

Supplement A: Participant's Claude project instructions

For the article Coexistence of mathematical reasoning and cognitive offloading: A longitudinal case study of one student's chatbot use across a term of proof-based mathematics. Submitted to *Educational Studies in Mathematics*

Jonathan Cardoso-Silva^{1,*} and Johannes Ruf²

¹Department of Methodology, London School of Economics and Political Science, London, United Kingdom

²Department of Mathematics, London School of Economics and Political Science, London, United Kingdom

*Corresponding author

ORCID: Jonathan Cardoso-Silva: 0000-0002-9566-7400; Johannes Ruf: 0000-0003-3616-2194

Correspondence

Jonathan Cardoso-Silva

j.cardoso-silva@lse.ac.uk

Compiled 12 October 2026

A.1 The project instructions

Here is the system prompt used by the participant in her Claude project for the module. The only edit the student made to the template that comes with Anthropic's study project was the line "When you write mathematical formulas, please make sure to use the correct latex formulation."

Instruction · Claude project instructions

You are assisting a student with their academic learning for their course: MA421. Follow
→ these guidelines:

When you write mathematical formulas, please make sure to use the correct latex formulation.

1. When to act as a tutor vs. direct assistant:
 - Act as a tutor when students are asking for help understanding concepts, working
→ through problems, or seeking explanations
 - Provide direct assistance when students explicitly request creation of study materials,
→ summaries, or organizational tools
 - If unclear which mode is appropriate, briefly ask for clarification
2. As a tutor (for conceptual questions and problem-solving):
 - Use leading questions rather than direct answers
 - Ask targeted questions that guide students toward understanding
 - Provide gentle nudges when they're headed in the wrong direction
 - Balance between pure Socratic dialogue and direct instruction
 - Do not solve homework problems directly
3. As a direct assistant (for resource creation):
 - When students request specific tools like flashcards, study guides, or outlines, create
→ them directly
 - Focus on delivering what was requested efficiently
 - Organize information clearly and effectively for the requested format
 - Include examples and illustrations where appropriate
 - Don't delay creation by asking multiple preliminary questions unless essential
→ information is missing
4. For study materials creation (in either mode):
 - Organize information hierarchically with clear relationships between concepts
 - Include multiple learning modalities (visual, verbal, examples)
 - Create materials that encourage active recall and spaced repetition

- Adapt explanations to the apparent knowledge level of the student

5. When helping a struggling student:

- Break down complex ideas into simpler components
- Relate new concepts to those they likely already understand
- Use analogies and real-world examples appropriate to their field of study
- Suggest different approaches if one explanation isn't working

6. Always prioritize clear communication and effective learning tools over lengthy
↳ explanations.

7. If the student uses a sample prompt (Create study flashcards, Visualize key concepts,
↳ Build comprehensive study guide) without uploading study materials:

- Politely inform them that you notice they haven't uploaded any course materials
- Present two clear options:
 - a) They can return to the Project home page, upload their course materials, and then
↳ rerun the prompt for personalized assistance
 - b) You can provide a sample demonstration based on general MA421 concepts to show how
↳ the tool works
- If they choose option b, create a demonstration using common concepts from that type of
↳ course
- Keep your sample demonstration relevant to their course subject but generic enough to
↳ be universally applicable

Supplement B: Code scheme for chat exchanges

For the article Coexistence of mathematical reasoning and cognitive offloading: A longitudinal case study of one student's chatbot use across a term of proof-based mathematics. Submitted to *Educational Studies in Mathematics*

Jonathan Cardoso-Silva^{1,*} and Johannes Ruf²

¹Department of Methodology, London School of Economics and Political Science, London, United Kingdom

²Department of Mathematics, London School of Economics and Political Science, London, United Kingdom

*Corresponding author

ORCID: Jonathan Cardoso-Silva: 0000-0002-9566-7400; Johannes Ruf: 0000-0003-3616-2194

Correspondence

Jonathan Cardoso-Silva

j.cardoso-silva@lse.ac.uk

Compiled 12 October 2026

About this supplement

This supplement presents the prompt and response types used in the article: the prompt type records what the student was trying to get the chatbot to do, and the response type records what the chatbot provided. The definitions below state the adaptation used for the mathematics chatlogs; the worked examples come from those chatlogs.

The scheme was developed as part of the GENIAL project. The first author developed the scheme inductively, from his first-hand reading of students' chatlogs in the programming course, his interactions with those students, and samples of their Git histories. The first author and a research assistant then coded three full chatlogs independently and revised the codes until their inter-rater reliability (κ) was at least 0.7 for both prompt and response types. This reliability check concerned the original programming-course study, not the mathematics case reported here.

For this article, we applied the prompt and response types deductively. We omitted the programming-specific prompt type `DEBUG`, language-feature flags, action codes, and Git-commit labels. Instructions for the original coding app and examples involving programming assignments are also omitted here.

B.1 Coding an exchange

Read each student message and the chatbot's reply in the context of the preceding conversation. Assign a primary prompt type using Section B.2 and a primary response type using Section B.3. Code the request and the reply separately: a request for a solution need not receive a finished answer.

When a message combines functions, choose the dominant one. The original scheme permits an optional secondary code for a weaker function; the counts reported in this article use the primary codes. A final question does not by itself make a reply a `REDIRECT`: consider whether the chatbot has already supplied the substance of the requested explanation. Section B.4 illustrates these decisions.

The codes describe the interaction, not its success. An `EXPLANATION` does not establish that the student understood it, and an `ARTIFACT` need not be mathematically correct. As described in the article, mathematical and curricular adequacy, the student's visible reasoning, uptake, verification, and the resolution of an episode were examined separately.

When coding each exchange, we checked which lecture notes, problem sets and problem-set solutions the student had by then. Lecture notes and the matching problem set were released at 14:00 UK time on the Monday of the teaching week for that lecture, and solutions on the Friday of the same week, so a Monday exchange before 14:00 belongs to the previous week's materials. Because Week 6 was reading week, lecture N falls in LSE week $N + 1$ from Lecture 6 onwards.

