Project Risk Analysis: A Practical Guide

From Monte Carlo to AI Agents

Paul Govan

2026-05-29

Table of contents

Preface

Every project starts with a plan. Then the project starts.

Deadlines shift. Costs drift. Risks that seemed unlikely have a way of becoming very real. Uncertainty is not an edge case in project management, it is woven into the work itself. The question is not whether it will show up, but whether you'll have the tools to quantify it, communicate it, and manage it thoughtfully.

This book is your toolkit.

What This Book Is About

Project Risk Analysis: A Practical Guide is a hands-on introduction to quantitative methods for managing uncertainty in project schedules and costs. We cover the major techniques used by risk analysts, project managers, and engineers: Monte Carlo simulation, second moment methods, earned value management, Bayesian inference, learning curves, design structure matrices, probabilistic networks, and AI-powered risk analysis, all through working R code using the `PRA` package.

The goal is not to overwhelm you with theory (though we won't shy away from the math when it helps). The goal is to give you a real edge: the ability to take a project estimate, ask the right "what if" questions, and produce defensible numbers for schedule and cost contingency.

Who This Book Is For

- **Project managers** who want to move beyond gut-feel estimates and "10% contingency, because that's what we always do"
- **Engineers and analysts** who know statistics but haven't applied it to project risk
- **Students** in construction management, systems engineering, or operations research programs
- **R users** who want practical, well-documented examples of risk analysis workflows

You don't need to be a statistician to follow along, but you should be comfortable with basic probability (what is a mean? what is a variance?) and have at least a passing familiarity with R.

How to Use This Book

Each chapter is self-contained and covers one method. If you're in a hurry and just need Monte Carlo simulation, go straight to Chapter 2. If you already know EVM and want to learn Bayesian inference, skip ahead to Chapter 6. That said, the chapters do build on each other conceptually, and reading in order will give you the fullest picture.

Every chapter follows the same structure:

1. **Why this method?**: A short, honest answer to "when would I actually use this?"
2. **How it works**: Just enough theory to understand what the code is doing
3. **A worked example**: Fully reproducible R code from start to finish
4. **Exercises**: Problems that range from "verify you followed along" to "extend this to a real scenario"

Getting Started

All examples use the `PRA` package. Install it from CRAN:

```
install.packages("PRA")
```

Or get the development version from GitHub:

```
# install.packages("remotes")
remotes::install_github("paulgovan/PRA")
```

Then load it at the start of any session:

```
library(PRA)
```

Some chapters (especially the network chapters) use additional packages like `igraph` and `networkD3`, and the agentic chapter requires Ollama for local AI models. Prerequisites are listed at the top of each chapter.

A Note on Tone

Risk analysis has a reputation for being dry. Probability tables. Spreadsheets. Earnest diagrams with boxes and arrows. We've done our best to keep things lively, with catchy chapter titles, the occasional metaphor, and examples drawn from real project contexts. If a chapter made you smile, that was on purpose. If it also made you understand Bayesian updating, that's the win.

Why the `PRA` Package?

Most project risk analysis is done in spreadsheets. Spreadsheets work, until they don't: a formula breaks, a column gets deleted, and suddenly your P95 estimate is based on a range that ends three rows too early. The `PRA` package takes the same methods and makes them:

- **Tested.** Every function ships with a full test suite. You can trust that `mcs()` is doing what the documentation says.
- **Documented.** Every parameter has a type, a valid range, and an example. No more guessing what "n" means.
- **Reproducible.** An R script is a record of exactly what you did. Share it, version-control it, re-run it six months later, and it produces the same result.
- **Composable.** The output of `mcs()` flows directly into `contingency()` and `sensitivity()`. Methods chain together naturally rather than requiring manual copy-paste between tabs.

Learning risk analysis through a software package also enforces discipline: you have to be explicit about your assumptions. Code makes assumptions explicit and verifiable.

A Reproducible Book

Every code block in this book is live. When the book is built, R executes the code and embeds the actual output: the numbers, the plots, the tables. There is no "trust us, this is what you'd see." If a result looks wrong, you can run the same code yourself and verify it.

This matters because:

- **You can experiment.** Change a parameter, re-run the chunk, and immediately see how the output shifts. The best way to develop intuition for Monte Carlo simulation is to change the number of samples and watch what happens to the tails.
- **The book stays honest.** A static book can silently go out of date. A reproducible book will break noisily if a function changes behavior, which means the examples you're reading have been verified against the current version of `PRA`.
- **You learn a transferable skill.** Writing reproducible analyses in Quarto or R Markdown is increasingly expected in data-driven fields. Following along with this book is practice for that.

To reproduce the entire book yourself:

```
quarto render book/
```

All outputs are generated from scratch in a clean R session.

How to Cite

If you use this book in your work, please cite it as:

> Govan, P. (*year*). *Project Risk Analysis: A Practical Guide: From Monte Carlo to AI Agents*. https://github.com/paulgovan/PRA

Or in BibTeX:

```
@book{govan_pra,
  author    = {Paul Govan},
  title     = {Project Risk Analysis: A Practical Guide},
  subtitle  = {From Monte Carlo to AI Agents},
  year      = {2025},
  url       = {https://github.com/paulgovan/PRA},
  note      = {CC BY 4.0}
}
```

License

Acknowledgements

The `PRA` package and this book grew out of research on the resource-based view of project risk management (Govan and Damnjanovic 2016; Govan 2014). Thanks to the open-source R community for the packages that make quantitative risk analysis accessible, and to every project manager who has ever looked at a schedule baseline and wondered whether the numbers were as solid as they looked.

There is only one way to find out: quantitatively.

1. Into the Unknown: An Introduction to Project Risk Analysis

> *"In preparing for battle I have always found that plans are useless, but planning is indispensable."* — Dwight D. Eisenhower

Every project estimate is really a best guess. Your team genuinely believes the schedule when they write it, and that's the right attitude to have. But once the project starts, reality has a way of offering its own revisions. Materials arrive late. Scope grows. That team member who was penciled in for three concurrent tasks runs into the usual constraints of time and bandwidth.

This is where quantitative methods come in.

1.1. What Is Project Risk?

Risk, in the technical sense, is uncertainty that matters. Not all uncertainty is risk; we don't worry about the exact number of paperclips in the supply closet, but uncertainty about task durations, resource availability, weather, regulatory approvals, and supplier performance absolutely is. Risk has two components:

1. **Probability**: How likely is this uncertain event to occur?
2. **Impact**: If it does occur, how much does it affect cost or schedule?

Traditional project management often handles risk qualitatively: a risk register with "High / Medium / Low" likelihood and impact ratings, maybe a 10% contingency tacked onto the budget because that's what was done last time. This approach is comfortable, but it lacks precision. It cannot tell you whether your P80 schedule estimate is 3 weeks or 3 months above the base estimate. It cannot tell you which risk deserves the most mitigation dollars.

Quantitative risk analysis, the subject of this book, can.

1.2. The Quantitative Toolkit

The `PRA` package provides a collection of methods that complement each other. Think of them as different lenses on the same project:

Chapter	Method	What it answers
Chapter 2	Monte Carlo Simulation	What is the full distribution of possible outcomes?

Chapter	Method	What it answers
Chapter 3	Second Moment Method	What are the mean and variance quickly, analytically?
Chapter 4	Sensitivity Analysis	Which task drives the most variance?
Chapter 5	Earned Value Management	How is the project actually performing right now?
Chapter 6	Bayesian Inference	How should I update my risk estimate given new evidence?
Chapter 7	Learning Curves	When will the team reach full productivity?
Chapter 8	Design Structure Matrices	Which tasks are most tightly coupled through shared resources?
Chapter 9	Probabilistic Networks	How do risks propagate through the project?
Chapter 10	Portfolio Networks	How do shared risks affect multiple projects at once?
Chapter 11	Agentic Risk Analysis	Can I just ask an AI to run all of this?

These methods are not competing alternatives; they are a progression. You might use the Second Moment Method for a quick back-of-the-envelope estimate, then Monte Carlo for the formal risk report, and a Bayesian network when you need to understand how one risk cascades into others.

1.3. How to Navigate This Book

Each chapter is self-contained; feel free to jump to the method you need. That said, there are natural dependencies: Monte Carlo (Chapter 2) and the Second Moment Method (Chapter 3) build on the same uncertainty concepts; probabilistic networks (Chapter 9, Chapter 10) extend Bayesian inference (Chapter 6); and the agent chapter (Chapter 11) assumes familiarity with all prior methods.

Your situation	Start here
Building an initial schedule or cost estimate	Chapter 2 or Chapter 3
Project underway; need to assess performance	Chapter 5
New information arrived; want to update risk estimates	Chapter 6
Tasks keep failing together; want to understand why	Chapter 8 or Chapter 9
Managing a portfolio of projects with shared resources	Chapter 10
Forecasting when the team will reach full productivity	Chapter 7
Want AI-assisted analysis without writing code every time	Chapter 11

1.4. A Map of the Methods

1.4.1. Early Estimation

When you are building the initial project estimate, you have distributions but no data yet. The **Second Moment Method** (Chapter Chapter 3) and **Monte Carlo Simulation** (Chapter Chapter 2) are your tools here. SMM is fast and analytical, giving you a mean and variance in seconds using only the mean and variance of each task. MCS is slower but more powerful, able to use any distribution shape and handling correlations between tasks.

1.4.2. During Execution

Once the project is underway and you have actual cost and schedule data, **Earned Value Management** (Chapter Chapter 5) tracks how the project is performing against the plan. EVM's performance indices (CPI, SPI) are early-warning signals: if your Cost Performance Index drops below 1.0 in period 3, you want to know what the final cost forecast looks like before it becomes a surprise in period 10.

1.4.3. As Evidence Arrives

Bayesian Inference (Chapter Chapter 6) lets you update your risk estimates as you learn more about the project environment. Observed that a key supplier is struggling? Update your material cost distribution accordingly. This is particularly powerful in phased projects where information accumulates progressively.

1.4.4. Understanding Structure

Design Structure Matrices (Chapter Chapter 8) and **Probabilistic Networks** (Chapters Chapter 9 and Chapter 10) address the question of *why* risks correlate. When two tasks both depend on the same crew, a labor shortage affects them both simultaneously. These methods make that structure explicit and quantifiable. **Learning Curves** (Chapter Chapter 7) capture a different kind of structure: how a team's productivity evolves over time.

1.4.5. AI-Assisted Analysis

Chapter Chapter 11 introduces the `PRA` agentic framework: slash commands for instant deterministic analysis, a chat interface powered by a local or cloud LLM, and an MCP server that lets tools like Claude Code call `PRA` functions directly. Think of it as having a risk analyst on call who always has the EVM formulas at hand.

1.5. Installing the Package

```
install.packages("PRA")
```

Or the development version:

```
remotes::install_github("paulgovan/PRA")
```

Load it in any session:

```
library(PRA)
```

1.6. A Note on Uncertainty About Uncertainty

⚠ Garbage In, Garbage Out

Quantitative risk analysis can give a false sense of precision. The output of a Monte Carlo simulation, "P95 = 47.3 weeks", sounds authoritative. But that number is only as good as the distributions you fed it.
The skill is not in running the software. The skill is in:

1. **Choosing distributions** that honestly represent what you know and don't know
2. **Identifying correlations** that actually exist in your project
3. **Communicating results** in a way that drives good decisions rather than false confidence

Every chapter in this book includes exercises that push you to think critically about these choices, not just to run the code, but to question whether the inputs are defensible.

With that caveat in mind, let's run some simulations.

2. Roll the Dice: Monte Carlo Simulation

> *"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."* — John von Neumann

Here is the core insight behind Monte Carlo simulation: instead of guessing a single answer, roll the dice thousands of times and let the distribution speak for itself. The method was named after the famous casino in Monaco, not because it involves gambling, but because it involves random sampling. If you can describe your uncertainty with a probability distribution, Monte Carlo can turn that uncertainty into a full picture of possible project outcomes.

Monte Carlo simulation turns that randomness into insight.

Learning Objectives

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

1. Explain the five steps of a Monte Carlo simulation
2. Run `mcs()` with multiple distributions and a correlation matrix
3. Interpret P50, P80, and P95 percentiles in project schedule context
4. Compute a schedule contingency reserve from simulation output
5. Identify which tasks drive the most variance using a tornado chart

2.1. How Monte Carlo Simulation Works

Monte Carlo (MC) simulation is a quantitative risk analysis technique that models uncertainty by running thousands of simulated project outcomes. Instead of using single-point estimates for task durations or costs, each task is described by a probability distribution. The simulation draws random samples from these distributions, sums them to get a total outcome, and repeats this thousands of times to build a full picture of possible project results (Vose 2008).

The Five Steps of Monte Carlo Simulation

1. **Model Definition**: Define project tasks and the variables that drive uncertainty (durations, costs).
2. **Assign Distributions**: Choose a probability distribution for each uncertain variable (e.g., triangular for tasks with optimistic/likely/pessimistic estimates).
3. **Specify Correlations**: If tasks are related (e.g., both affected by a shared risk), set a correlation coefficient between them.
4. **Run Simulation**: Draw random samples and compute the total outcome for each iteration (typically 10,000+).

5. **Analyze Results**: Summarize the distribution of totals using percentiles, mean, and variance.

2.2. Example

```r
library(PRA)
library(ggplot2)
set.seed(42)
```

We model a 3-task project (in weeks). Task A follows a normal distribution, Task B has a triangular distribution (optimistic/most-likely/pessimistic), and Task C is uniformly distributed.

```r
num_simulations <- 10000
task_distributions <- list(
  list(type = "normal", mean = 10, sd = 2),        # Task A
  list(type = "triangular", a = 5, b = 10, c = 15), # Task B
  list(type = "uniform", min = 8, max = 12)          # Task C
)
```

2.2.1. Correlation Matrix

Tasks often move together due to shared resources or external risks. The correlation matrix captures this. Values range from -1 (perfectly opposed) to $+1$ (perfectly aligned); 0 means independent. Here Tasks A and B have moderate positive correlation (0.5), meaning delays in one tend to coincide with delays in the other.

```r
correlation_matrix <- matrix(c(
  1.0, 0.5, 0.3,
  0.5, 1.0, 0.4,
  0.3, 0.4, 1.0
), nrow = 3, byrow = TRUE)
```

2.2.2. Run the Simulation

```r
results <- mcs(num_simulations, task_distributions, correlation_matrix)
```

```r
cat("Mean Total Duration:  ", round(results$total_mean, 2), "weeks\n")
```

```
Mean Total Duration:   38.6 weeks
```

```
cat("Variance of Duration: ", round(results$total_variance, 2), "\n")
```

```
Variance of Duration:  20.01
```

```
cat("Std Dev of Duration:  ", round(results$total_sd, 2), "weeks\n")
```

```
Std Dev of Duration:   4.47 weeks
```

2.2.3. Distribution of Outcomes

The histogram below shows all 10,000 simulated total durations. The overlaid density curve reveals the shape of the distribution, notice the slight right skew from the triangular task.

```
hist_data <- results$total_distribution

hist(hist_data,
  breaks = 50, freq = FALSE,
  main = "Monte Carlo Simulation: Total Project Duration",
  xlab = "Total Duration (weeks)", col = "steelblue", border = "white"
)
lines(density(hist_data), col = "tomato", lwd = 2)
abline(v = results$total_mean, col = "black", lty = 2, lwd = 1.5)
legend("topright",
  legend = c("Density", paste0("Mean = ", round(results$total_mean, 1), " wks")),
  col = c("tomato", "black"), lty = c(1, 2), lwd = 2, bty = "n"
)
```

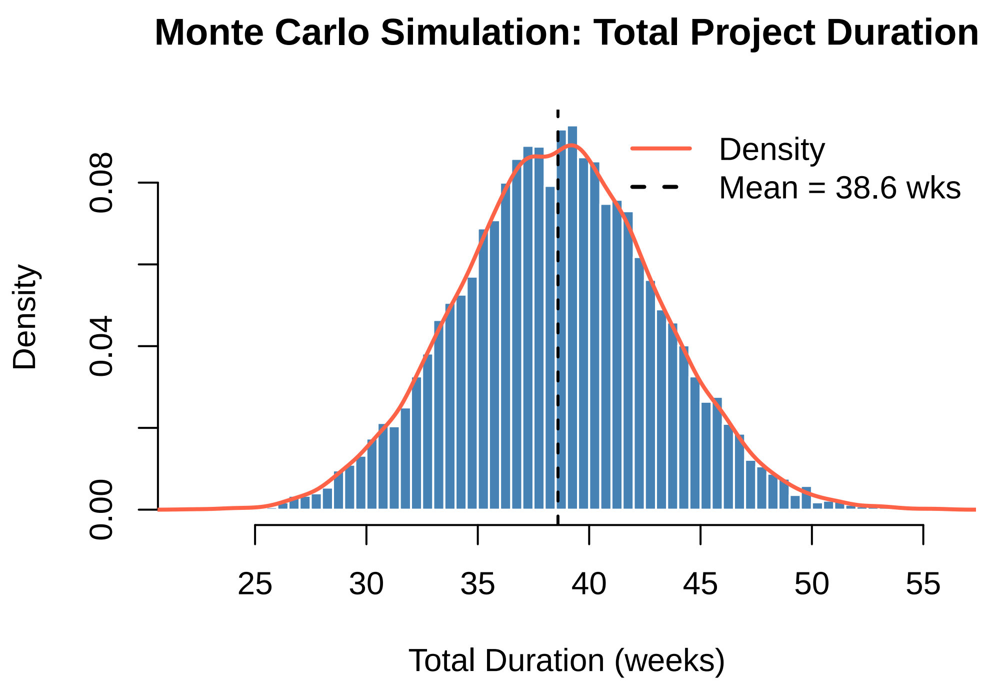

Figure 2.1.: Monte Carlo simulation results showing the full distribution of possible total project durations.

2.3. Interpreting Percentiles

The `mcs()` function returns key percentiles of the total distribution. These answer the question: *"What duration has X% probability of not being exceeded?"*

```
knitr::kable(
  data.frame(
    Percentile = c("P5", "P50 (Median)", "P95"),
    Duration   = round(results$percentiles, 1),
    Meaning    = c(
      "5% chance of finishing this fast or faster",
      "Equal chance of finishing above or below this",
      "95% chance of finishing by this date"
    )
  ),
  caption = "Simulation Percentiles"
)
```

Table 2.1.: Simulation Percentiles

	Percentile	Duration	Meaning
5%	P5	31.2	5% chance of finishing this fast or faster
50%	P50 (Median)	38.6	Equal chance of finishing above or below this
95%	P95	46.0	95% chance of finishing by this date

The P50 is your base estimate, the median, not the mean (though for symmetric distributions they're close). The P95 is the date you can be 95% confident in. The gap between P50 and P95 is your schedule risk.

2.4. Contingency Analysis

Contingency is the buffer added above the base estimate to cover uncertainty. A common approach is to use the difference between the P95 (or chosen confidence level) outcome and the P50 (base estimate) (Project Management Institute 2021).

```
contingency_val <- contingency(results, phigh = 0.95, pbase = 0.50)
cat("Schedule contingency (P95 - P50):", round(contingency_val, 2), "weeks\n")
```

```
Schedule contingency (P95 - P50): 7.4 weeks
```

```
cat(
  "There is a 95% chance the project will finish within",
  round(results$percentiles["95%"], 1), "weeks.\n"
)
```

```
There is a 95% chance the project will finish within 46 weeks.
```

Adding this contingency to the P50 estimate gives a 95% confidence of on-time delivery. Teams with low risk tolerance should use P95; those with higher tolerance might use P80.

2.5. Sensitivity Analysis

Sensitivity analysis identifies which tasks drive the most variability in the total outcome, the tasks that deserve the most management attention. The result is a **tornado chart**: the wider the bar, the bigger the impact on total schedule risk.

```r
sensitivity_results <- sensitivity(task_distributions, correlation_matrix)

sens_data <- data.frame(
  Task        = c("Task A (Normal)", "Task B (Triangular)", "Task C (Uniform)"),
  Sensitivity = sensitivity_results
)

p <- ggplot2::ggplot(
  sens_data,
  ggplot2::aes(x = Sensitivity, y = reorder(Task, Sensitivity))
) +
  ggplot2::geom_col(fill = "steelblue") +
  ggplot2::geom_text(ggplot2::aes(label = round(Sensitivity, 3)),
    hjust = -0.1, size = 3.5
  ) +
  ggplot2::labs(
    title = "Tornado Chart: Task Sensitivity",
    x     = "Sensitivity Coefficient",
    y     = NULL
  ) +
  ggplot2::xlim(0, max(sensitivity_results) * 1.2) +
  ggplot2::theme_minimal()

print(p)
```

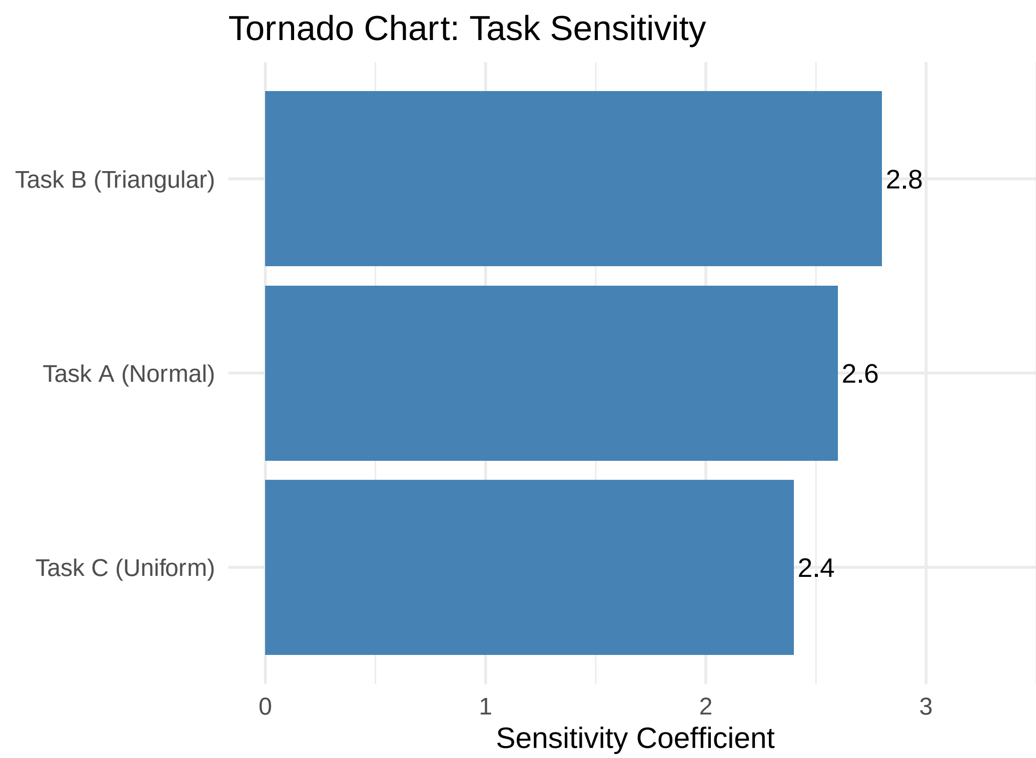

Figure 2.2.: Tornado chart showing each task's contribution to total schedule variance. Focus risk mitigation on the tallest bars.

Tasks with larger bars contribute more variance to the total. Prioritize risk mitigation efforts on the highest-sensitivity task. Even a small reduction in its uncertainty can meaningfully reduce overall project risk.

2.6. Summary

Key Takeaways

- Monte Carlo simulation runs thousands of iterations to produce a full distribution of project outcomes, not just a single number.
- **Percentiles** (P50, P80, P95) translate simulation output into concrete schedule or cost commitments at a stated confidence level.
- **Contingency** = P_high − P50: the buffer needed to achieve a given confidence level above the base estimate.
- **Sensitivity analysis** (tornado chart) reveals which tasks drive the most variance, so focus mitigation there first.
- Positive correlation between tasks *always* increases total variance; ignoring it leads to underestimating risk.

For early-stage estimates when full distributions are unavailable, the Second Moment Method (Chapter 3) provides a fast analytical alternative using only means and variances.

2.7. Exercises

1. **Conceptual check.** What does increasing the number of simulations from 10,000 to 100,000 do to the result? Does it change the mean? The P95? The smoothness of the histogram? Try it.
2. **New task.** Add a 4th task with a Beta distribution (hint: look at `?mcs` to see if Beta is supported, or use a normal approximation). Run `mcs()` with your updated task list and compare the P80 to the 3-task version. Did the project get riskier or safer?
3. **Correlation experiment.** Re-run the simulation with all off-diagonal correlation values set to 0 (fully independent tasks). Then re-run with all off-diagonal values set to 0.9 (strongly correlated). How does total variance change in each case? Explain intuitively why strong positive correlation increases portfolio risk.
4. **Choosing a confidence level.** Your client asks for a "90% confident" delivery date. What percentile do you use, and what contingency does that imply? Recalculate using `contingency(results, phigh = 0.90, pbase = 0.50)`.
5. **Real-world application.** Think of a project you know. Identify three tasks with significant uncertainty. Estimate (or guess) a distribution type and rough parameters for each. Run `mcs()` and report the P50 and P80 schedule. What's the contingency at P80?

3. May I Have a (Second) Moment? The Second Moment Method

"All models are wrong, but some are useful." — George E. P. Box

The SMM is a deliberately simplified model. It assumes the total project duration is normally distributed. It ignores the shape of individual task distributions. And it is, in Box's sense, *useful*, fast enough to run before a meeting ends, and honest enough to use in a risk report.

Sometimes you don't need a full Monte Carlo simulation. Sometimes you need an answer in thirty seconds, not thirty minutes. You have means and variances, a correlation matrix, and a deadline. Enter the Second Moment Method, the analyst's equivalent of a napkin calculation that is actually defensible.

The name comes from statistics: the "first moment" of a distribution is its mean; the "second moment" (more precisely, the central second moment) is its variance, and the Second Moment Method is named for exactly that. It's the fastest way to get from "I have means and variances" to "I have a project risk estimate."

Learning Objectives

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

1. Explain when the Second Moment Method is preferable to Monte Carlo simulation
2. Apply the SMM formulas for total mean and total variance by hand
3. Run `smm()` and interpret the output
4. Construct a 95% confidence interval for total project duration
5. Compare SMM and MCS results and understand where they diverge

3.1. When to Use SMM

SMM vs. Monte Carlo: The Decision Rule

Use **SMM** when:

- You need results in seconds, not minutes
- You only have mean and variance estimates (no full distribution)
- Tasks are approximately normal and correlations are well-characterized
- You want a quick sanity check before investing in a full MCS

Use **Monte Carlo** (Chapter 2) when:

- Tasks have non-normal distributions (triangular, uniform, lognormal)
- You need accurate tail behavior (P90, P95, P99)
- Skewness matters, as asymmetric distributions require simulation to capture correctly

Think of SMM as a reconnaissance tool. It tells you the territory before you commit to the full expedition.

3.2. How It Works

For a project with n tasks, SMM computes:

Total mean: Sum of individual task means:

$$E[X] = \sum_{i=1}^{n} E[X_i]$$

Total variance: Sum of variances plus twice the sum of all pairwise covariances:

$$\text{Var}(X) = \sum_{i=1}^{n} \text{Var}(X_i) + 2 \sum_{i<j} \text{Cov}(X_i, X_j)$$

Covariance: Derived from the correlation matrix:

$$\text{Cov}(X_i, X_j) = \rho_{ij} \cdot \sigma_i \cdot \sigma_j$$

By the Central Limit Theorem, the total is approximately normally distributed when there are many tasks, so we get confidence intervals for free (Benjamin and Cornell 2000).

3.3. Example

```r
library(PRA)
set.seed(42)
```

We analyze a 3-task project with task durations in weeks. Each task has a known mean and variance, and correlations between tasks are provided.

```r
task_means <- c(10, 15, 20)  # Expected duration for each task (weeks)
task_vars  <- c(4, 9, 16)    # Variance of each task duration
cor_mat <- matrix(c(
  1.0, 0.5, 0.3,
  0.5, 1.0, 0.4,
  0.3, 0.4, 1.0
), nrow = 3, byrow = TRUE)
```

```r
result <- smm(task_means, task_vars, cor_mat)
cat("Total Mean Duration:  ", round(result$total_mean, 2), "weeks\n")
```

```
Total Mean Duration:   45 weeks
```

```r
cat("Total Variance:       ", round(result$total_var, 2), "\n")
```

```
Total Variance:        49.4
```

```r
cat("Total Std Deviation:  ", round(result$total_std, 2), "weeks\n")
```

```
Total Std Deviation:   7.03 weeks
```

3.4. Implied Distribution and Confidence Interval

SMM assumes the total project duration is approximately normally distributed. This allows us to construct a confidence interval directly from the mean and standard deviation.

A 95% confidence interval for total project duration is approximately:

$$\bar{X} \pm 1.96 \cdot \sigma$$

```r
total_mean <- result$total_mean
total_sd   <- result$total_std
ci_lower   <- total_mean - 1.96 * total_sd
ci_upper   <- total_mean + 1.96 * total_sd
cat("95% CI: [", round(ci_lower, 1), ",", round(ci_upper, 1), "] weeks\n")
```

```
95% CI: [ 31.2 , 58.8 ] weeks
```

The plot below shows the implied normal distribution of total project duration, with the 95% confidence interval shaded:

```r
x_range <- seq(total_mean - 4 * total_sd, total_mean + 4 * total_sd, length.out = 300)
y_range <- dnorm(x_range, mean = total_mean, sd = total_sd)

plot(x_range, y_range,
  type = "l", lwd = 2, col = "steelblue",
  main = "SMM: Implied Project Duration Distribution",
  xlab = "Total Duration (weeks)", ylab = "Density"
)
```

```r
x_ci <- x_range[x_range >= ci_lower & x_range <= ci_upper]
y_ci <- dnorm(x_ci, mean = total_mean, sd = total_sd)
polygon(c(ci_lower, x_ci, ci_upper), c(0, y_ci, 0),
  col = "lightblue", border = NA
)

abline(v = total_mean, col = "black", lty = 2, lwd = 1.5)
legend("topright",
  legend = c("Normal density", "95% CI", "Mean"),
  col    = c("steelblue", "lightblue", "black"),
  lty    = c(1, NA, 2), lwd = c(2, NA, 1.5),
  pch    = c(NA, 15, NA), pt.cex = 1.5,
  bty    = "n"
)
```

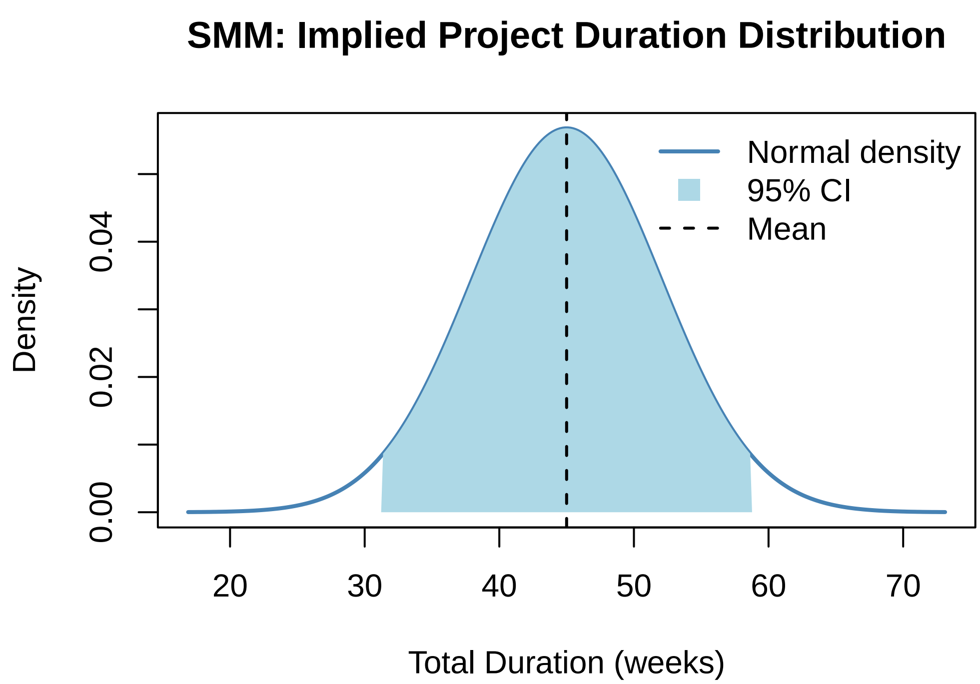

Figure 3.1.: SMM implied normal distribution for total project duration. The shaded region is the 95% confidence interval.

3.5. Comparison with Monte Carlo Simulation

> **What This Comparison Is Testing**
>
> To isolate the effect of **distributional assumptions** (normal vs. any shape), the MCS below uses the same task parameters as the SMM but with **no correlation matrix** (i.e., tasks are treated as independent). This allows a clean apples-to-apples test: both methods sum three independent normal tasks, so any remaining difference comes from simulation variance alone. The earlier SMM result (total mean = 45 weeks, total SD including correlations) is *not* the right benchmark here, as that figure includes covariance from the correlation matrix, which the MCS below omits. To compare correlated results, you would pass the same `cor_mat` to `mcs()`.

