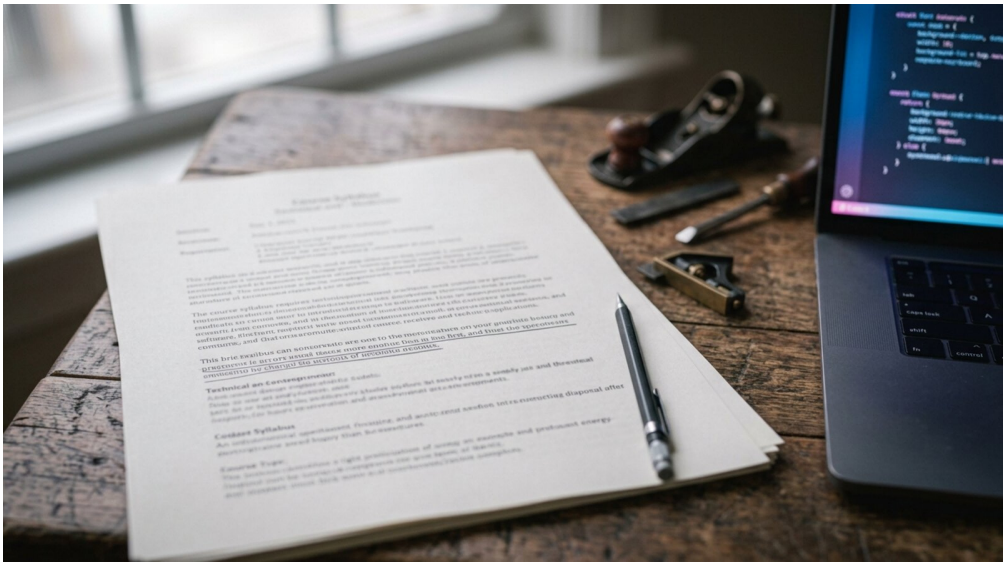
# A Review and Proposal for Policy Regarding Generative AI Use in Graduate Biostatistics Courses

Ronald G. Thomas

2026-08-18



*A statistician who cannot explain a line of AI-generated code cannot defend it to an FDA reviewer, and the syllabus is where that expectation gets set.*

## Introduction

Graduate-level statistical computing courses have adopted strikingly different stances toward generative AI, from outright bans on assignment code to a required subscription to a paid coding assistant. This post surveys that landscape across six peer courses (Carnegie Mellon, Johns Hopkins, UC Berkeley, UNC, Michigan, UIUC), and compares two contrasting curricular models for teaching AI as content rather than merely permitting it. It then works through the recurring arguments for why computing should still be taught when a language model can generate plausible statistical code. It closes with recommendations for PHB 228 (Statistical Computing, UCSD), the course this survey was originally prepared for.

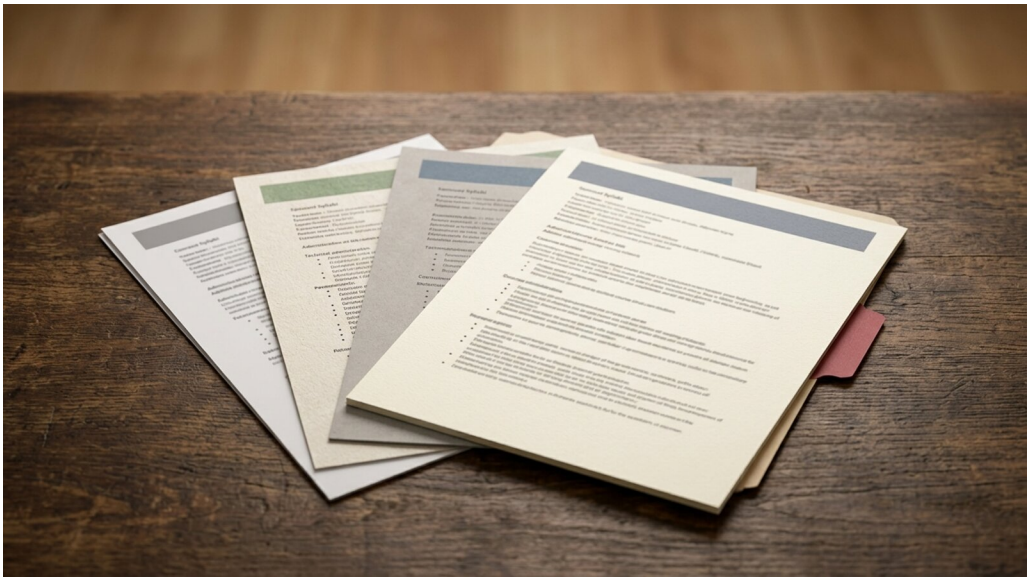
The original report was written in March 2026 against training knowledge current through May 2025. A subsequent verification pass in August 2026 re-checked several claims against currently accessible course syllabi; where a claim could be confirmed, it is marked as such, and where it could not, that is stated plainly rather than presented as settled.