B.2 Prompt types

Code what the student is trying to obtain, rather than particular words such as “how” or “why”. The distinctions below follow the prompt types in the article’s code table.

CHECK

The student asks the chatbot to verify something they have done or to confirm their understanding. There is a claim or an attempt to assess. A request to explain an unfamiliar result without such a claim is **EXPLAIN**.

REFINE

The student asks the chatbot to modify, improve, or extend something it previously produced. The request builds on an existing answer; a request for a complete answer from scratch is **SOLUTION**.

EXPLAIN

The student asks for an explanation of a concept, derivation, or result. The main request is to understand the mathematics. If the student instead asks the chatbot to supply a complete worked answer, use **SOLUTION**, even if an explanation is also requested.

CONCEPT

The student asks more abstractly about a strategy, an approach, or why one method is preferable to another. The focus is broader than a specific step they are working through. A request to unpack that specific step is **EXPLAIN**.

META

The student reflects on their own working process or asks for help thinking about how to approach their work. The focus is on studying, organising, or managing their reasoning, rather than on the mathematical content of a particular step.

SOLUTION

The student asks for a complete solution or worked answer. The defining feature is the request to have the answer supplied, rather than to receive guidance on how to proceed.

HOWTO

The student asks how to perform a procedure or follow a sequence of steps, without showing their own attempt. Distinguish a request for steps to follow from a request to understand why an argument works (**EXPLAIN**) or to have the full answer supplied (**SOLUTION**).

RESPOND

The student answers a question the chatbot posed in the preceding turn. An answer can include an objection or a request to clarify the same step. If a new request becomes the dominant purpose of the message, code that request instead.

OTHER

None of the other prompt types fits. Use this for messages without a discernible request or answer, rather than forcing them into a substantive category.

B.3 Response types

Code what the chatbot actually provides, rather than what the student requested or what the chatbot says it intends to do.

EXPLANATION

The reply explains a concept, answers a question, or describes how something works. It may use a worked example to illustrate the reasoning. The presence of calculations alone does not make it an **ARTIFACT**: the distinction is whether the reply explains the mathematics or supplies a finished answer to the student's task.

ARTIFACT

The reply supplies a finished or nearly finished answer that the student could submit largely as given. An explanation may accompany it. Judge the extent of the answer supplied, not whether it is correct or suitable for the course.

REDIRECT

The reply withholds a direct answer and hands the next step back to the student, through a guiding question or a direction to consult course material. It may first provide some explanation. The key is that the student is left to carry out the substantive next step.

CORRECTION

The reply identifies an error in the student's reasoning or flags a misconception. A correction targets a particular fault; a reply that replaces the whole answer may instead be an **ARTIFACT**. Where correction and explanation are combined, apply the dominant-function rule in Section B.1.

CLARIFICATION

The reply asks for further information or confirmation about the student's request before proceeding. Distinguish this from a **REDIRECT**, which asks the student to do the next mathematical step. Whether the requested clarification was actually necessary is a separate question about the reply's adequacy.

AFFIRMATION

The reply confirms or reassures without adding substantive new information. If it confirms and then adds an explanation, correction, or other substantive contribution, code by that contribution.

B.4 Worked examples from the mathematics chatlogs

The following examples illustrate the primary coding of five exchanges. Exchange identifiers refer to the chatlog corpus supplied with the article. Quoted student messages retain their original wording; descriptions of the chatbot's replies are summaries unless quotation marks are used.

B.4.1 A redirect and a subsequent explanation: c072 and c074

In c072, the student asks about finding a duplicate in an unsorted stream: "I'm confused on the finding a duplicate (multi-pass). can you explain to me what it does? since its not ordered, can't the duplicates just be inconveniently placed so we never find them?" The prompt is **EXPLAIN**: she wants to understand the algorithm. The chatbot introduces a pigeonhole argument, splitting the range of possible values in half, but asks the student to determine what follows from having more items than possible values. The primary response is **REDIRECT**. There is explanatory material, but the chatbot leaves the conclusion to the student.

After another exchange, the student replies in c074: "but I the stream coming in is 22341, my lower half would be 22 and my upper 341, no? or how do I find the mid - what would it be here". The prompt is **RESPOND**: she is answering the chatbot's preceding question and challenging the split it proposed. The chatbot explains that the split is by value, not position, and works through her stream: three items lie in $\{1, 2\}$ and two in $\{3, 4\}$, so the lower range must contain a duplicate. It then explains how to narrow the range for another pass. The primary response is **EXPLANATION**. Its closing question about the next pass does not displace the explanation already supplied.

B.4.2 A correction followed by a question: c093

The student gives an algebraic expression in answer to a question about a probability bound, so the prompt is **RESPOND**. The chatbot opens "Not quite — be careful with the algebra", restates the fraction to be simplified, and asks her to simplify it step by step. The primary response is **CORRECTION**: it identifies her answer as erroneous and directs attention to the algebra. The further question is a redirecting element, but it does not determine the primary code on its own.

B.4.3 An unnecessary clarification: c131

The student asks "can you explain the setup of $\text{Rev}(B) = \dots$ in d)", coded **EXPLAIN**. The chatbot suggests that she may be mixing up parts (c) and (d), then asks whether she means part (d). The primary response is **CLARIFICATION**, because it seeks confirmation of which part she means before proceeding. The student had already specified part (d). The code therefore records the reply's function without endorsing the need for the question.

B.4.4 A complete worked answer: c133

Continuing the same episode, the student says "don't ask, just show". This is **SOLUTION**: she asks the chatbot to supply the working rather than another question. The chatbot derives the winning bidder's payment in region B , sets up the expected-revenue integral, explains its factors, and evaluates it as $1/24$. The primary response is **ARTIFACT**. Although it includes explanation, it supplies the complete derivation requested. This differs from c074 (Section B.4.1), where the numerical example illustrates how the algorithm works. The response code alone does not establish what the student subsequently understood or did with the derivation.

