Forecasting Demand, Sales, and Campaign Performance

From Customers to Calendars

Dr. Jose Mendoza, Academic Director and Clinical Associate Professor
Version 1.0 · July 2026
Except where otherwise noted, this chapter is licensed under CC BY 4.0.

Chapter Information

ABSTRACT

This chapter develops forecasting as the time-indexed branch of the predictive discipline built in Chapters 8 and 9, with StyleCraft's holiday open-to-buy plan as its running decision. It establishes the time series as a data structure whose order carries information, uses decomposition to read trend, seasonality, cycle, and noise, and separates recurring calendar effects from structural breaks. It builds the forecasting ladder from the naive and seasonal-naive opponents through the exponential smoothing family, then rebuilds honest evaluation for ordered data: rolling-origin selection, a sealed holdout that resembles the decision, MAPE with its denominator failure, and accuracy reported by horizon. The forecast widens into a distribution through simulated paths and scenarios, and the human hand is disciplined by the written override. The chapter closes with managerial interpretation, industry practice, the ethics of the laundered guess, and exercises.

KEYWORDS

forecasting; time series; seasonality; structural break; seasonal-naive; exponential smoothing; MAPE; prediction interval; forecast horizon; judgmental adjustment

VERSION AND DATE

Version 1.0 · July 2026 · Language: English (United States)

SUGGESTED CITATION

Mendoza, J. (2026). Forecasting demand, sales, and campaign performance. In *Applied business analytics for marketing decision-making: Business analytics and data visualization* (Chapter 10, Version 1.0) [Open educational resource]. CC BY 4.0.

LICENSE AND RIGHTS

Copyright © 2026 Jose Mendoza. Except where otherwise noted, this work is licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0). You may share and adapt this material for any purpose, provided appropriate credit is given. Third-party trademarks, screenshots, figures, and other materials remain subject to their respective rights and licenses.

Google Colab is a product of Google LLC. "Python" and the Python logos are trademarks or registered trademarks of the Python Software Foundation. pandas, NumPy, Matplotlib, and scikit-learn are sponsored or affiliated projects of NumFOCUS, a 501(c)(3) nonprofit charity in the United States. statsmodels is a community-developed project distributed under the modified BSD license. ChatGPT is a product of OpenAI, Claude of Anthropic, Gemini and NotebookLM of Google LLC, and GitHub Copilot

of GitHub, Inc. Product names are used for identification only and do not imply endorsement. StyleCraft Collective is a fictional company created for instruction.

COMPANION REPOSITORY
Datasets, notebooks, and figure sources for this chapter: Applied Business Analytics companion repository on GitHub

Chapter Learning Objectives

By the end of this chapter, students should be able to:

1. Explain what makes a time series a distinct data structure — ordered observations whose sequence carries information — and construct a daily revenue series from transaction-grain data by a deliberate grain change, per Section 3.3 and the groupby machinery of Chapter 4.
2. Name the four components of a marketing time series — trend, seasonality, cycle, and noise — read each from a decomposition plot at concept level, and state which component each forecasting method in this chapter is built to track.
3. Distinguish recurring calendar effects from structural breaks, explain why history from before a break misleads a model after it, and identify StyleCraft's store-opening inflection as a designed break announced in the company's own data.
4. State the boundary between the weekly seasonality a period-7 method carries and the annual, moving, and promotional calendar effects it does not, and name the three instruments that carry the second group.
5. Construct naive and seasonal-naive forecasts as the time extension of the baselines of Section 8.7, and defend the seasonal-naive rule as the opponent every forecasting model must beat before it earns a planning decision.
6. Describe the exponential smoothing family at working level — simple smoothing, Holt's trend method, damped trend, and Holt-Winters seasonal smoothing — and state what each added layer tracks, what it costs, and how each can overfit.
7. Explain why random train/test splits fail for ordered data, and build the two-layer evaluation this chapter requires: rolling-origin comparison inside the training data for method selection, and one sealed final holdout, chosen to resemble the decision, for the grade.
8. Compute and interpret forecast accuracy with MAPE alongside the MAE and RMSE of Section 8.6, guard MAPE's denominator in code, and report accuracy as a function of horizon estimated across several forecast origins rather than one.
9. Distinguish a point forecast from the distribution it summarizes, build daily, weekly, and season-total prediction intervals by aggregating simulated paths before taking quantiles, and state what a simulation band does and does not price.

10. State when a judgmental adjustment to a statistical forecast is legitimate, size an override as a ramp from controlled evidence with a stated range, and apply the six-element written-override discipline.
11. Recognize the three standing forecasting failure modes — extrapolating through structural breaks, overfitting seasonality, and ignoring promotions — and diagnose each from a forecast exhibit.
12. Verify an AI-drafted forecasting pipeline with the five-point audit of Section 8.11 plus the forecasting supplement, predicting the holdout error before running, and documenting the audit per Appendix D.
13. Assemble a forecast deliverable a planning meeting can hold accountable: method and opponent stated, sealed-holdout grade in MAPE and dollars, intervals communicated in words, overrides documented, and the forecast translated into a declared planning unit whose limits are stated.

Chapter 9 ended with a scored file and a signature. The Comeback Edit's targeting rule was locked — the churn label defined in writing, the threshold re-derived from the program's own twelve-dollar economics, the fairness appendix catching the model before it condemned the newest stores for being new — and the analyst of record signed a deliverable that made predictions about individual people and owned what the program would do to them. Every prediction in that deliverable shared a quiet structural assumption: the customers were exchangeable. The split shuffled them freely, the model scored them in any order, and nothing about customer C004471's row depended on which row came before it. This chapter forfeits the assumption, because the next family of marketing questions abandons the individual for the aggregate and the snapshot for the flow of time itself. What will revenue be in December? How will the new stores' ramp reshape the season? What should the buy plan assume about demand that has not happened yet? The observations are no longer eight thousand interchangeable customers but roughly seven hundred and thirty ordered days, each one glued to its neighbors by trend, rhythm, and event — and the ordering is not an inconvenience to be shuffled away but the single most informative thing about the data. The machinery of Part II rides along where it can: the frame is still declared before fitting, the baseline is still named before the model exists, the test data is still the future the model has not seen. But time rewrites the mechanics of every one of those disciplines, adds two instruments the customer-grain chapters never needed — the decomposition that separates a series into its stories, and the prediction interval that keeps a forecast honest — and installs this chapter's hardest lesson at the exact place StyleCraft's own history planted it: a model trained on the past can only extend the past, and when the business has deliberately broken with its past — an expansion, an opening, a new market — the break is something the analyst must bring to the model, because no amount of fitting will make the model see it on its own.

CONCEPT

What This Chapter Is Really About

Chapters 8 and 9 graded models against a future made of held-out customers — a future the analyst could manufacture at will, because customers are exchangeable and any random twenty percent of them stands in for the rest. Time offers no such courtesy. The future of a time series is not a random sample of its past; it is the next segment of a single unrepeatable path, connected to everything before it and different from all of it. That one fact rearranges the chapter. Evaluation must move the wall of Section 8.3 from between customers to between yesterday and tomorrow, because a shuffled split lets the model interpolate — predict a missing Tuesday from the Tuesdays on either side of it — which is a party trick, not a forecast. The baseline must learn the calendar, because "tomorrow resembles today" is free, obvious, and brutally hard to beat. And the model must be caught at the boundary of its competence: every method in this chapter works by extending patterns the history contains, which means every method is structurally blind to the two things StyleCraft's planning team most needs it to see — the deliberate break with history that the expansion is, and the annual holiday shape that two Decembers cannot teach it.

The chapter's real subject is the resulting division of labor. The model owns the historical rhymes: level, trend, and the weekly rhythm. The analyst owns the future calendar the model cannot read: the announced opening, the moving holiday, the planned promotion, the future the company intends. And the forecast the meeting can trust is the one where both contributions are visible, graded, and signed — the model's on a holdout chosen to resemble the decision, the analyst's in a written override.

Source: Course concept developed for this guide, informed by Hyndman and Athanasopoulos (2021) and Fildes et al. (2009).

10.1 Marketing Decision Context: The Buy Plan That Assumes a Future

Every deliverable in Part II so far has predicted people. The Backstage list ranked customers by expected spend; the Comeback Edit ranked them by risk of leaving. The decision that opens this chapter predicts no one in particular — and commits more money than both programs combined. It is late summer at StyleCraft, and the head of merchandise planning owns the two documents that will define the company's fourth quarter. The first is the holiday open-to-buy plan: the dollar envelope of receipts the November–December assortment may commit, which must be signed roughly six weeks from now because apparel lead times are long and the factories' calendars do not negotiate. The second is the Q4 media pacing plan: how the season's marketing budget spreads across the weeks from late October through the December peak. Both documents assume a future. An open-to-buy envelope is a bet on December demand placed in August; a pacing plan is a bet on which weeks the demand will arrive in. And the bet is bigger and stranger this year than it has ever been, because the number both plans need — expected revenue, week by week, through the holiday season — must span the thing StyleCraft has spent two years doing to itself: the expansion. Eight Wave-4 stores have opened across the trailing year, several of them recently

enough that their ramp curves are still visibly climbing in the weekly numbers. Whatever last December looked like, this December will not look like it, by design.

Three forecasts are already circulating, and — the pattern is by now the guide's oldest running joke — each arrives confident and none is accountable. The first is finance's trendline: last year's holiday revenue, grown by the year-over-year rate of the spring, drawn forward as a straight line. It is hopeful in the precise sense that it assumes the future will be the past plus a constant, and it makes no statement about what would have to be true for that to hold. The second lives inside the planning module of StyleCraft's inventory software, which offers a Demand Forecast button and, when pressed, produces a number — a single number, to the dollar, with no interval, no stated method, and documentation that says only that it uses "advanced machine learning." The third belongs to the merchant who has been right before: the founder's longtime head of buying, whose position is that the suburban stores will "come alive for holiday gifting" and that the season will be up thirty percent. She may well be right. But her number cannot be interrogated, cannot be graded, and — the planning lead's actual complaint — cannot be defended to the CFO when the receipts it justified become markdowns in January. Three numbers, three provenances, and not one of them can answer the only question that matters about a forecast before you bet on it: how wrong does this method tend to be, and how would we know?

So the planning lead has borrowed the VP of Marketing's analyst — still you — and commissioned the deliverable this chapter builds. Not a number: a forecast, in this guide's full sense. A stated method, fitted to the daily revenue history that StyleCraft's own transactions generate, and chosen by comparison rather than preference. A stated opponent — the seasonal-naive rule this chapter will introduce, which any funded model must visibly beat. A grade earned the only way a seasonal forecast can be graded honestly: by holding out a stretch of the known past that resembles the decision, forecasting it as if it were still unknown, and measuring the miss — including the test the expansion makes non-negotiable, whether the method sees the inflection the store openings printed into the series, and what it takes from the analyst to make it see the next one. Prediction intervals, because the plan needs to know not just the center of the future but its plausible width, and because a point forecast delivered without one is — Section 10.15 will use the hard word — a laundered guess. And a written override log, because the analyst knows things the history cannot: the openings already announced for the fall, the promotion calendar marketing has drafted, the moving date of Thanksgiving.

One boundary belongs in the brief rather than in a footnote nine sections later, because it constrains what the whole project can claim. This chapter forecasts chain-level revenue. A revenue forecast can size an open-to-buy envelope in dollars and pace media by week. It cannot, on its own, tell a buyer how many units of which dress in which size to send to which store, because that translation needs realized price, category and size mix, margin, markdown expectation, opening inventory, and store allocation — none of which a chain-level revenue series contains. The program is therefore commissioned as a revenue-and-pacing deliverable with a declared dollar envelope, and the unit-level assortment buy is named as a separate forecast

at a separate grain, with its own baselines and its own holdout. Naming the boundary in the brief is what keeps Section 10.13 from promising quantities the analysis cannot produce.

The receipts lock in six weeks. Sections 10.2 through 10.4 teach the reader to see the series. Sections 10.5 and 10.6 build the forecasting ladder from free baselines to Holt-Winters. Sections 10.7 and 10.8 rebuild honest evaluation for ordered data and widen the forecast from a number into a distribution. Section 10.9 disciplines the human hand. Section 10.10 catalogs the failures. Section 10.11 turns to the AI assistant, which will draft any of it fluently and grade none of it honestly. The labs in Section 10.12 run the whole discipline on the certified series, and Sections 10.13 through 10.15 rehearse the meeting where a forecast becomes a commitment — and someone becomes accountable for the difference.

10.1.1 Opening Case Questions

Keep these questions in mind while reading, and return to them after completing the labs.

- Finance's trendline is not stupid — it is a model, with assumptions. State the assumptions in one sentence each, and name the two features of StyleCraft's recent history (this chapter will call them a component and a break) that the trendline has no way to represent.
- The planning module's Demand Forecast button produces a single number with no interval and no method. Write the three questions you would need answered before the number could enter the plan, and note which chapters of this guide trained each question.
- The merchant's gut number cannot be graded in advance — but it can be used. This chapter will argue there is exactly one legitimate way to bring human knowledge like hers into a forecast. Predict what it is, and check your prediction against Section 10.9.
- A forecast that misses high and a forecast that misses low do different damage to a buy plan. Describe the cost of each direction in inventory terms, and name the section of this guide that taught you to expect the two costs to differ — and to say so before the analysis, not after.
- The brief separates a dollar envelope from a unit buy. State the practical consequence of that boundary for what the deliverable may hand the buying team, and what it must hand them instead.

10.2 Time Series as a Data Structure

The opening case asked for a forecast of revenue by week; this section starts further back, with the shape of the data the forecast will stand on, because the shape is the chapter's first genuinely new idea. Every dataset in Part II so far has been a table of exchangeable rows. The eight thousand customers of the feature table could be shuffled without destroying anything: no customer's row depended on which row preceded it, and the train/test split of Section 8.5 exploited exactly that freedom. A time series abandons the freedom. Its observations are indexed by time — for StyleCraft, one row per calendar day — and the index is not a label but a structure: each day's revenue is stitched to its neighbors by the trend the business is riding, the

weekly rhythm of retail traffic, the season the calendar imposes, and the aftermath of whatever happened yesterday. Shuffle the rows of a time series and the object is destroyed; the information was in the order.

DEFINITION

Time Series

A time series is a sequence of observations of the same quantity, recorded at successive points in time — usually at a regular interval such as a day, week, or month — in which the temporal ordering is part of the data's meaning. The order must be preserved because observations may exhibit serial dependence, trend, seasonality, or structural change, and because those are the patterns a forecast extends. Not every time series is serially dependent — a series of independent draws recorded in time order is still a time series — but the business series this chapter forecasts almost always are, and any procedure that treats their observations as exchangeable discards the temporal context needed to diagnose and forecast them.

Source: Adapted from Hyndman and Athanasopoulos (2021).

In other words, time order is information — and that is the sentence that separates this chapter from the two before it. In the customer-grain chapters, knowing that one row was "next to" another told you nothing; here it tells you nearly everything, because tomorrow is constrained by today in a way that one customer never constrained another. The constraint is what a forecaster harvests: the whole enterprise of this chapter is the disciplined use of a series' own past as evidence about its future, and the methods of Sections 10.5 and 10.6 differ mainly in how much of the past they consult and how quickly they forget it.

Before any of that, the series must exist, and for StyleCraft it does not exist natively — it must be manufactured by a grain change. The company's source of truth is the transactions fact table, one row per order line, per Section 3.3's grain discipline; a daily revenue series is that table aggregated to one row per day, summing line_revenue within each order date — the deliberate grain change of Section 3.3, executed with the groupby machinery Chapter 4's labs built and this guide does not re-teach. The certified dataset ships the result as marketing_daily, the daily table that joins the transaction roll-up with campaign activity, and Code 10.3 performs the reconciliation this guide's known-truth discipline requires: rebuild the daily revenue column from the fact table yourself, and confirm the shipped series matches to the cent, because a forecast inherits every defect of the aggregation underneath it.

Two properties of the manufactured series deserve a sentence each. The grain is a choice, and it is part of the frame: the open-to-buy plan thinks in weeks, the pacing plan in days, and a series can always be aggregated upward (days to weeks) but never disaggregated downward without new assumptions — so this chapter works at the daily grain and rolls up. And the series must be complete: a day with no transactions is a revenue of zero, not a missing row, for exactly the reason a customer with no outcome-window purchases was a label of zero in Section 8.3 — absence is an observation, and dropping it would teach the model a calendar with holes in it. That rule has a mechanical consequence the lab enforces rather than assumes: reindexing a series

onto a complete daily calendar inserts missing days as blanks, not as zeros, and turning the blanks into zeros is a decision the analyst makes and states, not a default the software supplies.

10.3 The Components of a Marketing Time Series

Plot StyleCraft's daily revenue across its twenty-four months and the line looks, at first, like noise with a mood. The skill this section installs is seeing the line as the sum of separable stories — because every forecasting method in this chapter is, underneath its arithmetic, a claim about which of those stories will continue. The decomposition tradition gives the stories standard names, and this guide states them once, formally, in the box below, then spends the section teaching the reader to point at each one on StyleCraft's own plot.

DEFINITION

Trend, Seasonality, and Cycle

The trend of a time series is its long-run direction — the smooth underlying path the series follows once short-run fluctuation is set aside. Seasonality is a repeating pattern with a fixed and known period: the rhythm of days of the week, or of months of the year, that recurs on schedule at a constant number of observations apart. A cycle is a rise and fall that recurs without a fixed period — economic expansions, fashion waves — distinguishable from seasonality precisely because its timing cannot be read off a calendar. What remains when trend, seasonality, and cycle are accounted for is noise (the remainder or irregular component): the unexplained residual variation no pattern claims.

Source: Adapted from Hyndman and Athanasopoulos (2021).

Each component has a StyleCraft face, and the dataset's designers planted all four deliberately. The trend is the expansion story itself: the urban business's steady organic growth, plus the stair-step contributions of each opening — a trend with a complication Section 10.4 will name. The seasonality operates at two nested periods, and the difference between them organizes the rest of the chapter. The weekly rhythm is retail's heartbeat: quiet early weekdays, a build through Friday, a weekend peak — a pattern that repeats roughly one hundred and four times in the history, which is why every method in this chapter will be given a seasonal period of seven. The annual rhythm is the merchandising calendar: the back-to-school lift and the November–December holiday surge, patterns the history contains only twice — a scarcity with consequences Sections 10.4 and 10.10 will price, and the reason this chapter never asks a period-7 method to carry them. Cycles, at this guide's concept level, are the honest admission that not every wave has a schedule: a viral product moment, a macroeconomic squeeze on discretionary apparel, a competitor's collapse. The twenty-four-month window is too short to estimate cycles and this chapter does not try; the term is owned here so the reader stops mislabeling every unexplained wave "seasonality." And the noise is everything else — weather, whim, and the irreducible randomness of a few hundred purchase decisions a day — the component no method forecasts and every honest interval respects.

The instrument that makes the components visible is decomposition, and this guide uses it at concept level: as a reading tool, not a forecasting engine. A decomposition procedure takes the observed series and splits it into estimated trend, seasonal, and remainder parts — classical procedures do this with moving averages, and the modern standard, STL, does it with iterated local smoothing (Cleveland et al., 1990) — under an assumed structure, most simply the additive one: observed value equals trend plus seasonal effect plus remainder. The mechanics stay in the software; what the analyst owns is the reading. A decomposition plot stacks four panels — observed, trend, seasonal, remainder — and Code 10.4 will put StyleCraft's own on screen with the store-opening dates drawn on it. Read the trend panel for the expansion: the climb, and the visible steps where openings land. Read the seasonal panel for the heartbeat: the seven-day sawtooth, regular as a pulse, whose peak-to-trough height in dollars is exactly what the seasonal-naive rule of Section 10.5 carries for free. And read the remainder panel the way Chapter 4 taught you to read every residual exhibit: not as garbage but as a question list, because a spike in the remainder is a day the components could not explain — a promotion the decomposition does not know about, a storm, a data defect — and Section 10.10's third failure mode begins with an analyst who never looked.

DEFINITION

Decomposition

Decomposition is the separation of an observed time series into estimated components — trend, seasonal, and remainder (with cycle folded into trend at short horizons) — under an assumed structure, most commonly additive (the components sum to the observed value). It is used in this guide as an analytic reading instrument: a way to see which stories a series contains and how large each is, before choosing a forecasting method that must track them. A decomposition is fitted at one declared period; a decomposition at period 7 estimates the weekly pattern and folds everything annual into trend and remainder.

Source: Adapted from Cleveland et al. (1990) and Hyndman and Athanasopoulos (2021).

Table 10.1 collects the components with their StyleCraft instances and — the column that makes the table load-bearing for the rest of the chapter — the instrument that will be responsible for each. The table is a map of Sections 10.5 through 10.9 before the reader gets there: the seasonal-naive baseline handles the weekly pattern by memory, Holt's method adds a tracked trend, Holt-Winters tracks both, and the three lower rows are the ones no fitted method in the toolkit handles — the annual calendar, which needs an explicit calendar model; the break, which belongs to the analyst; and the noise, which belongs to the interval.

Table 10.1

The components of StyleCraft's daily revenue series, and what carries each

Component	StyleCraft instance	Carried by
Trend	Urban organic growth plus the expansion's cumulative lift	Holt's method and Holt-Winters (Section 10.6)
Weekly seasonality	Quiet Tuesdays, weekend peaks — period 7, about 104 repetitions in the history	Seasonal-naive baseline (Section 10.5); Holt-Winters at seasonal_periods=7 (Section 10.6)
Annual and moving calendar effects	Back-to-school lift; the November–December surge; Thanksgiving and Black Friday, whose dates move	Not carried by any period-7 method. An explicit calendar model, a prior-holiday benchmark, or documented scenarios (Sections 10.4, 10.8, 10.9)
Promotions	Chosen-date discount events in the campaigns table; future ones in a calendar the model has never seen	The analyst, via the declared calendar and documented overrides (Sections 10.4, 10.9)
Cycle	Fashion and macro waves with no fixed period	No method here; named to prevent mislabeling
Structural break	Store-opening steps and ramps; any future opening or policy change	The analyst, via declared knowledge and sized overrides (Sections 10.4, 10.9)
Noise	Irreducible day-to-day randomness	No method; priced by the prediction interval (Section 10.8)

10.4 Calendar Effects and Structural Breaks

The seasonal component of Section 10.3 deserves a sharper look before any method is built on it, because "seasonality" in marketing data is really a bundle of calendar effects — and most members of the bundle are not carried by a seasonal period at all. Calendar effects are the systematic influences the calendar exerts on a business series: the day-of-week pattern; the fixed-date holidays; the floating retail events — Thanksgiving and the Black Friday weekend it anchors — whose dates move within a window; and paydays, school calendars, and the merchandising rhythms a drops-driven brand imposes on itself. What unites them is knowability: every one of them is on next year's calendar already. What divides them is whether a fitted seasonal component can carry them, and this is the distinction the draft of this chapter blurred and the audit of it corrected.

CONCEPT

Knowable Is Not the Same as Carried

It is tempting to reason that because calendar effects are known in advance, a method with seasonal memory will carry them. Knowable dates and repeating seasonal indices are not the same thing, and the difference decides what the analyst must supply.

A day-of-week rhythm is carried automatically by a period-7 seasonal component, because it recurs at a constant number of observations apart, every time, and a two-year daily history contains about a hundred repetitions of it. Christmas is known, but the weekday it falls on changes, so its effect on a daily series lands at a different position in the weekly cycle each year. Thanksgiving and Black Friday move within November entirely, so a day-of-year index misaligns them by up to six days. A promotion is a planned event, not a season: the company chooses the date, and next year's dates live in a marketing calendar the model has never seen. A store opening is a structural change, not a seasonal effect at all.

The working rule: calendar effects are knowable in advance, but only stable fixed-period patterns are carried automatically by seasonal memory. Moving holidays, promotions, and openings must be supplied explicitly, through calendar variables, a prior-period benchmark, documented scenarios, or written overrides. Daily retail series routinely contain more than one seasonality, and moving holiday effects are handled with explicit indicators rather than a fixed seasonal lag.

Source: Course concept developed for this guide, informed by Hyndman and Athanasopoulos (2021).

The boundary that follows is stated once here and honored everywhere after, because half the chapter's evaluation design exists to enforce it. This chapter's Holt-Winters model is fitted at seasonal_periods=7. It tracks the weekly rhythm and nothing else seasonal. StyleCraft's annual holiday demand is two observations of a 365-position pattern, one of them mid-expansion; it cannot be estimated reliably from that, and this chapter does not pretend otherwise. Annual holiday structure and moving retail events are carried by three instruments instead, all of them the analyst's: an explicit calendar layer estimated from the prior holiday and aligned on the moving date rather than the calendar date (Code 10.9); documented scenarios that vary the calendar assumption (Code 10.20); and written overrides with owners and review dates (Section 10.9). The seasonal-naive rule does not rescue this either: at a seven-day lag it repeats last week, not last December, and a daily annual seasonal-naive rule would have to reckon with 365- and 366-day calendars and with holidays that move. The chapter does not need dynamic harmonic regression to make this point. It needs to stop implying that a period-7 model carries annual holiday structure — and then to build an evaluation that proves it does not.

