Chapter 4 – Just Enough Modelling: Physics-Based, Data-Driven, and Hybrid Models

4.0 Before you start

Where we are. Part I is behind you. Chapter 1 established that a twin is bought to improve a decision, and that fidelity is a requirement derived from a value metric rather than a virtue pursued for itself. Chapter 2 drew the model/shadow/twin boundary and defined a *model* as a description of behaviour that does not itself run. Chapter 3 drew the model as a box in an architecture and called it "a versioned artefact with an interface".

Three times now this book has said "a model" and moved on. Part II opens the box.

What Part II is for, and what it is not for. Part II is your interface to the modelling experts. It is enough to ask good questions and evaluate the answers. It is not enough to build models, and it is not trying to be. You will finish this chapter unable to derive a heat-transfer model for a turbine blade, and able to sit across from someone who can and tell whether their model is fit for the decision your twin has to make. Those are different skills, and only one of them is yours.

What you are assumed to know. Everything Part I assumed. Beyond that: school-level physics (things are conserved; rates add up), and the ability to read a difference equation, which is a loop body. **No calculus is assumed in working order.** Exactly one derivative appears in this chapter, in Sec. 4.3; it is explained in words before it is written down, used once, and converted immediately into arithmetic you could implement. No integrals, no matrices, no probability notation.

What this chapter covers. What a model is made of; three families (physics-based, data-driven, hybrid) derived and compared on the *same* problem – the demonstrator's pot; the white-to-black-box scale; validity envelopes and how each family fails outside its own; a decision procedure for choosing; and five questions to ask a modelling expert.

What this chapter deliberately does not cover. How to *fit* the parameters – that is calibration, Chapter 7. Whether to *believe* the fitted model – verification and validation, also Chapter 7. How a model is *executed* – Chapters 5 and 6. Machine-learning method detail and its specific risks – Chapter 8. State estimation – Chapter 6. Uncertainty quantification – named where it matters, owned by Chapter 7. Discrete-event models, state machines and the other formalisms – named in Sec. 4.10 so you recognise them, taught in Chapters 5 and 6.

Learning objectives

By the end of this chapter, you will be able to:

1. **Identify** the state, inputs, parameters, outputs and assumptions of a proposed model, and **distinguish** its structure from its fitted parameters.
2. **Derive** a simple physics-based model from a conservation statement, and **enumerate** the assumptions the derivation made.
3. **Compare** physics-based, data-driven and hybrid models on the same problem, and **predict** what each one demands in data, expertise and maintenance.

4. **State** a model's validity envelope and **predict** how it fails outside it, distinguishing silent failure from loud failure.
5. **Conduct** a fidelity conversation with a modelling expert: ask the five questions of Sec. 4.9 and **evaluate** the answers you get.

4.1 Why a software engineer needs this

You are not going to write the model. Here is why you nonetheless cannot treat it as somebody else's box.

Because you own its failure modes. Chapter 3 put the model behind an interface, and interfaces hide implementation, not consequences. A model that returns a plausible number under conditions it was never valid for returns a `200 OK`. Nothing in your monitoring will fire. If that number feeds Chapter 2's command path, your system waters a pot that did not need it, and the incident review will ask you what the model assumed. "The modelling team owns that" is a true sentence that will not survive the meeting.

Because you are the one who notices the deployment mismatch. The modeller optimises for accuracy. You are the person who knows the model has to run every ten minutes on a machine that also runs the ingest loop, that the input it wants is sampled at a different rate than it assumes, and that its parameters were fitted on a pot that has since been repotted. Those are not modelling questions. They are integration questions, and nobody else in the room is looking at them.

Because the vocabulary gap is where projects fail. Chapter 2 quoted a practitioner survey in which respondents observed that colleagues "mean simulation when referring to DTs". The same confusion runs one level down: "the model" means a set of equations to one person, a trained network to another, and a Computer-Aided Design (CAD) file to a third. Reviews of twin content in manufacturing find models grouped into geometric, physical, behaviour and rule types [1] – four different things, one word.

And because the literature agrees this is where the substance is. A chapter-length treatment of modelling for twins opens by saying that at the heart of a twin sit one or more models standing for the real system, that such systems present many facets, and that many types of model are available, each with different strengths [2]. Three main classes of approach are consistently reported – model-based, data-driven, and hybrid [3]. Those three are this chapter.

What "just enough" means, concretely. By the end you should be able to say, out loud, in a design review: *"That is a physics-based model with two fitted parameters, its validity envelope is one watering interval, and outside that it fails silently. What is our plan for detecting that?"* That sentence is the deliverable.

4.2 What a model is made of

Chapter 3 said a model is something that takes a starting state, inputs and a time horizon, and returns a trajectory. That is the interface. Here are the parts behind it. Every model

you meet has all five, whether or not anyone has written them down – Figure 4.1 names them in the order the rest of this section takes them.

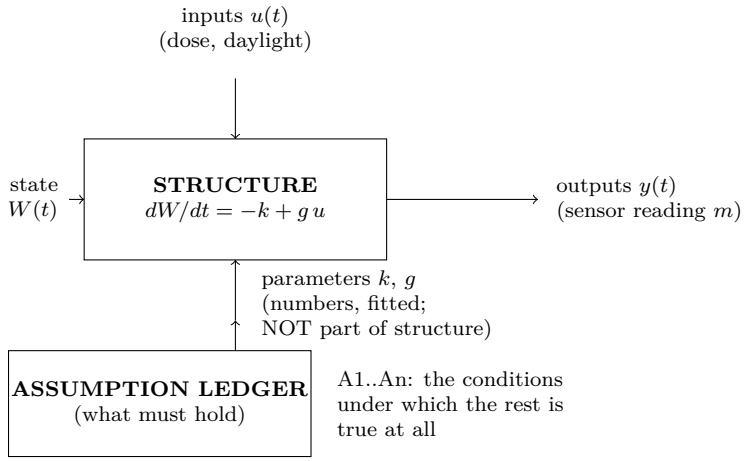


Figure 4.1 The five parts of any model. Structure and parameters are separated deliberately: changing a number is calibration, changing the equation is a new model.

The separation the figure insists on – structure on one side, parameters on the other – is the one that saves arguments later. "The model is wrong" is two completely different problems depending on which box it lands in, and Sec. 4.2.3 is about why.

4.2.1 State

State is the smallest set of quantities that, together with future inputs, determines the model's future behaviour.

For the pot, the state is one number: `W`, the amount of water held in the soil, in millilitres.

Two things are worth noticing immediately, because both come up in review.

State is a choice, not a fact. A more elaborate model of the same pot might have three state variables – water in the upper soil, water in the lower soil, water in the plant – and be more accurate and more expensive. The pot did not change. The model's ambition did.

State is what the model represents, and therefore also a statement of what it does not. Our one-number state cannot represent water sitting on the surface, unabsorbed. That is not a bug; it is a documented limit, and Sec. 4.3.4 will list it.

4.2.2 Inputs and outputs

Inputs are things from outside that affect the model and that the model does not explain. For the pot: the watering dose `d`, in millilitres, and – if we choose to include them – air temperature, humidity and light.

Outputs are what the model predicts and what can be compared against a measurement. For the pot, the output is `m`, the moisture *reading* the sensor produces.