Supplement C: Prompts used in the analysis

For the article Coexistence of mathematical reasoning and cognitive offloading: A longitudinal case study of one student's chatbot use across a term of proof-based mathematics. Submitted to *Educational Studies in Mathematics*

Jonathan Cardoso-Silva^{1,*} and Johannes Ruf²

¹Department of Methodology, London School of Economics and Political Science, London, United Kingdom

²Department of Mathematics, London School of Economics and Political Science, London, United Kingdom

*Corresponding author

ORCID: Jonathan Cardoso-Silva: 0000-0002-9566-7400; Johannes Ruf: 0000-0003-3616-2194

Correspondence

Jonathan Cardoso-Silva

j.cardoso-silva@lse.ac.uk

Compiled 12 October 2026

C.1 Scope

After coding all 196 exchanges as described in the article, the first author used Claude Opus 4.6 to produce chronological annotations and Level 1 codes for each exchange from the original chatlogs and course materials. The first author coded the same exchanges independently in a separate application, and checked each annotation against the source. Where the two sets of codes diverged, the first author's codes were the primary record. After the interview, we prompted Claude with the transcript alongside one window of exchanges at a time, asking it to identify what the interview had to do with each exchange. The model quoted the participant verbatim and named candidate topics grounded in her words. The first author approved each theme-log entry before it was written. Interview tag blocks were reviewed subsequently in the compiled annotations. An earlier attempt to supply the annotations and transcript together and ask for candidate themes produced summaries that flattened contradictions or misrepresented the participant's perspective, and those were set aside. We then used Claude to revise the findings in separate sessions, supplying the accumulated log without any chatlog or annotation text. The first author decided which claims entered the paper. No theme was retained, renamed, or removed without the first author's approval, and no claim produced by the model was kept without verification against the underlying chatlogs and transcript.

Prompts A.1–A.4 restate the workflow instructions that governed each analysis step named in the article, together with illustrative outputs retained for each step. The bordered Instruction blocks reproduce the production instruction text used in the analysis. An early Level 1 checklist that existed only in the Claude Project interface is not reproduced here; Prompt A.2 states the annotation and coding rules used in the chronological reconstruction.

Table C.1: Analysis steps and prompts.

Step in the article	Prompt
Provision each analysis window	Prompt A.1
Chronological annotation and Level 1 coding	Prompt A.2
Interview tagging, one window at a time	Prompt A.3
Findings revision from the theme log alone	Prompt A.4

C.2 Prompt A.1: Window preparation

Purpose

Cut the 196-exchange timeline into contiguous materials windows and build empty LaTeX slice shells so Prompt A.2 can annotate without assembling sources in the chat.

Stage card

- **Actor.** First author, with scripting help to export timelines and slice shells.

- **Model.** None.
- **Inputs.**
 - Merged coding timeline and Level 1 database for all twelve Claude sessions (196 exchanges, c000–c195).
 - Temporal release rules: which lecture notes and problem-set solutions were eligible in each LSE week.
 - First author’s Level 1 codes and margin notes where already filled.
 - Annotations-book LaTeX structure (main file, slice folder, exchange macros).
- **Outputs.**
 - One prep folder per window: full chat timeline, margin-note export, short corpus brief naming eligible course files.
 - One LaTeX slice per window: student and assistant bubbles filled, coder notes filled where they exist, empty annotation braces for every exchange.
 - Agreed window cut (exchange range and materials eligibility) recorded before handoff to Prompt A.2.
- **Human approval.** First author confirms each window cut and any decision to leave a range uncoded at Level 1 before the slice is handed to Claude.

Instruction (production contract)

Instruction · Window preparation: stage table

Stage table

- **window** — Fix materials cut and exchange range. Actor: either; Tooling: Read manifest and timestamps; propose cA–cB and eligible files; Outputs: agreed range + eligibility note.
- **gaps** — Check Level 1 coverage. Actor: agent; Tooling: an export script that reports what is still uncoded for the range; Outputs: report of what is still uncoded.
- **pilot-files** — Timeline and margin notes for the range. Actor: agent; Tooling: an export script (Markdown output); agent drafts a corpus brief for the first author to check; Outputs: a timeline, the coder’s margin notes, a corpus brief.
- **latex-slice** — LaTeX slice shell and plumbing. Actor: agent; Tooling: an export script (LaTeX output) (bubbles, {coder} codes/notes, empty \exchangeannotation{}); annotate title, Pattern, LSE week; add it to the annotation slice files; register it in the annotations book and in Claude’s instructions; Outputs: a slice file ready for annotation, with instructions.
- **handoff** — Point Claude. Actor: human; Tooling: the first author opens Claude with instructions and first pending id; Outputs: annotation session starts.

Actor: human — agent — either. Stage ids listed in `human_only_stages` in the workflow registry must not be attempted by the agent (checklist only).

Cadence: One materials window per invoke unless the first author asks to batch. Stop after `latex-slice` for the first author’s go before handoff. The whole procedure runs again for the next window once the current one is handed off (see Open below for the run history). Each window is a fresh invocation of the whole procedure, not a return to an earlier stage within one invocation. Human stops (confirm, fill in what is still uncoded, handoff) use the workflow progress board, including actor and step reminders on each stage row.

C.3 Prompt A.2: Forensic annotation and Level 1 coding

Purpose

For each exchange, write a chronological annotation that reconstructs the student’s position and justifies Level 1 codes.

Stage card

- **Actor.** Claude (Opus 4.6) drafts; first author reviews exchange by exchange.
- **Unit of work.** One exchange per draft–review cycle.
- **Inputs.**
 - Pseudonymised student and assistant messages for the current exchange (already in the slice).

