, a function $\beta(\cdot, \cdot) : R_+ \times R_+ \to R_+$* **class** $\mathcal{KL}$ *if $\beta(., t)$ is class $\mathcal{K}$ for each $t \ge 0$ and $\beta(s, .)$ is of class $\mathcal{L}$ for each $s \ge 0$.*

Definition 2 *A function $V : R_+ \times R^n \to R$ is said to be a* **locally positive definite function (lpdf)** *if (i) it is continuous, (ii) $V(t, 0) = 0$ for all $t \ge 0$, and (iii) there exist a constant $r > 0$ and a function α of class K such that*

$$\alpha(||x||) \le V(t, x), \ \textit{for all } t \ge 0, \ \textit{and all } x \in B_r$$

Definition 3 *A function V is said to be a* **positive definite function (pdf)** *if B_r can be all R^n.*

Note we can define a function V as **locally negative definite function or lndf** (or **negative definite function or ndf**) if $-V$ as **locally positive definite function** (or **positive definite function**)

Definition 4 *A function $f : X \to Y$ is said to be* **continuous** *at $x_0 \in X$ if for every $\epsilon > 0$ there is a δ such at $||f(x) - f(x_0)|| < \epsilon$ whenenver $||x - x_0)|| < \delta$. (The δ can vary with ϵ and x_0.)*

Definition 5 *A function f is said to be* **continuously differentiable** *and in* **class C1** *if its derivative exists and this derivative is a continuous everywhere.*

4.2.4 Stability concepts

A vector $x_0 \in R^n$ is called an equilibrium of a system (4.2) if

$$f(t, x_0) = 0$$

for all $t \ge 0$ where $\mathbf{0} \in \mathbf{R^n}$ denotes the vector of zeros.

Noting we can always do a change of variables $\bar{x} = x - x_0$ and note that we would have $\dot{\bar{x}}(t) = f(t, \bar{x}(t))$ and now have the condition $\bar{f}(t, 0) = 0$ in terms of the changed state variable $\bar{x}$.

Therefore, to simplify the presentation that follows, we assume we are considering systems where the equilibrium is at $x_0 = \mathbf{0}$.

Definition 6 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **stable** *if for each* $\epsilon > 0$ *there is a* δ *such that* $||x_0|| < \delta$ *implies* $||s(t, t_0, x_0)|| < \epsilon$ *for all* $t \geq t_0$*. Here* δ *can depend on* ϵ*, and* ϵ *and* δ *can both depend on* t_0*. If* ϵ *and* δ *do not depend on* t_0 *then we say the equilibrium is* **uniformly stable***. A system is* **unstable** *if it is not stable.*

Definition 7 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **attractive** *if for each* $t_0 \geq 0$ *there is a* $\eta(t_0) > 0$ *such that* $||x_0|| < \eta(t_0)$ *implies* $||s(t, t_0, x_0)|| \to 0$ *as* $t \to \infty$*.*

Definition 8 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is (uniformly)* **asymptotically stable** *if is it (uniformly) stable and attractive.*

Definition 9 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **exponentially stable** *if there exist constants* $r, a, b > 0$ *such that* $||s(t, t_0, x_0)|| \leq a||x_0|| \exp(-bt)$ *for all* $t \geq t_0$ *and all* $x_0 \in B_r$*.*

In this way, exponentially stable is a stronger property than asymptotically stable which is stronger than stable and attractive.

Definition 10 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **globally exponentially stable** *if there exist constants* $r, a, b > 0$ *such that* $||s(t, t_0, x_0)|| \leq a||x_0|| \exp(-bt)$ *for all* $t \geq t_0$ *and all* $x_0 \in R^n$ *(that is, the ball is all of* R^n*)*

This leads to the following theorem that provides an alternative description of stability.

Theorem 4.1 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **stable** *if and only if for each* $t \geq t_0$ *there exist a ball radius* $r(t_0)$ *and a function* ϕ_{t_0} *of class* K *such that* $||s(t, t_0, x_0)|| \leq \phi_{t_0}(||x_0||)$ *for all* $t \geq t_0$ *and all* $x_0 \in B_{r(t_0)}$*.*

4.3 Lyapunov's direct method of stability

Theorem 4.2 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **locally stable** *if there exists a* C^1 *lpdf* $V : R_+ \times R^n \to R$ *and a constant* $r > 0$ *such that*

$$\dot{V}(t, x) \leq 0$$

for $t \geq t_0$ *and all* $x_0 \in B_r$*. If* $\dot{V}$ *is lndf then stability is asymptotic.*

If there are global properties and $\dot{V}$ is negative definite (not just semi-negative definite) we have the following global result.