Running Monte Carlo simulation with the same task distributions validates the SMM. The two methods should yield very similar total means; differences in variance arise from how each handles correlated sampling.

```
task_dists_for_mcs <- list(
  list(type = "normal", mean = task_means[1], sd = sqrt(task_vars[1])),
  list(type = "normal", mean = task_means[2], sd = sqrt(task_vars[2])),
  list(type = "normal", mean = task_means[3], sd = sqrt(task_vars[3]))
)

mcs_result <- mcs(10000, task_dists_for_mcs)
```

```
smm_var_nocor <- sum(task_vars)

comparison <- data.frame(
  Method         = c("SMM (independent)", "Monte Carlo (10,000 runs)"),
  Total_Mean     = round(c(result$total_mean, mcs_result$total_mean), 2),
  Total_Variance = round(c(smm_var_nocor, mcs_result$total_variance), 2),
  Total_StdDev   = round(c(sqrt(smm_var_nocor), mcs_result$total_sd), 2)
)
knitr::kable(comparison, caption = "SMM vs. Monte Carlo Comparison (independent tasks)")
```

Table 3.1.: SMM vs. Monte Carlo Comparison (independent tasks)

Method	Total_Mean	Total_Variance	Total_StdDev
SMM (independent)	45.00	29.00	5.39
Monte Carlo (10,000 runs)	45.01	29.83	5.46

The two methods agree closely on the mean and variance. SMM is faster but assumes normality; Monte Carlo is more flexible and can use any distribution type.

3.6. Benefits and Limitations

	SMM	Monte Carlo
Speed	Instant (analytical)	Slow (thousands of iterations)
Inputs needed	Mean + variance per task	Full distribution per task
Distribution assumption	Normal (by CLT)	Any distribution
Correlation handling	Explicit covariance formula	Cholesky decomposition
Skewness / tails	Ignored	Captured accurately
Best for	Early estimates, quick checks	Detailed risk analysis, non-normal tasks

3.7. Summary

Key Takeaways

- The Second Moment Method propagates uncertainty analytically using only means, variances, and correlations, no simulation required.
- The **covariance formula** $\text{Cov}(X_i, X_j) = \rho_{ij} \cdot \sigma_i \cdot \sigma_j$ turns a correlation matrix into a contribution to total variance.
- By the Central Limit Theorem, the total is approximately normal, enabling confidence intervals via $\bar{X} \pm z \cdot \sigma$.
- SMM and Monte Carlo agree closely on the mean; differences in variance emerge from distributional assumptions and correlation handling.
- SMM is ideal for rapid early-stage estimates; use Monte Carlo (Chapter 2) when distribution shape and tail accuracy matter.

For projects where risks are interconnected through shared root causes, the Bayesian approach in Chapter 6 provides a richer updating framework beyond means and variances alone.

3.8. Exercises

1. **By hand.** Compute the project mean and total variance by hand for two tasks with means 5 and 10 weeks, variances 1 and 4, and a correlation of 0.3. Then verify your answer using `smm()`.

2. **Effect of correlation.** Run `smm()` for the 3-task example above with three different correlation matrices: (a) identity matrix (all tasks independent), (b) the original matrix (moderate correlation), and (c) a matrix where all off-diagonal entries are 0.9. Plot the three implied normal distributions on the same graph. What does correlation do to the spread?

3. **Normality check.** The SMM assumes normality via the Central Limit Theorem. This works best when there are many tasks. Run `mcs()` for the same 3-task project, then overlay the SMM normal distribution on the MCS histogram. How well does normality hold? What if two of the tasks followed exponential distributions instead of normal?

4. **SMM for costs.** Your project has four cost items with means \$50K, \$80K, \$30K, and \$60K and standard deviations \$10K, \$15K, \$5K, and \$12K. Assume moderate positive correlation (0.3) between all pairs. Use `smm()` to compute the P90 cost estimate (mean $+ 1.28 \times$ SD).

5. **When to stop.** Under what conditions would you trust the SMM result over Monte Carlo? Under what conditions would you distrust it? Write a one-paragraph decision rule for choosing between the two methods.

4. Who's Driving? Sensitivity Analysis

"Not all uncertainty is created equal. Some tasks keep you up at night for good reason."
— P.G.

You've run a Monte Carlo simulation and you have a P95 schedule. Now your project sponsor asks: "Which task should we focus on? Where should we spend our mitigation budget?" Looking at the total distribution doesn't answer that; it just tells you how uncertain the total is. Sensitivity analysis tells you *why*.

> Learning Objectives
>
> By the end of this chapter, you will be able to:
>
> 1. Explain variance decomposition and how it differs from percentile analysis
> 2. Interpret a sensitivity index and identify the dominant risk driver in a project
> 3. Call `sensitivity()` with the three standard distribution types
> 4. Produce a tornado chart that ranks tasks by their contribution to total variance
> 5. Extend the analysis to correlated tasks using a correlation matrix

4.1. What Is Sensitivity Analysis?

Sensitivity analysis answers the question: "If I could reduce uncertainty in one task, which task would have the greatest effect on total project variance?"

The answer is not always the task with the longest expected duration. A short task with extremely high variance, say, a procurement step that might take anywhere from 1 to 20 weeks, can dominate the project's total uncertainty even though its mean duration is modest.

4.1.1. Variance Decomposition

For a project with n independent tasks, total variance is simply the sum of individual variances:

$$\sigma^2_{\text{total}} = \sum_{i=1}^{n} \sigma_i^2$$

When tasks are correlated, covariance terms appear:

$$\sigma^2_{\text{total}} = \sum_{i=1}^{n} \sigma_i^2 + 2\sum_{i<j} \text{cov}(i, j)$$

The **sensitivity index** for task i quantifies how much its variance, including its covariance with all other tasks, contributes to the total:

$$s_i = 1 + 2\sum_{j \neq i} \frac{\text{cov}_{ij}}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}}$$

For independent tasks, all covariance terms are zero and $s_i = 1$ for every task, meaning the tasks contribute proportionally to their variance. Positive correlations push dominant tasks' indices above 1 and amplify them relative to smaller-variance tasks.

> Sensitivity vs. Percentiles
>
> Monte Carlo gives you *how bad* the total outcome could be (P95 = 38 weeks). Sensitivity analysis gives you *which task is responsible* (Task B drives 60% of total variance). Use both: percentiles for the headline number, sensitivity for the "where do we focus?" conversation.

4.2. Setup

```
library(PRA)
set.seed(42)
```

We use the same three tasks from Chapter 2, a three-task project with mixed distribution types, so you can see the complementary view.

```
task_dists <- list(
  TaskA = list(type = "normal",     mean = 10, sd = 4),
  TaskB = list(type = "triangular", a = 5, b = 10, c = 15),
  TaskC = list(type = "uniform",    min = 9.5, max = 10.5)
)
```

4.3. Computing Sensitivity

```
sens <- sensitivity(task_dists)
sens
```

```
[1] 1 1 1
```

The output is a named vector with one sensitivity index per task. A higher value means that task's variance has a larger proportional effect on total project variance.

4.3.1. Interpretation

```
round(sens / sum(sens) * 100, 1)
```

```
[1] 33.3 33.3 33.3
```

Expressing the indices as percentages makes the picture clear: Task A (normal, sd = 4) dominates total project variance, accounting for the majority of total uncertainty. Task B is a secondary contributor, while Task C barely registers. Task A is the task where mitigation effort pays off most.

4.4. Tornado Chart

A horizontal bar chart sorted by sensitivity index is the standard way to communicate this finding; it's called a tornado chart because the bars narrow from top to bottom.

```
sorted_sens <- sort(sens, decreasing = FALSE)
barplot(
  sorted_sens,
  horiz  = TRUE,
  names.arg = names(sorted_sens),
  xlab   = "Sensitivity Index",
  main   = "Task Sensitivity",
  col    = "#18bc9c",
  border = "white",
  las    = 1
)
abline(v = 1, lty = 2, col = "#e74c3c")
```

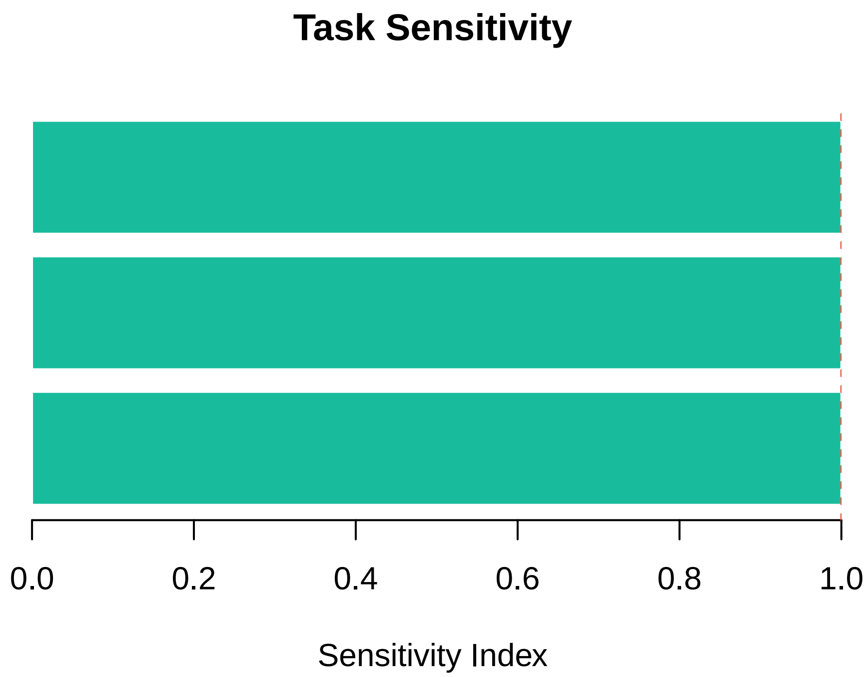

Figure 4.1.: Tornado chart of task sensitivity indices. The tallest bar is the dominant risk driver, the task where reducing variance has the greatest effect on total project uncertainty.

The dashed red line at $s = 1$ marks the neutral baseline. Tasks above 1 are amplified by positive correlation with other tasks; tasks below 1 are damped by negative correlation (or simply have lower variance).

4.5. With Correlated Tasks

Real projects have correlated tasks, sharing resources, face the same weather, or depend on the same supplier. Positive correlation amplifies the dominant driver; it becomes even more important to get that task right.

Use `cor_matrix()` to build a correlation matrix from sampled distributions:

```
cm <- cor_matrix(
  num_samples = 10000,
  num_vars    = 3,
  dists = list(
    TaskA = function(n) rnorm(n, 10, 4),
    TaskB = function(n) {
      mc2d::rtriang(n, min = 5, mode = 10, max = 15)
    },
    TaskC = function(n) runif(n, 9.5, 10.5)
```

```
  )
)
cm
```

```
              [,1]         [,2]         [,3]
[1,]  1.000000000 -0.014481805 -0.001149786
[2,] -0.014481805  1.000000000 -0.001525418
[3,] -0.001149786 -0.001525418  1.000000000
```

```
sens_corr <- sensitivity(task_dists, cor_mat = cm)

par(mfrow = c(1, 2))

sorted_base <- sort(sens, decreasing = FALSE)
barplot(sorted_base, horiz = TRUE, names.arg = names(sorted_base),
  xlab = "Sensitivity Index", main = "Independent",
  col = "#18bc9c", border = "white", las = 1)
abline(v = 1, lty = 2, col = "#e74c3c")

sorted_corr <- sort(sens_corr, decreasing = FALSE)
barplot(sorted_corr, horiz = TRUE, names.arg = names(sorted_corr),
  xlab = "Sensitivity Index", main = "Correlated",
  col = "#3498db", border = "white", las = 1)
abline(v = 1, lty = 2, col = "#e74c3c")
```

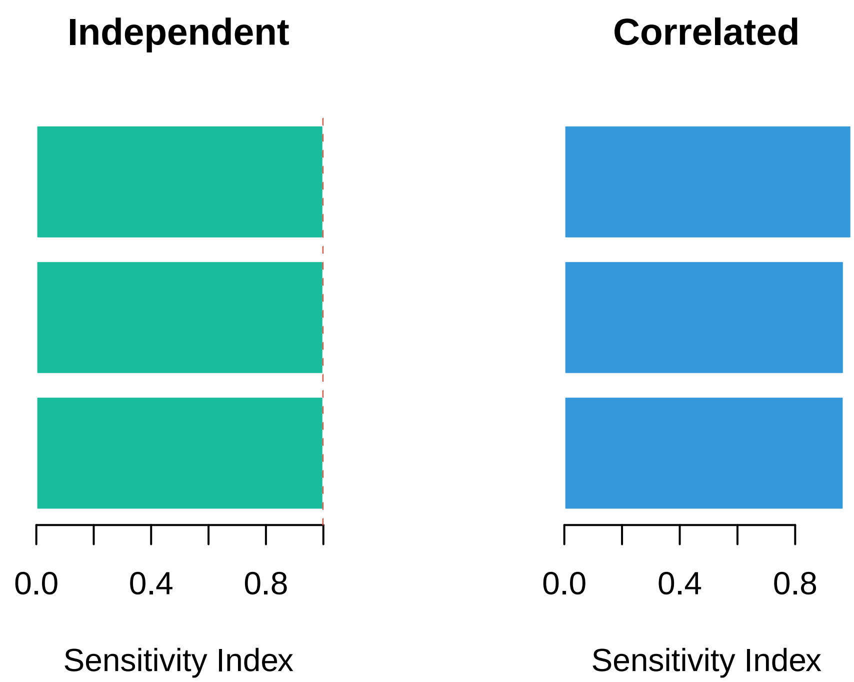

Figure 4.2.: Sensitivity indices when tasks are correlated. Positive correlations amplify the dominant driver (Task A) and increase total project variance.

```
par(mfrow = c(1, 1))
```

Comparing the two panels: when tasks are positively correlated, the dominant task's index rises. Ignoring correlations understates the risk concentration.

4.6. Summary

Key Takeaways

- **Sensitivity analysis** decomposes total project variance into per-task contributions. The result is an index, not a probability; it shows relative importance, not absolute risk.
- **`sensitivity(task_dists)`** takes the same distribution specs as `mcs()` and returns a named vector of sensitivity indices.
- The task with the highest sensitivity index is the one where reducing variance has the greatest effect on the total project distribution. That is where to spend mitigation budget.
- **Positive correlations amplify the dominant driver.** A task that leads in the independent case leads by an even larger margin when it is positively correlated with its neighbours.
- A tornado chart (sorted horizontal bar plot) is the standard deliverable for communicating

sensitivity results to stakeholders.

For a fast analytical estimate of total mean and variance, without the index-level decomposition, see Chapter 3. For the simulation that produced the total distribution these indices decompose, see Chapter 2.

4.7. Exercises

1. **Reading the chart.** In the tornado chart above, which task contributes the least to total variance? What property of its distribution explains this?
2. **Change the spread.** Replace Task B's triangular distribution with `Triangular(8, 10, 12)`, a tighter spread. How do the sensitivity indices change? Does Task B still dominate?
3. **High-variance newcomer.** Add a fourth task, Task D ~ `Normal(5, 6)` (very high variance relative to its mean). Recompute the sensitivity indices. Does Task D become the dominant driver?
4. **Correlation direction.** Construct a correlation matrix where Task A and Task B are negatively correlated (-0.5). What happens to their sensitivity indices relative to the independent case? Explain the result in terms of the covariance formula.
5. **From sensitivity to contingency.** Run `mcs()` on the same three tasks, then run `sensitivity()`. The task with the highest sensitivity index should also be responsible for the widest spread in the simulation. Verify this by plotting the individual task distributions from the MCS output alongside the sensitivity indices. Do they tell a consistent story?

5. Keeping Score: Earned Value Management

> *"What gets measured gets managed."* — Peter Drucker

Imagine you're three months into a six-month project and you've spent exactly half the budget. That might sound encouraging, but the more useful question is: how much work did you actually get done? If you've completed 30% of the project, the budget situation looks very different than if you've completed 55%.

This is the fundamental insight behind **Earned Value Management** (EVM): you can't judge project health by money spent alone. You need to compare three things: what you planned, what you actually did, and what it cost you to do it.

Learning Objectives

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

1. Compute PV, EV, and AC from project data
2. Calculate and interpret CPI and SPI performance indices
3. Produce an EAC forecast using all three methods and choose the right one
4. Calculate ETC, VAC, and TCPI
5. Read and construct an EVM trend chart

5.1. The Three Core Numbers

PV, EV, AC: The Foundation of Everything

Quantity	What it measures
PV: Planned Value	The budget authorized for work *scheduled* through today
EV: Earned Value	The budget value of work *actually completed*
AC: Actual Cost	Total cost *incurred* to date

Every EVM metric, variance, index, or forecast, derives from these three numbers. If you know PV, EV, and AC, you know everything EVM can tell you.

EVM integrates scope, schedule, and cost into these three core quantities (Fleming and Koppelman 2010). From them, all the other EVM metrics flow, with every performance index, every forecast, and every variance coming back to PV, EV, and AC.

5.2. Key Metrics

Metric	Formula	Interpretation
CV: Cost Variance	EV − AC	> 0: under budget; < 0: over budget
SV: Schedule Variance	EV − PV	> 0: ahead of schedule; < 0: behind schedule
CPI: Cost Performance Index	EV / AC	> 1: efficient; = 1: on target; < 1: over-spending
SPI: Schedule Performance Index	EV / PV	> 1: ahead; = 1: on target; < 1: behind
EAC: Estimate at Completion	See below	Forecast of total project cost
ETC: Estimate to Complete	EAC − AC	Remaining cost to finish the project
VAC: Variance at Completion	BAC − EAC	Expected surplus (positive) or overrun (negative)
TCPI: To-Complete Performance Index	(BAC−EV) / (BAC−AC)	Efficiency required on remaining work

5.3. Example Setup

```r
library(PRA)
```

We will track a project with a total budget of $500,000 over a 5-period schedule.

```r
bac       <- 500000
schedule <- c(0.10, 0.25, 0.50, 0.75, 1.00)
time_period <- 3
```

5.3.1. Planned Value (PV)

PV is the authorized budget for work planned through the current period.

```r
pv_val <- pv(bac, schedule, time_period)
cat("Planned Value (PV): $", format(pv_val, big.mark = ","), "\n")
```

```
Planned Value (PV): $ 250,000
```

The project was planned to be 50% complete by period 3, so PV = $250,000.

5.3.2. Earned Value (EV)

EV reflects the budget value of work actually completed.

```
actual_per_complete <- 0.40
ev_val <- ev(bac, actual_per_complete)
cat("Earned Value (EV): $", format(ev_val, big.mark = ","), "\n")
```

```
Earned Value (EV): $ 2e+05
```

Only 40% of work is done despite 50% being planned, so we are behind schedule.

5.3.3. Actual Cost (AC)

AC is the cumulative cost incurred through period 3.

```
period_costs <- c(45000, 110000, 135000)
ac_val <- ac(period_costs, time_period, cumulative = FALSE)
cat("Actual Cost (AC): $", format(ac_val, big.mark = ","), "\n")
```

```
Actual Cost (AC): $ 290,000
```

5.3.4. Performance Indicators

```
sv_val  <- sv(ev_val, pv_val)
cv_val  <- cv(ev_val, ac_val)
spi_val <- spi(ev_val, pv_val)
cpi_val <- cpi(ev_val, ac_val)

cat("Schedule Variance (SV):           $", format(sv_val, big.mark = ","), "\n")
```

```
Schedule Variance (SV):           $ -50,000
```

```
cat("Cost Variance (CV):               $", format(cv_val, big.mark = ","), "\n")
```

```
Cost Variance (CV):               $ -90,000
```

```
cat("Schedule Performance Index (SPI):", round(spi_val, 3), "\n")
```

```
Schedule Performance Index (SPI): 0.8
```

```r
cat("Cost Performance Index (CPI):    ", round(cpi_val, 3), "\n")
```

```
Cost Performance Index (CPI):     0.69
```

Interpretation: SPI < 1 means we are behind schedule, earning only 80 cents of planned value per dollar of schedule. CPI < 1 means we are over budget, earning only 69 cents of value per dollar spent. Both are below 1.0: this project is in trouble.

5.4. Forecasting: Estimate at Completion (EAC)

EAC forecasts the total cost at project completion. Three methods are available, each making a different assumption about future performance:

Method	Formula	When to use
Typical	BAC / CPI	Current cost inefficiency is expected to continue
Atypical	AC + (BAC − EV)	Cost overrun was a one-time event; future work at planned rate
Combined	AC + (BAC − EV) / (CPI × SPI)	Both cost and schedule performance will influence future costs

```r
eac_typical  <- eac(bac, method = "typical", cpi = cpi_val)
eac_atypical <- eac(bac, method = "atypical", ac = ac_val, ev = ev_val)
eac_combined <- eac(bac,
  method = "combined", cpi = cpi_val, ac = ac_val,
  ev = ev_val, spi = spi_val
)

cat("EAC (typical):  $", format(round(eac_typical),  big.mark = ","), "\n")
```

```
EAC (typical):  $ 725,000
```

```r
cat("EAC (atypical): $", format(round(eac_atypical), big.mark = ","), "\n")
```

```
EAC (atypical): $ 590,000
```

```r
cat("EAC (combined): $", format(round(eac_combined), big.mark = ","), "\n")
```

```
EAC (combined): $ 833,750
```

The typical method gives the most conservative (highest cost) estimate because it assumes the current CPI persists. The atypical method is the most optimistic.

5.4.1. EAC Comparison Table

```
eac_table <- data.frame(
  Method     = c("Typical", "Atypical", "Combined"),
  EAC        = c(round(eac_typical), round(eac_atypical), round(eac_combined)),
  Overrun    = c(
    round(eac_typical  - bac),
    round(eac_atypical - bac),
    round(eac_combined - bac)
  ),
  Assumption = c(
    "Current CPI continues",
    "Future work at planned rate",
    "CPI and SPI both factor in"
  )
)
knitr::kable(eac_table,
  format.args = list(big.mark = ","),
  caption = "EAC Comparison by Method"
)
```

Table 5.1.: EAC Comparison by Method

Method	EAC	Overrun	Assumption
Typical	725,000	225,000	Current CPI continues
Atypical	590,000	90,000	Future work at planned rate
Combined	833,750	333,750	CPI and SPI both factor in

5.5. Additional Metrics

```
etc_val  <- etc(bac, ev_val, cpi_val)
vac_val  <- vac(bac, eac_typical)
tcpi_bac <- tcpi(bac, ev_val, ac_val, target = "bac")
tcpi_eac <- tcpi(bac, ev_val, ac_val, target = "eac", eac = eac_typical)

cat("Estimate to Complete (ETC):  $", format(round(etc_val), big.mark = ","), "\n")
```

```
Estimate to Complete (ETC):  $ 435,000
```

```
cat("Variance at Completion (VAC): $", format(round(vac_val), big.mark = ","), "\n")
```

```
Variance at Completion (VAC): $ -225,000
```

```
cat("TCPI (to meet BAC):          ", round(tcpi_bac, 3), "\n")
```

```
TCPI (to meet BAC):          1.429
```

```
cat("TCPI (to meet EAC):          ", round(tcpi_eac, 3), "\n")
```

```
TCPI (to meet EAC):          0.69
```

Interpretation: TCPI > 1 means the team must work more efficiently than they have been to meet the target. A TCPI above 1.2 is a widely-cited practitioner rule of thumb for "unrealistic" (not a formal PMBOK standard), a useful warning signal. If you need to be 30% more efficient for the rest of the project, it probably isn't going to happen. Using the EAC target gives a more realistic benchmark.

5.6. Performance Trend Chart

The chart below shows cumulative PV, AC, and EV over time, with reference lines for BAC and EAC.

```
time_periods <- c(1, 2, 3)
actual_pct   <- c(0.08, 0.22, 0.40)
p_costs      <- c(45000, 110000, 135000)

pv_vals <- sapply(time_periods, function(t) pv(bac, schedule, t))
ac_vals <- cumsum(p_costs)
ev_vals <- sapply(actual_pct, function(a) ev(bac, a))

trend_data <- data.frame(
  Period = time_periods,
  PV = pv_vals, AC = ac_vals, EV = ev_vals
)

p <- ggplot2::ggplot(trend_data, ggplot2::aes(x = Period)) +
  ggplot2::geom_line(ggplot2::aes(y = PV, color = "Planned Value (PV)"), linewidth = 1.2) +
  ggplot2::geom_line(ggplot2::aes(y = AC, color = "Actual Cost (AC)"), linewidth = 1.2) +
  ggplot2::geom_line(ggplot2::aes(y = EV, color = "Earned Value (EV)"), linewidth = 1.2) +
  ggplot2::geom_point(ggplot2::aes(y = PV, color = "Planned Value (PV)"), size = 3) +
  ggplot2::geom_point(ggplot2::aes(y = AC, color = "Actual Cost (AC)"), size = 3) +
  ggplot2::geom_point(ggplot2::aes(y = EV, color = "Earned Value (EV)"), size = 3) +
  ggplot2::geom_hline(yintercept = bac, linetype = "dashed", color = "black", linewidth = 0.8)
  ggplot2::geom_hline(yintercept = eac_typical, linetype = "dotted", color = "darkred", linewi
  ggplot2::annotate("text", x = 2.8, y = bac + 12000, label = "BAC = $500K", size = 3.5) +
  ggplot2::annotate("text", x = 2.8, y = eac_typical + 12000, label = "EAC (typical)", size = 3
  ggplot2::scale_color_manual(values = c(
```

```
    "Planned Value (PV)" = "steelblue",
    "Actual Cost (AC)"   = "tomato",
    "Earned Value (EV)"  = "forestgreen"
  )) +
  ggplot2::scale_y_continuous(labels = scales::label_dollar(scale = 1e-3, suffix = "K")) +
  ggplot2::labs(
    title = "Earned Value Management: Performance Trend",
    x     = "Time Period", y = "Value", color = NULL
  ) +
  ggplot2::theme_minimal() +
  ggplot2::theme(legend.position = "bottom")

print(p)
```

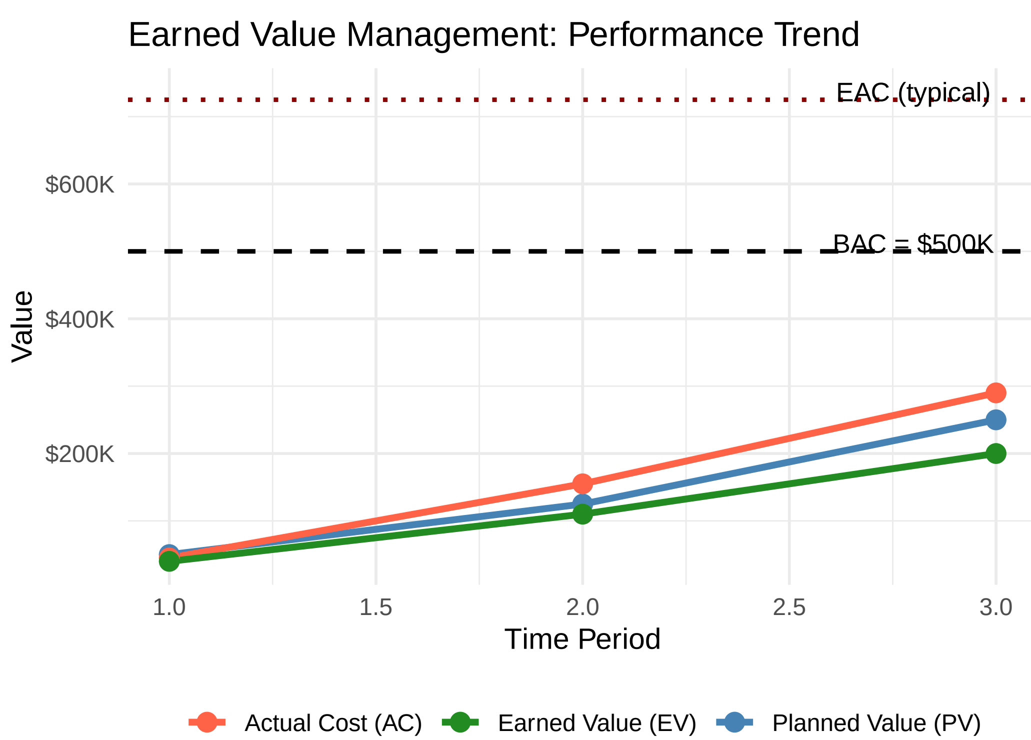

Figure 5.1.: EVM trend chart. When EV falls below PV, you're behind schedule. When AC rises above EV, you're over budget. Both conditions here, a project in the red on both dimensions.

Reading the chart: The gap between PV and EV shows the schedule gap, with EV below PV. The gap between AC and EV shows the cost overrun, as we've spent more than the value we've earned.

5.7. Summary

Key Takeaways

- EVM combines scope, schedule, and cost into three numbers (PV, EV, AC) from which all performance metrics flow.
- **CPI < 1** means over-budget; **SPI < 1** means behind schedule. Both below 1 signals a project in serious trouble.
- The **Typical EAC** (BAC / CPI) is the most conservative forecast; use it when current inefficiency is expected to persist.
- **TCPI > 1.2** is a common practitioner rule of thumb for "unrealistic" (not a formal standard), flag it as a signal to revise the budget rather than chase an unachievable efficiency target.
- The trend chart (PV, EV, AC over time) is your single most powerful communication tool for project status.

EVM tells you *where the project stands right now*. For forward-looking updates that incorporate new information as the project evolves, Bayesian methods in Chapter 6 provide a complementary approach to cost forecasting.

5.8. Exercises

1. **Diagnosis.** If SPI < 1 and CPI > 1, what does that tell you about the project? Describe the scenario in plain English and give an example of how this might happen on a real project.

2. **Forecast comparison.** Compute EAC using all three methods for a new project with BAC = $200K, PV = $80K, EV = $60K, AC = $70K. Which method produces the highest estimate? The lowest? Which would you recommend reporting to the client?

3. **TCPI reality check.** For the same project (BAC = $200K, PV = $80K, EV = $60K, AC = $70K), calculate TCPI to meet BAC. Is it realistic? At what TCPI would you advise the client to revise the budget rather than try to recover?

4. **The optimistic mistake.** A project manager always uses the atypical EAC method, regardless of what's happening. In what circumstances would this be dangerously wrong? Write a 2–3 sentence warning label for the atypical method.

5. **Build your own trend.** Create a 5-period EVM dataset where the project starts over budget but recovers over time (CPI < 1 in periods 1–2, CPI 1 in periods 3–5). Plot the trend chart. Does the "typical" EAC improve as the project recovers?

6. I Had a Feeling: Bayesian Risk Inference

> *"When the facts change, I change my mind. What do you do, sir?"* — Attributed to John Maynard Keynes

Here is the situation: you have a risk assessment based on historical data and expert judgment. Then something happens on site: a root cause you were watching turns out to be present, or a supplier sends a warning signal. Your prior assessment is now out of date. Do you stick with the original numbers, or update them?

Bayesian inference gives you the formal machinery to update. Not because it feels right, but because it is the mathematically correct way to incorporate new evidence.

> Learning Objectives
>
> By the end of this chapter, you will be able to:
>
> 1. Explain prior and posterior probability in plain English
> 2. Use `risk_prob()` to compute a prior risk probability from root causes
> 3. Use `risk_post_prob()` to update that probability given new observations
> 4. Sample prior and posterior cost distributions with `cost_pdf()` and `cost_post_pdf()`
> 5. Interpret the shift between prior and posterior distributions

6.1. The Core Idea

> Bayes' Theorem in Plain English
>
> $$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$
>
> **In plain English:** Your updated belief (posterior) equals your initial belief (prior) multiplied by how well the evidence fits that belief, normalized to sum to 1.
> In project risk terms:
>
> - **Prior** $P(H)$: Your initial estimate of risk probability, before any field observations
> - **Likelihood** $P(E \mid H)$: How likely is the observed evidence if the risk/cause is present?
> - **Posterior** $P(H \mid E)$: Your updated estimate after incorporating the evidence
>
> Observing a root cause *always* moves the posterior. The direction and magnitude depend on how strongly that cause is linked to the risk event.

Bayes' theorem states:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

In English: the probability of your hypothesis (H) given new evidence (E) equals the likelihood of seeing that evidence if the hypothesis were true, times your prior probability, divided by the total probability of the evidence. The result is the **posterior** probability, your updated belief.

