INTERACTIVE CONTROL ENGINEERING
Principles and Applications

Volume 9 Model Predictive Control and Economic Optimization

CHAPTER 1

Introduction: Planning Ahead With a Model

Pages 1–21

Edited by John D. Hedengren

How to cite this chapter. "Introduction: Planning Ahead With a Model," in *Model Predictive Control and Economic Optimization*, John D. Hedengren, Ed., vol. 9 of *Interactive Control Engineering: Principles and Applications*. Advanced Process Solutions, LLC, 2026, ch. 1, pp. 1–21.

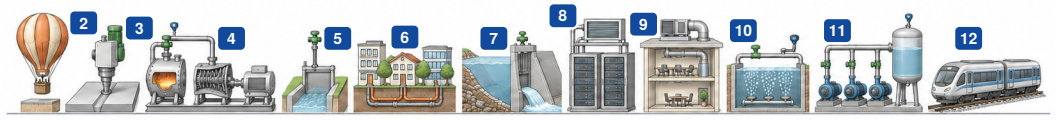
The page numbers in this file are those of the printed volume.

The volume online: contents, other chapters and citation records
https://apmonitor.com/apps/books/v09-mpc-optimization/

Copyright © 2026 Advanced Process Solutions, LLC. All rights reserved. Produced by Advanced Process Solutions, LLC.

CHAPTER 1

Introduction: Planning Ahead With a Model



The chapters 2 hot-air balloon 3 friction stir weld 4 boiler-turbine unit 5 irrigation canal 6 district heating 7 flood reservoir 8 data center cooling 9 office building 10 aeration basins 11 water network pumps 12 electric train

Notice that the first four units are regulated by a predictive controller, the next five decide what the loops should hold, where a price, a permit or a forecast sets the target, and the last two are planned in whole decisions. Illustration, not to scale.

IN PLAIN TERMS **Control That Looks Ahead**

The main idea. An ordinary controller reacts to the error it sees now. A predictive controller uses a model of the plant to guess what comes next, and it plans its moves several steps ahead.

The challenge. The plan is only as good as the model and the forecast behind it. Limits, prices and machines that are either on or off make the best plan harder to find, and it has to be found again every few moments.

Why it matters. Such plans run heat networks, dams, data halls, offices, water works and trains, where they save energy and money.

THREE WAYS TO READ THIS CHAPTER

Explore	7 min	This page, Section 1.2 with its timeline and two History boxes, the Did you know? boxes in Sections 1.4.2, 1.4.7 and 1.4.13, Figure 1.2, and Key points in Section 1.8. No equations.
Study	45 min	The whole chapter in order, with each number of the tank example worked, then Exercises 1 to 10.
Practice	5 min	This page, Section 1.4.13 with its two tables, Section 1.5 with Figure 1.6, Figure 1.2, and Section 1.8.

1.1 The Question This Volume Answers

A feedback loop acts on the error it sees now. A predictive controller acts on the errors a model says are coming, and an optimizer chooses what the loops should aim for. Both cost something: a model to build and keep right, a program to solve every period, and a layer of logic that operators must trust. This volume answers one question: when does planning ahead with a model pay for itself against simpler control, and how is such a plan built so that it survives a wrong model, a limit and an uncertain forecast?

Eleven chapters answer it on eleven plants, each with an app that the reader runs in a browser. Chapters 2 to 5 use prediction to regulate: a balloon behind a dead time, a weld whose gain changes along the joint, a boiler that trades pressure for megawatts and a canal whose gates are kilometers apart. Chapters 6 to 10 use it to decide what the regulators should hold, on a heating network, a flood reservoir, a data hall, an office building and an aeration basin, where a price, a permit or a forecast sets the target. Chapters 11 and 12 plan in whole decisions: pumps that are on or off, and a train that powers, holds, coasts or brakes.

A reader who has worked through the volume can do the following on a plant that is new to the reader.

- Write the prediction of a step-response or state-space model and the quadratic program a predictive controller solves each period.

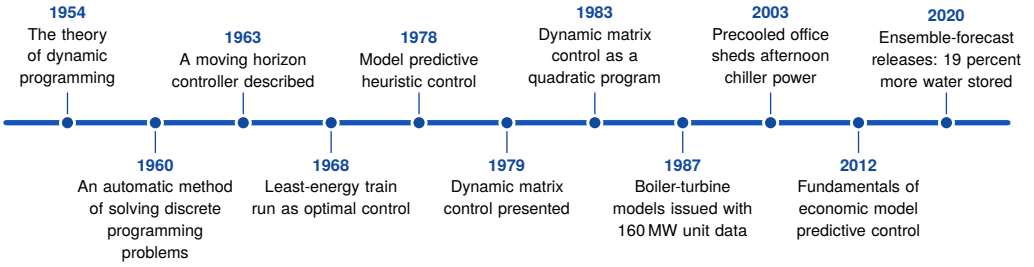


Figure 1.1 Milestones of prediction, optimization and planning in the systems of this volume. The dates are from [5] (1954), [6] (1960), [2] (1963, 1979 and 1983), [8] (1968), [7] (1978), [9] (1987), [10] (2003), [11] (2012) and [12] (2020). Notice that the optimization came first and the plants that store energy or water came last. Events are evenly spaced, not to a time scale.

- Choose a horizon, a move suppression and a disturbance model, and say what each does to speed, robustness and offset.
- Decide which limits are hard and which are soft, and price a soft limit so that it is broken only when it must be.
- Find the active constraint of a steady-state optimum and the back-off that variability forces.
- Set up an economic plan, a scenario tree, a price-coordinated plan and an integer schedule, and read their answers through the structure that optimal control predicts.
- Measure what each of these buys against a well-tuned simpler controller.

This chapter develops each of these on one small example and names the chapter that uses it. Surveys of the theory and of industrial practice [1–3] and a textbook treatment [4] give fuller derivations. PID forms and tuning are taken as known from Volume 2.

1.2 A Short History

The milestones of Figure 1.1 follow the methods of this volume through very different plants. Two came from mathematics before any controller ran them. A paper of 1954 set out the theory of dynamic programming [5], the backward recursion with which Chapter 12 plans a run and Chapter 7 values the water it holds back, and a paper of 1960 gave an automatic method of solving discrete programming problems [6], the branch and bound that schedules the pumps of Chapter 11.

> ⧖ HISTORY **1963 to 1979: Planning Ahead Moves Into Plants**
>
> Solving an open-loop optimization again at every step is older than its industrial use. A moving horizon controller was described in 1963, and a 1967 optimal control text anticipated the practice. The linear quadratic Gaussian controller, grown from Kalman's work of the early 1960s, had little impact on the process industries, whose plants brought constraints, nonlinearity, model error and performance criteria of their own. Industry built its own methods. A conference paper of 1976, summarized in 1978, described model predictive heuristic control on an impulse-response model, applied to the main fractionator of a cracking unit, a power plant steam generator and a polyvinyl chloride plant. Engineers at an oil company first applied a method of their own in 1973 and presented it as dynamic matrix control at a national meeting in 1979 [2, 7].

Optimal control reached one of this volume's plants before predictive control reached any. A paper published in 1968 asked for the run between two stations that uses the least energy and still stops at the next one at a specified instant. Because the speed limit must be observed strictly, it treated the run as a problem with a bounded state variable and obtained the optimal control law [8], the question of Chapter 12.

> ⧖ HISTORY **1983 to 1986: Limits Become Rows of a Quadratic Program**
>
> The first algorithms controlled plants without limits well, but they handled limits in an ad hoc way. A 1980 application to a cracking unit's reactor and regenerator added an equation to the model whenever a predicted input came close to its limit, to drive it back. A 1983 conference paper instead posed dynamic matrix control as a quadratic program in which input and output limits appear explicitly, and a fuller description followed in 1986 [13]. Its Hessian is positive definite for a linear plant, so the program is convex and standard codes solve it. On a pyrolysis furnace the controller set the fuel gas pressure of three burners to hold three temperatures, and it proved most profitable inside on-line optimization, moving a plant smoothly from one constrained operating point to another [2].

The later milestones are the plants of this volume. A report of December 1987 gave nonlinear models of boiler-turbine-alternator units with an extensive set of data from a 160 MW oil-fired unit, and offered both on a disk [9]. Chapter 4 runs its model. In October 2003 an occupied office held its zones at the low end of the comfort range until 2 pm and then let them float, and its chiller power fell by 80 to 100 percent from 2 to 5 pm without a comfort complaint [10], the band that Chapter 9 spends. A 2012 paper set out the fundamentals of economic model predictive control [11], the form of Section 1.4.8. At one flood-control reservoir, releases recommended from a 15-day ensemble streamflow forecast were tried in operation, and in the 2020 water year, the third driest in a 127-year record, the lake ended the winter with 19 percent more water stored [12], the forecast that Chapter 7 plans on.

1.3 Notation and Conventions

The decision index is k, one control period apart, and the controller plans N periods ahead, the prediction horizon. The plan moves the input M times, the move horizon, often in blocks of several periods. The setpoint is r, the predicted output is $\hat{y}$ and the input change is Δu. The weight on tracking is Q and the move suppression is R, as in Equation (1.19) of the framework section. A slack $s \geq 0$ relaxes a soft limit and carries a quadratic price ρ and a linear price ν. An energy or fuel price is π, a Lagrange multiplier or coordinating price is λ, a costate is p and a value function is V. Money is in US dollars.