- First author’s margin note and Level 1 codes for that exchange, when present.
- Corpus brief and eligible lecture notes or problem sets for that point in the term.
- Level 1 codebook: GENIAL fields `prompt_type` and `response_type` (v0.2.1).
- Reader model and annotation structure instructions.
- **Outputs.**
 - Filled annotation brace: Exposure, Student state, Response, Codes.
 - Level 1 labels on the exchange (`prompt_type`, `response_type`, secondary when needed).
 - Friction notes when the codebook did not fit cleanly (no new labels invented).
- **Coverage.** Annotations for exchanges c000 through c195 across fifteen windows.
- **Primary record on disagreement.** First author’s codes, not Claude’s.
- **Human approval.** First author reviews each annotation before the next exchange is written; final codebook adjudication stays with the first author.

Instruction (production contract)

Instruction · Forensic annotation: reader model and annotation rules

Reader model (fixed for this workflow)

State this once per session, not per exchange. Correct it when the first author corrects it.

1. **Who reads this.** The first author (computer scientist, educational researcher, data scientist) and the second author (mathematician). The first author does not keep all of the MA421 details in working memory but can follow them when re-stated. The second author taught the course but may not remember every example cold.
2. **Arrival context.** Reading a compiled LaTeX appendix. The student and assistant bubbles are above the annotation. The reader has already seen the chat text.
3. **Reading purpose.** First: what was the student trying to do here, and what does her message tell us about where she is in her understanding. Second: are the codes defensible, and did the chatbot help or hinder. The annotation reconstructs the student’s position, not the chat content.
4. **Tolerance.** High for mathematical content, zero for padding and repeated chat content. Enough maths that the first author can follow the student-state reconstruction without reopening the lecture PDF, but do not re-explain what the bubbles already show.
5. **Speaker.** Annotator as colleague briefing two co-investigators. Direct claims, evidence tags where inference is required.

Annotation rules

These are the lessons from the c0 pilot session, promoted to rules.

Voice

Write as a colleague briefing the first author and the second author, not as an agent filing a report. The annotation reconstructs what the student was trying to do, not what the chat transcript says. Do not describe what the reader can already see in the bubbles above.

Structure

Four paragraphs per annotation, each led by a short bold label and separated by blank lines inside the `\exchangeannotation{}` brace group.

1. **Exposure.** When in the term this exchange falls. What course material the student has access to. Keep this short and factual.
2. **Student state.** What the student is trying to do. What her message shows about her understanding or confusion. Re-state enough of the mathematical content that the first author can follow without reopening the PDF, but do not re-derive what the bubble shows. Assume she has opened the relevant PDFs unless the chat indicates otherwise.
3. **Response.** What the assistant did: accurate or inaccurate, appropriate scaffolding or not, Socratic hold-back or full answer. Note Learning-mode features when visible. If the same Learning-mode pattern repeats across two or more consecutive exchanges, this paragraph may compress to a single sentence naming the pattern and citing the first

exchange it appeared in (for example, "Same Socratic hold-back as c14."), rather than re-describing it in full each time.

4. **Codes.** Codebook labels with brief justification. Primary and secondary where needed. End with "Cite cN."

Emphasis

When something in the exchange illuminates the phenomena under study, whether an action by the student, an inaction, or a chatbot response that could shape how the student works, wrap the observation in `\exchangefinding{...}` (subtle Atelier ink wash, like `blog .ink-highlight`). Use it when the observation matters for the study, not for decorative stress.

What not to do

- Do not assume the first author knows MA421 content. Re-state the mathematical claim enough to follow the student-state argument.
- Do not over-explain what the bubbles show. The reader has the chat text.
- Do not write `[agent inference]` for the assumption that the student opened the PDF. That is the default. Flag `[agent inference]` only when inferring something not grounded in the timeline, corpus brief, or prior turns.
- Do not use catalogue syntax, contrastive setups, or spec-outline formatting in the annotation.
- Do not continue to the next exchange without the first author's go.
- Do not put `\textbf{...}` inside `$. . . $` in student or assistant bubbles. Use `\textbf{$. . . $}` for bold maths.

Codes

Assign `prompt_type` and `response_type` from the Level 1 codebook (v0.2.1 `genial-codebook-level1.qmd`). Use the v0.2.1 labels (DELEGATE, ARTIFACT, REDIRECT, etc.), not the retired v0.1 labels (PASTE, CODE). Use secondary codes when the primary does not fully capture the response. Place codes in `\exchange{...}` and justify briefly in the Codes paragraph.

The first author's codes in the margin notes are a reference. Disagree in the annotation if the codebook decision rules point elsewhere, and say why.

Register

Apply the writing register and phrase translations by hand. Run the banned-terms grep in the sandbox after drafting. The file still needs a Vale pass in Cursor before it counts as clean.

C.4 Prompt A.3: Interview–exchange tagging

Purpose

After the interview, set the transcript beside one annotated window at a time: write tag blocks where the interview bears on an exchange, then write that window's theme-log entry. Findings revision is Prompt A.4 and runs in a different session.

Stage card

- **Actor.** Claude drafts tags and the log entry; first author rules on the log entry before filing.
- **Unit of work.** One slice (materials window) per session.
- **Inputs.**
 - Full verbatim interview transcript (interview ran 8 July 2026; filename dated 7 July after a cancelled booking).
 - One completed annotations-book slice: bubbles, first-author codes and notes, forensic annotations.
 - Conventions for tag blocks and quotation macros.
 - After tags are drafted: the theme log so far (for recurrence counts), not before.
- **Withheld.**
 - Findings section of the article.
 - Any numbered theme list or theme titles from the article.
 - Codebook, corpus manifest, prep folders, and course PDFs (already used in Prompt A.2; tags work from the annotation on the page).

- **Outputs.**

- Zero or more interview tag boxes under exchanges the interview bears on (label line plus short paragraphs; verbatim quotes).
- One chronological theme-log entry for the slice, with cumulative recurrence tables updated.
- Explicit “Not in the interview” paragraphs where the annotation found something the transcript never covers.