Theorem 4.3 *The equilibrium* $\mathbf{0}$ *of* (4.2) *is* **globally asymptotic stable** *if there exists a* C^1 *pdf* $V : R_+ \times R^n \to R$, $\dot{V}$ *is ndf and* $V(x) \to \infty$ *as* $||x|| \to \infty$.

See below some examples for motion controllers to understand how this is used in practice.

4.4 Lyapunov's indirect method of stability

When we can consider linear systems we have $f(t,x) = Ax$ we have the much simpler result described in the following theorem.

Theorem 4.4 *The following 3 conditions are equivalent:*

1. *The equilibrium* $\mathbf{0}$ *of* (4.2) $f(t,x) = Ax$ *is globally asymptotically stable (exponentially stable).*
2. *All eigenvalues of* A *have negative real parts.*
3. *For any positive definite symmetric matrix* Q, *there exists a unique positive definite symmetric matrix* P *which is the solution of the following Lyapunov equation*
$$AP + A'P = -Q.$$

The key step in the proof of this theorem is establishing the condition of the 3rd condition, which helps establish the 1st and 2nd conditions. If the 3rd condition holds then $V(x) = x'Px$ is a suitable Lyapunov function for the system, and here $\dot{V} = -x'Qx$, showing the first condition/ The second condition also follows via an argument about eigenvalues of P and Q (full details in [9]).

4.4.1 Linearisation

We can linearise the system (4.2) to seek to understand local stability properties. Let us consider a time-variance system and suppose $f(0) - 0$. We now write

$$f(x) = Ax + f_1(x)$$

where $A = \frac{\partial f}{\partial x}$ and $f_1(x) \overset{\text{def}}{=\!=} f(x) - Ax$. We assume f is continuous w.r.t to x in the sense that

$$\lim_{||x|| \to 0} \frac{||f_1(x)||}{||x||} = 0 \tag{4.15}$$

We can consider

$$\dot{z} = Az$$

to be considered a linearised version of the system around the equilibrium $\mathbf{0}$, and understand the stability properties as follows:

Theorem 4.5 *Consider (4.2) and assume f is time-variant continuously differentialable.. Assume $f(0) = 0$ and A, f_1 are defined above that that (4.15) holds. Assume A satisfies the conditions of Thm 4.4. Then* **0** *is a (locally) exponentially stable equilibrium of (4.2).*

Proof. Using the Lyapunov equation of Thm 4.4 to find a suitable P for $Q = I$ (the identity matrix of the required size). We note for this P there exists a β (say maximum eigenvalue of P) such that

$$x'Px \leq \beta x'x.$$

Now consider $V = x'Px$. Then we can write

$$\dot{V} = -x'x + 2x'Pf_1(x)$$

Due to (4.15), we can pick a r and a $\rho < 0.5$ such that

$$||f_1(x)|| \leq \frac{2\rho}{\beta}x'x$$

for all $t \geq 0$ and all x with $||x|| < r$. Hence, for all $t \geq 0$ and all x with $||x|| < r$, we can bound

$$|2x'Pf_1(x)| \leq \frac{2\rho}{\beta}x'x$$

Therefore, for all $t \geq 0$ and all x with $||x|| < r$,

$$\dot{V} < -(1 - 2\rho)x'x$$

and hence $\dot{V}$ is a locally positive definite function, the conditions of Thm 4.3 are satisfied and this theorem local exponential stability result follows. ∎

4.5 Discrete time analogs

Repeating the presentation from [3, Section 2.4]A commonly used discrete-time model of a nonlinear system is

$$x_{k+1} = f_c(k, x_k, u_k) \tag{4.20}$$

where $k = 0, 1, \ldots$ denotes a time variable, $x_k \in R^n$ denotes a time-varying state process and $u_k \in R^m$ is a time-varying input process

If u_k is a design state feedback control input we are often interested in the system, where the control has already been incorporated:

$$x_{k+1} = f(k, x_k) \tag{4.21}$$

We will say x_0 is an equilbruim of (4.21) if $f(k, x_0) = x_0$ for all $k \geq 0$.