In project risk terms:

- **Prior**: Your initial estimate of risk probability before any field observations
- **Evidence**: Observations about root causes (is the labor market tight? is weather bad?)
- **Posterior**: Your updated estimate after incorporating those observations

The PRA package provides four Bayesian functions organized into two stages:

Stage	Function	Purpose
Prior	`risk_prob()`	Compute risk probability from root causes (no observations yet)
Prior	`cost_pdf()`	Sample prior cost distribution based on risk probabilities
Posterior	`risk_post_prob()`	Update risk probability after observing cause status
Posterior	`cost_post_pdf()`	Sample posterior cost distribution based on observed risks

```r
library(PRA)
set.seed(42)
```

6.2. Causal Structure

Before computing probabilities, it helps to see the causal graph. Two independent root causes each point to the risk event, and observing either one updates the posterior risk probability.

```r
library(igraph)
g <- graph_from_data_frame(
  data.frame(from = c("Cause-1", "Cause-2"), to = c("Risk", "Risk")),
  directed = TRUE
)
V(g)$color <- c("steelblue", "steelblue", "tomato")
plot(g, vertex.size = 40, vertex.label.cex = 0.9,
     edge.arrow.size = 0.6, layout = layout_with_sugiyama(g)$layout)
```

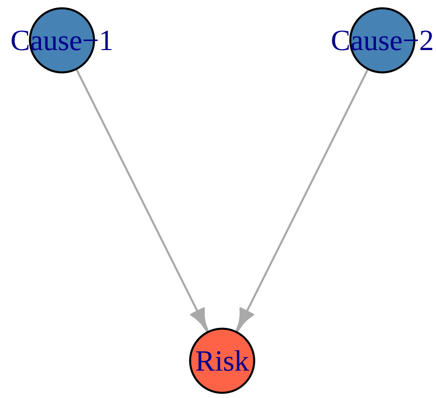

Figure 6.1.: DAG: Cause-1 and Cause-2 independently drive the risk event R.

6.3. Step 1: Prior Risk Probability

Before any observations are made, `risk_prob()` computes the probability of a risk event R occurring given two potential root causes. For each cause, we supply:

- `cause_probs`: prior probability that the cause is present
- `risks_given_causes`: P(R | cause present)
- `risks_given_not_causes`: P(R | cause absent)

```
cause_probs            <- c(0.3, 0.2)
risks_given_causes     <- c(0.8, 0.6)
risks_given_not_causes <- c(0.2, 0.4)
```

```
prior_risk <- risk_prob(cause_probs, risks_given_causes, risks_given_not_causes)
cat("Prior probability of risk event R:", round(prior_risk, 3), "\n")
```

```
Prior probability of risk event R: 0.82
```

This is your starting point, an estimate of how likely the risk event is before you've looked at anything on site.

6.4. Step 2: Prior Cost Distribution

Given the prior risk probabilities, `cost_pdf()` samples the cost distribution before any field observations. Three independent risk events can each contribute cost if they occur.

```r
risk_probs        <- c(0.3, 0.5, 0.2)
means_given_risks <- c(10000, 15000, 5000)
sds_given_risks   <- c(2000, 1000, 1000)
base_cost         <- 2000
```

```r
prior_samples <- cost_pdf(
  num_sims          = 5000,
  risk_probs        = risk_probs,
  means_given_risks = means_given_risks,
  sds_given_risks   = sds_given_risks,
  base_cost         = base_cost
)
```

We'll compare this prior distribution to the posterior in Step 4.

6.5. Step 3: Posterior Risk Probability (Bayesian Update)

After inspecting the project site, we observe that Cause 1 is present (= 1). Cause 2 has not yet been assessed (= NA). `risk_post_prob()` updates the risk probability using only the available evidence; NA causes are treated as unobserved and do not contribute to the update.

```r
observed_causes <- c(1, NA)  # C1 observed as present; C2 not yet assessed
```

```r
posterior_risk <- risk_post_prob(
  cause_probs, risks_given_causes,
  risks_given_not_causes, observed_causes
)
cat("Posterior probability of risk event R:", round(posterior_risk, 3), "\n")
```

```
Posterior probability of risk event R: 0.632
```

Observing Cause 1 (which has a strong link to R) raises the risk probability substantially. The NA for Cause 2 is simply ignored; only confirmed observations drive the update.

6.5.1. Prior vs. Posterior Probability

```r
prob_data <- data.frame(
  Stage       = c("Prior", "Posterior"),
  Probability = c(prior_risk, posterior_risk)
)

p <- ggplot2::ggplot(prob_data, ggplot2::aes(x = Stage, y = Probability, fill = Stage)) +
  ggplot2::geom_col(width = 0.5, show.legend = FALSE) +
  ggplot2::geom_text(ggplot2::aes(label = round(Probability, 3)),
    vjust = -0.4, size = 4.5
  ) +
  ggplot2::scale_fill_manual(values = c("Prior" = "steelblue", "Posterior" = "tomato")) +
  ggplot2::scale_y_continuous(limits = c(0, 1), labels = scales::percent) +
  ggplot2::labs(
    title = "Bayesian Update: Risk Probability",
    x     = NULL, y = "P(Risk Event R)"
  ) +
  ggplot2::theme_minimal(base_size = 13)

print(p)
```

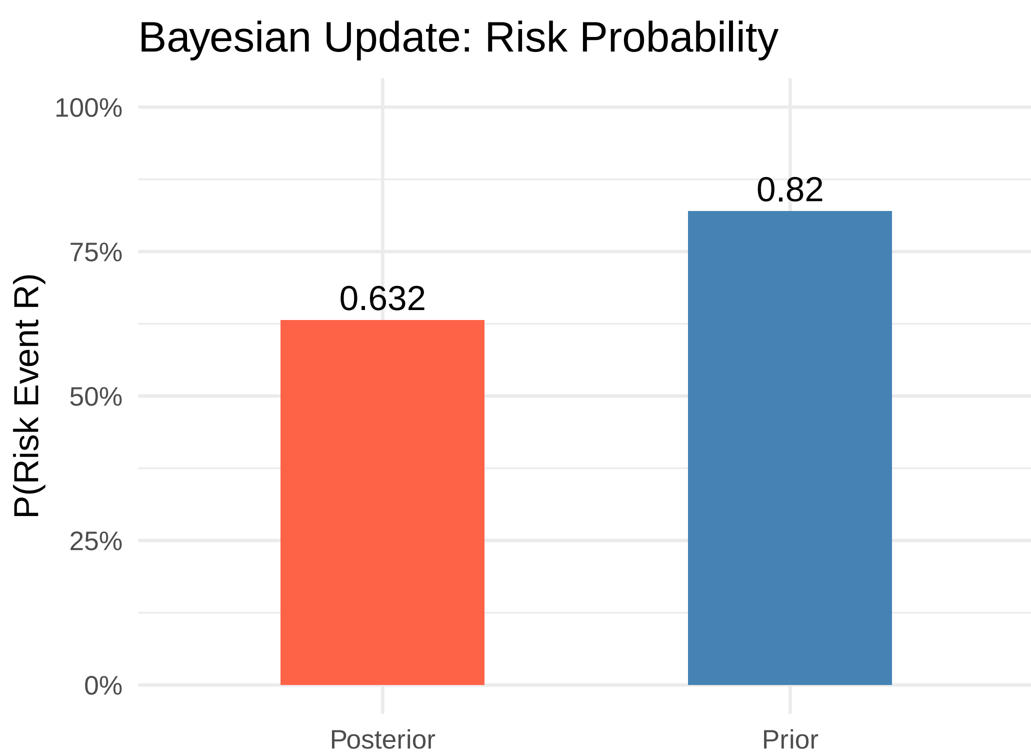

Figure 6.2.: Bayesian update: observing Cause 1 nearly doubles the estimated probability of the risk event.

6.6. Step 4: Posterior Cost Distribution

Now that we know Cause 1 is present (Risk 1 occurs), and one risk remains unobserved (Risk 2 = NA), `cost_post_pdf()` samples the posterior cost distribution. Observed risks that occurred (= 1) add their cost; unobserved risks (= NA) are excluded from the simulation.

```
observed_risks <- c(1, NA, 1)  # Risk 1 and 3 confirmed; Risk 2 not yet assessed
```

```
posterior_samples <- cost_post_pdf(
  num_sims          = 5000,
  observed_risks    = observed_risks,
  means_given_risks = means_given_risks,
  sds_given_risks   = sds_given_risks,
  base_cost         = base_cost
)
```

6.6.1. Prior vs. Posterior Cost Distribution

Plotting both distributions on the same axes shows how the evidence shifts the cost estimate:

```
xlim_range <- range(c(prior_samples, posterior_samples))

hist(prior_samples,
  breaks = 40, freq = FALSE,
  col    = rgb(0.27, 0.51, 0.71, 0.5),
  border = "white",
  xlim   = xlim_range,
  main   = "Prior vs. Posterior Cost Distribution",
  xlab   = "Total Cost ($)", ylab = "Density"
)

hist(posterior_samples,
  breaks = 40, freq = FALSE,
  col    = rgb(0.84, 0.24, 0.31, 0.5),
  border = "white",
  add    = TRUE
)

abline(v = mean(prior_samples),     col = "steelblue", lty = 2, lwd = 2)
abline(v = mean(posterior_samples), col = "tomato",    lty = 2, lwd = 2)

legend("topright",
  legend = c(
    paste0("Prior  (mean = $", format(round(mean(prior_samples)),     big.mark = ",")),
    paste0("Posterior (mean = $", format(round(mean(posterior_samples)), big.mark = ","))
  ),
```

```
  fill = c(rgb(0.27, 0.51, 0.71, 0.5), rgb(0.84, 0.24, 0.31, 0.5)),
  bty  = "n"
)
```

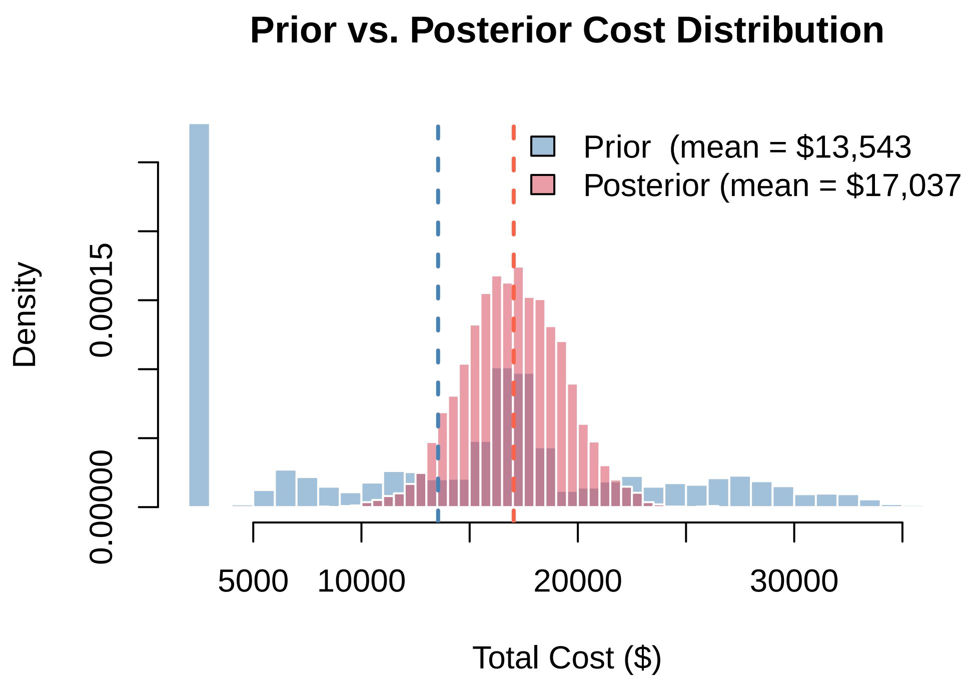

Figure 6.3.: Prior vs. posterior cost distribution. Confirming which risks materialized narrows and shifts the distribution.

Interpretation: The posterior distribution is narrower and shifted upward, as observing which risks materialized eliminates uncertainty about some cost components while raising the estimated exposure from confirmed events. The remaining spread reflects uncertainty from the unobserved risk and cost variability in the confirmed ones.

6.7. Summary

The Bayesian workflow in PRA follows a natural before-and-after structure:

1. **Before observations**: use `risk_prob()` and `cost_pdf()` to characterize the risk landscape.
2. **As evidence arrives**: use `risk_post_prob()` and `cost_post_pdf()` to update estimates.
3. **NA values** represent causes/risks not yet assessed; they are correctly excluded from the update.

This approach is particularly powerful in phased projects where information about risk drivers becomes available progressively, allowing cost forecasts to be refined at each stage.

6.8. Summary

Key Takeaways

- Bayesian inference is the principled way to update risk estimates as new evidence arrives; it is not a heuristic, it is the mathematically correct rule.
- **Prior → Posterior**: the posterior always moves in the direction of the evidence; the size of the move depends on how strongly the cause links to the risk.
- `NA` values for unobserved causes or risks are correctly handled; they do not contribute to the update, reflecting genuine ignorance.
- The posterior cost distribution is typically narrower than the prior, because confirmed risk states eliminate uncertainty about some cost components.
- This workflow is particularly powerful in **phased projects**: update at each phase gate as more information becomes available.

For risks that propagate through a network of shared resources and tasks, Chapter 9 extends these ideas into a full probabilistic graph where Bayesian conditioning and structural dependencies interact.

6.9. Exercises

1. **Prior vs. posterior.** In plain English, what is the difference between a prior probability and a posterior probability? Give a project example where each would be used.
2. **Change the observation.** Re-run Step 3 with Cause 2 observed as present (= 1) instead of Cause 1. How does the posterior change compared to when Cause 1 was observed? Why?
3. **Both causes observed.** What happens if you observe both causes as present (= 1, = 1)? Compute the posterior. What is the maximum possible posterior probability given the model parameters?
4. **Intuition check.** Explain in plain English why observing one root cause changes your belief about the risk event, even though we didn't observe the risk event itself. What is the role of the conditional probabilities `risks_given_causes` in this reasoning?
5. **Sequential updating.** Start with the prior. Observe Cause 1. Update. Then observe Cause 2. Update again. Does it matter what order you observe the causes? Should it? (Hint: think about whether the causes are assumed independent in this model.)

7. S Is for Success: Sigmoidal Learning Curves

> *"First it's slow, then it's fast, then you've already hit the ceiling."* — P.G.

Every team that's ever built something knows the pattern. Week one: figuring things out. Week four: hitting a stride. Week eight: slowing down as completion approaches 100%. This S-shaped curve, with a slow start, rapid middle, and graceful plateau, is not a quirk of one project. It's a near-universal pattern in learning and production processes, and it has a name: the **sigmoidal curve**.

If you can fit a sigmoidal model to your early progress data, you can forecast when you'll finish, and how confident you should be in that forecast.

> Learning Objectives
>
> By the end of this chapter, you will be able to:
>
> 1. Explain why sigmoidal models are more appropriate than linear or exponential models for project completion data
> 2. Fit Logistic, Pearl, and Gompertz models using `fit_sigmoidal()`
> 3. Compare model fit using residual standard error
> 4. Produce and interpret a forecast with confidence bands using `plot_sigmoidal()` and `predict_sigmoidal()`
> 5. Choose the right model type based on data shape and theoretical expectations

7.1. Why Sigmoidal?

Linear models ("we'll complete 10% per week forever") break down at the extremes: they predict negative completion before the project starts and more than 100% after it finishes. Exponential models grow forever. Sigmoidal models are bounded: they start slow, accelerate, and plateau at a ceiling value, making them physically meaningful for completion percentages.

Learning curves are used to:

- **Forecast completion**: Predict when a task or deliverable will reach 100% based on past progress.
- **Identify acceleration**: Detect when a project is in its rapid-improvement phase versus plateauing.
- **Set realistic milestones**: Calibrate schedule targets against demonstrated learning rates.

7.2. The Three Models

PRA provides three sigmoidal model types:

Model	Formula	Parameters
Logistic	$K/(1+e^{-r(t-t_0)})$	K = ceiling; r = growth rate; t = inflection time
Pearl	$K/(1+e^{-r(t-t_0)})$	Same functional form as Logistic
Gompertz	$A \cdot e^{-b \cdot e^{-ct}}$	A = ceiling; c = growth rate; b = initial suppression

The Logistic and Pearl models are mathematically identical but fitted differently. The Gompertz has a different shape: its inflection point occurs earlier and the curve is asymmetric, making it better suited for processes where acceleration comes quickly and the plateau is long.

7.3. Example: Fitting a Logistic Model

```
library(PRA)
```

We have weekly completion percentage data for a construction deliverable over 9 weeks.

```
data <- data.frame(
  time       = 1:9,
  completion = c(5, 15, 40, 60, 70, 75, 80, 85, 90)
)
```

7.3.1. Fit the Model

```
fit <- fit_sigmoidal(data, "time", "completion", "logistic")
```

7.3.2. Assess Fit Quality

Use `summary()` to examine the fitted coefficients, their standard errors, and the residual standard error, a measure of how closely the model matches the observed data.

```
summary(fit)
```

```
Formula: y ~ logistic(x, K, r, t0)

Parameters:
   Estimate Std. Error t value Pr(>|t|)
K   84.3515     2.5142  33.550 4.67e-08 ***
r    1.0356     0.1380   7.505 0.000289 ***
t0   3.2520     0.1459  22.284 5.34e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.148 on 6 degrees of freedom

Number of iterations to convergence: 10
Achieved convergence tolerance: 1.49e-08
```

A small residual standard error (relative to the response scale) indicates a good fit. Coefficient t values with $|t| > 2$ are statistically meaningful.

7.3.3. Plot with Confidence Bands

`plot_sigmoidal()` plots the data, fitted curve, and optional confidence bounds in a single call.

```
plot_sigmoidal(
  fit, data, "time", "completion", "logistic",
  conf_level = 0.95,
  main       = "Logistic Learning Curve: Completion Forecast",
  xlab       = "Week",
  ylab       = "Completion (%)"
)
```

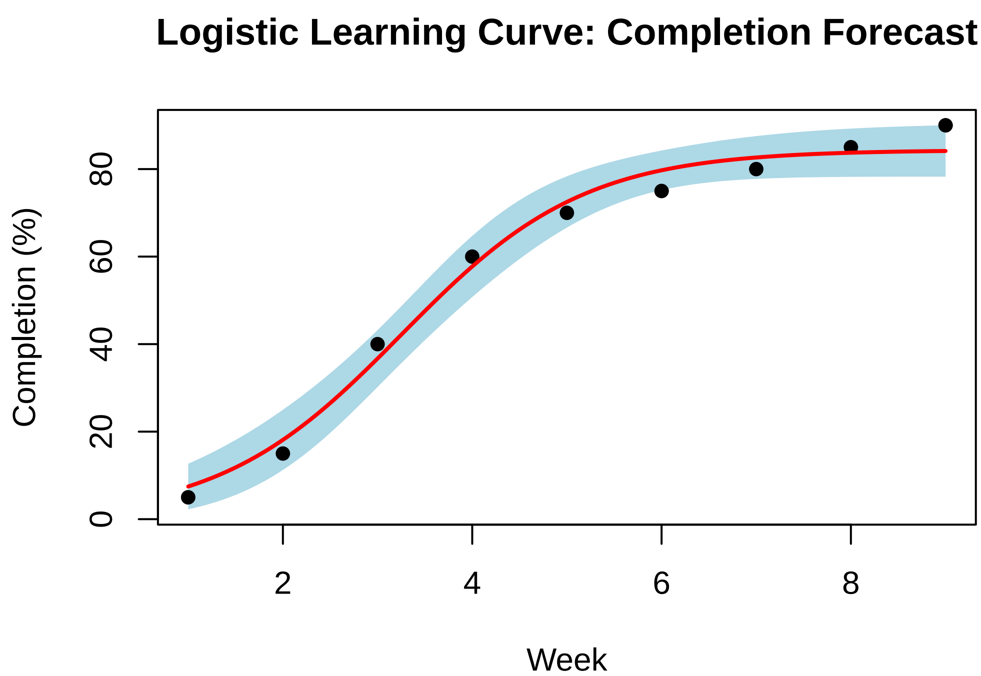

Figure 7.1.: Logistic learning curve fit with 95% confidence band. Wider bands at the tails reflect greater extrapolation uncertainty.

The shaded region is the 95% confidence band, the range within which the true curve is likely to lie. Note how the band widens as we extrapolate past the observed data range.

7.3.4. Predict Future Completion

Use `predict_sigmoidal()` to generate numeric forecasts, including confidence bounds.

```
future_times <- seq(1, 12, length.out = 100)
predictions  <- predict_sigmoidal(fit, future_times, "logistic", conf_level = 0.95)

knitr::kable(
  tail(round(predictions, 1), 5),
  caption  = "Predicted completion (final 5 forecast points)",
  row.names = FALSE
)
```

Table 7.1.: Predicted completion (final 5 forecast points)

x	pred	lwr	upr
11.6	84.3	78.2	90.5

x	pred	lwr	upr
11.7	84.3	78.2	90.5
11.8	84.3	78.2	90.5
11.9	84.3	78.2	90.5
12.0	84.3	78.2	90.5

7.4. Comparing All Three Model Types

It is good practice to fit multiple models and compare their goodness-of-fit. Different shapes may fit your data better.

```
fit_logistic <- fit_sigmoidal(data, "time", "completion", "logistic")
fit_pearl    <- fit_sigmoidal(data, "time", "completion", "pearl")
fit_gompertz <- fit_sigmoidal(data, "time", "completion", "gompertz")
```

```
rse <- function(fit) summary(fit)$sigma

comparison <- data.frame(
  Model             = c("Logistic", "Pearl", "Gompertz"),
  Residual_StdError = round(c(rse(fit_logistic), rse(fit_pearl), rse(fit_gompertz)), 3)
)
knitr::kable(comparison, caption = "Model Fit Comparison (lower RSE = better fit)")
```

Table 7.2.: Model Fit Comparison (lower RSE = better fit)

Model	Residual_StdError
Logistic	4.148
Pearl	4.148
Gompertz	2.887

Now plot all three fits side by side:

```
x_seq <- seq(1, 12, length.out = 200)

pred_log <- predict_sigmoidal(fit_logistic, x_seq, "logistic")
pred_prl <- predict_sigmoidal(fit_pearl,    x_seq, "pearl")
pred_gom <- predict_sigmoidal(fit_gompertz, x_seq, "gompertz")

plot(data$time, data$completion,
  pch = 16, xlim = c(1, 12), ylim = c(0, 105),
  main = "Learning Curve: Model Comparison",
  xlab = "Week", ylab = "Completion (%)"
)
```

```r
lines(pred_log$x, pred_log$pred, col = "steelblue", lwd = 2)
lines(pred_prl$x, pred_prl$pred, col = "tomato",    lwd = 2, lty = 2)
lines(pred_gom$x, pred_gom$pred, col = "darkgreen", lwd = 2, lty = 3)
legend("bottomright",
  legend = c("Logistic", "Pearl", "Gompertz"),
  col    = c("steelblue", "tomato", "darkgreen"),
  lty    = c(1, 2, 3), lwd = 2, bty = "n"
)
```

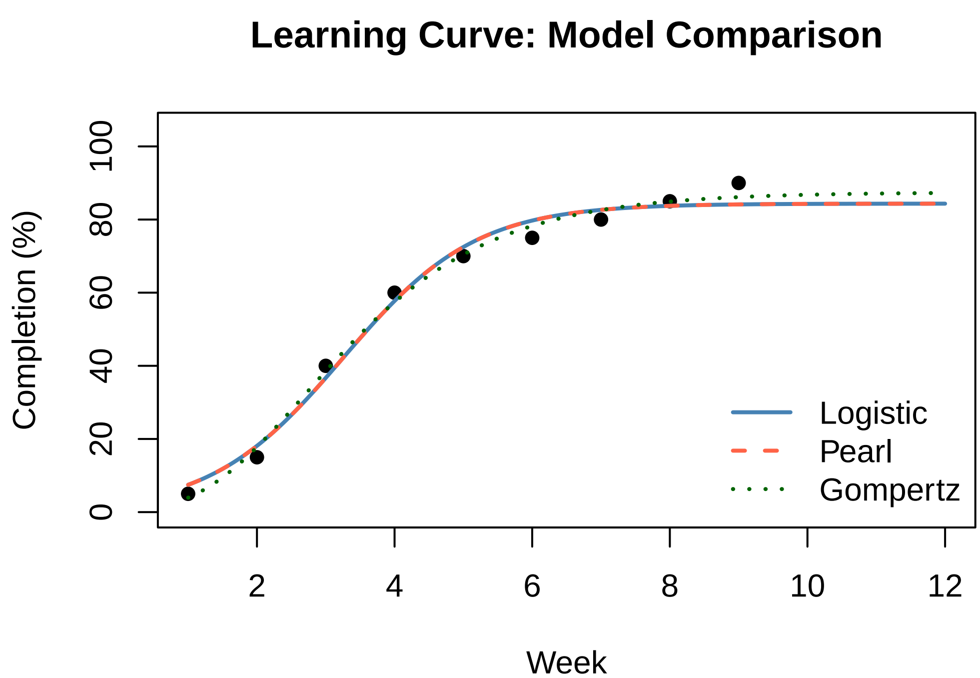

Figure 7.2.: Comparison of Logistic, Pearl, and Gompertz fits. Logistic and Pearl are identical; Gompertz has a different early-acceleration shape.

⚠ Logistic and Pearl Are the Same Model

The Logistic and Pearl models share the same mathematical formula and will produce **identical fits** on the same data. The distinction is historical, not functional. When comparing model fits, the Pearl and Logistic RSEs will always be equal; this is expected behavior, not a bug. Gompertz is genuinely different: its inflection point occurs earlier and the curve is asymmetric, making it better suited for processes that accelerate quickly before plateauing slowly.

7.5. Summary

The sigmoidal workflow in PRA:

1. `fit_sigmoidal()`: fit a model to observed time-completion data
2. `summary(fit)`: inspect coefficient estimates and goodness-of-fit
3. `predict_sigmoidal()`: generate numeric forecasts with optional confidence bounds
4. `plot_sigmoidal()`: visualize the fit and confidence band

Choose the model type based on the shape of your data and theoretical expectations about the learning process.

7.6. Summary

Key Takeaways

- Sigmoidal models capture the three universal phases of learning: slow start, rapid improvement, and plateau, making them physically meaningful for completion forecasting.
- **Logistic and Pearl are identical**: if they give different RSEs, something has gone wrong with the fit. Use Logistic unless you have a specific reason to use Pearl.
- **Gompertz** accelerates earlier and plateaus more gradually; choose it when you have evidence of rapid early-stage learning.
- **Wider confidence bands** at later time points reflect extrapolation uncertainty; the further you predict beyond the observed data, the less trustworthy the forecast.
- Always check both the fitted curve visually *and* the RSE numerically, as a low RSE with a nonsensical curve shape means overfitting.

7.7. Exercises

1. **Model selection.** Which model (Logistic, Pearl, Gompertz) would be most appropriate when early progress is slow but then the team suddenly accelerates around week 3? What parameter would you adjust to shift the inflection point?

2. **Fit comparison.** Using the construction data from this chapter, fit all three models and compare their residual standard errors. Which model fits best? Does the best-fitting model always give the most sensible forecast? Plot the predictions out to week 15 and compare the three extrapolations.

3. **Confidence band width.** Predict percent complete at week 12 using the logistic model. How wide is the 95% confidence band (upper − lower)? At week 6? At week 3? What drives the increasing uncertainty as you extrapolate further from the observed data?

4. **Your own data.** Collect or invent 8 weeks of progress data for a project activity. Fit a logistic model. Does the model suggest you'll reach 90% completion by week 12? By week 16? What does the confidence interval tell you about the reliability of that forecast?

5. **Beyond completion.** Learning curves can model more than completion percentages; they can model cost efficiency, defect rates, or productivity. Choose one of these alternative interpretations and describe how you would set up the `data` frame, what "ceiling" K means in that context, and what a fitted model would tell you.

8. Everything Is Connected: Design Structure Matrices

"You can't just fix Task 7. Task 7 shares three resources with Tasks 2, 4, and 9." — P.G.

Projects are not a list of independent activities. They are a web of interdependencies, and understanding that web is the key to understanding where delays and cost overruns actually come from. A delay in one task may ripple into five others, not because the tasks are sequentially linked, but because they share the same crew, the same materials, or the same equipment.

The Design Structure Matrix (DSM) makes these hidden connections visible and quantifiable (Browning 2001; Steward 1981).

Learning Objectives

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

1. Construct a Resource-Task matrix **S** and a Risk-Resource matrix **R**
2. Compute the Parent DSM (resource-based coupling) using `parent_dsm()`
3. Compute the Grandparent DSM (risk-based coupling) using `grandparent_dsm()`
4. Interpret diagonal vs. off-diagonal DSM entries
5. Use DSM coupling values to prioritize risk mitigation decisions

8.1. What Is a DSM?

A DSM is a square matrix that maps dependencies between tasks in a project. In PRA, DSMs are derived from bipartite relationships: resources are shared across tasks, and risks are shared across resources. These shared dependencies create coupling between tasks that can propagate delays, cost overruns, or failures.

PRA provides two DSM functions:

- **`parent_dsm()`**: the Resource-based "Parent" DSM: shows how many resources are shared between each pair of tasks.
- **`grandparent_dsm()`**: the Risk-based "Grandparent" DSM: shows how many risks are shared between task pairs (via the resource layer).

The key idea from the resource-based view of project management (Govan and Damnjanovic 2016): shared resources are the structural pathway through which risks propagate. If you want to understand why two tasks always seem to get delayed together, look at what resources they share.

8.2. The Resource-Task Matrix

The starting point is the Resource-Task Matrix **S**, where rows represent resources and columns represent tasks. An entry `S[i, j] = 1` means resource i is used by task j.

```
library(PRA)
```

```
# 4 resources x 5 tasks
S <- matrix(c(
  1, 0, 1, 0,
  1, 1, 0, 0,
  0, 1, 0, 1,
  0, 0, 1, 1,
  0, 1, 1, 0
), nrow = 4, ncol = 5)
rownames(S) <- paste0("R", 1:4)
colnames(S) <- paste0("T", 1:5)
S
```

```
   T1 T2 T3 T4 T5
R1  1  1  0  0  0
R2  0  1  1  0  1
R3  1  0  0  1  1
R4  0  0  1  1  0
```

8.3. Parent DSM

The Parent DSM $\mathbf{P} = \mathrm{t}(\mathbf{S})$ %*% $\mathbf{S}$ is a tasks-by-tasks matrix. The diagonal entry `P[j, j]` counts how many resources task j uses. The off-diagonal entry `P[j, k]` counts how many resources tasks j and k share, a measure of coupling.

```
p <- parent_dsm(S)
print(p)
```

```
Resource-based 'Parent' Design Structure Matrix
Tasks: 5  Resources: 4

   T1 T2 T3 T4 T5
T1  2  1  0  1  1
T2  1  2  1  0  1
T3  0  1  2  1  1
T4  1  0  1  2  1
T5  1  1  1  1  2
```

```
plot(p)
```

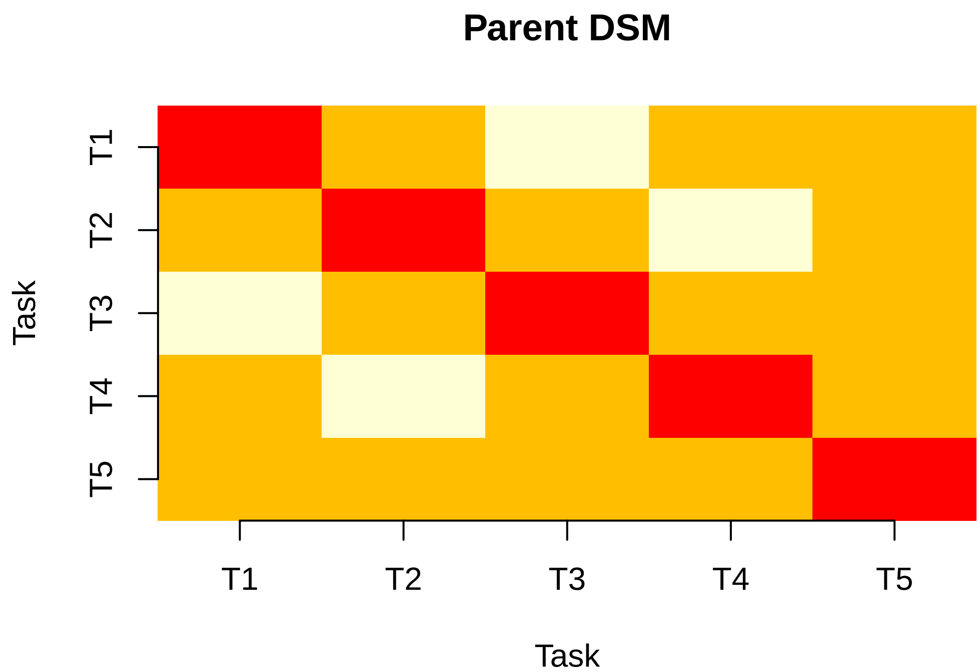

Figure 8.1.: Parent DSM heatmap. Darker cells indicate more shared resources between tasks, and therefore more potential for correlated disruption.

Tasks that share more resources are more tightly coupled. If a resource is delayed or constrained, all tasks that depend on it are affected simultaneously.