The distinction between `W` and `m` is the one beginners collapse, and it matters. `W` is water. `m` is a number a capacitive sensor reports. They are related, and they are not the same quantity, and the relationship between them is itself a modelling assumption (Sec. 4.3.3). Any twin whose model outputs a quantity that no sensor reports has, somewhere, a hidden assumption connecting the two. Finding it is a productive question in a review.

4.2.3 Parameters, and why they are separate from structure

A **parameter** is a number that is constant over the timescale of interest and is fitted rather than derived. For the pot: `k` and `g`.

The **model structure** is the form of the relationship with no numbers in it at all – "the reading falls at a steady rate and steps up when water arrives".

This split is the thing Chapter 3's model registry existed to manage, and now you can see why it deserved its own component:

- **Structure is shared.** Every pot on the bench uses the same structure.
- **Parameters are specific.** `k` and `g` belong to *this* pot, with this soil, this plant, this drainage.
- **They change on different schedules.** The structure changes when someone decides the model is wrong. The parameters change when the plant grows, the soil compacts, or the pot is repotted.

Chapter 3 warned that losing track of which parameters go with which pot produces a twin that is confidently wrong while every component works. Now you can see the mechanism: the structure is still right, so nothing looks broken.

4.2.4 The assumption ledger

Every model is a set of decisions to ignore things. The **assumption ledger** is the written list of those decisions. It is not standard terminology; the practice is standard, and giving it a name makes it askable-for.

An assumption ledger has one row per assumption, and each row answers three questions: *what did we assume, why is it reasonable, and what happens when it stops being true?* Sec. 4.3.4 builds one.

Why insist on this? Because a model's validity envelope (Sec. 4.7) is nothing more than the intersection of the conditions under which its assumptions hold. A model with no written assumptions has no statable envelope, which means nobody can say when to stop believing it.

4.2.5 What a model is not

Two adjacent things get called "the model" and are not.

It is not the estimator. A model says how the pot behaves. An **estimator** combines the model with noisy measurements to say what the pot's state currently is. Chapter 3 drew them as separate components; Chapter 6 supplies the estimator's mechanism. When someone says "the model tracks the moisture", ask which of the two they mean.

It is not the simulator. Chapter 2 settled this: the simulator executes the model. Chapters 5 and 6 own it.

4.3 Family 1: physics-based, derived from scratch

Now we build the model that Chapters 2 and 3 have been using on credit.

A **physics-based model** gets its structure from a first principle – a conservation law, or another statement that holds independently of this particular pot. Its parameters are few, and they usually mean something you could measure another way.

4.3.1 Start from a conservation statement

The first principle here is that water does not appear or vanish:

> The water in the pot changes at a rate equal to what goes in minus what comes out.

That is the whole physics. In symbols, writing `dW/dt` for "the rate at which `W` changes with time" – this is the one derivative in the chapter, and it means exactly what the words say:

```
dW/dt = (water in) - (water out)
```

Water in arrives in doses from the pump. Water out leaves by evaporation from the soil and transpiration through the plant, which we lump into one outflow because our one-number state cannot tell them apart.

4.3.2 Make it specific, one assumption at a time

Between waterings there is no inflow, so:

```
dW/dt = -(outflow rate)
```

To go further we must say something about outflow, and there is no way to do that without an assumption. Take the simplest one:

> **Assumption A1.** Over the interval we care about, the outflow rate is constant. Call it `e`, in millilitres per hour.

Then the water falls in a straight line:

```
W(t) = W0 - e * t
```

where `W0` is the water at the start and `t` is hours elapsed.

Was A1 reasonable? Partly, and the honest treatment of "partly" is the most useful thing in this section. A drier soil gives up water more slowly, and a wilting plant transpires less, so outflow really falls as `W` falls. A better assumption is that outflow is proportional to the water present:

```
dW/dt = -W / tau
```

which gives a curve that decays and flattens rather than a straight line.

Here is the payoff. For times short compared with `tau`, that curve and the straight line are the same to within a small error – the straight line is the curve's first-order approximation near the start. So **A1 is not a different model; it is the good model, restricted to short intervals.**

That single observation is the chapter in miniature. The linear model is not "wrong", and it is not "right". It is *correct within an envelope*, and the envelope is "a time short compared with `tau`". Chapter 4's whole job is to make you the person in the room who asks what the envelope is.

For the demonstrator, `tau` is on the order of days and the interval we reason over is one watering cycle, about eight hours. Short compared with days. A1 holds. Use the straight line.

4.3.3 Get from water to a sensor reading

The model predicts `W`. The sensor reports `m`. Bridging them needs another assumption:

> **Assumption A2.** Over the operating range, the reading is proportional to the water present, plus an offset: `m = a * W + b`.

Real capacitive soil sensors are not exactly proportional across their whole range, and they are close enough over a band in the middle. Since `a` and `b` are constants, we can fold them in and work directly in reading units, which is what Chapters 2 and 3 have been doing:

```
m(t) = m0 - k * t          between waterings
```

with **`k` in reading units per hour**. One parameter, and it absorbs both the evaporation rate and the sensor's scale. Note what we have quietly bought: we never have to know `a`, `b` or `e` separately, because the decision only needs `m`.

Now the dose. When the pump delivers `d` millilitres:

> **Assumption A3.** All of the delivered water reaches the sensed region of soil, and mixes through it quickly compared with the sampling interval.

Then the reading steps up by an amount proportional to the dose:

```
m -> m + g * (d / 100)     at a watering
```

with **`g` in reading units per 100 ml**. Two parameters, `k` and `g`, which is exactly the model Chapters 2 and 3 committed to.

4.3.4 The assumption ledger

Writing it down, in the shape Sec. 4.2.4 specified.

#	Assumption	Why it is reasonable	What happens when it breaks
A1	Outflow rate is constant over the interval	The interval (about 8 h) is short compared with the drying time constant (days)	Over long dry spells the model predicts drying faster than reality; the reading appears to level off "unexpectedly"
A1b	The same rate `k` applies by day and by night	Keeps the model to one parameter	A plant transpires far less in the dark, so a single `k` over-predicts overnight drying and under-predicts daytime drying. Chapter 2 wrote `k` as a rate "per hour of daylight", anticipating exactly this refinement
A2	Reading is affine in water content	True over the middle of a capacitive sensor's range	Near saturation or bone-dry, the sensor flattens; a real change produces no reading change, and the model sees a fault that is not there
A3	Delivered water reaches the sensed soil and mixes fast	Small pot, dripper positioned over the root zone	Channelling down the side of the pot, or a dripper knocked aside, produces a small step – **which is indistinguishable from the fault the twin is looking for**
A4	Parameters are constant over weeks	Soil and plant change slowly	After repotting or rapid growth, `k` and `g` are wrong and every prediction drifts
A5	One number describes the pot's water	The decision needs only the sensed region	Surface pooling, or a dry lower layer under wet upper soil, is invisible to the model

Three rows deserve a second look.

A3 is a design-limiting assumption, not a caveat. Chapter 1's whole value case was detecting that a dose failed to arrive. A3 says that a dose which arrives but *misses the sensor* looks identical. So the twin cannot distinguish "pump failed" from "dripper displaced". Is that acceptable? For Chapter 1's value metric – experiment-weeks lost to undetected watering faults – yes, because both are faults and both want a human to walk over. Notice how the answer came from the value metric, not from the model.