Definition 11 *The equilibrium* $\mathbf{0}$ *of* (4.21) *is* **stable** *if for each* $\epsilon > 0$ *there is a* $\delta > 0$ *such that* $||x_0|| < \delta$ *implies* $||x_k|| < \epsilon$ *for all* $k \geq 0$.

Definition 12 *The equilibrium* $\mathbf{0}$ *of* (4.21) *is* **asymptotically stable** *if it is stable and* $\delta > 0$ *can be selected such that* $||x_0|| < \delta$ *implies* $||x_k|| \to 0$ *as* $k \to 0$.

Using class $\mathcal{KL}$ functions, we note that the equilibrium $\mathbf{0}$ of (4.21) is asymptotically stable if there exist a $\beta(\cdot,\cdot) \in$ class $\mathcal{KL}$ such that for all x_0 and all $k > 0$ we have

$$||x_k|| \leq \beta(||x_0||, k)$$

That is, the system state decreases to $\mathbf{0}$ as $k \to \infty$.

Lyapunov's direct method of stability

Theorem 4.6 *The equilibrium* $\mathbf{0}$ *is* **stable** *if there exists a Lypunov function of the state* $V(\cdot) : R^n \to R_+$ *and* $a_1(\cdot), a_2(\cdot), a_3(\cdot) \in$ *class* $\mathcal{K}_\infty$ *such that the lypunov function satisfies [3, 5]*

$$a_1(||x_k||) \leq V(x_k) \leq a_2(||x_k||) \tag{4.26}$$

$$V(x_{k+1}) - V(x_k) \leq -a_3(||x_k||) \tag{4.27}$$

for all x_k *and all* $k \geq 0$.

This theorem provides a very general tool for establishing the stability of nonlinear discrete-time systems. Unfortunately, it can be difficult to design the Lypunouv, and the inability to find a V does not mean the system is unstable.

Lyapunov's indirect method of stability When we can consider linear systems we have $f(t, x) = Ax_k$ we have the much simpler result described in the following theorem.

Theorem 4.7 *The equilibrium* $\mathbf{0}$ *of* (4.21) *having* $f(t, x) = Ax_k$ *is globally exponentially stable if all the eigenvalues of* A *have a magnitude less than 1.*

This stability result can quickly be seen in a number of ways, including from the application of Theorem 4.6. Consider the choice of $V(x_k) = ||x_k||$, $a_1(||x_k||) = 0.9||x_k||$, $a_2(||x_k||) = 1.1||x_k||$ and $a_3(||x_k||) = (1 - \lambda_A)||x_k||$ where λ_A is the magnitude of largest eigenvalue of A. Note that $a_1(\cdot), a_2(\cdot), a_3(\cdot) \in$ class $\mathcal{K}_\infty$, and these choices immediately satisfy the first (upper and lower bounding) condition of Theorem 4.6. We then note that $||x_{k+1}|| = ||Ax_k|| \leq \lambda_A||x_k||$ from properties of norms (spectral radius of matrices and matrix-induced norms). Hence if $\lambda_A < 1$ we can write the second (decreasing) condition of Theorem 4.6.

4.5.1 Why useful to us?

Can be used to confirm the stability properties of any proposed controller (including non-linear situations) through the application of the Lyapunov direct method to the resulting closed-loop system. This includes stability properties of:

- model identification algorithms
- direct adaptive control mentioned earlier,
- non-linear control design paradigms, such as backstepping [15] (and see example below in Section 4.6.1),
- various applications such as reference tracking (marine craft vehicle example [16]), visual-servoing (see example below in Section 4.6.4) and more.

4.6 Design Examples

4.6.1 Toy continuous-time problem: Backstepping control design

Example from [15, p. 21].

Consider the 2D system

$$\dot{\xi} = \xi^4 + \sin(\xi) + \eta \tag{4.29}$$

$$\dot{\eta} = \nu \tag{4.30}$$

where ξ, η are the state, and ν is the backstepping control to be designed. Here we consider η to be a virtual control.

Designing a virtual controller: We first look for a stablising control $\eta = \phi(\xi)$ for (4.29). A simple choice is

$$\phi(\xi) = -\xi^4 - \sin(\xi) - k_1\xi$$

with $k_1 > 0$. Notice how the control choice cancels the non-linear terms, and adds a linear stabilising term, so that we obtain in closed loop $\phi(\xi) = -k_1\xi$, which is asymptotically stable.