The chapters keep their own symbols, and several collide. Chapter 2 and Chapter 3 write the step response as a_i, the prediction horizon as P and the move suppression as λ, and Chapter 5 and Chapter 10 also price moves with λ. This chapter writes the step response g_i, because the framework's a is the controller action. Chapter 6 uses λ_j for the price of heat, Chapter 12 uses λ for the price of time and p for the costate, and Chapter 7 calls the costate of stored water its water value w. Linear slack prices appear as ν, σ_i and σ_1 in Chapters 5, 8 and 10. The framework's h is its integration step, so no symbol here uses it.

1.4 Prediction and Optimization on One Heated Tank

1.4.1 The Running Example

The running example is a tank of liquid that a process draws from, kept warm by an electric heater. The process needs the liquid at 50 °C or warmer, and the tank must stay below 80 °C. Each hour the tank keeps 80 % of its excess over the 20 °C room, and each kilowatt-hour of heat raises it by one kelvin, as shown in Equation (1.1).

$$T_{k+1} = 20 + \phi\,(T_k - 20) + u_k - d_k, \qquad \phi = 0.8, \quad 0 \leq u_k \leq 20\,\text{kW}, \quad 50 \leq T_k \leq 80\,^{\circ}\text{C} \tag{1.1}$$

where T_k is the temperature in °C at hour k, u_k the heater power in kW held over the hour and d_k the heat leaving with the drawn liquid, 2 kW on an ordinary hour. The equation is the exact hourly sample of a first-order store with a steady gain of $K = 1/(1-\phi) = 5\,\text{K/kW}$ and a time constant of 4.48 h, a fast loss chosen for clean arithmetic rather than for a particular product. Holding 50 °C takes

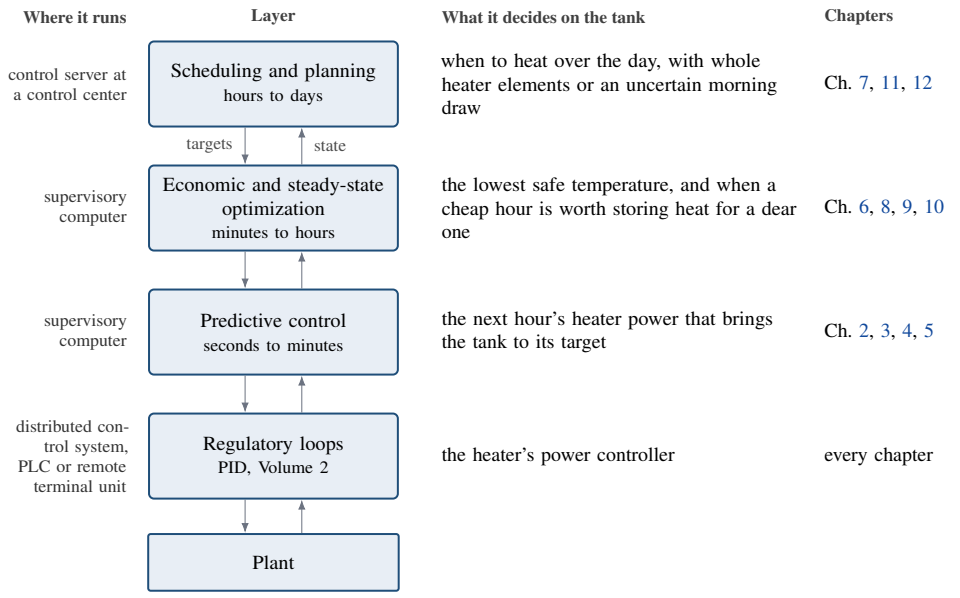


Figure 1.2 Layers of a Control System, Where Each Runs, What Each Decides on the Running Example and the Chapters That Work in It. Down Arrows Carry Targets, Up Arrows Carry Measurements and Limits. Platforms from [2, 14] and the chapters' technology sections.

$u = 8\,\text{kW}$. Electricity costs \$0.10 per kWh from 00:00 to 06:00, \$0.40 from 17:00 to 21:00 and \$0.20 otherwise, and the draw rises to 6 kW from 07:00 to 09:00 and from 18:00 to 21:00.

Figure 1.2 places the volume's chapters in the layers of a plant's control system and states what each layer decides on the tank. The regulatory layer is the subject of Volume 2. The layers above it are this volume's subject, and the sections below climb them in order.

1.4.2 Prediction Models

A predictive controller needs the effect of a move on the future output. The oldest form is the step response, the output change i periods after a unit step of the input. For the tank it is Equation (1.2).

$$g_i = \frac{1-\phi^i}{1-\phi} = 5\left(1-0.8^i\right), \qquad g_1 = 1,\ g_2 = 1.8,\ g_3 = 2.44,\ g_4 = 2.952\ \text{K/kW} \tag{1.2}$$

The prediction splits into a free response f, what the output would do if the input stayed where it is, and a forced response from the planned moves, as shown in Equation (1.3).

$$\hat{y} = f + G\Delta u, \qquad G_{ij} = g_{i-n_j} \text{ for } i > n_j, \text{ else } 0 \tag{1.3}$$

where $\hat{y}$ stacks the N predicted outputs, Δu stacks the M moves and move j is made at period n_j. The matrix G is the dynamic matrix of dynamic matrix control (DMC). The same prediction follows from a state-space model such as Equation (1.18) by repeated substitution, and Equation (1.1) is that model with one state. A step response is easy to obtain from a plant test and needs no physics. A state-space model carries a state that an estimator can correct and handles an unstable or integrating plant without a truncated response.

The chapters use every form. Chapter 2 identifies a step response once from a reduced copy of its model, and Chapter 3 writes one in closed form from a fitted first-order model, with two candidates from two step tests on the same weld. Chapter 5 fits an integrator with a dead time to each reach. Chapter 4 linearizes its balances at the estimate at every decision, Chapter 10 linearizes along the

current plan, and Chapter 6 adds a predictor built only from a building's recorded data. In each case the controller's model is kept apart from the plant's, so a parameter the reader changes is a model error, as it would be on a real plant.

> ✦ DID YOU KNOW? **A Controller Whose Clock Followed the Load**
>
> In one of the first predictive control applications, on a power plant steam generator, the response time of the process varied inversely with the load. The controller met that nonlinearity by executing with a variable sample time [2]. Chapter 4 meets a gain that changes with load by linearizing its model at every decision instead.

1.4.3 The Quadratic Program and the Receding Horizon

The plan minimizes the tracking error over the horizon plus a price on the moves, the objective of Equation (1.19). Substituting the prediction gives a quadratic function of the moves, the quadratic program (QP) of Equation (1.4).

$$\min_{\Delta u} \; (r - f - G\Delta u)^{\top} Q\,(r - f - G\Delta u) + R\,\Delta u^{\top}\Delta u \quad \text{s.t.} \quad u_{\min} \le u \le u_{\max},\;\; y_{\min} \le \hat{y} \le y_{\max} \tag{1.4}$$

Its Hessian $G^{\top}QG + RI$ is positive definite whenever $R > 0$, so the program has one minimum. With no limit active the minimum is linear in the predicted error, as shown in Equation (1.5).

$$\Delta u = \left(G^{\top}QG + RI\right)^{-1} G^{\top}Q\,(r - f) = K_{\text{mpc}}\,(r - f) \tag{1.5}$$

The gain matrix K_{mpc} can be computed once, which is how the original DMC ran [1]. Only the first move is applied. At the next period the output is measured, the prediction is corrected and the plan is made again, the receding horizon of every predictive controller in this volume [15].

With one move ($M = 1$), unit weights and no move suppression, Equation (1.5) reduces to $\Delta u = \sum g_i e_i / \sum g_i^2$, where $e_i = r - f_i$. For a setpoint step from 50 to 55 °C on the tank, every e_i is 5 K. A horizon of one hour gives $\Delta u = 5/g_1 = 5\,\text{kW}$, the move that reaches the new setpoint in one hour. A horizon of three hours gives $5 \times 5.24/10.19 = 2.57\,\text{kW}$. As N grows the move tends to $5/K = 1\,\text{kW}$, the move that reaches the setpoint only at steady state.

When a limit is active the program has no closed form. It is solved by an active-set or interior-point method, a dual active-set method in the framework's apps (Section 1.6.3), and the solution is not the unconstrained plan clipped to the limits. The difference matters most for output limits, which clipping cannot honor at all, and for plans with several moves, where a limit on a later move changes the best first move [13]. A day's plan for the tank with $N = 24$ has 24 moves, 24 input bounds and 48 temperature bounds, a program the browser solves in milliseconds.

Chapter 2 and Chapter 3 compute the gain matrix once and handle the input box with a few projected-gradient steps or a clamp. Chapter 4, Chapter 5, Chapter 9 and Chapter 8 solve the full program with limits on outputs at every decision.

1.4.4 Horizons and Tuning

Table 1.1 runs the one-move controller on the tank for the 5 K step, with the output-bias correction of Section 1.4.5, and Figure 1.3 shows three of the runs. A short horizon is fast and fragile. With $N = 1$ the tank settles in one hour, but a plant whose gain is twice the model's sits on the stability limit. A long horizon is slow and robust: with $N = 48$ the move is barely more than the steady move, the loop takes 13 hours to stay within 0.25 K, and the plant's gain may be 9.5 times the model's before the loop goes unstable. The horizon acts as a tuning knob, much as the closed-loop time constant does in a PI tuning rule.