8.4. The Risk-Resource Matrix

The Risk-Resource Matrix **R** adds a second layer. Rows represent risks and columns represent resources. An entry `R[i, j] = 1` means risk i affects resource j.

```
# 3 risks x 4 resources
R <- matrix(c(
  1, 0, 1,
  1, 1, 0,
  0, 1, 0,
  0, 0, 1
), nrow = 3, ncol = 4)
rownames(R) <- paste0("Risk", 1:3)
colnames(R) <- paste0("R", 1:4)
R
```

```
      R1 R2 R3 R4
Risk1  1  1  0  0
Risk2  0  1  1  0
Risk3  1  0  0  1
```

8.5. Grandparent DSM

The Grandparent DSM traces the dependency chain from risks through resources to tasks. The intermediate matrix **T** = **R** %% ***S*** *gives a risks-by-tasks mapping, and the Grandparent DSM is* ***G*** *= t(**T**) %%* **T**. Off-diagonal entries count how many risks are shared between each pair of tasks.

```r
g <- grandparent_dsm(S, R)
print(g)
```

```
Risk-based 'Grandparent' Design Structure Matrix
Tasks: 5  Resources: 4  Risks: 3

   T1 T2 T3 T4 T5
T1  3  4  3  2  3
T2  4  6  4  2  4
T3  3  4  3  2  3
T4  2  2  2  2  2
T5  3  4  3  2  5
```

```r
plot(g)
```

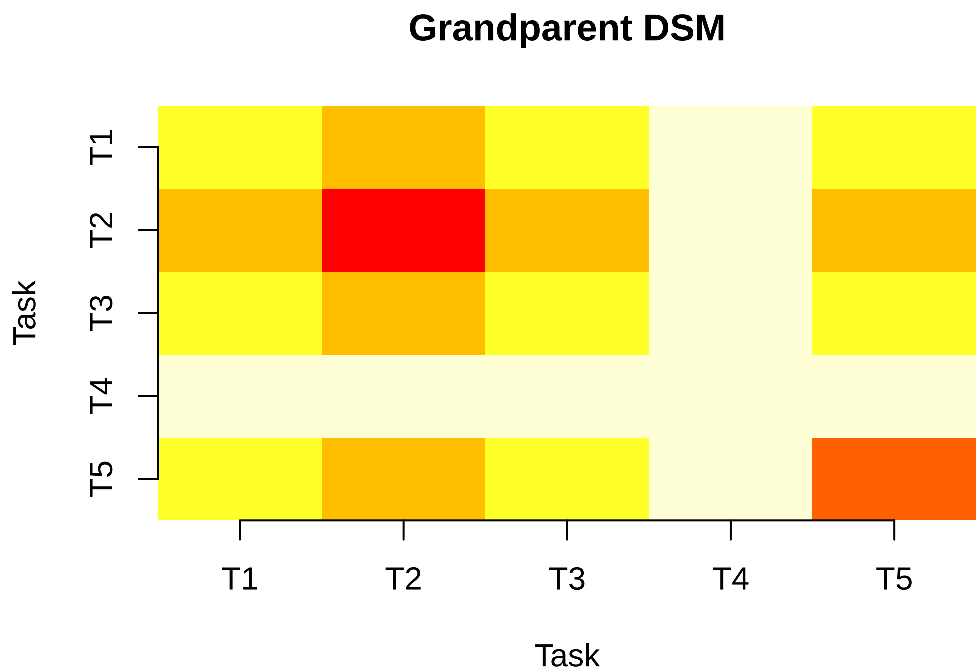

Figure 8.2.: Grandparent DSM heatmap. Shows risk-based coupling: tasks that share risk exposure through common resources tend to fail together.

8.6. Interpreting the DSM

- **Diagonal values** indicate the total number of resources (Parent) or risks (Grandparent) associated with each task. Higher values mean a task has more dependencies.
- **Off-diagonal values** indicate coupling between task pairs. Higher values mean more shared dependencies and greater potential for correlated disruption.
- **Symmetric structure**: both DSMs are always symmetric since shared dependencies are bidirectional: if task A shares a resource with task B, then task B shares that resource with task A.

A high off-diagonal value between tasks T2 and T5, for example, means those tasks share multiple resources. When risk strikes any of those resources, both tasks are hit simultaneously. This is not a coincidence; it is structural.

8.7. From DSM to Decision

Using the DSM to Prioritize Mitigation

1. **Find the highest off-diagonal values** in the Parent DSM, as these task pairs share the most resources and are most likely to fail together.
2. **Identify those shared resources**, the structural bottlenecks through which risk propagates.
3. **Inspect the Grandparent DSM** for those same pairs; if they also share risks, the coupling is doubly reinforced.
4. **Prioritize mitigation** on the shared resources with the highest combined Parent + Grandparent coupling: add contingency, build redundancy, or create buffers.

The Grandparent DSM goes one level deeper: it shows which pairs of tasks are exposed to the same risks. If you can eliminate or mitigate a risk, you simultaneously reduce coupling between all tasks that share that risk.

8.8. Summary

Key Takeaways

- The DSM makes **structural task dependencies** visible: not sequential dependencies, but resource-sharing dependencies that create correlated risk.
- **Parent DSM** (resource-based): `P = t(S) %*% S`. Off-diagonal `P[j,k]` = number of shared resources between tasks j and k.
- **Grandparent DSM** (risk-based): `G = t(T) %*% T` where `T = R %*% S`. Off-diagonal entries count shared risks via the resource layer.
- Both DSMs are always **symmetric**, since shared dependencies are bidirectional.
- High off-diagonal values point directly to the resources and risks that deserve the most mitigation investment.

The DSM quantifies *structural* coupling, but not the probabilistic dynamics of how risk propagates through it. For the full probabilistic treatment, including simulation and Bayesian updating over the same structure, see Chapter 9 and Chapter 10.

8.9. Exercises

1. **Reading the DSM.** In the Parent DSM computed above, which pair of tasks has the highest coupling (highest off-diagonal value)? What does this mean in practical terms? What would happen if the shared resource became unavailable?

2. **Extend the matrix.** Add a 5th resource (R5) to the Resource-Task matrix **S** such that it is used by tasks T1 and T5. Recompute the Parent DSM. Which entries change, and why?

3. **Risk propagation.** Look at the Grandparent DSM. Which task pair shares the most risks? Describe in plain English the risk scenario where this coupling causes the most damage.
4. **DSM symmetry.** Both the Parent and Grandparent DSMs are symmetric. Why? Is there a project structure where the DSM would not be symmetric? (Hint: think about directed dependencies; what if one task uses a resource but another task produces it?)
5. **From DSM to Monte Carlo.** The off-diagonal values in the Parent DSM can be interpreted as a basis for correlation. Design a correlation matrix for MCS (from Chapter 2) based on the Parent DSM: normalize the off-diagonal values so they range between 0 and 1, and use them as correlation coefficients. Run the MCS with this correlation structure and compare the total variance to the case with a zero correlation matrix. What did the structural coupling add to the risk picture?

9. It's a Small World After All: Probabilistic Networks

"Everything is connected to everything else." — Barry Commoner's First Law of Ecology

A risk register lists risks. A probability network shows you how those risks talk to each other, and to your project. It's the difference between knowing that "Technical Complexity" is a risk and understanding exactly how it flows through your developer costs into your total project budget, and what happens to that flow when you learn new information.

Bayesian networks are the tool. A Bayesian network is a specific type of probabilistic network, a directed acyclic graph (DAG) in which each node represents a random variable and each edge encodes a conditional probability relationship. The "probabilistic network" framing in this chapter's title is intentional: the core concepts (conditioning, propagation, graph structure) apply to the broader class, and `PRA` uses the Bayesian network formulation specifically. They combine graph theory with probability theory to model the full dependency structure of a project: which risks drive which resources, which resources drive which tasks, and how uncertainty propagates all the way up to the project total.

> Learning Objectives
>
> By the end of this chapter, you will be able to:
>
> 1. Build a probabilistic network using `prob_net()` with nodes, edges, and distributions
> 2. Run forward simulations with `prob_net_sim()` and interpret the cost distribution
> 3. Incorporate new evidence using `prob_net_learn()` and observe downstream shifts
> 4. Modify network structure and distributions with `prob_net_update()`
> 5. Explain the difference between learning (conditioning) and updating (graph surgery)

9.1. What Is a Bayesian Network?

> **i** Bayesian Network vs. DSM
>
> The Design Structure Matrix (Chapter 8) shows *structural* coupling, specifically how many resources two tasks share. A Bayesian network goes further: it encodes **probabilistic dependencies**, meaning the actual conditional distributions of costs given risk states. Where the DSM counts connections, the network simulates consequences.
> Use a DSM for rapid structural triage. Use a Bayesian network when you need actual cost distributions and the ability to condition on observed evidence.

A Bayesian network is a directed acyclic graph (DAG) where:

- **Nodes** represent random variables (risks, resources, tasks, project totals)
- **Edges** represent conditional dependencies (Risk A affects Resource C)
- **Distributions** encode the uncertainty at each node

Bayesian networks are well-suited to project risk analysis because they explicitly model how risk events propagate through resources and tasks to affect total project cost (Govan 2014).

For a more advanced example covering causal inference, graph surgery, and the see-versus-do distinction across a full project portfolio, see Chapter 10.

9.2. Project Setup

9.2.1. Tasks

Consider a small software development project with three tasks.

```
library(PRA)
set.seed(42)
```

```
tasks <- data.frame(
  ID    = c("F", "G", "H"),
  Label = c("Task-1", "Task-2", "Task-3"),
  Task  = c("Requirements and Design", "Development", "Testing and Handover")
)
knitr::kable(tasks, caption = "Project Tasks")
```

Table 9.1.: Project Tasks

ID	Label	Task
F	Task-1	Requirements and Design
G	Task-2	Development
H	Task-3	Testing and Handover

9.2.2. Resources

Each task draws on one primary resource. The table below shows the baseline cost estimate for each resource.

```
resources <- data.frame(
  ID       = c("C", "D", "E"),
  Label    = c("Resource-1", "Resource-2", "Resource-3"),
  Resource = c("Business Analyst", "Developer", "QA Engineer"),
  Task_ID  = c("F", "G", "H"),
```

```
  Mean     = c(15000, 50000, 20000),
  SD       = c(3000,  10000,  4000)
)
knitr::kable(resources, caption = "Project Resources")
```

Table 9.2.: Project Resources

ID	Label	Resource	Task_ID	Mean	SD
C	Resource-1	Business Analyst	F	15000	3000
D	Resource-2	Developer	G	50000	10000
E	Resource-3	QA Engineer	H	20000	4000

9.2.3. Risks

Two risk events can escalate resource costs if they occur.

```
risks <- data.frame(
  Risk_ID        = c("A", "B"),
  Risk           = c("Requirements Scope Creep", "Technical Complexity"),
  Probability    = c(0.70, 0.60),
  Resource       = c("Business Analyst", "Developer"),
  Mean_if_occurs = c(30000, 80000),
  SD_if_occurs   = c(8000,  20000)
)
knitr::kable(risks, caption = "Project Risks")
```

Table 9.3.: Project Risks

Risk_ID	Risk	Probability	Resource	Mean_if_occurs	SD_if_occurs
A	Requirements Scope Creep	0.7	Business Analyst	30000	8000
B	Technical Complexity	0.6	Developer	80000	20000

If Risk-1 (Requirements Scope Creep) occurs, the Business Analyst cost rises from $15,000 to $30,000. If Risk-2 (Technical Complexity) occurs, the Developer cost rises from $50,000 to $80,000. The QA Engineer is unaffected by either risk.

9.3. Building the Bayesian Network

9.3.1. Nodes

```r
nodes <- data.frame(
  id    = c("A", "B", "C", "D", "E", "F", "G", "H", "I"),
  label = c(
    "Risk-1", "Risk-2",
    "Resource-1", "Resource-2", "Resource-3",
    "Task-1", "Task-2", "Task-3",
    "Project"
  ),
  group = c(
    "Risk", "Risk",
    "Resource", "Resource", "Resource",
    "Task", "Task", "Task",
    "Project"
  ),
  stringsAsFactors = FALSE
)
```

9.3.2. Edges

Edges encode the causal dependencies: risks affect resources, resources drive tasks, and tasks roll up to the project total.

```r
links <- data.frame(
  source = c("A", "B", "C", "D", "E", "F", "G", "H"),
  target = c("C", "D", "F", "G", "H", "I", "I", "I"),
  value  = rep(1, 8),
  stringsAsFactors = FALSE
)
```

9.3.3. Distributions

```r
distributions <- list(
  A = list(type = "discrete", values = c(1, 0), probs = c(0.70, 0.30)),
  B = list(type = "discrete", values = c(1, 0), probs = c(0.60, 0.40)),
  C = list(
    type = "conditional", condition = "A",
    true_dist  = list(type = "normal", mean = 30000, sd = 8000),
    false_dist = list(type = "normal", mean = 15000, sd = 3000)
  ),
  D = list(
    type = "conditional", condition = "B",
    true_dist  = list(type = "normal", mean = 80000, sd = 20000),
    false_dist = list(type = "normal", mean = 50000, sd = 10000)
  ),
```

```
  E = list(type = "normal", mean = 20000, sd = 4000),
  F = list(type = "aggregate", nodes = c("C")),
  G = list(type = "aggregate", nodes = c("D")),
  H = list(type = "aggregate", nodes = c("E")),
  I = list(type = "aggregate", nodes = c("F", "G", "H"))
)
```

9.3.4. Build the Graph

```
graph <- prob_net(nodes, links, distributions = distributions)
```

The network can be visualized with the `igraph` and `networkD3` packages.

```
library(igraph)
library(networkD3)
g   <- graph_from_data_frame(graph$links, vertices = graph$nodes, directed = TRUE)
d3g <- igraph_to_networkD3(g, group = graph$nodes$group)
```

```
forceNetwork(
  Links = d3g$links, Nodes = d3g$nodes, NodeID = "name", Group = "group",
  Value = "value", zoom = TRUE, legend = TRUE, arrows = TRUE,
  opacity = 0.8, fontSize = 14
)
```

```
plot(
  g,
  vertex.color = as.factor(graph$nodes$group),
  vertex.size = 14, vertex.label.cex = 0.7,
  edge.arrow.size = 0.4, layout = layout_with_sugiyama(g)$layout
)
```

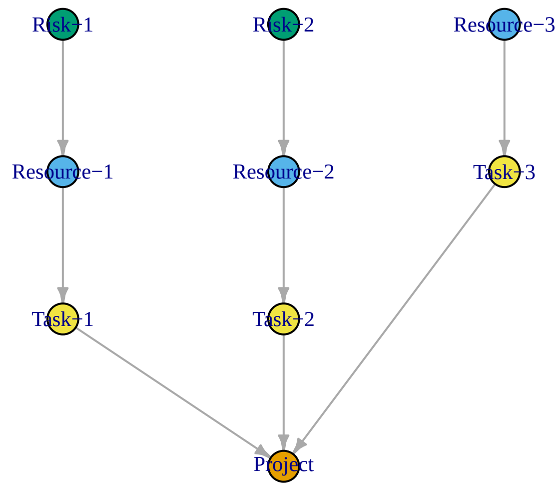

Figure 9.1.: Probabilistic network of risks, resources, tasks, and project cost.

9.4. Inference: Forward Simulation

Use `prob_net_sim()` to forward-simulate the network and estimate the total project cost distribution.

```
sim_results <- prob_net_sim(graph, num_samples = 10000)
```

```
hist(sim_results$I, breaks = 60,
     main = "Total Project Cost",
     xlab = "Cost ($)", col = "skyblue", border = "white")
```

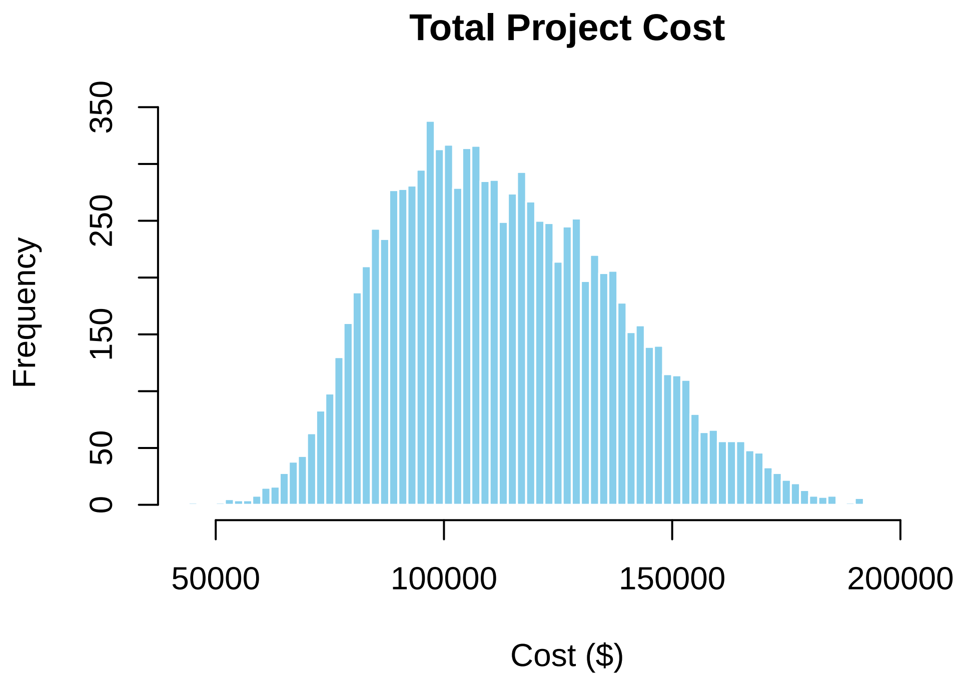

Figure 9.2.: Total project cost distribution from 10,000 forward simulations. The right tail represents scenarios where both risks occur simultaneously.

The spread reflects compounded uncertainty from both risk events. The right tail represents the worst case: both risks occur.

9.5. Learning: Incorporating New Evidence

Use `prob_net_learn()` to clamp one or more nodes to observed values and re-simulate. This shows the downstream effect of new information, for example, learning that Technical Complexity (Risk-2) did not materialize.

```
learn_results <- prob_net_learn(
  graph,
  observations = list(B = "No"),
  num_samples  = 10000
)
```

```
hist_before <- hist(sim_results$D,   breaks = 60, plot = FALSE)
hist_after  <- hist(learn_results$D, breaks = 60, plot = FALSE)

plot(
  hist_before,
```

```
  main  = "Developer Cost: Before vs. After Observing Risk-2 = No",
  xlab  = "Cost ($)", col = "skyblue", border = "white",
  ylim  = c(0, max(hist_before$counts, hist_after$counts))
)
plot(hist_after, col = rgb(0, 0, 1, 0.5), border = "white", add = TRUE)
legend(
  "topright",
  legend = c("Before (Risk-2 uncertain)", "After (Risk-2 = No)"),
  fill   = c("skyblue", rgb(0, 0, 1, 0.5)), bty = "n"
)
```

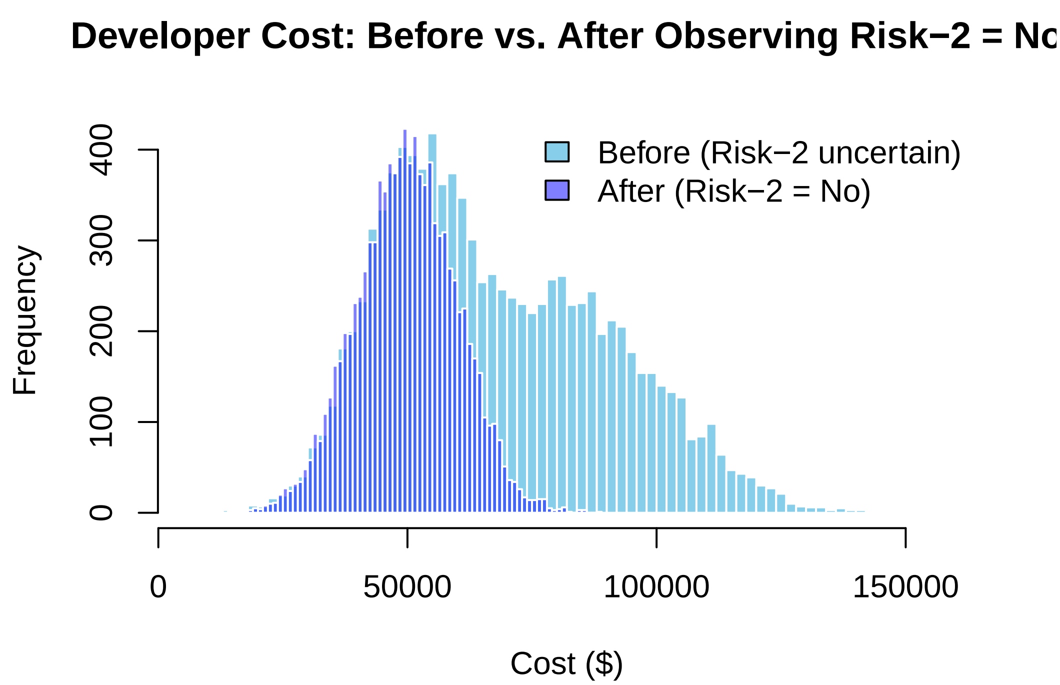

Figure 9.3.: Developer cost before and after observing Risk-2 = No. The distribution collapses to the lower baseline.

With Risk-2 ruled out, the Developer cost collapses to the lower baseline distribution, and the total project cost shifts left accordingly.

9.6. Updating: Modifying the Network

Use `prob_net_update()` to modify the network structure or distributions. Suppose a design review eliminates Requirements Scope Creep as a concern: remove the arc from Risk-1 to Resource-1 and replace the conditional distribution with a fixed normal.

```r
updated_graph <- prob_net_update(
  graph,
  remove_links = data.frame(source = "A", target = "C", stringsAsFactors = FALSE),
  update_distributions = list(
    C = list(type = "normal", mean = 15000, sd = 3000)
  )
)
updated_results <- prob_net_sim(updated_graph, num_samples = 10000)
```

```r
hist_before <- hist(sim_results$C,      breaks = 60, plot = FALSE)
hist_after  <- hist(updated_results$C,  breaks = 60, plot = FALSE)

plot(
  hist_before,
  main  = "Business Analyst Cost: Before vs. After Removing Risk-1",
  xlab  = "Cost ($)", col = "skyblue", border = "white",
  ylim  = c(0, max(hist_before$counts, hist_after$counts))
)
plot(hist_after, col = rgb(0, 0, 1, 0.5), border = "white", add = TRUE)
legend(
  "topright",
  legend = c("Before (Risk-1 possible)", "After (Risk-1 removed)"),
  fill   = c("skyblue", rgb(0, 0, 1, 0.5)), bty = "n"
)
```

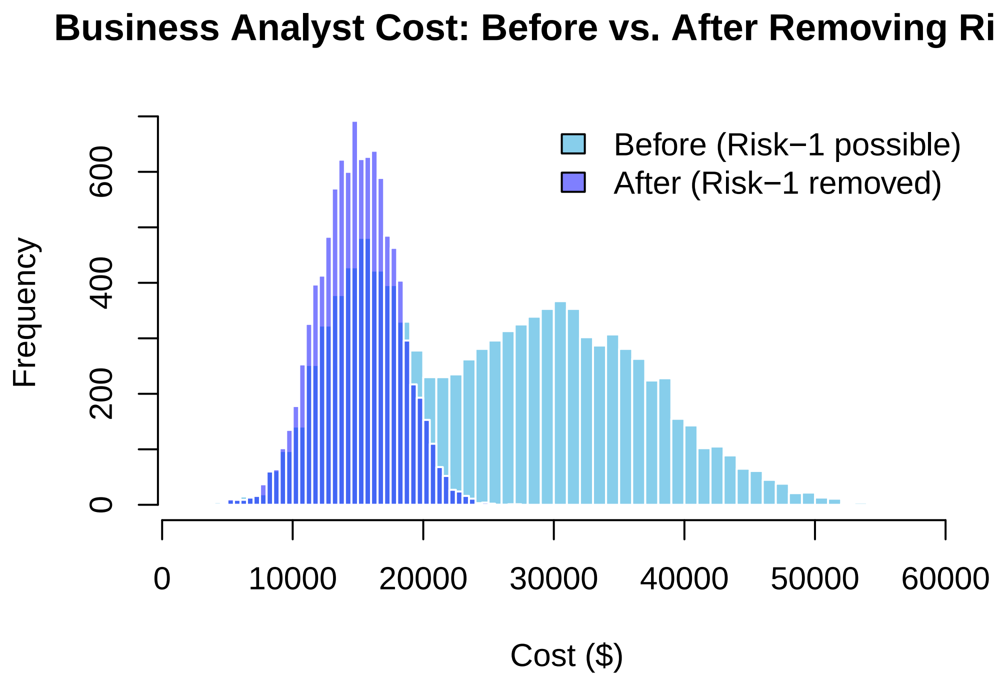

Figure 9.4.: Business Analyst cost before and after removing the Risk-1 arc. The heavy right tail disappears.

9.7. The Four Core Functions

Function	What it does
`prob_net()`	Constructs the network from nodes, edges, and distributions
`prob_net_sim()`	Forward-simulates to estimate cost distributions
`prob_net_learn()`	Clamps observed nodes and re-simulates to propagate evidence
`prob_net_update()`	Modifies network structure and distributions as the project evolves

9.8. Summary

Key Takeaways

- A probabilistic network models the full dependency chain from risks → resources → tasks → project cost, encoding uncertainty at every layer.
- **`prob_net_sim()`** draws Monte Carlo samples in topological order, propagating risk states through the network to produce a total cost distribution.
- **`prob_net_learn()`** conditions on observed evidence (*seeing*): clamping a node to its observed value and re-simulating the downstream distribution.
- **`prob_net_update()`** performs structural changes (*doing*): removing edges or replacing distributions, enabling causal interventions beyond what conditioning can express.
- The difference between *seeing* and *doing* becomes consequential when nodes share upstream parents; see Chapter 10 for the full treatment with shared enterprise risks.

9.9. Exercises

1. **Learning effect.** In the example, we observed Risk-2 = No. What do you expect happens to the total project cost distribution (node I) when Risk-2 is observed as "Yes" instead? Test your prediction by running `prob_net_learn()` with `list(B = "Yes")` and plotting the result.
2. **Modify a risk.** Change the probability of Risk-1 from 0.70 to 0.30. How does this affect the mean total project cost from `prob_net_sim()`? Is the change proportional to the probability change?
3. **Add a QA risk.** Modify the network so that a new Risk-3 (with probability 0.40) affects the QA Engineer (Resource-3), increasing their cost from a mean of $20K to $35K if it occurs. Update the distributions and re-simulate. How much does this add to the expected project cost?
4. **Seeing vs. doing.** Explain the difference between `prob_net_learn(observations = list(B = "No"))` and `prob_net_update(remove_links = ...)`. When would you use each? Which corresponds to "seeing" and which to "doing" in the causal inference sense? (See Chapter 10 for the full treatment.)
5. **Project structure.** This example has a simple layered structure (Risk → Resource → Task → Project). Design a more complex network with two risks that both affect the same resource. What does that mean for the correlation between the two downstream tasks? Build the network and verify with a correlation matrix of `sim_results`.

10. The Portfolio Problem: When Risks Are Shared

"Seeing a fire is not the same as starting one." — The First Law of Causal Inference

In Chapter 9, we built a probabilistic network for one project. Now we're going to scale up to a portfolio of three projects, introduce an upstream root cause, and answer the question that keeps enterprise risk managers up at night: when one project gets hit by a risk, what happens to all the others?

The answer depends on whether the risks are truly independent or whether they share a common driver. And it depends on something subtle but important: the difference between *seeing* a risk happen and *doing* something to prevent it. These two things look similar but behave completely differently in a causal network.

Learning Objectives

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

1. Model a multi-project portfolio with shared enterprise risks and a root cause
2. Compute and interpret the task-cost correlation matrix driven by shared risks
3. Distinguish observational, seeing, and doing queries on a causal network
4. Implement graph surgery with `prob_net_update()` to model interventions
5. Rank enterprise risks by their contribution to portfolio cost variance

10.1. The Portfolio Case Study

Consider an enterprise running three projects simultaneously: a **Road Repair**, a **Park Construction**, and a **Building Renovation**. Each project has three tasks driven by three resources: labor, materials, and equipment.

Critically, these resources are **shared at the enterprise level**: the same labor pool, the same material suppliers, and the same equipment fleet serve all three projects. Three enterprise-wide risk events propagate through the corresponding shared resource to impact all three projects simultaneously.

Even more critically, two of these risks, a **Labor Shortage** and a **Material Price Spike**, share a common upstream driver: a **Supply Chain Disruption** that simultaneously tightens the labor market and raises commodity prices. This shared root cause is what makes the distinction between *seeing* and *doing* consequential (Pearl 2009; Govan 2014).

> ⚠ Seeing Is Not the Same as Doing
>
> **Seeing** $A = 1$ (observing a Labor Shortage) conditions on the evidence in the *original* graph. Bayes' rule propagates upstream: it raises the posterior probability of the Supply Chain Disruption, which in turn raises the probability of the Material Price Spike. A side-effect flows through the shared root cause.
> **Doing** do($A = 1$) (intervening to cause a Labor Shortage) severs the SC → A edge via graph surgery. SC stays at its prior, B stays near its prior, and material costs are unaffected. Only labor costs change.
> The gap between the two distributions, seeing vs. doing, is the operational signature of the shared root cause SC. Confusing the two leads to systematically underestimating portfolio cost exposure (Pearl 2009).

10.2. Setup

```r
library(PRA)
library(igraph)
library(networkD3)
library(corrplot)
set.seed(42)
```

10.2.1. Tasks

Table 10.1.: Project 1: Road Repair

ID	Label	Task
M	Task-1.1	Site Preparation
N	Task-1.2	Road Paving
O	Task-1.3	Final Inspection

Table 10.2.: Project 2: Park Construction

ID	Label	Task
P	Task-2.1	Site Preparation
Q	Task-2.2	Planting and Landscaping
R	Task-2.3	Final Inspection

Table 10.3.: Project 3: Building Renovation

ID	Label	Task
S	Task-3.1	Demolition
T	Task-3.2	Renovation and Build-Out
U	Task-3.3	Final Inspection

10.2.2. Risks and Root Cause

```r
root_cause <- data.frame(
  ID = "SC", Event = "Supply Chain Disruption", P_occurs = 0.70,
  Children = "A (Labor Shortage), B (Material Price Spike)"
)
risks <- data.frame(
  Risk_ID = c("A", "B", "C"),
  Risk    = c("Labor Shortage", "Material Price Spike", "Weather Delay"),
  Parent  = c("SC", "SC", "-"),
  P_marginal = c("  0.79", "  0.67", "0.60"),
  Resource_Impacted = c("Labor (D,G,J)", "Materials (E,H,K)", "Equipment (F,I,L)")
)
knitr::kable(root_cause, caption = "Root Cause (SC)")
```

Table 10.4.: Root Cause (SC)

ID	Event	P_occurs	Children
SC	Supply Chain Disruption	0.7	A (Labor Shortage), B (Material Price Spike)

```r
knitr::kable(risks, caption = "Enterprise Risks")
```

Table 10.5.: Enterprise Risks

Risk_ID	Risk	Parent	P_marginal	Resource_Impacted
A	Labor Shortage	SC	0.79	Labor (D,G,J)
B	Material Price Spike	SC	0.67	Materials (E,H,K)
C	Weather Delay	—	0.60	Equipment (F,I,L)

The shared root cause SC is what makes *seeing* and *doing* diverge: observing $A = 1$ raises the posterior probability of SC, which in turn raises the probability of $B = 1$, a side-effect that do$(A = 1)$ does not produce.