Stability can be confirmed with the following candidate Lyapunov function:

$$V_i(\xi) = \frac{1}{2}\xi^2.$$

We can quickly see that

$$\dot{V}_i(\xi) = \xi\dot{\xi} = -k_2\xi^2$$

and Theorem 4.3 applies and stability established.

Backstepping to achieve the virtual control: To introduce our virtual control for backstepping we introduce a change of variables

$$z = \eta - \phi(\xi) = \eta + \xi^4 + \sin(\xi) + k_1\xi. \tag{4.32}$$

Then it follows that

$$\dot{\xi} = \xi^4 + \sin(\xi) + \eta = z - k_1\xi.$$

We also find that

$$\dot{z} = \dot{\eta} + (4\xi^3 + \cos\xi + k_1)\dot{\xi} = (4\xi^3 + \cos\xi + k_1)(z - k_1\xi) + \nu.$$

Hence, we can re-express the system as

$$\begin{aligned}\dot{\xi} &= z - k_1\xi\\ \dot{z} &= (4\xi^3 + \cos\xi + k_1)(z - k_1\xi) + \nu.\end{aligned}$$

Using standard backstepping ideas, we consider the candidate total Lyapunov function

$$V(\xi, z) = V_1(\xi) + \frac{1}{2}z^2 = \frac{1}{2}(\xi^2 + z^2).$$

We find that

$$\dot{V} = \xi\dot{\xi} + z\dot{z} = \xi(z - k_1\xi) + z[(4\xi^3 + \cos\xi + k_1)(z - k_1\xi) + \nu].$$

After re-arrangement we have

$$\dot{V} = -k_1\xi^2 + z[\xi + (4\xi^3 + \cos\xi + k_1)(z - k_1\xi) + \nu].$$

Therefore, if we select

$$\nu = -\xi - (4\xi^3 + \cos\xi + k_1)(z - k_1\xi) - k_2\nu \tag{4.33}$$

we will obtain

$$\dot{V} = -k_1\xi^2 - k_2z^2$$

and Theorem 4.3 can be applied and stability established.

We can simulate the closed-loop response of the system (4.29)-(4.30) with control (4.33) (using the change of variables (4.32)). Consider gains $k_1 = 5$ & $k_2 = 5$, with initial conditions $\xi(0) = 1.8$ and $\eta(0) = 7.4$; the response is shown in Figure 7 with MATLAB code below.

MATLAB to simulate closed-loop response: The problem is slightly stiff, so ode23 is more reliable than ode45.

```
Y0=[1.8, 7.4]';
TSPAN=[0,5];
ODEFUN='Ex1p1closedloop';
[TOUT,YOUT] = ode23(@Ex1p1closedloop,TSPAN,Y0');
```