A2 explains a fault report you will otherwise chase for a week. A pot watered to saturation shows no step, because the sensor is already at the top of its range. The twin alerts. Nothing is wrong. The alert is correct behaviour from a model used outside its envelope.

A1b is how a model grows a parameter. Splitting `k` into a daytime rate and a night-time rate turns two parameters into three and makes the model better. It also makes it more expensive to fit, more expensive to keep, and one row longer in Chapter 3's registry. Whether to do it is not a modelling question – it is Chapter 1's question, asked again: does the extra fidelity change the decision? For a fault detector comparing an expected step against an observed one, it does not, and one `k` is the right answer. For a twin scheduling doses to a target moisture, it might. Same pot, same physics, different model, because a different decision.

4.3.5 A numeric instance

Abstract structure becomes usable when you put numbers in it. These are illustrative; the real ones come from fitting, which is Chapter 7.

Suppose that over one dry night, twelve hours with no watering, the reading falls from 640 to 592. Then:

```
k = (640 - 592) / 12 = 4 reading units per hour
```

And suppose a 100 ml dose takes the reading from 592 back to 640:

```
g = 48 reading units per 100 ml
```

Recall from Chapter 1 that the shipped schedule doses 100 ml at 09:16 and 120 ml at 17:05. Predict the reading just before the evening dose, starting from 640 immediately after the morning one. The gap is 7 hours 49 minutes, call it 7.8 hours:

```
m = 640 - 4 * 7.8 = 640 - 31.2 = 608.8
```

And the expected step from the 120 ml evening dose:

```
g * (120 / 100) = 48 * 1.2 = 57.6 reading units
```

Now the diagnostic. If the evening dose produces a step of 55, the model is working. If it produces a step of 3, water is not arriving. That comparison – expected step against observed step – is the residual Chapter 1 built its entire value case on, and you have now derived it from the statement that water is conserved.

How much did this cost? One conservation statement, five assumptions, two parameters, and about a page. That is the appeal of physics-based models, and it is not a small thing: the whole model fits in a code review.

4.4 Family 2: the same job, data-driven

Now solve the same problem without knowing any of that.

A **data-driven model** takes a generic structure – one that could describe almost anything – and gets its behaviour by fitting to observed data. You supply examples of inputs and outputs; the fitting procedure supplies the relationship. In twin practice these are described as models that reproduce the input-output relationship from sampled data, of which the simplest example is a linear regression [2], and they are reached for exactly when the underlying physics or principles are not fully available [4].

4.4.1 What it looks like here

Take every ten-minute sample from the last three months. For each one, build a row:

```
inputs :  m[n], dose delivered since last sample, air temperature,
          humidity, light level, hour of day
target :  m[n+1]
```

Fit any function you like to predict the target from the inputs. The result predicts the next reading, which is what the diagnostic needs – compare predicted with observed, alert on a gap.

Notice what we did *not* need: any idea what a capacitive sensor measures, any notion of evaporation, any decision about linear versus exponential decay. That is the appeal, and it is real.

4.4.2 What it buys

It captures effects you did not think of. Our physics model ignores air temperature, humidity and light. They plainly matter – a hot bright day dries a pot faster. The physics model would need a new derivation to include them. The data-driven model needs a new column.

It does not require a modelling expert. It requires a different expert, which is often easier to find.

It handles messiness the physics does not describe. The plant grows. The soil settles. A learned model refitted monthly tracks that drift without anyone deriving anything.

4.4.3 What it costs, and the cost that decides this case

It needs data covering the conditions you care about. A model fitted over a mild spring predicts July badly, and it will not tell you so. This is **extrapolation**, and it is the characteristic failure of the family.

Its parameters mean nothing. In the physics model, `k = 4` is inspectable by a human: is four reading units per hour plausible for this plant? Someone can answer. A learned model's coefficients afford no such check. Chapter 7 will need something to check, and this family offers less of it.

It cannot easily be interrogated about its assumptions. Sec. 4.3.4's ledger has no counterpart. The assumptions are still there – they are in the choice of inputs and in what the training data happened to contain – but they are implicit, which is precisely what the ledger exists to prevent.

And the one that decides this particular case. Chapter 1's value metric is detecting watering faults. The training data is three months of historical operation – during which, by Chapter 1's own baseline, *three watering faults occurred and went undetected for four days each.* A model fitted to that data has been taught that a dose producing no moisture step is a normal thing that happens sometimes.

A data-driven model trained on unlabelled history learns the faults as normal. That is not a subtle statistical point; it is a direct consequence of what fitting does. The physics model is immune, because its expectation of a moisture step comes from a conservation law rather than from a record of what usually happened.

This is the strongest single argument in the chapter, and it generalises: **a model used to detect abnormality should not be taught what is normal by data that contains the abnormality.**

4.5 Family 3: hybrid

The third family combines the two, and it is where a great deal of current practice sits. The argument for it is that both pure approaches have shortfalls, and that a combined framework aims at removing them [5]. Reviews of specific domains report the same trend: relying solely on data-driven models for precise prediction is difficult, physics-based modelling is often needed to aid the predictive models, and the result is a rising use of hybrid models [6].

"Hybrid" covers at least three distinct things. Distinguishing them is useful, because they solve different problems and only one of them fits our pot.

4.5.1 Corrected physics

Keep the physics model. Add a learned term that predicts its *error*.

```
m[n+1] = physics(m[n], dose)  +  learned_correction(temperature, humidity, light)
```

The physics carries the part we understand – water is conserved, a dose produces a step – and the learned term absorbs what the derivation ignored. Published work does exactly this shape: combining analytical white-box friction models with data-driven black-box models against live data, because the friction depends on load, temperature and time in ways that are hard to model analytically [7].

Why this one fits the pot. It keeps `g`, the expected step per 100 ml, derived from conservation rather than learned from history – so Sec. 4.4.3's fatal objection does not apply. And it lets temperature and light improve the decay prediction, which is where the physics model is weakest. The interpretable parameter survives; the messiness gets absorbed.

4.5.2 Constrained learning

Fit a data-driven model, but penalise it during fitting for producing predictions that violate physics. The common realisation adds a physics-informed term to the loss function, which penalises predictions that do not comply with first principles and so constrains training toward physically compliant solutions – reported as the commonest route to a hybrid physics-and-machine-learning model [8].

Why not here. It is heavier machinery than a two-parameter problem justifies, and it needs someone who can express the constraint properly. Its place is where the learned model is doing most of the work and you need to stop it inventing impossible behaviour.

4.5.3 Surrogates

Fit a fast model to the outputs of a slow one. Surrogate and reduced-order models sit at the intersection of high-fidelity physics simulation and data-driven modelling, and they trade some accuracy for much greater computational speed [5]. The motivation is that traditional large-scale physics-based models are often intractable in real-time or many-query settings, so time is invested offline to build a cheap approximation that can be used online [9]. The same reasoning drives model order reduction for twins meant to be executed in deployment [10], and healthcare work describes an AI model used as a surrogate of a knowledge-driven model, giving comparable predictions far more cheaply to compute [11].