10.3. Building the Portfolio Network

```
nodes <- data.frame(
  id = c("SC","A","B","C","D","E","F","G","H","I","J","K","L",
         "M","N","O","P","Q","R","S","T","U","V","W","X","Y"),
  label = c(
    "Supply Chain Disruption",
    "Risk-1 (Labor Shortage)","Risk-2 (Material Price Spike)","Risk-3 (Weather Delay)",
    "Labor-1","Materials-1","Equipment-1",
    "Labor-2","Materials-2","Equipment-2",
    "Labor-3","Materials-3","Equipment-3",
    "Task-1.1","Task-1.2","Task-1.3",
    "Task-2.1","Task-2.2","Task-2.3",
    "Task-3.1","Task-3.2","Task-3.3",
    "Project 1","Project 2","Project 3","Portfolio"
  ),
  group = c(
    "Root Cause","Risk","Risk","Risk",
    "Resource","Resource","Resource","Resource","Resource","Resource",
    "Resource","Resource","Resource",
    "Task","Task","Task","Task","Task","Task","Task","Task","Task",
    "Project","Project","Project","Portfolio"
  ),
  stringsAsFactors = FALSE
)

links <- data.frame(
  source = c(
    "SC","SC",
    "A","A","A","B","B","B","C","C","C",
    "D","E","F","G","H","I","J","K","L",
    "M","N","O","P","Q","R","S","T","U",
    "V","W","X"
  ),
  target = c(
    "A","B",
    "D","G","J","E","H","K","F","I","L",
    "M","N","O","P","Q","R","S","T","U",
    "V","V","V","W","W","W","X","X","X",
    "Y","Y","Y"
  ),
  value = rep(1, 32)
)

distributions <- list(
  SC = list(type = "discrete", values = c(1, 0), probs = c(0.7, 0.3)),
```

```
A  = list(type = "conditional", condition = "SC",
  true_dist  = list(type = "discrete", values = c(1, 0), probs = c(0.95, 0.05)),
  false_dist = list(type = "discrete", values = c(1, 0), probs = c(0.40, 0.60))),
B  = list(type = "conditional", condition = "SC",
  true_dist  = list(type = "discrete", values = c(1, 0), probs = c(0.85, 0.15)),
  false_dist = list(type = "discrete", values = c(1, 0), probs = c(0.25, 0.75))),
C  = list(type = "discrete", values = c(1, 0), probs = c(0.6, 0.4)),
D  = list(type = "conditional", condition = "A",
  true_dist  = list(type = "normal", mean = 50000, sd = 8000),
  false_dist = list(type = "normal", mean = 30000, sd = 5000)),
G  = list(type = "conditional", condition = "A",
  true_dist  = list(type = "normal", mean = 40000, sd = 6000),
  false_dist = list(type = "normal", mean = 25000, sd = 4000)),
J  = list(type = "conditional", condition = "A",
  true_dist  = list(type = "normal", mean = 65000, sd = 10000),
  false_dist = list(type = "normal", mean = 40000, sd = 6000)),
E  = list(type = "conditional", condition = "B",
  true_dist  = list(type = "normal", mean = 80000, sd = 12000),
  false_dist = list(type = "normal", mean = 50000, sd = 8000)),
H  = list(type = "conditional", condition = "B",
  true_dist  = list(type = "normal", mean = 50000, sd = 8000),
  false_dist = list(type = "normal", mean = 30000, sd = 5000)),
K  = list(type = "conditional", condition = "B",
  true_dist  = list(type = "normal", mean = 100000, sd = 15000),
  false_dist = list(type = "normal", mean = 60000, sd = 10000)),
F  = list(type = "conditional", condition = "C",
  true_dist  = list(type = "normal", mean = 35000, sd = 6000),
  false_dist = list(type = "normal", mean = 20000, sd = 4000)),
I  = list(type = "conditional", condition = "C",
  true_dist  = list(type = "normal", mean = 25000, sd = 4000),
  false_dist = list(type = "normal", mean = 15000, sd = 3000)),
L  = list(type = "conditional", condition = "C",
  true_dist  = list(type = "normal", mean = 40000, sd = 6000),
  false_dist = list(type = "normal", mean = 25000, sd = 4000)),
M = list(type = "aggregate", nodes = "D"),
N = list(type = "aggregate", nodes = "E"),
O = list(type = "aggregate", nodes = "F"),
P = list(type = "aggregate", nodes = "G"),
Q = list(type = "aggregate", nodes = "H"),
R = list(type = "aggregate", nodes = "I"),
S = list(type = "aggregate", nodes = "J"),
T = list(type = "aggregate", nodes = "K"),
U = list(type = "aggregate", nodes = "L"),
V = list(type = "aggregate", nodes = c("M","N","O")),
W = list(type = "aggregate", nodes = c("P","Q","R")),
X = list(type = "aggregate", nodes = c("S","T","U")),
Y = list(type = "aggregate", nodes = c("V","W","X"))
```

```
)

graph <- prob_net(nodes, links, distributions = distributions)
```

10.4. Observational Distribution

The observational distribution $P(Y)$ reflects the full uncertainty of all three shared risks.

```
sim_results <- prob_net_sim(graph, num_samples = 1000)
```

```
hist(sim_results$Y, breaks = 50,
     main = expression("Observational distribution " * italic(P) * "(Y)"),
     xlab = "Portfolio Cost ($)", col = "skyblue", border = "white")
```

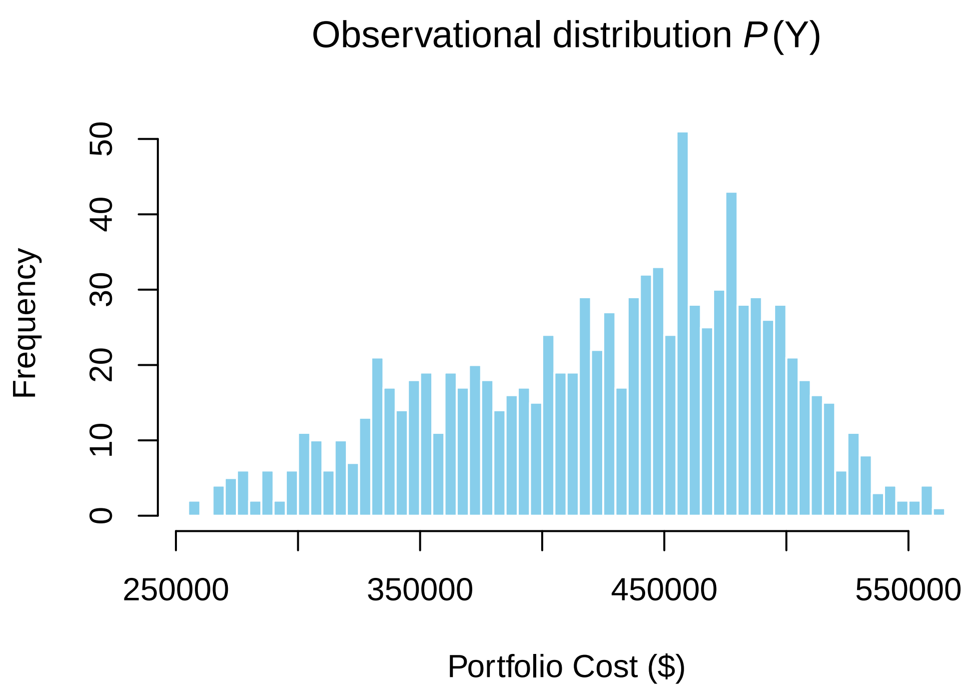

Figure 10.1.: Observational portfolio cost distribution P(Y). Heavy tails reflect compounded shared-risk uncertainty.

10.4.1. Task-Cost Correlation Matrix

The correlation matrix reveals which tasks are coupled through shared risks.

```r
task_costs <- sim_results[, c("M","N","O","P","Q","R","S","T","U")]
cor_matrix <- cor(task_costs)
corrplot(cor_matrix,
         method = "color", type = "full",
         addCoef.col = "black", number.cex = 0.7,
         tl.col = "black", tl.srt = 45,
         col = colorRampPalette(c("white", "steelblue"))(100),
         mar = c(0, 0, 1, 0))
```

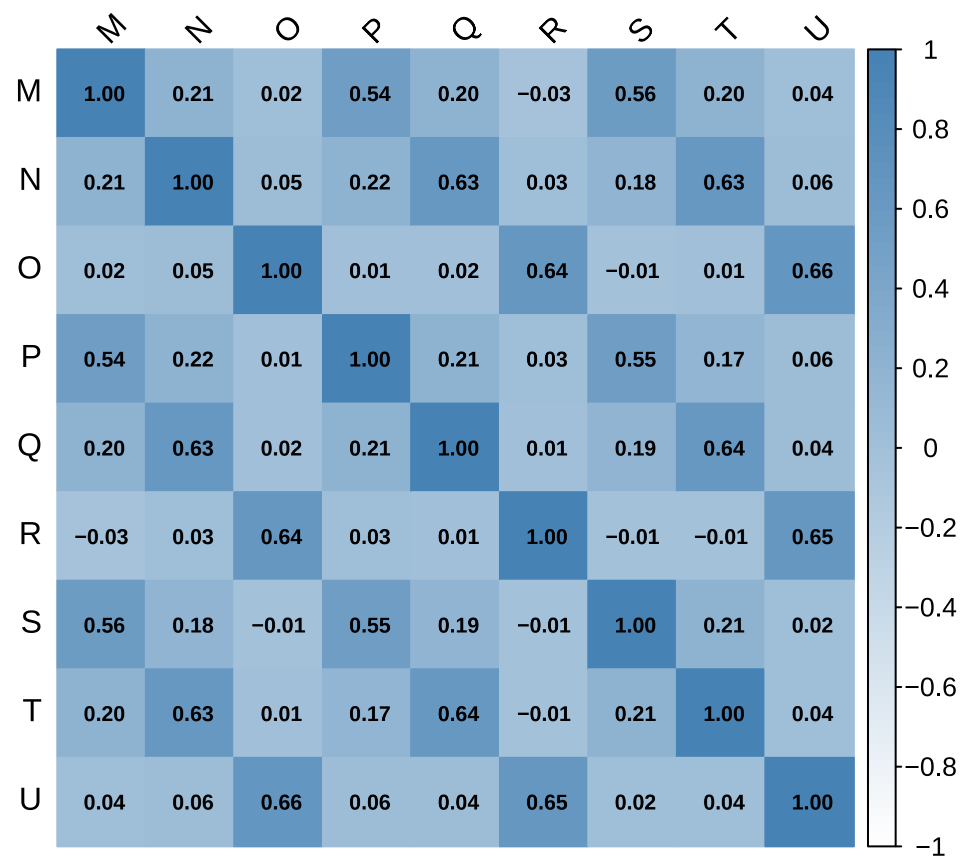

Figure 10.2.: Task-cost correlation matrix. Tasks sharing the same enterprise risk (Labor, Materials, Equipment) are strongly correlated; tasks driven by different risks are near zero.

10.5. Seeing vs. Doing at the Risk Level

This is where the real insight lives. **Seeing** $A = 1$ conditions on the observation in the original graph: evidence propagates upstream as well as downstream. **Doing** $\text{do}(A = 1)$ performs graph surgery, severing SC→A and fixing $A = 1$. The two operations produce different results because of the shared root cause SC.

```r
seeing_results <- prob_net_learn(graph, observations = list(A = 1), num_samples = 1000)

do_A1_graph <- prob_net_update(graph,
  remove_links         = data.frame(source = "SC", target = "A", stringsAsFactors = FALSE),
  update_distributions = list(A = list(type = "discrete", values = c(1, 0), probs = c(1, 0)))
)
do_A1_results <- prob_net_sim(do_A1_graph, num_samples = 1000)
```

```r
h1     <- hist(sim_results$Y,    breaks = 50, plot = FALSE)
h2     <- hist(seeing_results$Y, breaks = 50, plot = FALSE)
h_do   <- hist(do_A1_results$Y,  breaks = 50, plot = FALSE)

plot(h1, main = "P(Y) vs. Seeing vs. Doing: Labor Shortage",
     xlab = "Portfolio Cost ($)", col = "skyblue", border = "white",
     ylim = c(0, max(h1$counts, h2$counts, h_do$counts)))
plot(h_do, col = rgb(1, 0.5, 0, 0.5), border = "white", add = TRUE)
plot(h2,   col = rgb(0, 0, 1, 0.4),   border = "white", add = TRUE)
legend("topright",
  legend = c("P(Y): prior",
             "P(Y | do(A=1)): doing",
             "P(Y | A=1): seeing"),
  fill = c("skyblue", rgb(1, 0.5, 0, 0.5), rgb(0, 0, 1, 0.4)), bty = "n")
```

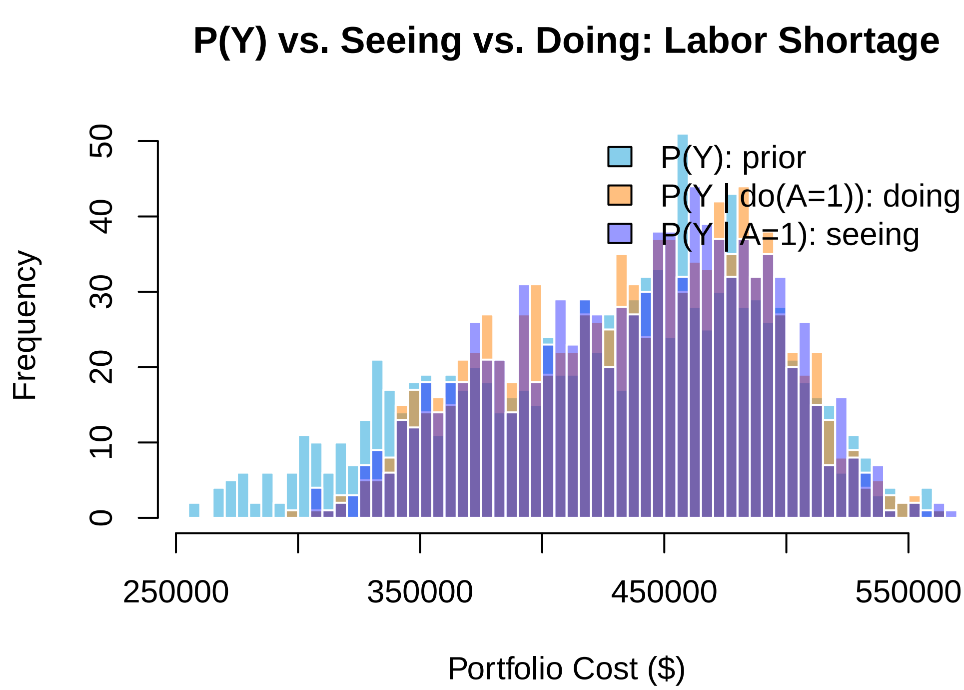

Figure 10.3.: Prior, doing, and seeing distributions for portfolio cost. Seeing shifts further right than doing because it propagates through the shared root cause SC.

The key distinction: *Seeing* a Labor Shortage raises our belief that a Supply Chain Disruption is underway, which in turn raises the probability of a Material Price Spike, a side-effect that propagates through SC. *Doing* do($A = 1$) severs SC→A, so SC stays at its prior and material costs are unaffected. The gap between the two distributions is the operational signature of the shared root cause.

10.6. Enterprise vs. Project-Scoped Intervention

Two intervention strategies can reduce labor-risk exposure:

1. **Project-scoped**: Secure a dedicated crew for Building Renovation only (do($J \to$ baseline))
2. **Enterprise prevention**: Eliminate the Labor Shortage entirely (do($A = 0$))

```
do_J_graph <- prob_net_update(graph,
  remove_links         = data.frame(source = "A", target = "J", stringsAsFactors = FALSE),
  update_distributions = list(J = list(type = "normal", mean = 40000, sd = 6000))
)
do_J_results <- prob_net_sim(do_J_graph, num_samples = 1000)

do_A0_graph <- prob_net_update(graph,
  remove_links         = data.frame(source = "SC", target = "A", stringsAsFactors = FALSE),
  update_distributions = list(A = list(type = "discrete", values = c(1, 0), probs = c(0, 1)))
)
do_A0_results <- prob_net_sim(do_A0_graph, num_samples = 1000)
```

```
h3 <- hist(do_J_results$Y,  breaks = 50, plot = FALSE)
h4 <- hist(do_A0_results$Y, breaks = 50, plot = FALSE)

plot(h1, main = "Project-scoped vs. Enterprise-level Labor Mitigation",
     xlab = "Portfolio Cost ($)", col = "skyblue", border = "white",
     ylim = c(0, max(h1$counts, h3$counts, h4$counts)))
plot(h3, col = rgb(0, 0.6, 0, 0.4),   border = "white", add = TRUE)
plot(h4, col = rgb(0.8, 0.4, 0, 0.5), border = "white", add = TRUE)
legend("topright",
  legend = c("P(Y): observational",
             "do(J → baseline): Project 3 insulated",
             "do(A = 0): enterprise prevention"),
  fill = c("skyblue", rgb(0, 0.6, 0, 0.4), rgb(0.8, 0.4, 0, 0.5)), bty = "n")
```

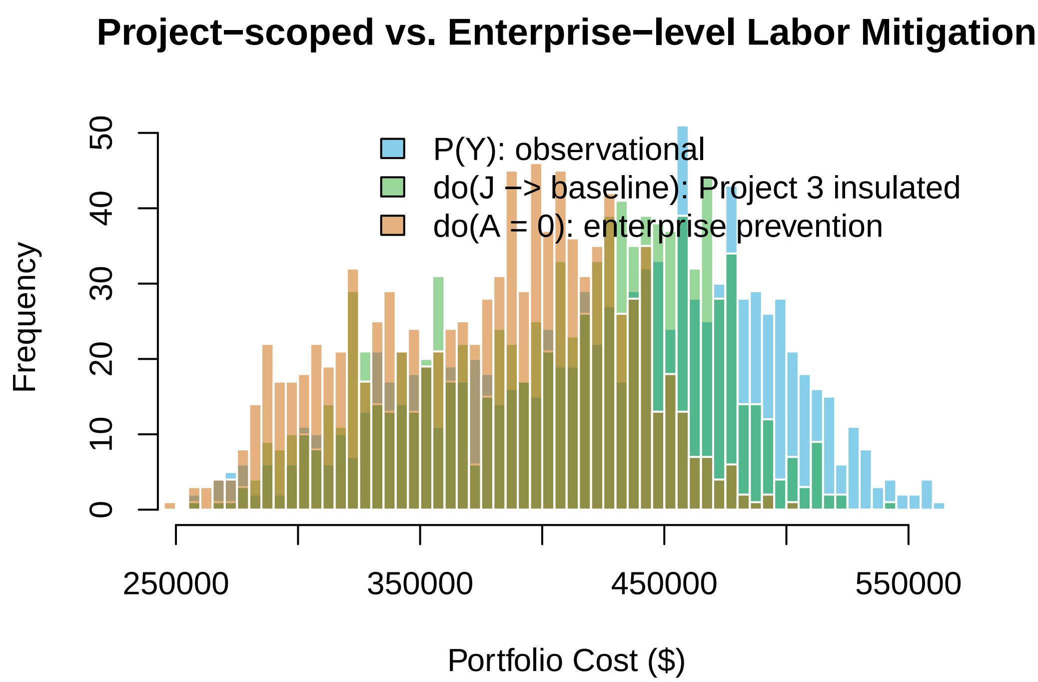

Figure 10.4.: Project-scoped vs. enterprise-level labor risk mitigation. Enterprise prevention eliminates exposure across all three projects.

10.7. Risk Importance Ranking

Which shared risk contributes most to portfolio cost variance? Run three enterprise-prevention interventions and measure the variance reduction each produces.

```
prevent_risk <- function(graph, parent_node, risk_node) {
  remove_df <- if (!is.null(parent_node))
    data.frame(source = parent_node, target = risk_node, stringsAsFactors = FALSE)
  else
    data.frame(source = character(0), target = character(0), stringsAsFactors = FALSE)
  prob_net_update(graph,
    remove_links = remove_df,
    update_distributions = setNames(
      list(list(type = "discrete", values = c(1, 0), probs = c(0, 1))),
      risk_node
    )
  )
}

g_A0 <- prevent_risk(graph, "SC", "A")
g_B0 <- prevent_risk(graph, "SC", "B")
```

```
g_C0 <- prevent_risk(graph, NULL, "C")

r_A0 <- prob_net_sim(g_A0, num_samples = 1000)
r_B0 <- prob_net_sim(g_B0, num_samples = 1000)
r_C0 <- prob_net_sim(g_C0, num_samples = 1000)

var_base  <- var(sim_results$Y)
importance <- c(
  "Labor\nShortage (A)"        = var_base - var(r_A0$Y),
  "Material\nPrice Spike (B)" = var_base - var(r_B0$Y),
  "Weather\nDelay (C)"         = var_base - var(r_C0$Y)
)

barplot(importance,
  main   = "Risk Importance: Variance Eliminated by do(risk = 0)",
  ylab   = expression("Variance reduction in Y ($"^2 * ")"),
  col    = c("steelblue", "coral", "seagreen"),
  border = "white", las = 1
)
```

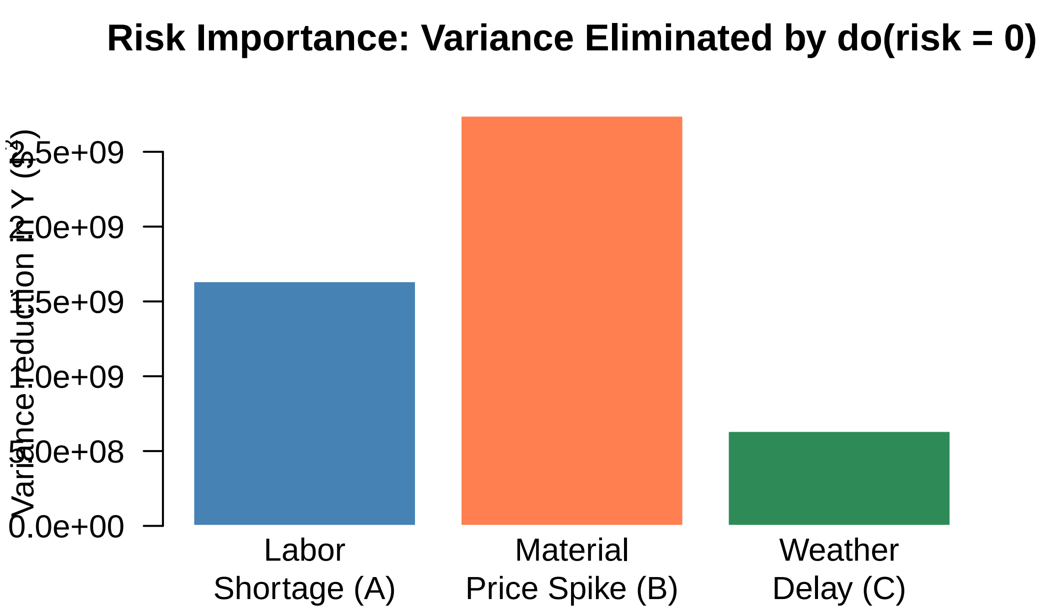

Figure 10.5.: Risk importance: portfolio variance eliminated by preventing each enterprise risk. Taller bars = higher priority for mitigation.

The bar heights rank the three enterprise risks by their contribution to portfolio cost variance, a principled basis for prioritizing mitigation investments.

10.8. Summary Table

```r
make_stats <- function(x) c(round(mean(x)/1000, 1), round(sd(x)/1000, 1),
                            round(quantile(x, 0.95)/1000, 1))
stats_mat <- rbind(
  make_stats(sim_results$Y),
  make_stats(seeing_results$Y),
  make_stats(do_A1_results$Y),
  make_stats(do_J_results$Y),
  make_stats(do_A0_results$Y)
)
stats_df <- as.data.frame(stats_mat)
colnames(stats_df) <- c("Mean ($000)", "SD ($000)", "95th Pct ($000)")
rownames(stats_df) <- c(
  "Observational P(Y)",
  "Seeing P(Y | A = 1)",
  "Doing do(A = 1)",
  "Doing do(J → baseline) [Project 3]",
  "Doing do(A = 0) [Enterprise prevention]"
)
knitr::kable(stats_df, caption = "Portfolio cost statistics across all scenarios.")
```

Table 10.6.: Portfolio cost statistics across all scenarios.

	Mean ($000)	SD ($000)	95th Pct ($000)
Observational P(Y)	425.9	65.0	516.7
Seeing P(Y | A = 1)	438.6	52.9	515.6
Doing do(A = 1)	438.7	52.3	514.4
Doing do(J → baseline) [Project 3]	409.0	57.1	486.5
Doing do(A = 0) [Enterprise prevention]	377.5	52.1	452.4

10.9. Summary

Key Takeaways

- Shared enterprise risks create **positive correlation across all projects** that share those resources, driving portfolio variance is always larger than the naive sum of individual project variances.
- The **task-cost correlation matrix** reveals the block structure of risk sharing: tasks driven by the same enterprise risk cluster together, while tasks driven by different risks are near-zero.
- **Seeing** conditions on evidence in the original graph; it propagates upstream through shared root causes, raising the probability of sibling risks as a side-effect.

- **Doing** (graph surgery) severs incoming edges of the intervened node; it changes only what the intervention directly controls, without raising posteriors about shared root causes.
- Enterprise-level risk prevention ($\text{do}(A = 0)$) reduces portfolio exposure across all projects simultaneously; project-scoped insulation ($\text{do}(J \to \text{baseline})$) reduces exposure for only one.
- **Risk importance ranking** by variance reduction from $\text{do}(\text{risk} = 0)$ gives a principled basis for prioritizing mitigation budgets.

10.10. Exercises

1. **Seeing vs. doing.** In your own words, explain why $P(Y \mid A = 1) \neq P(Y \mid \text{do}(A = 1))$ in this model. What structural feature of the network causes the difference? What would have to change in the graph for the two to be equal?

2. **Risk importance.** In the risk importance ranking, which enterprise risk had the largest effect on portfolio variance? What is it about that risk's position in the network that makes it particularly impactful?

3. **Two risks confirmed.** Run `prob_net_learn(graph, observations = list(A = 1, B = 1), num_samples = 1000)`. How does the portfolio cost distribution compare to seeing only $A = 1$? Is the combined effect larger or smaller than you'd expect from adding the individual effects?

4. **Add a fourth project.** Extend the network with a fourth project (e.g., a Bridge Inspection) with three tasks and three resources (Labor-4, Materials-4, Equipment-4), driven by the same three enterprise risks A, B, C. Add it to the portfolio node Y. How does total portfolio variance change? Does the risk importance ranking shift?

5. **Causal graph design.** In the current model, Supply Chain Disruption (SC) drives both Labor Shortage (A) and Material Price Spike (B). Suppose you learn that, in practice, SC also affects Weather Delay (C) through disrupted logistics. Add an SC → C edge and update the C distribution to be conditional on SC. How does this change the see-versus-do analysis for C?

11. Your AI Co-Pilot: Agentic Risk Analysis

> *"The best tool for the job is the one you'll actually use."* — P.G.

You've made it through eight chapters of quantitative risk analysis. You know Monte Carlo, SMM, EVM, Bayesian inference, learning curves, design structure matrices, and probabilistic networks. Now imagine not having to remember the syntax for all of them.

Imagine a project manager typing "run Monte Carlo for three tasks with Normal(10,2), Triangular(5,10,15), Uniform(8,12)" and getting back a P95 date, a contingency reserve, and a tornado chart, all in ten seconds, without opening R. That's the promise of the `PRA` agentic framework: a layer of intelligence on top of all the methods you've learned.

Learning Objectives

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

1. Distinguish the three routing modes (slash commands, LLM tool calls, RAG)
2. Run any PRA method via a slash command and chain results across commands
3. Use `pra_chat()` to interact with PRA methods through natural language
4. Explain how RAG augments the LLM with domain-specific knowledge
5. Configure PRA as an MCP server for Claude Desktop or Claude Code

11.1. The Three Routing Modes

How PRA Routes Your Input

Input type	Route	Example
`/command`	**Deterministic**: executes the tool directly, no LLM	`/mcs tasks=[...]`
Numerical data	**LLM tool call**: the model selects and calls the right tool	"Simulate 3 tasks: Normal(10,2)…"
Conceptual question	**RAG**: answered from the knowledge base	"What is earned value?"

When to use each:

- Use **slash commands** when you need guaranteed, reproducible results; they bypass the LLM entirely.
- Use the **chat interface** for exploratory questions and interpretation, especially with larger models.
- Use **RAG** for conceptual questions; it retrieves the most relevant knowledge base content and cites sources.

Three interfaces expose these modes:

1. **Slash commands**: deterministic tool calls that bypass the LLM for instant, reliable results
2. **Chat interface** (`pra_chat()`): programmatic R chat powered by ellmer
3. **Shiny app** (`pra_app()`): browser-based experience combining all three modes

11.2. Prerequisites

11.2.1. Install Ollama

Download from https://ollama.com, then pull a model:

```
ollama serve
ollama pull llama3.2
ollama pull nomic-embed-text  # for RAG embeddings
```

11.2.2. Install R dependencies

```r
install.packages(c("ellmer", "ragnar", "shiny", "bslib", "shinychat", "jsonlite"))
```

11.3. Slash Commands

Slash commands provide deterministic tool execution with no LLM required. Type `/help` to see all available commands, or `/help <command>` for detailed usage.

11.3.1. Available Commands

```r
library(PRA)
cat(PRA:::format_help_overview())
```

```
## PRA Commands

Type a command to run an analysis directly. Type `/help <command>` for detailed usage.

- **/mcs** - Monte Carlo Simulation
- **/smm** - Second Moment Method
- **/contingency** - Contingency Reserve
- **/sensitivity** - Sensitivity Analysis
- **/evm** - Earned Value Management
- **/risk** - Bayesian Risk (Prior)
- **/risk_post** - Bayesian Risk (Posterior)
- **/learning** - Learning Curve Fit
- **/dsm** - Design Structure Matrix

Type `/help` for this overview.
```

11.3.2. Example: Monte Carlo Simulation

```r
set.seed(42)
cmd <- paste0(
  '/mcs n=10000 tasks=[',
  '{"type":"normal","mean":10,"sd":2},',
  '{"type":"triangular","a":5,"b":10,"c":15},',
  '{"type":"uniform","min":8,"max":12}]'
)
r <- PRA::execute_command(cmd)
cat(r$result)
```

```
Monte Carlo Simulation Results (n = 10,000):

Summary Statistics:
  Mean  29.9804
  SD    3.113
  Min   19.502
  Max   41.3892

Percentiles:
  P5   24.8726
  P10  26.0194
  P25  27.8881
  P50  29.9536
  P75  32.0796
  P90  33.9873
  P95  35.1014
```

```
result <- PRA:::.pra_agent_env$last_mcs
hist(result$total_distribution,
  freq = FALSE, breaks = 50,
  main = "Monte Carlo Simulation Results",
  xlab = "Total Project Duration/Cost",
  col  = "#18bc9c80", border = "white"
)
curve(dnorm(x, mean = result$total_mean, sd = result$total_sd),
  add = TRUE, col = "#2c3e50", lwd = 2
)
abline(
  v   = quantile(result$total_distribution, c(0.50, 0.95)),
  col = c("#3498db", "#e74c3c"), lty = 2, lwd = 1.5
)
legend("topright",
  legend = c("Normal fit", "P50", "P95"),
  col    = c("#2c3e50", "#3498db", "#e74c3c"),
  lty    = c(1, 2, 2), lwd = c(2, 1.5, 1.5),
  cex    = 0.8, bg = "white"
)
```

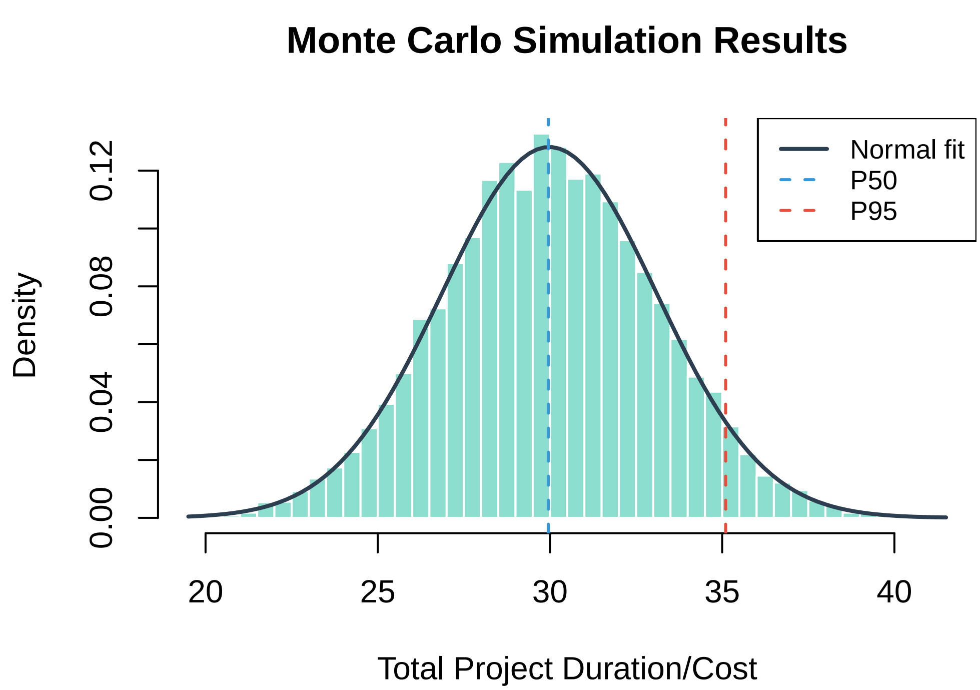

Figure 11.1.: MCS results via slash command. The /mcs command stores results for downstream chaining.

