Workflows and Automation

Philip Trammell[1]

October 3, 2025

I present a model of cross-task learning by doing. Tasks are partitioned into "workflows", and performing a task increases one's productivity at other tasks in the same workflow. This can explain why labor is typically bundled into jobs instead of transacted by the task. Compared to standard task-based models, cross-task learning changes automation's impact on output in three ways. First, making the first few tasks in a workflow automatable increases output less, since it remains efficient for workers to perform some automatable tasks to increase their productivity at the non-automatable tasks. Second, making later tasks in a workflow automatable increases output by more than earlier tasks, as it becomes efficient to automate all automatable tasks, as in a standard model. Third, advanced automation increases output by even more, if machines are intelligent enough also to "learn by doing" and can do more than any single human worker.

[1]Digital Economy Lab, Stanford University. Contact: trammell@stanford.edu. I thank Pamela Mishkin for the conversation that inspired this paper and Chris Tonetti for helpful feedback.

1 Introduction

The task-based framework. In existing task-based models of production, output exhibits constant returns to scale in the performance of a set of necessary tasks (see in particular Zeira (1998) and Aghion et al. (2019)).[1] Each can be performed by labor and/or capital. As long as capital is plentiful enough, labor performs only those tasks which capital cannot. Making more tasks automatable increases output by allowing labor to shift to the remaining non-automatable tasks, which are in shorter supply.

Task-based models have proven especially influential in forecasting the impact on output of advances in AI. Aghion et al. (2019) use their task-based model to argue that AI may simply continue a long-running process in which constant growth is sustained by a constant process of task-by-task automation. Acemoglu (2025) combines a similar model with Eloundou et al.'s (2024) estimates of each O*NET task's LLM exposure to forecast the GDP impact of LLMs. Aghion and Bunel (2024), responding to a draft of Acemoglu's paper, contest practically all of Acemoglu's assumptions except the central one: that output is a function of task quantities, where each task's quantity is proportional to the capital or labor assigned to it.

Three puzzles for the task-based framework. Taking these models literally, it is not clear why labor comes in the task-bundles we call "jobs", as labor could just as efficiently be transacted at the task level. A simple answer in the spirit of Coase's (1937) theory of the firm might be that each labor transaction comes with some fixed cost, and that total transaction costs are therefore minimized when each worker sells all her labor to a single firm. But this would not explain why, as e.g. the O*NET database testifies, firms so often hire multiple individuals to perform the same *suite* of tasks, when the firm is large enough that it could seemingly perform all the tasks in equal quantities by assigning only one task to each employee.[2]

[1]Unlike e.g. Acemoglu and Restrepo (2018), we will focus here on the case in which the set of tasks is fixed.

[2]Korinek and Suh (2024) use an Aghion et al.-style task-based framework to model how output may rise, and wages fall, on the transition to a world of artificial general intelligence (AGI), which they take to be equivalent to full automation. They argue that the model cannot straightforwardly be applied to the O*NET task database, but only because each O*NET task is too *large*. On their view, log(output) is a separable function of "atomistic tasks", each of which is narrower than the O*NET task(s) of which it is a part. This view only exacerbates the "bundling" puzzle.

The CRS task-based framework also fails to capture the possibility of gains from specialization. Relatedly, it fails to capture the fact that in many occupations, compensation increases superlinearly in hours worked.[3] Indeed, these puzzles in some sense exacerbate the first. The simplest way to resolve them in a task-based model would presumably be to introduce within-task learning by doing, so that a worker's productivity at a given task increases in the quantity of the task he performs. But in this case workers are more productive when they perform fewer tasks, so it is most efficient to assign each worker as few tasks as possible.

Cross-task learning. A brief examination of realistic workflows suggests an explanation for all three patterns. Consider the O*NET occupation of Economist. One of its 14 associated tasks (#7538) is to "analyze... data to explain economic phenomena...". Another (#21106) is to "[e]xplain [the] economic impact of policies to the public". Though data analysis and public communication are in some sense very distinct tasks, they cannot easily be assigned to different people: analyzing the data makes an economist much better at credibly discussing its implications. Or consider the Computer Systems Engineer: "[v]erify[ing the] stability, interoperability, portability, security, or scalability of [a] system architecture" (#14669) is much easier when one is also the architect (#14676, #14689).

In short, our productivity at Task B often increases in our performance of Task A. One some margins, performing Task A may even increase our productivity at Task B by more than performing more of Task B. If so, then working longer hours may increase output per hour, yet specialization only yields gains up to a point: beyond it, it is efficient for each worker to perform multiple tasks.

The two examples above illustrate cross-task learning within the context of a particular work project, but cross-task learning may also occur across projects, in a sense

[3] Ameriks et al. (2020) find that older Americans would generally prefer to reduce their working hours gradually if it were possible to do so without reducing their hourly wage, instead of moving discretely from full-time work to retirement, implying that the relative dearth of high-paying part-time work is a feature of the production function and not of worker preferences. Recent work by Jarosch et al. (2025) analogously finds that workers in Germany and the UK (though interestingly not the US) would prefer fewer hours without sacrificing hourly pay. Intuitively, the very long hours often worked in law, consulting, and medical residencies seem attributable in part to the difficulty of partitioning large, time-sensitive projects across multiple employees rather than entirely to employee preferences for long hours: see e.g. Prescott et al.'s (2009) model in which workers choose workweek length given a potentially superlinear function from hours worked to labor services provided.

more reminiscent of conventional learning by doing. For example, one's productivity in teaching may increase in both the number of hours one has spent teaching and the number of hours one has spent doing research, including research on subjects other than the one being taught. The model of this paper is intended to cover both cases.

Outline. I introduce a simple model of cross-task learning by doing that can account for the three phenomena above. I then argue that it sheds light on how we should expect the GDP impact of automation, especially from AI, to unfold.