The structural break is the calendar effect's dangerous cousin: an event that changes the series' underlying behavior rather than decorating it. A seasonal effect visits and leaves; a break moves in. After a break, the level, the trend, or even the seasonal pattern of the series is different — permanently, or at least indefinitely — and the history from before the break describes a process that no longer exists.

DEFINITION

Structural Break

A structural break is a point in time at which the data-generating process behind a series changes — a shift in level, trend, or seasonal pattern that persists after the event rather than reverting. History from before a break is evidence about a different regime: models fitted across an unmodeled break blend the two regimes into a description of neither, and models fitted only on pre-break data will systematically mis-forecast the post-break series. In marketing, breaks are frequently self-inflicted and announced in advance — store openings, price restructurings, channel launches — which makes them knowable to the analyst even though they are invisible to any method that reads only the series' past.

Source: Adapted from Hyndman and Athanasopoulos (2021).

StyleCraft's series is a curriculum of breaks by design. Every store opening in the roster added a step to daily revenue: a new level, sustained, with a ramp as the store found its customers — the suburban and flagship step-ups the dataset specification plants explicitly, and the reason Section 10.3's trend panel shows stairs instead of a slope. The expansion inflection the planning lead must forecast across is exactly this: the cumulative effect of the Wave-4 openings, several recent enough that their ramps are still completing, which means the series' most recent months are governed by a regime the earlier history only partially describes. Why does history before a break mislead after it? Because every method in this chapter forecasts by assuming continuation — of a level, a trend, a rhythm — and a break is precisely a discontinuation. A model fitted through an unacknowledged opening will read the step as either an alarming trend (and over-extrapolate it) or an outlier (and ignore it); a model fitted only on pre-opening data will confidently forecast the smaller company that no longer exists. Neither model is broken. Both are answering the question they were asked — continue the past — and the past stopped being the operative regime on a date printed, in StyleCraft's case, in the stores table's opening_date column.

That last clause is the section's real payoff, and it earns the chapter's division of labor its box. Marketing's structural breaks are unusual among forecasting problems in being disproportionately scheduled: the company decides to open the store, launch the channel, restructure the prices — and writes the date down before it happens. The information exists; it simply does not live in the series. It lives in the stores table, the campaign calendar, the board deck. The analyst who joins that knowledge to the model — through the overrides of Section 10.9, or simply through choosing what history a model is allowed to learn from — is not contaminating the forecast with opinion. She is supplying the one input the method is structurally incapable of finding, and the whole verification design of Code 10.12 is built to make the point unforgettable: the model will be graded, on the known past, on an inflection the reader can see coming and the model cannot.

CONCEPT

The Calendar Is a Feature Only the Analyst Can Read

Every fitted method in this chapter consumes exactly one input: the series' own past. Everything else StyleCraft knows about its future — the openings already announced, the promotion calendar already drafted, the date Thanksgiving falls on this year, the media plan already budgeted — is invisible to the model unless the analyst carries it in. This is the forecasting form of a discipline the guide has taught since Chapter 2: the analysis serves a decision, and the decision-maker's knowledge is part of the evidence.

The working rule, enforced in the labs and the deliverable: before fitting anything, write down the known calendar — past breaks and events the model's training window contains, future breaks and events its forecast window will contain — and treat every item on the second list as an open obligation that either an override or an explicit stated assumption must discharge. A forecast delivered without its calendar is a forecast that has silently assumed nothing is scheduled to happen — which, at a company mid-expansion, six weeks before a holiday season, is not a simplification but a falsehood.

Source: Course concept developed for this guide, informed by Hyndman and Athanasopoulos (2021) and Fildes et al. (2009).

10.5 Baselines in Time: Naive and Seasonal-Naive

The toolkit proper opens where Chapter 8 taught every predictive project to open: with the opponent. Section 8.7 operationalized baselines as deliberately simple rules whose defeat is the model's entry fee, and its Table 8.3 introduced the last-value baseline — predict that the previous window repeats — noting it as the incumbent with teeth. This section is that row's time extension, and one sentence of reminder replaces any re-teach: a baseline is a fully specified prediction rule, graded on the same origins with the same metric as every candidate it opposes. What time adds is a sharper pair of rules and a humbling literature about how hard they are to beat.

The naive forecast is the last-value rule at the grain of the series: tomorrow will equal today. For a multi-day horizon it extends flat — every future day forecast at the last observed value — which sounds too simple to dignify until one remembers what it encodes: the series' most recent level, which for a trending, breaking series like StyleCraft's is often closer to the truth than an average over a history the expansion has outgrown. The seasonal-naive forecast upgrades the memory from yesterday to the rhythm: each future day is forecast by the last observed value from the same position in the seasonal cycle. Its examples must be stated at the right grain, because conflating them is how students end up believing a weekly rule carries an annual pattern. On StyleCraft's daily series at period 7, the rule is *this Saturday equals last Saturday* — and that is the only version this chapter uses. On a monthly series at period 12, the rule is *this December equals last December*. There is no clean daily version of the second: a daily annual seasonal-naive forecast would have to align a 365-day cycle against a 366-day one and would place Black

Friday on the wrong day every year, which is exactly the moving-holiday problem of Section 10.4. The rule's charm, at the weekly period, is that it inherits the entire weekly pattern for free — every stable fixed-period calendar effect carried automatically, at the cost of zero fitted parameters.

DEFINITION

Naive and Seasonal-Naive Forecasts

The naive forecast predicts that every future value equals the most recent observed value — the last-value baseline of Section 8.7 applied at the series' own grain, extended flat across the horizon. The seasonal-naive forecast predicts that every future value equals the most recent observed value from the same position in the seasonal cycle at a declared period: at period 7 on daily data, each Saturday is forecast by last Saturday; at period 12 on monthly data, each December by last December. Both are fully specified, zero-parameter rules, computable from the series alone, and they are the standard opponents of time series forecasting: a fitted model that cannot beat the seasonal-naive rule on an honest evaluation has not yet demonstrated that its fitting bought anything.

Source: Adapted from Hyndman and Athanasopoulos (2021).

Students meet these rules expecting them to be strawmen, and the field's most instructive empirical tradition exists to correct the expectation. The M-competitions — decades of blind forecasting tournaments across thousands of real business series, most recently the M4 competition's one hundred thousand — return the same finding with the reliability of a season: simple methods are embarrassingly competitive, many sophisticated entrants fail to beat them, and the average gap between a tuned statistical model and a seasonal-naive rule is far smaller than the gap between either and a bad decision about which history to trust (Makridakis et al., 2020). The lesson is not that fitting is pointless — the best methods do win, and Section 10.6's family is among them — but that the win is a margin, usually a modest one, and a forecaster who cannot state her margin over seasonal-naive is a forecaster who does not know whether her model is contributing anything. StyleCraft's series is designed to teach the calibrated version of this humility: the weekly rhythm is strong, so seasonal-naive posts a respectable score, and the labs' designed expectation is that Holt-Winters beats it by a real but unspectacular margin earned almost entirely where the baseline is blind — the trend the rule cannot climb, since last Saturday is always one week of growth behind this one.

The baseline thread the guide has carried since Chapter 2 is now fully assembled, and it is worth one sentence of signposting: baselines entered as interpretive discipline (Section 2.7), operationalized into opponents (Section 8.7), and here acquire a calendar — and when Chapter 12 gives them visual form as reference lines, the thread will be complete. For this chapter's purposes the working rule is the one the frame of Code 10.6 declares before any model is fitted: the seasonal-naive forecast, at the weekly period, is the opponent; the naive forecast is reported beside it as the floor; and every exhibit in the deliverable carries the margin over both, in the declared metric, on the declared evaluation — because a forecast's value to the plan is not its

accuracy but its advantage over the free rule the planning lead could have used without hiring anyone.

10.6 Smoothing: Moving Averages and the Exponential Smoothing Family

The baselines remember one value; fitted methods earn their keep by remembering more, and this section builds the chapter's working family in the order of what each layer adds. The entry instrument is the moving average: replace each day's value with the mean of a window of days around or behind it. A seven-day trailing moving average of StyleCraft's revenue erases the weekly sawtooth — each window contains exactly one of each weekday — and what remains is the smoothed level and trend, which is why the moving average is the classical decomposition's engine and the analyst's fastest reading tool for "what is the business actually doing under the rhythm." Two disciplines attach. The window is a dial: seven days removes weekly rhythm, twenty-eight removes most calendar texture, and a window wider than the feature of interest removes the feature — smoothing is deletion, and the analyst should know what she is deleting. And a moving average used for reading is centered or trailing as convenience dictates, but a moving average used inside any forecasting pipeline must be trailing only — a centered window consumes future days, which at forecast time do not exist, and Section 10.11's audit will treat a centered window in a feature the way Section 8.4 treated any information from beyond the wall.

DEFINITION

Moving Average

A moving average replaces each observation with the mean of a fixed window of surrounding observations — trailing (the window ends at the observation) or centered (the window straddles it) — smoothing away fluctuation at periods shorter than the window. A window equal to the seasonal period removes the seasonal pattern entirely, which makes the moving average both the classical instrument for extracting trend and a standing reminder that smoothing is deliberate information removal.

Source: Adapted from Hyndman and Athanasopoulos (2021).

The moving average smooths history; the exponential smoothing family turns smoothing into forecasting, and its governing idea is a graceful improvement on the flat window: weight the past by recency, with weights that decay exponentially, so that yesterday matters most and last quarter still whispers. Simple exponential smoothing maintains exactly one running quantity — the level, an exponentially weighted average of everything seen so far — updated each day by moving a fraction (the smoothing parameter) toward the newest observation. Its forecast is that level, extended flat: simple smoothing is, in effect, a naive forecast with a better memory, appropriate for series with no trend and no season, which StyleCraft's is not. Holt's method adds a second running quantity — the trend, the series' current per-day climb, itself exponentially smoothed — and forecasts by extending the level along the trend: a line, not a flat (Holt, 2004). A damped variant of Holt's method shrinks that climb as the horizon lengthens, on the

empirically well-supported ground that trends estimated from recent data usually flatten rather than continuing forever; it is a candidate in this chapter's comparison precisely because a company mid-expansion invites over-extrapolation, and whether damping helps is a question for the evidence rather than for taste (Gardner, 2006). Holt-Winters completes the family by adding the third quantity this chapter has been circling: a set of seasonal effects, one per position in the cycle — seven, at StyleCraft's declared weekly period — each updated when its weekday comes around, and applied to the trended level so the forecast carries the rhythm forward (Winters, 1960). Fifty years of practice have kept the family central for the reasons this guide values: it is transparent enough to explain in a meeting, fast enough to refit weekly, and empirically hard to beat on exactly the seasonal, trending business series marketing generates (Gardner, 2006).

DEFINITION

Exponential Smoothing, Holt's Method, and Holt-Winters

Exponential smoothing is a family of forecasting methods that maintain running, recency-weighted estimates of a series' components, updating each by a smoothing parameter as new observations arrive. Simple exponential smoothing tracks a level only and forecasts it flat. Holt's method adds a smoothed trend and forecasts along it; a damped variant shrinks the projected trend as the horizon lengthens. The Holt-Winters method adds smoothed seasonal effects at a declared period and forecasts the trended level with the seasonal pattern reapplied — level, trend, and season, each with its own parameter, fitted by minimizing historical forecast error. The declared period is a modeling decision: a model fitted at period 7 carries a weekly pattern and no other.

Source: Adapted from Holt (2004), Winters (1960), and Gardner (2006).

Table 10.2 states the ladder in the form the chapter will use it: what each rung adds, what it costs, and how it fails — because each added component is also an added way to be wrong. The costs compound quietly. Each layer introduces a parameter fitted from history, and fitted parameters are flexibility in exactly Section 8.8's sense: a trend parameter can mistake a two-week surge for a climb and extrapolate it; seven seasonal effects estimated from a noisy stretch can enshrine noise as rhythm — overfitting's forecasting costume, caught the same way Chapter 8 caught it, by out-of-sample evaluation. And one failure is shared by the whole family, stated here so Table 10.2 can carry it: every rung forecasts by continuation, so no rung sees a structural break coming, and no rung at period 7 sees a holiday season coming either. The ladder climbs trends and rides weekly rhythms, and walks, at full confidence, straight past the announced opening and into December.

Table 10.2

The smoothing ladder: what each layer adds and what it costs

Method	Tracks	Adds over the rung below	Characteristic failure
Naive / seasonal-naive (Section 10.5)	Last value; last cycle at the declared period	Nothing — the free opponents	Blind to trend; one bad reference day is copied forward
Moving average	Smoothed level (a reading tool)	Noise suppression at a chosen window	Deletes features wider than intended; centered windows consume the future
Simple exponential smoothing	Level	Recency-weighted memory	Forecasts flat; trend and season leak into the level estimate
Holt	Level + trend	Climbs with the series	Extrapolates temporary surges as permanent trends
Holt, damped trend	Level + shrinking trend	Restrains long-horizon extrapolation	Under-projects genuine sustained growth; one more fitted parameter
Holt-Winters, period 7	Level + trend + weekly season	Carries the weekly rhythm forward	Overfits seasonal effects on thin stretches; carries no annual or moving calendar effect; like all rungs, cannot see a break coming

10.7 Honest Evaluation in Time: Rolling Origins, the Sealed Holdout, and Forecast Accuracy

Section 8.5 closed with a bridge this section now crosses: forecasting cannot randomly split its data, and here is why. The random split of Chapter 8 worked because its wall in time lived inside each customer's label — features before the snapshot, outcome after — so shuffling customers could not shuffle the future into the past. In a time series, the observations are the time, and a random split shatters the wall into confetti. Hold out a random twenty percent of days and every held-out day sits surrounded by training days: last Tuesday in the training set, next Tuesday in the training set, and the "test" Tuesday between them needing only interpolation — an average of its neighbors — to be predicted with flattering precision. The flattery is structural, not accidental. Adjacent days share trend, season, and even weather; a model graded on interpolation is being graded on a job that never occurs in deployment, because the deployed forecast never knows the

future side of the gap. In Section 8.4's vocabulary, a random split of a time series is leakage by design: every training day after a test day is information from the future, and the resulting score measures how well the future predicts itself — Code 10.11 manufactures the demonstration, and the designed result is a shuffled "error" smaller than any honest forecast achieves.

The repair is the time-aware evaluation: put the wall back where deployment puts it. Train on the past, forecast the future, with every training observation dated strictly before every evaluated observation. The verification is a one-line date-order check — the latest training date must precede the earliest test date — and the check earns permanent membership in this chapter's audit because it is the forecasting sibling of Section 8.12's recency range check: a single violated inequality convicts an entire evaluation.

But a single contiguous holdout at the end of the series, which is where most forecasting tutorials stop, is not enough, and the reason is the same reason Chapter 8 needed cross-validation. Comparing four candidate methods on one final block makes that block part of model selection: if Holt-Winters is retained because it won there, the same block cannot also serve as an unbiased final grade. Chapter 8 solved this with k-fold cross-validation inside the training data. Time series solve it with the rolling origin.

DEFINITION

Rolling-Origin Evaluation

Rolling-origin evaluation estimates out-of-sample forecast accuracy by repeating one exercise from several successively later forecast origins: at each origin, fit on everything up to that date, forecast a fixed number of periods ahead, record the errors against what actually happened, and then move the origin forward. Every training set precedes its own test observations, so no evaluation consumes the future, and averaging across origins gives a more stable estimate than one path can. Because each origin produces one error per step ahead, the design also permits accuracy to be reported separately at each forecast horizon. It is the time series counterpart of cross-validation, and it is performed inside the training data so that a final holdout can stay sealed.

Source: Adapted from Tashman (2000) and Hyndman and Athanasopoulos (2021).

The two instruments do different jobs, and this chapter uses both in a fixed order that Table 10.3 states. Rolling origins inside the training data choose the method: naive, seasonal-naive, Holt-Winters, and damped-trend Holt-Winters are compared on the same origins with the same metrics, and the winner is named before anything else happens. Then the method is frozen and one final holdout is opened, once, for the grade. And the final holdout is chosen for a property most tutorials ignore: it must resemble the decision. This matters enough to be the chapter's largest design commitment. The plan under discussion is a holiday plan. A method graded on eight weeks of late spring has been tested on nothing that matters to it — no November–December demand, no holiday promotions, no Black Friday timing, no annual seasonal effect. So the sealed holdout in this chapter is November 1 through December 31, 2025, forecast from data ending October 31 — a prior holiday season, forecast from before it, which is the same shape as the actual commission. The result is uncomfortable and instructive: the weekly model,

graded on the season it was never told about, misses by roughly twice its rolling-origin error and misses in one direction. That is the point. A holdout that resembles the decision is the only holdout that can expose the gap the decision will fall into.

Table 10.3

The two evaluation layers, and the third window that is not an evaluation at all

Layer	Window in this chapter	What it is for	What it may not do
Rolling-origin comparison	Origins every 28 days inside July 2024 – October 2025, each forecasting 56 days	Choose the method and configuration; estimate accuracy by horizon across several origins	It may not be reported as the final grade, because the candidates were compared on it
Sealed final holdout	November 1 – December 31, 2025, forecast from data ending October 31	Grade the frozen method once, on a window shaped like the decision	It may not be consulted twice, and it may not be used to pick a method
Planning forecast	July 1 – December 31, 2026, refitted on all certified history	Produce the number the plan consumes, with the calendar applied and scenarios attached	It may not be called a graded forecast; no actuals exist, and its horizon exceeds the graded one

With the where settled, the what: forecast accuracy metrics. The dollar metrics transfer whole — MAE and RMSE were built in Section 8.6 and are not re-taught; one sentence of recontextualization suffices. Each evaluated day's error is actual revenue minus forecast revenue, MAE prices the typical daily miss in dollars, RMSE amplifies the large misses, and the choice between them still follows the decision's cost structure per Section 2.7. What forecasting adds is the metric Section 8.6 deliberately deferred to this chapter: the percentage form. Planning conversations run on percentages — "we missed the quarter by four percent" — because a percentage travels across scales: it lets the planning lead compare forecast quality across a $9,000 Tuesday and a $14,000 Saturday, across this year and last, across StyleCraft and the benchmarks in her trade association's survey. The standard instrument is MAPE, the mean absolute percentage error: each day's absolute error divided by that day's actual value, averaged, times one hundred.

DEFINITION

MAPE (Mean Absolute Percentage Error)

MAPE is the average of a forecast's absolute errors expressed as percentages of the actual values: for each period, the absolute error divided by the actual, averaged across the evaluated periods and multiplied by 100. It is scale-free, which makes it the planning world's shared currency for forecast accuracy — and it is undefined at actual values of zero and explosive near them, because a small denominator turns a modest dollar miss into an enormous percentage. MAPE is therefore reported only after its denominators are checked, and it is disqualified for series that touch or approach zero, where the dollar metrics of Section 8.6 must lead.

Source: Adapted from Hyndman and Koehler (2006).

The near-zero failure deserves the paragraph the definition promises, because it is not a curiosity — it is the most common way a competent forecast acquires a headline number that destroys its credibility. Divide by a near-zero actual and the quotient is arithmetic dynamite: Code 10.2 stages the demonstration, where a single storm-closed day of a few hundred dollars converts a forecast whose typical miss is five percent into a reported MAPE in the hundreds — one denominator, not one hundred bad forecasts, produced the number. StyleCraft's chain-level daily revenue sits comfortably above zero, so MAPE is a legitimate lead metric for this deliverable; but the moment the same pipeline is pointed at a single store's series, a single channel's, or a new store's launch weeks, near-zero actuals arrive and the metric must be benched. This chapter therefore does not leave the check to the analyst's memory. The MAPE helper of Code 10.1 asserts that every denominator is strictly positive and refuses to return a number otherwise, and the reporting helper prints the minimum denominator and a count of thin days beside every MAPE it displays. Hyndman and Koehler (2006) supply the deeper survey of scale-free alternatives for readers who meet series this guide's rule disqualifies.

One more quantity completes the evaluation vocabulary, because time gives the frame's horizon (Section 8.2) a structure it did not have at the customer grain. A forecast is not one claim but a sequence of claims — tomorrow, next week, week eight — and the claims are not equally hard. Uncertainty compounds with each step ahead: trend estimates drift further from the truth, and the world has more time to break. So expected error grows with horizon, and the plan must price it, because the December weeks it cares most about sit at the horizon's far, foggy end. But the sentence has to be said carefully, and the draft of this chapter said it too strongly. Realized error in one evaluation period need not rise monotonically, because each week ahead also differs in promotions, openings, weather, seasonal level, and plain luck — a spike in week seven may be a promotion rather than a consequence of being seven weeks out. This is exactly why horizon accuracy is estimated across several rolling origins rather than read off a single forecast path: with one path, a horizon curve is the history of one particular two months; with seven origins, each horizon has seven errors and the pattern begins to mean something. Code 10.10 builds the table that way and prints how many errors sit behind each row.

DEFINITION

Forecast Horizon

The forecast horizon is the span of future periods a forecast claims — the distance, in the series' own grain, from the last observed value to the furthest forecast one. Because uncertainty compounds with each step ahead, forecast uncertainty and expected error generally increase with horizon, although realized error need not rise monotonically in any one evaluation period. Honest evaluation therefore reports accuracy as a function of horizon rather than as a single pooled number, estimates each horizon's error from several forecast origins rather than one, and states how many errors stand behind each figure. A method's one-week-ahead grade and its eight-week-ahead grade are different facts, and the decision consumes the one that matches its own lead time.

Source: Adapted from Hyndman and Athanasopoulos (2021) and Tashman (2000).

10.8 Point Forecasts, Prediction Intervals, and Scenarios

Everything so far has graded a forecast as a number meeting a number; this section widens the forecast into what it actually is, because the widening is the deliverable's most consequential exhibit. A fitted method, asked about a future Tuesday, does not know a value — under its own assumptions it implies a distribution of plausible values, centered where the components point and spread by what the components cannot carry. The point forecast is a one-number summary of that distribution — its center, essentially — and the summary is legitimate right up until it is mistaken for the claim itself. The claim is the distribution, and the instrument that communicates it is the prediction interval.

DEFINITION

Prediction Interval

A prediction interval is a range around a point forecast derived from the forecast distribution the fitted model implies. Under the model and its assumptions, an 80% interval is constructed so that comparable forecast intervals would contain their future observations about 80% of the time. Its width is a claim about uncertainty: wide where the method knows little, growing with horizon, and honest only if the method's error assumptions are honest. A point forecast reported without an interval conceals exactly the information — the plausible range of the future — that a planning decision most needs.

Source: Adapted from Chatfield (1993) and Hyndman and Athanasopoulos (2021).

The definition's opening clause carries the qualification that keeps the instrument honest, and this chapter states it twice because it is the sentence practitioners most often drop. An interval is not a statement about the future. It is a statement about the future *under the model* — its structure, its estimated parameters, its assumption that whatever generated the past will keep generating the future. This chapter builds its intervals by simulation: draw many random future paths forward

from the fitted Holt-Winters state, then read percentiles across the paths. That method propagates future disturbances through the fitted states and compounds them across the horizon, which is genuinely the largest part of the uncertainty at short horizons. What it does not do is re-estimate the model's parameters for each simulated path, so the band is conditional on the parameters that were fitted, and it prices nothing at all about the calendar the model was never told. The working statement, which belongs verbatim in the deliverable: the simulation band reflects future innovation uncertainty under the fitted Holt-Winters model and the propagation of that uncertainty across the horizon; it is conditional on the estimated parameters and may understate total uncertainty when the parameter estimates or the structural assumptions are themselves unstable. The labs make the omission concrete rather than rhetorical: on the sealed holiday holdout the daily band achieves close to its nominal coverage while the season total lands near the top of its range, because the band prices noise faithfully and prices the missing holiday not at all.

Three readings make the interval a working tool rather than a decoration. First, width is honesty, not weakness. An interval that spans plus-or-minus twelve percent of December revenue is not the analyst hedging; it is the method reporting the size of its own ignorance, measured from its own errors — and a competitor who quotes the same future with a bare point number has not reduced the uncertainty, only concealed it. The planning instinct to prefer the confident number is precisely backward, and Section 10.15 will make the ethical version of the argument. Second, intervals fan. Because uncertainty compounds with horizon (Section 10.7), the band widens with each step ahead — the plotted forecast wears a cone, narrow tomorrow and wide in week eight — and the cone's shape is itself decision-relevant information: it tells the planning lead how much of the plan can be committed early at low risk and how much should wait for the re-forecast. It also tells her when the band has widened past usefulness, which for a period-7 method asked about a date six months out is a real and reportable finding rather than an embarrassment. Third, intervals are themselves gradable — but grading them takes more than one number.

Coverage is the obvious grade: what share of evaluated periods actually landed inside the band. It is necessary and, alone, badly incomplete, because an arbitrarily wide interval achieves perfect coverage while saying nothing. The draft of this chapter over-read a single coverage figure in both directions, treating a number near eighty as vindication and a number well above it as proof the band was too wide. Neither inference is safe on one short, serially dependent holdout, and the correction is a reporting standard rather than a rule of thumb: report the nominal coverage, the realized coverage, the average interval width, the number of evaluated periods, and coverage by horizon where the periods permit it — then read the set together. Coverage of 94 percent on sixty-one dependent daily observations is a reason to investigate, not a verdict; coverage of 70 percent is likewise a question. Distributional forecasts are ideally evaluated with instruments that reward coverage and sharpness at once, which this guide names and leaves to later coursework (Hyndman & Athanasopoulos, 2021); at this level, the discipline is simply to publish the width beside the coverage so that no one can buy the second by inflating the first.