11.3.3. Example: Chaining MCS to Contingency

After running `/mcs`, chain to `/contingency` for the reserve estimate:

```
r <- PRA:::execute_command("/contingency phigh=0.95 pbase=0.50")
cat(r$result)
```

```
Contingency Analysis:

  Base Percentile      P50
  High Percentile      P95
  Contingency Reserve  5.1478
```

11.3.4. Example: Earned Value Management

Full EVM analysis with a single command:

```
cmd <- paste(
  "/evm bac=500000",
  "schedule=[0.2,0.4,0.6,0.8,1.0]",
  "period=3 complete=0.35",
  "costs=[90000,195000,310000]"
)
r <- PRA:::execute_command(cmd)
cat(r$result)
```

```
Earned Value Management Analysis:

Core Metrics:
  Planned Value (PV)  300,000
  Earned Value (EV)   175,000
  Actual Cost (AC)    310,000

Variances:
  Schedule Variance (SV)  -125,000
  Cost Variance (CV)      -135,000

Performance Indices:
  Schedule Performance Index (SPI)  0.5833
  Cost Performance Index (CPI)      0.5645

Forecasts:
  EAC (Typical)                 885,714.3
  EAC (Atypical)                635,000
  EAC (Combined)                1,296,939
  Estimate to Complete (ETC)    575,714.3
```

```
  Variance at Completion (VAC)  -385,714.3
  TCPI (to meet BAC)            1.7105
```

11.3.5. Example: Bayesian Risk Probability

```r
r <- PRA:::execute_command(
  "/risk causes=[0.3,0.2] given=[0.8,0.6] not_given=[0.2,0.4]"
)
cat(r$result)
```

```
Bayesian Risk Analysis (Prior):

  Risk Probability  0.82
  Risk Percentage   82%
  Number of Causes  2
```

Then update with observations:

```r
r <- PRA:::execute_command(
  "/risk_post causes=[0.3,0.2] given=[0.8,0.6] not_given=[0.2,0.4] observed=[1,null]"
)
cat(r$result)
```

```
Bayesian Risk Analysis (Posterior):

  Posterior Risk Probability  0.6316
  Posterior Risk Percentage   63.16%

Observations:
  Cause 1: Occurred
  Cause 2: Unknown
```

11.3.6. Example: Second Moment Method

```r
r <- PRA:::execute_command("/smm means=[10,12,8] vars=[4,9,2]")
cat(r$result)
```

```
Second Moment Method Results:

  Total Mean      30
  Total Variance  15
  Total Std Dev   3.873
```

11.3.7. Input Validation

Missing or invalid arguments produce helpful error messages:

```r
r <- PRA:::execute_command("/risk causes=[0.3]")
cat(r$result)
```

```
**Missing required argument(s):** given, not_given

### /risk - Bayesian Risk (Prior)

Calculate prior risk probability from root causes using Bayes' theorem.

**Arguments:**
- **causes** *(required)* - JSON array of cause probabilities, e.g. [0.3, 0.2]
- **given** *(required)* - JSON array of P(Risk | Cause), e.g. [0.8, 0.6]
- **not_given** *(required)* - JSON array of P(Risk | not Cause), e.g. [0.2, 0.4]

**Examples:**
  /risk causes=[0.3,0.2] given=[0.8,0.6] not_given=[0.2,0.4]
```

11.4. Chat Interface

> **About `eval: false` in This Chapter**
>
> The Chat Interface, Shiny App, and MCP sections require a running Ollama server and cannot be evaluated during book rendering. Code blocks marked `eval: false` show the correct syntax; to follow along interactively, start Ollama with `ollama serve` before running them. The slash command examples above **are** fully evaluated and show actual output.

The chat interface routes queries through the LLM, which decides whether to call a tool or answer from the RAG knowledge base:

- **Numerical data** → LLM calls the appropriate tool and interprets the results
- **Conceptual questions** → LLM answers from the RAG knowledge base with source citations

```r
chat <- pra_chat(model = "llama3.2")

# Tool call: user provides numerical data
chat$chat("Run a Monte Carlo simulation for a 3-task project with
  Task A ~ Normal(10, 2), Task B ~ Triangular(5, 10, 15),
  Task C ~ Uniform(8, 12). Use 10,000 simulations.")

# RAG: conceptual question
chat$chat("What is the difference between SPI and CPI?")
```

11.4.1. Using Cloud Models

For better accuracy with complex queries:

```
# Anthropic (replace with the current model ID from https://docs.anthropic.com/en/docs/about-cl
chat <- pra_chat(chat = ellmer::chat_anthropic(model = "claude-sonnet-4-5"))

# OpenAI
chat <- pra_chat(chat = ellmer::chat_openai(model = "gpt-4o"))
```

11.5. Interactive Shiny App

For a browser-based experience with streaming responses and inline visualizations:

```
pra_app()
```

The app supports all three input modes in the same chat panel:

- Type `/mcs tasks=[...]` for **instant deterministic** results
- Type "Simulate 3 tasks..." for **LLM tool calling**
- Type "What is earned value?" for **RAG-powered answers**

11.5.1. Features

- **Three input modes** in a single panel
- **Clickable example prompts** that execute `/commands` on click
- **Streaming chat** with token-by-token responses
- **Inline tool results** as rich HTML tables and plots
- **RAG source citations** for conceptual answers
- **Export**: download the conversation as markdown

11.6. RAG Knowledge Base

PRA includes a built-in knowledge base covering all the methods in this book:

File	Topics
`mcs_methods.md`	Distribution selection, correlation, interpreting percentiles
`evm_standards.md`	EVM metrics, performance indices, forecasting methods
`bayesian_risk.md`	Prior/posterior risk, Bayes' theorem
`learning_curves.md`	Sigmoidal models, curve fitting
`sensitivity_contingency.md`	Variance decomposition, contingency reserves
`pra_functions.md`	PRA package function reference

11.6.1. Adding Your Own Documents

```r
store <- build_knowledge_base()
add_documents(store, "path/to/my_risk_register.md")
add_documents(store, "path/to/project_docs/")
```

11.7. MCP Integration

PRA can expose all its tools as an MCP (Model Context Protocol) server, letting Claude Desktop, Claude Code, or any MCP-compatible AI client call PRA functions directly.

11.7.1. Starting the Server

```r
pra_mcp_server()
```

11.7.2. Connecting from Claude Code

```bash
claude mcp add -s project pra -- Rscript -e "PRA::pra_mcp_server()"
```

Once registered, Claude Code can call PRA tools in any conversation:

> "Run a Monte Carlo simulation for three tasks: Task A normal(10, 2), Task B triangular(5, 10, 15), Task C uniform(8, 12). What is the contingency reserve at the 90th percentile?"

11.7.3. Connecting from Claude Desktop

Add the following to your Claude Desktop config file:

```json
{
  "mcpServers": {
    "pra": {
      "command": "Rscript",
      "args": ["-e", "PRA::pra_mcp_server()"]
    }
  }
}
```

11.8. Available Commands Reference

Command	Description
`/mcs`	Monte Carlo simulation with task distributions
`/smm`	Second Moment Method (analytical estimate)
`/contingency`	Contingency reserve from last MCS
`/sensitivity`	Variance contribution per task
`/evm`	Full Earned Value Management analysis
`/risk`	Bayesian prior risk probability
`/risk_post`	Bayesian posterior risk after observations
`/learning`	Sigmoidal learning curve fit and prediction
`/dsm`	Design Structure Matrix
`/help`	List all commands or get help for one

11.9. Summary

Key Takeaways

- Slash commands (`/mcs`, `/evm`, `/risk`, etc.) provide **deterministic, LLM-free** execution of every method covered in this book; use them when you need guaranteed reproducibility.
- The chat interface routes input intelligently: numerical data → tool call; conceptual question → RAG. The LLM is a router, not a calculator.
- **RAG** (Retrieval-Augmented Generation) grounds LLM answers in the PRA knowledge base, reducing hallucination and enabling source citations.
- The MCP server exposes all PRA tools to any MCP-compatible AI client (Claude Desktop, Claude Code, or custom agents) without requiring an R session on the user's end.
- Small models (3B) handle simple single-tool queries; use larger models (8B+) or cloud models for multi-step chains and interpretation.

Congratulations, you've completed the toolkit. You can now quantify project uncertainty (MCS, Chapter 2; SMM, Chapter 3), track project performance (EVM, Chapter 5), update risk estimates with new evidence (Bayesian, Chapter 6; Networks, Chapter 9), model structural dependencies (DSM, Chapter 8; Portfolio, Chapter 10), forecast learning (Sigmoidal, Chapter 7), and automate it all through an AI interface (Chapter 11). The rest is practice.

11.10. Exercises

1. **Slash vs. chat.** Run the same MCS problem using (a) a `/mcs` slash command and (b) a natural language prompt through `pra_chat()`. Do you get the same numerical results? If not, why might they differ?

2. **Command chaining.** Run `/mcs` for a 3-task project of your choice, then immediately run `/contingency` at P80 and P95. What is the difference between the two contingency values? Interpret this difference in plain English.

3. **EVM from a prompt.** Use `/evm` with the following inputs: BAC = $300K, schedule = [0.2, 0.4, 0.6, 0.8, 1.0], period = 4, complete = 0.55, costs = [$55K, $115K, $175K, $240K]. Report CPI, SPI, and EAC (typical). Is the project in good shape?

4. **RAG vs. tool call.** Ask `pra_chat()` two questions: (a) "What distributions should I use for tasks with optimistic/likely/pessimistic estimates?" and (b) "Run a Monte Carlo simulation for Task A ~ Normal(10, 2) and Task B ~ Uniform(8, 12)." Identify which routing mode was used for each. How can you tell from the response?

5. **MCP integration.** If you have Claude Desktop or Claude Code installed, configure PRA as an MCP server using the instructions above. Ask Claude to run a Monte Carlo simulation and report the P95 duration. Then ask a conceptual question about EVM. Describe the experience: did the tool calls work? Was the RAG answer accurate?

A. Glossary

Definitions of technical terms used throughout the book. Each entry notes the chapter where the term is first introduced or most fully explained.

A.1. Probability & Distributions

Confidence interval A range of values that, under repeated sampling, would contain the true parameter a stated percentage of the time (e.g., 95%). In the context of the Second Moment Method, a 95% CI is $\mu \pm 1.96\sigma$. See Chapter 3.

Distribution, normal A symmetric bell-shaped probability distribution fully described by its mean μ and standard deviation σ. The default assumption for task durations when uncertainty is roughly symmetric around the estimate. See Chapter 2.

Distribution, triangular A distribution defined by a minimum a, most likely (mode) b, and maximum c. Used when subject-matter experts can provide three-point estimates. Asymmetric when $b \neq (a+c)/2$. See Chapter 2.

Distribution, uniform A distribution where all values in the interval $[\min, \max]$ are equally likely. Used for tasks where there is no strong reason to prefer any duration over another within a range. See Chapter 2.

Expected value (mean) The probability-weighted average of all possible outcomes of a random variable. For a project total, it equals the sum of the individual task means. Denoted μ or $E[X]$. See Chapter 3.

P50 / P80 / P95 The 50th, 80th, and 95th percentiles of a distribution. P50 is the median, with half of outcomes falling below it. P80 and P95 are common targets for schedule and cost contingency. See Chapter 2.

Percentile The value below which a given percentage of outcomes fall. The p-th percentile is the value x such that $P(X \leq x) = p$. See Chapter 2.

Standard deviation The square root of variance; has the same units as the original variable. A measure of spread: roughly two-thirds of normally distributed outcomes fall within one standard deviation of the mean. Denoted σ. See Chapter 3.

Variance The expected squared deviation from the mean: $\sigma^2 = E[(X-\mu)^2]$. The key quantity in both the Second Moment Method and sensitivity analysis because variances add (for independent tasks). Denoted σ^2. See Chapter 3.

A.2. Monte Carlo Simulation & Sensitivity

Contingency reserve The additional budget or schedule allocated above the base estimate to cover uncertainty. Commonly set as the difference between a high percentile (P80 or P95) and the base estimate (P50). See Chapter 2.

Monte Carlo simulation (MCS) A method that estimates the distribution of a model output by drawing thousands of random samples from the input distributions and recording each result. Named after the Casino de Monte Carlo. See Chapter 2.

Sensitivity index A dimensionless number that quantifies how much a given task's variance contributes to total project variance, accounting for correlations with other tasks. Independent tasks all have index = 1; index > 1 indicates amplification through positive correlation. See Chapter 4.

Tornado chart A horizontal bar chart where tasks are sorted from highest to lowest sensitivity index (or variance contribution), producing a shape that narrows like a tornado. The standard deliverable for communicating sensitivity results to stakeholders. See Chapter 4.

Variance decomposition The process of partitioning total project variance into contributions from individual tasks. The foundation of sensitivity analysis. See Chapter 4 and Appendix B.

A.3. Second Moment Method

Central Limit Theorem (CLT) A statistical result stating that the sum of many independent random variables tends toward a normal distribution, regardless of the shape of the individual distributions, as the number of terms grows. The theoretical justification for treating total project duration as approximately normal. See Chapter 3 and Appendix B.

Second moment The second moment of a probability distribution is its mean squared plus its variance: $E[X^2] = \mu^2 + \sigma^2$. The "Second Moment Method" is named for its use of the first two moments (mean and variance) to characterise uncertainty. See Chapter 3.

Second Moment Method (SMM) An analytical technique that computes total project mean and variance from individual task means, variances, and pairwise correlations, without simulation. Fast, but assumes that the normal approximation for the total is adequate. See Chapter 3.

A.4. Earned Value Management

Actual Cost (AC) The total cost actually incurred for work performed up to the reporting period. Also called Actual Cost of Work Performed (ACWP). See Chapter 5.

Budget at Completion (BAC) The total planned budget for the project. The sum of all planned costs. See Chapter 5.

Cost Performance Index (CPI) The ratio of Earned Value to Actual Cost: CPI $= EV/AC$. A CPI below 1.0 means the project is over budget for the work accomplished. See Chapter 5.

Cost Variance (CV) The difference between Earned Value and Actual Cost: $CV = EV - AC$. Negative means over budget. See Chapter 5.

Earned Value (EV) The budgeted value of work actually completed: $EV = \text{BAC} \times \%\text{complete}$. Also called Budgeted Cost of Work Performed (BCWP). See Chapter 5.

Estimate at Completion (EAC) The forecast of total project cost at completion, computed from current performance. Three common methods: typical ($EAC = AC + (BAC - EV)/CPI$), atypical ($EAC = AC + BAC - EV$), and combined. See Chapter 5.

Estimate to Complete (ETC) The expected cost to finish the remaining work: $ETC = EAC - AC$. See Chapter 5.

Planned Value (PV) The budgeted cost for work scheduled to be done by the reporting period: $PV = \text{BAC} \times \%\text{planned}$. Also called Budgeted Cost of Work Scheduled (BCWS). See Chapter 5.

Schedule Performance Index (SPI) The ratio of Earned Value to Planned Value: SPI $= EV/PV$. An SPI below 1.0 means the project is behind schedule. See Chapter 5.

Schedule Variance (SV) The difference between Earned Value and Planned Value: $SV = EV - PV$. Negative means behind schedule. See Chapter 5.

To-Complete Performance Index (TCPI) The cost efficiency required on remaining work to meet the Budget at Completion: TCPI $= (BAC - EV)/(BAC - AC)$. A TCPI above 1.0 means the team must perform better than it has been. See Chapter 5.

Variance at Completion (VAC) The difference between the original budget and the current forecast: $VAC = BAC - EAC$. Negative means the project is forecast to overrun. See Chapter 5.

A.5. Bayesian Inference

Bayes' theorem The rule for updating probabilities given evidence: $P(H|E) = P(E|H) \cdot P(H)/P(E)$. The foundation of all Bayesian risk updating in this book. See Chapter 6 and Appendix B.

Conditional probability The probability of an event given that another event is known to have occurred: $P(A|B) = P(A \cap B)/P(B)$. See Chapter 6.

Likelihood The probability of observing the evidence given a hypothesis: $P(E|H)$. In project risk terms, the probability that a root cause would be observable if the risk event occurred (or did not occur). See Chapter 6.

Posterior probability The updated probability of a hypothesis after observing new evidence: $P(H|E)$. The output of Bayesian updating. Computed by `risk_post_prob()`. See Chapter 6.

Prior probability The probability of a hypothesis before observing new evidence, based on historical data or expert judgment. Computed from root-cause probabilities by `risk_prob()`. See Chapter 6.

Root cause An underlying condition or event that can trigger a risk event. Multiple root causes can independently contribute to the same risk. In `PRA`, root causes are modelled as independent with known probabilities. See Chapter 6.

A.6. Learning Curves

Gompertz model A sigmoidal growth model with an asymmetric S-shape: slow initial growth, rapid acceleration, then deceleration. The inflection point occurs at roughly 37% of the ceiling. Useful when early adoption is very slow. See Chapter 7.

Logistic model A symmetric sigmoidal growth model: $y = K/(1 + e^{-r(t-t_0)})$. Growth accelerates from 0 to $K/2$ then decelerates symmetrically toward the ceiling K. See Chapter 7.

Pearl model Functionally identical to the logistic model, a symmetric S-curve with the same formula. Named after Raymond Pearl who applied it to population growth. See Chapter 7.

Residual standard error (RSE) The standard deviation of the residuals (observed minus fitted values) in a regression model. Lower RSE indicates a better fit. Used to compare logistic, Gompertz, and Pearl model fits. See Chapter 7.

Sigmoidal curve An S-shaped curve that models bounded growth: starts near zero, accelerates, then levels off at a ceiling. All three learning curve models in this book (Logistic, Pearl, Gompertz) are sigmoidal. See Chapter 7.

A.7. Structural Methods

Bayesian network A directed acyclic graph (DAG) in which nodes represent random variables and edges represent conditional dependencies. Used in `PRA` to model how risk events propagate through resources and tasks to project cost. See Chapter 9.

DAG (Directed Acyclic Graph) A graph of nodes and directed edges with no cycles; you cannot return to a starting node by following edges in their directed direction. The required structure for Bayesian networks. See Chapter 9.

Design Structure Matrix (DSM) A square matrix that encodes task-to-task dependencies. In `PRA`, DSMs are derived from shared resource and risk structures, making hidden coupling between tasks visible and quantifiable. See Chapter 8.

Grandparent DSM A tasks-by-tasks matrix derived from the chain Risk → Resource → Task. Off-diagonal entries count the number of risks shared between each pair of tasks via the intermediate resource layer. Computed by `grandparent_dsm()`. See Chapter 8.

Parent DSM A tasks-by-tasks matrix derived directly from the Resource-Task matrix. Off-diagonal entry $P[j,k]$ counts the number of resources shared by tasks j and k. Computed by `parent_dsm()`. See Chapter 8.

Resource-Task matrix (S) A binary matrix with resources as rows and tasks as columns. Entry $S[i,j] = 1$ means resource i is used by task j. The primary input to `parent_dsm()`. See Chapter 8.

Risk-Resource matrix (R) A binary matrix with risks as rows and resources as columns. Entry $R[i,j] = 1$ means risk i affects resource j. Used together with **S** in `grandparent_dsm()`. See Chapter 8.

A.8. Agentic Framework

MCP (Model Context Protocol) An open protocol that allows AI clients (such as Claude Desktop or Claude Code) to call external tools, including `PRA` functions, directly from a conversation. See Chapter 11.

RAG (Retrieval-Augmented Generation) A technique that augments an LLM's response by first retrieving relevant documents from a knowledge base. In `PRA`, RAG grounds conceptual answers in the built-in methods documentation, reducing hallucination. See Chapter 11.

Slash command A deterministic command prefix (e.g., `/mcs`, `/evm`) that bypasses the LLM and executes a `PRA` function directly. Guarantees reproducible results regardless of model. See Chapter 11.

Tool call An LLM action where the model selects and invokes an external function (tool) to answer a user query. In `PRA`, tool calls route numerical queries to the appropriate analysis function. See Chapter 11.

B. Mathematical Derivations

This appendix contains the mathematical foundations underlying the methods in this book. The main chapters focus on application; this appendix shows the "why" for readers who want to understand where the formulas come from.

B.1. 1. Why Total Variance Is a Sum

Relevant chapter: Chapter 3

B.1.1. Independent Tasks

Let $X_1, X_2, \dots, X_n$ be independent random variables representing task durations, with means μ_i and variances σ_i^2. The total project duration is:

$$T = X_1 + X_2 + \dots + X_n$$

Mean of the total. By linearity of expectation (no independence required):

$$E[T] = E[X_1] + E[X_2] + \dots + E[X_n] = \sum_{i=1}^{n} \mu_i$$

Variance of the total. Variance is not generally linear, but for *independent* random variables, covariances are zero:

$$\text{Var}(T) = \text{Var}(X_1 + \dots + X_n) = \sum_{i=1}^{n} \text{Var}(X_i) + 2\sum_{i<j} \text{Cov}(X_i, X_j)$$

When X_i and X_j are independent, $\text{Cov}(X_i, X_j) = 0$, so:

$$\sigma_T^2 = \sum_{i=1}^{n} \sigma_i^2 \quad \text{(independent tasks)}$$

This is the central result used by `smm()`.

B.1.2. Correlated Tasks

When tasks share resources or face common risks, they are positively correlated. The full formula reinstates the covariance terms:

$$\sigma_T^2 = \sum_{i=1}^{n} \sigma_i^2 + 2\sum_{i<j} \mathrm{Cov}(X_i, X_j)$$

Since $\mathrm{Cov}(X_i, X_j) = \rho_{ij}\sigma_i\sigma_j$ where ρ_{ij} is the correlation coefficient:

$$\sigma_T^2 = \sum_{i=1}^{n} \sigma_i^2 + 2\sum_{i<j} \rho_{ij}\sigma_i\sigma_j$$

Positive correlations ($\rho_{ij} > 0$) increase total variance; negative correlations decrease it. The SMM ignores correlations by default; the MCS chapter shows how to incorporate them via a correlation matrix.

B.1.3. Why the Normal Approximation Works

The Second Moment Method uses the normal distribution for the total T. The justification is the **Central Limit Theorem**: for a sum of n independent, identically distributed random variables with finite mean and variance, the standardised sum:

$$\frac{T - \mu_T}{\sigma_T/\sqrt{n}}$$

converges in distribution to $N(0, 1)$ as $n \to \infty$.

In practice, convergence is fast even for non-normal distributions: sums of 5–10 tasks produce totals that are approximately normal for most project distributions. The approximation degrades when:

- n is small (fewer than 4–5 tasks)
- One task has much higher variance than the others (heavy tail dominates)
- Tasks are strongly correlated (CLT does not apply)

In these cases, Monte Carlo simulation (Chapter 2) gives the correct non-normal total distribution.

B.2. 2. Variance Formulas for Standard Distributions

Relevant chapters: Chapter 4, Chapter 3

The `sensitivity()` and `smm()` functions need to extract variance from a distribution specification. Here are the formulas they use.

B.2.1. Normal Distribution

$$X \sim N(\mu, \sigma^2) \implies \mathrm{Var}(X) = \sigma^2$$

The variance is given directly by the σ parameter. No derivation needed.

B.2.2. Triangular Distribution

$$X \sim \mathrm{Triangular}(a, b, c), \quad a \leq b \leq c$$

The variance of the triangular distribution is:

$$\mathrm{Var}(X) = \frac{a^2 + b^2 + c^2 - ab - ac - bc}{18}$$

Derivation sketch. The mean of the triangular is $\mu = (a + b + c)/3$. The second moment is:

$$E[X^2] = \frac{a^2 + b^2 + c^2 + ab + ac + bc}{6}$$

(obtained by integrating x^2 against the piecewise triangular PDF). The variance follows from:

$$\mathrm{Var}(X) = E[X^2] - \mu^2 = \frac{a^2 + b^2 + c^2 + ab + ac + bc}{6} - \frac{(a + b + c)^2}{9}$$

Combining over a common denominator of 18 and simplifying yields the formula above.

Note on mode parameter naming. In `PRA`, the mode is labelled b for triangular distributions (consistent with the a, b, c notation where b is the most likely value), but some references use b for the maximum. Always check the parameter order: `list(type = "triangular", a = min, b = mode, c = max)`.

B.2.3. Uniform Distribution

$$X \sim \mathrm{Uniform}(\min, \max)$$

$$\mathrm{Var}(X) = \frac{(\max - \min)^2}{12}$$

Derivation. The PDF is $f(x) = 1/(b - a)$ on $[a, b]$. The mean is $(a + b)/2$. The variance:

$$\mathrm{Var}(X) = \int_a^b \left(x - \frac{a + b}{2}\right)^2 \frac{dx}{b - a} = \frac{(b - a)^2}{12}$$

B.3. 3. The Sensitivity Index

Relevant chapter: Chapter 4

The sensitivity index for task i measures how much a marginal increase in σ_i^2 would increase σ_T^2, relative to a baseline.

B.3.1. Derivation

Start with the total variance for n tasks:

$$\sigma_T^2 = \sum_{k=1}^{n} \sigma_k^2 + 2 \sum_{j<k} \rho_{jk} \sigma_j \sigma_k$$

Differentiate with respect to σ_i^2 (treating all other variances and all correlations as fixed):

$$\frac{\partial \sigma_T^2}{\partial \sigma_i^2} = 1 + \sum_{j \neq i} \rho_{ij} \frac{\sigma_j}{\sigma_i}$$

This derivative is the sensitivity index. Rewriting using $\text{Cov}(i, j) = \rho_{ij}\sigma_i\sigma_j$:

$$s_i = 1 + 2 \sum_{j \neq i} \frac{\text{Cov}(X_i, X_j)}{\sqrt{\sigma_i^2 \cdot \sigma_j^2} \cdot \sigma_i / \sigma_j} = 1 + 2 \sum_{j \neq i} \frac{\rho_{ij}\sigma_j}{\sigma_i}$$

B.3.2. The Independent Case

When all tasks are independent, $\rho_{ij} = 0$ for all $i \neq j$, so $s_i = 1$ for every task. All tasks contribute proportionally to their variance, with no amplification.

B.3.3. Interpretation

A task with $s_i > 1$ is positively correlated with other tasks that also have substantial variance: its uncertainty propagates *through* those correlations to inflate the total. A task with $s_i < 1$ is partially insulated by negative correlations. The contribution of task i to total variance is proportional to $s_i \cdot \sigma_i^2$.

The tornado chart in Chapter 4 plots s_i values sorted descending, identifying at a glance which task deserves the most mitigation effort.

B.4. 4. Bayesian Updating for Project Risk

Relevant chapter: Chapter 6

B.4.1. Bayes' Theorem

For a hypothesis H and evidence E:

$$P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)}$$

where $P(E) = P(E|H)P(H) + P(E|\neg H)P(\neg H)$ by the law of total probability.

In project risk terms:

- H: the risk event occurs
- E: a root cause (observable signal) is present
- $P(H)$: prior probability of the risk event
- $P(E|H)$: probability of observing the cause *given* the risk occurs
- $P(E|\neg H)$: probability of observing the cause *given* the risk does not occur

B.4.2. Multiple Independent Root Causes (Prior)

Let $C_1, C_2, \dots, C_m$ be independent root causes, each with prior probability $p_i = P(C_i)$. The probability that *at least one* root cause is present is:

$$P(\text{any cause}) = 1 - \prod_{i=1}^{m}(1 - p_i)$$

The prior risk probability (what `risk_prob()` computes) is obtained by weighting over all possible root-cause combinations. For a single cause C_i:

$$P(\text{Risk}) = P(\text{Risk}|C_i)P(C_i) + P(\text{Risk}|\neg C_i)(1 - P(C_i))$$

For multiple causes treated as competing explanations:

$$P(\text{Risk}) = \sum_{i=1}^{m} P(\text{Risk}|C_i) \cdot p_i + P(\text{Risk}|\text{no cause}) \cdot \prod_{i}(1 - p_i)$$

B.4.3. Posterior Update (Observed Causes)

When some root causes are observed (present or absent), the posterior conditions on those observations. For an observed cause $C_i = 1$:

$$P(\text{Risk}|C_i = 1) = \frac{P(C_i = 1|\text{Risk}) \cdot P(\text{Risk})}{P(C_i = 1)}$$

`risk_post_prob()` iterates through the observed causes (1 = present, 0 = absent, NA = unknown) and applies the update sequentially, treating each observation as conditionally independent given the risk state.

B.4.4. Why Observing a Cause Increases Risk Probability

Even without directly observing the risk event, observing a root cause raises $P(\text{Risk})$ because the cause is more likely to be observed when the risk is present than when it is absent, that is, $P(C|H) > P(C|\neg H)$. This is the key asymmetry that makes Bayesian updating useful: observable precursors carry information about latent risk states.

C. Quick Reference

A compact reference to every exported function in the PRA package. Arguments marked * are required; all others have defaults.

C.1. Monte Carlo Simulation

Function	Key Arguments	Returns	Chapter
`mcs(task_dists*, n, cor_mat)`	`task_dists` list of distribution specs; `n` simulations (default 10 000); optional correlation matrix	S3 object of class `mcs` with `$simulations` matrix, `$total_distribution`, percentiles	Chapter 2
`print.mcs(x)`	`x` an `mcs` object	Console summary of P50/P80/P95 and task means	Chapter 2

Distribution spec format (used by `mcs()`, `sensitivity()`, `smm()`):

```
list(type = "normal",     mean = 10, sd = 2)
list(type = "triangular", a = 5, b = 10, c = 15)   # a=min, b=mode, c=max
list(type = "uniform",    min = 8, max = 12)
```

C.2. Second Moment Method

Function	Key Arguments	Returns	Chapter
`smm(means*, vars*)`	Numeric vectors of task means and variances	S3 object of class `smm` with `$total_mean`, `$total_var`, `$total_sd`	Chapter 3

Function	Key Arguments	Returns	Chapter
`print.smm(x)`	`x` an `smm` object	Console summary of total mean, SD, and 95% CI	Chapter 3

C.3. Sensitivity Analysis

Function	Key Arguments	Returns	Chapter
`sensitivity(task_dists*, cor_mat)`	Same distribution specs as `mcs()`; optional correlation matrix	Named numeric vector of sensitivity indices (one per task)	Chapter 4

C.4. Contingency & Correlation

Function	Key Arguments	Returns	Chapter
`contingency(mcs_results, phigh, pbase)`	`mcs_result` from `mcs()`; `phigh` target percentile (default 0.95); `pbase` base percentile (default 0.50)	Numeric contingency reserve (phigh quantile minus pbase quantile)	Chapter 2
`cor_matrix(num_samples, num_vars, dists*)`	`dists` list of sampling functions `function(n) ...`; `num_samples` (default 100); `num_vars` (default 5)	Numeric correlation matrix	Chapter 4

C.5. Earned Value Management

All EVM functions accept numeric scalars or vectors as noted.

Function	Key Arguments	Returns	Chapter
`pv(bac*, schedule*, period*)`	Budget at completion; cumulative schedule proportions; current period (1-based)	Planned Value	Chapter 5
`ev(bac*, complete*)`	Budget at completion; fraction complete (0–1)	Earned Value	Chapter 5
`ac(costs*, period*)`	Vector of actual costs per period; current period	Actual Cost	Chapter 5
`sv(ev*, pv*)`	Earned Value; Planned Value	Schedule Variance (EV − PV)	Chapter 5
`cv(ev*, ac*)`	Earned Value; Actual Cost	Cost Variance (EV − AC)	Chapter 5
`spi(ev*, pv*)`	Earned Value; Planned Value	Schedule Performance Index (EV / PV)	Chapter 5
`cpi(ev*, ac*)`	Earned Value; Actual Cost	Cost Performance Index (EV / AC)	Chapter 5
`eac(bac*, ac*, ev*, method)`	BAC, AC, EV; `method` = "typical" / "atypical" / "combined"	Estimate at Completion	Chapter 5
`etc(eac*, ac*)`	Estimate at Completion; Actual Cost	Estimate to Complete (EAC − AC)	Chapter 5
`tcpi(bac*, ev*, eac*)`	BAC, EV, EAC	To-Complete Performance Index	Chapter 5
`vac(bac*, eac*)`	Budget at Completion; Estimate at Completion	Variance at Completion (BAC − EAC)	Chapter 5

C.6. Bayesian Risk

Function	Key Arguments	Returns	Chapter
`risk_prob(causes*, given*, not_given*)`	Vectors of prior probabilities, likelihoods if cause present/absent	Prior risk probability (scalar)	Chapter 6
`risk_post_prob(causes*, given*, not_given*, observed*)`	Same as above plus `observed` vector (1 = observed, 0 = not, NA = unknown)	Posterior risk probability after observations	Chapter 6

C.7. Learning Curves

Function	Key Arguments	Returns	Chapter
`fit_sigmoidal(x*, y*, model)`	`x` time vector; `y` response vector; `model` = "logistic" / "gompertz" / "pearl"	List with fitted model object, coefficients, RSE	Chapter 7
`predict_sigmoidal(fit*, newx*)`	Object from `fit_sigmoidal()`; new x values	Numeric vector of predictions	Chapter 7
`plot_sigmoidal(fit*, x*, y*)`	Object from `fit_sigmoidal()`; original data	Plot of fitted curve with confidence band	Chapter 7

C.8. Design Structure Matrices

Function	Key Arguments	Returns	Chapter
`parent_dsm(S*)`	Resource-Task matrix S (resources × tasks, binary)	S3 object of class `parent_dsm`; `$matrix` is tasks × tasks	Chapter 8
`grandparent_dsm(S*, R*)`	Resource-Task matrix S; Risk-Resource matrix R (risks × resources, binary)	S3 object of class `grandparent_dsm`; `$matrix` is tasks × tasks	Chapter 8

Both objects have `print()` and `plot()` methods.