The distinction to hold on to. A surrogate is fitted to *a model*, not to reality. It therefore inherits every one of that model's assumptions, and adds approximation error of its own. It can never be more accurate than what it was fitted to. A surrogate solves a *speed* problem, not an *accuracy* problem – and that is why it belongs to Chapter 6, where speed becomes a design constraint.

Why not here. Our physics model evaluates in nanoseconds. There is nothing to accelerate. Recording that is worthwhile: reaching for a surrogate when the base model is already fast is a common and expensive misreading of what surrogates are for.

4.6 The white-to-black-box scale

You will meet a second vocabulary for the same territory, and you should be able to move between them.

The box metaphor is a systems-engineering framing: a box takes inputs, performs operations, and produces outputs, and the colour describes how much of the operation you can see [12].

- **White box.** The internals are visible and derived from known principles. Our Sec. 4.3 model.
- **Black box.** Only inputs and outputs are visible; the internals are fitted. Our Sec. 4.4 model.
- **Grey box.** Some of each. Our Sec. 4.5 model.

The scale is used widely across twin-adjacent fields. Structural health monitoring explains

the twin concept through white, grey and black boxes [13], and offshore predictive modelling describes grey-box work as sharpening what physics-based white-box models can predict, by adding machine-learning components where the physics is not fully understood [14]. Modelling techniques get placed on an explicit white-to-black scale when assembling a twin's overall model set [3]. The same words are reused for model order reduction, where black-box approaches need no knowledge of the underlying system, grey-box approaches need some, and white-box approaches need full access [10] – a reminder to check which sense is meant.

Figure 4.2 places this chapter's three families on that scale, and places the caution that follows on a second axis, because the two are independent.

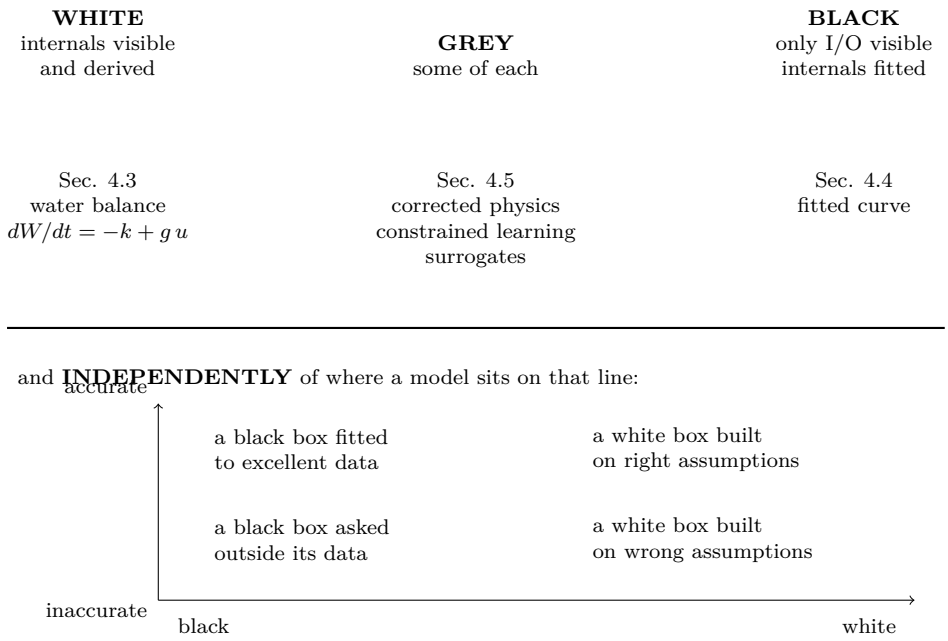


Figure 4.2 Colour is one axis; being right is another. All four corners of the lower box are occupied by real models.

The one caution, which is the second axis of Figure 4.2. Colour describes *visibility of internals*, not *accuracy*, *trustworthiness* or *quality*. A white-box model built on wrong assumptions is wrong and inspectable. A black-box model fitted to excellent data covering every condition you care about can be more accurate than anything you could derive. The scale tells you what you can inspect and therefore what kind of argument you can make about the model – which matters enormously for Chapter 7, and is not the same as telling you which model is better.

4.7 Validity envelopes, and how each family fails

This is the section Chapter 1 was pointing at.

> **Validity envelope.** The range of conditions over which a model's assumptions hold and its predictions may be believed.

The principle is stated bluntly in the literature: every use of a model, twins included, depends critically on the model being applied only inside the range where it is valid

[15]. And the companion point, which is Chapter 1's fidelity principle arriving from the modelling side: the appropriate level of fidelity is not automatically the highest one available; it follows from the use case [16].

For our pot, the envelope reads straight off the assumption ledger:

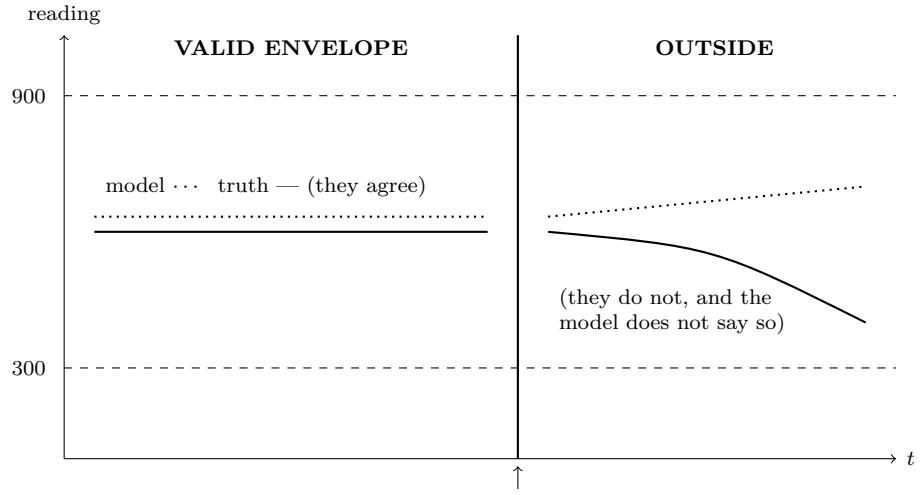
> Valid for intervals up to about a day, with readings in the middle of the sensor's range, in a pot that has not been repotted since the parameters were fitted, with the dripper positioned over the sensed soil.

Every clause of that came from a row of Sec. 4.3.4's table. That is what the ledger was for.

4.7.1 The failure modes differ by family, and this is the useful part

Family	How it fails outside its envelope	Can you tell?
Physics-based	Predictions drift in a direction the assumptions predict – e.g. A1 breaking makes it dry too fast	Often yes: the error has a recognisable shape, and the ledger tells you which assumption produces it
Data-driven	Extrapolation: confident output in a region with no training data	Usually no, unless the model reports its own uncertainty or you check input coverage
Hybrid (corrected physics)	The physics part degrades predictably; the correction extrapolates	Partly: the physics half stays inspectable, which is a real advantage
Surrogate	Inherits the base model's failures, plus its own approximation error	Only against the base model, which is the thing you were avoiding running

Figure 4.3 is why that table matters more than it looks. Outside the envelope the model does not stop; it keeps returning numbers that look exactly like the ones inside it.



The fix is not a better model. It is a runtime check the CALLER owns:

```
if not (300 <= reading <= 900):
  refuse, loudly
```

Figure 4.3 Silent failure. A model that crashes outside its envelope is a good model with a bad interface; one that keeps answering is the actual hazard.