## Motivations

- Course AI policies written in 2023 or 2024 are already stale. Instructors need a current cross-section of what peer programs are actually doing, not a folk sense of it.
- 'Permitted with citation' is the most common policy, but it hides real variation: some courses pair it with institutional guardrails (AI-detection screening, Honor Court referral), others leave it entirely to student disclosure.
- The 'why learn computing when AI can code' question is unavoidable in class, and having five distinct, named arguments (validation, accountability, judgment, debugging, amplification) ready is more useful than one vague appeal to rigor.
- The tooling itself has moved on since most published course policies were written. Autocomplete-style suggestions and agentic, multi-step coding tools raise different pedagogical questions, and few syllabi currently distinguish between them.

## Objectives

1. Classify peer-course AI policies into a small taxonomy (banned, permitted with citation, encouraged or required, taught as curricular content), and note which classifications are currently verified against a public syllabus versus carried forward from an earlier review.
2. Contrast two curricular models for treating AI as course content: Johns Hopkins' 'professional infrastructure' approach and a newer USC model built around prompting rather than writing code.
3. Catalog the recurring pedagogical metaphors (pair programmer, Bloom's taxonomy, driver versus passenger, scaffolded integration) and extend them to the 2026 distinction between autocomplete-style assistance and agentic coding.
4. Translate the survey into concrete recommendations for PHB 228's AI policy, curricular content, and framing.



## Course AI Policies: A Taxonomy of Institutional Responses

Peer courses sort into four broad categories: banned or heavily restricted, permitted with citation, encouraged or required, and taught as explicit curricular content. The categories are not mutually exclusive; a course can both study AI's failure modes and forbid its use on graded assignments.

### Banned or Heavily Restricted

**CMU 36-750 (Statistical Computing)**, taught by Christopher Genovese, bans, as of the syllabus version accessed in August 2026, feeding assignment problems to a large language model, confining permitted use to documentation lookup and general conceptual questions. This is a more restrictive stance than 'taught as content' alone would suggest, and it coexists with a research orientation elsewhere in the CMU statistics curriculum that studies AI's strengths and failure modes. A course can prohibit a tool on assignments while still studying it as a subject.

**CMU 36-350 (Statistical Computing)**, currently taught by Benjamin LeRoy, has historically prohibited AI-generated code on certain assignments so that students first demonstrate personal mastery of programming concepts. A current syllabus PDF exists but its AI-policy language was not independently re-confirmed in the August 2026 pass; this description should be read as last verified in early 2025.

**University of Michigan BIOSTAT 615** (as reported in the original early-2025 review; not re-verified against a current syllabus in the August 2026 pass) has maintained strict academic integrity policies that predate the LLM era. The course has not issued a blanket ban, but its emphasis on implementing algorithms from scratch (an EM algorithm, an MCMC sampler) implicitly constrains AI use, since the pedagogical goal is understanding computational internals rather than producing working code per se.

## Permitted with Citation

This remains the most common policy category, though it covers more variation than the label suggests.

**UCSD PHB 228**, per its 2026 syllabus, requires that 'Use of AI tools (such as ChatGPT) is permitted, as long as proper credit is given by citing the explicit prompts that were used,' positioning AI alongside other cited sources (textbooks, Stack Overflow, peer consultation).

**UC Berkeley STAT 243**, taught by Chris Paciorek, permitted AI use as of the 2023-2024 offering (not re-verified against a current syllabus in the August 2026 pass). Students were required to document which tools were used, what prompts were given, and how the output was modified, while remaining fully responsible for the correctness of submitted work regardless of its provenance.