- **Human approval.** First author approves each log entry before it is written to the file. Tag boxes had no separate review step before filing. The first author read them in the compiled book and could revise later.

Instruction (production contract)

Instruction · Interview tagging: writing rules

What this workflow is for

The tag blocks go in the compiled book, one under each exchange, after the navy forensic annotation and the burnt-red coder margin note. The first author and the second author read them there while reading the corpus.

A block says what the interview has to do with the exchange above it. Three things qualify:

- **What she described, happening in the trace.** Quote her, then name the feature of the exchange that corresponds.
- **Where the trace and her account part company.** She said the questioning mode helped her start problems, and here it withheld a step she had already identified. Divergences are worth as much as matches.
- **What the annotation found that she never mentioned.** The first author, the second author, or the forensic annotation noticed something, and the interview says nothing about it. This is a finding about the limits of a retrospective account, and it belongs under its own label in the block.

Name candidate topics in plain words. Where an observation has no name yet, propose one in the block: “we could call this the cost of typing mathematics”. Naming a topic here proposes it. It does not register it. Whether a topic becomes a theme is settled at interview-theme-memo, with the first author.

What the log entry is for

The theme log is the accumulating record, one entry per slice. Its own header states the three writing rules and they govern, not this file.

The log is mostly for the agent, and for the first author months out when he writes this up. It is chronological because the corpus is. An entry says what the student did, what the response did, and how the two relate, exchange by exchange, with her words quoted wherever a finding opens on something she said. Its two cumulative tables at the bottom are the count that decides whether an observation is on its third appearance.

A verdict does not belong here. The log records what an exchange shows, and the memo states what it proves.

What the agent does not read

The Findings draft, at any stage. The candidate themes there are readings of one interview, six from a single pass in July and the rest added since. An agent that opens this slice already knowing them writes tags that confirm them, and a log entry that records which theme each exchange supports instead of what each exchange shows. Reading the slice without the memo is what makes a tag evidence and not an illustration. Only the Findings revision (Prompt A.4), with the first author, edits that file.

The theme log, before the blocks are written. The log is this workflow’s own second output, so it is read, but not first. Reading it before tag-slice tells the agent what earlier slices found and what to look for. It opens at map-annotations, once the blocks are done.

Do not open the Findings draft at any stage of this workflow, and do not open the theme log before tag-slice is complete.

The codebook, the corpus manifest, the prep folders, the course PDFs. Those were read at the forensic annotation stage. This workflow works from the annotations already in the slice. When a tag needs a claim about course content the annotation does not make, the annotation was incomplete, and the agent says so instead of opening the PDF.

Why the writing rules are what they are

The writing-for-a-reader instructions are binding at tag-slice and checked at reader-check. They fix who is speaking, who reads the block, what the block has to do on its own, and what counts as a checked fact. The rules in this section are what those four answers come to for this workflow, and they do not replace those instructions.

This workflow puts two sources of evidence next to each other, and neither one tells the whole story on its own. The work is to find where they agree, where they contradict each other, and where the interview says nothing about something the exchanges show.

What she said is retrospective self-report from one conversation, shaped by what was salient in July, by the account she has built of herself as a student, and by who she was talking to. What the chatlogs show is behaviour without intent. The prompt is visible. Whether she understood, what she did offline, and why she stopped are not.

Neither is the truth. The relation between them is what this study examines, and a relation can only be examined while both terms are still on the page.

Quote her, do not summarise her. A summary has already committed to a reading, and once committed the original is gone. "The same mismatch of level she complained about in professors" is a claim about what she meant. Her sentence lets a reader check it.

Record the observation, not the verdict. Say what the exchange shows and what she said. The judgement about what that settles belongs to the first author, at synthesis or in the write-up.

Name which object the finding is about. Three appear in this corpus: the student, the way the tool is configured, and the teaching material. A finding about Learning mode filed as a finding about her learning misdescribes what the study observed.

Do not grant the tool agency. This study examines how a student attributes capability to a tool. Write "the response explains" and "a response that told her to stop", not "Claude explains".

Judge the tool against its own configuration. This is vanilla Claude with a study template, not a course tutor. It has no syllabus, no lecture notes, and no way to know that a supervisor exists. Faulting it for not routing her to the second author judges it against something it could not do. Ask what this setup could have done.

Who reads the blocks

The first author (computer scientist, educational researcher, data scientist) and the second author (mathematician), in the book. The second author was not in the interview and has not read the transcript.

What the reader has on the page: the student's message, the reply, the first author's Level 1 codes and margin note, and the forensic annotation directly above the box. Nothing else. Not the transcript, not another slice, not a block read twenty pages earlier. A block that depends on any of those has failed before it is read. Four rules follow. Most drafts fail the first.

Establish the interview the first time a block mentions it. Say that an interview took place, when, and what she was being asked about at the moment quoted. "In an interview in July, after the term ended, she was asked when the Socratic questioning helped her" costs one clause and gives a reader who knows none of this a quotation they can use. A block opening on a bare "she said" reads as though the reader missed a page.

The block decides the order. Some blocks open on her account and turn to the exchange: "In the interview, the student said X. This exchange shows Y." Others open on what the exchange does and bring her in second, which suits a block whose finding is about the exchange and where her words corroborate it or complicate it. Choose by which of the two the block is mainly about. Do not apply one order to every block in a slice.

Use the words this corpus already uses. The forensic annotations, the first author's margin notes and the student's own prompt at c17 all call the behaviour Socratic. The first author's margin note at c0 identifies Claude Learning mode. The interview confirms the setup she built. Write Socratic, Claude Learning mode, or the study template. Do not invent a neutral phrase to cover a distinction you have not checked: an invented term reads to the second author as though the book defined it somewhere he missed.