Silent failure is the hazard. A model that crashes outside its envelope is a good model with a bad interface. A model that returns 608.8 when the truth is 400 is the problem, because 608.8 is a plausible number and nothing downstream can tell. Chapter 3's guard exists partly for this, and it can only refuse on conditions someone stated.

The engineering response, and it is yours, not the modeller's: make the envelope a runtime check. If the model is valid for readings between 300 and 900, then the code that calls it refuses – loudly – outside that band. You now know enough to ask for the numbers that check needs, and asking is Sec. 4.9.

4.7.2 What this chapter is not claiming

Stating an envelope is not the same as establishing one. Sec. 4.3.4's rows are *claims* about where the assumptions hold, and claims need evidence. Getting that evidence – fitting the parameters, testing the model against held-out reality, quantifying how wrong it might be – is calibration, validation and uncertainty quantification, and all three belong to Chapter 7. This chapter gives you the vocabulary to demand them.

4.8 Choosing a family

A decision procedure. Run it in order; the first clear answer wins.

1. Is there a usable first principle, and does the decision depend on few quantities? If yes, start physics-based. Cheap, inspectable, small, and it extrapolates in a way you can reason about. This is the pot.

2. Is the model being asked to distinguish normal from abnormal, using history that may contain the abnormal? If yes, the expectation must not be learned from that history. Physics-based, or hybrid with the physics carrying the expectation. Sec. 4.4.3.

3. Is the phenomenon one nobody can write down, with plenty of data covering the conditions you care about? If yes, data-driven is the sensible answer and the honest one. Do not derive equations for the sake of looking rigorous.

4. Do you have a principle for part of it and not the rest? Hybrid, corrected-physics form. This is the most common real answer, and it is why the hybrid trend exists [5], [6].

5. Do you have an accurate model that is too slow for how often you must run it? Surrogate – and note this is a *speed* question, which means you already had a model, and it is Chapter 6's territory.

A cross-check to run afterwards. Whatever you chose, ask: what will it cost to *keep*? Chapter 1 named model maintenance as the largest forgotten cost and Chapter 14 owns it. A physics model is re-fitted when the physical twin changes. A data-driven model may need re-fitting whenever conditions drift, which is a standing operational commitment, and a model that changes weekly is a versioned artefact changing weekly underneath a system that may be commanding hardware (Chapter 3, Sec. 3.2.4).

4.9 Five questions to ask a modelling expert

Chapter 1 said Part II would let you hold up your end of a fidelity conversation. This is it. Five questions, what a good answer sounds like, and what a worrying answer sounds like.

Q1. What is the model's state, and what does it deliberately not represent?

Good: "One variable, the water in the sensed region. It does not represent surface pooling or layering." *Worrying:* "It models the pot." An answer that does not name what is left out means the exclusions have not been thought about, and the exclusions are where the surprises live.

Q2. Which parameters are fitted to this specific physical twin, and when were they last fitted?

Good: "Two, `k` and `g`, fitted for this pot on 3 March, over a two-week window." *Worrying:* "They came with the model." That means the parameters belong to somebody else's pot. This is Chapter 3's binding problem arriving in conversation.

Q3. What is the validity envelope?

Good: a list with numbers and units, traceable to assumptions. *Worrying:* "It is accurate to about five per cent." That is a single accuracy figure, which is not an envelope – accurate to five per cent *under what conditions?* Push once, politely. If a numeric envelope does not exist, you have learned something important: nobody has written the assumptions down, and Sec. 4.7's runtime check cannot be specified.

Q4. How does it fail outside the envelope – loudly or silently?

Good: "It drifts high, and it drifts in a way you can spot because the error grows with elapsed time since the last watering." *Worrying:* "It should not be used outside the envelope." That is a statement about intent, not behaviour. Your system will use it outside the envelope one day, because a sensor will fail or a plant will be repotted. Ask what it *does*.

Q5. What measurement would change your mind about this model?

Good: a specific answer – "if a dose produced a step more than twenty per cent off the predicted one across several pots, the structure is wrong, not the parameters." *Worrying:* no answer. A model that no observation could count against cannot be validated, and Chapter 7 will have nothing to work with.

A note on tone, which matters more than the questions. These are not challenges, and asking them badly will cost you the working relationship that the rest of the project depends on. The framing that works is the one Chapter 1 gave you: *the decision the twin has to improve*. "The decision we have to support is whether a dose landed, within eight hours. What does the model need to be true for that to work?" You are not auditing their competence. You are bringing them the only thing they cannot get without you, which is what the software will actually do with their answer.

4.10 Other formalisms you will meet

The chapter has treated one kind of model: a continuous quantity changing over time. That is the right first case and it is not the only one. Named here so you recognise them, and handed onward.

- **Geometric models** – shape and space. Chapter 2 already declined to call these twins on their own.
- **Behaviour models** – state machines and discrete-event descriptions, for systems whose interesting dynamics are transitions rather than quantities. A pump is either on or off; a batch is queued, running or done. Reviews group twin content into geometric, physical, behaviour and rule models for exactly this reason [1], [17], and practical modelling stacks place continuous time, discrete time, state machines and discrete-event models on one scale [3].
- **Rule models** – explicit logic, from operating procedures or regulations. Ordinary software, and routinely forgotten to be part of the model at all – which means it also escapes the versioning of Chapter 3's registry and the assumption ledger of Sec. 4.2.4.
- **Hybrid formalisms** – for systems that are both, such as our pot with its continuous drying and its discrete watering events. Handling discrete and continuous behaviour together in one principled execution is a named problem with named machinery, and it is Chapter 6's [2], [18].

Notice that our pot model is already mildly hybrid in this second sense: the drying is continuous, the doses are discrete events. We got away with treating it informally because the events are rare and simple. Chapter 6 explains what to do when they are not.

4.11 Faded example: the offshore turbine

Chapter 3 designed components for the turbine anomaly service: vibration and temperature stream to shore, a service compares them against a model of that turbine, a work order goes to a planner. Chapter 3 also noted that the quantity of interest – accumulated fatigue, or bearing condition – is not measured by any sensor, but inferred.

Worked, for the first two steps.

State. Not the vibration. Vibration is an *output*, a thing you measure. The state is the turbine's condition – crack length, bearing wear – which no sensor reports. This is the `W` versus `m` distinction of Sec. 4.2.2, and it is much starker here: in the pot, the reading tracked the state closely. Here the state must be inferred from a signal that relates to it indirectly.

Which family? Run Sec. 4.8. Step 1: is there a usable first principle? Partly – fatigue accumulation under load has well-developed theory, so the physics is available for part of the problem. Step 4 is therefore the likely landing point: hybrid, with physics carrying fatigue accumulation and a learned component carrying the map from vibration spectra to condition, which nobody can write down. That is the shape the offshore literature describes as grey-box: physics-based prediction enhanced with machine-learning components where the physics is not fully understood [14].

Now it is your turn.