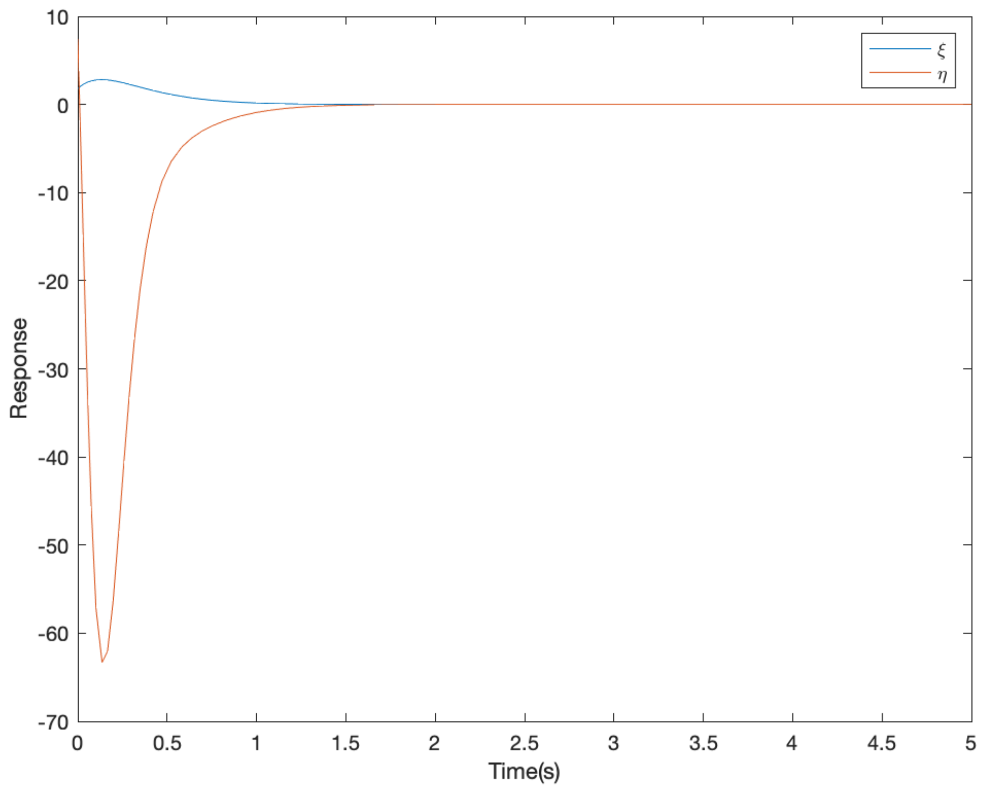

Figure 7: Example of closed-loop response.

```
plot(TOUT,YOUT)
xlabel('Time(s)')
ylabel('Response')
legend('\xi','\eta')

function ds=Ex1p1closedloop(t,x)
e=x(1);
n=x(2);
k1=5;  % gains
k2=5;
z= n+e.^4+sin(e)+k1.*e;  %transformed variable
v=(-e-(4.*e.^3+cos(e)+k1).*(z-k1.*e)-k2.*z);  %virtual control
ds(1,1)= e.^4+sin(e)+n;  % system
ds(2,1)= v;
end
```

You can find a number of other backstepping design examples in [15].

4.6.2 General backstepping control design

A general approach works as follows. If we have a non-linear system as

$$\begin{aligned}\dot{\xi} &= f(\xi) + \eta \\ \dot{\eta} &= \nu\end{aligned}$$

where we assume that $f(.)$ is C1. The first step is to design a virtual controller $\eta = \phi(\xi)$ having

$$\phi(\xi) = -f(\xi) - k_1\xi$$

with $k_1 > 0$ and then introduce a change of variables

$$z = \eta - \phi(\xi).$$

If we consider the Lyapunov function

$$V(\xi, z) = \frac{1}{2}(\xi^2 + z^2)$$

we find that

$$\dot{V} = \xi\dot{\xi} + z\dot{z} = -k_1\xi^2 + z(\xi + \dot{z}) = -k_1\xi^2 + z(\xi + \dot{\phi}(\xi) + \nu)$$

where we note that $f(.)$ is C1 so that $\dot{\phi}(\xi)$ will make sense. Then if we select the controller

$$\nu = -\xi - \dot{\phi}(\xi) - k_2 z$$

with $k_2 > 0$ we will have

$$\dot{V} = -k_1\xi^2 - k_2 z^2$$

and Theorem 4.3 can be applied and stability established.

4.6.3 Robotics example in continuous-time: Wheeled robot

Taken from [19, Exercise 9.8], original paper [20]. Consider the dynamics of a wheeled robot

$$\dot{x}^c = u_1\cos(\theta) \tag{4.38}$$

$$\dot{y}^c = u_1\sin(\theta) \tag{4.39}$$

$$\dot{\theta} = u_t \tag{4.40}$$

where(x^c, y^c) R^2 is the vehicle's position in a cartesian coordinate frame, $\theta \in R$ is heading angle (angle from x-axis, counter clockwise), and u_1 is forward speed.

The stability is easier to analyse in polar coordinates. Let z_1 be the radial ($\sqrt{(x^c)^2 + (y^c)^2}$), z_2 be the angular coordinate ($\tan^{-1}(y^c/x^c)$), and define $z_3 = z_2 - \theta$. The dynamics then become

$$\begin{aligned}
\dot{z}_1 &= u_1 \cos z_3 \\
\dot{z}_2 &= -\frac{u_1 \sin z_3}{z_1} \\
\dot{z}_3 &= -\frac{u_1 \sin z_3}{z_1} - u_t.
\end{aligned}$$

We consider the controller:

$$\begin{aligned}
u_1 &= -z_1 \cos z_3 \\
u_t &= z_3 + \frac{(z_2 + z_3) \cos z_3 \sin z_3}{z_3},
\end{aligned}$$

where we define $u_t = z_2 + 2z_3$ when $z_3 = 0$ (its limit value, ensuring continuous controller). Note that these can be modified to include gains, as shown in [20].