Say what she was asked, and mark what she volunteered. A sentence about who asked, answered, raised, or conceded is a factual claim about the interview. Those go wrong more often than any other kind, because the verb feels like connective writing. When the transcript shows her offering something without being asked, say so, because a point volunteered is stronger evidence than one drawn out. When the first author's margin note asks a question the interview never put to her, the block says the question was never put to her and belongs under `\textbf{Not in the interview.}`

What goes inside `\exchangeinterviewtag{}`

The box is titled "Interview theme tag (Claude)". The agent drafts what is in it, and the first author reads every block in the compiled book and revises it where needed.

Structure. A bold label line, then one short paragraph per point.

1. **Label line.** `\textbf{Bears on:}` followed by what the block covers, in plain words. Name one thing and stop. A trailing clause beginning `and the` or `which` promises a second topic and restates the first. The second thing goes in a paragraph, which needs a verb and a claim of its own to stand. Where a candidate topic has a name from an earlier slice, use that name. Where it does not, propose one.
1. **Paragraphs.** Two to four sentences each. Every quotation goes through `\sq{}` and names what she was being asked about at the time. Within the paragraph, her account may come first or the exchange may come first, per [section] Who reads the blocks.
1. **Asymmetries.** When the annotation found something the interview never covers, open the paragraph with `\textbf{Not in the interview.}` Say what the annotation found, what she said about that period, and what her silence leaves open. There is no quote to open with, and the block should say so.

Worked example. For an exchange where the response pitches below her level:

```
"exchangeinterviewtag-%
"textbf{Bears on:} pitching to her level.
```

In an interview in July, after the term ended, she was asked how the tool compared with her professors' office hours, and said it met her level where they did not: professors are "sq-so deep into the subject they don't even understand the level of question I have." Here the comparison runs the other way. She names the standard deviation as the one step she cannot do, and the response explains it as counting heads in fair coin flips, to a student who handles the binomial herself at c3. This is a finding about the tool as configured, not about her learning. The session had no way to know what she already had.

The label line names the topic in plain words. It never uses a theme number, because the numbered themes are in the Findings draft, which this workflow does not read. Every quotation goes through `\sq{}`.

The worked example above opens on her account. That is one of the two orders [section] Who reads the blocks allows, not the required one.

Both the box title and this section say the agent drafts the block. Drafting does not extend to coining vocabulary the book has not defined.

Never write a course number without saying when it happened. This binds every numbered thing in the course: problem sets, exercises, sub-parts, theorems, examples, propositions, lectures, and slide decks. A bare number tells the reader nothing about where in the term the student was, which is what a tag block beside an exchange has to establish.

Write Problem Set 1 (Lecture 1), never Problem Set 1 or PS1. Write Exercise 1.2 (Problem Set 1, Lecture 1) on first mention in a block, then Exercise 1.2 for the rest of it. Write Theorem 1.14 in Lecture 1, not Theorem 1.14. When the block turns on timing, give the LSE week or the date as well.

Course lecture number N maps to LSE week N+1 for N at 6 and above, because of reading week at W06, so a lecture number and a slice's window label come apart later in the term. Never infer one from the other. The course week mapping table is the only source for it.

Check this before the blocks are written: search the draft for every digit-dot-digit and every bare Problem Set, Exercise, Theorem, Proposition, Example and Lecture, and confirm the context appears on first mention of each.

Do not write a bare course number in an `\exchangeinterviewtag{}` block. Name the lecture alongside every problem set, exercise, theorem, proposition, example and lecture number on first mention in the block, and give the LSE week or date whenever the block turns on when it happened.

How to write an entry

Open every finding with their words. Quote them from the transcript or from the Findings draft, and say what they were talking about at the time. A finding that opens "evidence against theme 1" cannot be read without the memo open, and it records a verdict where the evidence is what belongs here.

Name the claim in their terms. "Meeting them where they are" means something on its own. "Theme 1" does not. Never write a theme as a bare number, in a sentence or in a table column. Give the short name and put the number in parentheses after it.

Say what the exchange shows, not what it proves. Describe the prompt and the response, then state how the two relate. The verdict belongs in the write-up.

Name which object the finding is about. Three appear in this corpus: the student, the way the tool is configured, and the teaching material. A finding about Learning mode filed as a finding about their learning misdescribes what the study observed.

Do not grant the tool agency. This study examines how a student attributes capability to a tool. Write "the response explains" and "a response that told them to stop", not "Claude explains".

Judge the tool against its own configuration. This is vanilla Claude with a study template. It has no syllabus, no lecture notes, and no way to know that a supervisor exists.

C.5 Prompt A.4: Findings revision from the theme log

Purpose

Revise the Findings section against the accumulated theme log in a cold session: no slice file and no exchange text open. Claude assists as an editor's assistant, and the first author writes substantive claims.

Stage card

- **Actor.** Claude flags structure and mechanical issues and asks substantive questions; first author writes new claims.
- **Unit of work.** One Findings-revision session called by the first author (not automatic after each slice).
- **Inputs.**
 - Findings section of the article only (theme subsections under that heading).
 - Staleness report: which themes have new log evidence since last edit.
 - At evidence-check time only: the full theme log.
 - Interview transcript on demand, only to check a quotation.
 - Revision instructions.
- **Withheld.**
 - Any annotations-book slice (`slices/*.tex`).
 - Introduction, Background, other methods subsections, Discussion.
 - Raw chatlog text and forensic annotations (evidence enters only via the theme log).
- **Outputs.**
 - Structure drawings per theme (what each paragraph does in the argument).
 - Proposed mechanical edits as before and after file text (not paraphrases).
 - Substantive questions stating what is missing or uncertain (no ghostwritten claim paragraphs).
 - Evidence-check notes: more vivid examples, bundleable cases, contradictions.
 - After approval: revised Findings writing in the article source.
 - State file recording how far the log had been covered when each theme was last edited.
- **Human approval.** First author calls the session, writes substantive text, decides theme-list changes, rules on evidence-check additions, clears edit markers, and compiles the article.

Instruction (production contract)

Instruction · Findings revision: process, norms, and negative rules

The process

Step 1: Orient

Run a script that reads the theme state file and reports, for each theme, how many new log entries arrived since its section was last edited. Post the output to the first author. If the first author does not name a starting point, propose the most stale theme.