The last column of Table 1.1 is the honest one. A gain error of 1.7, the ratio between the two step tests of Chapter 3, overshoots by 5 to 9 K at every horizon. A longer horizon keeps the loop stable

Table 1.1 One-Move DMC on the Tank for a 5 K Setpoint Step, by Prediction Horizon ($R = 0$)

Horizon N (h)	First move (kW)	Within 0.25 K after (h)	Plant gain tolerated	Overshoot at gain ×1.7 (K)
1	5.00	1	×2.00	9.1
2	3.30	3	×3.03	6.2
3	2.57	5	×3.89	5.1
6	1.74	8	×5.75	6.0
12	1.31	10	×7.64	6.9
48	1.05	13	×9.50	7.6

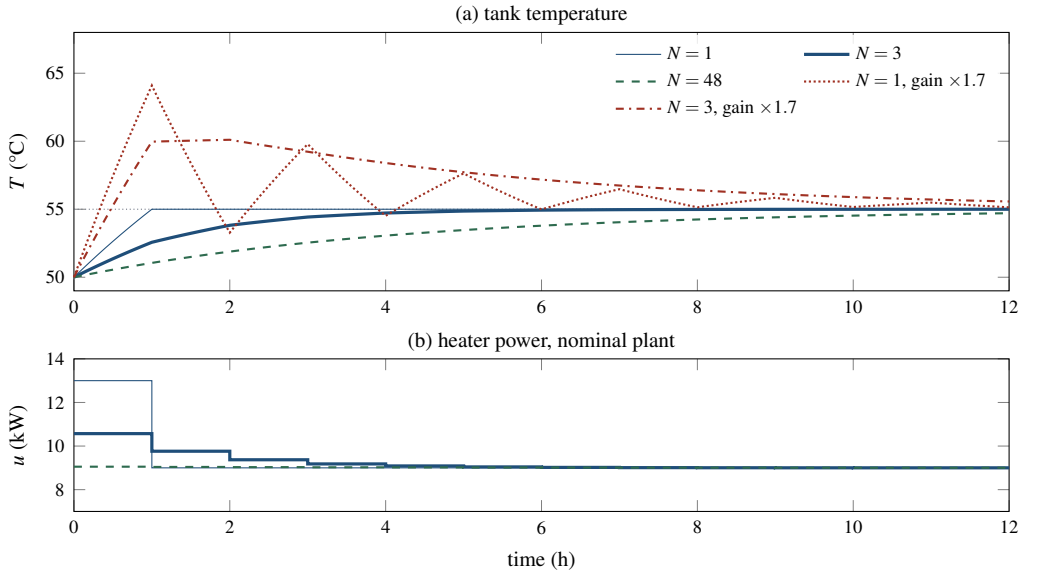


Figure 1.3 A 5 K Setpoint Step Under One-Move DMC. Thin Solid $N = 1$, Thick Solid $N = 3$, Dashed $N = 48$ on the Nominal Plant. Dotted ($N = 1$) and Dash-Dotted ($N = 3$) on a Plant Whose Heater Gain Is 1.7 Times the Model's. The Dotted Run Drives the Heater to Zero in Its Second Hour.

but does not make the first move right, because the first move is sized by the model's gain. Move suppression has the same effect as a longer horizon: with $N = 3$, $R = 10$ halves the first move to 1.30 kW and raises the tolerated gain to 10.8. Several moves ($M > 1$) with $R = 0$ return the deadbeat plan, which is why practice pairs a long horizon with a few blocked moves and a small R [2].

Chapter 2 finds a horizon too long for its model, whose linear step response fails over long horizons, and fixes it by shortening the horizon and weighting the end. Chapter 3 scales R to the model so that one setting means the same aggressiveness for both of its models, and Chapter 5 and Chapter 10 fix the horizon by its physics first and tune the prices afterward.

1.4.5 Offset-Free Tracking

A predictive controller has no integral term, so it removes a steady error only if its prediction learns about the error. Let the draw rise from 2 to 4 kW at hour 2 and not be measured. Figure 1.4(a) shows three answers with $N = 3$. A controller that predicts from the measured temperature but keeps the nominal draw settles 3.89 K low, and its offset grows with the horizon, to 9.50 K at $N = 48$. Two disturbance models remove it [16, 17]. The output bias of DMC adds the present prediction error $b_k = y_k - \hat{y}_{k|k-1}$ of an open-loop model to every point of the horizon. An input-disturbance estimate corrects the draw itself, $\hat{d}_k = \hat{d}_{k-1} - L(y_k - \hat{y}_{k|k-1})$ with $L = 0.5$ here, which is the disturbance state

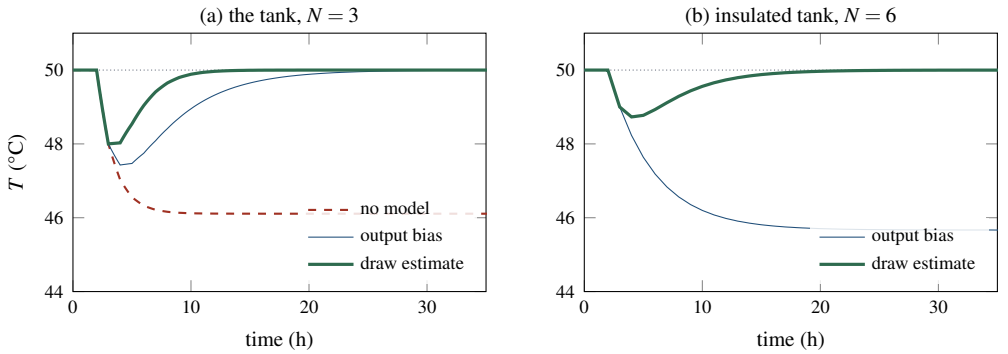


Figure 1.4 An Unmeasured Rise in the Draw at Hour 2. (a) The Tank of Equation (1.1), Draw 2 to 4 kW: No Disturbance Model (Dashed), Output Bias (Thin Solid), Draw Estimate (Thick Solid). (b) The Tank With $\phi = 1$, Draw 2 to 3 kW: the Output Bias (Thin Solid) Leaves a Steady Offset.

of the Kalman filter in Section 1.6.3. Both settle at 50 °C. The bias needs 15 hours to come within 0.25 K, because it treats a load at the input as a step at the output that the model's 4.48 h lag then releases slowly. The estimator needs 7 hours.

The output bias fails on an integrating plant. With perfect insulation ($\phi = 1$) the tank integrates, $g_i = i$, and a draw 1 kW above the model makes the open-loop model drift away from the plant by $\delta = 1$ K an hour. A constant bias cannot represent a drift, and setting the one-move law to zero at steady state gives the offset of Equation (1.6).

$$T_\infty - r = -\delta \frac{\sum_{i=1}^{N} i\, g_i}{\sum_{i=1}^{N} g_i} = -\delta \frac{2N+1}{3} \tag{1.6}$$

At $N = 6$ it is 4.33 K, as Figure 1.4(b) shows. The draw estimate settles at 50 °C, and a filtered ramp of the bias, the drift term of Chapter 2, gives the same trajectory here. That chapter measures the same failure on its integrating altitude. Chapter 4 adds a fuel-quality disturbance to its Kalman filter and checks that the augmented model stays observable. Chapter 5, Chapter 9 and Chapter 10 estimate an unexplained inflow or load as a random walk, and Chapter 3 keeps the output bias on a self-regulating temperature.

1.4.6 Hard and Soft Constraints

An input limit is physical and always holds, so it is written as a hard constraint. An output limit is a wish that a disturbance can make impossible. Suppose a cleaning cycle draws 16 kW from 03:00 to 06:00. From 50 °C, full heat gives 48.0, 46.4 and 45.1 °C at the next three hours, so a plan that insists on $T \geq 50$ has no feasible point and returns nothing [18]. The usual remedy relaxes the limit with a slack, as shown in Equation (1.7).

$$\hat{y}_k \geq y_{\min} - s_k, \qquad s_k \geq 0, \qquad J \leftarrow J + \sum_k \left(\rho\, s_k^2 + \nu\, s_k\right) \tag{1.7}$$

With the slack the plan heats at full power and the tank falls to 45.1 °C, the least violation the heater allows. A plan that sees the cleaning cycle from midnight does better. It needs 59.5 °C at 03:00, reaches it by heating at 17.5 kW from 02:00, and holds the floor throughout at full power, for 5.6 kWh more heat than the plan that breaks it. Preview turns an infeasible limit into a feasible one.

The price decides when the limit gives way. At steady state one more kelvin costs $\pi(1-\phi) = \$0.04$ an hour at \$0.20 per kWh, the Lagrange multiplier of the floor. A quadratic price alone lets the plan sit below the limit by $\lambda/2\rho$, 2 K at $\rho = \$0.01$ per kelvin squared and hour, whenever saving energy

is worth it. A linear price larger than every multiplier makes the soft limit exact: it is broken only when it cannot be held. The largest multiplier sits at the last hour of the horizon, where nothing later pays for the heat, and is the full price, \$0.20 per kelvin. On a 24-hour plan $\nu = 0.19$ lets the tank sag to 42 °C at the end, and $\nu = 0.21$ holds 50 °C throughout.

Chapter 5, Chapter 8, Chapter 10 and Chapter 6 price their slacks with both terms. Chapter 4 keeps its pressure band soft and adds tube MPC, which tightens the nominal limits by the largest gap a bounded disturbance can open. Chapter 11 adds slacks only after a hard program fails, and flags the plan.