To understand stability we consider the Lyapunov function $V(z) = (1/2)z'z$. We see that

$$\begin{aligned}
\dot{V} &= z\dot{z} \\
&= -z_1 z_1 \cos z_3 \cos z_3 + z_2 \frac{z_1 \cos z_3 \sin z_3}{z_1} + z_3 \left(\frac{z_1 \cos z_3 \sin z_3}{z_1} - z_3 - \frac{(z_2 + z_3) \cos z_3 \sin z_3}{z_3} \right) \\
&= -(z_1 \cos z_3)^2 + z_2 \cos z_3 \sin z_3 + z_3 \left(\cos z_3 \sin z_3 - z_3 - \frac{(z_2 + z_3) \cos z_3 \sin z_3}{z_3} \right) \\
&= -(z_1 \cos z_3)^2 + z_2 \cos z_3 \sin z_3 + z_3 \cos z_3 \sin z_3 - (z_3)^2 - (z_2 + z_3) \cos z_3 \sin z_3 \\
&= -(z_1 \cos z_3)^2 - (z_3)^2
\end{aligned}$$

and hence the controller system is stable. A number of simulation studies are presented in the original paper [20].

A nice alternative controller is presented in [21]. The analysis in this paper is in the cartesian coordinate frame.

4.6.4 Robotics example in discrete-time: Visual-servoing

This example comes from [14]. We will simplify by assuming the image Jacobian is not partitioned (M=6 degrees of freedom in the paper) and the feature correspondence problem has already been solved. A more complex situation is handled in [14] but the key Lyapunov aspects remain the same as her.

Consider N observed image feature s_i defined an image feature vector $s = (s_1 \dots s_N)$. Consider also a vector of reference image features $s^* = (s_1^* \dots s_N^*)$, which implicitly defines the desired camera position and orientation (pose) to view the target. The image feature error e is then defined as the difference between the observed and reference image features.

Consider the discrete-time system dynamics:

$$s_{k+1} = s_k + L_k u_k \Delta T$$

where $L_k \in 2N \times 6$ is the image Jacobian and $\Delta T \in R$ is sample period. The image Jacobian L_k related sensor spatial velocities to pixel velocities and is a function of the image features s_k and the range to the image features (assume it is completely known), see [17, 14] for more details. A video explanation of the image Jacobian can be found at [18].

The Image Jacobian: The image Jacobian or interaction matrix L_i for the ith point feature s_i represented in spherical coordinates can be separated such that [14]:

$$L_i = [L_{i,a}, L_{i,b}]$$

where the subscripts a and b denote two separate partitions of the image Jacobian (partitions that can be arbitrarily selected). Below we present the partitions corresponding to separating the image Jacobian into the components corresponding to the translational (a) and rotational (b) degrees of freedom:

$$L_{i,a} = \begin{bmatrix} \frac{-C(s_i^1)C(s_i^2)}{r_i} & \frac{-C(s_i^1)S(s_i^2)}{r_i} & \frac{S(s_i^1)}{r_i} \\ \frac{S(s_i^2)}{r_i S(s_i^1)} & \frac{-C(s_i^2)}{r_i S(s_i^1)} & 0 \end{bmatrix}$$

$$L_{i,b} = \begin{bmatrix} S(s_i^2) & -C(s_i^2) & 0 \\ \frac{C(s_i^2)C(s_i^1)}{S(s_i^1)} & \frac{S(s_i^2)C(s_i^1)}{S(s_i^1)} & -1 \end{bmatrix}$$

and $s_i = [s_i^1, s_i^2]$, $C(\cdot) = \cos(\cdot)$ and $S(\cdot) = \sin(\cdot)$. Given a set of N point features, the complete $N \times 6$ image Jacobian

$$L = \begin{bmatrix} L_{1,a} & L_{2,a} \\ \vdots & \vdots \\ L_{N,a} & L_{N,a} \end{bmatrix}$$

An alternative choice for image Jacobian can be found in [17].