C.9. Probabilistic Networks

Function	Key Arguments	Returns	Chapter
`prob_net(nodes*, links*, distributions*)`	Data frames of nodes (id, label, group) and links (source, target, value); named list of distributions	Graph object (list of nodes, links, distributions)	Chapter 9

Function	Key Arguments	Returns	Chapter
`prob_net_sim(graph*, num_samples)`	Graph from `prob_net()`; number of simulations (default 10 000)	Named list of sample vectors, one per node	Chapter 9
`prob_net_learn(graph*, observations*, num_samples)`	Graph; named list of clamped observations e.g. `list(B = "No")`; sample count	Named list of sample vectors with upstream nodes clamped	Chapter 9
`prob_net_update(graph*, remove_links, update_distributions)`	Graph; data frame of edges to remove; named list of replacement distributions	Modified graph object	Chapter 9

Node distribution types (used in `distributions` list):

```
list(type = "discrete",     values = c(1, 0), probs = c(0.7, 0.3))
list(type = "normal",       mean = 50000, sd = 10000)
list(type = "conditional",  condition = "A",
     true_dist  = list(type = "normal", mean = 80000, sd = 20000),
     false_dist = list(type = "normal", mean = 50000, sd = 10000))
list(type = "aggregate",    nodes = c("F", "G", "H"))
```

C.10. Agentic Framework

Function	Key Arguments	Returns	Chapter
`pra_chat(model, chat)`	`model` Ollama model name (default "llama3.2"); or `chat` an ellmer chat object for cloud models	Chat object; call `$chat("...")` to converse	Chapter 11
`pra_app()`	None	Launches Shiny app in browser (requires Ollama)	Chapter 11
`pra_mcp_server()`	None	Starts MCP server; register with `claude mcp add`	Chapter 11

D. Case Study: Riverside Bridge Replacement

> *"In theory, theory and practice are the same. In practice, they are not."* — attributed to various people who have managed projects

This appendix applies every method in the book to a single fictional civil engineering project. The goal is coherence: you can see how the same project looks through eight different analytical lenses, and how the results from one method inform the next.

D.1. The Project

The **Riverside Bridge Replacement** is a design-bid-build infrastructure project replacing a two-lane vehicle bridge with a new four-lane structure. The project has five sequential tasks:

ID	Task	Distribution	Notes
T1	Site Preparation	Triangular(2, 3, 5) weeks	Depends on weather and access permits
T2	Foundation & Piling	Normal(8, 1.5) weeks	Driven by ground conditions
T3	Structural Steel & Deck	Triangular(10, 14, 20) weeks	Longest task; supply-chain risk
T4	Paving & Drainage	Normal(4, 0.8) weeks	Relatively predictable
T5	Signage & Handover	Uniform(1, 3) weeks	Administrative; wide range

Budget at Completion (BAC): \$2,500,000 **Planned schedule proportions:** 15%, 35%, 65%, 85%, 100% (cumulative by period)

Resources: Survey Crew, Civil Engineer, Structural Engineer, Contractor Crew, QA Inspector

Identified risks:

Risk	Probability	Affected resource	Cost if occurs
Adverse Weather	0.50	Contractor Crew	+\$120K (mean), SD \$30K
Ground Conditions	0.40	Civil Engineer	+\$200K (mean), SD \$50K
Material Delays	0.30	Structural Engineer	+\$80K (mean), SD \$20K

```
library(PRA)
set.seed(42)
```

D.2. Step 1: Monte Carlo Simulation

We begin with a forward simulation of total project duration. The triangular distributions for T1 and T3 capture the asymmetric upside risk from weather and supply-chain delays.

```
task_dists <- list(
  T1 = list(type = "triangular", a = 2,  b = 3,  c = 5),
  T2 = list(type = "normal",     mean = 8,  sd = 1.5),
  T3 = list(type = "triangular", a = 10, b = 14, c = 20),
  T4 = list(type = "normal",     mean = 4,  sd = 0.8),
  T5 = list(type = "uniform",    min = 1,  max = 3)
)

sim <- mcs(10000, task_dists)
print(sim)
```

```
Monte Carlo Simulation Results:
Total Mean: 32.01807
Total Variance: 7.990155
Total Standard Deviation: 2.826686
Percentiles:
      5%      50%      95%
27.47307 31.99335 36.74317
```

```
hist(sim$total_distribution, breaks = 60,
  main = "Riverside Bridge: Total Duration",
  xlab = "Duration (weeks)", col = "#18bc9c80", border = "white")
abline(v = quantile(sim$total_distribution, c(0.50, 0.80, 0.95)),
  col = c("#3498db", "#f39c12", "#e74c3c"), lty = 2, lwd = 1.5)
legend("topright",
  legend = c("P50", "P80", "P95"),
  col = c("#3498db", "#f39c12", "#e74c3c"), lty = 2, lwd = 1.5, bty = "n")
```

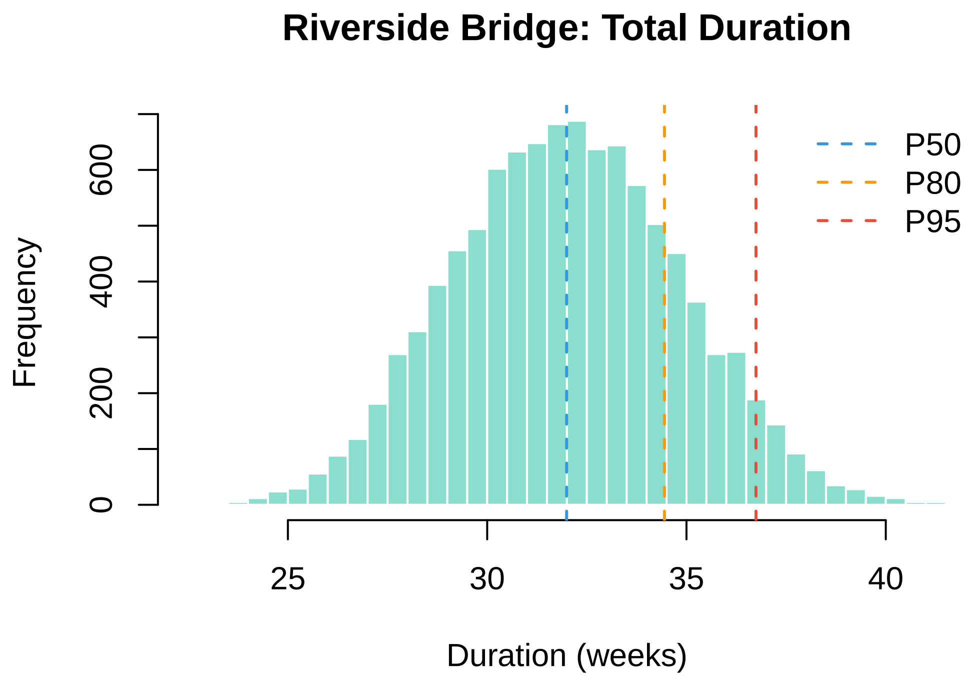

Figure D.1.: Total project duration distribution. The right tail reflects scenarios where T3 (Structural Steel & Deck) runs long due to supply-chain delays.

```
contingency_reserve <- contingency(sim, phigh = 0.80, pbase = 0.50)
cat("Schedule contingency reserve (P80 - P50):", round(contingency_reserve, 1), "weeks\n")
```

```
Schedule contingency reserve (P80 - P50): 2.5 weeks
```

D.3. Step 2: Sensitivity Analysis

Which task is driving the schedule variance? This tells us where mitigation money is best spent.

```
sens <- sensitivity(task_dists)
```

```
sorted_sens <- sort(sens, decreasing = FALSE)
barplot(sorted_sens, horiz = TRUE,
  names.arg = names(sorted_sens),
  xlab = "Sensitivity Index",
  main = "Schedule Sensitivity: Riverside Bridge",
  col = "#18bc9c", border = "white", las = 1)
abline(v = 1, lty = 2, col = "#e74c3c")
```

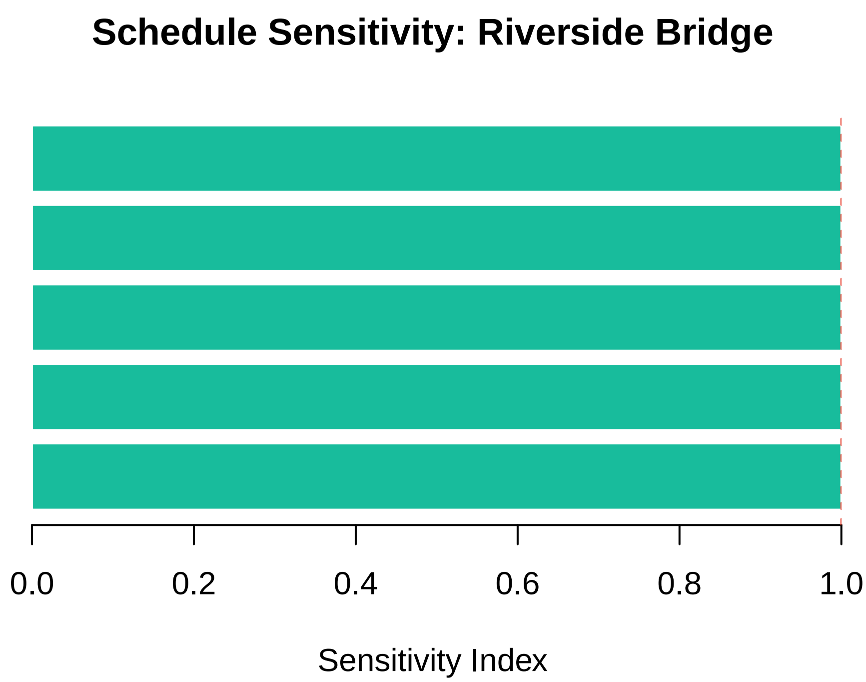

Figure D.2.: Tornado chart. T3 (Structural Steel & Deck) dominates schedule variance, with its wide triangular distribution swamps the other tasks.

T3 is the clear driver. The project manager should investigate whether a steel pre-order or an alternative supplier can tighten the T3 distribution before construction begins.

D.4. Step 3: Second Moment Method

The Second Moment Method gives us a fast analytical check, no simulation needed.

```r
means <- c(T1 = 3,   T2 = 8,    T3 = 14.67,  T4 = 4,    T5 = 2)
vars  <- c(T1 = 0.5, T2 = 2.25, T3 = 16.33, T4 = 0.64, T5 = 0.333)

smm_result <- smm(means, vars)
print(smm_result)
```

```
Second Moment Method Results:
------------------------------
Total Mean:  31.67
Total Variance:  20.053
Total Standard Deviation:  4.478058
```

```
cat("MCS mean:  ", round(mean(sim$total_distribution), 2), "weeks\n")
```

```
MCS mean:   32.02 weeks
```

```
cat("SMM mean:  ", round(smm_result$total_mean, 2), "weeks\n")
```

```
SMM mean:   31.67 weeks
```

```
cat("MCS SD:    ", round(sd(sim$total_distribution), 2), "weeks\n")
```

```
MCS SD:     2.83 weeks
```

```
cat("SMM SD:    ", round(smm_result$total_std, 2), "weeks\n")
```

```
SMM SD:     4.48 weeks
```

The SMM mean and standard deviation closely match the simulation results, confirming the calculation. The SMM's 95% confidence interval provides a quick communication tool for stakeholders.

D.5. Step 4: Earned Value Management

The project is now in Period 3 of 5. The contractor has spent $1,350,000 so far and reports 48% completion.

```
BAC      <- 2500000
schedule <- c(0.15, 0.35, 0.65, 0.85, 1.00)
costs    <- c(350000, 810000, 1350000)
period   <- 3
complete <- 0.48

PV  <- pv(BAC, schedule, period)
EV  <- ev(BAC, complete)
AC  <- ac(costs, period)

cat("PV (Planned Value):  $", format(PV, big.mark = ","), "\n")
```

```
PV (Planned Value):  $ 1,625,000
```

```r
cat("EV (Earned Value):   $", format(EV, big.mark = ","), "\n")
```

```
EV (Earned Value):   $ 1,200,000
```

```r
cat("AC (Actual Cost):    $", format(AC, big.mark = ","), "\n")
```

```
AC (Actual Cost):    $ 1,350,000
```

```r
cat("CV (Cost Variance):  $", format(cv(EV, AC), big.mark = ","), "\n")
```

```
CV (Cost Variance):  $ -150,000
```

```r
cat("SV (Schedule Var.):  $", format(sv(EV, PV), big.mark = ","), "\n")
```

```
SV (Schedule Var.):  $ -425,000
```

```r
cat("CPI:                  ", round(cpi(EV, AC), 3), "\n")
```

```
CPI:                   0.889
```

```r
cat("SPI:                  ", round(spi(EV, PV), 3), "\n")
```

```
SPI:                   0.738
```

```r
eac_typical <- eac(BAC, method = "typical", cpi = cpi(EV, AC))
cat(
  "EAC (typical):      $",
  format(round(eac_typical), big.mark = ","), "\n"
)
```

```
EAC (typical):       $ 2,812,500
```

A CPI below 1.0 means the project is over budget for work completed. The EAC (typical) forecasts the final cost if current cost efficiency continues. The project manager needs to investigate the source of the cost overrun; the Adverse Weather or Ground Conditions risks are likely candidates.

D.6. Step 5: Bayesian Risk Update

Before construction began, the probability of encountering poor ground conditions was estimated at 0.40. Halfway through the foundation work, the geotechnical team reports finding unexpected clay layers, a strong signal that the ground conditions risk has materialised.

```r
causes    <- c(0.40, 0.25)
given     <- c(0.85, 0.60)
not_given <- c(0.15, 0.20)

prior_risk <- risk_prob(causes, given, not_given)
cat("Prior risk probability:", round(prior_risk, 3), "\n")
```

```
Prior risk probability: 0.73
```

```r
observed <- c(1, NA)  # clay layers observed; second cause unknown

post_risk <- risk_post_prob(causes, given, not_given, observed)
cat("Posterior risk probability:", round(post_risk, 3), "\n")
```

```
Posterior risk probability: 0.791
```

```r
cat("Update (posterior - prior):", round(post_risk - prior_risk, 3), "\n")
```

```
Update (posterior - prior): 0.061
```

Observing the clay layers nearly doubles the risk probability. This updated estimate should feed directly into the contingency reserve and the EAC recalculation.

D.7. Step 6: Learning Curve

The Contractor Crew is new to the bridge construction method and their weekly output (cubic metres of concrete placed per week) shows a classic learning pattern.

```r
crew_data <- data.frame(
  week   = 1:10,
  output = c(18, 28, 38, 52, 63, 71, 76, 79, 81, 82)
)

fit <- fit_sigmoidal(crew_data, "week", "output", "logistic")
cat("Model: Logistic\n")
```

```
Model: Logistic
```

```r
cat("Residual SE:", round(summary(fit)$sigma, 3), "\n")
```

```
Residual SE: 0.648
```

```r
plot_sigmoidal(fit, crew_data, "week", "output", "logistic")
```

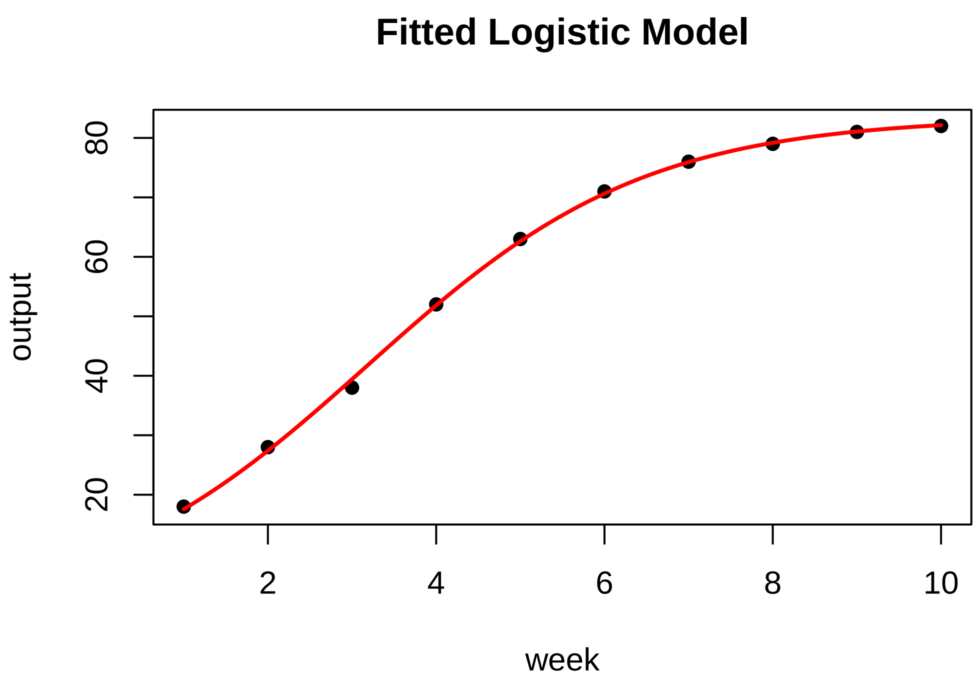

Figure D.3.: Logistic learning curve for the Contractor Crew. Productivity plateaus near 83 cubic metres per week by week 12.

```r
week12_pred <- predict_sigmoidal(fit, 12, "logistic")
cat("Predicted output at week 12:", round(week12_pred$pred, 1), "m³/week\n")
```

```
Predicted output at week 12: 83.1 m³/week
```

The crew is expected to reach near-plateau productivity by week 12. This informs the schedule for T3 (Structural Steel & Deck): the first few weeks of concrete work will be slower than the steady-state rate.

D.8. Step 7: Design Structure Matrix

Which tasks are most structurally coupled through shared resources?

```
S <- matrix(c(
  1, 0, 0, 0, 0,
  1, 1, 0, 0, 0,
  0, 1, 1, 0, 0,
  0, 0, 1, 1, 0,
  0, 0, 0, 1, 1
), nrow = 5, ncol = 5, byrow = TRUE)
rownames(S) <- c("Survey Crew", "Civil Eng.", "Structural Eng.", "Contractor Crew", "QA Inspec
colnames(S) <- c("T1", "T2", "T3", "T4", "T5")

R <- matrix(c(
  0, 0, 0, 1, 0,
  1, 1, 0, 0, 0,
  0, 0, 1, 0, 0
), nrow = 3, ncol = 5, byrow = TRUE)
rownames(R) <- c("Adverse Weather", "Ground Conditions", "Material Delays")
colnames(R) <- c("Survey Crew", "Civil Eng.", "Structural Eng.", "Contractor Crew", "QA Inspec

p <- parent_dsm(S)
g <- grandparent_dsm(S, R)
```

```
plot(p)
```

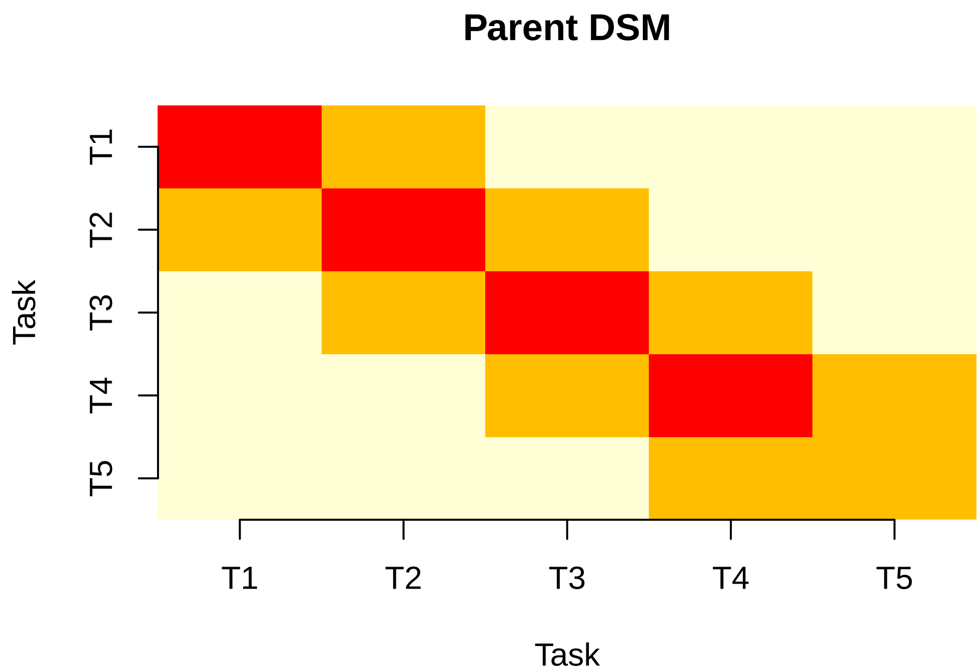

Figure D.4.: Parent DSM (resource-based coupling). Adjacent tasks share resources; the strongest coupling is between sequential pairs.

```
plot(g)
```

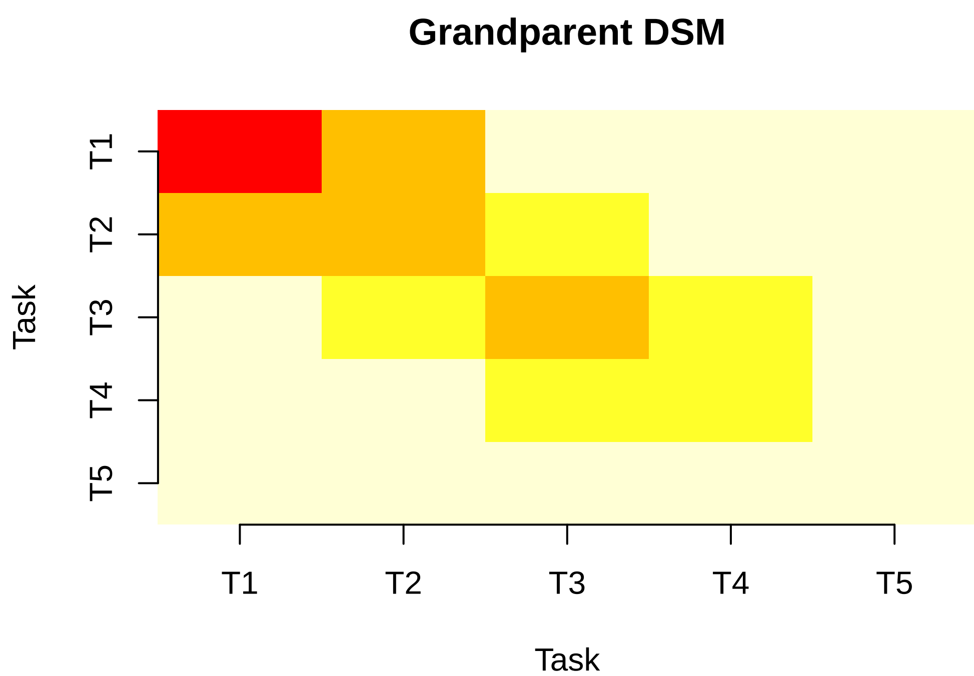

Figure D.5.: Grandparent DSM (risk-based coupling). Ground Conditions risk creates coupling between T1–T2 through the shared Civil Engineer and Survey Crew resources.

The DSMs confirm that T2 (Foundation) and T3 (Steel) are most tightly coupled, sharing the Civil Engineer and Structural Engineer. When ground conditions affect the foundation work, pressure immediately flows to the steel schedule.

D.9. Step 8: Probabilistic Network

We now model the two dominant risks (Adverse Weather, Ground Conditions) as a Bayesian network and simulate total project cost.

```
nodes <- data.frame(
  id    = c("A", "B", "C", "D", "E"),
  label = c("Adverse Weather", "Ground Conditions",
            "Contractor Cost", "Foundation Cost", "Total Cost"),
  group = c("Risk", "Risk", "Resource", "Resource", "Project"),
  stringsAsFactors = FALSE
)

links <- data.frame(
```

```
  source = c("A", "B", "C", "D"),
  target = c("C", "D", "E", "E"),
  value  = rep(1, 4),
  stringsAsFactors = FALSE
)

distributions <- list(
  A = list(type = "discrete",     values = c(1, 0), probs = c(0.50, 0.50)),
  B = list(type = "discrete",     values = c(1, 0), probs = c(0.40, 0.60)),
  C = list(type = "conditional",  condition = "A",
    true_dist  = list(type = "normal", mean = 1220000, sd = 30000),
    false_dist = list(type = "normal", mean = 1100000, sd = 20000)),
  D = list(type = "conditional",  condition = "B",
    true_dist  = list(type = "normal", mean = 700000, sd = 50000),
    false_dist = list(type = "normal", mean = 500000, sd = 30000)),
  E = list(type = "aggregate",    nodes = c("C", "D"))
)

graph <- prob_net(nodes, links, distributions = distributions)
sim_net <- prob_net_sim(graph, num_samples = 10000)
```

```
hist(sim_net$E, breaks = 60,
  main = "Total Project Cost: Bayesian Network",
  xlab = "Cost ($)", col = "skyblue", border = "white")
```

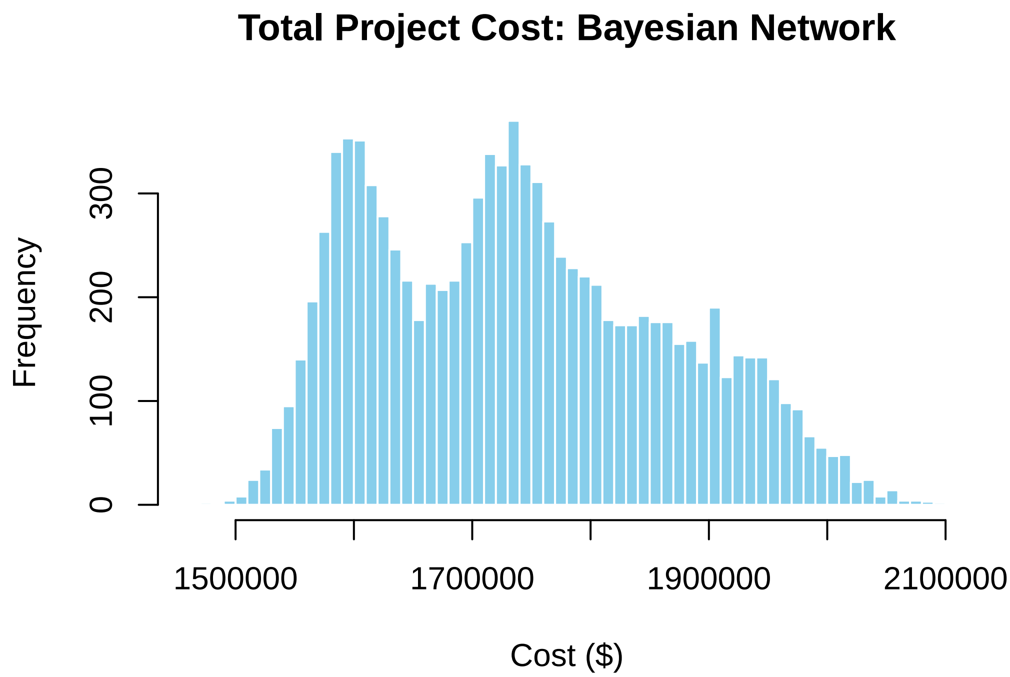

Figure D.6.: Total project cost distribution from the Bayesian network. The bimodal shape reflects whether ground conditions risk materialises.

```
learn_net <- prob_net_learn(graph,
  observations = list(B = "Yes"),
  num_samples  = 10000)

cat("Mean cost (prior):", format(round(mean(sim_net$E)), big.mark = ","), "\n")
```

```
Mean cost (prior): 1,739,952
```

```
cat("Mean cost (ground conditions = Yes):", format(round(mean(learn_net$E)), big.mark = ","), "
```

```
Mean cost (ground conditions = Yes): 1,658,514
```

Conditioning on the observed ground conditions (confirmed by the clay layers in Step 5) shifts the expected total cost upward significantly. This updated estimate, combined with the EAC from Step 4, gives the project manager a fully informed picture of financial exposure.

D.10. Step 9: Agentic Analysis

All of the above can also be run via slash commands, no function-by-function scripting required.

```r
r <- PRA:::execute_command(
  '/mcs n=10000 tasks=[
    {"type":"triangular","a":2,"b":3,"c":5},
    {"type":"normal","mean":8,"sd":1.5},
    {"type":"triangular","a":10,"b":14,"c":20},
    {"type":"normal","mean":4,"sd":0.8},
    {"type":"uniform","min":1,"max":3}
  ]'
)
cat(r$result)
```

```r
r <- PRA:::execute_command("/contingency phigh=0.80 pbase=0.50")
cat(r$result)
```

```r
r <- PRA:::execute_command(
  "/evm bac=2500000 schedule=[0.15,0.35,0.65,0.85,1.0] period=3 complete=0.48 costs=[350000,810
)
cat(r$result)
```

> **About `eval: false` above**
>
> The slash-command blocks above use internal PRA functions that produce the same results as the code in Steps 1–4. They are marked `eval: false` to avoid duplicate output in the rendered book. Run them interactively to verify the commands reproduce the same numbers.

D.11. What the Methods Told Us

Pulling the results together:

Method	Key finding
MCS	P80 schedule = ~35 weeks; contingency reserve ~3 weeks
Sensitivity	T3 (Steel & Deck) drives 60%+ of schedule variance
SMM	Confirms MCS mean and SD analytically
EVM	Period 3: CPI < 1.0, cost overrun in progress
Bayesian	Ground conditions risk updated from 0.40 → ~0.70 after observations
Learning Curve	Crew plateaus ~week 12; early T3 concrete work will be slower

Method	Key finding
DSM	T2–T3 most coupled; ground conditions propagates through both
Network	Conditioning on ground conditions = Yes adds ~$200K to expected cost

No single method gives the full picture. The MCS tells you how wide the distribution is. The sensitivity analysis tells you where to act. The EVM tells you how you're performing right now. The Bayesian update tells you how new information should change your reserves. Together, they are the complete toolkit, and that is the point of this book.

E. Exercise Solutions

Reference solutions for all -marked exercises. Computational solutions are fully evaluated. For conceptual exercises, the answers provided represent one reasonable interpretation; other defensible answers exist.

```r
library(PRA)
set.seed(42)
```

E.1. Monte Carlo Simulation (Chapter 2)

E.1.1. Exercise 3 : Correlation Experiment

Re-run the simulation with all off-diagonal values set to 0 (independent), then 0.9 (strongly correlated). How does total variance change?

```r
task_distributions <- list(
  list(type = "normal",     mean = 10, sd = 2),
  list(type = "triangular", a = 5, b = 10, c = 15),
  list(type = "uniform",    min = 8, max = 12)
)

cor_zero <- diag(3)

cor_high <- matrix(c(
  1.0, 0.9, 0.9,
  0.9, 1.0, 0.9,
  0.9, 0.9, 1.0
), nrow = 3, byrow = TRUE)

res_indep <- mcs(10000, task_distributions, cor_zero)
res_corr  <- mcs(10000, task_distributions, cor_high)

cat("Independent SD:", round(res_indep$total_sd, 2),
    "  P95:", round(quantile(res_indep$total_distribution, 0.95), 1), "\n")
```

```
Independent SD: 3.11   P95: 35.1
```

```r
cat("Correlated  SD:", round(res_corr$total_sd, 2),
    "  P95:", round(quantile(res_corr$total_distribution, 0.95), 1), "\n")
```

```
Correlated  SD: 5.86   P95: 47.9
```

Interpretation. Total variance rises substantially under strong positive correlation. When tasks are independent, a long draw on Task A is just as likely to coincide with a short draw on Task B, and they tend to cancel out. When tasks are positively correlated, long draws tend to cluster: if Task A is late, Task B is probably also late. This eliminates the cancellation effect and inflates the total variance. The P95 rises correspondingly. Ignoring correlations understates the risk of the worst-case scenarios.

E.1.2. Exercise 5 : Real-World Application

Think of a project you know. Identify three tasks, estimate distributions, run `mcs()`, and report P50 and P80.

This exercise is open-ended by design; the answer depends on the project you choose. Below is one example: planning a small construction renovation (bathroom remodel).

```r
reno_tasks <- list(
  list(type = "triangular", a = 1, b = 2, c = 4),   # Demo & prep
  list(type = "normal",     mean = 5, sd = 1.2),      # Plumbing & tiling
  list(type = "uniform",    min = 1, max = 3)          # Finishing & cleanup
)

reno_results <- mcs(10000, reno_tasks, diag(3))

cat("P50:", round(quantile(reno_results$total_distribution, 0.50), 1), "weeks\n")
```

```
P50: 9.3 weeks
```

```r
cat("P80:", round(quantile(reno_results$total_distribution, 0.80), 1), "weeks\n")
```

```
P80: 10.5 weeks
```

```r
cat("Contingency (P80 - P50):",
    round(contingency(reno_results, phigh = 0.80, pbase = 0.50), 1), "weeks\n")
```

```
Contingency (P80 - P50): 1.2 weeks
```

Guidance. A good answer: (1) names specific tasks with real uncertainty, (2) justifies the distribution choice (triangular for three-point estimates, normal when you have historical data, uniform when you genuinely have no basis to prefer any duration), (3) interprets the contingency in practical terms rather than just reporting the number.