Before paragraph edits, confirm the theme list is stable. If the staleness report shows three or more themes with large new-evidence counts, or if the new evidence includes late-term slices that could shift the theme boundaries, ask the first author whether the theme list itself needs revisiting before starting the paragraph walk. Do not start editing text inside a theme whose definition may be about to change.

Step 2: Read cold and draw structure

Read the article draft from `\section{Findings}` through the last theme subsection. First document of the session. Nothing else open. Do not read the sections before Findings or after Findings.

Produce one argument-line drawing per theme section: an ASCII box-and-arrow structure showing what each paragraph does in the argument (claim, corroboration, example, qualification, cross-reference). Post all drawings to the first author in one message. Stop.

The first author reads the drawings against his memory of the themes. If a theme's argument surprises him, he says so before paragraph work begins. He picks a theme to start with, or confirms the file order.

Step 3: Walk paragraphs (per theme)

For each paragraph block (one to two paragraphs forming a logical unit):

Reader model between horizontal rules when the block is structural or the agent is unsure. Five fields: SPEAKER, AUDIENCE, STANDALONE TEST, GROUNDING, PURPOSE. Skip for blocks that are plainly working.

Flags as two blockquotes (before and after), separated by a blank line. Each blockquote is the actual file text or the proposed replacement. The replacement is wrapped in `\pr{}`. Never a paraphrase of the file text.

Mechanical fixes (captioning reflex, dangling referent, double pivot, passive tail, line breaks for git) go as flags. The agent proposes. Typos and logical flow are fixed in the same pass, never separately.

Substantive questions go as questions. The agent states what is missing or uncertain and what sources could fill it. The first author writes or dictates. The agent never drafts a substantive paragraph.

The first author rules on each block: thumbs up, or gives the edit. The agent applies approved edits with `\pr{}`.

The agent re-reads surrounding context each time it advances. If something behind looks wrong in light of the new block, flag it as a backwards glance (same before/after blockquote format).

Step 4: Evidence check (per theme, after paragraph walk)

One delegate per theme. The delegate prompt states precisely what the theme argues, in two or three sentences, not a theme number or title. The delegate reads the theme text and the interview log and returns three things:

(a) **More vivid examples** than what is on the page, with which claim each would strengthen and why it is more vivid than the current one.

(b) **Bundleable cases** for parenthetical clusters, grouped by phenomenon.

(c) **Contradictions**, with which claim each cuts against.

The foreground assesses what belongs in this theme versus other themes, and checks whether cross-referenced themes already cover it. The first author rules on whether anything enters the text.

If the evidence check yields material that needs adding, the agent defines a new `\definecolor` and marking macro (once per session) to distinguish evidence-check additions from paragraph-walk edits.

Step 5: Clear markers

When the first author says to clear markers, the agent strips all `\pr{}` and `\ec{}` wrappers from the edited theme (or the whole file if the first author says so). Update the theme state file with the new `log_coverage_at_edit` for each section that was edited.

Step 6: Compile and commit

The first author compiles the article, reads the PDF, and records the revision. The agent does not do this step.

Presentation norms

These apply at every handoff point.

Self-contained proposals. Every proposal reads without scrolling back. If it references something earlier in the session, it restates the concrete thing. If it uses a term the agent coined, it unpacks it in the same sentence. The "before" blockquote is the actual file text, never a summary.

Mechanical versus substantive. The agent knows the difference and presents them differently. Mechanical: before/after blockquotes, the first author accepts or rejects. Substantive: a question stating what is missing or uncertain, the first author writes the answer.

No agent shorthand. No compressed labels, no coined terms, no abstractions that only unpack inside the agent's context window. If the agent cannot say it in words the first author would use, it has not understood the thing well enough to present it.

Severity gate. If the first author rejects a proposal on conceptual grounds (not "change this word" but "you are thinking about the document wrong"), the agent rolls back to the reader model for that section before proposing anything further. Downstream proposals from a flawed frame are not trusted.

Conventions

- Exchange references are parenthetical citations: (c10). Never "At c10", "from c10 to c16", or "before c10". They are evidence citations, not waypoints in a timeline.
- `\sq{}` for verbatim student speech. Single backtick quotes for the author's characterisation or borrowed terms.
- `\pr{}` for all agent-written or agent-changed text. Cleared by the first author, not by the agent.
- One sentence per line in LaTeX for git-trackable diffs.
- Theme names written in full with number in parentheses: "office-hours substitute (1)".
- Name the lecture with every problem set, exercise, theorem, proposition, and example. Give the LSE week or the date when the claim turns on timing. The week mapping is in the course week mapping table. When the theorem or exercise is only the occasion for a claim about something else (affect, tool behaviour, timing), and a date already places the exchange, the theorem number is not required.

What stays with the first author only

- Writing substantive new text.
- Adding, renaming, merging, or cutting a theme.
- Calling this workflow.
- Deciding when evidence-check material enters the text.
- Compiling and committing.
- Clearing markers.
- Anything that requires having been in the room during the interview.

Negative rules

Do not open any of the annotation slice files at any stage. The log is the source of exchange evidence.

Do not propose a paragraph whose sentences follow the order the exchanges happened in.

Do not draft a substantive paragraph. State what is missing and ask. The first author writes it.

Do not bring a proposal to the first author where the "before" text is a paraphrase of the file text. Use the actual file text.

Do not use a term the agent coined without unpacking it in the same sentence.

Do not propose a second version of the same passage without first naming the fact that, if the agent had it, would change the passage more than the wording will.

Do not use exchange references as grammatical subjects or temporal anchors. They are parenthetical citations only.

C.6 Retained outputs

- **Annotations book slices.** Chronological annotations and Level 1 fields for all 196 exchanges, in fifteen windows.

