Bayesian Data Analysis for Software Engineering

Richard Torkar Carlo A. Furia Robert Feldt
Chalmers and University of Gothenburg, Sweden
USI Lugano, Switzerland

ICSE 2021 Technical Briefings

Part 3: Using Bayesian Data Analysis in SE?

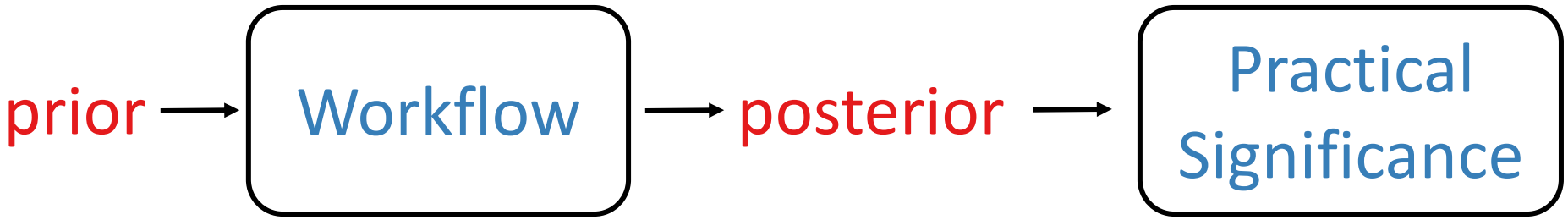
Bayesian Data Analysis for Software Engineering

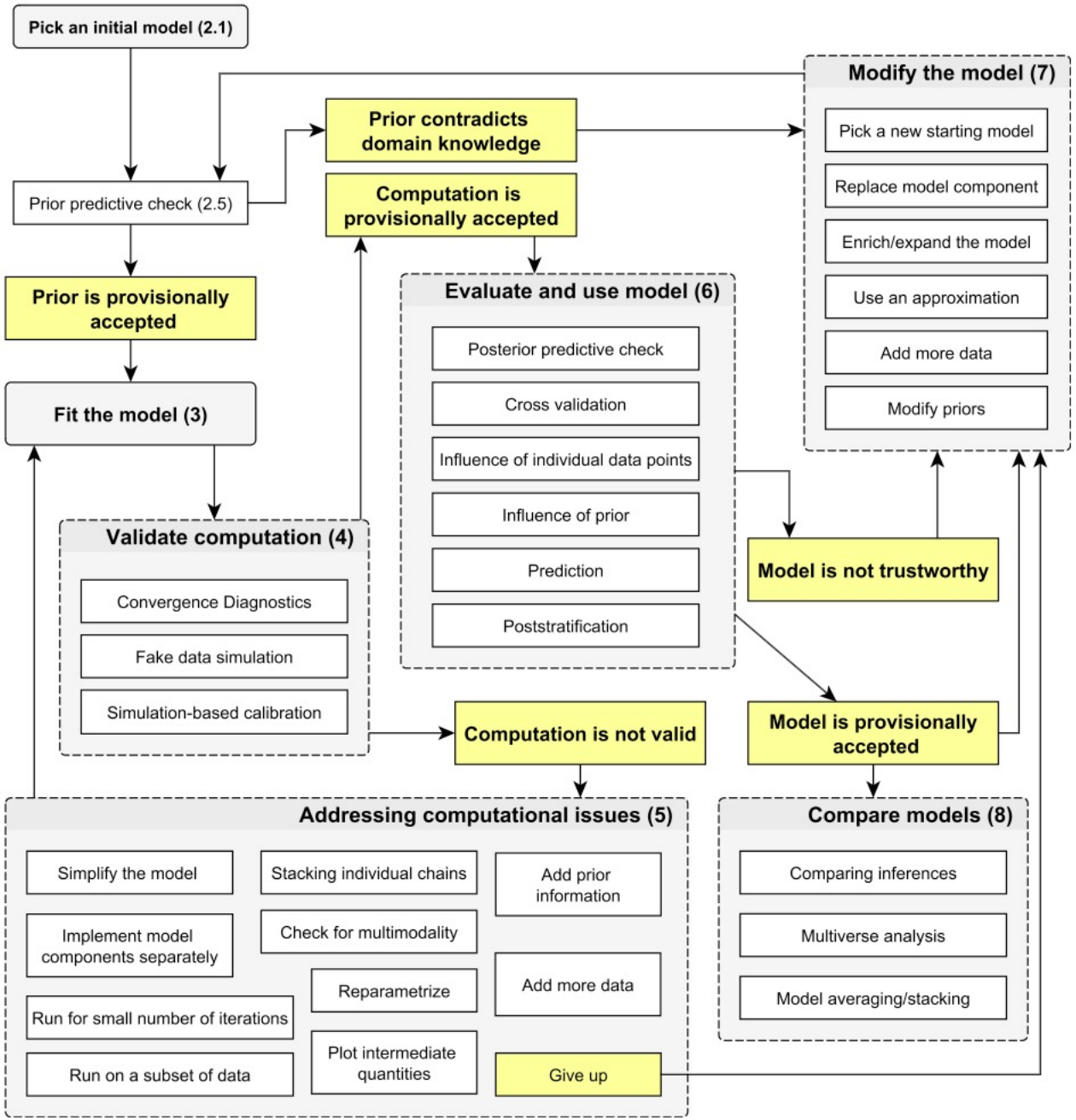
Richard Torkar Carlo A. Furia Robert Feldt