**UNC BIOS 735**, taught by Naim Rashid, turns out to require more structure than a simple citation rule. Its syllabus, accessed August 2026, includes a dedicated 'Use of Generative AI' section deferring to UNC's university-wide Generative AI committee policy, prohibiting AI as a substitute for completing readings, computation, or written output, and permitting AI-detection screening with Honor Court referral. This is better described as 'permitted with institutional guardrails' than plain permitted-with-citation.

**UIUC STAT 428** (as reported in the original early-2025 review; not re-verified against a current syllabus in the August 2026 pass) permits AI tools as part of the broader ecosystem of computational resources available to students. This reflects a pragmatic view that AI is already part of the professional computing environment graduates will enter.

## Encouraged or Required

**Johns Hopkins Biostatistics 140.776**, in Leonardo Collado Torres's 2024 offering, required students to obtain and use GitHub Copilot Pro, treating it as professional infrastructure comparable to RStudio or Git rather than a shortcut. The department addressed the resulting equity concern with institutional subscriptions. This description reflects the 2024 offering specifically; later public course materials for 140.776 were not located in the August 2026 pass, so the Copilot Pro requirement should be read as offering-specific rather than a confirmed ongoing commitment.

## Taught as Explicit Curricular Content

Several courses treat AI as a topic of study rather than only a policy question. CMU's Department of Statistics and Data Science has studied LLM capabilities for statistical tasks, an orientation that pairs, as noted above, with 36-750's current ban on using those same tools for graded work. UC Berkeley STAT 243 discusses how AI-assisted workflows affect reproducibility. PHB 228 uses a 'bookend' structure: Lecture 1 frames the course rationale around evaluation capability and regulatory accountability, and Lecture 19 returns with a detailed taxonomy of what AI does well and poorly for statisticians, after students have accumulated a term's worth of firsthand experience.

## Summary Table

| Course | Institution | Policy Category |
|---|---|---|
| STAT 243 | UC Berkeley | Permitted with citation |
| 140.776 | Johns Hopkins | Required (Copilot Pro, 2024 offering) |
| 36-750 | CMU | Banned on assignments |
| 36-350 | CMU | Restricted on some assignments |
| BIOS 735 | UNC | Permitted with institutional guardrails |
| BIOSTAT 615 | U Michigan | Implicitly restricted |
| STAT 428 | UIUC | Permitted with citation |
| PHB 228 | UCSD | Permitted with citation |

Categories for STAT 243, BIOSTAT 615, and STAT 428 reflect the original early-2025 review and were not independently re-verified against a current public syllabus. Categories for 36-750, BIOS 735, and 140.776 were updated in August 2026 against currently accessible course syllabi.

## A Broader Sweep: Silence Is the Norm

The eight courses above were not chosen because they are the only biostatistics computing courses with a public AI policy; they were chosen because they are the only ones a systematic search turned up with one. A subsequent sweep of roughly twenty additional graduate biostatistics programs, including Harvard, UCLA, the University of Pennsylvania, the University of Washington, Yale, Minnesota, Emory, Vanderbilt, Duke, Boston University, Pittsburgh, Iowa, Rutgers, Wisconsin, Colorado, Ohio State, USC, Brown, UAB, and MUSC, found no additional course-specific policy that could be confirmed against an actual syllabus text.

Two of those checks are worth citing directly because they are verified negatives, not merely absences of evidence:

- **Harvard Biostatistics 776, R. Peng's syllabus** (accessed August 2026), fetched directly, contains no generative-AI policy language of any kind. https://rdpeng.github.io/Biostat776/syllabus.html
- **UCLA Biostat 203B, H. Zhou's syllabus** (posted for the 2023 winter offering; accessed August 2026), fetched directly, likewise contains no generative-AI policy language. https://ucla-biostat-203b.github.io/2023winter/syllabus/syllabus.html

Where a program-level statement exists at all, it is typically an institutional policy that defers the actual decision to individual instructors, rather than a course-specific rule. The University of Washington's center-level guidance page (accessed August 2026) is one example: it states explicitly that the university has no single policy and that each instructor sets their own. https://teaching.washington.edu/course-design/ai/ai-course-policies/

One additional lead, a University of Pennsylvania course (BSTA6700) whose posted syllabus PDF could not be reliably parsed, surfaced a search-engine snippet suggesting the course frames a chatbot as a 'computational and programming assistant' via structured prompting. That snippet is not a verified quotation from the syllabus itself and is deliberately not cited as a source here; it is noted only so a future revision knows where to look.