1.4.7 Steady-State Optimization and Active Constraints

Above the predictive layer, an optimizer chooses the setpoints. At steady state the tank's cost is linear in its temperature, as shown in Equation (1.8).

$$c(T) = \pi u = \pi\left[(1-\phi)(T-20)+d\right], \qquad \lambda = \frac{\mathrm{d}c}{\mathrm{d}T} = \pi(1-\phi) \tag{1.8}$$

At \$0.20 per kWh it is \$1.60 an hour at 50 °C and \$2.00 at 60 °C, so the optimum lies on the floor, an active constraint, and $\lambda =$ \$0.04 per kelvin and hour is what the constraint costs. This is the usual outcome of a steady-state economic optimization, and the optimum often moves from one constraint to another as prices and weather change [19, 20].

A setpoint on an active constraint cannot be held exactly. Let the draw vary by 1 kW (one standard deviation) from hour to hour. Under the draw estimator with $N = 3$ the temperature varies by 1.09 K. Placing the setpoint two standard deviations above the floor keeps the tank below 50 °C for 2.3 % of the hours, at a back-off of 2.17 K that costs \$2.09 a day. An aggressive tuning ($N = 1$, $L = 1$) varies by 1.41 K and costs \$2.71. No hourly feedback goes below 1 K, because a draw that arrives within the hour is past before the heater can answer it. Reducing variance and moving the setpoint closer to the limit is the most common way advanced control earns its keep [21].

Chapter 8 searches a grid of two setpoints every three minutes, reports which constraint binds and watches it move with the wet bulb. Chapter 10 holds a permit ceiling of 2.2 g N/m^3 inside a limit of 2.5, a back-off, and Chapter 4 chooses a sliding pressure as a steady-state decision.

> **DID YOU KNOW? The Money Is Made One Level Up**
>
> The first predictive control papers argued that the benefit comes less from steadier control than from the level above it, where steadier control lets a setpoint move closer to a limit. A main fractionator gained \$150,000 a year that way, by raising the flow of its light product [2].

1.4.8 Economic MPC and the End of the Horizon

A tracking controller holds a setpoint chosen above it. An economic controller puts the cost itself in the objective and lets the output go wherever the cost says, inside its limits [11, 22]. For the tank the objective is $\sum_k \pi_k u_k$ subject to Equation (1.1) and its limits, a linear program.

Heat bought at hour j and needed at hour n loses a fifth of its excess each hour in between, so its effective price is that of Equation (1.9).

$$\pi_{j\to n} = \frac{\pi_j}{\phi^{\,n-j}} \tag{1.9}$$

For the evening peak at 17:00 the effective prices of the hours 14, 15 and 16 are \$0.39, \$0.31 and \$0.25, all below \$0.40, while 13:00 at \$0.49 is not. The plan follows: it holds the floor through the day, heats at full power for two hours before each price rise, lets the tank coast to 71.6 °C and back to the floor and resumes heating as the stored heat runs out (Figure 1.5). It buys 229.2 kWh against 212 kWh for holding the floor and pays \$44.35 against \$46.40, 4.4 % less. Economic control spends more energy to spend less money.

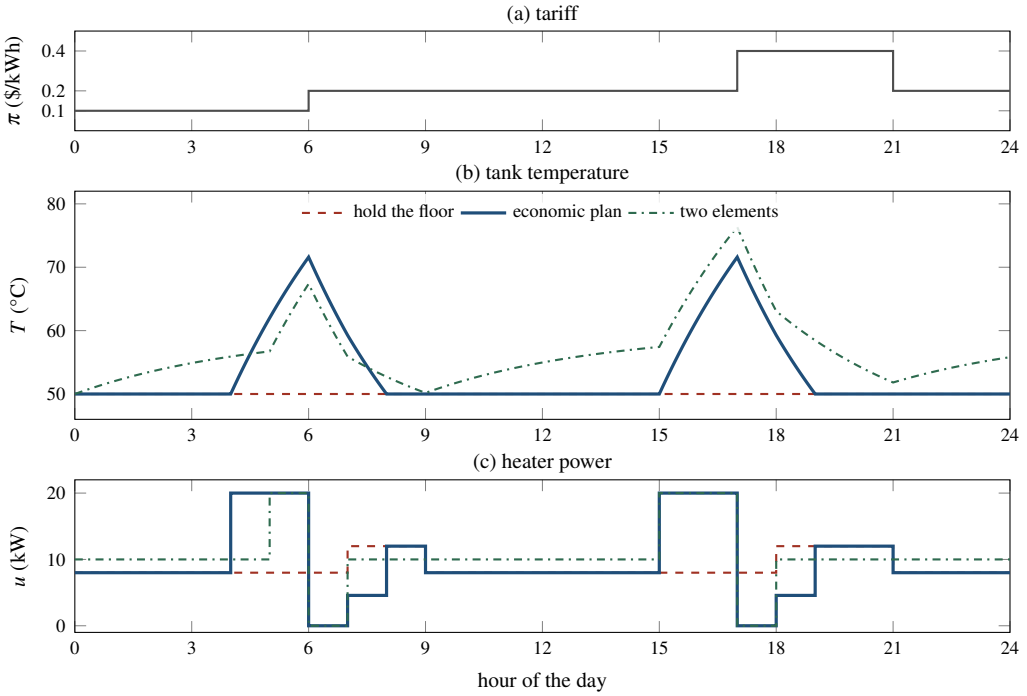


Figure 1.5 One Day of the Tank. Dashed: Holding the 50 °C Floor With a Continuous Heater. Thick Solid: the Economic Plan With a Continuous Heater. Dash-Dotted: the Best Schedule of Two 10 kW Elements. The Draw Rises From 07:00 to 09:00 and From 18:00 to 21:00.

The horizon must reach the price rise, but no further than the stored heat lasts. Run with a receding horizon, the plan pays \$46.40 a day at $N = 1$, the same as holding the floor, \$44.60 at $N = 2$, \$44.43 at $N = 3$ and the optimum from $N = 4$ on. A longer horizon buys nothing here, because the best plan never stores heat for more than four hours before using it. A terminal value on the heat left at the horizon's end can stand in for a longer horizon. Valued at its marginal worth at the floor, $\phi\,\pi$ per kelvin, it brings $N = 2$ to the optimum but costs \$45.04 at $N = 3$, because the value of stored heat falls as the tank fills. A terminal value is only as good as the price it puts on what is left.

Chapter 9 plans a day of cooling against the tariff and a demand charge and holds the slab's stored heat near its present value at the end. Chapter 6 values heat left in the tank and the pipes at the horizon's end. Chapter 8 and Chapter 10 plan the same cost their supervisors use, and Chapter 11 ends each day's plan at the level the first plan had there.

1.4.9 Optimal Control and Dynamic Programming

The factor ϕ^{n-j} of Equation (1.9) is a discrete costate. In continuous time, attaching a costate p to the tank's equation $\tau\dot{T} = 20 - T + K(u-d)$ gives the Hamiltonian of Equation (1.10).

$$H = -\pi u + \frac{p}{\tau}\left[20 - T + K\left(u - d\right)\right], \qquad \dot{p} = \frac{p}{\tau} \tag{1.10}$$

H is linear in u, so the best input sits at a bound, full power or none, or on an arc that holds a state constraint [23]. The costate grows toward the deadline, so heat is best bought late. To heat the tank from 50 to 60 °C in six hours with the least energy, the plan holds the floor at 8 kW for five hours and heats at 18 kW in the last, 58.0 kWh. A constant 10.71 kW uses 64.3 kWh. In continuous time the switch to full power comes 0.82 h before the deadline and the energy is 57.8 kWh.

Dynamic programming reaches the same plans without the calculus. It computes the least cost from every state to the end, backward in time, as shown in Equation (1.11) [5].

$$V_k(T) = \min_{0 \le u \le u_{\max}} \left[\pi_k u + V_{k+1}(T') \right], \qquad T' = 20 + \phi\,(T - 20) + u - d_k \tag{1.11}$$

On a 0.1 K grid of temperatures it prices the tank's day at \$44.348 against the linear program's \$44.346, and the slope of V_k at the floor is the costate of each hour, \$0.08, \$0.16 or \$0.32 per kelvin. A grid handles any nonlinearity and any integer input but grows with the number of states, which is why it suits the one or two states of a train or a lake.

Chapter 12 finds this structure on a train. Its Hamiltonian is linear in the force, so the best drive is full power, holding speed, coasting and full braking, and a dynamic program on a distance and speed grid places the switches. Chapter 7 computes a value function on storage by stochastic dynamic programming and uses its slope, the water value, as the terminal price of a shorter plan.

1.4.10 Integer Scheduling

Many actuators come in whole units: pumps, compressors, burners, heater elements. Suppose the tank's heater is two 10 kW elements, so $u_k \in \{0, 10, 20\}$, a mixed-integer program. Its relaxation, with $0 \le u_k \le 20$ continuous, is the economic plan of Section 1.4.8 and bounds every integer schedule from below at \$44.35. Rounding the relaxed plan to the nearest element count costs \$44.00, less than the bound, because it breaks the floor in seven hours and drops the tank to 45.1 °C. A rounded plan is not a feasible plan.

