統計程式能力定位與
生成式 AI 協作倫理

Statistical Programming Competency &
Ethical Collaboration with Generative AI

115 學年度第 1 學期 | 程式設計與統計軟體

課程講義

課程教師：吳漢銘

更新日期：2026 年 9 月 10 日

本講義的核心問題： 當 AI 已經可以快速產生 R/Python 程式時，學生真正需要建立的能力是什麼？本講義的答案是：能定義問題、理解程式、驗證統計結果、維持可重現性，並對 AI 協作的結果負責。

課程網站：115-1 程式設計與統計軟體

學習目標

完成本講義後，學生應能說明「會叫 AI 寫程式」與「具備統計程式能力」之間的差異，並建立可被檢驗的 AI 協作習慣。

- 辨識統計程式能力的主要構面：統計推理、程式理解、資料處理、除錯、可重現性與軟體工程習慣。
- 利用四級能力架構評估自己目前的位置，並設定可觀察的進步目標。
- 區分 AI 可以協助的工作與必須由人承擔的判斷與責任。
- 建立生成式 AI 使用的八項倫理原則：責任、透明、驗證、隱私、誠信、可重現、公平與最小權限。
- 能撰寫簡潔的 AI 使用聲明，並保留必要的 prompt、版本與人工驗證紀錄。
- 面對「AI 產生的程式可以跑，但統計上錯誤」時，能提出系統性的檢查流程。

1. AI 時代，為什麼還要學統計程式？

生成式 AI 已能完成大量語法層級的工作，例如產生函數、改寫迴圈、解釋錯誤訊息或撰寫繪圖程式。然而，統計程式設計的核心從來不只是「把語法打出來」。真正的能力在於把一個分析問題轉換為正確、可檢驗、可重現的計算程序。

AI 很擅長	AI 不能替你承擔
產生常見語法與範例	研究/分析問題是否定義正確
解釋函數與錯誤訊息	資料是否適合回答問題
提出多種程式寫法	統計方法與假設是否適切
重構、加註解、產生測試雛形	結果是否可信、是否被誤解
快速探索不熟悉的套件	研究倫理、隱私、授權與最終責任

判斷標準： 如果把 AI 拿走，你是否仍能說明「這段程式為什麼這樣寫、用了什麼統計假設、如何知道結果是對的」？若不能，代表目前主要是工具操作能力，而不是獨立的統計程式能力。

2. 統計程式能力的四個核心構面

構面	核心問題	可觀察行為

構面	核心問題	可觀察行為
A. 統計推理	我知道為什麼要做這個分析嗎？	能說明 estimand、假設、模型選擇、限制與解釋界線。
B. 程式與資料能力	我能把分析轉成可靠程式嗎？	能處理資料型態、缺失值、函數、控制流程、向量化與套件。
C. 驗證與可重現性	我如何證明結果可信？	會做 unit check、sanity check、模擬驗證、set.seed、session info、版本管理。
D. AI 協作素養	我能把 AI 用成助理，而不是代替思考嗎？	會提供 context、限制權限、檢查輸出、揭露使用方式並保存重要紀錄。

3. 四級能力定位

以下不是考試分數，而是一個「你能獨立做到什麼」的能力定位。AI 可以出現在每一級，但不能直接把使用者提升到更高一級。

層級	名稱	典型表現	AI 協作的合理角色
Level 1	執行者（Operator）	能執行既有 script、修改路徑/參數、看懂簡單輸出，但遇到錯誤高度依賴範例。	解釋語法、提供小範例；不應直接代做完整作業。
Level 2	功能性使用者（Functional User）	能自行完成資料匯入、整理、基本圖表與常見統計分析，會閱讀錯誤訊息。	協助除錯、比較寫法、補文件與測試。
Level 3	獨立統計程式設計者（Independent Statistical Programmer）	能把問題拆成模組、撰寫函數、驗證結果、管理專案並重現分析。	作為 pair programmer、reviewer、測試產生器；人主導架構與統計決策。
Level 4	專業/研究級（Professional / Research-grade）	能設計穩健 workflow、處理大型/複雜資料、版本控制、測試、效能、審計與團隊協作。	可使用 coding agents 執行明確任務，但需權限邊界、code review、provenance 與責任歸屬。