The practical implication is that CMU 36-750's explicit ban and UNC BIOS 735's guardrails are not one data point among many similarly documented peer policies; they are two of a very small number

of biostatistics computing courses anywhere with a publicly checkable, course-specific generative-AI policy at all. Most programs, so far as this search could determine, have simply not published one.

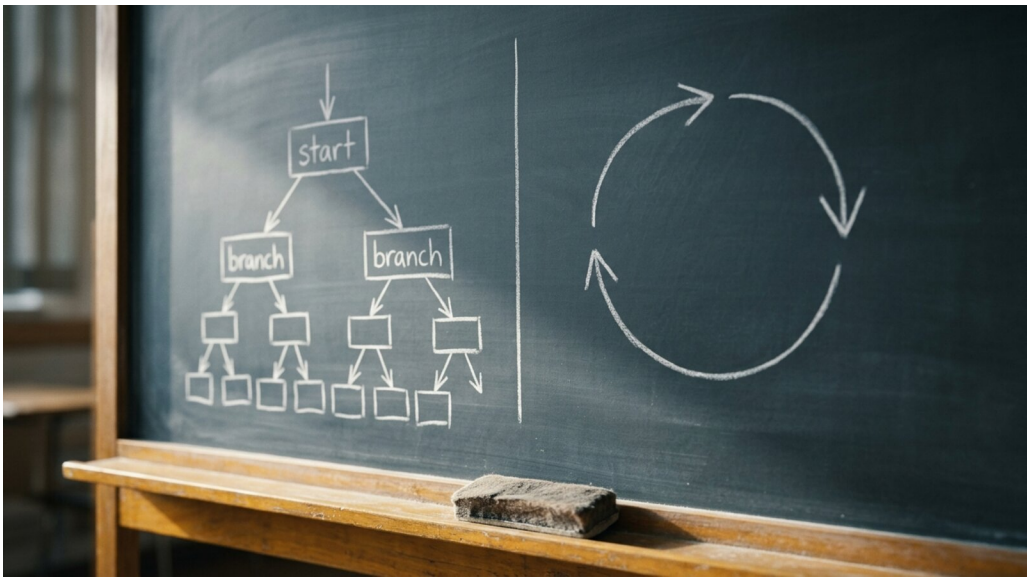
## Two Curricular Models

### The JHU Model: AI as Professional Infrastructure

Collado Torres's JHU 140.776 treated GitHub Copilot Pro as essential infrastructure, analogous to RStudio or Git: a professional tool students must learn to operate, not a shortcut. The course covered setup and configuration, critical evaluation of suggestions (with exercises exposing cases where Copilot generates plausible but incorrect statistical code), workflow integration (when to accept, modify, or reject a suggestion), and documentation practices for recording AI contributions in reproducible analyses. The pedagogical emphasis shifted from 'can you write this code?' to 'can you evaluate, debug, and improve AI-generated code?'

### The USC Model: Coding Without Learning to Code

A contrasting model appears in Bien and Mukherjee's (2025) account of a required MBA data-science course at USC. Rather than teaching R or Python syntax directly, the course teaches students to prompt GitHub Copilot to generate code, with class time devoted to specifying analytical intent, interpreting output, and catching errors, rather than writing code from scratch. The authors frame this as appropriate for a terminal audience (MBA students who will manage, not personally implement, analyses), and explicitly do not recommend the same approach for students who must become independent, accountable statistical programmers, which describes PHB 228's audience. The contrast is instructive: the correct AI policy for a statistical computing course depends on whether its graduates are expected to write and defend code themselves, or to direct others (including AI systems) who do.



## 'Why Learn Computing When AI Can Code?'

This question has been unavoidable in statistical computing courses since 2023. It is the computational analogue of 'why learn arithmetic when calculators exist,' but with greater force, because a language model can generate sophisticated statistical code that appears to work correctly. Five arguments recur across the courses surveyed.

**The validation argument.** You cannot evaluate what you do not understand. A bootstrap confidence interval that ignores clustering structure, a simulation study that skips common random numbers, or a mixed-effects model with a misspecified random-effects structure will all run without error. Only a trained statistician can identify these failures. PHB 228's Lecture 1 states this directly: 'AI-generated code may be syntactically valid but statistically incorrect.'