Branch and bound finds the best feasible one [6]. It solves the relaxation, picks an input with a fractional value, and splits the problem into two, one with the input rounded down and one rounded up, discarding any branch whose relaxation already costs more than the best schedule found so far. On the window from 12:00 to 20:00 it proves the best of the $3^8 = 6561$ schedules, \$22.00 against a relaxation of \$19.43, after 401 relaxations. The best day with two elements costs \$49.00 (Figure 1.5), \$4.65 above the continuous heater, because 10 kW cannot hold 8 kW and the tank runs warm. The simplest controller of whole elements, the fewest elements that keep the next hour above the floor, costs \$54.00. Scheduling the integers is worth \$5.00 a day here, more than the continuous plan saves over holding the floor.

Chapter 11 schedules two fixed pumps and a variable-speed pump hourly by branch and bound, with minimum run and rest times written as integer rows. Chapter 12 chooses among eleven discrete drive moves at each stage of its dynamic program.

1.4.11 Distributed Plans and Prices

One plan for a whole plant can be too large, or the units can belong to different owners. A distributed controller gives each unit its own plan and coordinates them [24]. Let two tanks share a feeder of 14 kW in one hour. Tank 1 wants 10 kW and serves a forgiving process, tank 2 wants 8 kW and serves a sensitive one. Each minimizes its own discomfort $\frac{1}{2} w_i (u_i^* - u_i)^2$ plus a price λ on the power it draws, with $w_1 = 0.25$ and $w_2 = 0.75$ in dollars per square kilowatt. Each answers $u_i = u_i^* - \lambda / w_i$, and the feeder raises the price while it is overloaded, as shown in Equation (1.12).

$$\lambda \leftarrow \max\Big[0,\ \lambda + \alpha\Big(\sum_i u_i - C\Big)\Big], \qquad \text{error factor } 1 - \alpha \sum_i \frac{1}{w_i} \tag{1.12}$$

The price settles at \$0.75 per kWh, and both tanks get 7 kW: the forgiving tank gives up three kilowatts, the sensitive one only one. First come, first served would give tank 2 only 4 kW and four times the total discomfort. The step size α decides the iteration. Each round multiplies the error by $1 - 5.33\,\alpha$, so $\alpha = 0.0625$ needs ten rounds to bring the overload under 0.1 kW, $\alpha = 0.1875$ needs one, and from $\alpha = 0.375$ on the price swings between two values forever. With linear costs and no quadratic term each tank takes all or nothing and no price settles.

Table 1.2 What Each Strategy Cost on One Day of the Tank (Draws Known)

Strategy	Energy (kWh)	Cost ($)	Lowest T (°C)
Hold the floor, continuous heater	212.0	46.40	50.0
Economic plan, $N \geq 4$	229.2	44.35	50.0
Economic plan, $N = 2$	216.0	44.60	50.0
Fewest elements to stay above the floor	250.0	54.00	50.0
Best schedule of two elements	250.0	49.00	50.0
Relaxed plan rounded to elements	240.0	44.00	45.1

Chapter 6 coordinates six substations and a plant by a price of heat in this way, with a quadratic term to make the price settle, and reports the gap between its bound and its plan. Chapter 5 divides one plan among three gates that exchange their plans until they agree, and measures how much of the central plan's result the division keeps.

1.4.12 Plans Under an Uncertain Forecast

A forecast is a set of possible futures, not one future [25]. On half of the days a batch draws 12 kW from 07:00 to 09:00, and the draw is seen only as it happens. The heat for 07:00 to 08:00 must be decided before then. A plan for the light morning lets the tank fall to 40 °C on a heavy one, and a plan for the mean draw to 45 °C. A scenario tree plans both futures at once, as shown in Equation (1.13).

$$\min \sum_{s} p_s \sum_{k} \pi_k \, u_{s,k} \quad \text{s.t.} \quad u_{s,k} = u_k \;\; \text{for } k < k_r, \qquad T_{s,k} \geq 50 \;\; \text{for every } s,k \tag{1.13}$$

where s indexes the scenarios with probabilities p_s and k_r is the hour from which the scenario is known. The shared decisions before k_r are the non-anticipativity constraint: the plan cannot act on what it does not yet know. The tree heats 10.6 kW in the shared hour, holds the floor on both mornings and costs $15.92 on average from midnight to noon. A controller told the morning in advance would pay $15.72, so the guarantee costs $0.20 a day. Holding a limit on only some of the scenarios, a chance constraint, trades that cost against the share of days the limit may break.

Chapter 7 plans releases on a tree of seven ensemble members with two shared days, and its chance constraint holds the channel on a declared number of them. Chapter 9 and Chapter 10 compare a plan with a forecast against one without, and the aeration plan saves energy only when it is given the typical load curve.

1.4.13 Where Prediction and Optimization Pay

Table 1.2 collects the tank's results. Prediction paid where the plan could see something a feedback loop could not: a price rise, a cleaning cycle, a morning that might be heavy. It also paid where the decision was combinatorial. It paid nothing when the horizon was longer than the stored heat lasts, and against an unmeasured draw within the hour it could do no better than feedback. A model error of 1.7 cost more than any tuning recovered.

Table 1.3 lists, for each chapter, the simpler control the predictive or optimizing layer is measured against and what that layer can exploit. Chapter 13 adds the measured results and asks where prediction or optimization paid for itself.

> **DID YOU KNOW? More Than 4600 Applications**
>
> A survey published in 2003 counted more than 4600 predictive control applications reported by vendors, over twice the number of five years before. Refining held 67 percent of the classified ones, and their size ran from a single variable to several hundred [2].

Table 1.3 The Simpler Control Each Chapter Measures Against, Its Predictive or Optimizing Layer and What That Layer Can Exploit

Ch.	Simpler control	Predictive or optimizing layer	What it can exploit
2	Altitude PID, three-deep cascade	DMC with a terminal weight	Dead time, integrating altitude, a touchdown limit
3	Two ITAE-tuned PI loops	DMC from a plunge or a traverse step test	A known response, if the test matches the regime
4	Boiler-following, turbine-following, coordinated PI	Offset-free MPC, tube MPC	Coupling of pressure and output, a pressure limit, fuel quality
5	Distant downstream PI with feedforward	Central and distributed MPC	Published orders, three coupled gates
6	Outdoor-compensated supply curve, scheduled boost	Central MPC, dual decomposition	Storage in pipes and tank, a delay set by the flow, fuel prices
7	Rule curve with forecast pre-release	Stochastic dynamic program, scenario tree	Ensemble forecast, flood pool, energy price
8	Fixed setpoints, grid-search supervisor	Economic MPC on the same cost	A moving active constraint, continuous decisions
9	Priced supervisory rule	Day-ahead economic MPC	Slab storage, tariff, demand charge, curtailment notice
10	Fixed oxygen, ammonia-based PI supervisor	Day-ahead MPC on a linearized model	An eight-hour permit mean, a typical load curve
11	Level-band pump rule	Hourly mixed-integer schedule	Tariff, tank storage, pressure floor, whole pumps
12	Flat out, one coast point	Dynamic program with a price of time	Time budget, grades, speed limits

1.5 Instruments and Final Elements in This Volume

Figure 1.6 lists what the controllers of each chapter read and what they move. Temperature, pressure and flow are each read in five chapters, and a modulating valve or a flow loop is a final element in six. The open marks set this volume apart: eight of the eleven planners read something that no instrument on the plant measures, a forecast, a price or a timetable. The quality behind a decision is seldom read at all. Only Chapter 10 reads analyzers, oxygen probes and an ammonium analyzer whose result comes two minutes late and once every five minutes, and Chapter 4 estimates its fuel quality as a disturbance on the fuel from the drum pressure.

Several of the levels are read as a pressure. A column of liquid exerts a pressure at its bottom that is set by its height and its density, whatever the width or shape of the vessel, so a pressure measured there gives the level [26]. The tank transmitter of Chapter 11 works this way, and a density that changes, as in the boiling drum of Chapter 4, changes the reading. Air and water are divided by throttling elements. In a butterfly valve a disk turns in the stream: parallel to the flow it barely obstructs it, across the flow it blocks it, and a tight shutoff needs a soft seat [26]. Butterfly valves share the air of Chapter 10 among its basins, and gates play the same part in the canal and at the dam, as Figure 1.7 shows.

The controllers sit in the layers of Figure 1.2. A survey of industrial practice draws a plant-wide economic optimization every day, a local one every hour, predictive control every minute and the basic loops of the distributed control system every second [2]. Supervisory and lower loops run with cycle times from milliseconds to minutes, and on a network spread over a wide area a control server polls remote terminal units or programmable controllers at intervals such as 5 or 60 s and sends them new setpoints [14]. The chapters span that range: the balloon plans every 2 s, the boiler every 5 s, the data hall every minute, the canal every 300 s, the heat network every 5 min, the building and the aeration basins every 15 min, the water network every hour and the reservoir once a day. Figure 1.2 names the platform of each layer.