We now present a control design and related stability results

Lemma 4.1 *Consider those dynamics controlled using control input u_k for all k such that*

$$\bar{L}_k(u)'(\bar{L}_k(u) + 2e_k) < 0$$

where $e_k = s_k - s^*$ and we define

$$\bar{L}_k(u) = L_k u_k \Delta T.$$

Then the system converges in the sense

$$|e_k|_1 \to 0, \; k \to \infty.$$

Proof. Consider the candidate Lyapunov function V_k defined as

$$V_k = e_k' e_k$$

then we can write

$$\begin{aligned} V_{k+1} &= e_{k+1}' e_{k+1} \\ &= (s_{k+1} - s^*)'(s_{k+1} - s^*). \end{aligned}$$

Letting us use the shorthand $\bar{L}_k = \bar{L}(u, k)$ we can now write

$$\begin{aligned} V_{k+1} &= (s_k + \bar{L}_k - s^*)'(s_k + \bar{L}_k - s^*) \\ &= s_k' s_k + 2\bar{L}_k'(s_k - s^*) + \bar{L}_k'\bar{L}_k - 2s_k s^* + (s^*)'s^* \\ &= (s_k - s^*)'(s_k - s^*) + \bar{L}_k'\bar{L}_k + 2\bar{L}_k'(s_k - s^*) \\ &= V_k + \bar{L}_k'\bar{L}_k + 2\bar{L}_k' e_k \\ &= V_k + \bar{L}_k'(\bar{L}_k + 2e_k). \end{aligned}$$

It follows that if theorem condition holds that $\bar{L}(u,k)'(\bar{L}(u,k) + 2e_k) < 0$ for all k then $V_{k+1} < V_k$ for all k and Theorem 4.6 applies and the theorem claim holds. ∎

This result is useful in the sense that it provides a method for selecting u_k at each time instant. We have L_k, ΔT and e_k, so we are free to select u_k to met the condition $\bar{L}(u,k)'(\bar{L}(u,k) + 2e_k) < 0$ (assuming that if possible).

For an example of a possible algorithmic form, we could propose the control

$$u_k = -\frac{\gamma}{\Delta T}\widehat{L}_k^+ e_k$$

where $0 < \gamma < 2$, and $\widehat{L}_k^+$ is Moore-Penrose pseudo inverse of L_k (generalisation of matrix inverse, when matrix inverse does not exist). Then we can see that $\bar{L}_k = -\gamma\widehat{L}_k^+ L_k e_k$ and $\bar{L}(u,k)'(\bar{L}(u,k) + 2e_k) = -\gamma(\widehat{L}_k^+ L_k e_k)'(2I - \gamma\widehat{L}_k^+ L_k)e_k = -\gamma e_k' \check{L}_k e_k$ where $\check{L}_k = (\widehat{L}_k^+ L_k)'(2I - \gamma\widehat{L}_k^+ L_k)$. When the matrix inverse of L_k exist, $\widehat{L}_k^+ = L_k^{-1}$ and $\check{L}_k = (2-\gamma)I$, and we can see that $\bar{L}(u,k)'(\bar{L}(u,k) + 2e_k) = -\gamma(2-\gamma)e_k' e_k$ and the theorem condition holds if $0 < \gamma < 2$.

Theorem condition also holds as long as γ is selected so that $\widehat{L}_k^+ L_k$ is never too strongly positive definite (in the sense $\check{L}_k$ remains positive definite).

We could even adjust γ at each time step to maximise the rate of convergence; that is, select γ to make $\bar{L}(u,k)'(\bar{L}(u,k) + 2e_k)$ as negative as possible.

References

[1] S. Bennett, "A brief history of automatic control," in IEEE Control Systems, vol. 16, no. 3, pp. 17-25, Jun 1996. https://ieeexplore.ieee.org/document/506394

[2] K. J. Astrom and P. R. Kumar, "Control: A perspective," Automatica, vol. 50, no. 1, pp. 3-43, Jan. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.automatica.2013.10.012

[3] O. Techakesari, Filter and control performance bounds in the presence of model uncertainties with aerospace applications. PhD thesis, Queensland University of Technology, 2013. https://eprints.qut.edu.au/63956/

[4] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, "Control System Design", Prentice Hall, New Jersey, 2001.

[5] H. K. Khalil, Nonlinear Systems, 3rd ed. Upper Saddle River, N.J.: Prentice-Hall, 2002.