- **(a)** Write three rows of the assumption ledger for the physics half. For each, say what happens operationally when it breaks – not what happens mathematically.
- **(b)** The learned half is fitted on data from a fleet of turbines, not this one. Name the two distinct risks that creates, using this chapter's vocabulary. One of them is a Chapter 3 concept, not a Chapter 4 one.
- **(c)** State a validity envelope for the combined model, with units. Then say what the runtime check of Sec. 4.7.1 would look like, and which component of Chapter 3's architecture would perform it.
- **(d)** Ask Q4 of Sec. 4.9 about this model and answer it yourself. Does it fail loudly or silently, and does your answer differ for the two halves?

Hint for (b): one risk is about the conditions the fleet data covers. The other is about which physical twin the fitted parameters belong to, and Chapter 3 has a word for it.

4.12 Posed problem: the vendor's model card

A vendor supplies a twin for a hospital's chilled-water plant – the same one Chapter 3 reviewed. Asked about the model, they say:

> "It is a deep-learning model trained on two years of plant data. It predicts chilled-water supply temperature fifteen minutes ahead with a mean absolute error of 0.3 degrees. It retrains nightly on the last ninety days."

Produce a one-page model assessment. It must:

1. Classify the model by family and by box colour, and state which of Sec. 4.8's steps their choice corresponds to – or say that it corresponds to none.

2. Ask the five questions of Sec. 4.9 and, for each, say what you would accept as an answer and what you would escalate on.
3. Identify the one property of their description that should worry you most, given that the plant's value case is *detecting when a chiller is degrading*. Justify it from Sec. 4.4.3.
4. Recommend either accepting the model, accepting it with a stated condition, or proposing a different family – and defend the recommendation on the value metric, not on modelling taste.

There is no single right assessment. There is a wrong one, and it is the assessment that argues about the 0.3 degrees.

4.13 Summary

Against the objectives.

1. **Every model has five parts:** state, inputs, parameters, outputs and assumptions (Sec. 4.2). Structure is shared across physical twins and parameters belong to one – which is what Chapter 3's model registry existed to keep straight, and why losing the binding produces a twin that is confidently wrong while everything works.
2. **A physics-based model comes from a first principle plus a stack of assumptions** (Sec. 4.3). We derived Part I's two-parameter water balance from "water is conserved", and found that its linear form is the good exponential model restricted to short intervals – so the model is neither wrong nor right, but *correct within an envelope*.
3. **Three families, compared on one problem** (Secs. 4.3 to 4.5). Physics-based: small, inspectable, few parameters, needs a principle. Data-driven: no principle needed, captures the unmodelled, but extrapolates silently and – decisively for our case – **learns the faults as normal when trained on unlabelled history**. Hybrid: three distinct forms, of which corrected-physics fits our pot, constrained learning is heavier machinery, and surrogates solve a speed problem rather than an accuracy one.
4. **The validity envelope is the intersection of the assumptions** (Sec. 4.7), it reads straight off the ledger, and the failure modes differ by family. Silent failure is the hazard, and the engineering response – checking the envelope at runtime – is yours rather than the modeller's.
5. **Five questions** (Sec. 4.9) give you the fidelity conversation Chapter 1 promised. The framing that makes them land is the decision the twin has to improve, not the model's quality.

The sentence to carry forward: **a model is not right or wrong, it is correct within an envelope** – and its companion, **a model used to detect abnormality must not learn what is normal from data containing the abnormality.**

4.14 Exercises

Objectives in brackets. Solutions and hints follow.

4.14.1 (Objective 1, easy). For a model that predicts a lithium battery pack's voltage from current draw and temperature, classify each as state, input, parameter or output: (a) current draw; (b) internal resistance; (c) state of charge; (d) terminal voltage; (e) ambient temperature.

4.14.2 (Objective 1, easy). Sec. 4.2.3 says structure is shared and parameters are specific. Give one example from any system you have worked on where something you treated as a constant in code was really a parameter belonging to one specific deployment. What went wrong, or what would have?

4.14.3 (Objective 2, easy). Using `k = 4` and `g = 48` from Sec. 4.3.5, predict the reading at 08:00 the next morning, starting from 640 just after the 17:05 dose of 120 ml. State which assumption in the ledger you are leaning on hardest, and why.

4.14.4 (Objective 4, medium). Assumption A2 says the sensor is affine over its middle range. Design the runtime check of Sec. 4.7.1 for it: what does the calling code test, what does it do when the test fails, and what does it tell the operator? Then say why "clamp the reading to the valid range and carry on" is the wrong answer.

4.14.5 (Objective 3, medium). Sec. 4.4.3 argues a data-driven model trained on unlabelled history learns the faults as normal. Suppose you have three months of history *and* a maintenance log recording exactly when the three faults occurred. Does that fix the problem? Answer carefully – the honest answer is "partly", and the interesting part is what remains.

4.14.6 (Objective 3, medium). For each, choose a family using Sec. 4.8 and name the step that decided it: (a) predicting queueing time at a hospital scanner from arrival patterns; (b) predicting stress in a bracket under a known load; (c) predicting when a paint line's finish quality will drop, where nobody can say what causes it and there are five years of inspection records; (d) running a thermal model of a data hall two thousand times to choose fan speeds, where one run takes four minutes.

4.14.7 (Objective 5, medium). Complete Sec. 4.11, parts (a) through (d).

4.14.8 (Objectives 2 and 4, hard). Derive a model for the *water reservoir* that feeds the demonstrator's pumps: the tank level falls by the delivered dose at each watering and is refilled by a human occasionally. Write the structure, name the parameters, and produce an assumption ledger of at least four rows. Then answer: could this model detect the "empty reservoir" fault from Chapter 1's baseline, and how does that compare with detecting it from the pot's moisture?

4.14.9 (Objective 5, hard). Take a model you or a colleague have shipped – any predictive component at all, including a heuristic. Write its assumption ledger retrospectively. Then state its validity envelope, and say whether the running system checks it.

4.14.10 (Objectives 1 through 5, hard, open-ended). Do Sec. 4.12, the vendor's model card, in full.

Solutions and hints

4.14.1. (a) input; (b) parameter – it is fitted, roughly constant over the timescale of a discharge, and it drifts over the pack's life, which is exactly the profile of Sec. 4.2.3; (c) state; (d) output; (e) input. Worth noticing: internal resistance is the row that would be

a *state* variable in an ageing model spanning years. Whether something is a parameter or a state depends on the timescale the model is for, which is the same purpose-dependence as everything else in this chapter.

4.14.2. Open. Assessed on whether the example distinguishes a genuine constant of nature from a fitted number that happened to be typed in. Strong answers name a timeout, a threshold or a calibration factor that was right for the first customer.

4.14.3. From 640 at 17:05 to 08:00 is 14.9 hours, so `640 - 4 * 14.9 = 640 - 59.6 = 580.4`. The assumption under most strain is **A1**: fifteen hours is no longer short compared with a drying time constant of a couple of days, so the true reading will be a little *higher* than 580.4, because real drying slows as the soil dries. A very good answer also notes that this bias is in a known direction, which is Sec. 4.7.1's point about physics-based models failing in a shape you can predict.

4.14.4. *Hint:* the check is a band test on the input reading, and the action is refusal plus an alert naming the reason. On the second half: a clamp produces a plausible number from an implausible one, which is exactly Sec. 4.7's silent failure, manufactured deliberately. The reading being out of range is information – it means the sensor is saturated or broken, and that is worth telling somebody.