重要：「AI 產生了 200 行程式」不等於 Level 4。能力定位看的是使用者能否獨立理解、判斷、驗證、維護與承擔結果。

4. 統計程式能力自我檢核表

能力項目	L1	L2	L3	L4
讀懂程式	能辨識主要函數	能逐段解釋	能追蹤資料流與副作用	能做跨檔案 code review
寫程式	修改範例	獨立完成一般任務	函數化、模組化	設計可維護架構/API
除錯	依賴他人指出	讀 error/warning	建立最小可重現例	系統化定位、測試與回歸檢查
資料處理	基本匯入	整併/轉換/缺失	驗證 schema 與 edge cases	建立可靠資料 pipeline
統計驗證	看輸出數字	檢查基本假設	交叉驗證/模擬/sanity check	設計審計與敏感度分析
可重現性	儲存 script	固定 seed/路徑	Project + version control	環境鎖定、CI/自動測試
AI 使用	問答案	問解釋與除錯	用 AI review/測試並自行核對	設計 agent 權限、稽核與協作流程

5. 一個更實用的定位：沒有 AI 時也要通過的三個測試

1. 解釋測試：關掉 AI 後，你能逐段解釋程式在做什麼，以及每個統計步驟的理由嗎？
2. 修改測試：題目稍微改變（例如 factor 編碼、缺失值規則、模型假設）時，你能自己調整程式嗎？
3. 驗證測試：AI 給你一個看似漂亮的結果時，你能設計至少兩種獨立方法檢查它嗎？

6. 生成式 AI 的正確定位：從「代寫」轉為「協作」

建議將 AI 視為一名速度很快、知識廣泛、但會犯錯且不了解完整情境的助理。最佳工作方式不是「把問題丟給 AI → 接受答案」，而是形成有人工控制點的循環。

階段	人的主要責任	AI 可協助

階段	人的主要責任	AI 可協助
1. Define	定義問題、資料、estimand、限制	協助澄清問題、列可能遺漏
2. Plan	決定分析策略與驗證方法	提出替代方案、風險清單
3. Code	選擇架構、理解每段程式	產生草稿、函數、註解、測試
4. Execute	控制資料與執行權限	在允許範圍內執行命令/測試
5. Verify	統計與數值驗證	產生 edge cases、比較實作、review
6. Interpret	做實質判斷與適度推論	協助文字整理，但不可代替專業責任
7. Document	留下資料、程式、版本與決策紀錄	協助產生文件、變更摘要與使用揭露

課程口訣： 人定義（Define）→ AI 協作（Assist）→ 人驗證（Verify）→ 人負責（Own）。

7. AI 使用的「綠燈 / 黃燈 / 紅燈」判斷

區域	典型用途	原則
綠燈：一般可使用	解釋 error、比較函數、產生 toy example、加註解、改善命名、提出測試案例	前提是學生最後理解並驗證。
黃燈：可用但必須揭露/加強驗證	產生核心分析 code、模型選擇建議、結果解釋、整段重構、coding agent 修改多個檔案	需要保留重要 prompt、核對文件/數值、人工 code review。
紅燈：不應使用或需明確授權	考試禁用情境、代寫整份作業、偽造結果/資料/引用、上傳機密資料、繞過權限或學校規範	違反學術誠信、安全或資料治理。

8. 生成式 AI 協作的八項倫理原則

1. 人類責任（Human accountability）

AI 可以建議，但最終提交者必須對程式、分析、圖表、解釋與錯誤負責。不能以「AI 產生的」作為免責理由。

2. 透明與揭露（Transparency）

當 AI 對成果有實質貢獻時，應依課程、研究或機構規範揭露工具與用途。揭露重點是「用在哪裡、如何驗證」，不是列出所有聊天內容。

3. 驗證優先（Verification）

LLM 會 hallucinate；函數、套件、引用、統計結論與數值都必須能回到原始資料、官方文件或獨立計算驗證。

4. 隱私與機密（Privacy & confidentiality）

不要將學生個資、醫療/財務資料、未公開研究資料、密碼、token 或具識別性的原始資料任意送入外部模型。

5. 學術誠信（Academic integrity）

AI 不應取代課程要評量的核心能力。若作業的學習目標是撰寫函數或推導分析，完全交由 AI 代做會破壞評量的有效性。

6. 可重現與來源紀錄（Reproducibility & provenance）

保留資料版本、code、套件版本、模型/工具、重要 prompt 或 agent 修改摘要，以及人工驗證步驟。

7. 公平與偏誤（Fairness & bias）

AI 建議可能反映訓練資料偏誤；涉及分類、評分、推薦或敏感群體時，要檢查群體差異與不當代理變數。

8. 最小權限（Least privilege）

coding agent 只取得完成任務所需的檔案與執行權限。先 read-only，再 patch，再 execute；避免在整顆硬碟或含敏感資料的根目錄啟動 agent。