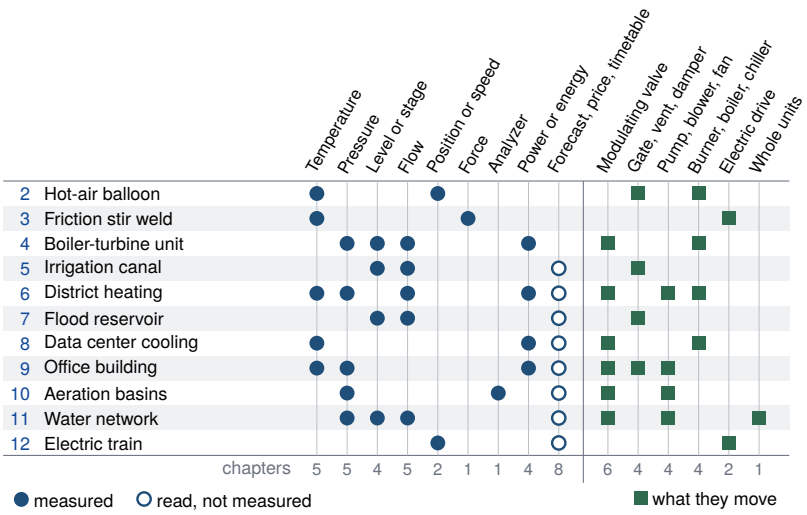


Figure 1.6 What the controllers of each chapter read (round marks) and move (square marks), from each chapter's variables table and technology section. Pressure includes differential pressure, a flow set through a flow loop or a gate counts as the valve or the gate, the analyzer column holds the oxygen probes and the ammonium analyzer, and a computed value is not marked. Notice the open marks, which are read from a forecast, a tariff or a timetable, not from the plant.

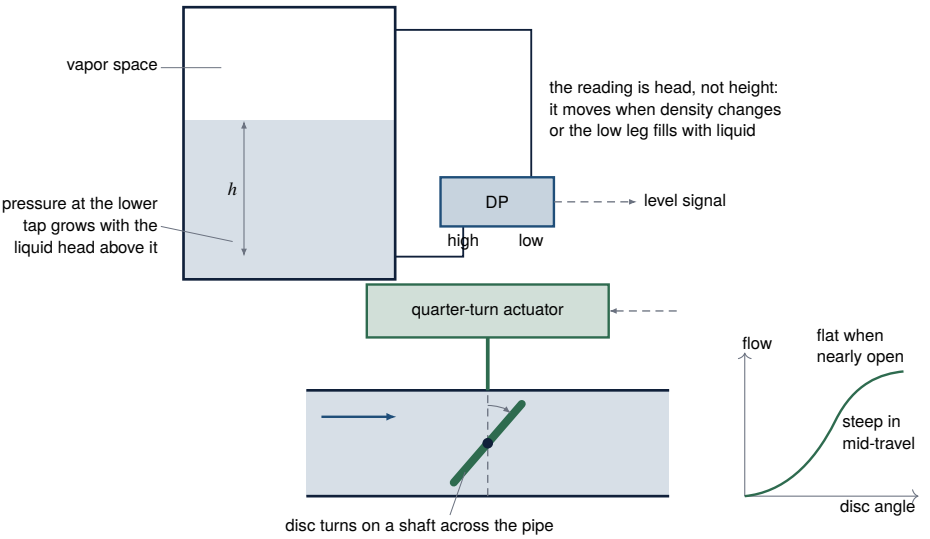


Figure 1.7 Principle of level by differential pressure (left) and of a butterfly valve with a quarter-turn actuator (right). Notice that the pressure reading is a head, so the density of the liquid sets the level it reports.

1.6 The Simulator Framework

Every chapter of this book has an app that runs in a web browser, collected with the others at https://apmonitor.com/apps/. About 240 apps exist across the series. All but a few are built on one shared framework, so the engine, the PID algorithm, the trends and the data export behave the same way in every chapter. The exceptions are standalone studios, such as the cascade studio and the two Temperature Control Lab apps, which follow the same ideas in their own code. This section describes

Table 1.4 What Every App Provides. Standalone Apps Are Described in Their Own Chapters.

Element	What it does
Run, Pause, Reset	Start, hold and restart the run. Reset keeps the reader's settings.
Speed	Simulated time per second of real time. It changes the pace and never the result.
Manual and Auto	Hand each actuator to the reader or to the controller, without a bump.
Scenario list	Numbered cases, each with a briefing, a completion rule and a reload button.
Control panel	Setpoints, manual outputs, tuning, options and disturbances, at a Beginner or an Advanced level.
Trends	SP and PV above the MV for each loop, with limits, event marks and a cursor.
Scorecard	IAE, overshoot, settling time, MV travel and time at a limit for each loop.
Learn card	Predict, Try It, two hints and Explain, opened by the reader in that order.
Show Model	Equations, live variable values, editable parameters and assumptions.
Export CSV	Every logged sample of the run as a text file with a header row.
Instructions	A separate page with background, an operating guide and the model's limitations.

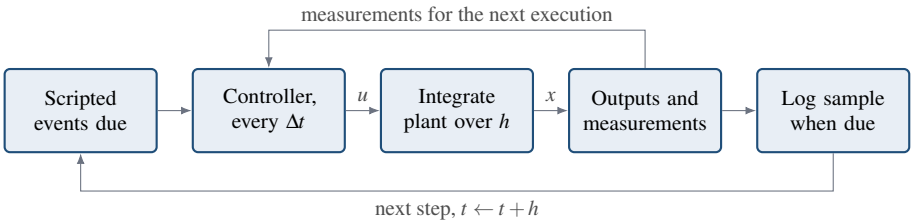


Figure 1.8 Work of the Engine in One Integration Step

the framework once and states where the standalone apps differ. Table 1.4 lists what a reader meets.

1.6.1 The Simulation Engine

Each app carries a plant model written as ordinary differential equations $\frac{dx}{dt} = f(x,u,d,p)$ in the states x, the inputs u, the disturbances d and the parameters p. The engine integrates the model with the classical fourth-order Runge–Kutta method on a fixed step h set for the fastest dynamics of the app. Some apps supply their own step, such as explicit Euler for a discontinuous right-hand side or an exact update for a model driven by events.

Three clocks run at separate rates. The plant advances every step h. The controller executes every control period Δt, equal to h unless the app declares a slower controller, and holds its outputs between executions. The log takes one sample at a fixed interval of simulated time for the trends, the scorecard and the exported file. The order of work in one step is shown in Figure 1.8. The controller also executes once when a scenario is loaded, so every run starts from an initialized controller.

Drawing the screen is kept apart from these clocks. A speed of 5 means five units of the app's time base per second of real time. Speed, frame rate and pauses change how fast a run is shown and never what it computes, so a phone and a workstation produce the same trajectory and the same file.

Noise comes from a deterministic pseudorandom generator. Its seed is declared by the app, or by a scenario that needs its own noise record, and it is restored whenever a scenario is loaded or Reset is pressed. The framework gives the reader no seed setting, so *default seed* in this book means the seed the app ships with. A run repeats bit for bit when the same scenario is loaded and the same settings are made before Run. A change made by hand during a run lands on a different step each time, which is one reason numeric answers carry a tolerance. The standalone simulators draw noise from the browser's own generator, so a noisy run there repeats in its statistics and not sample for sample.

Run and Pause start and hold the clock. Reset restarts the run at $t = 0$, paused, and keeps the reader's settings: every setpoint, tuning value, option, model parameter, disturbance slider and mode the reader changed is put back before the plant and the controller are initialized. Scripted events

replay, and a one-shot disturbance button pressed during the run is cleared. The reload button beside the scenario list restores the scenario's own settings instead.

1.6.2 The Velocity-Form PID Algorithm

One PID implementation serves the PID loops of every framework app. It is the velocity (incremental) form: each execution adds the change in the proportional and derivative terms to a running sum, and the integral is a separate term, as shown in Equations (1.14) to (1.17) for execution k and control period Δt.

$$e_k = a\,(\mathrm{SP}_k - \mathrm{PV}_k), \qquad P_k = K_c\,e_k, \qquad D_k = -a\,K_c\,\tau_D\,\frac{\mathrm{PV}^f_k - \mathrm{PV}^f_{k-1}}{\Delta t} \tag{1.14}$$

$$S_k = S_{k-1} + (P_k - P_{k-1}) + (D_k - D_{k-1}) \tag{1.15}$$

$$v_k = u_b + S_k + I_{k-1}, \qquad u_k = \mathrm{sat}(v_k) \tag{1.16}$$

$$I_k = I_{k-1} + \left[\frac{K_c}{\tau_I}\,e_k + \frac{u_k - v_k}{T_t}\right]\Delta t \tag{1.17}$$

where $a = \pm 1$ is the controller action, PV^f is the measurement after an optional first-order filter, u_b is the output bias, S is the running sum of proportional and derivative increments, v is the output before limiting, u is the output sent to the actuator, I is the integral term and T_t is the tracking time constant, equal to τ_I unless a loop sets another value. The function sat clamps v_k to the actuator range and then applies a rate limit if the loop declares one.

The limit acts on the emitted output u_k only. The sum S_k is never clamped, so it equals $P_k + D_k$ and the output before limiting is the position form $u_b + K_c e_k + D_k + I_{k-1}$. Anti-windup touches the integral alone. While the output is limited, the last term of Equation (1.17) drives I toward the value that reproduces the output applied, which is back-calculation [27]. With no integral action, Equation (1.17) is skipped. Clamping the running sum would instead discard each increment that crosses a limit and keep each one that returns, so symmetric noise would push the output off the limit even in a proportional-only loop.

The derivative acts on the measurement, so a setpoint step gives a proportional kick and no derivative kick. A change of K_c acts on the whole error at the next execution, as in the position form. The action $a = +1$ suits a process whose PV rises with the MV, a reverse-acting controller in the ISA convention, and $a = -1$ gives the direct-acting one.