CONCEPT

You Cannot Add Up the Edges of an Interval

The plan does not consume daily numbers. It consumes a week, a month, and a season total — so the daily interval has to be aggregated, and there is exactly one correct way to do it and one very natural wrong way. The wrong way is to add up the daily bounds: sum the seven daily 10th percentiles to get a weekly lower bound, sum the seven 90th percentiles to get an upper one. That arithmetic is not the weekly interval. The sum of seven daily 10th percentiles is the total that would occur if every one of the seven days landed at its own 10th percentile simultaneously, which is a far rarer event than the week's total landing at its 10th percentile — so the summed band is too wide, and by an amount that depends on how the days move together.

The right way is to aggregate first and quantile second. Take each simulated path, sum it over the seven days to get that path's weekly total, and then read the 10th and 90th percentiles across the paths' weekly totals. The same procedure gives the eight-week and season totals: sum each path over the whole window, then take percentiles of the totals. Dependence among the days is then handled automatically, because each path carries its own dependence with it.

How wrong is the shortcut, and in which direction? Neither question has a general answer, and overstating this is its own error. The rule is that a quantile of a sum is not the sum of the marginal quantiles: the discrepancy depends on the marginal distributions and on the dependence among the days, so the resulting band may be too wide, too narrow, or miscalibrated in ways that are not a single width at all. In StyleCraft's Holt-Winters simulations the shortcut happens to produce a wider band, because the simulated paths are strongly dependent across days — a path that drifts high stays high — so the summed edges are only a few percent too wide. Destroy that dependence by shuffling the paths day by day, as Code 10.17 does, and the same arithmetic overstates the width by a factor near the square root of seven. Same wrong formula, an order of magnitude difference in how wrong it is, and no way to know which without the paths. The correct method, by contrast, is invariant: aggregate every simulated path first, then take quantiles across the resulting totals.

Source: Course concept developed for this guide, informed by Hyndman and Athanasopoulos (2021).

The interval prices the uncertainty the method can see; the scenario prices the uncertainty it cannot. A scenario is a conditional forecast: the same machinery run under a stated assumption about the calendar the model cannot read — the openings ramp as planned, the openings stall; the holiday behaves like last year's, it behaves like the weaker of the two years on record. Scenarios and intervals answer different questions and the deliverable needs both: the interval says how wrong the method tends to be when the world stays in regime, and the scenario set says how the forecast moves when the analyst's declared assumptions do. For this plan, the practice is the three-line table the planning world already speaks — a base case carrying the declared calendar, a downside and an upside carrying the named assumption changes — with the assumptions written beside the numbers, because a scenario whose assumption is unstated is just three guesses wearing a rubric. And the deliverable must say which instrument is the planning range at which horizon. For the eight weeks after the forecast origin, the simulation band is meaningful and the scenarios refine it. For December, six months out, the band has widened past

usefulness and the scenario envelope carries the range — an assumption set, honestly labeled, rather than a graded forecast wearing a graded forecast's clothes.

10.9 Judgmental Adjustment and the Documented Override

The merchant's thirty percent has been waiting since Section 10.1, and this section is where the guide keeps its promise about the one legitimate way to use it. The temptation the section disciplines is universal: every statistical forecast, presented, is immediately met by humans who know things — and some of them do. The planning lead knows the fall openings are announced; marketing knows the promotion calendar; everyone in the room knows what date Thanksgiving falls on, which is more than the model knows. A forecasting practice that ignores such knowledge wastes real information; a practice that lets every stakeholder nudge the number produces the laundered consensus Section 10.15 examines. The discipline that threads the needle is the judgmental adjustment, made under rules.

DEFINITION

Judgmental Adjustment

A judgmental adjustment is a deliberate, human-made change to a statistical forecast, intended to incorporate information the fitted method could not access — a scheduled event, a structural change, contextual knowledge outside the series. Its legitimacy depends on its provenance and its paperwork: adjustments carrying genuine event knowledge improve forecasts, while frequent small adjustments — optimism, anchoring, target pressure wearing a forecast's clothes — reliably degrade them, and the empirical record shows the difference is discipline, not intuition.

Source: Adapted from Fildes et al. (2009).

The empirical record deserves its sentence because it cuts both ways with unusual clarity. Fildes and colleagues examined more than sixty thousand forecasts and adjustments inside real supply-chain forecasting operations and found that judgmental adjustments are extremely common; that larger adjustments tended to improve accuracy more than small ones, which mostly added noise; and that positive adjustments were substantially less reliable than negative ones — upward revisions were, in effect, optimism applied with confidence (Fildes et al., 2009). The lesson is not "never touch the model" or "trust the humans" — it is that the touch must clear a bar, and the bar is information the model provably lacks. The openings on the fall calendar clear it: they are structural breaks, announced, with sizes estimable from the earlier openings' ramps — Codes 10.13 through 10.15 size one. The holiday calendar clears it: the model is fitted at period 7 and cannot carry an annual pattern, so a benchmark estimated from the prior holiday and aligned on Thanksgiving is not opinion but arithmetic the model was structurally unable to do. The merchant's gifting thesis may clear it: it is a claim about a season the history barely samples, and it can enter as a named scenario assumption with its size argued from occasionwear mix. "The team feels good about Q4" does not clear it, and the discipline exists to say so out loud.

Two implementation refinements separate an override that carries information from an override that carries a mood, and both are lessons the labs learn the hard way. The first is shape. The prose of every planning deck describes new stores as ramping — finding their customers over weeks — and then the arithmetic adds a single flat number from the first day. That is not the stated business process; it assumes the store contributes its steady-state lift immediately, and it will overshoot the first week while undershooting the fourth. An override that matches the process estimates a profile — days 1 to 7, 8 to 14, 15 to 21, 22 to 28 — from prior openings, and adds the week-specific figure. The model can stay intentionally simple; what must not stay wrong is the mismatch between what the chapter says the business does and what the code does.

The second refinement is the evidence's provenance, and it has two halves. The first is contamination control. A before-and-after mean is the natural estimator and an uncontrolled one: the four weeks after an opening differ from the four weeks before it in season, promotions, weather, neighboring openings, and organic trend, and all of that lands in the estimate alongside the store. The labs therefore control what can be controlled and disclose what cannot. Equal-length windows keep the weekday mix balanced on both sides. Openings whose windows overlap another opening are excluded, because two steps in one window cannot be separated. Any opening whose evidence window *touches* November or December is excluded — not merely those whose own date falls there, since a late-October opening measures December in its fourth week just as surely. Promotional days are counted across the pre and post windows together, because a promotion sitting in the *before* window inflates the baseline and depresses the apparent step exactly as much as one in the *after* window exaggerates it. Estimates are stratified by store type where the count permits, since a flagship and a resort store have no reason to lift equally. Every surviving opening's own estimate is printed rather than averaged away, so the reader can see the spread; the override then carries a range rather than a point, which is the only honest form for an estimate built from a handful of observations; and the code asserts that at least one uncontaminated opening survives rather than quietly averaging an empty table.

The second half is which estimate the override is allowed to use, and this is where the chapter departs from the obvious answer. A trend-tracking method fitted on a series full of recent openings has already extrapolated part of the expansion into its own trend term. Adding the raw before-and-after step on top of that adds the same growth twice. So the labs compute the lift two ways for every surviving opening: the *raw* step, which is what the series did, and the *net* step, which is the post-opening actual minus what the fitted method itself forecast for those same days — the part the model demonstrably did not already carry. The net profile is the base adjustment, because it is the only one that cannot double-count. The raw profile is retained as an upper-bound sensitivity and becomes the upside scenario of Section 10.13's envelope. The gap between the two rows is not noise to be tidied away; it is a measurement of how much of the apparent step the model was already carrying, and the override log records it. Two consequences follow that students should expect rather than be surprised by. A net week-one adjustment can legitimately be negative, which is why the log says the forecast was *adjusted* rather than *raised*. And where

the net range across openings spans zero, the evidence has not established a lift at all, and the honest instrument is a named scenario rather than a point adjustment.

The instrument of the discipline is the written override, and its form is the point. Every adjustment to the statistical forecast is recorded with six elements: what changed (the periods and the amounts), the direction and size, the evidence (the announced opening; the prior-holiday benchmark; the estimated ramp from earlier openings, with its range and its net-of-trend figure), the owner (who is accountable for this adjustment, by name), the review date (when actuals will grade it), and the baseline preserved (the unadjusted statistical forecast, kept beside the adjusted one so the adjustment's contribution can be measured after the fact). The log is the forecasting form of disciplines the guide has installed twice already — the verification log of Chapter 4, the analytic specification of Section 2.5 — and it does the same job: it converts an invisible act of judgment into an auditable decision. It also quietly settles a distinction the planning meeting will otherwise blur, and the deliverable states it in one sentence: a forecast is a probability statement about what will happen; a target is an aspiration someone chose; a plan is a commitment of resources — and the moment a target is allowed to edit the forecast that justifies the plan, all three words mean nothing, per the measures-as-targets warning of Section 2.7.

CONCEPT

The Override Is Written, or It Is Not an Override

An undocumented adjustment is indistinguishable — to the meeting, to the January review, to the analyst herself — from optimism. The written override's six elements are not bureaucracy; they are what makes human knowledge a gradable input instead of an unfalsifiable mood. The log's deepest payoff arrives at review time: with the statistical baseline preserved beside the adjusted forecast, January can compute what each override contributed — the announced-opening ramp will, on the designed data, earn its keep; the enthusiasm will not — and the organization learns, in numbers, which of its judgments deserve to keep being made. That learning loop is the entire difference between a forecasting practice that improves and one that argues.

Source: Course concept developed for this guide, informed by Fildes et al. (2009).

The section closes where the chapter's decision lives: what the forecast is for. Three planning consumers take the same forecast and stress different parts of it. The open-to-buy envelope consumes the seasonal peak and its range — receipts must be committed against a December band, and the asymmetry of Section 2.7 governs which side to lean toward, since unsold inventory becomes markdown budget while stockouts become lost revenue and disappointed stores, and the two costs are not equal. The media pacing plan consumes the weekly shape — when demand arrives, so spend can lead it — and cares more about the seasonal profile's fidelity than the level's. And targets consume the forecast last and most dangerously, as the reality check against which aspiration is set; the deliverable's job there is to keep the words separate. One forecast, three uses, three different tolerances for error — which is why the deliverable of Section 10.13 reports accuracy by horizon and shape, not as one flattering number.

10.10 Forecasting Failure Modes

The toolkit is complete; this section names the three ways it most often fails in marketing practice, each demonstrated on StyleCraft's own series and each assigned its repair. They are collected here, just before the AI section, deliberately: every one of them is a failure an assistant will commit fluently, and Table 10.4 is the checklist the audit of Section 10.11 will run.

The first and costliest is extrapolating through a structural break — the failure Sections 10.4 and 10.6 built the vocabulary for. The mechanism: a method fitted on history that ends before (or straddles) a break forecasts the regime it learned, and the regime is gone. The StyleCraft demonstration is Code 10.12: train Holt-Winters with a cutoff just before a Wave-4 opening, forecast across it, and watch a well-fitted, baseline-beating model undershoot the post-opening weeks — not because the fitting failed but because the question was malformed; the model was asked to continue a past the company had already decided to end. The repair is never more fitting; it is the analyst's calendar (Section 10.4) and the documented override (Section 10.9) — or, where a break sits inside the training window, the harder judgment of which history still describes the current regime, since a model fed pre-break years is being fed evidence about a different company.

The second is overfit seasonality — Section 8.8's disease at the calendar's grain — and this chapter's design turns it from a warning into a demonstrated boundary. The mechanism: seasonal effects are parameters, one per period position, and estimating them from thin repetitions turns noise into rhythm. StyleCraft's weekly pattern is safe: about a hundred repetitions estimate seven effects generously. Its annual pattern is the trap: two Decembers of history — one of them mid-expansion, one of them containing a store opening — are two observations of a 365-position pattern, and a pipeline coaxed into fitting granular annual seasonality will faithfully reproduce whichever accidents those two Decembers contained, including the promotion spikes and the opening ramps that were never seasonal at all. The repair is structural modesty, and this chapter enacts it rather than recommending it: weekly seasonality is fitted; annual structure is never fitted, and is carried instead by an explicit calendar layer estimated from the prior holiday and aligned on the moving date, by documented scenarios that vary that layer across the two years of evidence available, and by overrides with owners. The standing reflex is the one question that catches this everywhere: how many times has the history actually seen this pattern?

The third is the ignored promotion, the quiet one. The mechanism: promotions move revenue on chosen dates, and a pipeline that never joins the campaigns table misfiles those movements — past spikes read as seasonality (if promotions recur near the same weeks) or as noise (if they do not), and future promotions are absent entirely. The forecast then fails twice: it carries phantom lift into weeks where no promotion is planned, and it misses real lift in the weeks where one is. The StyleCraft demonstration is Code 10.5, where the ten largest decomposition remainders are joined to the campaign calendar and the hit rate is compared against the share of days that are promotional — the null expectation, printed beside the count, so that the exhibit is a comparison rather than an impression. The repair at this guide's level is the calendar discipline of Section 10.4 and the override machinery of Section 10.9: promotions declared, past ones flagged in the

training window's reading and in the override's evidence windows, future ones entered as documented adjustments with sizes argued from past lifts.

Table 10.4

The three standing failure modes of marketing forecasting

Failure	Mechanism	StyleCraft demonstration	Repair
Extrapolating through a break	Methods continue the learned regime; breaks end it	Holt-Winters trained to a pre-opening cutoff undershoots the post-opening weeks (Code 10.12)	The analyst's calendar; a sized ramp override; choosing which history still applies
Overfit or absent seasonality	Seasonal effects are parameters; thin repetitions enshrine noise, and a period-7 model carries no annual pattern at all	The period-7 method misses the sealed holiday holdout by roughly twice its rolling-origin error, in one direction (Code 10.9)	Fit only well-repeated periods; carry annual and moving effects by an explicit calendar layer, benchmarks, and scenarios; count the repetitions
Ignored promotions	Chosen-date events misfiled as season or noise; future events absent	Campaign dates explain more of the largest remainders than chance predicts (Code 10.5)	Join the campaigns calendar; flag promotion days in every evidence window; enter future ones as sized, documented overrides

10.11 AI as a Forecasting Assistant

The division of labor this guide has refined across four chapters — the assistant drafts mechanics, the analyst audits meaning — meets its sternest test in forecasting, for a structural reason worth stating before the failure modes: forecasting is the first task in this guide where the standard tooling's defaults are wrong. The train_test_split habit, the shuffled cross-validation, the interpolation-friendly evaluation — the general-purpose machine learning reflexes an assistant has absorbed from a million repositories are calibrated for exchangeable rows, and time series data punishes every one of them. An assistant asked to "build a forecast and report accuracy" will, with real probability, deliver a pipeline that shuffles days into a random split (the leakage-by-design of Section 10.7, committed in one default argument), quotes a wonderful error number earned by interpolation, and narrates the result with congratulations. The pipeline will run. The number will be excellent. And the whole artifact will be the vendor button of Section 10.1 rebuilt at home.

The forecasting-specific failure modes, named for the audit. The shuffled split: random or k-fold evaluation applied to ordered data — caught in one line by the date-order check. The reused holdout: even with a genuine time-aware block, a pipeline that compares four methods on that block and then reports the winner's score on it has spent its own grade; the repair is rolling origins for selection and one sealed block for the verdict, and an assistant will not build that separation unless the prompt demands it. The mismatched season: a holdout chosen for convenience — the last eight weeks, whatever they are — rather than for resemblance to the decision, so a holiday plan is validated on late spring. The invented season: a seasonal period guessed at 12 on daily data, or granular annual seasonality fitted from two repetitions. The silent extrapolation: a fluent forecast straight through the announced openings and straight through December, because no prompt mentioned them and no period-7 method can find them. The benched-metric revival: MAPE quoted on a series with near-zero days, the denominator check never run. The summed interval: weekly and season-total bounds produced by adding daily bounds together, which is the arithmetic error of Section 10.8 and the one most likely to survive review because the numbers look plausible. The naked point: forecasts delivered without intervals, uncertainty unmentioned. And the method upgrade: asked for a forecast, the assistant reaches for the most impressive machinery it knows — auto-tuned black-box models the student cannot explain — which fails this course's standing rule that the analyst of record must be able to explain the method to the meeting that funds it, and which usually cannot beat seasonal-naive on two years of daily retail data anyway, per Section 10.5's M-competition humility.

Where the assistant genuinely helps, use it deliberately, and in forecasting the honest list is long: the groupby-and-reconcile scaffold of Code 10.3; decomposition and plotting code, which is fiddly and perfectly specifiable; the baseline computations and the fixed leaderboard table; the rolling-origin loop itself, which is tedious date arithmetic under frozen rules — exactly what Section 8.9's discipline makes safe to delegate; error tables by horizon; the path-aggregation arithmetic of Code 10.17, once the analyst has specified which order the operations go in; and the first draft of the interval-communication paragraph, audited hard before it ships. The governing instrument extends the audit lineage one more time: Table 8.5's five points run unchanged — frame, leakage, split hygiene, baseline, error in decision units — and Table 10.5 adds the four checks forecasting makes necessary. Together they are this chapter's verification theme made procedural, and the theme's distinctive move is stated in the box: predict the number before you run the pipeline, because the analyst who has computed the baselines already knows the plausible band, and a result outside the band — in either direction — is a finding about the pipeline, not the future.

Table 10.5

The forecasting supplement to the five-point audit (run with Table 8.5)

Supplemental point	**The check**	**Fails when**
S1. Time-aware evaluation	The date-order check: latest training date strictly before earliest evaluated date, at every forecast origin; no shuffling anywhere; no feature or smoother consumes post-cutoff data	Any split call shuffles; any window is centered; train and test dates interleave
S2. Selection separated from the grade	Method comparison runs on rolling origins inside the training data; the final holdout is opened once, for the frozen method, and is chosen to resemble the decision	Candidates were compared on the block that is later reported as the grade, or the holdout season does not match the decision season
S3. Calendar and breaks declared	The known calendar written down (Section 10.4); training-window breaks and promotions identified; forecast-window events matched to overrides or stated assumptions; annual and moving effects assigned to an explicit instrument rather than to a seasonal period	The forecast extrapolates silently through an announced event or a holiday season; a period-7 model is credited with annual seasonality
S4. Metric and interval honesty	Declared metric led; MAPE's denominators asserted and its minimum printed; error shown by horizon across several origins; intervals present, aggregated path-first for weekly and total figures, coverage reported beside width and period count	Metrics shuffle; MAPE hides a near-zero denominator; a point ships without its interval; weekly bounds are sums of daily bounds

AI IN PRACTICE

The Forecast You Priced Before You Ran It

The two-prompt pattern of Sections 8.11 and 9.11, with this chapter's twist: the prediction comes first, in writing, before the assistant runs anything.

Step one, the frame as specification: paste the series definition (daily revenue from the certified marketing_daily, reconciled per Code 10.3), the three windows with their dates (rolling-origin selection through October 31, 2025; sealed holdout November 1 to December 31, 2025; planning horizon July to December 2026), the declared metric (MAPE with asserted denominators and minimum printed, MAE beside it), the opponents (seasonal-naive at period 7, naive as floor), the seasonal structure to fit (weekly only; no annual fitting, under any circumstances), the known calendar (openings and promotions in every window, and the date of Thanksgiving in each year touched), and the instruction that closes the escape routes: "Compare candidates on rolling origins inside the selection window only; do not touch the sealed holdout until I name the method; report every candidate and both baselines in one table, by horizon week, with the number of errors behind each figure; produce 80% intervals and aggregate paths before quantiling for any weekly or total figure; narrate nothing yet."

Step two, the prediction: before running, write down the rolling-origin MAPE you expect for all four rows — you can compute the baselines' by hand from the series, and the model's should land between the seasonal-naive score and the designed noise floor. Then write down, separately, what you expect the sealed holiday holdout to do to that number, and why. Step three, run, then audit: Table 8.5's five points, then Table 10.5's four. Step four, only after survival: "Draft the plan paragraph: the December range under each named scenario, the margin over seasonal-naive in dollars, the overrides listed with owners and review dates." Audit its verbs — a forecast is expected, projected, likely between; it is never will be — file both exchanges per Appendix D, and sign the forecast as the analyst of record, interval and all.

Source: Course concept developed for this guide, informed by Hyndman and Athanasopoulos (2021) and Tashman (2000).

10.12 Hands-On Application in Python and Google Colab

The preceding sections built the vocabulary; this section spends it on the planning commission, in two labs that mirror the chapter's argument. Lab 10.1 makes every number touchable: the baselines and MAPE by hand at miniature scale — including the denominator failure, staged on purpose — then the certified series built, reconciled, decomposed, and audited against the campaign calendar at full scale. Lab 10.2 runs the honest evaluation: three declared windows, a rolling-origin comparison that chooses the method, a sealed holiday holdout that grades it once on a season shaped like the decision, accuracy by horizon estimated across origins, the shuffled-split demonstration, the inflection test that is this chapter's verification theme, a ramp override sized from controlled evidence and written down, intervals aggregated in the correct order, and

the actual holiday-season 2026 planning forecast with its scenarios and its declared planning unit.

The labs use the certified marketing_daily, transactions, stores, and campaigns files; column references follow the data dictionary. Three conventions hold across every cell. Every metric is computed by the helpers of Code 10.1, so no MAPE anywhere in the notebook is printed without its minimum denominator beside it. Every candidate method is a function of a training series and a horizon and nothing else, so the rolling-origin loop cannot accidentally hand one candidate information another did not get. And the notebook pins its environment. These cells are written for statsmodels 0.14.6, the current stable release, in which the simulation seed argument is spelled random_state. The development documentation for the forthcoming 0.15.0 release standardizes the argument as rng and deprecates the older name while keeping it working through a transition period, so a notebook run against that development branch will still execute and should be updated when it becomes a stable release. Pinning a stated version, which is what this chapter does, is what makes the figures reproduce exactly. The labs use statsmodels throughout (Seabold & Perktold, 2010). AI assistants may draft any code cell (Appendix C has templates; Appendix A covers Colab mechanics); every output is predicted before it is computed, and every exchange is documented per Appendix D.

10.12.1 Lab 10.1, Part A: Baselines and MAPE by Hand in Miniature

The miniature changes shape this chapter, as the data structure demands: not ten customers but fourteen days — two Monday-through-Sunday weeks of daily revenue, rounded to whole dollars for hand work and consistent with the designed scale of the certified series. Week one is the history; week two is the future to be forecast. The values carry the chapter's lessons in fourteen numbers: a weekly rhythm (quiet early weekdays, a Friday build, a weekend peak) and a gentle week-over-week climb. Before running anything, predict which rule forecasts week two better — the naive rule, which knows only yesterday, or the seasonal-naive rule, which knows the rhythm — and write one sentence of reasoning.

Code 10.1. Verify the baselines and the safe-MAPE helper

```
import pandas as pd

days = pd.date_range("2026-03-02", periods=14, freq="D")   # 2 weeks
rev = [8000, 7600, 7900, 8300, 9600, 12400, 11200,      # wk 1: history
       8400, 8100, 8200, 8900, 10100, 13000, 11900]     # wk 2: future
mini = pd.Series(rev, index=days, name="revenue")

def mae(actual, forecast):
    """Mean absolute error, in the series' own units."""
    actual = pd.Series(actual)
    forecast = pd.Series(forecast, index=actual.index)
    assert actual.notna().all() and forecast.notna().all()
    return (actual - forecast).abs().mean()

def mape(actual, forecast):
    """MAPE, with the denominator checked rather than assumed."""
    actual = pd.Series(actual)
    forecast = pd.Series(forecast, index=actual.index)
    assert actual.notna().all() and forecast.notna().all()
    assert (actual > 0).all(), (
        "MAPE is undefined at zero and unstable near it: report MAE "
        "and investigate the small denominators")
    return (actual - forecast).abs().div(actual).mean() * 100

def report(name, actual, forecast):
    """No MAPE is printed in this chapter without its denominator."""
    thin = int((actual < 0.25 * actual.median()).sum())
    print(f"{name:<24s} MAE {mae(actual, forecast):>9,.2f}"
          f"   MAPE {mape(actual, forecast):>7.2f}%"
          f"   min denom {actual.min():>9,.0f}   thin days {thin}")

week2 = mini.iloc[7:]
report("naive", week2, mini.shift(1).iloc[7:])
report("seasonal-naive", week2, mini.shift(7).iloc[7:])
```

Expected output: two lines, one per rule, each carrying MAE, MAPE, the minimum denominator behind that MAPE, and a count of thin days.

Input: fourteen literal days. Transformation: two zero-parameter forecast rules and three metrics. Output: the leaderboard of Section 10.5 at a scale where every figure can be checked by hand. The cell's real payload is the three helper functions, because they are used unchanged for the rest of the chapter and each encodes a discipline the prose argued for. mae asserts that nothing is missing before it averages. mape asserts that every denominator is strictly positive and refuses to return a number otherwise — the check of Section 10.7 made structural rather than remembered, which matters because the failure it prevents is silent. And report is the only printing route the labs use for accuracy, so the minimum denominator travels with every percentage automatically. A guide that told students to check denominators and then printed unguarded MAPEs for twenty cells would be teaching one thing and modeling another.