ICSE 2021 Technical Briefings

Bigger picture of using BDA in SE

- In addition to motivation and steps of Bayesian inference we need:
  - A Bayesian workflow for SE data
- Using the output:
  1. Science: SE Chains of evidence, i.e. “My posterior is their prior” (& vice versa)
  2. Engineering: Practical significance of SE results





More detailed workflows are in development:
Gelman et al, ”Bayesian Workflow”, arxiv.org/pdf/2011.01808.pdf

Our condensed SE version: arxiv.org/pdf/2101.12591.pdf

Applying Bayesian Analysis Guidelines to Empirical Software Engineering Data

The Case of Programming Languages and Code Quality

Carlo A. Furia[1] · Richard Torkar[2,3] · Robert Feldt[2]

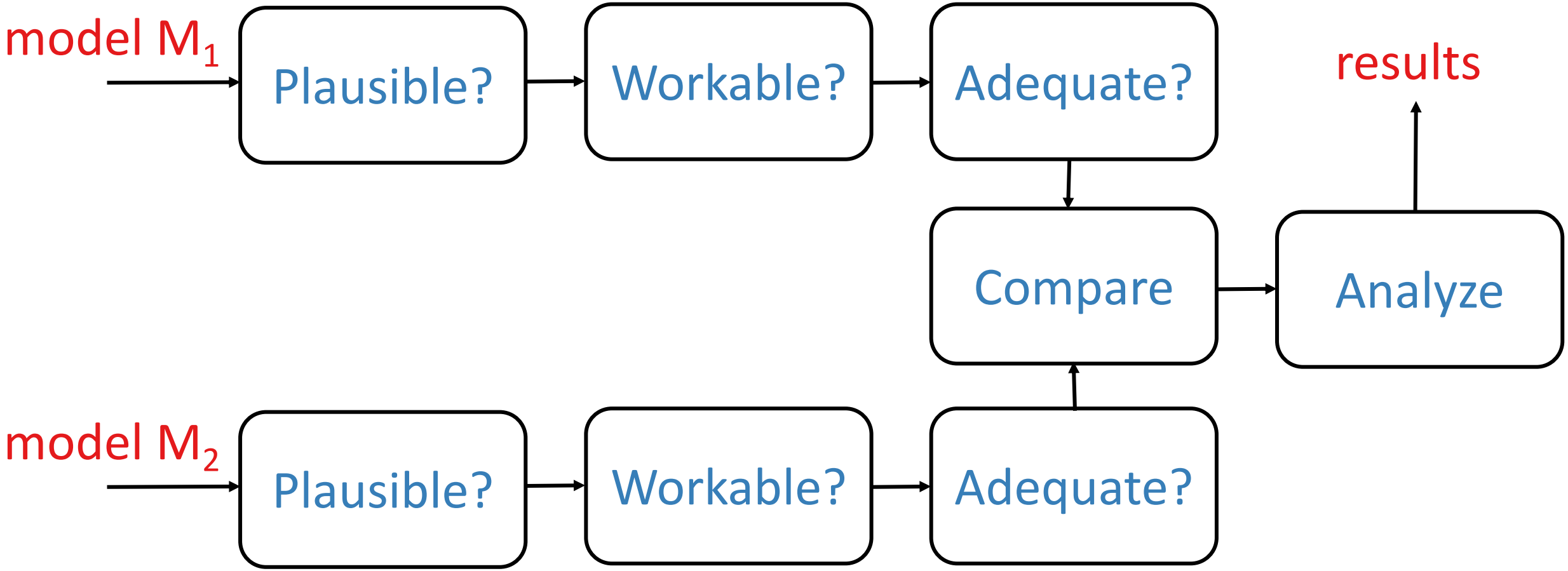
[1] Software Institute, USI Università della Svizzera Italiana, Switzerland
[2] Chalmers and the University of Gothenburg, Sweden
[3] Stellenbosch Institute for Advanced Study (STIAS), South Africa

2021–02–01

Incremental modeling is key

- We build model in steps, incrementally
  - Unlikely: find right model “level” or ”resolution” directly
  - Explore/adapt guided by results and reasoning (Agile)
- Two main starting points:
  - Simplest thinkable model (then build up = add complexity/realism)
  - Prior/frequentist model (then explore by simplifying or adding)

SE BDA Workflow: Incremental build-compare-analyze



Plausible: consistent with domain knowledge?