9. 「程式跑得動」不代表「統計上正確」

AI 最危險的輸出往往不是語法錯誤，而是「看起來合理、可以執行、甚至有漂亮圖表，但統計邏輯錯誤」的程式。

層次	檢查問題	例子
語法	程式能否執行？	函數名、參數、括號、套件
資料	資料型態與處理規則正確嗎？	factor 被當 numeric；NA 被無意刪除
設計	分析有尊重資料產生機制嗎？	paired data 當 independent；重複量測忽略 subject
模型	假設與 estimand 合理嗎？	線性/常態/獨立假設；錯誤 outcome 編碼

層次	檢查問題	例子
數值	能否用另一方法重算？	手算小例、模擬、第二套件、closed-form check
解釋	結論有超出資料支持範圍嗎？	association 說成 causation；p-value 誤解
重現	別人能重跑嗎？	seed、版本、資料與 preprocessing 記錄

10. AI 程式碼的六步驗證程序

1. Read：先閱讀，不直接全選執行；辨認資料輸入、主要轉換與輸出。
2. Reduce：把問題縮成小例子或 toy data，確認邏輯。
3. Reference：查 R help、套件官方文件或原始方法說明，核對函數與定義。
4. Cross-check：用第二種程式、手算、模擬或另一套件驗證重要數值。
5. Stress-test：測試 NA、空資料、極端值、只有一群、factor level 改變等 edge cases。
6. Explain：提交前，用自己的話說明程式、統計假設、限制與 AI 的角色。

```
# 一個簡單的 cross-check 思維
x <- c(2, 4, 7, 9, 13)

# 方法 1：使用 R 函數
mean(x)

# 方法 2：依定義獨立計算
sum(x) / length(x)

# 重要結果至少要能用另一條路驗證，而不是只相信 AI 產生的第一個答案。
```

11. AI 使用揭露：應該寫到什麼程度？

揭露的目的不是懲罰使用 AI，而是讓讀者/教師知道成果中的人機分工，以及哪些部分經過人工驗證。以下是可直接採用的簡潔格式。

情境	建議揭露
輕度使用	「本作業使用 ChatGPT 協助解釋 R 錯誤訊息與改善程式註解；分析方法與最終程式由本人完成並驗證。」

情境	建議揭露
程式協作	「本作業使用 Claude/ChatGPT/Gemini 協助產生部分 R 程式草稿與測試案例；本人逐段檢查、修改，並以官方文件及獨立計算驗證主要結果。」
Coding agent	「本專案使用 AI coding agent 協助修改/重構指定檔案與執行測試；所有變更均經人工 code review，重要分析結果另以獨立程序驗證。」

不建議的揭露： 只寫「有使用 AI」但沒有說明用途；或反過來貼出大量聊天紀錄卻沒有說明人工驗證。好的揭露應簡短、具體、可稽核。

12. 建議的課程 AI 使用規範（教師可直接採用或修改）

情境	建議規則
平時練習	原則上可使用 AI；學生必須能解釋所有提交程式並完成指定驗證。
程式作業	可使用 AI 作為助理，但若 AI 對核心 code 有實質貢獻，需附簡短使用聲明。
團隊專題	允許 AI 協作；需記錄工具、用途、主要 agent 修改與人工 review。
考試/能力檢核	依教師規定；若目標是測量個人程式能力，可明確禁止 AI、網路或自動補全。
研究/機密資料	須遵守資料授權、IRB/合約/機構政策；未經允許不得上傳外部 AI 服務。
責任歸屬	提交者對所有內容負責；「AI 寫的」不能作為錯誤、抄襲、假引用或資料外洩的理由。

13. 課堂討論：六個情境，你怎麼判斷？

情境	案例	討論方向
情境 A	同學不懂 lm() 的 residual standard error，請 ChatGPT 用不同例子解釋，最後再查 R help。	通常可接受；屬於學習與概念澄清。
情境 B	把作業題全部貼給 AI，直接複製 150 行 code，自己無法說明。	不符合能力評量目的；即使 code 正確，也不是獨立能力。
情境 C	AI 建議一個不存在的 R package；學生沒有查證，直接在報告引用。	驗證失敗；屬於 hallucination 造成的來源問題。
情境 D	把含學號、姓名、成績的 CSV 上傳公開 AI 服務，請 AI 分析。	高風險；涉及個資與資料治理。
情境 E	Claude Code 在 RStudio Project 中修改三個 .R 檔，學生逐一 review diff 並執行 tests。	可視為合理的 agent 協作；關鍵在範圍、review、測試與揭露。
情境 F	AI 產生 logistic regression，程式可跑，但把 0/1 outcome 方向解釋反了。	典型統計錯誤；說明為什麼「能執行」不足以證明正確。