VERIFICATION CHECK

Before you run: compute both rules by hand. The naive rule forecasts each week-two day with the previous day's actual — its Monday forecast is Sunday's 11,200 — and its seven absolute errors (2,800; 300; 100; 700; 1,200; 2,900; 1,100) sum to 9,100, for an MAE of exactly 1,300.00; its percentage errors average to a MAPE of 12.79. The seasonal-naive rule forecasts each day with the same weekday last week, and its errors (400; 500; 300; 600; 500; 600; 700) sum to 3,600, for an MAE of 514.29 and a MAPE of 5.25. Predict also what the minimum-denominator column will show, and why it will reassure you here and alarm you in Code 10.2.

After you run: reconcile every figure, then read the anatomy of the defeat, because it is Table 10.2's first row demonstrated on fourteen numbers. The naive rule is not uniformly bad — midweek, where tomorrow does resemble today, its misses are tiny — but it is blindsided at every turn of the rhythm, forecasting Monday from Sunday's peak and Saturday from Friday's build, and the rhythm turns twice a week. The seasonal-naive rule inherits the rhythm free and loses only to the week-over-week climb: all seven of its errors are positive, a persistent undershoot of roughly five percent — bias, not noise — which is precisely the signature of a trend the rule has no way to carry, and precisely the margin a trend-tracking method exists to claim. Write the two sentences the miniature has earned: the weekly rhythm is worth about 786 dollars of daily MAE (1,300.00 against 514.29), and whatever beats seasonal-naive must earn its living from the climb.

Investigate if: your hand arithmetic disagrees with the printed figures anywhere. Every number in this cell is reproducible with a calculator, and a mismatch is arithmetic rather than convention.

The second half of the miniature stages the metric failure of Section 10.7. Suppose a storm closes the stores on week two's Tuesday and revenue collapses to 400 dollars. Rerun the seasonal-naive evaluation against the storm-struck actuals — predict both metrics' movements first.

Code 10.2. Demonstrate MAPE's denominator failure

```
snaive_fc = mini.shift(7).iloc[7:]
storm = week2.copy()
storm.iloc[1] = 400                        # the storm-closed Tuesday

report("seasonal-naive", week2, snaive_fc)
report("seasonal-naive, storm", storm, snaive_fc)

ape = (storm - snaive_fc).abs().div(storm) * 100
worst = ape.idxmax()
print(f"\nthe one term that produced the headline: {worst.date()}"
      f"  actual {storm[worst]:,.0f}"
      f"  forecast {snaive_fc[worst]:,.0f}  APE {ape[worst]:,.0f}%")
print(f"days forecast within 10%: {int((ape <= 10).sum())} of {len(ape)}")
```

Expected output: the clean and storm-struck seasonal-naive rows side by side, then the single date responsible for the headline with its absolute percentage error, and a count of days that were forecast within ten percent.

Input: the same fourteen days with one value replaced. Transformation: nothing but the substitution. Output: the most instructive pair of numbers in the chapter. The MAE moves honestly, from 514.29 to 1,471.43, because the forecast genuinely missed a collapsed day by a lot. The MAPE does something else: the Tuesday term is 7,200 divided by 400, or 1,800 percent, one term of a seven-term average, and the reported figure explodes from 5.25 to 261.52. Verify the arithmetic by hand — the sum of the seven percentage errors divided by seven — and then note what the guarded helper did and did not do. It did not refuse: 400 is positive, so the assertion passes and the number is computed. What it did was print the minimum denominator and the thin-day count beside the number, which is the difference between a metric that lies and a metric that lies while showing you why.

VERIFICATION CHECK

Before you run: predict both metrics' movements in direction and rough size. Then predict the thin-day count, and state what value of the storm Tuesday would have made the mape helper stop rather than compute.

After you run: write the sentence that must accompany every MAPE this pipeline ever reports — the denominator, not the forecast, produced the number. Six of the seven days were forecast within ten percent; the headline says the method missed by two hundred sixty. Then write the honest one-line summary of the storm week that the standing rule requires: the MAE moved from 514 to 1,471, one day closed, and MAPE is disqualified for this week.

Investigate if: you are tempted to fix this by dropping the storm day. Say out loud what that would mean — deleting the largest genuine forecast miss in the window because it embarrasses a chosen metric — and then reach for the metric that survives instead of editing the data that does not flatter it.

10.12.2 Lab 10.1, Part B: The Daily Series, Reconciled, Decomposed, and Audited

Part B manufactures the real object. The certified marketing_daily ships the daily series; the known-truth discipline requires rebuilding its revenue column from the fact table before trusting it — the grain change of Section 3.3, executed with Chapter 4's groupby, cited and not re-taught. Two mechanical traps sit in the reconciliation and the draft of this chapter fell into both, so read the cell for them specifically.

Code 10.3. Build and reconcile the daily series

```
import numpy as np

md = pd.read_csv("marketing_daily.csv", parse_dates=["date"])
md = md.set_index("date").sort_index().asfreq("D")

# asfreq inserts missing calendar days as NaN; it does not make them
# zero-revenue days. A quiet day is a zero, not a hole (Section 10.2) --
# so fill it explicitly and state the rule. Where a gap instead means an
# incomplete extract, stop rather than filling silently.
gaps = md["revenue"].isna()
print("calendar days with no source row:", int(gaps.sum()))
assert gaps.sum() <= 3, "too many gaps to be closures -- audit it"
md.loc[gaps, "revenue"] = 0.0

tx = pd.read_csv("transactions.csv", parse_dates=["order_date"])
daily_tx = (tx.groupby("order_date")["line_revenue"].sum()
              .reindex(md.index, fill_value=0.0)   # quiet day = 0
              .rename("revenue_from_tx"))

assert md.index.is_monotonic_increasing and md.index.is_unique
assert md["revenue"].notna().all()
assert np.isclose(md["revenue"], daily_tx, atol=0.01).all(), (
    "the shipped series disagrees with the transaction roll-up")

series = md["revenue"].rename("revenue")
print("days:", len(series), " span:", series.index.min().date(),
      "->", series.index.max().date())
print(f"zero-revenue days: {int((series == 0).sum())}"
      f"   minimum: {series.min():,.0f}"
      f"   median: {series.median():,.0f}")
```

Expected output: the count of calendar days with no source row, then the series length, span, zero-revenue count, minimum, and median; three assertions pass silently.

Input: the certified daily table and the transaction fact table. Transformation: a complete-calendar reindex, an explicit zero-fill rule, a groupby roll-up reindexed onto the same calendar, and an element-wise comparison. Output: series, the certified daily object every later cell uses, plus the evidence that it means what Section 10.2 says it means.

The two traps are worth naming because they are silent. First, asfreq builds a complete daily index and inserts the missing days as blanks, not as zeros — so a series that "has no gaps" after an asfreq call may be full of blanks that every later arithmetic operation quietly propagates or skips. The rule from Section 10.2 has to be executed: a day with no transactions is a zero, so fill it and say so; a run of blanks too long to be closures is an incomplete extract, so stop. The cell does both, with the threshold visible in the assertion rather than buried in a comment. Second, the transaction roll-up omits dates on which nothing was sold, so comparing it to the daily table by index alignment compares two differently shaped objects and can report agreement on the intersection while ignoring exactly the days in dispute. Reindexing the roll-up onto the series' own calendar with fill_value=0.0 forces both sides onto the same days before anything is

compared, and the closeness check then runs on every day in the series rather than on whichever days both objects happened to contain.

VERIFICATION CHECK

Before you run: write three predictions. The span: the certified history runs July 2024 through June 2026, so predict roughly 730 rows. The reconciliation: the closeness assertion should pass on every day; anything less is the lab's first exercise in aggregation forensics — find one disagreeing date and determine which side dropped or double-counted a line, per the join disciplines of Chapter 4. The zero count: predict it near zero for a chain with an always-on online channel, and write down what a nonzero count would oblige you to investigate.

After you run: confirm the assertions passed and record the minimum daily revenue in your notes, because it is the number that licenses MAPE for the rest of the chapter. Then state, in one sentence, what would change about the whole chapter if the minimum were 40 dollars rather than several thousand.

Investigate if: the gap count is above zero. Before filling anything, look at the dates: a scatter of isolated days is plausibly closures, a contiguous run at either end of the file is a truncated extract, and the two require opposite responses.

Code 10.4. Decompose the series and mark the openings

```
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

RESERVED = {"S00", "S99"}        # Online and App are not physical stores
stores = pd.read_csv("stores.csv", parse_dates=["opening_date"])
opens = (stores[~stores["store_id"].isin(RESERVED)]
         .dropna(subset=["opening_date"])
         .sort_values("opening_date"))
in_window = opens[opens["opening_date"]
                  .between(series.index.min(), series.index.max())]

dec = seasonal_decompose(series, model="additive", period=7)

fig, ax = plt.subplots(4, 1, figsize=(9, 9), sharex=True)
panels = [(series, "observed"), (dec.trend, "trend"),
          (dec.seasonal, "seasonal, period 7"), (dec.resid, "remainder")]
for a, (part, label) in zip(ax, panels):
    a.plot(part.index, part.to_numpy(), lw=0.8)
    a.set_ylabel(label, fontsize=9)
for d in in_window["opening_date"]:      # the calendar HW cannot read
    for a in ax[:2]:
        a.axvline(d, color="#57068C", lw=0.8, ls="--")
ax[1].set_title("dashed: store openings from the stores table",
                fontsize=9, loc="left")
fig.tight_layout()

print("openings inside the series window:", len(in_window))
print(in_window[["store_name", "store_type", "opening_date"]]
      .to_string(index=False))
swing = dec.seasonal.max() - dec.seasonal.min()
print(f"\nweekly peak-to-trough: {swing:,.0f}"
      f"  ({swing / series.mean():.0%} of mean daily revenue)"
      f"  <- what seasonal-naive carries free")
```

Expected output: a four-panel decomposition figure with the store-opening dates drawn as dashed vertical lines on the observed and trend panels, then the count and roster of openings inside the series window, then the weekly seasonal swing in dollars and as a share of mean daily revenue.

Input: the certified series and the stores roster. Transformation: an additive decomposition at period 7, and a filter that removes the reserved non-physical locations before anything is plotted. Output: the reading instrument of Section 10.3, with the calendar the model cannot read drawn on top of it. Two details are deliberate. The reserved set is removed by store identifier rather than by hoping that Online and App carry a null opening date, because a roster row that acquires a date later would otherwise silently enter every opening analysis in the chapter. And the openings are marked on the plot rather than described in a caption, because the whole argument of Section 10.4 is that the steps land where the calendar says they do, and a reader who has to take that on trust has not seen the evidence.

The seasonal swing printed at the end is the number Section 10.5 promised: the peak-to-trough height of the weekly pattern is, to a first approximation, exactly what the seasonal-naive rule carries for free and what any fitted model must therefore stop congratulating itself for.

VERIFICATION CHECK

Before you run: sketch what you expect from the designed story — an upward drift, visible stair-steps at the opening dates, two November–December surges, a late-August lift, and a weekly sawtooth over all of it. Predict where in the four panels each of those five features will appear, and name the one that will appear in the remainder panel because a period-7 decomposition has nowhere else to put it.

After you run: write one sentence per panel. Confirm that the trend panel's steps line up with the dashed opening lines rather than merely resembling them. Confirm that the seasonal panel is a stable seven-day sawtooth across both years, and record its dollar height. Then look hard at the remainder panel around each November and December, and say what the presence of large remainders there proves about what a period-7 seasonal component does and does not carry — this is the visual form of the boundary Section 10.4 declared and Code 10.9 will grade.

Investigate if: the trend panel's steps do not align with the opening dates, or if the seasonal panel drifts in amplitude across the two years. The first suggests a roster date that does not match the series; the second suggests the additive assumption is straining and a multiplicative decomposition is worth reading beside it.

Code 10.5. Audit the remainder against the campaign calendar

```
campaigns = pd.read_csv("campaigns.csv",
                        parse_dates=["start_date", "end_date"])
promo_days = pd.DatetimeIndex(np.unique(np.concatenate(
    [pd.date_range(r.start_date, r.end_date, freq="D").to_numpy()
     for r in campaigns.itertuples()]))).intersection(series.index)

top10 = dec.resid.abs().nlargest(10).index.sort_values()
audit = pd.DataFrame({
    "date": top10.date,
    "revenue": series.loc[top10].round(0).to_numpy(),
    "remainder": dec.resid.loc[top10].round(0).to_numpy(),
    "in_promo": top10.isin(promo_days)})
audit["campaign"] = [
    ", ".join(campaigns.loc[(campaigns["start_date"] <= d)
                            & (campaigns["end_date"] >= d),
                            "campaign_name"].unique()) or "--"
    for d in top10]
print(audit.to_string(index=False))

hits, share = int(audit["in_promo"].sum()), len(promo_days) / len(series)
print(f"\n{hits} of the 10 largest remainders sit in a campaign window")
print(f"promotion days are {share:.0%} of the series, so chance alone "
      f"predicts about {10 * share:.1f} of 10")
print("unexplained days, for the override watch list:",
      [str(d) for d in audit.loc[~audit["in_promo"], "date"]])
```

Expected output: a ten-row table of the largest absolute remainders with revenue, remainder, a promotion flag, and the campaign names covering that date; then the promotion hit count, the share of all days that are promotional as the chance expectation, and the list of unexplained dates.

Input: the decomposition remainder and the campaigns table. Transformation: a date-expanded promotion calendar, a top-ten selection by absolute remainder, and a join. Output: Section 10.10's third failure mode caught before any forecast has been run, and the override log's opening watch list.

The line that makes this an exhibit rather than an impression is the chance expectation. If a fifth of all days sit inside some campaign window, then two of any ten days will land inside one by accident, and a hit count of three proves nothing. Printing the null beside the count is the difference between "promotions explain the spikes" as a finding and as a feeling — the same discipline Section 10.5 applies to model accuracy, applied here to a pattern claim. The unexplained dates are then genuinely interesting: they are the days the decomposition could not explain and the campaign calendar cannot either, and on StyleCraft's designed data they cluster where you would expect once you have read Code 10.4's remainder panel.

10.12.3 Lab 10.2, Part A: Three Windows and the Opponents

Lab 10.2 opens, per the standing rule, with the frame in a markdown cell — the predictive frame of Section 8.2 wearing this chapter's clothes. Unit of prediction: one prediction per future day, rolled up to declared weekly blocks for the deliverable. Target: daily chain revenue as reconciled in Code 10.3. Information: the series' own past up to each forecast origin, plus the declared calendar (openings from the stores table; promotions from the campaigns table; the date of Thanksgiving in each year the analysis touches). Declared metric: MAPE with asserted denominators and minimum printed, MAE beside it, error shown by horizon week. Opponents: seasonal-naive at period 7, naive as floor. Seasonal structure to fit: weekly only. And then the part that is new to this chapter and is the whole point of Table 10.3 — not one holdout but three windows, with different rights.

Code 10.6. Declare the three evaluation windows

```
# THE FRAME IN CODE. Three windows, declared before anything is fitted.
SERIES_END    = series.index.max()                       # 2026-06-30
SELECT_END    = pd.Timestamp("2025-10-31")   # rolling-origin selection
HOLDOUT_START = pd.Timestamp("2025-11-01")   # sealed holiday holdout
HOLDOUT_END   = pd.Timestamp("2025-12-31")
PLAN_START    = pd.Timestamp("2026-07-01")   # the actual planning horizon
PLAN_END      = pd.Timestamp("2026-12-31")

H = 56              # rolling-origin forecast length, in days
MIN_TRAIN = 240     # shortest training window any candidate may see
ORIGIN_STEP = 28    # days between successive forecast origins
MAE_TOLERANCE = 0.05   # the selection rule's practical-tie band

select = series.loc[:SELECT_END]
holdout = series.loc[HOLDOUT_START:HOLDOUT_END]

assert select.index.max() < holdout.index.min()   # date-order check S1
assert holdout.index.min() == HOLDOUT_START
assert holdout.index.max() == HOLDOUT_END
assert PLAN_START == SERIES_END + pd.Timedelta(days=1)

origins = pd.date_range(select.index.min() + pd.Timedelta(days=MIN_TRAIN),
                        SELECT_END - pd.Timedelta(days=H),
                        freq=f"{ORIGIN_STEP}D")
assert len(origins) >= 5, "too few origins to average over"

print(f"selection : {select.index.min().date()} -> {SELECT_END.date()}"
      f"   {len(select)} days, {len(origins)} origins, H={H}")
print(f"holdout   : {HOLDOUT_START.date()} -> {HOLDOUT_END.date()}"
      f"   {len(holdout)} days -- a holiday season, like the decision")
print(f"plan      : {PLAN_START.date()} -> {PLAN_END.date()}"
      f"   {(PLAN_END - PLAN_START).days + 1} days, no actuals exist")
print("origins:", ", ".join(str(o.date()) for o in origins))
```

Expected output: three lines describing the selection, holdout, and planning windows with their lengths, then the list of forecast origins; four assertions pass silently.

Input: the certified series. Transformation: date declarations and an origin schedule. Output: the objects every later cell in Lab 10.2 consumes, and the assertions that make the design auditable in the notebook rather than asserted in a methods paragraph. The date-order assertion is supplement point S1 made executable, and the lab requires it in every notebook this chapter grades. The assertion that the planning horizon begins the day after the series ends is the smaller and more easily broken one: it is what stops a "future" forecast from silently overlapping history that already exists, which is the version of leakage that survives every other check because the dates look tidy.

Read the three windows against Table 10.3 and say what each may and may not do. The selection window may be consulted as often as the analyst likes, because it is training data. The holdout may be consulted once, for one frozen method, and its date range was chosen for a reason no convenience-based holdout can claim: it is a holiday season, forecast from before it, which is the shape of the decision. The planning window may not be graded at all, because no

actuals exist — and the honest label on it is not "forecast" but "forecast plus declared calendar plus stated assumptions," which is what Sections 10.9 and 10.13 spend their time enforcing.

Code 10.7. Declare the candidates, the model factory, and the selection rule

```
from statsmodels.tsa.holtwinters import ExponentialSmoothing

# ONE place a model is fitted. Every later cell that needs a fitted
# object calls this with CHOSEN, so the method that wins the comparison
# is the method that produces the intervals and the planning forecast.
MODEL_CONFIG = {"Holt-Winters": False, "Holt-Winters, damped": True}


def fit_candidate(name, train):
    """None for the zero-parameter baselines, which fit nothing."""
    if name not in MODEL_CONFIG:
        return None
    return ExponentialSmoothing(
        train, trend="add", damped_trend=MODEL_CONFIG[name],
        seasonal="add", seasonal_periods=7,
        initialization_method="estimated").fit()


def fit_forecast(name, train, h):
    """Every candidate is a function of (train, h) and nothing else."""
    if name == "naive":
        return np.repeat(float(train.iloc[-1]), h)
    if name == "seasonal-naive":
        last = train.iloc[-7:].to_numpy()
        return np.tile(last, int(np.ceil(h / 7)))[:h]
    return fit_candidate(name, train).forecast(h).to_numpy()


CANDIDATES = ["naive", "seasonal-naive", "Holt-Winters",
              "Holt-Winters, damped"]
SIMPLICITY = {n: i for i, n in enumerate(CANDIDATES)}   # tiebreak order
print(f"candidates: {len(CANDIDATES)}   practical-tie band: "
      f"{MAE_TOLERANCE:.0%} of the best rolling MAE")
print("""selection rule, declared before any fit: among the candidates
inside the tie band on rolling MAE, take the lowest week-8 MAE, then
the smallest absolute bias, then the simpler method.""")
```

Expected output: the candidate count, the declared tie band, and the selection rule printed in full; nothing is fitted here.

Input: none. Transformation: two declarations and two function definitions. Output: the single place in the chapter where a model is fitted, and the rule that will name the winner — both written before any evidence exists.

This cell exists because of a defect the first revision of this chapter still carried, and the defect is worth naming because it is the most common way a careful forecasting notebook quietly cheats. The comparison selected a method; the interval and the planning forecast were then built by a separately typed Holt-Winters call. Whenever the comparison happened to choose something else — the damped variant, or a baseline — the deliverable silently came from a

method that had never won anything. The repair is structural: fit_candidate is the only route to a fitted object, every later cell calls it with CHOSEN, and the baselines return None because they genuinely fit nothing, which forces the interval cell to declare what it will do when a zero-parameter rule wins rather than substituting a convenient model that can simulate.

The selection rule is the second declaration, and it is precommitted for exactly the reason Section 8.9 gave: a rule written after the leaderboard is visible can be written to produce the preferred winner. The rule states a practical-tie band on rolling MAE, then breaks ties on the two columns the prose says the plan cares about — the week-eight grade the December weeks will consume, and the absolute bias that says whether a method is systematically low — and prefers the simpler method last. Note what the band buys: a candidate that wins the pooled average by a fraction of a percent no longer wins automatically, which is the numeric form of the chapter's own warning that a margin smaller than the noise is not yet a finding.

Code 10.8. Run the rolling-origin comparison and apply the rule

```
rows = []
for name in CANDIDATES:
    e_all, ape_all, wk8, signed = [], [], [], []
    for origin in origins:
        train = series.loc[:origin]
        test = series.loc[origin + pd.Timedelta(days=1):][:H]
        e = test.to_numpy() - fit_forecast(name, train, H)
        e_all.append(np.abs(e))
        ape_all.append(np.abs(e) / test.to_numpy() * 100)
        wk8.append(np.abs(e)[-7:])
        signed.append(e)
    rows.append({"candidate": name,
                 "rolling MAE": np.concatenate(e_all).mean(),
                 "MAPE %": np.concatenate(ape_all).mean(),
                 "week-8 MAE": np.concatenate(wk8).mean(),
                 "bias": np.concatenate(signed).mean()})

board = pd.DataFrame(rows).set_index("candidate").round(2)
print(f"{len(origins)} origins x {H} days = {len(origins) * H} "
      "daily errors per candidate\n")
print(board.to_string())

best = board["rolling MAE"].min()
band = best * (1 + MAE_TOLERANCE)
eligible = board[board["rolling MAE"] <= band].copy()
eligible["abs bias"] = eligible["bias"].abs()
eligible["simplicity"] = [SIMPLICITY[i] for i in eligible.index]
CHOSEN = eligible.sort_values(
    ["week-8 MAE", "abs bias", "simplicity"]).index[0]

col = board["rolling MAE"]
print(f"\nwithin the {MAE_TOLERANCE:.0%} tie band: {len(eligible)} of "
      f"{len(board)} candidates -> {list(eligible.index)}")
win = board.loc[CHOSEN]
print(f"SELECTED: {CHOSEN}   week-8 MAE {win['week-8 MAE']:,.0f}"
      f"   bias {win['bias']:+,.0f}")
print(f"margin over the opponent: "
      f"{col.loc['seasonal-naive'] - col.loc[CHOSEN]:,.0f} of daily MAE."
      "\nThe sealed holdout is still closed.")
```

Expected output: the number of daily errors behind each candidate, a four-row table carrying rolling-origin MAE, MAPE, week-eight MAE and mean signed error for naive, seasonal-naive, Holt-Winters and damped-trend Holt-Winters, then the selected method, its margin over the opponent in daily MAE, and a statement that the sealed holdout is still closed.

Input: the selection window only. Transformation: for each candidate and each origin, fit on everything up to the origin and forecast the next fifty-six days, then pool the errors. Output: the leaderboard that names the method — and the only evidence the analyst is permitted to use in naming it.

Two design choices carry the section's arguments. Every candidate is expressed as one function of a training series and a horizon, including the two baselines, so that "same origins, same metric, same baselines" is enforced by the code rather than remembered by the analyst —

the direct time-series analogue of Chapter 8's insistence that a baseline be an estimator scored by the same call. And the table carries four columns rather than one, because a single pooled MAE hides the two things the plan cares about: the week-eight column is the far-horizon grade the December weeks will actually consume, and the bias column is the signed error, which is where a trending, expanding series announces whether a method is systematically low. Those two columns are not decoration — the rule declared in Code 10.7 sorts on them, so a candidate that wins the pooled average inside the tie band but carries a worse far-horizon grade does not win the chapter.

VERIFICATION CHECK

Before you run: predict the ordering of all four candidates on rolling-origin MAE, and predict the sign of the bias column for each. Then predict how many candidates will fall inside the declared tie band, and say what you will conclude if the answer is more than one. Write down the band you would consider implausible in either direction — far worse means a broken fit, implausibly better means the evaluation is dirty.

After you run: grade every prediction. Confirm the arithmetic behind the error count: origins times horizon length, which is the number this table's stability rests on. Then read the bias column across all four rows and write the sentence it forces — every candidate that cannot climb is biased low on an expanding series, which is a fact about the business, not about the software. State the winner's margin in dollars per day and in dollars across a fifty-six-day window, because the second is the sentence that survives into the deliverable.

Investigate if: the damped variant wins, or if naive beats seasonal-naive. The designed expectation is that the undamped Holt-Winters wins on this data, but a different result is a finding to trace, not an error to edit away. A damped winner says the series' recent trend is not projecting well and the plan should expect flatter growth than the raw trend implies — and it is precisely the case the model factory of Code 10.7 exists to carry safely into the interval and the planning forecast. Naive beating seasonal-naive would say the weekly rhythm has broken down, which on this data would mean the series or the reconciliation is wrong, and Code 10.3 is where you go. If a baseline wins outright, read the interval cell's fallback branch before doing anything else.