- Prior + Model consistent with our domain knowledge?
  - Prior predictive simulations = “implications of priors in context of a generative model”
- Distribution of project efforts, our prior should exclude:
  - Physical or logical limits:
    - Negative project length/effort
    - Longer than Homo Sapiens have existed
  - “Common sense”:
    - Very short projects, say, on the order of hours?!
    - Very long projects, say, on the order of decades/centuries?!
  - Prior research:
    - SW projects are typically on the order of weeks up to 3-5 years!?
    - If including maintenance maybe up to 25-40 years (but depends on what we study here)!?
- Model factors/variables:
  - One (modern) programming language unlikely to give 100/1000 more effort!?

Workable: computationally tractable

- Computational problems are a feature of a BDA workflow!
  - The model is speaking to us! ☺
- Inference takes days/weeks => model is too complex? Simplify!
- Computation diverges => reparameterize
- Caveats:
  - Better tools can make more complex models tractable
  - Automation will help more here than in other steps

Adequate: model/data relation

- How well does our model capture reality, i.e. our empirical data?
- Are there some parts of the data that are missed by the model?
- Posterior predictive checks = prior predictive checks but with parameter distributions after fitting actual data

Compare: which model does what better?

- Simplicity vs. Accuracy trade-off
  - Bias vs Variance
  - Many metrics/tools in BDA for this!
- Not only about accurate fit!
  - Scientific costs:
    - Complex models are harder to learn from and build on
  - Practical costs:
    - Harder to understand => less likely to be used
    - More costly to collect data => less likely to be (used and) maintained

Practical significance: what it really means?

- Posterior distribution includes the uncertainty
- Simulate from posterior and answer practical questions of domain
- For Language to project effort models:
  - Which language should we choose in this new project?
  - Is it cost-effective to switch language given the project costs we have?
- Typically need to add cost-related information
  - But engineers/practitioners often good at “ballpark” estimates

A Method to Assess and Argue for Practical Significance in Software Engineering

Richard Torkar, Carlo A. Furia, Robert Feldt, Francisco Gomes de Oliveira Neto,
Lucas Gren, Per Lenberg, and Neil A. Ernst

Abstract—A key goal of empirical research in software engineering is to assess practical significance, which answers whether the observed effects of some compared treatments show a relevant difference in practice in realistic scenarios. Even though plenty of standard techniques exist to assess statistical significance, connecting it to practical significance is not straightforward or routinely done; indeed, only a few empirical studies in software engineering assess practical significance in a principled and systematic way. In this paper, we argue that Bayesian data analysis provides suitable tools to assess practical significance rigorously. We demonstrate our claims in a case study comparing different test techniques. The case study's data was previously analyzed (Afzal et al., 2015) using standard techniques focusing on statistical significance. Here, we build a multilevel model of the same data, which we fit and validate using Bayesian techniques. Our method is to apply cumulative prospect theory on top of the statistical model to quantitatively connect our statistical analysis output to a practically meaningful context. This is then the basis both for assessing and arguing for practical significance.