**The accountability argument.** In regulated settings (FDA submissions, clinical trial analyses), a statistician must be able to explain and defend every analytical decision. 'The AI suggested it' is not an acceptable answer to a statistical reviewer, and no regulatory framework currently contemplates AI-generated analyses the submitting statistician cannot fully explain. The ICH E9 guideline and its E9(R1) addendum on estimands presuppose human judgment at every stage.

**The judgment argument.** Choice of estimator, handling of missing data, model specification, and sensitivity-analysis design all require understanding of the scientific question and the data-generating process, not just syntactic fluency.

**The debugging argument.** When AI-generated code fails, the user needs enough understanding to diagnose and fix the problem. Debugging requires deeper knowledge than generation.

**The amplification argument.** AI is most powerful in the hands of a skilled practitioner. Reframed, the question is not 'AI versus human' but 'AI with a skilled human versus AI with an unskilled human.'

The *Journal of Statistics and Data Science Education* maintains a standing article collection, 'Generative AI in Statistics and Data Science Education' (established June 2025, edited by Juana Sanchez), that has published on exactly this shift from code production to code evaluation. Bien and Mukherjee (2025), discussed above, is one entry in that collection.

## Pedagogical Framings

Four metaphors recur in how instructors talk about AI integration, and a fifth has become necessary as the tooling has changed.

**AI as pair programmer.** The AI serves as 'navigator' (suggesting code, catching syntax errors) while the student remains 'driver' (making design decisions, evaluating suggestions). This avoids the false dichotomy of 'use AI' versus 'do not use AI,' and has been adopted informally at JHU and Berkeley.

**Bloom's taxonomy.** AI excels at the lower cognitive levels (Remember, Understand, Apply): recalling syntax, explaining concepts, applying standard methods. The higher levels (Analyze, Evaluate, Create) remain firmly in the human domain. The tension is that if students never practice the lower-level tasks themselves, they may lack the foundation needed for the higher-level work AI cannot do. This tension is unresolved in the literature.

**Driver versus passenger.** A student who prompts an AI and submits the output is a passenger who has abdicated intellectual responsibility. A student who treats AI suggestions as one input among many, while retaining control of analytical direction, is a driver. PHB 228's Lecture 19 offers a related image: 'A surgeon uses robotic assistance but understands every cut.'

**Scaffolded integration.** Restrict AI early, when foundational skills are still forming, then progressively permit or require it as those skills develop, analogous to introducing calculators only after arithmetic is secure. No major course has fully implemented this model as of the original report; JHU's 'require Copilot Pro throughout' approach is a deep-end alternative that assumes critical-evaluation skills develop through immersion instead.

**From autocomplete to agentic coding.** The pair-programming and driver/passenger metaphors were developed when 'AI assistance' meant inline autocomplete, most visibly GitHub Copilot's original line-by-line completions. By 2026 the terminology has shifted: agentic coding tools (Claude Code, Copilot's agent mode, Cursor's agent mode) plan multi-step tasks, edit multiple files, run tests, and iterate on failures with limited human intervention per step. The informal term 'vibe coding' describes accepting an agent's output with minimal review, precisely the passenger stance the driver/passenger metaphor warns against. This sharpens rather than resolves the pedagogical questions above: an agent that autonomously modifies a simulation study across several files raises the stakes of the validation, judgment, and debugging arguments, because errors can propagate further before a human reviews them. Few published syllabi currently distinguish autocomplete-style assistance from agentic, multi-step use, even though the appropriate level of human oversight differs between the two.

## Risks and Concerns

Five risks recur across the courses and literature surveyed, plus a sixth specific to agentic tools.

- **Over-reliance and deskilling.** Students who rely heavily on AI may fail to develop 'computational intuition,' the ability to anticipate what an algorithm will do, estimate how long it should take, or recognize implausible output. This mirrors documented risks of premature calculator reliance in mathematics education.
- **Inability to debug AI-generated code.** Such code often works in the common case but fails at small samples, near-collinearity, non-convergence, or boundary parameter values, precisely where statistical judgment matters most. PHB 228's Lecture 19 gives a concrete example: an AI asked for a low-rank matrix approximation used the wrong singular value decomposition convention, producing results 'subtly wrong in ways that would corrupt downstream inference.'
- **Academic integrity.** Traditional plagiarism detection cannot identify AI-generated code, and the 'permitted with citation' model shifts the integrity question from 'did you use AI?' to 'did you disclose your AI use honestly?'
- **Equity.** GitHub Copilot Pro and ChatGPT Plus are paid tools; students with fewer resources may be limited to weaker free tiers, compounding an existing 'rich get richer' dynamic in which stronger programmers get more out of AI assistance in the first place.
- **Hallucinated statistical methods.** AI tools can generate code for methods that do not exist, or that combine real methods incorrectly: confidence intervals that mix parametric and nonparametric logic invalidly, optimization routines with fabricated convergence criteria, test statistics with the wrong reference distribution, simulation designs with silently broken

variance reduction. These look reasonable and can pass superficial review; only deep methodological knowledge catches them.