10.12.4 Lab 10.2, Part B: The Sealed Holiday Holdout

The method is now frozen. Part B opens the sealed window once, and it does so with one addition declared before the window opens: the calendar layer. Section 10.4 established that a period-7 model carries no annual pattern, and the honest response is not to hope but to supply one. The layer is estimated entirely inside the training data by running the same evaluation a year earlier — fit through October 2024, forecast the 2024 holiday, and record the ratio of what actually happened to what the method said, day by day. That ratio is what the weekly model misses in a holiday, measured rather than assumed. And it is indexed by days from Thanksgiving rather than by calendar date, because Thanksgiving moved between the two years and a day-of-year index would misalign the Black Friday weekend by a full day in one direction.

Code 10.9. Open the sealed holiday holdout, once

```
def thanksgiving(year):
    """Fourth Thursday in November -- a date the calendar moves."""
    first = pd.Timestamp(f"{year}-11-01")
    first += pd.Timedelta(days=(3 - first.dayofweek) % 7)
    return first + pd.Timedelta(days=21)

def holiday_index(s, year, method):
    """Actual over statistical forecast, day by day, across one prior
    holiday -- indexed by DAYS FROM THANKSGIVING so that the moving
    retail events line up instead of drifting by weekday."""
    cut = pd.Timestamp(f"{year}-10-31")
    actual = s.loc[f"{year}-11-01":f"{year}-12-31"]
    fc = pd.Series(fit_forecast(method, s.loc[:cut], len(actual)),
                   index=actual.index)
    idx = (actual / fc).rolling(7, center=True, min_periods=4).mean()
    idx.index = (actual.index - thanksgiving(year)).days
    return idx

IDX24 = holiday_index(select, 2024, CHOSEN)
print("Thanksgiving:", ", ".join(f"{y} {thanksgiving(y).date()}"
                                 for y in (2024, 2025, 2026)))
print(f"2024 index: min {IDX24.min():.2f}  peak {IDX24.max():.2f}"
      f" at day {IDX24.idxmax():+d}")

train, h = select, len(holdout)
fc_stat = pd.Series(fit_forecast(CHOSEN, train, h), index=holdout.index)
sn_fc = pd.Series(fit_forecast("seasonal-naive", train, h),
                  index=holdout.index)
off25 = pd.Series((holdout.index - thanksgiving(2025)).days,
                  index=holdout.index)
fc_cal = fc_stat * off25.map(IDX24).ffill()

print("\n--- SEALED HOLDOUT OPENED ONCE: Nov 1 - Dec 31, 2025 ---")
for name, fc in [("seasonal-naive", sn_fc), (CHOSEN, fc_stat),
                 (f"{CHOSEN} + holiday calendar", fc_cal)]:
    report(name, holdout, fc)
    bias = (holdout - fc).mean()
    print(f"{'':<24s} signed error {bias:>+9,.0f}"
          f"   season total miss "
          f"{(fc.sum() - holdout.sum()) / holdout.sum():>+7.1%}")
```

Expected output: the Thanksgiving dates for 2024, 2025 and 2026; the 2024 holiday index's minimum, peak, and the offset from Thanksgiving at which the peak falls; then three graded rows — seasonal-naive, the frozen method, and the frozen method with the holiday calendar — each with MAE, MAPE, minimum denominator, mean signed error, and the season-total miss as a percentage.

Input: the selection window and the sealed holdout. Transformation: one calendar layer estimated from the prior holiday, and three forecasts scored on the sealed window. Output: the chapter's central grade, and the exhibit that proves the boundary of Section 10.4 rather than asserting it.

Read the three rows in order and the design explains itself. Seasonal-naive repeats late October forever and is destroyed by the season. The frozen Holt-Winters, which beat every

candidate on rolling origins, misses the holiday by roughly twice its rolling-origin error and misses it in one direction — a large positive signed error, which is to say a systematic undershoot, which is to say exactly what a model that carries a weekly rhythm and no annual pattern must do when December arrives. Adding the calendar layer cuts the miss substantially and, on this data, overshoots slightly in the other direction, because the single prior holiday available for estimating it contained a store opening of its own and therefore overstates the uplift a little. Every part of that sentence belongs in the deliverable. An override estimated from one observation is an assumption with a range, and the direction of its error on the one occasion it could be graded is the most useful thing anyone knows about it.

VERIFICATION CHECK

Before you run: this is the prediction the chapter has been building toward, so write it in full sentences. Predict the sign of the signed-error column for seasonal-naive and for the frozen method, and say why both signs are forced rather than likely. Predict how the sealed-holdout MAPE compares to the rolling-origin MAPE of Code 10.8, and by roughly what factor. Then predict whether the calendar layer will over- or under-correct, and name the contaminating event in the 2024 holiday window that decides the answer — the roster from Code 10.4 has it.

After you run: confirm the signs, then write the two sentences the deliverable needs. First: the method chosen by rolling origins on non-holiday windows is graded, on the season the plan is actually about, at a materially worse error than its selection score — so the selection score may not be quoted as the plan's expected accuracy. Second: the gap between those two numbers is not a defect in the model but the measured size of the calendar the model was never given, which is why the analyst's layer is a component of the forecast rather than a garnish on it. Then record the calendar layer's own direction of error, with its cause, in the override log's evidence field.

Investigate if: the calendar layer makes things worse rather than better, or if the frozen method's signed error is negative. The designed expectation is a large positive signed error for both baseline and model, and a calendar layer that reduces the miss; a different result is a finding to trace, not an error to edit away. A layer that makes things worse would say the prior holiday is too contaminated to serve as a benchmark, and the response is a scenario range rather than a point adjustment. A negative signed error would mean the model is over-forecasting the holiday, which on a period-7 fit would be surprising enough to send you back to the trend term and to Code 10.4's trend panel.

Code 10.10. Measure accuracy by horizon across origins

```
rows = []
for origin in origins:
    train = series.loc[:origin]
    test = series.loc[origin + pd.Timedelta(days=1):][:H]
    e = np.abs(test.to_numpy() - fit_forecast(CHOSEN, train, H))
    ape = e / test.to_numpy() * 100
    block = np.arange(H) // 7 + 1        # eight blocks of exactly 7 days
    for w in range(1, H // 7 + 1):
        m = block == w
        rows.append({"week_ahead": w, "MAE": e[m].mean(),
                     "MAPE": ape[m].mean()})

by_h = (pd.DataFrame(rows).groupby("week_ahead")
        .agg(origins=("MAE", "size"), MAE=("MAE", "mean"),
             MAPE=("MAPE", "mean"), MAE_sd=("MAE", "std")).round(2))
by_h["daily_errors"] = by_h["origins"] * 7      # 7 days per weekly block
print(by_h.to_string())
rise = by_h["MAE"].iloc[-1] / by_h["MAE"].iloc[0]
print(f"\nweek 8 MAE is {rise:.2f}x week 1's")
print("monotone across all eight weeks:",
      bool((by_h["MAE"].diff().dropna() > 0).all()),
      "  <- expected error rises; realized error need not, every step")
```

Expected output: an eight-row table giving, per week ahead, the number of forecast origins behind the figure, the mean MAE and MAPE, the standard deviation of MAE across origins, and the implied count of daily errors; then the week-eight to week-one ratio and a flag stating whether the increase was monotone.

Input: the selection window's origins. Transformation: fifty-six-day forecasts from each origin, cut into eight consecutive seven-day blocks, averaged by block across origins. Output: the horizon curve of Section 10.7, estimated the way it has to be estimated.

Three properties of this table are corrections to the draft of this chapter, and each is worth reading for. The blocks are exactly seven days counted from the forecast origin, not calendar weeks from a resample call, so no block is a partial week and no bin boundary is decided by a library default. Two count columns appear rather than one, and the distinction matters: origins is the number of independent forecast paths behind the row, and daily_errors is that number times seven, because each origin contributes a whole week of daily errors to its weekly MAE. Quoting the larger figure as if it were the number of independent observations would overstate the row's stability sevenfold; quoting only the smaller one would understate how much data the mean rests on. The table prints both. And the monotone flag is printed rather than assumed, because the honest claim is that expected error and uncertainty grow with horizon while realized error in a finite sample need not rise at every step. On StyleCraft's data it does not: some middle week will sit below the week before it, and the standard-deviation column shows why — the spread across origins at a given horizon is comparable to the differences between adjacent horizons.

VERIFICATION CHECK

Before you run: predict the direction of the MAE column and, separately, predict whether it will rise at every single step. Predict the week-eight to week-one ratio to within a factor of two. Then write down what you will conclude if a middle week dips.

After you run: grade both predictions and write the sentence the plan needs — the December-distance weeks carry roughly the week-seven and week-eight grade, not the headline average. Then find the calendar event sitting under any week that spikes or dips out of pattern, using the promotion calendar from Code 10.5 and the roster from Code 10.4, and record whether it is a horizon effect or a calendar effect wearing one.

Investigate if: the curve is flat or falling. On this data that would suggest the origins are too close together to be informative, or that the horizon is short relative to how fast this series actually changes; widen the origin spacing or lengthen the horizon and report what you changed.

Code 10.11. Demonstrate the shuffled-split failure

```
rng = np.random.default_rng(42)
hidden = rng.choice(series.index[7:-7], size=100, replace=False)

interp = (series.shift(7)[hidden].to_numpy()
          + series.shift(-7)[hidden].to_numpy()) / 2   # uses the FUTURE
actual = series[hidden]
fake = np.abs(actual.to_numpy() - interp) / actual.to_numpy()

print(f"random-day 'holdout' MAPE : {fake.mean() * 100:>6.2f}%"
      "   <- means nothing")
roll = board.loc[CHOSEN, "MAPE %"]
print(f"rolling-origin MAPE       : {roll:>6.2f}%   <- graded fairly")
print(f"sealed holiday holdout    : "
      f"{mape(holdout, fc_stat):>6.2f}%   <- the decision's own season")
print("\nthe mechanism in one line: shift(-7) hands the evaluation next")
print("week's revenue, which no deployed forecast possesses.")
```

Expected output: three MAPE figures on one screen — the shuffled pseudo-holdout, the honest rolling-origin score, and the sealed holiday score — followed by the two-line statement of the mechanism.

Input: the whole series. Transformation: one hundred randomly hidden days, each "predicted" from the seasonal lag on both sides. Output: the number an unframed pipeline would have reported, printed beside the two numbers that mean something.

The comparison is the lesson, and the three-number layout is the improvement over the draft, which showed the shuffled score alone. Placing all three side by side makes the whole chapter's evaluation argument visible in six lines of output: the shuffled score flatters, the rolling-origin score is honest about the average window, and the sealed holiday score is honest about the window the decision lives in — and they are, in that order, increasingly large and increasingly relevant. State the mechanism in writing before reading the numbers: the backward-looking term

hands the evaluation next week's revenue, information no deployed forecast possesses, and each hidden day is being interpolated between a known past and a known future.

> **VERIFICATION CHECK**
>
> Before you run: predict where the shuffled figure will fall relative to the other two, and commit to the prediction in writing, because the point of the cell is lost if you look first.
>
> After you run: write the audit sentence for the vendor meeting and the AI round trip alike — any forecast accuracy claim earned under shuffling is this number wearing a straight face — and then name the one-line test that unmasks it. Then write the sharper second sentence this three-way layout earns: even an honest evaluation flatters a seasonal plan if its window is chosen for convenience, which is why the holdout in this chapter was chosen for resemblance.
>
> Investigate if: the shuffled figure is not the smallest of the three. That would mean the interpolation is not working as designed, which usually means the seasonal lag was applied in one direction only — check the sign on both shift calls.

10.12.5 Lab 10.2, Part C: The Opening, the Ramp, and the Documented Override

Part C is the chapter's verification theme at full scale: does the model see the expansion inflection? The test forecasts the known past across a store opening, so the answer can be graded. The opening is selected by rule rather than by eye, because a hand-picked break is not reproducible and may not have the four post-opening weeks the test needs.

Code 10.12. Forecast across a programmatically selected opening

```
# Choose the opening programmatically. It must sit far enough inside the
# series to have a real training window and four full post-opening weeks.
valid = opens[(opens["opening_date"]
               >= series.index.min() + pd.Timedelta(days=MIN_TRAIN))
              & (opens["opening_date"]
                 <= series.index.max() - pd.Timedelta(days=27))]
assert not valid.empty, "no opening has both a training and a test window"
break_date = valid["opening_date"].max()
store = valid.loc[valid["opening_date"].idxmax()]

others = opens.loc[opens["opening_date"] != break_date, "opening_date"]
near = others[(others - break_date).abs() <= pd.Timedelta(days=28)]
test_win = pd.date_range(break_date, periods=28, freq="D")
print(f"selected opening: {store.store_name} ({store.store_type}) "
      f"on {break_date.date()}")
print(f"other openings within 28 days: {len(near)}"
      f"   promotion days in the test window: "
      f"{int(test_win.isin(promo_days).sum())} of 28")

cut = break_date - pd.Timedelta(days=1)
tr_b, te_b = series.loc[:cut], series.loc[break_date:][:28]
for name in ["seasonal-naive", CHOSEN]:
    fc = pd.Series(fit_forecast(name, tr_b, 28), index=te_b.index)
    bias = (te_b - fc).mean()
    print(f"{name:<24s} MAPE {mape(te_b, fc):>6.2f}%"
          f"   signed error {bias:>+9,.0f}"
          f"   ({'undershoot' if bias > 0 else 'overshoot'})")
```

Expected output: the selected opening with its store type and date, the count of other openings within twenty-eight days, the number of promotion days inside the test window, and then two graded rows — seasonal-naive and the frozen method — each with MAPE, mean signed error, and a direction label; one assertion passes silently.

Input: the series and the filtered roster. Transformation: an eligibility filter on the roster, then a training cutoff at the day before the opening and a twenty-eight-day forecast across it. Output: the exhibit that teaches a planning meeting what a model can and cannot know.

The filter is the correction the draft needed. Selecting the last row of a sorted roster and adjusting by hand is not reproducible, and it can select an opening with too few post-opening days to evaluate. The rule states both requirements in code — enough history before, four full weeks after — asserts that at least one opening satisfies them, and then takes the most recent survivor. The two disclosure lines that follow are the honest half of the exercise: an opening whose window overlaps another opening or sits inside a heavy promotion is a contaminated test, and the reader is told the contamination before being shown the result rather than after.

VERIFICATION CHECK

Before you run: write the prediction in full sentences. Neither method can see the opening: seasonal-naive repeats a pre-opening week, and the frozen method extends a pre-opening level and trend. Both should therefore undershoot the post-opening series, and the signed-error column should print positive for both. Predict also whether the fitted method's undershoot will be larger or smaller than the baseline's, and why the answer is not obvious.

After you run: confirm the signs, and note what the exercise establishes — the model did not fail; the well-fitted model and the free rule failed identically in direction, because the missing information was never in the series. Then read the disclosure lines and write one sentence on how much the promotion days inside the test window complicate the reading.

Investigate if: either signed error is negative, or if the fitted method's undershoot is dramatically smaller than the baseline's. Both would suggest the model's trend term has already absorbed a run of earlier openings and is extrapolating them forward, which is a real phenomenon on this data, is the reason Code 10.14 prints a net-of-trend diagnostic, and is a finding to report rather than a problem to hide.

Code 10.13. Declare the ramp estimator and its contamination controls

```
# Two estimates of the same opening, because they answer two questions.
# RAW is the before/after step: what the series did. NET is the step the
# fitted method did NOT already extrapolate, which is the only part an
# override may add without double-counting the model's own trend.
PRE = POST = 28
MAX_PROMO = 7                    # of 28 days, counted on BOTH windows
NET_MIN_TRAIN = 180              # shortest fit the net estimate allows


def ramp_profile(d):
    """Week-by-week raw and net lift for one opening, or None."""
    before = series.loc[d - pd.Timedelta(days=PRE):
                        d - pd.Timedelta(days=1)]
    after = series.loc[d:][:POST]
    if len(before) < PRE or len(after) < POST:
        return None
    wk, a = np.arange(POST) // 7, after.to_numpy()
    out = {f"raw{w + 1}": float(a[wk == w].mean() - before.mean())
           for w in range(4)}
    train = series.loc[:d - pd.Timedelta(days=1)]
    if len(train) >= NET_MIN_TRAIN:
        resid = a - fit_forecast(CHOSEN, train, POST)
        out.update({f"net{w + 1}": float(resid[wk == w].mean())
                    for w in range(4)})
    else:
        out.update({f"net{w + 1}": np.nan for w in range(4)})
    return out


def promo_share(d):
    """Promotional days across the pre AND post windows, of 56."""
    win = pd.date_range(d - pd.Timedelta(days=PRE), periods=PRE + POST,
                        freq="D")
    return int(win.isin(promo_days).sum())
```

Expected output: nothing — this cell declares the controls and defines the ramp estimator.

Input: none. Transformation: four declared thresholds and one function. Output: the estimator Code 10.14 applies, with its assumptions written above it rather than discovered inside it. Read the function for two properties. The windows are equal in length, twenty-eight days on each side, which balances the weekday mix so that a step estimate is not partly an artifact of counting five Saturdays before and four after. And the function returns a week-by-week profile rather than a single number, because the prose of every planning deck describes new stores as ramping and an override that adds a flat step from day one contradicts the process it claims to model.

The estimator returns two profiles rather than one, and which of them the override may use is the cell's real argument. The raw profile is the before-and-after step: what the series did. The net profile is the post-opening actual minus what the fitted method itself forecast for those same days: the part the model demonstrably had not already extrapolated. A trend-tracking method fitted on a series full of recent openings has usually absorbed some of the expansion into its own trend term, so adding the raw step on top of the forecast adds that growth twice. The net profile is therefore the base adjustment and the raw profile is retained as an upper-bound sensitivity — a

distinction the code enforces and the override log records, rather than a caveat the prose supplies after the fact.

Two smaller controls sit in the same cell and both close real holes. Promotional days are counted across the pre and post windows together, because a promotion inside the *before* window inflates the baseline and shrinks the apparent step exactly as much as one inside the *after* window exaggerates it. And the net estimate requires a minimum training length, because a fit on ten weeks of history is not evidence about what a method would have projected; where that minimum is not met the net columns are blank rather than invented, and the printed table shows which openings carry which estimate.

Code 10.14. Apply the exclusion rules and report every surviving estimate

```
RAW = [f"raw{i}" for i in range(1, 5)]
NET = [f"net{i}" for i in range(1, 5)]
clean, dropped = [], []
for r in opens.loc[opens["opening_date"] < break_date].itertuples():
    d = r.opening_date
    other = opens.loc[opens["opening_date"] != d, "opening_date"]
    win = pd.date_range(d - pd.Timedelta(days=PRE),
                        periods=PRE + POST, freq="D")
    p, n_promo = ramp_profile(d), promo_share(d)
    if p is None:
        why = "window falls outside the series"
    elif ((other - d).abs() <= pd.Timedelta(days=PRE)).any():
        why = "windows overlap another opening"
    elif win.month.isin([11, 12]).any():
        why = "evidence window touches the holiday season"
    elif n_promo > MAX_PROMO * 2:
        why = f"{n_promo} of 56 window days promotional"
    else:
        why = None
    if why:
        dropped.append(f"{d.date()} ({why})")
    else:
        clean.append({"opening": d.date(), "type": r.store_type,
                      "promo/56": n_promo, **p})

ramps = pd.DataFrame(clean)
assert not ramps.empty, "no uncontaminated opening evidence survives"
print("individual estimates, reported rather than averaged away:\n")
print(ramps.round(0).to_string(index=False))
print("\nexcluded:", "; ".join(dropped), sep="\n  ")

same = ramps[ramps["type"] == store.store_type]
use = same if len(same) >= 3 else ramps     # stratify where n allows
net_ok = use.dropna(subset=NET)
if len(net_ok) < 3:                         # net evidence is thinner
    net_ok = ramps.dropna(subset=NET)
profile = net_ok[NET].mean()                  # the BASE adjustment
raw_mean = use[RAW].mean()                  # upper-bound sensitivity
lo_p, hi_p = net_ok[NET].min(), net_ok[NET].max()
STRAT = "same store type" if len(same) >= 3 else "all clean types"
print(f"\nraw profile from {len(use)} openings ({STRAT});"
      f" net profile from {len(net_ok)}")
for lbl, v in [("net (base) ", profile), ("raw (upper)", raw_mean)]:
    print(lbl + " " + "  ".join(
        f"wk{i + 1} {v.iloc[i]:>+7,.0f}" for i in range(4)))
print("net range   " + "  ".join(
    f"wk{i + 1} [{lo_p.iloc[i]:+,.0f}, {hi_p.iloc[i]:+,.0f}]"
    for i in range(4)))
print("""
The gap between the rows is the trend the method already carried;
adding raw on top double-counts it. Where the net range spans zero the
evidence has not established a lift, and a scenario is the honest
instrument.""")
```

Expected output: a table of every surviving opening with its store type, promotion-day count, four weekly lift figures and its net-of-trend week-four figure; then the list of excluded openings

each with its reason; then the evidence set actually used, the mean ramp profile, the range across openings at each week, and the net-of-trend mean.

Input: the roster, the promotion calendar, and the series. Transformation: four exclusion rules applied in a stated order, then stratification by store type where the count permits. Output: the override's evidence, in the form Section 10.9 requires — individual estimates visible, exclusions justified, and a range rather than a point.

Every line of the exclusion list is an argument. Openings whose windows fall outside the series cannot be measured. Openings whose windows overlap another opening cannot be separated from it. Any opening whose evidence *window* touches November or December is excluded, not only those whose own date falls there — a late-October opening measures December in its fourth week just as surely as a November opening does, and the earlier draft of this rule caught only the second kind. Windows with more than a quarter of their fifty-six days promotional measure the promotion. An assertion then refuses to proceed on an empty table, because averaging nothing is worse than stopping.

The stratification rule does what common sense demands: a flagship and a resort store have no reason to lift equally, so when at least three openings share the type of the opening being forecast, only those are used; otherwise the wider set is used and the widening is printed. The net profile has its own fallback for the same reason, since the minimum-training rule can leave fewer net estimates than raw ones. What survives is a small number of estimates with a wide range, which is the truthful characterization of what two years of history can say about a store opening — and it is why the override in Code 10.16 carries a range, a net-versus-raw distinction, and a review date rather than a decimal.

VERIFICATION CHECK

Before you run: predict how many openings will survive the four filters, and which specific roster rows will fall to which rule. Predict the shape of the mean profile — rising across the four weeks, or flat — and say which shape would falsify the ramp story the chapter has been telling.

After you run: read the individual estimates before either mean, which is the whole point of printing them. Compare the raw row and the net row week by week and write the two sentences the gap forces: some part of the apparent step was already in the model's trend, and the size of that part is not stable across openings — so the base override is the net profile, carrying a stated range, and the raw profile is a sensitivity rather than a rival. Then ask whether a plan built on the net mean would have been safe under the lowest surviving net estimate.

Investigate if: fewer than three openings survive, if the net set falls back to mixed store types, or if the net range at week four spans zero. The first two mean the override's evidence field must say which set it rests on. The third means the evidence does not establish that openings lift revenue beyond what the method already projects, and the honest instrument is a named scenario rather than a point adjustment — which is a legitimate outcome of this lab, not a failure of it.

Code 10.15. Apply the ramp override and write the log

```
wk_of = np.clip((te_b.index - break_date).days // 7, 0, 3)
fc_b = pd.Series(fit_forecast(CHOSEN, tr_b, 28), index=te_b.index)
day = (te_b.index - break_date).days
w1, w4 = day < 7, day >= 21
flat = float(np.nanmean(profile.to_numpy()))
for name, fc in [
        (CHOSEN, fc_b),
        ("+ flat step (net mean)", fc_b + flat),
        ("+ net ramp (base)", fc_b + profile.to_numpy()[wk_of]),
        ("+ raw ramp (upper)", fc_b + raw_mean.to_numpy()[wk_of])]:
    print(f"{name:<24s} MAPE {mape(te_b, fc):>6.2f}%"
          f"   wk1 {(te_b - fc)[w1].mean():>+8,.0f}"
          f"   wk4 {(te_b - fc)[w4].mean():>+8,.0f}")
print("""
The flat step and the net ramp share a mean by construction and differ
in shape. Read the raw row as a sensitivity, not as a rival: whatever
advantage it shows here is bought by re-adding growth the method had
already extrapolated, and it would not survive on a series where the
method had not.""")

print(f"""
OVERRIDE LOG -- {break_date.date()} {store.store_name} opening
 1 change     the 28 forecast days from {te_b.index.min().date()},
              ADJUSTED by a week-specific ramp, not a flat step
 2 size       {profile.iloc[0]:+,.0f}/day in week 1 to
              {profile.iloc[3]:+,.0f}/day in week 4, net of the trend
              the method already carried; an early week may be negative
 3 evidence   {len(use)} controlled openings ({STRAT}); the net
              profile rests on {len(net_ok)}; week-4 net range
              [{lo_p.iloc[3]:+,.0f}, {hi_p.iloc[3]:+,.0f}]; raw week-4
              step {raw_mean.iloc[3]:+,.0f}, carried as the upper-bound
              sensitivity and as the upside scenario, not as the base
 4 owner      <your name>, analyst of record
 5 review     {(break_date + pd.Timedelta(days=28)).date()}, when the
              four weeks of actuals mature
 6 baseline   the unadjusted {CHOSEN} row above, preserved""")
```

Expected output: three graded rows — the frozen method, the same method plus a flat-step override, and the same method plus the ramp override — each with MAPE and the signed error in week one and week four; a two-line note on why the means coincide; and the six-element override log.