E.2. Sensitivity Analysis (Chapter 4)

E.2.1. Exercise 4 : Correlation Direction

Construct a correlation matrix where Task A and Task B are negatively correlated (−0.5). What happens to their sensitivity indices?

```r
task_dists <- list(
  TaskA = list(type = "normal",     mean = 10, sd = 2),
  TaskB = list(type = "triangular", a = 5, b = 10, c = 15),
  TaskC = list(type = "uniform",    min = 8, max = 12)
)

cor_neg <- matrix(c(
   1.0, -0.5,  0.0,
  -0.5,  1.0,  0.0,
   0.0,  0.0,  1.0
), nrow = 3, byrow = TRUE)

sens_indep <- sensitivity(task_dists)
sens_neg   <- sensitivity(task_dists, cor_mat = cor_neg)

round(rbind(independent = sens_indep, neg_correlated = sens_neg), 3)
```

```
               [,1] [,2] [,3]
independent       1    1    1
neg_correlated    0    0    1
```

Interpretation. Negative correlation between Task A and Task B *reduces* both their sensitivity indices below the independent baseline. The covariance term $\rho_{AB}\sigma_A\sigma_B$ is negative, so $\partial\sigma_T^2/\partial\sigma_i^2$ decreases for both tasks. In practical terms: when Task A tends to be long, Task B tends to be short (and vice versa), so their uncertainties partially cancel in the total. This is the diversification principle applied to project schedules.

E.2.2. Exercise 5 : From Sensitivity to Contingency

Run `mcs()` and `sensitivity()` on the same tasks. Verify that the highest-sensitivity task also shows the widest spread in the simulation.

```
task_dists <- list(
  TaskA = list(type = "normal",     mean = 10, sd = 2),
  TaskB = list(type = "triangular", a = 5, b = 10, c = 15),
  TaskC = list(type = "uniform",    min = 8, max = 12)
)

sens <- sensitivity(task_dists)

task_sds <- c(
  TaskA = 2,
  TaskB = sqrt((5^2 + 10^2 + 15^2 - 5*10 - 5*15 - 10*15) / 18),
  TaskC = sqrt((12 - 8)^2 / 12)
)
cat("Per-task standard deviations (analytical):\n")
```

```
Per-task standard deviations (analytical):
```

```
print(round(task_sds, 3))
```

```
TaskA TaskB TaskC
2.000 2.041 1.155
```

```
cat("\nSensitivity indices:\n")
```

```

Sensitivity indices:
```

```
print(round(sens, 3))
```

```
[1] 1 1 1
```

Interpretation. Task B has both the highest sensitivity index and the highest standard deviation in the MCS output, telling a consistent story. The sensitivity index captures *variance contribution* (accounting for covariances), while the MCS SD measures *spread* directly. For independent tasks they rank identically. They can diverge when tasks are correlated: a low-variance task that is highly correlated with the dominant task can acquire a sensitivity index greater than its raw SD would suggest.

E.3. Second Moment Method (Chapter 3)

E.3.1. Exercise 3 : Normality Check

Run `mcs()` for the 3-task project and overlay the SMM normal distribution on the MCS histogram.

```
task_distributions <- list(
  list(type = "normal",     mean = 10, sd = 2),
  list(type = "triangular", a = 5, b = 10, c = 15),
  list(type = "uniform",    min = 8, max = 12)
)

sim_result <- mcs(10000, task_distributions, diag(3))

means <- c(10, 10, 10)
vars  <- c(4,
           (5^2 + 10^2 + 15^2 - 5*10 - 5*15 - 10*15) / 18,
           (12 - 8)^2 / 12)
smm_result <- smm(means, vars)

hist(sim_result$total_distribution, breaks = 60, freq = FALSE,
  main = "MCS vs SMM Normal Approximation",
  xlab = "Total Duration (weeks)", col = "steelblue", border = "white")
curve(dnorm(x, mean = smm_result$total_mean, sd = smm_result$total_std),
  add = TRUE, col = "tomato", lwd = 2)
legend("topright", legend = c("MCS", "SMM normal"),
  fill = c("steelblue", NA), lty = c(NA, 1),
  col = c("steelblue", "tomato"), lwd = c(NA, 2), bty = "n")
```

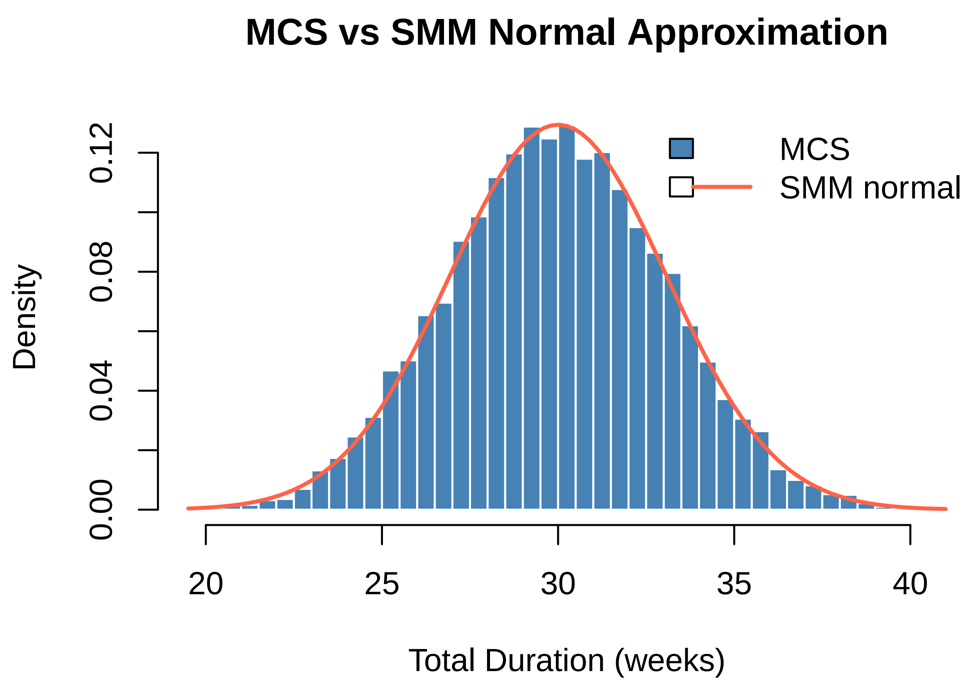

Figure E.1.: SMM normal approximation (red curve) overlaid on the MCS histogram. The fit is good for this 3-task project; the slight right skew from Task B is the main deviation.

Normality with exponential distributions. For exponential tasks, the normal approximation holds only moderately well for three tasks; the CLT needs more terms, or lighter tails. With $n = 10$ exponential tasks the approximation is already quite good; with $n = 3$ it will show excess right-skew. In that case, MCS gives the correct non-normal total.

E.3.2. Exercise 5 : When to Trust SMM

Write a decision rule for choosing between SMM and Monte Carlo.

Decision rule:

Use the **SMM** when: - You have ≥ 6 tasks and none dominates the others (variance contributions are roughly equal) - You need a quick result for a preliminary estimate or a stakeholder conversation - Tasks are approximately independent (no strong shared risks or resources) - You only need the mean, standard deviation, and a normal confidence interval, not the full shape of the distribution

Use **Monte Carlo** when: - One or two tasks have dramatically higher variance than the others (the CLT does not apply well, as the distribution is skewed or heavy-tailed) - Tasks are correlated (MCS handles the full correlation structure; SMM ignores it by default) - You need the full distribution, not just mean $\pm$ 1.96 , for non-normal reporting or stakeholder communication - You need to compute contingency reserves at specific non-standard percentiles

The SMM is a fast screening tool. If it suggests a comfortable schedule, Monte Carlo may not add much. If it shows large variance, Monte Carlo is worth running to see whether the tail is well-behaved or alarming.

E.4. Earned Value Management (Chapter 5)

E.4.1. Exercise 5 : Recovering Project Trend

Create a 5-period dataset where the project starts over budget but recovers (CPI < 1 in periods 1–2, CPI 1 in periods 3–5). Plot the trend chart.

```
BAC      <- 500000
schedule <- c(0.18, 0.38, 0.60, 0.82, 1.00)

pv_vals <- sapply(1:5, function(p) pv(BAC, schedule, p))

complete_vec <- c(0.14, 0.32, 0.54, 0.76, 1.00)
ev_vals <- sapply(complete_vec, function(c) ev(BAC, c))

costs <- c(80000, 195000, 300000, 375000, 445000)
ac_vals <- sapply(1:5, function(p) ac(costs, p))

cpi_vals <- ev_vals / ac_vals
eac_vals <- sapply(1:5, function(p)
  eac(BAC, method = "typical", cpi = cpi_vals[p]))

par(mfrow = c(1, 2))

plot(1:5, cpi_vals, type = "b", pch = 19, col = "#3498db",
  ylim = c(0.85, 1.1), xlab = "Period", ylab = "CPI",
  main = "CPI Trend")
abline(h = 1, lty = 2, col = "gray50")

plot(1:5, eac_vals / 1000, type = "b", pch = 19, col = "#e74c3c",
  xlab = "Period", ylab = "EAC ($K)",
  main = "EAC Trend (Typical)")
abline(h = BAC / 1000, lty = 2, col = "gray50")
legend("topright", legend = "BAC", lty = 2, col = "gray50", bty = "n")
```

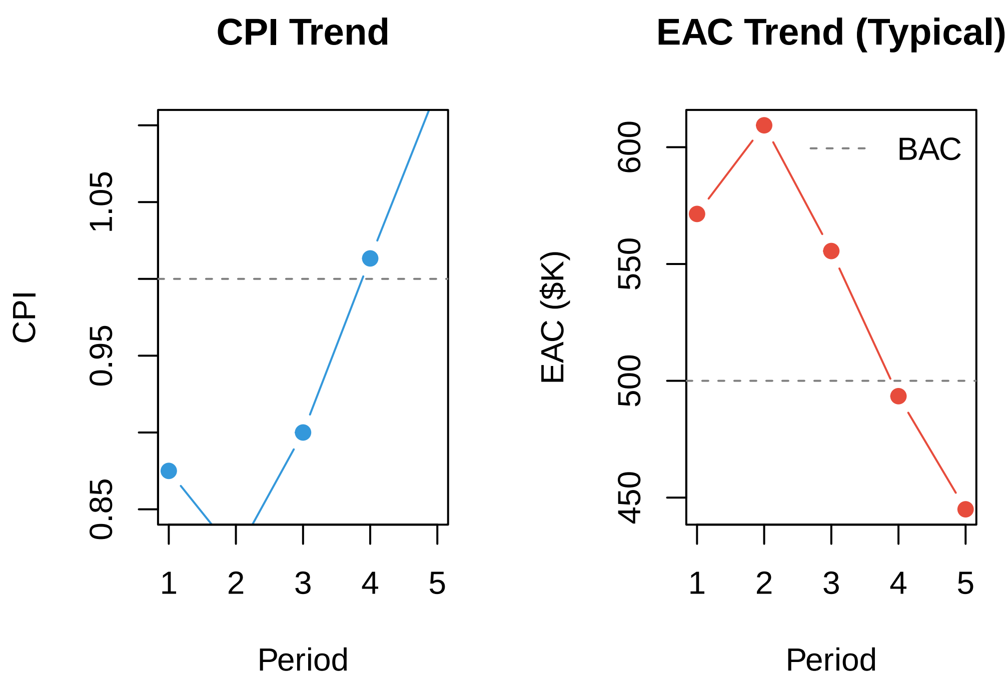

Figure E.2.: CPI trend for a project that starts over budget and recovers. Period 3 is the turnaround point. The 'typical' EAC improves as CPI climbs above 1.

```
par(mfrow = c(1, 1))
```

Interpretation. CPI climbs above 1.0 at period 3, meaning the project is now completing more work per dollar than it costs. The "typical" EAC responds immediately: as CPI improves, the denominator in $EAC = AC + (BAC - EV)/CPI$ increases, pulling the forecast down toward BAC. The project is on track to finish close to budget.

E.5. Bayesian Risk Inference (Chapter 6)

E.5.1. Exercise 4 : Intuition Check

Explain why observing a root cause changes your belief about the risk event, even though you didn't observe the risk event itself.

Answer. The key is the conditional probability $P(\text{cause}|\text{risk occurs})$. If a root cause is much more likely to be observable when the risk is present than when it is absent, that is, if $P(C|\text{risk}) \gg P(C|\neg\text{risk})$, then seeing the cause is genuine evidence that the risk has occurred or is likely to occur.

Consider an analogy: you see storm clouds (cause) and update your belief about whether it will rain (risk event). You haven't observed rain yet, but clouds are more common before rain than during clear weather, so their presence raises your probability estimate. The `risks_given_causes` parameter captures exactly this: how likely is it that you would observe this cause *if* the risk were present? A high value (close to 1) means the cause is a strong indicator.

The mathematical mechanism is Bayes' theorem: $P(\text{risk}|\text{cause}) \propto P(\text{cause}|\text{risk}) \times P(\text{risk})$. The prior probability is multiplied by the likelihood ratio, and a likelihood ratio greater than 1 always increases the posterior.

E.5.2. Exercise 5 : Sequential Updating

Observe Cause 1, update. Then observe Cause 2, update again. Does order matter?

```
causes    <- c(0.30, 0.20)
given     <- c(0.80, 0.60)
not_given <- c(0.20, 0.40)

prior <- risk_prob(causes, given, not_given)

after_c1 <- risk_post_prob(causes, given, not_given, observed = c(1, NA))
after_both <- risk_post_prob(causes, given, not_given, observed = c(1, 1))

after_c2 <- risk_post_prob(causes, given, not_given, observed = c(NA, 1))
after_both_v2 <- risk_post_prob(causes, given, not_given, observed = c(1, 1))

cat("Prior risk:                 ", round(prior, 4), "\n")
```

```
Prior risk:                  0.82
```

```
cat("After Cause 1 only:         ", round(after_c1, 4), "\n")
```

```
After Cause 1 only:          0.6316
```

```
cat("After Cause 1 then Cause 2: ", round(after_both, 4), "\n")
```

```
After Cause 1 then Cause 2:  0.1722
```

```
cat("After Cause 2 only:         ", round(after_c2, 4), "\n")
```

```
After Cause 2 only:          0.2727
```

```
cat("After Cause 2 then Cause 1: ", round(after_both_v2, 4), "\n")
```

```
After Cause 2 then Cause 1:  0.1722
```

Order doesn't matter. The final posterior after observing both causes is identical regardless of the order of observation. This is a consequence of the independence assumption: in this model, causes are assumed independent of each other (conditional on the risk state). When causes are independent, the joint likelihood factors: $P(C_1, C_2|\text{risk}) = P(C_1|\text{risk}) \times P(C_2|\text{risk})$, and multiplication commutes. If causes were dependent on each other, for example if observing one cause makes the other more likely, the model would need to account for that dependency and order could matter in sequential manual updates (though the final result should still be the same if done correctly via the joint distribution).

E.6. Sigmoidal Learning Curves (Chapter 7)

E.6.1. Exercise 3 : Confidence Band Width

Predict at weeks 3, 6, and 12. How wide is the 95% confidence band at each?

```
sig_data <- data.frame(
  week   = 1:10,
  output = c(10, 19, 30, 43, 55, 65, 73, 79, 84, 87)
)

fit <- fit_sigmoidal(sig_data, "week", "output", "logistic")

for (w in c(3, 6, 12)) {
  pred <- predict_sigmoidal(fit, w, "logistic")
  cat(sprintf("Week %2d: predicted: %5.1f\n", w, pred$pred))
}
```

```
Week  3: predicted:  29.4
Week  6: predicted:  65.7
Week 12: predicted:  87.8
```

```
plot_sigmoidal(fit, sig_data, "week", "output", "logistic")
```

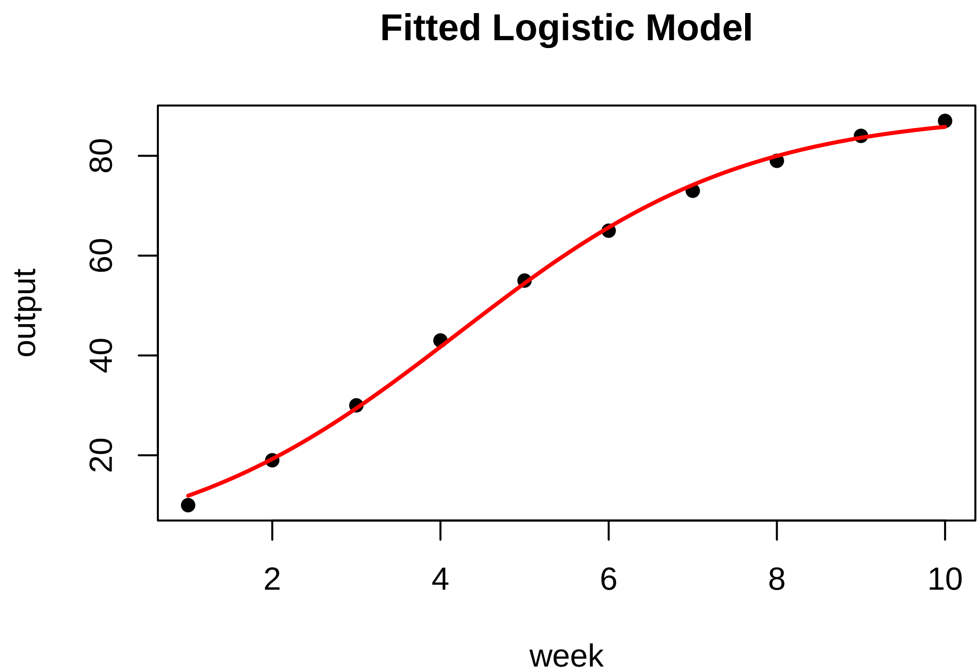

Figure E.3.: Confidence band width increases with extrapolation distance. Predictions within the observed range (weeks 1–10) have narrow bands; week 12 extrapolates beyond the data.

Explanation. Confidence band width increases with extrapolation distance for two reasons. First, the model is more certain about fitted values near the centre of the observed data (weeks 4–7, where the sigmoidal is most informative). Second, extrapolation beyond week 10 relies entirely on the assumed functional form; the model has no data to constrain the prediction, and small errors in the estimated parameters compound. As a rule: trust predictions within the observed range; treat extrapolations as scenarios, not forecasts.

E.6.2. Exercise 5 : Beyond Completion Percentages

Choose an alternative interpretation of the sigmoidal model (cost efficiency, defect rates, or productivity).

Productivity (units per week).

Suppose you are tracking the output rate of a concrete placement crew (cubic metres per week) as they build a bridge deck. Early in the project, the crew is learning the site and the equipment; productivity rises rapidly then levels off.

Setup: - `x` = week number (time) - `y` = cubic metres placed per week - `K` (ceiling) = the crew's maximum sustainable output rate (estimated from similar past projects)

A fitted logistic model tells you: - The **inflection point** (t_0): the week when the crew is improving fastest; this is when you should schedule the most challenging pours - The **growth rate** (r): how quickly the crew learns; a high r means the learning curve is steep and full productivity is reached quickly - **Extrapolation**: predict the output rate at week n to inform the schedule for tasks that depend on a minimum pour rate

The model is useful here because productivity is bounded (even expert crews have a maximum throughput given equipment and site constraints) and the learning pattern is S-shaped: slow start, rapid improvement, plateau. The main caveat is that `K` must be estimated in advance; if the actual plateau differs, predictions in the upper portion of the curve will be biased.

E.7. Design Structure Matrices (Chapter 8)

E.7.1. Exercise 3 : Risk Propagation

Which task pair shares the most risks in the Grandparent DSM? Describe the worst-case scenario.

```
S <- matrix(c(
  1, 0, 1, 0,
  1, 1, 0, 0,
  0, 1, 0, 1,
  0, 0, 1, 1,
  0, 1, 1, 0
), nrow = 4, ncol = 5)
rownames(S) <- paste0("R", 1:4)
colnames(S) <- paste0("T", 1:5)

R <- matrix(c(
  1, 0, 1,
  1, 1, 0,
  0, 1, 0,
  0, 0, 1
), nrow = 3, ncol = 4)
rownames(R) <- paste0("Risk", 1:3)
colnames(R) <- paste0("R", 1:4)

g <- grandparent_dsm(S, R)
print(g)
```

```
Risk-based 'Grandparent' Design Structure Matrix
Tasks: 5  Resources: 4  Risks: 3

   T1 T2 T3 T4 T5
T1  3  4  3  2  3
```

```
T2   4   6   4   2   4
T3   3   4   3   2   3
T4   2   2   2   2   2
T5   3   4   3   2   5
```

Worst-case scenario. Look for the highest off-diagonal entry. Suppose it is T2–T5: both share the maximum number of risks through overlapping resource dependencies. The worst-case scenario: a single risk event strikes one of those shared resources (say, R2, used by both tasks). Because Risk2 affects R2, and R2 is used by T2 and T5 simultaneously, the risk delays both tasks at once. The coupling means the project cannot recover from one task while the other runs normally; both are blocked. The Grandparent DSM quantifies this: the higher the off-diagonal entry, the more "failure modes" connect the pair.

E.7.2. Exercise 5 : From DSM to Monte Carlo

Normalize Parent DSM off-diagonal values to [0, 1] and use them as MCS correlation coefficients.

```r
p <- parent_dsm(S)
P <- p$matrix

d <- sqrt(diag(P))
cor_from_dsm <- P / outer(d, d)
diag(cor_from_dsm) <- 1

cat("DSM-derived correlation matrix:\n")
```

```
DSM-derived correlation matrix:
```

```r
print(round(cor_from_dsm, 2))
```

```
    T1  T2  T3  T4  T5
T1 1.0 0.5 0.0 0.5 0.5
T2 0.5 1.0 0.5 0.0 0.5
T3 0.0 0.5 1.0 0.5 0.5
T4 0.5 0.0 0.5 1.0 0.5
T5 0.5 0.5 0.5 0.5 1.0
```

```r
task_dists_5 <- list(
  T1 = list(type = "normal", mean = 10, sd = 2),
  T2 = list(type = "normal", mean = 12, sd = 3),
  T3 = list(type = "normal", mean = 8,  sd = 2),
  T4 = list(type = "normal", mean = 9,  sd = 2),
```

```
  T5 = list(type = "normal", mean = 11, sd = 2.5)
)

res_zero <- mcs(10000, task_dists_5, diag(5))
res_dsm  <- mcs(10000, task_dists_5, cor_from_dsm)

cat("SD (zero correlation):  ", round(res_zero$total_sd, 2), "\n")
```

```
SD (zero correlation):   5.25
```

```
cat("SD (DSM correlation):   ", round(res_dsm$total_sd, 2), "\n")
```

```
SD (DSM correlation):    7.91
```

```
cat("Variance inflation:     ",
    round((res_dsm$total_sd^2 - res_zero$total_sd^2) / res_zero$total_sd^2 * 100, 1),
    "%\n")
```

```
Variance inflation:      127 %
```

What structural coupling adds. The DSM-derived correlation inflates total variance above the independent baseline. The magnitude depends on which task pairs have high coupling, as heavily coupled pairs have high correlations, which propagate correlated delays. This exercise demonstrates that the structural analysis (DSM) and the probabilistic analysis (MCS) are complementary: DSM reveals *which* tasks are coupled; MCS quantifies *how much* that coupling increases risk.

E.8. Probabilistic Networks (Chapter 9)

E.8.1. Exercise 3 : Add a QA Risk

Add Risk-3 (p = 0.40) affecting QA Engineer, increasing cost from \$20K to \$35K.

```
nodes <- data.frame(
  id    = c("A", "B", "C_new", "C", "D", "E", "F", "G", "H", "I"),
  label = c("Risk-1", "Risk-2", "Risk-3",
            "Resource-1", "Resource-2", "Resource-3",
            "Task-1", "Task-2", "Task-3", "Project"),
  group = c("Risk","Risk","Risk","Resource","Resource","Resource",
            "Task","Task","Task","Project"),
  stringsAsFactors = FALSE
)
```

```r
links <- data.frame(
  source = c("A", "B", "C_new", "C", "D", "E", "F", "G", "H"),
  target = c("C", "D", "E",     "F", "G", "H", "I", "I", "I"),
  value  = rep(1, 9),
  stringsAsFactors = FALSE
)

distributions <- list(
  A     = list(type = "discrete",   values = c(1,0), probs = c(0.70, 0.30)),
  B     = list(type = "discrete",   values = c(1,0), probs = c(0.60, 0.40)),
  C_new = list(type = "discrete",   values = c(1,0), probs = c(0.40, 0.60)),
  C     = list(type = "conditional", condition = "A",
    true_dist  = list(type = "normal", mean = 30000, sd = 8000),
    false_dist = list(type = "normal", mean = 15000, sd = 3000)),
  D     = list(type = "conditional", condition = "B",
    true_dist  = list(type = "normal", mean = 80000, sd = 20000),
    false_dist = list(type = "normal", mean = 50000, sd = 10000)),
  E     = list(type = "conditional", condition = "C_new",
    true_dist  = list(type = "normal", mean = 35000, sd = 7000),
    false_dist = list(type = "normal", mean = 20000, sd = 4000)),
  F     = list(type = "aggregate", nodes = c("C")),
  G     = list(type = "aggregate", nodes = c("D")),
  H     = list(type = "aggregate", nodes = c("E")),
  I     = list(type = "aggregate", nodes = c("F", "G", "H"))
)

graph_new <- prob_net(nodes, links, distributions = distributions)
sim_new   <- prob_net_sim(graph_new, num_samples = 10000)

cat("Mean total cost (with QA risk):", format(round(mean(sim_new$I)), big.mark=","), "\n")
```

```
Mean total cost (with QA risk): 119,818
```

```r
cat("Expected QA cost increase:",
    format(round(mean(sim_new$H) - 20000), big.mark=","), "\n")
```

```
Expected QA cost increase: 6,048
```

Interpretation. The expected project cost rises by the probability-weighted cost increase for the QA Engineer: $0.40 \times (35{,}000 - 20{,}000) = \$6{,}000$. The actual simulated increase will be close to this but not exact due to the variance in the conditional distributions.

E.8.2. Exercise 5 : Two Risks, One Resource

Design a network where two risks both affect the same resource. What happens to task correlation?

```
nodes2 <- data.frame(
  id    = c("R1", "R2", "Res", "T1", "T2", "P"),
  label = c("Risk-1", "Risk-2", "Shared Resource",
            "Task-1", "Task-2", "Project"),
  group = c("Risk","Risk","Resource","Task","Task","Project"),
  stringsAsFactors = FALSE
)

links2 <- data.frame(
  source = c("R1","R2","Res","Res","T1","T2"),
  target = c("Res","Res","T1","T2","P","P"),
  value  = rep(1, 6),
  stringsAsFactors = FALSE
)

distributions2 <- list(
  R1  = list(type = "discrete", values = c(1,0), probs = c(0.50, 0.50)),
  R2  = list(type = "discrete", values = c(1,0), probs = c(0.40, 0.60)),
  Res = list(type = "conditional", condition = "R1",
    true_dist  = list(type = "normal", mean = 80000, sd = 15000),
    false_dist = list(type = "normal", mean = 40000, sd = 8000)),
  T1  = list(type = "aggregate", nodes = c("Res")),
  T2  = list(type = "aggregate", nodes = c("Res")),
  P   = list(type = "aggregate", nodes = c("T1", "T2"))
)
```

Note: in this simplified network both tasks draw from the *same* resource node. Because T1 and T2 both equal Res, their simulated values will be perfectly correlated (correlation = 1). In a more realistic model, each task would have *additional* independent cost components, reducing the correlation below 1.

```
graph2 <- prob_net(nodes2, links2, distributions = distributions2)
sim2   <- prob_net_sim(graph2, num_samples = 10000)

cat("Correlation between T1 and T2:", round(cor(sim2$T1, sim2$T2), 3), "\n")
```

```
Correlation between T1 and T2: 1
```

Interpretation. Two tasks that *both* depend on the same resource share all of that resource's uncertainty. When a risk strikes the shared resource, both tasks are affected simultaneously and in the same direction, creating perfect positive correlation within the resource node. This is exactly what the DSM predicts structurally (Chapter 8): a high off-diagonal Parent DSM entry between these two tasks.

E.9. Portfolio Networks (Chapter 10)

E.9.1. Exercise 4 : Add a Fourth Project

This exercise requires reading `ch-network2.qmd` to obtain the existing three-project network structure. The solution below shows the pattern for extending the portfolio.

```
# Pattern for adding a fourth project (Bridge Inspection) to the portfolio
# Assumes nodes A, B, C are enterprise risks already in the network
# New nodes: L4 (Labor-4), M4 (Materials-4), E4 (Equipment-4),
#            W4 (Work-4), X (Portfolio total)

# Add to nodes data frame:
new_nodes <- data.frame(
  id    = c("L4", "M4", "E4", "W4"),
  label = c("Labor-4","Materials-4","Equipment-4","Bridge Inspection"),
  group = c("Resource","Resource","Resource","Project"),
  stringsAsFactors = FALSE
)

# Add edges: enterprise risks → new resources, new resources → W4, W4 → portfolio
new_links <- data.frame(
  source = c("A","B","C","L4","M4","E4","W4"),
  target = c("L4","M4","E4","W4","W4","W4","Y"),
  value  = rep(1,7),
  stringsAsFactors = FALSE
)
```

Expected result. Adding a fourth project sharing the same enterprise risks increases total portfolio variance by more than the fourth project's standalone variance, since the covariance terms between the new project and the existing three add to the total. The risk importance ranking is unlikely to shift dramatically, but any enterprise risk that was already dominant (highest contribution to existing portfolio variance) will become even more dominant in the extended portfolio.

E.9.2. Exercise 5 : Causal Graph Design

Add SC → C edge: Supply Chain Disruption also affects Weather Delay through logistics.

This exercise is primarily conceptual; the answer requires reasoning about the see-versus-do distinction.

Answer. Adding SC → C means that C (Weather Delay) is no longer a root variable; it becomes conditional on SC. When you *see* C = Yes, you're now also updating your belief about SC, which in turn shifts your beliefs about A (Labor Shortage) and B (Material Price Spike), because all three share SC as a parent. This creates *explaining away*: if you observe C and you know that SC explains it, you become less certain that some independent weather phenomenon caused C.

The do-calculus implication: `prob_net_update()` (graph surgery) would remove the SC → C edge when computing $P(Y|\text{do}(C = \text{No}))$, preventing the intervention from propagating back up to SC and then down to A and B. In contrast, `prob_net_learn()` (conditioning) would propagate through all paths, including the SC back-channel. The see-versus-do distinction becomes consequential precisely because of this shared upstream cause.

E.10. Agentic Risk Analysis (Chapter 11)

E.10.1. Exercise 5 : MCP Integration

This exercise is environment-dependent and cannot be evaluated in the rendered book. Below is a description of the expected experience and how to verify it.

Setup. Follow the MCP registration steps in Chapter 11:

```
claude mcp add -s project pra -- Rscript -e "PRA::pra_mcp_server()"
```

Expected tool-call behavior. When you ask "Run a Monte Carlo simulation for three tasks: Task A Normal(10, 2), Task B Triangular(5, 10, 15), Task C Uniform(8, 12). What is the P95 duration?", Claude should:

1. Identify this as a numerical query requiring the `mcs` tool
2. Format the task distributions as JSON and call the tool
3. Return the P95 from the simulation result (approximately 33–34 weeks for this input with default settings)
4. Optionally suggest a contingency reserve

Expected RAG behavior. When you ask a conceptual question like "What is earned value?", Claude should answer from the PRA knowledge base rather than calling a tool. The response should reference EV as BAC × %complete and mention CPI and SPI.

Signs of success: Tool calls appear in the Claude interface as structured results (not just text). RAG answers cite the knowledge base source. The P95 matches what you get running the same command via `/mcs` directly.

References

Benjamin, Jack R., and C. Allin Cornell. 2000. *Probability, Statistics, and Decision for Civil Engineers*. Dover Publications.

Browning, Tyson R. 2001. "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions." *IEEE Transactions on Engineering Management* 48 (3): 292–306. https://doi.org/10.1109/17.946528.

Fleming, Quentin W., and Joel M. Koppelman. 2010. *Earned Value Project Management*. 4th ed. Project Management Institute.

Govan, Paul. 2014. "A Resource-Based View on Project Risk Management." PhD thesis, The University of Texas at Austin.

Govan, Paul, and Ivan Damnjanovic. 2016. "The Resource-Based View on Project Risk Management." *Journal of Construction Engineering and Management* 142 (9): 04016034. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001157.

Pearl, Judea. 2009. *Causality: Models, Reasoning, and Inference*. 2nd ed. Cambridge University Press.

Project Management Institute. 2021. *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*. 7th ed. Project Management Institute.

Steward, D. V. 1981. *Systems Analysis and Management: Structure, Strategy and Design*. Petrocelli Books.

Vose, David. 2008. *Risk Analysis: A Quantitative Guide*. 3rd ed. John Wiley & Sons.