In Manual the controller keeps executing and sets $I_k = u_{\mathrm{man}} - u_b - S_k$, so the output it would send equals the manual value u_{man}. The first automatic output differs from it by one period's change in P and D. In Auto the engine copies each controller output to the manual station, so a switch to Manual is also bumpless. The standalone studios with PID loops use velocity-form code of their own, with the limit on the output and anti-windup on the integral alone, in which a gain change applies only to later increments.

1.6.3 The Predictive Control and Estimator Options

More than 100 apps offer model predictive control (MPC) as an option beside the structure they open with, which stays the default. Volume 9 develops the theory. In most of them the controller plans with a small linear model in discrete time as shown in Equation (1.18).

$$x_{k+1} = A\,x_k + B\,u_k + E\,w_k + c, \qquad y_k = C\,x_k \tag{1.18}$$

where w is a measured or scheduled disturbance and c is a constant left by linearization. The model comes from a step test at the design point, from the app's balances linearized there or again at every decision, or from the model the reader fits. It never reads the true states or parameters of the run, so a parameter the reader changes in Show Model is model mismatch, as on a real plant.

At each decision the controller minimizes the objective of Equation (1.19) over a horizon of N steps.

$$J = \sum_{k=1}^{N} (y_k - r_k)^\top Q (y_k - r_k) + \sum_j \Delta u_j^\top R \Delta u_j + \sum_i \rho_i s_i^2 \tag{1.19}$$

where r is the setpoint, Q weights the tracking error, R is the move suppression on each input change Δu_j and ρ_i is the price of a slack $s_i \geq 0$ that lets an output band be broken when it cannot be held. Actuator limits are hard constraints. The quadratic program is solved in the browser by a dual active-set method [28]. The first move is applied, and the plan is made again at the next decision.

The plan starts from an estimate of the model state. About half of these apps use a steady-state Kalman filter on the model with an added disturbance state, which absorbs what the model did not predict and removes steady offset without an integral term [17]. Others correct the model output with a filtered bias to the same end. About a dozen apps also offer moving horizon estimation (MHE), which solves the estimation problem over a window of recent samples and can hold bounds that a Kalman filter cannot, such as a concentration that is never negative [29].

With the MPC selected, the tuning fields of the loops it replaces leave the Control panel. In most of these apps the Advanced level shows up to three settings, each a multiple of the shipped value: the prediction horizon, the move suppression and the band price. The trend draws the accepted plan ahead of the present time, and a list of constraint prices shows which limits bind. A plan that is not optimal, not finite or outside its limits is rejected, and a fallback, in most apps the shipped controller, holds the actuator until a plan succeeds. The Model Fitting and Tuning Studio builds the same kind of controller from the reader's own data in its tab, MPC from the fit.

1.6.4 Scenarios, Progress and Data

An app teaches through scenarios, the numbered cases at the top of the Control panel, nine to thirteen in most apps. Loading one sets the model parameters, the controller settings, the disturbances, the mode, the initial state and any scripted events. The information button beside the list opens a briefing in four parts: what the case is, why it matters, what to do and one detail for the practicing engineer. The Learn card holds the lesson back, with Predict and Try It before the run, hints on request and Explain afterward.

Each scenario has a completion rule that the app judges from the run on the simulated clock: a run length, the scripted events passed, a setting made by the reader, a state held for a stated time or a target met. Choosing a case earns nothing. A completed case is marked in the list, and a badge row shows the count and a level, Explorer after one case, Practitioner at half and Completed at all. Progress is kept in the reader's browser with no account. A certificate of progress or of completion can be issued from it, as a record of self-paced work and not an assessed credential.

One case, usually the last, is the graded challenge. It states a target, such as an error band to hold through a sequence of upsets. Changing the model parameters or the disturbances voids the attempt in most apps, and a record made with Reveal suggested tuning says so.

Show Model opens the equations, live values of the variables, the editable parameters and the assumptions. The trends plot SP and PV above the MV for each loop, with actuator limits and marks at setpoint moves, disturbances, mode changes and scripted events. The scorecard reports for each loop the integral of the absolute error (IAE), overshoot, settling time, MV travel and share of time at a limit, with the app's own measures, and it does not combine them into one score. IAE and travel accumulate from Reset, so two runs are compared at the same clock reading.

Export CSV saves the log as a text file with a header row of column names that carry their units and one row per logged sample. In most apps the time column is in seconds, minutes or hours according to the run length. The log holds the most recent 60 000 samples unless the app sets another size. Apps with the MPC option begin the file with comment lines that start with #, which a script must skip.

An app exercise in this book names the scenario by its number and name as the app shows them, every setting that differs from the scenario's defaults, the run length in simulated time, the seed

(the default seed unless stated), what to read and where (a trend, a scorecard tile or a column of the exported file) and a tolerance on a numeric answer. Settings are made before Run unless the exercise gives a time.

1.7 Working With AI Agents While Studying

A reader of this book is likely to have an AI assistant within reach, and the apps make its work easy to test. This section gives the editor's view of how to use one while working through a chapter. In short, the assistant may do the algebra and the scripting, and the reader keeps the checking.

What to delegate. Tasks with a clear input and a result that can be tested are good candidates. An assistant can derive the linearization of a chapter's balance at an operating point and the transfer function that follows from it. It can write a short script that reads the exported CSV file and fits a first-order-plus-dead-time model to a step test. It can propose a first tuning from that fit with a named rule, convert units, explain a term in a model or outline a report. The assistant knows nothing about the app until it is told. Give it the equations and parameter values from Show Model, the column names of the exported file and the scenario that produced the data, as described in Section 1.6.4. Ask it to state its assumptions and the units of every result.

What to check, and how. An assistant returns a fluent answer whether or not the answer is right, so each result needs a test that it could fail. Four tests cover most of the errors a chapter will produce.

- *Units.* Every term of an equation carries the same units, and a process gain is in PV units per MV unit. A gain computed with the valve as a fraction is 100 times the gain of a loop whose valve is in percent.
- *Signs.* Decide from the physics which way the PV moves when the MV rises. Compare that with the sign of the fitted gain and with the controller action of Section 1.6.2.
- *Steady state.* Set the derivatives to zero. The model must return the operating point that Show Model reports, and a linearization must give zero deviation there.
- *A run on the app.* Load the named scenario at the default seed, enter the proposed numbers and run. The trend and the scorecard decide. A tuning that reads well on paper and holds the valve at a limit for half the run has failed.

A fitted model gets one more test: compare it with data it was not fitted to, such as a second step in the opposite direction. When a check fails, report the failure to the assistant with the evidence and ask for the cause, then check the correction the same way.

How the exercises train the habit. One exercise in each chapter is marked *Review the agent*. It presents a tuning, a model or a claim as an AI assistant might propose it, with a flaw, and asks the reader to find the flaw in about five minutes. The flaws are ordinary ones: a unit, a sign, a steady state that does not close, a result measured on the wrong scenario or at a different run length, or a limit that was ignored. Each is found with one of the tests above. The purpose is not distrust. An engineer who can say why an answer is right can hand an assistant much larger tasks and still stand behind the result. That larger picture, in which agents do most of the routine work and the engineer supervises and checks, is the subject of Chapter 13.

1.8 Chapter Roadmap

Table 1.5 lists the chapters with the lesson each adds to this one. The level says how much of Section 1.4 a chapter assumes. The time is an estimate for reading the chapter once and running each case of its app once, at about eight minutes a case. Figure 1.2 places the same chapters in the layers of a control system.

Table 1.5 Roadmap of the Volume

Ch.	Plant	The lesson	Level	Time
2	Hot-air balloon	Step-response control of an integrating plant behind a dead time, and a landing under a speed limit.	introductory	2.5 h
3	Friction stir welder	One weld holds three plants. Which step test fills the dynamic matrix decides the result.	introductory	3 h
4	Drum boiler and turbine	Fast megawatts borrowed from pressure. Offset-free and tube MPC on a two-input unit.	advanced	2 h
5	Irrigation main canal	Delay, storage and resonance in each reach. One plan, three plans and plans that talk.	advanced	2.5 h
6	District heating network	A delay set by the flow. Heat priced among substations by dual decomposition.	advanced	2.5 h
7	Flood control reservoir	Releases decided on an ensemble forecast by a value function and a scenario tree.	advanced	2.5 h
8	Data center cooling plant	A cost optimum on a constraint that moves with the weather.	intermediate	3 h
9	Office building	A comfort band to spend, thermal mass, tariffs and a curtailment.	intermediate	3.5 h
10	Activated sludge aeration	Where a supervisory setpoint belongs under a permit that is a mean.	intermediate	3.5 h
11	Town water network	Integer pumps scheduled under a pressure floor and a tariff.	intermediate	2 h
12	Electric train	Where to coast, from optimal control and a dynamic program.	introductory	2 h

Chapters 2, 3 and 12 form a first course: two single-input predictive controllers and one optimal plan whose structure can be checked by hand. They rest on Sections 1.4.2 to 1.4.5 and 1.4.9. Chapters 8 to 11 are best read after Sections 1.4.6 to 1.4.10, in any order. Chapters 4, 5, 6 and 7 draw on Sections 1.4.11 and 1.4.12 as well and are best left until last. A reader who knows MPC from industrial practice can start at Section 1.4.7 and take Chapters 8, 9, 11 and 7.

KEY POINTS

- A predictive controller plans several moves on a model, applies the first and plans again at the next period.
- A longer horizon or a higher price on moves buys robustness at the price of speed, and neither corrects a wrong gain.
- An offset disappears only when the prediction learns about the disturbance, through a bias or an estimated input.
- Most of the money is made in the layer above: where the plant sits against its active limit, and when stored energy is bought.