Input: the pre-opening forecast, the ramp profile, and the test window. Transformation: a week-indexed adjustment. Output: the analyst's contribution, measured — and then written down in the form that makes it auditable.

The comparison is designed to isolate shape from size, which is why it reports week one and week four separately rather than one pooled bias. A flat step and a ramp built from the same evidence share a mean by construction, so a pooled signed error cannot distinguish them; the first and last weeks can. Four rows appear rather than three, and the fourth is the honest one: the raw ramp is shown beside the net ramp so the reader can see what the upper bound would have done. Read that row as a sensitivity rather than a rival. Whatever advantage it shows is bought by re-

adding growth the method had already extrapolated, and it would evaporate on a series where the method had not — which is exactly why the base override is the net profile and the raw profile becomes the upside scenario of Code 10.20 rather than the recommendation.

The log printed at the end is the deliverable, not the code. Read its six fields against Section 10.9 and note what each one costs to fill: the size field required a profile, the evidence field required the exclusions and the range and the net-of-trend figure, the review date required a decision about when this adjustment will be graded, and the baseline field required keeping the unadjusted row on the leaderboard where January can find it. An override log is expensive to fill honestly and worthless filled dishonestly, which is why the chapter prints one rather than describing one.

VERIFICATION CHECK

Before you run: predict whether the net ramp will beat the flat step on MAPE, and predict the direction of each override's week-one and week-four errors. Then predict where the raw ramp will land relative to both, and write down in advance why a better MAPE from the raw row would not make it the right choice.

After you run: confirm that the flat step and the net ramp share a pooled mean and that their weekly errors differ, then write the sentence that distinguishes them. State the net override's measured contribution in MAPE points and in dollars across the four weeks, because that figure is the analyst's own graded output. Then fill the log's owner field with your name and the review date in your calendar, and mean both.

Investigate if: the net ramp does not beat the flat step, or the net profile is negative in an early week. The designed expectation is that shape helps and that the raw row flatters itself; a different result is a finding to trace, not an error to edit away. Check whether the promotion days disclosed in Code 10.12 sit in the first or the last week, since a promotion in week one will make a flat step look better than it is — and remember that a negative early-week net adjustment is legitimate, which is why the log says the forecast was adjusted rather than raised.

10.12.6 Lab 10.2, Part D: Intervals That Aggregate in the Right Order

Part D produces the uncertainty the plan consumes. Three cells build it, and the middle one contains the single most important arithmetic correction in this chapter.

Code 10.16. Build and grade the daily prediction interval

```
REPS = 2000
selected_fit = fit_candidate(CHOSEN, select)


def empirical_paths(name, train, h, reps, seed=42):
    """Declared fallback when a zero-parameter rule wins: resample whole
    rolling-origin error paths, so the day-to-day dependence travels with
    the draw. Recycles beyond H and says so."""
    errs = np.array([
        series.loc[o + pd.Timedelta(days=1):][:H].to_numpy()
        - fit_forecast(name, series.loc[:o], H) for o in origins])
    g = np.random.default_rng(seed)
    draw = errs[g.integers(0, len(errs), reps)][:, np.arange(h) % H].T
    return pd.DataFrame(fit_forecast(name, train, h)[:, None] + draw,
                        index=holdout.index)


if selected_fit is None:
    INTERVAL_SOURCE = f"empirical rolling-origin residuals ({CHOSEN})"
    sims = empirical_paths(CHOSEN, select, len(holdout), REPS)
else:
    # statsmodels 0.14.6 spells the seed random_state=; 0.15 renames it.
    INTERVAL_SOURCE = f"{CHOSEN} simulation"
    sims = selected_fit.simulate(len(holdout), anchor="end",
                                 repetitions=REPS, error="add",
                                 random_state=42)
assert sims.shape == (len(holdout), REPS)
print("interval source:", INTERVAL_SOURCE, " paths:", sims.shape)

lo, hi = sims.quantile(0.10, axis=1), sims.quantile(0.90, axis=1)
inside = (holdout >= lo) & (holdout <= hi)
width = hi - lo
print(f"\nnominal coverage    80.0%")
print(f"realized coverage   {inside.mean():>5.1%}"
      f"   ({int(inside.sum())} of {len(holdout)} days)")
print(f"mean width          {width.mean():>9,.0f}"
      f"   ({width.mean() / holdout.mean():.0%} of mean daily revenue)")
print(f"evaluated periods   {len(holdout)} days from ONE forecast origin")

blk = np.arange(len(holdout)) // 7
full = blk < len(holdout) // 7               # the last days are a stub
diag = pd.DataFrame({"block": blk + 1, "inside": inside.to_numpy(),
                     "width": width.to_numpy()})[full]
print("\nby horizon week (7-day blocks, stub excluded):")
print(diag.groupby("block")
      .agg(days=("inside", "size"), coverage=("inside", "mean"),
           width=("width", "mean")).round(2).to_string())
print(f"stub: {int((~full).sum())} days beyond the eighth full week")
```

Expected output: the simulated array's shape and date span, then the nominal and realized coverage with the count of days behind it, the mean interval width in dollars and as a share of mean daily revenue, the number of evaluated periods and their single origin, and then coverage and width by seven-day block with the stub disclosed; one assertion passes silently.

Input: the fitted model and the sealed holdout. Transformation: two thousand simulated future paths, then daily percentiles across them. Output: the daily 80% band, and the five figures that constitute an honest grade of it.

The first line of the cell is the one that keeps the whole chapter honest. The band is built from the method the comparison actually selected, through the same factory Code 10.7 declared — not from a separately typed Holt-Winters call that would silently replace the winner. And because the two baselines fit nothing, the cell has to say in advance what it will do if a zero-parameter rule wins: it falls back to resampling that rule's own rolling-origin error paths, whole, so the day-to-day dependence travels with the draw, and it prints which source produced the band. A notebook that quietly substituted a model with a convenient simulate method would be reporting an interval that belongs to a method nobody chose.

The reporting standard is the second correction. A single coverage percentage cannot be read as a verdict, so the cell prints the whole set Section 10.8 specified — nominal, realized, mean width, period count, and coverage by horizon — and it labels the period count with the fact that all of them come from one forecast origin, which is the qualification that keeps a reader from treating sixty-one dependent daily observations as sixty-one independent tests. Read the by-week block with that in mind: seven days per block is far too few to estimate a coverage rate, so the column is a pattern to investigate rather than a set of measurements, and the width column beside it is the more informative of the two because it shows the cone opening.

VERIFICATION CHECK

Before you run: predict the realized coverage and, separately, predict what the width column will do across the eight blocks. Then predict something harder: given what Code 10.9 showed about this method on this season, predict whether the misses that fall outside the band will be balanced or one-sided.

After you run: read all five figures together before drawing any conclusion, then write the paragraph the deliverable needs. If realized coverage is near nominal, say what that does and does not license: it says the band prices day-to-day noise about right; it says nothing about the calendar, because a band centered on a forecast that systematically undershoots a season can still contain most individual days while being wrong about the total. If coverage is materially off in either direction, say which and record the width beside it, because a coverage figure quoted without a width can be bought by widening the band.

Investigate if: coverage is far above nominal and the mean width is a large share of mean daily revenue. That combination is the over-wide band the plan will be over-hedged by, and the response is to check the fitted error assumption against the remainder panel from Code 10.4 before shipping it.

Code 10.17. Aggregate paths into weekly intervals

```
# Weekly bins are declared, not inherited from a resample default: eight
# consecutive 7-day blocks from the forecast origin, plus a named stub.
labels = {b: str((holdout.index[b * 7]).date()) for b in np.unique(blk)}
tag = pd.Series([labels[b] for b in blk], index=holdout.index)

# WRONG: seven daily 10th percentiles added together is not the 10th
# percentile of the seven-day total.
wrong_lo, wrong_hi = lo.groupby(tag).sum(), hi.groupby(tag).sum()

# RIGHT: aggregate each simulated PATH first, then take quantiles across
# the weekly totals, which respects the dependence among the days.
wk_sims = sims.groupby(tag).sum()
right_lo = wk_sims.quantile(0.10, axis=1)
right_hi = wk_sims.quantile(0.90, axis=1)

cmp = pd.DataFrame({"actual": holdout.groupby(tag).sum(),
                    "forecast": wk_sims.mean(axis=1),
                    "lo80": right_lo, "hi80": right_hi,
                    "width_right": right_hi - right_lo,
                    "width_wrong": wrong_hi - wrong_lo})
print(cmp.round(0).to_string())
ratio = cmp["width_wrong"].mean() / cmp["width_right"].mean()
shuffle_rng = np.random.default_rng(42)      # local, not inherited
indep = sims.to_numpy().copy()               # destroy the dependence
for i in range(indep.shape[0]):
    shuffle_rng.shuffle(indep[i])
ind = pd.DataFrame(indep, index=sims.index).groupby(tag).sum()
ind_w = float((ind.quantile(0.90, axis=1)
               - ind.quantile(0.10, axis=1)).mean())
print(f"\nsummed edges are {ratio:.2f}x the correct width HERE. Shuffle "
      "the day-to-day\ndependence away and the same shortcut gives "
      f"{cmp['width_wrong'].mean() / ind_w:.2f}x. A quantile of a sum is "
      "not\nthe sum of the quantiles; the direction and the size of the "
      "error depend\non the marginals and their dependence, so neither "
      "can be reasoned about\nwithout the paths.")
```

Expected output: a nine-row table of declared seven-day blocks with the actual total, the forecast total, the correct 80% bounds, and the widths produced by the correct and the incorrect method side by side; then the ratio between them, and the same ratio recomputed after the day-to-day dependence has been shuffled away.

Input: the simulated paths and the holdout. Transformation: two aggregations of the same simulations in two different orders. Output: the correction of Section 10.8, made numerical.

The wrong method sums the daily bounds; the right method sums each path and then takes percentiles across the path totals. Note what the cell does not do: it does not merely assert that the second is correct. It computes both, prints the ratio, and then computes the ratio a third time on shuffled paths where the dependence has been destroyed. On the real simulations the summed-edges band is only a few percent too wide, because Holt-Winters paths drift together — a path that runs high runs high all week. On the shuffled version the same formula overstates the width by a factor near the square root of seven. Read the general claim carefully, because it is weaker and more useful than "summing makes the band too wide": a quantile of a sum is not the

sum of the quantiles, and the direction as well as the size of the discrepancy depends on the marginals and their dependence. Widening happens to be the direction here; another dependence structure need not produce it. That is why "the error is small in practice" is not a defense — the analyst who used the shortcut had no way to know how small, or which way.

The weekly bins are also declared rather than inherited. A resample to calendar weeks would cut the first and last blocks short and let a library default decide which days the plan's weeks contain; the cell instead counts seven-day blocks from the forecast origin, labels each by its start date, and discloses the stub of days left over at the end. When the managerial calendar requires weeks ending on a particular weekday, that is a separate declaration to be aligned with the reporting calendar on purpose — not a default to be discovered afterward.

VERIFICATION CHECK

Before you run: predict the direction of the error in the summed-edges band — too wide or too narrow — and, more importantly, write down whether you believe that direction is general or particular to this model. Then predict the size of the ratio on the real simulations, and predict how it will change when the dependence is shuffled away.

After you run: write the two sentences this cell exists to produce. First, the arithmetic one: a quantile of a sum is not the sum of the marginal quantiles, and the sum of seven daily tenth percentiles corresponds to all seven days landing at their own tenth percentiles at once, which is a much rarer event than the week's total landing at its tenth. Second, the epistemic one: the direction and the size of that discrepancy depend on the marginals and their dependence, so neither can be estimated without the paths — which means the shortcut is not a small approximation but an unquantified one.

Investigate if: the two widths are nearly identical. That would indicate near-total dependence across days within a path, which is worth confirming against the simulated paths themselves — and if it holds, it is a fact about the fitted model worth stating in the deliverable, because it means the weekly band and the daily band carry nearly the same information.

Code 10.18. Read the season total and plot the cone

```
season = sims.sum(axis=0)                 # one season total per path
s_lo, s_hi = season.quantile(0.10), season.quantile(0.90)
print(f"\nseason total across {REPS} paths:"
      f"   lo80 {s_lo:>11,.0f}   mean {season.mean():>11,.0f}"
      f"   hi80 {s_hi:>11,.0f}")
print(f"actual season total: {holdout.sum():,.0f}"
      f"   inside the band: "
      f"{bool(s_lo <= holdout.sum() <= s_hi)}")

fig, ax = plt.subplots(figsize=(9, 4))
ax.plot(select.index[-90:], select.iloc[-90:].to_numpy(), lw=0.8,
        color="0.4", label="history")
ax.plot(holdout.index, holdout.to_numpy(), lw=0.9, color="black",
        label="actual")
ax.plot(sims.index, sims.mean(axis=1).to_numpy(), lw=1.3,
        color="#57068C", label="forecast")
ax.fill_between(sims.index, lo.to_numpy(), hi.to_numpy(), alpha=0.20,
                color="#57068C", label="80% daily band")
ax.legend(fontsize=8, loc="upper left")
ax.set_title("the cone and what it omits", fontsize=9, loc="left")
fig.tight_layout()
```

Expected output: the season total's 80% bounds and mean across the simulated paths, the actual season total, whether it fell inside the band, and a forecast-cone figure showing history, actual, forecast, and the daily band.

Input: the simulated paths. Transformation: one sum per path, then percentiles across the totals. Output: the season-total range the plan consumes, and the cone that shows a meeting what a fanning interval looks like.

This cell is where the qualification of Section 10.8 stops being rhetorical. The daily band, as Code 10.16 showed, achieved close to its nominal coverage — and yet the season total lands high in its range, near the upper bound, because the forecast the band is centered on undershoots a holiday it was never told about. Both facts are true and they are not in tension: the band prices day-to-day innovation faithfully and prices the missing calendar not at all. Write that sentence into the deliverable verbatim, because it is the single most useful thing this chapter can tell a planning meeting about what an interval is. An interval is a statement about the future under the model. When the model is missing a component the business knows about, the interval is silent about exactly the risk the business is most exposed to — and the scenario table of Code 10.20 exists to carry what the band cannot.

10.12.7 Lab 10.2, Part E: The Planning Forecast, the Envelope, and the AI Round Trip

Everything so far has forecast the known past, so that it could be graded. Part E forecasts the future the commission actually asked about, which cannot be graded at all — and the discipline of the section is saying so clearly while still producing a number the plan can use.

Code 10.19. Refit on all history and declare the analyst's calendar

```
# Refit on ALL certified history, then forecast the horizon the decision
# spans. No actuals exist here; this is the deliverable, not a grade.
full_fit = fit_candidate(CHOSEN, series)      # the SELECTED method
assert full_fit is not None or CHOSEN in MODEL_CONFIG, (
    "a zero-parameter rule won: build the plan from empirical_paths "
    "and say so in the deliverable")
plan_idx = pd.date_range(PLAN_START, PLAN_END, freq="D")
stat_fc = pd.Series(fit_forecast(CHOSEN, series, len(plan_idx)),
                    index=plan_idx)

# THE ANALYST'S CALENDAR. None of this is in the series.
ANNOUNCED = pd.DataFrame({
    "store_name": ["Philadelphia", "Short Hills"],
    "store_type": ["Flagship", "Suburban"],
    "opening_date": pd.to_datetime(["2026-10-05", "2026-10-19"])})
HOL = (pd.Timestamp("2026-11-01"), pd.Timestamp("2026-12-31"))

IDX25 = holiday_index(series, 2025, CHOSEN)
both = pd.concat([IDX24.rename("2024"), IDX25.rename("2025")], axis=1)
print("holiday index by day from Thanksgiving -- two observations only:")
print(f"  peak 2024 {IDX24.max():.2f}   peak 2025 {IDX25.max():.2f}"
      f"   correlation {both.corr().iloc[0, 1]:.2f}")
```

Expected output: nothing but the holiday-index comparison — the peak index in each of the two available years and the correlation between them.

Input: the full certified series. Transformation: a refit on all history, a forecast across the planning horizon, and a second holiday index estimated from the 2025 season. Output: the statistical spine of the planning forecast, and the announced calendar the analyst brings to it.

The refit is on all certified history, because the sealed holdout has done its job and withholding data from the forecast that will actually be used would be superstition rather than discipline. The announced openings are declared as a small table in the notebook, which is exactly right: they are not in the series, they cannot be in the series, and the only place they can enter is a line of code an analyst wrote and signed. And the two holiday indices are printed side by side because that comparison is the whole evidence base for the annual layer. Two observations, one of them contaminated by an opening, correlated but not identical: that is what StyleCraft knows about its own holiday shape, and it is why the next cell produces three scenarios rather than one forecast.

Code 10.20. Build the holiday-season scenario envelope

```
# Coherent whole-year profiles, not a day-by-day minimum: the downside
# is the weaker observed holiday, not a path stitched from whichever
# year was lower at each offset. The ramp assumption moves with it.
off26 = pd.Series((plan_idx - thanksgiving(2026)).days, index=plan_idx)
in_hol = plan_idx.to_series().between(*HOL)
strength = both.mean(axis=0)
down_year, up_year = strength.idxmin(), strength.idxmax()
net_v, raw_v = profile.to_numpy(), raw_mean.to_numpy()
SCEN = {"downside": (both[down_year], net_v * 0.6, "net x0.6"),
        "base":     (both.mean(axis=1), net_v, "net"),
        "upside":   (both[up_year], raw_v, "raw upper")}
print(f"holiday profiles: downside {down_year}, upside {up_year}"
      f"  (mean index {strength[down_year]:.2f} vs"
      f" {strength[up_year]:.2f})")


def build(idx, ramp_vec):
    fc = stat_fc * off26.map(idx).ffill().bfill().where(in_hol, 1.0)
    for r in ANNOUNCED.itertuples():          # the ramp override, forward
        wk = (fc.index - r.opening_date).days // 7
        fc = fc + np.where(wk < 0, 0.0, ramp_vec[np.clip(wk, 0, 3)])
    return fc


plan = {k: build(i, v) for k, (i, v, _) in SCEN.items()}
season = {k: v.loc[HOL[0]:HOL[1]] for k, v in plan.items()}

# Preserve the fitted model's ADDITIVE error structure: centre the
# simulated paths and ADD them, rather than turning them into
# proportional shocks that grow with each scenario's level.
sim = full_fit.simulate(len(plan_idx), anchor="end", repetitions=400,
                        error="add", random_state=7).to_numpy()
innov = sim - sim.mean(axis=1, keepdims=True)
rows = []
for k, fc in plan.items():
    tot = (fc.to_numpy()[:, None] + innov)[in_hol.to_numpy()].sum(axis=0)
    rows.append({"scenario": k, "point": season[k].sum(),
                 "lo80": np.quantile(tot, .10),
                 "hi80": np.quantile(tot, .90),
                 "peak index": SCEN[k][0].max(), "ramp": SCEN[k][2]})
scen = pd.DataFrame(rows).set_index("scenario").round(2)
print("\nholiday-season revenue, Nov 1 - Dec 31 2026, assumptions shown:")
print(scen.to_string())

pt, base = scen["point"], scen.loc["base"]
half = (pt.max() - pt.min()) / (2 * pt.loc["base"])
sim = (base["hi80"] - base["lo80"]) / 2 / pt.loc["base"]
print(f"\nscenario envelope {pt.min():,.0f} to {pt.max():,.0f}"
      f"  (+/-{half:.0%} of base)")
print(f"simulation band   {base['lo80']:,.0f} to {base['hi80']:,.0f}"
      f"  (+/-{sim:.0%} of base)")
print("""
CAUTION. Rolling origins graded this method to 56 days; December sits
three times further out. Report both widths, say which dominates, and
label the envelope an assumption set rather than a graded forecast.""")
```

Expected output: a three-row scenario table with the Q4 point total, the 80% simulation bounds, and the two named assumptions behind each row; then the scenario envelope and the simulation band as separate ranges with their half-widths; then a five-line caution about the horizon.

Input: the refitted forecast, the two holiday indices, the announced openings, and the ramp profile from Code 10.14. Transformation: three scenarios, each combining a holiday index drawn from the available evidence with a multiplier on the opening ramp. Output: the planning range, and an explicit statement of which instrument carries it.

The scenarios are constructed from the evidence rather than from adjectives, and two details in that construction are easy to get wrong. The first is coherence. Taking a day-by-day minimum across the two available holiday indices would not produce "the weaker holiday"; it would produce a synthetic path stitched from whichever year happened to be lower at each offset — a season that never occurred and that no argument supports. The cell instead ranks the two observed profiles by their mean strength and uses whole years: the downside is the weaker holiday as it actually unfolded, the upside the stronger one, the base their average. The second is what moves with the holiday assumption. The ramp assumption travels with it, from a damped net override in the downside, through the net profile in the base, to the raw upper bound in the upside — so each row is one internally consistent world rather than a mix of assumptions from different ones. Both assumption columns are printed, which is the difference between a scenario set and three guesses wearing a rubric.

The uncertainty around each row also has to respect the model it came from. Holt-Winters was fitted with additive errors, so its simulated disturbances are dollar amounts, not percentages. Rescaling the simulated paths into multiplicative shocks — dividing by their own mean and multiplying by each scenario's level — would quietly convert them into proportional errors that grow with the scenario, which is a different model from the one that was graded. The cell centres the simulated paths and adds them, which preserves the fitted structure and keeps the three bands comparable.

The caution at the end is the cell's most important output and the most easily omitted. The rolling-origin evaluation graded this method to fifty-six days. December 2026 sits roughly one hundred eighty days from the forecast origin, three times the graded horizon, and the simulation band there has widened to a range no plan can act on. So the deliverable says which instrument is the planning range at which horizon: the band for the near weeks, the scenario envelope for the season — and the envelope is labeled as an assumption set rather than a graded forecast, with a monthly re-forecast cadence and a named trigger, the October actuals, at which the whole table is reopened.

VERIFICATION CHECK

Before you run: predict the ordering of the three scenarios (trivial) and then the harder thing — predict the ratio between the scenario envelope's half-width and the simulation band's half-width, and say which one you expect to be larger and why. Note that the half-width printed here is the envelope's span divided by twice the base case, which is a symmetric width around the base rather than a ratio of the extremes; a ratio of maximum to minimum is a different number and answers a different question.

After you run: write the paragraph the plan will actually consume. Name the base case, the envelope, the two assumptions that move it, and the horizon caveat, in that order. Then answer the question a planning lead will certainly ask: if the band is this wide, why is the point forecast worth anything at all? The honest answer is in Section 10.13 and it is worth rehearsing here — the point anchors the level and the weekly shape, both of which were graded; the range around it prices what was not.

Investigate if: the downside scenario falls below the prior year's actual season total. That would mean the downside is asserting a decline the evidence does not support, and its assumption column should be revised until it does. Do not treat a scenario envelope wider than the simulation band as a defect. Compare the two widths and disclose which source of uncertainty dominates: at a six-month horizon the calendar and structural assumptions may reasonably do more of the planning work than the fitted error distribution, and saying so is the deliverable's job rather than a problem to be tuned away.

Code 10.21. Translate the forecast into the open-to-buy envelope

```
# A DOLLAR envelope, every assumption declared and varied by scenario.
# Not a unit buy -- see the scope note below.
OTB = {"downside": (0.14, 0.52, 0.55),   # markdown, cost, Jan index
       "base":     (0.10, 0.50, 0.62),
       "upside":   (0.07, 0.48, 0.70)}
BOM_RETAIL = 1_150_000.0     # inventory on hand at retail, November 1
WOS_EOM = 4.0                # weeks of supply carried into January
MEDIA_LEAD_DAYS = 7          # declared: spend leads the demand it buys

rows = []
for k, fc in season.items():
    md, cost_pct, jan = OTB[k]
    sales = float(fc.sum())
    eom = fc.mean() * 7 * jan * WOS_EOM
    net = sales + eom + sales * md - BOM_RETAIL     # may be negative
    rows.append({"scenario": k, "holiday sales": sales,
                 "markdowns": sales * md, "EOM inventory": eom,
                 "net receipts": net,
                 "receipts (retail)": max(net, 0.0),
                 "excess inventory": max(-net, 0.0),
                 "open-to-buy (cost)": max(net, 0.0) * cost_pct})
otb = pd.DataFrame(rows).set_index("scenario").round(0)
print("open-to-buy envelope, Nov-Dec 2026 -- dollars, not units:")
print(otb.to_string())
b = otb["open-to-buy (cost)"]
wide = (f"({b.max() / b.min() - 1:.0%} wide)" if b.min() > 0 else
        "-- the downside needs no new receipts, which is a finding for"
        "\nthe buying team in August rather than an error")
print(f"\nenvelope {b.min():,.0f} to {b.max():,.0f}  {wide}")

wk = season["base"].groupby(
    np.arange(len(season["base"])) // 7).agg(["sum", "size"])
pace = pd.DataFrame({"revenue": wk["sum"].to_numpy(),
                     "days": wk["size"].to_numpy()})
pace["share"] = pace["revenue"] / pace["revenue"].sum()
pace["spend block"] = [
    str((season["base"].index[i * 7]
         - pd.Timedelta(days=MEDIA_LEAD_DAYS)).date())
    for i in range(len(pace))]
pace["paced spend"] = (pace["share"] * 480_000.0).round(0)
print(f"\nmedia pacing, base case, {MEDIA_LEAD_DAYS}-day lead:")
print(pace.set_index("spend block").round(3).to_string())
print("""
A demand-shaped pacing heuristic, not a causal or ROI-optimized
allocation; reconcile it with the campaign calendar before committing.

SCOPE. Revenue and open-to-buy dollars only. Assortment- and store-
level unit buys need a separate disaggregated forecast at those
grains, with its own baselines and its own holdout.""")
```

Expected output: a three-row open-to-buy table with Q4 sales, planned markdowns, ending inventory, receipts at retail and open-to-buy at cost; the envelope span; a nine-row media pacing table with revenue, days, share of the season and paced spend per declared block; and the scope statement.