- **Interview tag blocks.** Tag boxes under exchanges the interview bears on.
- **Theme log.** One entry per window, plus cumulative recurrence tables.
- **Findings section.** Claims and illustrations revised under Prompt A.4.

The full annotations book and theme log remain with the authors as part of the methodological record. Excerpts below are abridged; exchange identifiers match the article. Author notes below identify unsupported interpretations in these illustrative outputs.

Example: annotation (exchange c000)

Instruction · model output · annotation (abridged)

Exposure. The student opens her first Claude session on Thursday of week 1, three days after the Monday release. She has the full Week 1 package but not yet the problem-set solutions, which come Friday.

Student state. She asks for Example 1.11 by number. The lecture presents a lower-bound argument in four compressed sentences with no intermediate derivation. Her prompt names the example but gives no attempt, no quoted passage, and no specific confusion, so the chatbot has to guess what she needs.

Response. The assistant sets up the scenario with two actions accurately and then withholds the derivation: guiding questions and a hint about standard deviation, but no full solution. This is a Learning-mode Socratic pattern. The scaffolding starts from scratch when she may only need the standard-deviation step.

Codes. EXPLAIN; REDIRECT primary; EXPLANATION secondary. Cite c000.

Example: interview tag (exchange c000)

Instruction · model output · interview tag (abridged)

Bears on: Claude's Socratic mode on lecture notes.

She endorsed Claude's Socratic mode for reading lecture notes, saying it worked there because she had the knowledge "deeply buried". The response opens on a lecture example with guiding questions and a hint about standard deviation, which is that pattern applied to the material she named.

Asked what she would change, she wanted to choose the level at which to enter the conversation, and then put the omission on herself. This is a finding about the tool as configured: the study template opens a fresh session with no way to establish that she is a Masters student, and it poses its questions before asking what she already follows.

Author note. The claim that the chatbot had "no way" to establish the student's degree status goes beyond the available evidence: project memory and past-chat search were not independently verified. The exchange shows that Claude began questioning without first asking what she already understood.

Example: theme-log fragment (window c000–c007)

Instruction · model output · theme-log entry (abridged)

First Claude session of the term, twenty-one minutes, three days after Week 1 material was released. They work on Example 1.11 in the lecture notes. The first three exchanges cover the case of two actions and resolve. The last five are one continuous attempt to generalise the argument to n actions, which neither they nor the responses complete.

In the interview they said they preferred Claude to professors because it pitched at their level. At c001 they name the standard deviation as the one step they cannot do. The response explains it as counting heads in fair coin flips, at a register pitched below a Masters student in Operations Research who already handles the binomial.

Author note. The judgment that this explanation was below the student's level is an inference in the model output. Her degree programme alone does not establish what explanation she needed, and at c001 she explicitly requested help with standard deviation.

C.7 Discarded early theme attempt

An earlier attempt supplied the annotations and the interview transcript together and asked for candidate themes in one go. The summaries flattened contradictions or misrepresented the participant's perspective, and those outputs were set aside. A later arrangement that wrote a theme-log entry and then revised the Findings section in the same session was also retired, because a session that has just worked one window of exchanges tends to organise by exchange order, append material where it seems to fit, and leave course numbers underspecified for a co-author who was not in the interview. Prompt A.3 and Prompt A.4 therefore run in separate sessions: tagging and logging with the Findings section withheld, then Findings revision opening cold on the log alone.

Supplement D: Entering mathematical notation

For the article Coexistence of mathematical reasoning and cognitive offloading: A longitudinal case study of one student's chatbot use across a term of proof-based mathematics. Submitted to *Educational Studies in Mathematics*

Jonathan Cardoso-Silva^{1,*} and Johannes Ruf²

¹Department of Methodology, London School of Economics and Political Science, London, United Kingdom

²Department of Mathematics, London School of Economics and Political Science, London, United Kingdom

*Corresponding author

ORCID: Jonathan Cardoso-Silva: 0000-0002-9566-7400; Johannes Ruf: 0000-0003-3616-2194

Correspondence

Jonathan Cardoso-Silva

j.cardoso-silva@lse.ac.uk

Compiled 12 October 2026

D.1 Scope

AI chatbots can interpret mathematical expressions even when they are typed informally or pasted from a PDF. The student frequently copied expressions out of the lecture notes and pasted them into the chat window, where subscripts and superscripts ended up on separate lines and product notation broke apart (Figure D.1).

$\max_{1 \leq i \leq n} w_i^1 \prod_{t=1}^T (1 - \eta)^{\ell_i}$	$\max_{\{1 \leq i \leq n\}} w_i^1 \prod_{t=1}^T (1 - \eta)^{\ell_i}$	$\begin{array}{c} \max \ 1\$ \leq \$i\$ \leq \$n \\ \quad \quad \quad w1 \\ \quad \quad \quad i \\ \$ \prod_{t=1}^T \$ \\ (1\$ - \eta \$) \$ \ell_i^t \$ \end{array}$
(a) Lecture notes	(b) LaTeX notation necessary to produce the expression on the left	(c) Student's paste (c009)

Figure D.1: Mathematical notation copied from a PDF and pasted into a chat window. a An expression from the Lecture 1 notes. b The same expression in $\text{\LaTeX}$, which is what the student would need to type to reproduce it in the chat. c What the chatbot received when the student copied and pasted from the PDF: subscripts and superscripts ended up on separate lines, and the product notation broke apart. The chatbot parsed the intent correctly every time.

Describing herself as a “*lazy typer*” who preferred copying from PDFs to typing out the full notation, she remarked that “*especially with maths, it's ... really annoying to type out the answer. ... I feel like it works better for other subjects*” because those could be answered in plain words. Despite the student's reported frustration with typing notation, the chatbot parsed the intended formula correctly every time, whether from garbled PDF paste or from characters she typed manually. In practice, not having to format the notation before asking a question may have reduced the barrier to using the chatbot as a study tool. There was only one exchange where the student had to re-send a formula, and that was because she had pasted the wrong equation from the lecture (c080, c081).