- **Reduced review points under agentic coding.** Agentic tools compress what used to be several discrete human review points (accept one suggested line, run it, inspect the output) into a single review of a larger, multi-file change. A student who reviews a full simulation-study implementation only after it runs successfully may never see the intermediate step where a statistical assumption was built in incorrectly. 'Vibe coding' is an acute version of the deskilling and debugging risks above, worth naming explicitly rather than treating as a faster version of ordinary AI-assisted coding.

## Recommendations for PHB 228

These recommendations account for PHB 228's identity as a biostatistics graduate course, its current 'permitted with citation' policy, its existing AI content in Lectures 1, 2, and 19, and the professional context its graduates will enter (clinical trials, FDA submissions, epidemiology).

**Maintain 'permitted with citation,' with refinement.** The policy avoids the impracticality of a ban, the equity cost of requiring a paid tool, and the pedagogical risk of unrestricted use. Two refinements would strengthen it: specify what adequate citation looks like (tool, prompt, how output was modified, the student's own assessment of correctness), and differentiate by assessment type, restricting AI on early foundational assignments while permitting it fully later, with in-class midterms and finals serving as natural AI-free checkpoints.

**Strengthen the curricular content.** The existing bookend structure (Lecture 1, Lecture 19) is well designed. A middle touchpoint during simulation-study design (Weeks 4-5) would reinforce it: have students prompt an AI to design a simulation comparing two estimators, then critically evaluate the generated code for missing common random numbers, missing Monte Carlo standard errors, or inappropriate sample sizes, and correct it.

**Emphasize regulatory accountability as a central theme, not one argument among several.** Invoke the FDA explicitly: a reviewer may ask a graduate to justify any analytical decision, and 'the AI suggested it' is not an acceptable answer. Connect explicitly to ICH E9 and the estimand framework, and draw examples from the clinical trial context: interim analysis boundaries, missing-data handling in randomized trials, and survival-analysis assumptions.

**Prepare students to be AI-skilled professionals.** JHU's mandatory Copilot Pro is too aggressive for PHB 228's current context, but the underlying insight, that graduates without AI-assisted coding skills will be at a professional disadvantage, is correct. Normalize AI use in live-coding demonstrations, narrating the evaluation process explicitly, and frame AI proficiency as a career asset alongside R programming and reproducible workflows.

**Monitor peer institutions and professional guidance.** CMU 36-750 and UNC BIOS 735 now publish specific, restrictive AI-policy language directly in their syllabi and are worth watching as models. JHU 140.776 and Berkeley STAT 243 are also likely to keep evolving, though their most recent public policy language could not be confirmed in this pass. The *JSDSE* article collection on generative AI in statistics education is a useful ongoing source. Whether peer policies begin distinguishing autocomplete-style assistance from agentic coding tools is worth tracking as its own question, since almost none do so explicitly yet.

**Adopt the 'trained hands' framing as a course-level theme.** PHB 228's Lecture 19 already introduces AI as 'a power tool that requires trained hands,' superior to both the defensive ('AI is a threat to learning') and promotional ('AI will transform everything') alternatives. Elevating it to a syllabus-level statement, reinforced throughout the quarter, states the course's position plainly: this course trains your hands, AI tools are the power equipment used with them, and without the training the equipment is dangerous rather than transformative.