Input: the three Q4 scenarios. Transformation: a declared open-to-buy identity and a declared media-pacing share. Output: the deliverable's planning translation, and the boundary the opening case promised.

Read the open-to-buy arithmetic as a chain of declared assumptions rather than a formula. Net receipts at retail are planned sales plus planned ending inventory plus planned markdowns minus the inventory already on hand, and each of the four terms is either forecast or declared: sales come from the scenario, markdowns from a declared rate that varies by scenario, ending inventory from a declared weeks-of-supply target applied to a declared January index, and beginning inventory from the current position. The cost complement converts receipts at retail into dollars of buying authority. Every one of those numbers is visible, every one varies by scenario, and none of them is a unit count — which is the point of the scope statement at the end.

One arithmetic detail is a business fact rather than a rounding decision. The identity can return a negative number, and a negative result is meaningful: it says the inventory already on hand more than covers the scenario's planned sales, markdowns and ending position. It is not, however, negative buying authority. The cell therefore separates the diagnostic from the operational amount — reporting net receipts as computed, the operational receipts floored at zero, and the implied excess inventory as its own column — and the spread line refuses to divide by a lower edge of zero. A zero lower edge is a finding the buying team needs in August, phrased as such, rather than a percentage that fails to compute.

The media-pacing table carries two labels it must not lose. It is a demand-shaped pacing heuristic, not a causal or ROI-optimized allocation: it spends in proportion to forecast revenue and establishes nothing about what that spend causes, which is Chapter 11's question rather than this one's. And because Section 10.9 said media should lead the demand it buys, the table applies a declared lead — the shares are computed on the revenue blocks and reported against spend blocks shifted earlier by that many days — so the lead is a visible parameter someone can argue with rather than an implicit assumption that spend and demand land in the same week.

That scope statement is the answer to the question the audit of this chapter asked hardest. A chain-level revenue forecast can size a dollar envelope and pace media by week. It cannot tell a buyer how many units of which style in which size go to which store, because that translation needs realized price, category and size mix, margin, markdown expectation, and store allocation — none of which live in a daily revenue series. So the chapter declares the envelope, hands the buying team a dollar figure with a range and a named set of assumptions, and names the disaggregated forecast at category and store grain as separate work with its own baselines and its own holdout. Promising quantities the analytical frame cannot produce is a specific and common way for an honest forecast to become a laundered guess, and Section 10.15 will say so in those words.

VERIFICATION CHECK

Before you run: predict whether the open-to-buy envelope will be proportionally wider or narrower than the revenue envelope that produced it, and explain the mechanism before you look. Then predict which declared assumption the envelope is most sensitive to, and check your prediction afterwards by changing one number at a time.

After you run: run that sensitivity check for real. Move the markdown rate, the weeks-of-supply target, and the cost complement one at a time and record which one moves the envelope most, because that is the assumption the deliverable must foreground and the one the planning meeting will argue about. Then read the media pacing table against the cone from Code 10.18 and write the pacing sentence: which weeks lead demand, and how much of the budget the plan can commit before the October re-forecast.

Investigate if: any scenario's net receipts come back negative, so that the operational floor and the excess-inventory column both engage. That is arithmetically possible and operationally meaningful — it says that scenario needs no new receipts at all, which is a finding the buying team must hear in August rather than in November. Check also that the declared media lead is compatible with the campaign calendar from Code 10.5 before anyone paces against it.

The lab closes with the AI round trip. Hand an assistant the raw marketing_daily file and the deliberately loose prompt "forecast our daily revenue for the next eight weeks — maximize accuracy," and grade the response in writing with Table 8.5's five points followed by Table 10.5's four. Find the split and run the date-order check on it; the designed likelihood, per Section 10.11, is shuffling or a single reused block. Determine whether method selection and the reported grade happened on the same data, which is the failure the draft of this chapter committed and the audit caught. Reconstruct what the pipeline did about the openings, the promotions, and the holiday it was never told about — the answer will be nothing. Check every reported metric for the near-zero denominator and every forecast for its missing interval, and check any weekly or total interval for whether it was built by summing daily bounds. Then compare its headline number against the rolling-origin band you predicted before it ran, and against the sealed-holiday number, which is the one the plan would actually have lived in. Re-prompt with the full frame per the AI in Practice box and compare the two responses. Your audit memo, filed per Appendix D, is the lab's final deliverable — and the direct rehearsal of the vendor-button verdict Section 10.13 writes.

10.13 Marketing Interpretation and Managerial Insight

The lab's outputs are leaderboards, cones, and a log; the commitments run on sentences. This section translates — and, per the guide's standing practice, it does so partly by exhibiting the wrong managerial readings and correcting them, because forecasting mints a special class of misreading: its numbers sound like promises, its percentages sound like guarantees, and both will be spoken in the planning meeting by people whose only error is treating a probability statement as a commitment.

The first wrong reading is the one the leaderboard invites, and it will be voiced the moment the leaderboard goes up: "The model's MAPE is 9 percent. So we'll be within 9 percent in December — plan accordingly." Every clause is wrong in a way a section of this chapter has priced, and this chapter's evaluation design was built so that the correction is a number rather than a caution. A MAPE is an average, not a bound: the evaluated days missed by more and less, and roughly half the misses exceeded the average — the interval, not the MAPE, states the range. It is an average at measured horizons: December sits at the far end of the cone, where Code 10.10's horizon table showed error running well above the pooled number, and the December weeks deserve the week-seven-and-eight grade. It was measured on the wrong season: the 9 percent came from rolling origins across ordinary spring and summer windows, and the same frozen method, graded on the sealed November–December holdout, missed by close to twice that — in one direction. And a percentage of December is the largest dollar figure of the year: nine percent of the season's peak weeks is a different commitment problem than nine percent of a quiet April, which is why the deliverable translates every accuracy figure into dollars at the season's scale before the meeting does it badly. The corrected sentence, worth writing on the leaderboard itself: the method's rolling-origin error is X percent, its error on the last holiday season it was graded against was Y percent with a systematic undershoot, and the planning range for this December is the scenario envelope, not either percentage.

The second wrong reading is quieter and does more organizational damage: the forecast read as a commitment. "You forecast $1.7 million and we got $1.5 — you were wrong, and planning built the receipts on your number." The correction has two halves, and the deliverable must make both in advance, not in January. The first half is Section 10.8's: the forecast was a distribution summarized by a point, the range travelled with it, and a forecast is wrong when actuals fall outside the communicated range more often than the stated rate — not whenever the point misses, because the point always misses. The second half is Section 10.9's word discipline: the moment the midpoint was copied into the plan as a commitment — and the range quietly deleted in the copying — the organization converted a probability statement into a promise nobody made. The deliverable's defense is structural: the range travels with the number in every artifact, the scenario table names what would move the outcome to each edge, and the planning translation states explicitly which side of the range each decision leaned toward and why.

The third wrong reading is the one this chapter's own draft invited, and correcting it is the reason Section 10.1 put a boundary in the brief. "The forecast says $1.7 million, so buy $1.7 million of product." A revenue forecast is not a buy. Between the two sit realized price, category and size mix, gross margin, expected markdowns, inventory already on hand, sell-through, and store allocation — every one of them an assumption, none of them present in a chain-level daily revenue series. What the deliverable can honestly produce is what Code 10.21 produces: an open-to-buy envelope in dollars, built from the forecast by a declared identity whose four terms are visible and vary by scenario, with the assumption the envelope is most sensitive to named. What it cannot produce is a unit plan, and saying so is not a hedge — it is the difference between handing the buying team a number they can interrogate and handing them a number that will be

interrogated in January when it is too late. The unit-level forecast is real work at a different grain, with its own baselines and its own holdout, and naming it in the deliverable is the analysis pricing its own upgrade.

The deliverable that survives all three misreadings has a fixed anatomy, assembled entirely from lab outputs. Page one is the frame and the calendar: the series definition with its reconciliation certificate from Code 10.3, the three declared windows with their dates and their different rights per Table 10.3, and the known-calendar list of Section 10.4 — openings, promotions, and the date of Thanksgiving in every year the analysis touches — with each forecast-window item matched to an override or a stated assumption. The rolling-origin leaderboard follows, per Section 8.9's fixed rules: naive, seasonal-naive, Holt-Winters, and damped Holt-Winters on the same origins, in MAPE with minimum denominator and MAE beside it, with the week-eight column and the bias column shown and the margin over the opponent stated in dollars per day. Then the sealed holdout, once: the frozen method and the frozen method plus the calendar layer, graded on a prior holiday season, with the signed error and the season-total miss — the exhibit that tells the meeting what the model cannot know. Then the horizon table with its error counts. Then the inflection exhibit from Codes 10.12 through 10.15 — both methods undershooting the opening, and the net ramp override closing the shape as well as the level, with the raw profile shown beside it as the upper bound it is — because it is the one exhibit that inoculates the room against the vendor button's confidence. Then the override log, six elements per entry, with the net-versus-raw distinction in the evidence field and the statistical baseline preserved. Then the interval paragraph: the daily coverage with its width and its period count, the weekly and season-total bands built path-first, and the sentence about what the band does not price. Then the scenario table, one internally consistent world per row, with both assumption columns, the scenario envelope and the simulation band reported side by side, and a sentence naming which of the two dominates at this horizon. And finally the planning translation: the open-to-buy envelope at each edge with its declared assumptions and its sensitivity, the asymmetry of Section 2.7 argued explicitly — markdown cost against stockout cost, and which side the recommendation therefore leans — the media pacing shape at the declared weekly blocks, the scope statement naming what the envelope is not, and the re-forecast calendar: who reruns this, on what cadence, and which meeting reopens it when October actuals land. A forecast, in this guide's definition, is all of that; the number alone is just the part that fits in a cell.

10.14 Business Analytics in Practice

This section turns from the fictional case to how forecasting operates in demand planning, disruption management, and media pacing — where its outputs move physical goods and real budgets, and where the hardest-won lessons are about bias, breaks, and the difference between an inaccurate forecast and an unexplained one. The first vignette below draws on the published record of retail forecasting practice; the second draws on the documented industry response to

the 2020 disruption; the third is a composite of recurring professional patterns rather than a report about a single named organization.

10.14.1 Demand Planning, S&OP, and the Bullwhip

The first vignette is demand planning and sales-and-operations planning, the institutional home of forecasting in most product companies, and the place where a forecast most directly becomes money. The cadence is recognizable from this chapter's own decision context, run at industrial scale: a statistical baseline forecast is generated by item and location; planners and commercial teams adjust it in a consensus process; and the adjusted forecast drives purchase orders, production schedules, and inventory positions months ahead of the demand it predicts. Retail adds two complications the published research treats as central and this chapter has been building toward. Demand is hierarchical — the same forecast is wanted at chain, category, store, and SKU grain, and the accuracy that is achievable falls sharply as the grain gets finer, which is exactly why this chapter's deliverable declares a chain-level dollar envelope and refuses the unit plan. And promotions dominate the error: at store-SKU grain, a large share of the variance a retailer most needs to anticipate is generated by its own promotional calendar rather than by any underlying seasonal process (Fildes et al., 2022b).

Two failure patterns dominate the genre's scar tissue. The first is bias: consensus processes tilt optimistic, because targets, incentives, and enthusiasm all push one direction — which is why mature demand-planning organizations track signed error as seriously as absolute error, and audit whether human touches to the statistical forecast add accuracy at all, the practice the judgmental-adjustment evidence of Section 10.9 directly supports (Fildes et al., 2009). The second failure compounds down the supply chain: the bullwhip effect, in which modest variability or bias in demand signals amplifies as each tier — retailer, distributor, manufacturer — orders against its own forecast of the tier below, so a small forecast error at the shelf becomes a large swing at the factory (Lee et al., 1997). The practice lesson lands directly on this chapter's machinery: a forecast is an input to other people's decisions, its errors propagate with leverage, and the disciplines that look pedantic at the desk — bias tracking, preserved statistical baselines, documented adjustments — are, at supply-chain scale, the difference between a bad quarter and a warehouse of markdowns.

10.14.2 The Season the Models Never Saw

The second vignette is the structural break at civilization scale. In the spring of 2020, pandemic lockdowns delivered a break no training window contained, and forecasting systems across retail experienced it as a controlled experiment nobody wanted: models trained on years of stable seasonality met a world where grocery demand spiked, apparel and travel collapsed, and e-commerce absorbed years of channel shift in weeks. The instructive part is not that the forecasts were wrong — everything in Section 10.4 says they had to be — but what the field concluded afterward, and the postscript to the retail forecasting literature is unusually direct about it. Established forecasting approaches, tuned on stable history, degraded badly; the period exposed

how thin the research base was on forecasting under abrupt structural change and on instability in general; and the practical responses that helped were the ones that shortened the reach of history and widened the statement of uncertainty (Fildes et al., 2022a).

Read those responses against this chapter's repairs, because they are the same list applied under fire. Teams shortened training windows and down-weighted the broken period, which is the which-history-still-applies judgment of Section 10.10. They re-forecast more frequently, shrinking the horizon because expected error grows with it. They widened intervals and communicated ranges where they had shipped points, because the honest statement of uncertainty had become impossible to avoid. And they shifted weight from fitted history to structured judgment — documented adjustments carrying information no model possessed, which is Section 10.9 operating as the primary engine rather than the garnish. The durable lesson survived the emergency: the pandemic period now sits in every retailer's training history as a permanent monument to the principle that a model cannot announce its own obsolescence. The humans who knew the world had changed had to say so, in writing, and the organizations that could do that quickly were the ones whose forecasting practice already separated the statistical baseline from the judgmental layer.

10.14.3 Media Pacing as Weekly Micro-Forecasting

The third vignette is small, fast, and the one most students will live first; it is a composite of a practice pattern rather than a named case. Every performance marketing team runs a version of it: a monthly or quarterly budget, spend and revenue accruing daily, and a standing question — where will we land? — answered continuously from partial-period actuals. The naive arithmetic everyone starts with ("we are 40 percent through the month at 35 percent of budget, so we will underspend") is a forecast, and usually a bad one, because it ignores exactly the structure this chapter taught: day-of-week rhythm, so a month's spend is not linear in days; intra-month and moving-holiday seasonality, since auction prices and conversion shift toward month-end and toward the retail events whose dates move; and self-inflicted breaks — a creative refresh, a bid-strategy change, a competitor entering the auction. Teams that pace well run, in miniature, the full discipline: a seasonal-naive expectation of the period's shape, a short-horizon projection updated daily, explicit adjustments for the promotions and launches on the calendar, and a range rather than a point when the finance partner asks for the landing. The practice lesson is proportion: the horizon is days and the stakes are one budget line, but the habits — baseline first, calendar declared, adjustment documented, range communicated — are the identical habits the holiday plan needed, which makes pacing the training ground where a junior analyst can practice this chapter weekly with fast feedback.

10.14.4 In Your First Analyst Job

In your first analyst job, these vignettes compress into one expectation, and it is this chapter's closing thread: businesses forgive inaccurate forecasts and punish unexplained ones. Every organization that plans has been missed by its forecasts — the veterans in the room have watched

seasons collapse and surprises rally, and they know the future is not fully knowable. What they do not forgive is the miss that cannot be explained: the number with no method behind it, the point with no range around it, the adjustment nobody wrote down, the December surprise that turns out to have been an announced opening the forecast silently ignored, or a holiday the model was never told about. The analyst who ships the full deliverable — method, opponent, grade on a season that resembles the decision, ranges, calendar, override log, review date — will still be wrong, routinely, by roughly the amount the holdout promised. But she will be wrong in a way the organization can learn from, and her forecasts will be trusted more each season, because the trust was never really about accuracy. It was about accountability, and accountability is a document, not a talent.

10.15 Ethics, Forecast Accountability, and the Laundered Guess

The Business Analytics in Practice section ended on accountability as a professional survival skill; this section examines it as an ethical obligation, extending the guide's running discussions — data use (Section 1.13), problem framing (Section 2.12), measurement design (Section 3.13), cleaning as editorial power (Section 4.14), honest summarization (Section 5.14), differential treatment of segments (Section 6.16), causal language (Section 7.15), acting on predictions about people (Section 8.15), and fairness across groups (Section 9.15) — to what forecasting adds: a number about the future that other people will stake resources, jobs, and judgment on, produced by a process only the analyst can see.

The chapter's ethics angle has a blunt name: the laundered guess. Laundering, in the financial sense, is passing something questionable through a process that makes it look clean, and a guess is laundered the same way: a number of uncertain provenance — a gut feeling, an unvalidated model, a target wearing analytical clothes — is passed through the apparatus of precision (decimals, a chart, a planning template) and emerges looking like knowledge. The vendor button of Section 10.1 launders structurally: a method nobody can inspect emits a number to the dollar, and the precision of the digits impersonates the reliability of the process. The consensus meeting launders socially: a forecast nudged upward by target pressure leaves the room wearing the statistical model's name. And the analyst launders personally in three specific ways this chapter has now named. She launders when she ships a point forecast without its range, because she possesses the holdout errors and has chosen to present a confidence the method does not have. She launders when she quotes a grade earned on a convenient window as though it applied to the decision's own season. And she launders when she reports a quantity her analytical frame cannot produce — a unit buy derived from a revenue forecast by assumptions nobody wrote down. In every case the mechanism is the same: uncertainty that someone knew about was stripped somewhere between the method and the meeting, and the audience downstream inherited a certainty nobody actually held.

Why is this an ethical matter rather than a technical one? Because of who bears the cost when the point forecast is treated as a promise. The planning lead who commits receipts against the laundered number owns the markdowns; the store teams whose staffing and bonus targets were

set from it own the shortfall; the finance partner who guided the quarter on it owns the surprise. The analyst's exposure — an awkward January meeting — is routinely the smallest in the chain, which is precisely the asymmetry that makes the disclosure obligatory: the person most able to state the uncertainty is the person least exposed to the cost of concealing it. The interval is therefore not a statistical garnish but the ethical instrument of the trade — the sentence "between X and Y, eight times in ten, under the model, and here is what the model does not know" is the honest transfer of what the analyst knows to the people who will bear what she does not — and the guide's position is correspondingly plain: a point forecast delivered to a resource decision without its range, by an analyst who could have computed one, is a misrepresentation, however accurate the point later proves. Being right is not a defense; the audience was entitled to the uncertainty, and got confidence instead.

Accountability, the angle's other face, is what makes the obligation livable rather than paralyzing, and its instruments are the chapter's own. The forecast is signed — the analyst of record, per Chapter 1, extends through the forecast's assumptions and overrides, each owned by name with a review date. The uncertainty is communicated at the decision's grain — the interval with its width and period count, the horizon table, the scenario assumptions, and the statement of which instrument is the planning range at which horizon — so that leaning optimistic or conservative becomes the decision-maker's explicit, priced choice per Section 2.7 rather than the analyst's hidden one. The overrides are documented so that judgment is auditable, per Section 10.9, and reviewed against actuals so the organization learns which judgments to keep. The scope is declared, so that a dollar envelope is not silently consumed as a unit plan. And the forecast is kept separate, in writing, from targets and plans, because the measures-as-targets dynamic of Section 2.7 has a forecasting form with a long industrial history: the moment the forecast is graded on whether it matches the aspiration, it stops informing the aspiration, and the company is navigating by a compass wired to its own hull.

CONCEPT

The Interval Is the Ethics

Every discipline this chapter built converges on one exhibit: the honest range, stated in advance, carried into every artifact, graded after the fact. The interval discharges the analyst's deepest obligation in forecasting — to transfer, undamaged, her knowledge of her own ignorance to the people spending against it — and its absence is never neutral: someone downstream will fill the missing uncertainty with confidence, because that is what planning processes do to naked numbers.

This chapter adds one clause the earlier draft of it lacked, and it is the clause an experienced forecaster will test you on. An interval is honest about what the model knows. It is silent about what the model was never told. On the sealed holiday holdout, the daily band achieved close to its nominal coverage while the season total landed near the top of its range, because the band priced noise faithfully and priced the missing calendar not at all. So the standing rule has five parts rather than four: no point without its range; no range without its method; no range without a statement of what it does not price; no override without its owner; no forecast without its review date. An analyst who holds that line will sometimes be the least popular voice in the planning meeting and will be, over seasons, the only one the meeting still believes.

Source: Course concept developed for this guide, informed by Chatfield (1993) and Hyndman and Athanasopoulos (2021).

10.16 Chapter Summary

This chapter carried Part II's predictive discipline into time, and the move cost more than a change of index. The data structure came first: a time series is ordered, its order carries the information, and the exchangeability that let Chapters 8 and 9 shuffle customers freely died at the door — StyleCraft's daily revenue had to be manufactured from the fact table by a deliberate grain change, reconciled against its source on a complete calendar with the zero-fill rule stated rather than inherited from a library default, and read before it could be modeled. Reading meant decomposition: the series resolved into trend, seasonality, cycle, and noise, with the store-opening dates drawn onto the trend panel and the ten largest remainders joined to the campaign calendar against a printed chance expectation, so that "promotions explain the spikes" arrived as a comparison rather than an impression.

The chapter's organizing correction is a boundary. Calendar effects are knowable in advance, but only stable fixed-period patterns are carried automatically by seasonal memory; moving holidays, promotions, and openings must be supplied explicitly. A Holt-Winters model at seasonal_periods=7 tracks a weekly rhythm and nothing else seasonal, so StyleCraft's annual holiday structure — two observations of a 365-position pattern, one of them containing a store opening — is not fitted at all. It is carried instead by an explicit calendar layer estimated from the prior holiday and indexed on days from Thanksgiving rather than on calendar dates, by scenarios that vary that layer across the two years of evidence that exist, and by written overrides with owners and review dates. The toolkit was then built as a ladder with the opponent on the bottom

rung: the naive and seasonal-naive rules extended Chapter 8's baselines into time and proved, per the M-competitions' standing humility, brutally hard to beat; moving averages smoothed the rhythm away for reading; and the exponential smoothing family — level, then Holt's trend, then a damped variant, then Holt-Winters' weekly seasonal effects — added tracked components one at a time, each purchasing power with flexibility.

Honest evaluation was rebuilt for ordered data in two layers rather than one. Random splits were convicted of leakage by design — the shuffled evaluation grades interpolation, a job no deployed forecast performs — and replaced by rolling origins inside the training data, which chose the method across seven origins and fifty-six-day horizons, and by one sealed holdout opened once for the frozen method. The holdout was chosen for a property most tutorials ignore: it had to resemble the decision, so it is a prior holiday season forecast from before it, and the result is the chapter's most useful number. The method that won on ordinary windows missed the holiday by roughly twice its selection score, systematically low, which is not a defect but the measured size of the calendar the model was never given. MAPE joined the dollar metrics as planning's shared currency and acquired a guard in code — denominators asserted, minimum printed beside every figure — after the miniature showed one storm-closed day converting a five-percent forecast into a headline of two hundred sixty. Error was reported by horizon across several origins rather than one path, with the number of errors behind each row printed, and the claim was stated at the strength the evidence supports: expected error and uncertainty grow with horizon, while realized error in any one sample need not rise at every step.

The forecast was then widened from a number into a distribution, and the widening carried two corrections. Weekly and season-total intervals are built by aggregating each simulated path first and taking quantiles second, because the sum of seven daily tenth percentiles is not the tenth percentile of the seven-day total — and the lab measured how wrong the shortcut is on real paths and on paths whose dependence had been shuffled away, to show that the size of the error cannot be known without the paths. And the band was qualified rather than trusted: it is conditional on the fitted parameters, it prices future innovation and horizon compounding, and it prices the missing calendar not at all — which the sealed holdout demonstrated by achieving near-nominal daily coverage while the season total landed near its upper bound. Coverage was reported with width, period count, and horizon detail, so that no one can buy a coverage figure by inflating a band.

The human hand was disciplined rather than banished. Judgmental adjustment earned its legitimacy exactly where it carries information the history cannot — the announced opening, the moving holiday, the drafted promotion — and paid for it with implementation as well as paperwork: a ramp profile rather than a flat step, because the prose said the stores ramp; equal weekday windows, excluded overlapping openings, excluded holiday-season openings, a promotion-day cap, stratification by store type, every surviving estimate printed, and a range carried forward instead of a decimal; plus a net-of-trend diagnostic that asks how much of the apparent step the model was already extrapolating, whose instability is itself the finding. Then

the six-element written override, whose deepest payoff arrives in January when the preserved baseline lets the organization compute what each judgment contributed.