[6] Slotine, J.-J.E. and Li, W. Applied Nonlinear Control. Prentice Hall, New Jersey, 1991.

[7] P.A. Ioannou and J. Sun, Robust adaptive control, Prentice Hall, Englewood, Cliffs, NJ, ISBN: 0-13-439100-4

[8] Y. D. Landau and M. M'Saad, Adaptive control, 1st ed. 1998. London, England: Springer, 1998. doi: 10.1007/978-0-85729-343-5. Link to QUT library ebook https://qut.primo.exlibrisgroup.com/permalink/61QUT_INST/1g7tbfa/alma991010464462204001

[9] M. Vidyasagar, Nonlinear System Analysis, (2nd ed.). Society for Industrial and Applied Mathematics SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104. Link to QUT library ebook https://qut.primo.exlibrisgroup.com/permalink/61QUT_INST/1tofoc8/alma991009383291404001

[10] P.A. Ioannou and J. Sun, Robust adaptive control, Prentice Hall, Englewood, Cliffs, NJ, ISBN: 0-13-439100-4

[11] I. Mareels and J. W. Polderman, Adaptive Systems An Introduction, 1st ed. 1996. Boston, MA: Birkhäuser Boston, 1996. doi: 10.1007/978-0-8176-8142-5. Link to QUT library ebook https://qut.primo.exlibrisgroup.com/permalink/61QUT_INST/1g7tbfa/alma991010143442104001

[12] Jesse Haviland and Peter Corke, "Manipulator Differential Kinematics Part I: Kinematics, Velocity, and Applications", https://arxiv.org/abs/2207.01796

[13] Jesse Haviland and Peter Corke, "Manipulator Differential Kinematics Part II: Acceleration and Advanced Applications", https://arxiv.org/abs/2207.01794

[14] A. McFadyen, J. Ford and P. Corke, "Stable image-based visual servoing with unknown point feature correspondence," 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia, 2017, pp. 4278-4282, doi: 10.1109/CDC.2017.8264290. QUT eprints version: https://eprints.qut.edu.au/113191/

[15] Sundarapandian Vaidyanathan, Ahmad Taher Azar, Chapter 1 - An introduction to backstepping control, Editor(s): Sundarapandian Vaidyanathan, Ahmad Taher Azar, In Advances in Nonlinear Dynamics and Chaos (ANDC), Backstepping Control of Nonlinear Dynamical Systems, Academic Press, 2021, Pages 1-32, ISBN 9780128175828, https://doi.org/10.1016/B978-0-12-817582-8.00008-8. (https://www.sciencedirect.com/science/article/pii/B9780128175828000088)

[16] Justin M. Kennedy, Alejandro Donaire & Jason J. Ford (2022) Control of underactuated marine crafts with matched disturbances, International Journal of Control, 95:6, 1634-1644, DOI: 10.1080/00207179.2020.1866779 QUT eprints version: https://eprints.qut.edu.au/208533/

[17] F. Chaumette and S. Hutchinson, "Visual servo control part I: basic approaches," IEEE Robotics and Automation, vol. 1070, No. 9932, pp. 82-90, 2006.

[18] P. Corke, The Image Jacobian, QUT Robot Academy, https://robotacademy.net.au/lesson/the-image-jacobian/

[19] R. Tedrake, Underactuated Robotics Algorithms for Walking, Running, Swimming, Flying, and Manipulation http://underactuated.mit.edu/lyapunov.html

[20] M. Aicardi, G. Casalino, A. Bicchi and A. Balestrino, "Closed loop steering of unicycle like vehicles via Lyapunov techniques," in IEEE Robotics & Automation Magazine, vol. 2, no. 1, pp. 27-35, March 1995, doi: 10.1109/100.388294.

[21] Y. Kanayama, Y. Kimura, F. Miyazaki and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," Proceedings., IEEE International Conference on Robotics and Automation, Cincinnati, OH, USA, 1990, pp. 384-389 vol.1, doi: 10.1109/ROBOT.1990.126006.

[22] Collimator, "What is Lyapunov stability theory?" https://www.collimator.ai/reference-guides/what-is-lyapunov-stability-theory

[23] Byju "Lyapunov functions" https://byjus.com/maths/lyapunov-functions/#:~:text=Lyapunov%20functions%20can%20be%20used,points%20in%20non%2Drough%20systems.