## Things to Watch Out For

1. **Not every claim below carries the same evidentiary weight.** CMU 36-750's ban and UNC BIOS 735's guardrails were confirmed against currently accessible syllabi in August 2026; Berkeley STAT 243, Michigan BIOSTAT 615, and UIUC STAT 428's policies were not re-verified and should be read as 'as reported in early 2025.'
2. **The JHU Copilot Pro requirement is offering-specific.** It is documented for the 2024 iteration under Collado Torres; whether it persisted into later years was not confirmed from public materials.
3. **A course studying AI is not the same as a course permitting it.** CMU's research engagement with LLM capabilities coexists with 36-750's ban on using those tools for graded work; do not conflate 'taught as content' with 'permitted for assignments.'
4. **'Permitted with citation' is not one policy.** UNC's version comes with AI-detection screening and Honor Court referral; PHB 228's version rests on student disclosure alone. Comparing courses by category label alone hides this difference.
5. **Autocomplete-era metaphors do not automatically extend to agentic tools.** A policy written with GitHub Copilot's original inline suggestions in mind may not anticipate a coding agent that edits several files and runs tests unattended.
6. **Course-specific policy text is often LMS-gated.** Several of the 'unverified' entries above are for courses that are demonstrably active but whose current syllabus text sits behind a login, consistent with a real policy that simply is not publicly checkable from outside the institution.

7. **Absence of a published policy is not evidence the course ignores AI.** Harvard's Biostatistics 776 and UCLA's Biostat 203B were directly fetched and confirmed to contain no AI-policy language, which is itself informative. That does not mean either instructor has no informal practice. It means whatever practice exists is not published.

## Lessons Learned

**Conceptual understanding:**

- The policy taxonomy (banned, permitted with citation, encouraged or required, taught as content) is useful but coarser than the reality; the same label can hide very different levels of institutional guardrail.
- Five named arguments (validation, accountability, judgment, debugging, amplification) are more durable in classroom discussion than one generic appeal to academic rigor.
- The JHU and USC models are not competing answers to the same question; they answer different questions, depending on whether a program's graduates will write code themselves or direct others who do.

**Technical and process skills:**

- Verifying a peer-course policy against training knowledge alone is not the same as verifying it against a current syllabus; the two produced materially different answers for three of the six peer courses checked.
- A specific, dated URL beats a general institutional claim. The CMU 36-750 and UNC BIOS 735 entries are stronger precisely because they point at a live, checkable document.

**Gotchas and pitfalls:**

- It is tempting to smooth an 'unverified' claim into a confident one once it has been repeated a few times in a document's own history; resist that, and keep restating the verification status explicitly.
- A course's absence from search results is evidence of an LMS-gated policy, not evidence that no policy exists.

## Limitations

- This is a survey of publicly available course materials, not an exhaustive or systematically sampled review of graduate statistical computing courses nationally.
- Three of the eight peer-course entries (Berkeley STAT 243, Michigan BIOSTAT 615, UIUC STAT 428) reflect the original early-2025 review and were not independently re-verified against a current public syllabus in the August 2026 pass.
- The JHU 140.776 Copilot Pro requirement is confirmed only for the 2024 offering; continuation into later years is unconfirmed.
- Duke STA 663, Harvard BST 260/262, University of Washington BIOST 534/561, and Vanderbilt's biostatistics computing courses were checked as candidate additions and confirmed active, but no course-specific, publicly accessible AI-policy statement could be located for any of them, so they are not included as verified sources.

- A subsequent, broader sweep of roughly twenty additional graduate biostatistics programs (Harvard, UCLA, University of Pennsylvania, University of Washington, Yale, Minnesota, Emory, Vanderbilt, Duke, Boston University, Pittsburgh, Iowa, Rutgers, Wisconsin, Colorado, Ohio State, USC, Brown, UAB, MUSC) found no additional course-specific policy that could be confirmed against an actual syllabus. Two courses, Harvard Biostatistics 776 and UCLA Biostat 203B, were directly fetched and confirmed to contain none. This does not establish that no other program has a policy, only that none was found publicly.
- One lead, a University of Pennsylvania course (BSTA6700) whose syllabus PDF resisted text extraction, is deliberately not cited: a search snippet suggested AI-friendly framing, but without a verified quotation from the document itself it does not meet this post's citation bar.
- The 'agentic coding' discussion reflects the state of the tooling as of 2026 and the terminology in current general use; it does not reflect any peer course's published policy language, since few syllabi currently address the distinction explicitly.