4.14.5. *Partial.* It helps: you can exclude the fault windows from training, so the model no longer learns those three events as normal. What remains is the harder half. First, faults that were never detected are not in the log, so the excluded set is the faults you *found*, and Chapter 1's whole premise is that discovery took four days. Second, the four days *before* each discovery are the fault, and the log records the discovery date, not the onset – so your exclusion windows are wrong at the boundary, in the direction that leaves fault data in the training set. Third, even with perfect labels you have three positive examples, which is not enough to learn a fault signature from. Good answers reach at least the second point.

4.14.6. (a) Data-driven, step 3 – arrival patterns are behavioural and nobody has a first principle for them; note this is also the case where a discrete-event model (Sec. 4.10) is the real answer, so credit an answer that says so. (b) Physics-based, step 1 – structural mechanics is exactly the case the family was made for. (c) Data-driven, step 3, and five years of records is the condition that makes it defensible; watch for the Sec. 4.4.3 trap, and ask whether the inspection records label the drop or merely contain it. (d) Surrogate, step 5 – two thousand runs at four minutes is 133 hours, so the question is speed and you already have the model.

4.14.7. *Partial.* (a) Rows worth having: loading history is representative; the fatigue law's parameters suit this material and weld detail; damage accumulates independently of sequence. The operational consequence of the last one breaking is the interesting one – an unusual storm sequence makes the model's remaining-life estimate optimistic, and nothing in the data says so. (b) Risk one: the fleet's conditions may not cover this turbine's – extrapolation (Sec. 4.4.3). Risk two: **binding** (Chapter 3, Sec. 3.3.1) – fleet-fitted parameters are not this turbine's, and Sec. 4.9's Q2 is the question that surfaces it. (c) and (d) are yours; for (d), the expected answer is that the halves differ, which is the practical argument for the corrected-physics form of Sec. 4.5.1.

4.14.8. *Hint:* the structure is a running total – level decreases by each delivered

dose, increases by refills. Parameters: the tank's capacity, and the relationship between commanded dose and delivered volume, which is the pump calibration Chapter 1 quoted. Ledger rows worth having: every commanded dose is actually delivered (which is the very thing the twin doubts – note the circularity and say so); no leaks; refills are recorded; calibration is current. On the comparison: the reservoir model detects *empty reservoir* directly and early, but is blind to a blocked tube, because the water leaves the tank either way. The pot's moisture model detects both, later. The strongest answers conclude that the two together localise the fault, which neither does alone – and that this is an argument for a second model rather than a better one.

4.14.9. Open. Assessed on whether the envelope has numbers and units, and on the honesty of the last part. "No, nothing checks it" is a complete and common answer, and noticing that is the exercise.

4.14.10. *Hint:* no solution, but a test. The property that should worry you most is the nightly retraining on the last ninety days, given a value case of *detecting degradation*. A model that continuously relearns what recent operation looks like will absorb a slow degradation into its notion of normal and stop flagging it – Sec. 4.4.3's failure with a moving window attached. If your assessment argues about the 0.3 degrees, you have reviewed the model's accuracy instead of its fitness.

4.15 Where to go next

In this book. Chapter 5 takes the other half of Chapter 2's deferred vocabulary: what a simulation actually is, and what state and time mean once a model has to be executed. Chapter 6 supplies the simulators, the state estimator's mechanism, and the machinery for models that mix continuous and discrete behaviour (Sec. 4.10). Chapter 7 is where everything this chapter *claimed* gets established: fitting the parameters, testing the model, and quantifying how wrong it might be. Chapter 8 takes the data-driven family and its specific risks seriously. Chapter 14 is where the maintenance cost of Sec. 4.8's cross-check comes due.

In the literature.

- *Modelling for twins, the closest single reference to this chapter:* [2] introduces physics-based, data-driven and discrete formalisms together and works them on an incubator example; [19] is the containing book.
- *The three-family split and the hybrid argument:* [5] is the clearest statement of why neither pure approach suffices and what combining them buys, with [3] for the model-based / data-driven / hybrid classification, [6] for the hybrid trend in one domain, and [1] for the first-principles / data-driven / hybrid split applied to real manufacturing twins.
- *Hybrid mechanics:* [8] on physics-informed loss functions as the common realisation, [7] for a worked white-box-plus-black-box combination against live data, and [20] with [21] for multi-fidelity and surrogate-assisted approaches in civil and structural settings.
- *Surrogates and reduced-order models:* [9] on investing offline computation to make online use tractable, [10] on model order reduction for twins meant to be executed in deployment, and [11] for the surrogate-of-a-knowledge-model framing in health.

- *Validity and fidelity:* [15] on models being usable only within their validity range, and [16] for the statement that the appropriate fidelity is not the highest feasible – the modelling-side version of Chapter 1's principle.
- *The box scale:* [12] and [22] for the systems-engineering framing, [13] and [14] for its use in structural and offshore work.
- *Other formalisms, ahead of Chapters 5 and 6:* [18] on discrete-event abstraction and statecharts, [17] for the geometric / physical / behavioural / rule grouping, [23] for learned automata as twin models, and [24] on co-simulation, which is where multiple models meet.
- *Consulted, not drawn on above:* [25] on uncertainty and optimisation (Chapter 7), [26] on the open research gaps including model error and identifiability, [27] on models that learn their own structure, [28] on choosing between model types inside a systems engineering process, [29] on the practical difficulty of covering corner cases when the physical process is a black box, [30] for representation types inside a maintenance architecture, [31] on transferring models between similar physical twins, [4] and [32] for industrial model portfolios, and [33] on what happens when models from different twins must be integrated (Chapter 15).

In the demonstrator. Pull two weeks of moisture readings and watering events through the endpoints of Chapter 3, plot them, and estimate `k` and `g` by eye from the slopes and the steps. You are not calibrating – that is Chapter 7 – you are checking whether the *structure* of Sec. 4.3 looks like the data at all. If the decay is visibly curved over your window, A1 is already under strain, and you have learned something about the envelope before writing a line of the twin.

4.16 References

[1] T. Böttjer *et al.*, "A review of unit level digital twin applications in the manufacturing industry," *CIRP Journal of Manufacturing Science and Technology*, vol. 45, pp. 162–189, Oct. 2023, doi: 10.1016/j.cirpj.2023.06.011.

[2] G. Abbiati, C. Gomes, M. Sandberg, Z. Kazemi, S. T. Hansen, and P. G. Larsen, "Modelling for Digital Twins," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 89–127. doi: 10.1007/978-3-031-66719-0_5.

[3] N. Anwer, R. Stark, F. Tao, and J. Erkoyuncu, "Developing and leveraging digital twins in engineering design," *CIRP Annals*, vol. 2025, Jun. 2025, doi: 10.1016/j.cirp.2025.05.002.

[4] Y. Jiang, S. Yin, K. Li, H. Luo, and O. Kaynak, "Industrial applications of digital twins," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2207, p. 20200360, Aug. 2021, doi: 10.1098/rsta.2020.0360.

[5] A. Rasheed, O. San, and T. Kvamsdal, "Digital Twin: Values, Challenges and Enablers From a Modeling Perspective," *IEEE Access*, vol. 8, pp. 21980–22012, 2020, doi: 10.1109/ACCESS.2020.2970143.