14. 教師評量時，可以評什麼？

AI 普及後，單純看「最後 code 是否正確」會越來越難區分學生能力。可把評量移向可觀察的推理、驗證與修改能力。

評量面向	可觀察證據	建議比重示意
問題與統計思維	能說明為什麼這樣分析、假設與限制	25%
程式理解	可口頭/書面解釋核心 code 與資料流	20%
程式品質	正確、清楚、模組化、命名合理	20%
驗證能力	tests、sanity checks、交叉核對、edge cases	20%
可重現與 AI 揭露	Project、seed、版本、AI 使用說明	15%

15. 提交作業前的 12 項 AI 協作自我檢核

- □ 我可以不用 AI，說明這份分析要回答的問題。
- □ 我能逐段解釋提交的核心程式。
- □ 我確認 AI 提到的函數/套件確實存在且用法正確。
- □ 我檢查資料型態、NA、factor level 與單位。
- □ 我檢查統計假設與分析設計。
- □ 我至少用一種獨立方法核對重要數值或結論。
- □ 我測試至少一個可能的 edge case。
- □ 我沒有把密碼、token、個資或未授權資料送給 AI。
- □ 若使用 coding agent，我檢查了它修改的 diff。
- □ 我的程式可以在乾淨環境中合理重跑。
- □ 我遵守本次作業/考試允許的 AI 使用範圍。
- □ 若 AI 有實質貢獻，我已加入簡短、具體的使用聲明。

16. 20–30 分鐘課堂活動：辨識「AI 程式」的風險

教師提供一段「可以執行但有一個統計問題、一個資料處理問題、一個可重現性問題」的 R 程式。學生分組完成以下任務：

1. 不要先問 AI：先人工找出三個疑點。
2. 再使用任一生成式 AI 做 code review，但要求 AI 只列問題、不直接重寫。
3. 比較「人先找」與「AI 找到」的問題有何差異。
4. 修改程式並寫出兩個驗證步驟。
5. 最後用 2–3 句寫 AI 使用聲明。

活動目的： 讓學生體會 AI review 的價值，同時理解「先有自己的判斷框架」比單純把 code 丟給 AI 更重要。

17. 與統計專業倫理的連結

生成式 AI 並沒有取消原本的統計專業責任。美國統計學會（ASA）的統計實務倫理準則將責任涵蓋資料收集、處理、分析、解釋、呈現以及模型/演算法的開發與部署；這些責任在使用 AI 後仍然存在。UNESCO 的生成

式 AI 教育指引則特別強調 human agency、資料隱私、驗證機制與避免把高風險決策責任讓渡給 AI。ACM 的專業倫理也強調公共利益、避免傷害與專業責任。

統計人的底線： 工具可以改變，但「對資料、方法、結果與影響負責」不會因為使用 AI 而改變。

18. 本講義總結

不要只問	更應該問
「AI 能不能幫我寫？」	「這個任務真正要評量/解決的是什麼？」
「程式跑了嗎？」	「資料、統計、數值與解釋都正確嗎？」
「AI 很強，所以我不用學語法？」	「哪些基礎知識是我驗證 AI 的必要條件？」
「我要不要禁止 AI？」	「哪一部分應允許協作、哪一部分必須測量個人能力？」
「AI 出錯誰負責？」	「提交者如何建立可稽核的人工責任鏈？」

一句話總結： 在 AI 時代，統計程式教育的目標不是和 AI 比誰打字快，而是培養能夠定義問題、控制工具、驗證結果、維持可重現性並承擔專業責任的人。

參考資料

- American Statistical Association. Ethical Guidelines for Statistical Practice (approved 2022).
- UNESCO. Guidance for Generative AI in Education and Research (2023; UNESCO page updated 2026).
- Association for Computing Machinery. ACM Code of Ethics and Professional Conduct.
- 課程網站：115-1 程式設計與統計軟體。

版本提醒： AI 工具與學校/期刊政策會持續更新。實際課程規範應以授課教師當學期公告為準；研究工作則需同時遵守資料授權、研究倫理、機構與期刊規範。