The verification theme ran through all of it: the labs forecast the known past across a programmatically selected opening, watched the well-fitted model and the free rule fail identically in direction, and closed both the level and the shape with a documented ramp. And the deliverable finally reached the decision without overclaiming: a Q4 scenario envelope with its assumption columns, a horizon caution stating that the graded horizon was fifty-six days and December sits three times further out, an open-to-buy envelope built from a declared identity whose every term varies by scenario, a media pacing shape at declared seven-day blocks, and a scope statement naming what a chain-level revenue forecast cannot produce — the unit buy, which is separate work at a separate grain. Twenty numbered cells carried the argument into executable form, and the failure modes, the AI supplement, the interpretation, the practice mirror, and the ethics section circled one insistence from different distances: a forecast is a probability statement with a method, an opponent, a grade on a season that resembles the decision, a range with a stated scope of what it prices, and a signature. Everything less is a guess, laundered.

One forward note redeems a pointer the smoothing section deferred: the moving averages built here return in Chapter 14 as visual analytics, computed live on the Tableau canvas, where their window arithmetic — learned here — is the difference between reading them and merely displaying them.

Looking ahead, the chapter's machinery answers what will happen: December's range, the season's shape, the opening's step. It is silent, by construction, on the question the planning meeting will ask next — what caused it, and what would happen if StyleCraft acted differently? The forecast can say the suburban stores' revenue will step up; it cannot say the expansion caused the step, or what December would have looked like without it, because everything in this chapter learned from a single unrepeated history in which StyleCraft did only one thing. The next chapter takes up the machinery that can answer such questions: experiments, A/B testing, and causal evidence — random assignment as the instrument that manufactures the comparison history never provides, the known true lift the dataset ships so the analysis can grade itself, and the evidence hierarchy that finally places everything Part II has built — descriptions, models, forecasts — on the ladder of what each one licenses a marketer to claim.

10.17 Exercises for Practice and Homework

The following exercises practice the chapter's main habits: build and reconcile the series before modeling it, read the decomposition before choosing a method, name the opponent before fitting, separate method selection from the grade, choose a holdout that resembles the decision, report MAPE with its denominators and error by horizon across origins, aggregate intervals in the right order, document every override with its six elements, and declare the limits of the planning unit. They are organized into three groups. Core chapter practice is the required path and should be completed by every student, and it holds the two homework submissions from which your

instructor will assign a subset; Assignment #4, Forecasting, draws on the homework sets of this chapter. In-class activities are prepared for discussion rather than submitted. Extensions are optional. Each exercise also carries its assignment label so that instructors can assign selectively.

10.17.1 Core Chapter Practice

Exercise 10.1 Concept Check (Required Practice)

Answer each in two or three sentences, in your own words.

- State what makes a time series a different data structure from the customer feature table, and name the discipline from Chapter 8 that survives the change and the one that does not.
- Distinguish seasonality from a cycle and both from a structural break, with one StyleCraft instance of each.
- Explain why history from before a structural break misleads a model after it, and why this is not a defect the model can be tuned out of.
- A colleague says that because Christmas and Black Friday are known in advance, a Holt-Winters model will carry them. Correct the claim precisely, distinguishing knowable from carried, and name the three instruments that carry the effects the model does not.
- Write the naive and seasonal-naive forecasting rules for a daily series with weekly rhythm, and explain why this guide states the December-repeats-December version only for monthly data.
- State what Holt's method adds to simple exponential smoothing, what the damped variant adds to Holt, and what Holt-Winters adds to both — and name the new failure each addition makes possible.
- Explain why a random train/test split flatters a forecasting model, using the word interpolation, and state the one-line check that catches the flattery.
- Explain why comparing four candidate methods on the final holdout spends the grade, and describe the two-layer design that fixes it.
- This chapter's sealed holdout is a prior holiday season rather than the last eight weeks of the file. Give the reason, and state what a late-spring holdout would and would not have established about a holiday plan.
- A holdout MAPE of 6 percent was computed on a series whose minimum value is $180 against typical values near $11,000. May the MAPE lead the deliverable? Explain, citing the standing rule, and describe what the chapter's mape helper would have done.
- Explain why the sum of seven daily 80% intervals is not the weekly 80% interval, and state the correct procedure in one sentence.
- A stakeholder reads an 80% prediction interval as "the analysts are only 80% sure of their work." Correct the reading in two sentences, including what it would mean for the interval to be wrong and what the interval does not price at all.

Exercise 10.2 Baselines and MAPE by Hand (Required Practice)

Using only the fourteen-day miniature of Code 10.1 (no code): rebuild both forecast rules day by day and verify every figure in the Verification Check — the naive rule's errors and its MAE of 1,300.00 and MAPE of 12.79 percent; the seasonal-naive rule's errors and its MAE of 514.29 and MAPE of 5.25 percent. Identify the two week-two days on which the naive rule's error is largest and state, in calendar language, what each miss has in common. Show that all seven seasonal-naive errors share a sign, compute their mean, and write one sentence on which row of Table 10.1 the bias reveals. Then rebuild the storm scenario: verify the MAE of 1,471.43 and the MAPE of 261.52 percent, show which single term produced the explosion, and state what value the storm Tuesday would have had to take for the chapter's guarded mape helper to refuse to compute at all. Close with the two-sentence honest summary of the storm week that the standing denominator rule requires.

Exercise 10.3 Read the Calendar (Required Practice)

For each series below, list the calendar effects and structural breaks a forecaster should declare before fitting, classify each as living in the training window, the forecast window, or both, and state — per Section 10.4's obligation — how each forecast-window item should be discharged: by a fitted seasonal period, by an explicit calendar layer, by a documented override, by a stated scenario assumption, or by scope exclusion. Where you answer "fitted seasonal period," you must also state how many repetitions of the pattern the history contains.

- StyleCraft's daily chain revenue, forecast from July 2026 through the end of December 2026, given the stores table, the drafted holiday promotion calendar, and a board deck listing two further openings under consideration for October.
- A single Wave-4 resort store's daily revenue, opened five months ago, forecast eight weeks ahead.
- The email channel's daily attributed revenue, where sends occur three days a week and revenue on non-send days is frequently near zero.
- Weekly chain revenue for the four weeks around Thanksgiving, forecast one year ahead, where the analyst proposes to use last year's same-week values.

Exercise 10.4 Spot the Failure (Required Practice)

Each scenario below contains at least one failure from this chapter, or none. Name it, cite the section or table row, and state the repair.

- An analyst evaluates a daily revenue model with five-fold cross-validation, reports a MAPE of 2.1 percent, and notes the model "performs far better than the seasonal-naive rule's 7 percent."
- A pipeline's feature list includes a centered 14-day moving average of revenue.
- An analyst compares naive, seasonal-naive, and Holt-Winters on the final eight weeks of the series, keeps Holt-Winters because it wins, and reports its error on those eight weeks as the model's out-of-sample grade.

- A forecast for October–December is fitted on the full two-year history, and the analyst notes with satisfaction that the model has "learned the holiday season" from granular day-of-year seasonal effects.
- A team forecasts across an announced flagship opening using Holt-Winters, and when the forecast undershoots, concludes the smoothing parameters need retuning.
- A weekly forecast table reports 80% bounds obtained by summing the daily 80% bounds within each week. The analyst notes that the difference "is only a few percent anyway."
- An analyst reports the eight-week forecast as a weekly table of point values with a footnote that "actual results may vary," and separately keeps a spreadsheet of the intervals "to avoid confusing the audience."
- A planning lead fits the forecast, finds it 9 percent below the CEO's announced growth target, and raises the seasonal effects until the gap closes, noting the December numbers "looked low anyway."
- A deck reports a chain revenue forecast of $1.7 million for Q4 and, on the next slide, a recommended purchase quantity of 34,000 units. No other assumption appears anywhere in the deck.

Exercise 10.5 The Full Forecasting Deliverable (Homework Submission)

Complete Labs 10.1 and 10.2 on the certified files and assemble the planning deliverable per Section 10.13: the frame-and-calendar page with the reconciliation certificate and the three declared windows with their different rights; the rolling-origin leaderboard for all four candidates with MAPE, minimum denominators, MAE, the week-eight column, the bias column, and the margin over the opponent in dollars per day; the sealed-holdout grade, opened once, for the frozen method and the frozen method plus the calendar layer, with signed error and season-total miss; the horizon table with its error counts and the December-distance sentence; the inflection exhibit with both methods' bias across the programmatically selected opening and the ramp override's measured contribution to level and to shape; the override log with all six elements per entry, including the range and the net-of-trend figure in the evidence field; the interval paragraph with coverage reported beside width and period count, the weekly and season-total bands built path-first, and the sentence stating what the band does not price; the scenario table with its assumption columns and the horizon caution; and the planning translation with the open-to-buy envelope, its sensitivity to the assumption you identified in Code 10.21's Verification Check, the asymmetry of Section 2.7 argued explicitly, the media pacing shape, the scope statement, and the re-forecast calendar. Submissions are graded on the calendar page, the separation of selection from grade, the override log, the interval communication, and the scope statement as heavily as on the models.

Exercise 10.6 AI Forecast Audit (Homework Submission)

Give an AI assistant the raw marketing_daily file and this deliberately loose prompt: "Forecast our daily revenue for the next eight weeks — maximize accuracy." Before reading the response, write your predicted leaderboard band per the AI in Practice box: the seasonal-naive MAPE you

expect (compute it yourself) and the band an honest model should land in. Then audit the response in writing with Table 8.5's five points followed by Table 10.5's four: locate the split and run the date-order check; determine whether method selection and the reported grade happened on the same data; determine whether any feature or smoother consumed post-cutoff data; reconstruct what the pipeline assumed about openings, promotions, and the holiday; check every metric's denominators and every forecast for its missing interval, including whether any weekly or total interval was built by summing daily bounds; and compare the headline claim against your predicted band, classifying any excess as pipeline, not prophecy, until traced. Re-prompt with the full frame and compare leaderboards. Document both exchanges per the AI-use documentation template in Appendix D, and conclude with two sentences on which supplement point caught the most serious problem.

10.17.2 In-Class Activities

Exercise 10.7 The Override Meeting (In-Class Discussion)

Your team will rehearse the planning meeting's hardest conversation: which judgments enter the forecast. Prepare: the statistical baseline forecast and its two grades, the rolling-origin score and the sealed-holiday score, with a one-sentence explanation of why they differ; a proposed override for the fall's announced openings, sized per Codes 10.13 through 10.15, with its six elements complete and its range and net-of-trend figure in the evidence field; a proposed calendar layer for the holiday, with the contamination in its single prior observation disclosed; the merchant's thirty-percent gifting thesis, translated into the strongest legitimate form it can take under Section 10.9 — a named scenario with an argued size, prepared from occasionwear mix and the suburban stores' basket profile, per the certified findings of Chapter 5; and a written position on which of the three circulating forecasts from Section 10.1 contains information the final forecast should keep, and in what form. Half the class prepares the merchant's side seriously: her number cannot be graded, but her knowledge is real, and the strongest version of her case is what the override discipline must be able to absorb without breaking.

Exercise 10.8 Find the Flaw in the Forecast (In-Class Discussion)

Each scenario below contains at least one flaw from this chapter. Name it, cite the section, and state the repair.

- A vendor's forecasting module reports 97 percent accuracy. Its documentation does not state the evaluation design, and its demo notebook calls a generic train/test split on the daily rows.
- A forecast deck leads with a single MAPE for the eight-week horizon. The commitment concerns weeks seven and eight.
- A team validates its holiday forecasting method on the eight weeks ending in June, then uses it to set November and December receipts.

- A planning team's December forecast has come in optimistic four years running. Each year's miss was explained by a different one-off; no one has computed the four-year mean signed error.
- An analyst, told the openings would bias her forecast low, raises every week of the forecast by 6 percent, mentions it in the meeting, and moves on. The next analyst inherits the model and cannot reproduce the published numbers.
- A forecast's 80% intervals captured the actuals in 94 percent of holdout periods, and the team celebrates the intervals as "extra safe." No interval width is reported anywhere in the deck.
- An 80% interval on the season total is reported for a forecast issued six months ahead, with no statement of the horizon at which the method was graded.
- A dashboard recomputes the season forecast nightly, and the number the CFO quotes moves daily by amounts that dwarf the model's day-over-day information gain. Planning has begun averaging the last five days' numbers by hand.

10.17.3 Extensions

Exercise 10.9 Build the Horizon Curve Two Ways (Optional)

The horizon table of Code 10.10 averages errors by week-ahead across seven forecast origins. Rebuild it two ways and compare. First, from a single origin — take the last origin only and read its eight weekly blocks as a horizon curve, which is what the draft of this chapter did. Second, from the seven origins as the chapter now does. Report both tables side by side, with the number of errors behind each figure, and then answer three questions in writing. Where do the two curves disagree most, and what calendar event from Codes 10.4 and 10.5 sits underneath that week in the single-origin version? What does the standard-deviation column in the multi-origin table say about how much of the single-origin curve's shape was signal? And how many origins would you want before you were willing to publish a horizon curve as a claim about the method rather than a description of one two-month stretch — state a number and defend it. As an extension of the extension, vary the origin spacing from 28 days to 14 and report what changes and what does not, noting that overlapping forecast windows share observations and therefore share error.

10.18 Glossary of Terms

This glossary includes only the terms introduced in this chapter. Each definition is tied to the sources used in the chapter rather than added for decoration.

Calendar effect. A systematic influence the calendar exerts on a business series — day-of-week rhythm, fixed-date holidays, moving retail events, paydays, self-imposed merchandising rhythms. All are knowable in advance; only stable fixed-period patterns are carried automatically by seasonal memory, and the rest must be supplied explicitly (adapted from Hyndman & Athanasopoulos, 2021).

Cycle. A recurring rise and fall in a time series without a fixed, calendar-locked period — economic or fashion waves whose timing cannot be read from a calendar, distinguishing them from seasonality (adapted from Hyndman & Athanasopoulos, 2021).
Decomposition. The separation of an observed time series into estimated trend, seasonal, and remainder components under an assumed structure, most commonly additive; used in this guide as a reading instrument that shows which stories a series contains and how large each is, at one declared period (adapted from Cleveland et al., 1990; Hyndman & Athanasopoulos, 2021).
Exponential smoothing. A family of forecasting methods that maintain running, recency-weighted estimates of a series' components — level alone in simple smoothing, level and trend in Holt's method, level, trend, and season in Holt-Winters — updating each by a fitted smoothing parameter as observations arrive (adapted from Holt, 2004; Gardner, 2006).
Forecast horizon. The span of future periods a forecast claims, in the series' own grain. Because uncertainty compounds with each step ahead, forecast uncertainty and expected error generally increase with horizon, although realized error need not rise monotonically in any one evaluation period; honest evaluation reports accuracy by horizon, estimated across several forecast origins, with the number of errors behind each figure stated (adapted from Hyndman & Athanasopoulos, 2021; Tashman, 2000).
Holiday calendar layer. This guide's name for an explicit, analyst-supplied adjustment that carries annual and moving-holiday structure a fitted seasonal period cannot: a ratio of actual to statistical forecast measured across a prior holiday season and indexed by days from the moving anchor date rather than by calendar date. Estimated from one or two observations, it is an assumption with a stated range and a review date, not a fitted component (course concept developed for this guide, informed by Hyndman & Athanasopoulos, 2021).
Holt-Winters method. The seasonal member of the exponential smoothing family: smoothed level, smoothed trend, and smoothed seasonal effects at a declared period, forecast forward as a trended level with the seasonal pattern reapplied. A model fitted at period 7 carries a weekly pattern and no other (adapted from Winters, 1960; Gardner, 2006).
Judgmental adjustment. A deliberate human change to a statistical forecast intended to incorporate information the fitted method could not access; legitimate when carrying genuine event knowledge and documented as a written override, and reliably accuracy-destroying when small, frequent, and optimistic — with upward adjustments substantially less reliable than downward ones (adapted from Fildes et al., 2009).
MAPE (mean absolute percentage error). The average of absolute forecast errors expressed as percentages of the actual values; scale-free and planning's shared currency, but undefined at zero actuals and explosive near them, so it is computed only after its denominators are asserted, reported with its minimum denominator, and disqualified for series that approach zero (adapted from Hyndman & Koehler, 2006).
Moving average. The replacement of each observation with the mean of a fixed window of surrounding observations, trailing or centered; a window equal to the seasonal period removes the seasonal pattern, making the moving average both a trend-extraction instrument and a

deliberate act of information removal. Inside a forecasting pipeline it must be trailing only (adapted from Hyndman & Athanasopoulos, 2021).

Naive forecast. The rule that every future value equals the most recent observed value — the last-value baseline of Section 8.7 at the series' own grain, extended flat across the horizon (adapted from Hyndman & Athanasopoulos, 2021).

Open-to-buy envelope. This guide's name for the planning unit a chain-level revenue forecast can honestly produce: planned receipts at retail, computed as planned sales plus planned ending inventory plus planned markdowns minus inventory on hand, converted to dollars of buying authority by a declared cost complement. Every term is either forecast or declared and varies by scenario, and the envelope is a dollar figure rather than a unit plan (course concept developed for this guide).

Prediction interval. A range around a point forecast derived from the forecast distribution the fitted model implies: under the model and its assumptions, an 80% interval is constructed so that comparable intervals would contain their future observations about 80% of the time. Its width grows with horizon, it is gradable by realized coverage reported beside width and period count, and it is conditional on the fitted parameters — silent about structure the model was never given (adapted from Chatfield, 1993; Hyndman & Athanasopoulos, 2021).

Rolling-origin evaluation. Estimation of out-of-sample forecast accuracy by repeating a fit-and-forecast exercise from several successively later forecast origins, so that every training set precedes its own test observations and accuracy can be reported separately by horizon; the time series counterpart of cross-validation, performed inside the training data so a final holdout can stay sealed (adapted from Tashman, 2000; Hyndman & Athanasopoulos, 2021).

Sealed holdout. A contiguous block of the known past, withheld from every selection decision and consulted once for the frozen method. In forecasting it carries an additional obligation the customer-grain version did not: it should resemble the decision, in season and in horizon, because a method graded on an unlike window has not been tested on the risk the decision runs (course concept developed for this guide, informed by Tashman, 2000).

Seasonal-naive forecast. The rule that every future value equals the most recent observed value from the same position in the seasonal cycle at a declared period — this Saturday forecast by last Saturday on daily data at period 7, this December by last December on monthly data at period 12 — reproducing one cycle of the pattern forward at zero fitted parameters; this guide's declared opponent for forecasting models (adapted from Hyndman & Athanasopoulos, 2021).

Seasonality. A repeating pattern in a time series with a fixed and known period, recurring at a constant number of observations apart — days of the week, months of the year — and therefore carryable by seasonal memory. Moving holidays are calendar effects but not seasonality in this sense (adapted from Hyndman & Athanasopoulos, 2021).

Structural break. A point in time at which the process generating a series changes — a persistent shift in level, trend, or seasonal pattern — after which pre-break history describes a regime that no longer exists; in marketing, frequently self-inflicted and announced in advance (adapted from Hyndman & Athanasopoulos, 2021).

Time series. A sequence of observations of the same quantity at successive, usually regular, points in time, in which the temporal ordering is part of the data's meaning and must be preserved because observations may exhibit serial dependence, trend, seasonality, or structural change (adapted from Hyndman & Athanasopoulos, 2021).
Trend. The long-run direction of a time series — the smooth underlying path once short-run fluctuation is set aside; in StyleCraft's series, organic growth compounded by the expansion's stair-step contributions (adapted from Hyndman & Athanasopoulos, 2021).

10.19 Further Readings

Students who want additional background may begin with the following readings. The applied treatments are listed first, evidence and methods second, software last.

- Hyndman and Athanasopoulos (2021) for the standard modern textbook of applied forecasting — free online, code-first, and the source this chapter's definitions lean on most. The chapters on decomposition, exponential smoothing, and evaluation are the direct expansions of Sections 10.3, 10.6, and 10.7; the section on time series cross-validation is the formal treatment of the rolling origins this chapter's Code 10.8 implements; the section on weekly, daily, and sub-daily data is where the multiple-seasonality and moving-holiday problem of Section 10.4 is developed properly, including the harmonic-regression machinery this chapter deliberately leaves out; and the section on distributional accuracy is where coverage-plus-sharpness evaluation goes beyond what Section 10.8 does.
- Fildes et al. (2022b) for the field's own survey of retail forecasting research and practice — hierarchical demand, the dominance of promotions at fine grain, and the gap between what the literature studies and what retailers do; the direct expansion of Section 10.14.1 and the published support for this chapter's refusal to promise a unit plan from a chain-level series. Read it with its postscript, Fildes et al. (2022a), which is the short and pointed account of what the 2020 disruption did to established methods and why structural instability remains the field's weakest area.
- Makridakis et al. (2020) for the M4 competition's results — one hundred thousand series, sixty-one methods, and the empirical case for baseline humility that Section 10.5 built its opponent on.
- Tashman (2000) for the methodological treatment of out-of-sample testing: fixed versus rolling origins, why multiple origins are needed before a horizon curve means anything, and the design vocabulary behind Table 10.3.
- Hyndman and Koehler (2006) for the definitive survey of forecast accuracy measures, including the full indictment of MAPE's denominator behavior and the scale-free alternatives this guide leaves for later coursework.
- Fildes et al. (2009) for the large-scale field evidence on judgmental adjustments in real forecasting operations — which adjustments help, which reliably hurt, why upward

revisions are the least trustworthy, and why the written-override discipline of Section 10.9 is what separates them.

- Gardner (2006) for the authoritative survey of exponential smoothing's state of the art, including the damped-trend evidence behind this chapter's fourth candidate — the extended companion to Section 10.6, one level of formality above this chapter.
- Lee et al. (1997) for the bullwhip effect — the classic analysis of how forecast errors and ordering behavior amplify up a supply chain, and the reason Section 10.14.1 treats forecast bias as an operations problem rather than a statistics problem.
- The statsmodels documentation for the three functions this chapter's labs depend on — seasonal_decompose, ExponentialSmoothing, and HoltWintersResults.simulate (statsmodels developers, 2026a, 2026b, 2026c) — and the development release notes for the forthcoming 0.15.0 release (statsmodels developers, 2026d) for the simulation seed argument's planned rename from random_state to rng, which is the reason Section 10.12 pins a stable version and states it. The library itself is Seabold and Perktold (2010).

10.20 References

Chatfield, C. (1993). Calculating interval forecasts. *Journal of Business & Economic Statistics, 11*(2), 121–135. https://doi.org/10.1080/07350015.1993.10509938

Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics, 6*(1), 3–73.

Fildes, R., Goodwin, P., Lawrence, M., & Nikolopoulos, K. (2009). Effective forecasting and judgmental adjustments: An empirical evaluation and strategies for improvement in supply-chain planning. *International Journal of Forecasting, 25*(1), 3–23. https://doi.org/10.1016/j.ijforecast.2008.11.010

Fildes, R., Kolassa, S., & Ma, S. (2022a). Post-script—Retail forecasting: Research and practice. *International Journal of Forecasting, 38*(4), 1319–1324. https://doi.org/10.1016/j.ijforecast.2021.09.012

Fildes, R., Ma, S., & Kolassa, S. (2022b). Retail forecasting: Research and practice. *International Journal of Forecasting, 38*(4), 1283–1318. https://doi.org/10.1016/j.ijforecast.2019.06.004

Gardner, E. S., Jr. (2006). Exponential smoothing: The state of the art—Part II. *International Journal of Forecasting, 22*(4), 637–666. https://doi.org/10.1016/j.ijforecast.2006.03.005

Holt, C. C. (2004). Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting, 20*(1), 5–10. https://doi.org/10.1016/j.ijforecast.2003.09.015

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice* (3rd ed.). OTexts. https://otexts.com/fpp3/

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting, 22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Lee, H. L., Padmanabhan, V., & Whang, S. (1997). Information distortion in a supply chain: The bullwhip effect. *Management Science, 43*(4), 546–558. https://doi.org/10.1287/mnsc.43.4.546

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting, 36*(1), 54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014

Seabold, S., & Perktold, J. (2010). statsmodels: Econometric and statistical modeling with Python. In *Proceedings of the 9th Python in Science Conference* (pp. 92–96). https://doi.org/10.25080/Majora-92bf1922-011

statsmodels developers. (2026a). *statsmodels.tsa.seasonal.seasonal_decompose* [Software documentation]. https://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.seasonal_decompose.html

statsmodels developers. (2026b). *statsmodels.tsa.holtwinters.ExponentialSmoothing* [Software documentation]. https://www.statsmodels.org/stable/generated/statsmodels.tsa.holtwinters.ExponentialSmoothing.html

statsmodels developers. (2026c). *statsmodels.tsa.holtwinters.HoltWintersResults.simulate* [Software documentation]. https://www.statsmodels.org/stable/generated/statsmodels.tsa.holtwinters.HoltWintersResults.simulate.html

statsmodels developers. (2026d). *Development release notes for the forthcoming 0.15.0 release* [Software documentation]. https://www.statsmodels.org/dev/release/version0.15.0.html

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: An analysis and review. *International Journal of Forecasting, 16*(4), 437–450. https://doi.org/10.1016/S0169-2070(00)00065-0

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science, 6*(3), 324–342. https://doi.org/10.1287/mnsc.6.3.324