Where else this applies. Steady-state optimization above the loops in Volume 13, Chapter 11, and a buoy planned on a fading wave forecast in Volume 17, Chapter 5.

References

[1] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[2] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, 2003.

[3] M. Morari and J. H. Lee, "Model predictive control: Past, present and future," *Computers & Chemical Engineering*, vol. 23, no. 4–5, pp. 667–682, 1999.

[4] E. F. Camacho and C. Bordons, *Model Predictive Control*, ser. Advanced Textbooks in Control and Signal Processing. London: Springer, 2007.

[5] R. Bellman, "The theory of dynamic programming," *Bulletin of the American Mathematical Society*, vol. 60, no. 6, pp. 503–515, 1954.

[6] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960.

[7] J. Richalet, A. Rault, J. L. Testud, and J. Papon, "Model predictive heuristic control: Applications to industrial processes," *Automatica*, vol. 14, no. 5, pp. 413–428, 1978.

[8] K. Ichikawa, "Application of optimization theory for bounded state variable problems to the operation of train," *Bulletin of JSME*, vol. 11, no. 47, pp. 857–865, 1968.

[9] R. D. Bell and K. J. Åström, "Dynamic models for boiler-turbine-alternator units: Data logs and parameter estimation for a 160 MW unit," Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, Research Report TFRT-3192, 1987. [Online]. Available: https://lup.lub.lu.se/record/8602296

[10] P. Xu, P. Haves, M. A. Piette, and J. Braun, "Peak demand reduction from pre-cooling with zone temperature reset in an office building," Lawrence Berkeley National Laboratory, Berkeley, CA, Tech. Rep. LBNL-55800, 2004, in Proceedings of the ACEEE 2004 Summer Study on Energy Efficiency in Buildings. OSTI 838609.

[11] J. B. Rawlings, D. Angeli, and C. N. Bates, "Fundamentals of economic model predictive control," in *Proc. 51st IEEE Conference on Decision and Control*, 2012, pp. 3851–3861.

[12] Lake Mendocino FIRO Steering Committee, "Lake Mendocino forecast informed reservoir operations final viability assessment," UC San Diego, Tech. Rep., Dec. 2020. [Online]. Available: https://cw3e.ucsd.edu/FIRO_docs/LakeMendocino_FIRO_FVA.pdf

[13] C. E. García and A. M. Morshedi, "Quadratic programming solution of dynamic matrix control (QDMC)," *Chemical Engineering Communications*, vol. 46, no. 1–3, pp. 73–87, 1986.

[14] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, S. Lightman, A. Hahn, S. Saravia, A. Sherule, and M. Thompson, "Guide to operational technology (OT) security," National Institute of Standards and Technology, Gaithersburg, MD, Special Publication 800-82r3, 2023.

[15] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[16] K. R. Muske and T. A. Badgwell, "Disturbance modeling for offset-free linear model predictive control," *Journal of Process Control*, vol. 12, no. 5, pp. 617–632, 2002.

[17] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE Journal*, vol. 49, no. 2, pp. 426–437, 2003.

[18] P. O. M. Scokaert and J. B. Rawlings, "Feasibility issues in linear model predictive control," *AIChE Journal*, vol. 45, no. 8, pp. 1649–1659, 1999.

[19] M. L. Darby, M. Nikolaou, J. Jones, and D. Nicholson, "RTO: An overview and assessment of current practice," *Journal of Process Control*, vol. 21, no. 6, pp. 874–884, 2011.

[20] S. Skogestad, "Plantwide control: The search for the self-optimizing control structure," *Journal of Process Control*, vol. 10, no. 5, pp. 487–507, 2000.

[21] M. Bauer and I. K. Craig, "Economic assessment of advanced process control—a survey and framework," *Journal of Process Control*, vol. 18, no. 1, pp. 2–18, 2008.

[22] M. Ellis, H. Durand, and P. D. Christofides, "A tutorial review of economic model predictive control methods," *Journal of Process Control*, vol. 24, no. 8, pp. 1156–1178, 2014.

[23] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: Optimization, Estimation, and Control*. Routledge, 2018.

[24] R. Scattolini, "Architectures for distributed and hierarchical model predictive control—a review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[25] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[26] T. R. Kuphaldt, *Lessons in Industrial Instrumentation*, version 2.33 ed. Open textbook, Creative Commons Attribution 4.0 license, ibiblio.org/kuphaldt/socratic/sinst, 2024.

[27] K. J. Åström and L. Rundqwist, "Integrator windup and how to avoid it," in *Proc. 1989 American Control Conference*, 1989, pp. 1693–1698.

[28] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical Programming*, vol. 27, no. 1, pp. 1–33, 1983.

[29] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations," *IEEE Transactions on Automatic Control*, vol. 48, no. 2, pp. 246–258, 2003.

1.9 Exercises

1.1 Theory and calculation · 4 min

The tank of Equation (1.1) keeps $\phi = 0.8$ of its excess over the room each hour and gains one kelvin per kilowatt-hour.

(a) Find the step-response coefficient g_4. *Checkpoint:* 2.952 K/kW.

(b) A one-move DMC with $N = 2$ and $R = 0$ sees a setpoint step of 5 K. Find its first move. *Checkpoint:* 3.30 kW.

(c) Find the limit of the first move as N grows without bound. *Checkpoint:* 1 kW.

1.2 Concept check · 2 min

A one-move DMC was built from a step test on the tank, but the heater on the plant has 1.7 times the gain of the model. Which statement matches Table 1.1?

(a) a horizon of one hour removes the overshoot, because the plan reaches the setpoint in one period

(b) move suppression removes the overshoot, because it shrinks every move in proportion to the gain error

(c) a longer horizon keeps the loop stable but leaves an overshoot, because the first move is sized by the model's gain

(d) the output bias removes the overshoot at once, because it adds the measured error to the prediction

1.3 Theory and calculation · 4 min

The tank's draw rises from 2 to 4 kW and is not measured. A controller with $N = 1$ predicts from the measured temperature with the nominal draw of 2 kW and no disturbance model, and holds $r = 50\,^{\circ}\mathrm{C}$.

(a) Write the heater power it chooses as a function of the measured temperature T. *Checkpoint:* $u = 48 - 0.8\,T$.

(b) Find the temperature at which the tank settles. *Checkpoint:* 48 °C.

1.4 Theory and calculation · 3 min

The insulated tank ($\phi = 1$) has $g_i = i$. Its draw is 1 kW above the model, so the open-loop model drifts from the plant by 1 K an hour, and a one-move DMC with a constant output bias holds 50 °C. Use Equation (1.6) to find the steady offset for $N = 3$, and say how it changes with the horizon.

1.5 Concept check · 2 min

A 24-hour economic plan for the tank at \$0.20 per kWh softens the floor of 50 °C with a linear slack price ν per kelvin and hour. Which value of ν is the smallest of the four that keeps the floor wherever it can be held?

(a) \$0.21, a little above the price of one kilowatt-hour of heat

(b) \$0.04, the steady-state multiplier $\pi(1-\phi)$ of the floor

(c) \$0.05, a quarter above the steady-state multiplier of the floor

(d) \$0.01, since any positive linear price makes a soft limit exact

1.6 Theory and calculation · 4 min

The tank's setpoint is backed off from the 50 °C floor by two standard deviations of its temperature, at \$0.20 per kWh.

(a) Find the cost of one kelvin of back-off per day from Equation (1.8). *Checkpoint:* \$0.96 per kelvin and day.

(b) The draw estimator with $N = 3$ holds the temperature to a standard deviation of 1.086 K. Find the back-off and its cost per day. *Checkpoint:* 2.17 K, \$2.09 a day.

1.7 Theory and calculation · 4 min

The tank needs heat at 07:00, when power costs \$0.20 per kWh. Earlier hours cost \$0.10 up to 06:00 and \$0.20 from 06:00. Use Equation (1.9) to find the effective price of heat bought at 03:00, 04:00 and 05:00 for use at 07:00, and the earliest of these hours worth buying in.

1.8 Theory and calculation · 3 min

The tank is heated from 50 to 60 °C in six hours with the least energy, with the draw at 2 kW and the floor at 50 °C. The best plan holds the floor and heats once, at the end. Find the power in the last hour and the energy saved against the constant power of 10.71 kW.

1.9 Concept check · 2 min

The relaxed economic plan for the tank costs \$44.35 a day. Rounded to whole 10 kW elements it costs \$44.00, and the best schedule of whole elements costs \$49.00. What explains this?

(a) the rounded plan found a cheaper schedule that branch and bound missed, because it stopped early

(b) the relaxation is not a lower bound when the integer schedule must end at the same temperature

(c) whole elements are cheaper than a continuous heater, because they run at full efficiency

(d) the rounded plan breaks the floor in seven hours, so it is not a feasible schedule at all

1.10 Theory and calculation · 4 min

Two tanks share a feeder of 14 kW. They want 10 and 8 kW and weigh their shortfall with $w_1 = 0.25$ and $w_2 = 0.75$, so each answers a price λ with $u_i = u_i^{*} - \lambda / w_i$.

(a) Find the price at which the feeder is exactly loaded, and each tank's power. *Checkpoint:* \$0.75 per kWh, 7 and 7 kW.

(b) Find the step size α of Equation (1.12) that settles the price in one round, and the step size at which it swings forever. *Checkpoint:* 0.1875 and 0.375.