arxiv.org/pdf/1809.09849.pdf

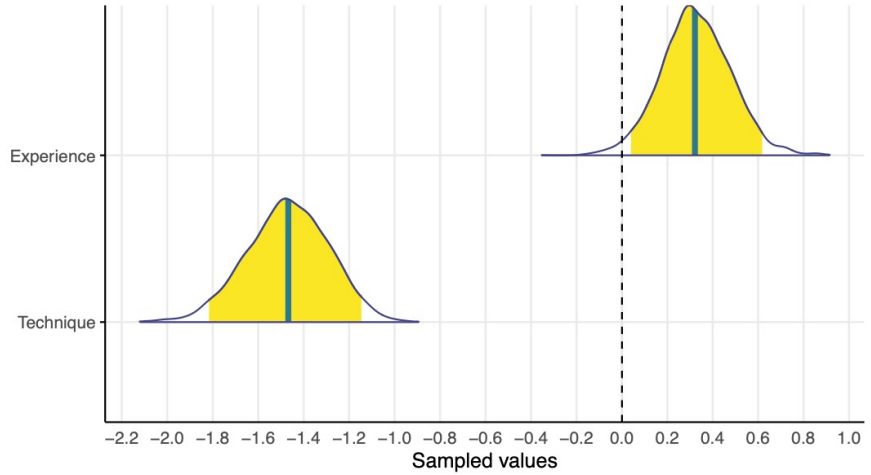


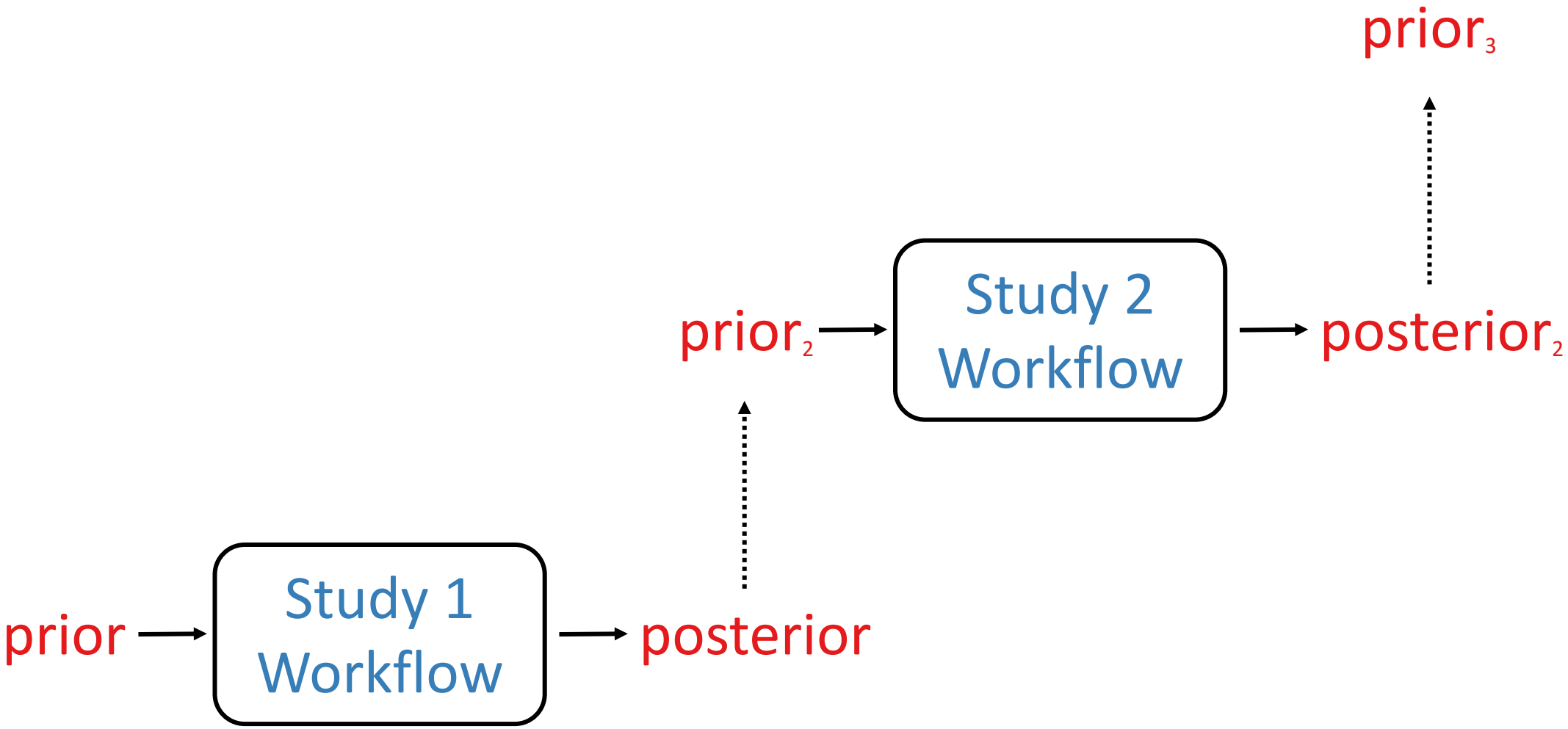
Fig. 3. Posterior marginal probability distributions of β_e (*experience*, top) and β_a (*approach*, bottom named 'Technique'). The thick lines mark the medians, and the yellow areas cover 94% of probability.

TABLE 3
Expected number of faults detected for different combinations of predictors in $\mathcal{M}_2$. Each row reports the range of *faults* corresponding to 94% probability and the mean on the posterior.

developers	fixed predictors	94% CI	mean
low experience	$experience = 0$	1, 8	4.02
high experience	$experience = 1$	1, 11	5.67
exploratory testing	$approach = 0$	6, 11	8.27
test-case testing	$approach = 1$	1, 2	1.42
exploratory and low	$approach = 0$ $experience = 0$	6, 8	6.92
exploratory and high	$approach = 0$ $experience = 1$	8, 12	9.61

Simulations + Estimated costs => Practical implications & significance

Building SE chains of evidence



License of these slides