[6] Z. Chen *et al.*, "Service oriented digital twin for additive manufacturing process," *Journal of Manufacturing Systems*, vol. 74, pp. 762–776, Jun. 2024, doi: 10.1016/j.jmsy.2024.04.015.

[7] M. Heithoff, M. Trinh, J. Michael, B. Rumpe, and C. Brecher, "A Digital Shadow for Accurate Robot Motion Control: Integrating Data with Friction Models," Grand Rapids, USA, Jul. 2025. doi: 10.5283/epub.77667.

[8] A. Thelen *et al.*, "A comprehensive review of digital twin — part 1: Modeling and twinning enabling technologies," *Structural and Multidisciplinary Optimization*, vol. 65, no. 12, p. 354, Nov. 2022, doi: 10.1007/s00158-022-03425-4.

[9] M. G. Kapteyn, D. J. Knezevic, and K. Willcox, "Toward predictive digital twins via component-based reduced-order models and interpretable machine learning," in *AIAA Scitech 2020 Forum*, in AIAA SciTech Forum., American Institute of Aeronautics; Astronautics, 2020. doi: 10.2514/6.2020-0418.

[10] D. Hartmann and H. Van der Auweraer, *The Executable Digital Twin: Merging the digital and the physics worlds*. 2022.

[11] M. Viceconti, M. De Vos, S. Mellone, and L. Geris, "Position Paper From the Digital Twins in Healthcare to the Virtual Human Twin: A Moon-Shot Project for Digital Health Research," *IEEE Journal of Biomedical and Health Informatics*, vol. 28, no. 1, pp. 491–501, Jan. 2024, doi: 10.1109/JBHI.2023.3323688.

[12] M. Grieves, "Digital Twins and Their Role in Reengineering Engineering Education," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 237–261. doi: 10.1007/978-3-031-67778-6_11.

[13] H. Pezeshki, H. Adeli, D. Pavlou, and S. C. Siriwardane, "State of the art in structural health monitoring of offshore and marine structures," *Proceedings of the Institution of Civil Engineers - Maritime Engineering*, vol. 176, no. 2, pp. 89–108, Apr. 2023, doi: 10.1680/jmaen.2022.027.

[14] U. T. Tygesen, K. Worden, T. Rogers, G. Manson, and E. J. Cross, "State-of-the-Art and Future Directions for Predictive Modelling of Offshore Structure Dynamics Using Machine Learning," in *Dynamics of Civil Structures, Volume 2*, S. Pakzad, Ed., Cham: Springer International Publishing, 2019, pp. 223–233. doi: 10.1007/978-3-319-74421-6_30.

[15] G. Lugaresi and H. Vangheluwe, *From Digital Twins to Twinning Systems*. 2025.

[16] Z. Ali, R. Biglari, J. Denil, J. Mertens, M. Poursoltan, and M. K. Traoré, "From modeling and simulation to Digital Twin: Evolution or revolution?" *SIMULATION*, vol. 100, no. 7, pp. 751–769, Jul. 2024, doi: 10.1177/00375497241234680.

[17] H. Zheng, T. Liu, J. Liu, and B. Jinsong, "Visual analytics for digital twins: A conceptual framework and case study," *Journal of Intelligent Manufacturing*, vol. 35, pp. 1–16, May 2023, doi: 10.1007/s10845-023-02135-y.

[18] P. Carreira, V. Amaral, and H. Vangheluwe, Eds., *Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems*. Springer Nature, 2020. doi: 10.1007/978-3-030-43946-0.

[19] J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., *The Engineering of Digital Twins*. Cham: Springer International Publishing, 2024. doi: 10.1007/978-3-031-66719-0.

[20] M. Torzoni, M. Tezzele, S. Mariani, A. Manzoni, and K. E. Willcox, "A digital twin framework for civil engineering structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 418, p. 116584, Jan. 2024, doi: 10.1016/j.cma.2023.116584.

[21] M. Torzoni, A. Manzoni, and S. Mariani, "A Deep Neural Network, Multi-fidelity Surrogate Model Approach for Bayesian Model Updating in SHM," in *European Workshop on Structural Health Monitoring*, P. Rizzo and A. Milazzo, Eds., Cham: Springer International Publishing, 2023, pp. 1076–1086. doi: 10.1007/978-3-031-07258-1_108.

[22] S. Sabri, K. Alexandridis, and N. Lee, Eds., *Digital Twin: Fundamentals and Applications*. Cham: Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-67778-6.

[23] Q. Xu, "Traversing the Data Spectrum: Path to Dependable Cyber-Physical Systems through Digital Twins," Doctoral thesis, 2023. Accessed: Apr. 05, 2024. [Online]. Available: https://www.duo.uio.no/handle/10852/106058

[24] G. Schweiger *et al.*, "An empirical survey on co-simulation: Promising standards, challenges and research needs," *Simulation Modelling Practice and Theory*, vol. 95, pp. 148–163, Sep. 2019, doi: 10.1016/j.simpat.2019.05.001.

[25] A. Thelen *et al.*, "A comprehensive review of digital twin—part 2: Roles of uncertainty quantification and optimization, a battery digital twin, and perspectives," *Structural and Multidisciplinary Optimization*, vol. 66, no. 1, p. 1, Dec. 2022, doi: 10.1007/s00158-022-03410-x.

[26] *Foundational Research Gaps and Future Directions for Digital Twins*. Washington, D.C.: National Academies Press, 2024. doi: 10.17226/26894.

[27] F. Emmert-Streib, H. Cherifi, K. Kaski, S. Kauffman, and O. Yli-Harja, "Complexity data science: A spin-off from digital twins," *PNAS Nexus*, vol. 3, no. 11, p. pgae456, Nov. 2024, doi: 10.1093/pnasnexus/pgae456.

[28] R. Honcak and A. Wooley, "An MBSE approach for Virtual Verification & Validation of Systems with Digital Twins," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*, Linz Austria: ACM, Sep. 2024, pp. 390–400. doi: 10.1145/3652620.3688252.

[29] R. Gu, T. Barbuceanu, N. Xiong, and T. Seceleanu, "Experiences in Building a Digital Twin Framework: Challenges and Possible Solutions," in *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, Jul. 2024, pp. 531–536. doi: 10.1109/COMPSAC61105.2024.00078.

[30] R. van Dinter, B. Tekinerdogan, and C. Catal, "Reference architecture for digital twin-based predictive maintenance systems," *Computers & Industrial Engineering*, vol. 177, p. 109099, Mar. 2023, doi: 10.1016/j.cie.2023.109099.

[31] P. Gardner, L. A. Bull, J. Gosliga, N. Dervilis, and K. Worden, "Foundations of population-based SHM, Part III: Heterogeneous populations – Mapping and transfer," *Mechanical Systems and Signal Processing*, vol. 149, p. 107142, Feb. 2021, doi: 10.1016/j.ymssp.2020.107142.

[32] S. Mihai *et al.*, "Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2255–2291, 2022, doi: 10.1109/COMST.2022.3208773.

[33] B. Combemale *et al.*, "On the Challenges of Integrating Digital Twins," in *2nd International Conference on Engineering Digital Twins (EDTconf 2025)*, 2025. Accessed: Sep. 29, 2025. [Online]. Available: https://inria.hal.science/hal-05221809/