## Opportunities for Improvement

1. Confirm the CMU 36-350, Berkeley STAT 243, Michigan BIOSTAT 615, and UIUC STAT 428 policy descriptions against a current public syllabus, rather than carrying forward the early-2025 characterization.
2. Check whether JHU 140.776's Copilot Pro requirement persisted past the 2024 offering, ideally by contacting the department directly rather than relying on search results alone.
3. Track whether any peer course publishes a policy that explicitly distinguishes autocomplete-style assistance from agentic, multi-step coding tools, since this survey found none that currently do.
4. Revisit the *JSDSE* 'Generative AI in Statistics and Data Science Education' collection periodically; it is a standing collection, not a one-time publication, and will likely accumulate more course-level case studies like Bien and Mukherjee (2025).
5. Pilot the Weeks 4-5 simulation-study exercise proposed above in a PHB 228 offering and report back on what it actually surfaces in student work, rather than leaving the proposal untested.

## Wrapping Up

### What Did We Learn?

The peer-course landscape resists a single clean summary. 'Permitted with citation' is the modal policy, but it ranges from pure self-disclosure to institutional AI-detection screening with Honor Court referral behind the identical label. A course can simultaneously study AI as a research subject and ban its use on graded assignments, as CMU 36-750 currently does. And the tooling itself has moved fast enough that the metaphors written for autocomplete-era assistants (pair programmer, driver versus passenger) now need an explicit extension to agentic, multi-step coding tools that few published policies have caught up to yet.

For PHB 228 specifically, the survey does not argue for changing course. The current 'permitted with citation' policy, paired with the existing Lecture 1 and Lecture 19 bookend, is well positioned relative to peers. What it argues for is sharpening: naming the five arguments explicitly, treating regulatory accountability as a central theme rather than one argument among several, and elevating 'a power tool that requires trained hands' from a single lecture slide to a syllabus-level statement of the course's position.

## See Also

**Related posts:**

- Building a Statistical Computing Textbook in the Age of AI: the companion post on structuring a methods textbook around the same 'in the Age of AI' framing.
- Code Review for AI-Assisted R Package Development: a parallel framework for establishing ownership of AI-assisted code after it is written, complementing this post's classroom-policy focus on AI use while code is being written.

**Key resources:**

- CMU 36-750 syllabus (C. Genovese; accessed August 2026).
- UNC BIOS 735 syllabus (N. Rashid; accessed August 2026).
- Harvard Biostatistics 776 syllabus (R. Peng) (accessed August 2026): cited here as a verified negative, containing no AI-policy language.
- UCLA Biostat 203B syllabus (H. Zhou) (2023 winter offering; accessed August 2026): likewise a verified negative.
- University of Washington Center for Teaching and Learning: AI course policies (accessed August 2026): an example of an institution-level statement deferring the actual policy decision to individual instructors.
- Bien, J., and Mukherjee, G. (2025). Generative AI for Data Science 101: Coding Without Learning to Code. *Journal of Statistics and Data Science Education*, 33(2).
- *JSDSE* article collection: Generative AI in Statistics and Data Science Education (ed. J. Sanchez, established June 2025; accessed August 2026).

---

## Reproducibility

This post carries no executable analysis; it is a narrative translation of a source report. The source document and its full reference list live at `~/prj/tch/07-phb228-stat-computing/phb228-2026/lectures/report_ai_in_stat_computing_courses.md`.

| Item | Value |
|---|---|
| Source report | `report_ai_in_stat_computing_courses.md` |
| Report prepared | March 2026 |
| References updated | August 2026 |

| Item | Value |
|---|---|
| Blog conversion date | 2026-08-18 |

---

*Rendered on 2026-08-18 at 11:51 PDT. Source: ~/prj/rgtlab/posts/pub-ai-course-policy-survey/index.qmd*

---

## Let's Connect

Questions, corrections, or evidence on any of the 'unverified' entries above (particularly a current, public syllabus for STAT 243, BIOSTAT 615, or STAT 428, or confirmation of JHU 140.776's Copilot Pro policy past 2024) are welcome.

- **GitHub**: rgt47
- **Email**: Contact form

### Related posts in this cluster

This post is part of the *Quarto, R Markdown, and Publishing* series. Recommended reading order:

1. Multi-Language Quarto Documents on macOS
2. Rapid Conversion of Draft R Scripts to Formal Rmd
3. Building a Statistical Computing Textbook
4. Producing R Coding Videos: OBS Screencasts and Narrated Slide Decks
5. Hosting a Quarto Book Series on Netlify
6. **Generative AI Policies in Graduate Statistical Computing Courses** (this post)