The space of tasks is partitioned into "workflows". In the simplest setting, introduced in Section 2, performing a task makes one more productive at some or all tasks in the same workflow, but no others. Furthermore, the learning drawn from each task within a workflow exhibits diminishing returns. It is thus efficient for each worker to perform (one or more) complete workflows rather than an arbitrary assortment of tasks. Nevertheless, specialization increases returns by allowing each worker to focus on fewer workflows.

I then consider the implications of the model for the GDP impact of advances in task-based automation. I find that its predictions differ from those of a standard task-based model in three ways.

1. Making the first few tasks in a workflow automatable increases output less than in a standard model. This is because it remains efficient for workers to perform some automatable tasks—i.e. to forego some automation—to increase their productivity at the non-automatable tasks.

2. Making later tasks in a workflow automatable increase output more than earlier tasks, as it becomes efficient to automate all automatable tasks, as in a standard model.

3. Finally, arguably a key distinction between AI-based and earlier waves of automation is that intelligent machines can themselves learn as they work, either continuously or because widespread deployment of one generation of AI and robotics systems facilitates the data collection used to train the next generation. On accounting for this, automation speeds growth by even more than in a standard model in the limit, given that the machines can do more (or learn from more instances of "doing") than any single human worker.

In short, I argue that the relationship between automation and growth should exhibit a convexity not found in standard task-based models.

In Section 3, I extend the model by weakening the distinction between tasks that are and are not in the same workflow. In particular, I allow the performance of tasks in workflow A to improve a worker's productivity in workflow B to an intermediate degree, which decreases continuously with the "distance" between A and B. This extension accommodates the intuition that, even though a worker may perform multiple workflows—so that, as noted above, there is room for further gains from specialization—it is efficient for each job to consist of a cluster of adjacent workflows instead of an arbitrary assortment. The extended model preserves all three qualitative results of the simple model for the GDP impact of automation. It also intensifies the third result, allowing machines to increase their productivity at a workflow not only as a result of performing its tasks at a large scale but also as a result of performing tasks in other workflows.

Section 5 concludes with a discussion of how models without cross-task learning may be leading us astray about the impact of automation in practice, and how a deeper understanding of how tasks and workflows compose into useful work might be valuable.

2 Simple model

Model. The global set of tasks is partitioned into workflows. Each workflow consists of a unit continuum of tasks. We will begin with the case in which there is only one workflow, as this is most similar to the setup of a standard task-based model.

Tasks i range from 0 to 1. For some threshold $I < 1$, tasks $i \leq I$ are *automatable*, in that they can be performed by capital or labor. Tasks $i > I$ can only be performed by labor. K_i and ℓ_i are the (integrable) densities of capital and labor assigned to task i respectively. For simplicity, we will say that output Y (equivalently, the output of the workflow) is Leontief in the quantities of the tasks Y_i, and that there is only worker.

The above is simply the Leontief case of a standard task-based model. To introduce cross-task learning within the workflow, we will say that an individual's productivity at task i, A_i, increases in her performance of tasks $j \leq i$. In particular, we will say that

$$A_i = (1 + \alpha \inf_{j \leq i} \ell_j), \quad \alpha > 0, \tag{1}$$

where $\alpha = 0$ would correspond to the case without learning.

Given capital stock K, an allocation of capital and labor across tasks is efficient if it maximizes

$$Y \equiv \inf_i Y_i \tag{2}$$

subject to

$$Y_i = \begin{cases} K_i + A_i \ell_i, & i \leq I; \\ A_i \ell_i, & i > I; \end{cases} \tag{3}$$

$$\int_0^1 K_i \, di \leq K, \quad \int_0^1 \ell_i \, di \leq \ell, \tag{4}$$

where ℓ is the (exogenous) quantity of labor the worker supplies.

Finally, to simplify even further, we will assume that as long as $I < 1$, lack of capital is not a constraint on the margin. Nevertheless, given learning, it may be efficient for our worker to spend $f > 0$ units of her labor on automatable tasks. The direct value of performing these tasks is zero, as she simply displaces the capital that could otherwise have performed them; the value is entirely the productivity gained on the remaining tasks.

Proposition 1 (The optimal allocation).
Given tasks up to I automatable, it is uniquely optimal (up to measure-zero deviations) to set

$$\ell_i = \begin{cases} \frac{f(I)}{I}, & i \leq I; \\ \frac{\ell - f(I)}{1-I}, & i > I, \end{cases} \tag{5}$$

yielding

$$Y = \left(1 + \alpha \frac{f(I)}{I}\right) \frac{\ell - f(I)}{1 - I}, \tag{6}$$

where

$$f(I) \equiv \begin{cases} \ell I, & I \leq \frac{\alpha\ell}{1+2\alpha}; \\ \frac{\alpha\ell - I}{2\alpha}, & \frac{\alpha\ell}{1+2\alpha} \leq I \leq \alpha\ell; \\ 0, & I \geq \alpha\ell. \end{cases} \tag{7}$$

Proof. Given that the worker spends $f \in [0, \ell]$ of her time on automatable tasks, it is uniquely optimal (up to measure-zero deviations) to set

$$L_i = \begin{cases} \frac{f}{I}, & i \leq I; \\ \frac{\ell - f}{1-I}, & i > I. \end{cases} \tag{8}$$

An unequal allocation of f to automatable tasks features $\ell_i < f/I$ for some $i \leq I$, and thus lower A_i (1) for $i > I$ than under allocation (8). An alternative allocation of $h - f$ to non-automatable tasks must feature a positive-measure task set $\mathcal{S} \subset (I, 1]$ such that $\ell_i < \frac{\ell - f}{1-I}$ for all $i \in \mathcal{S}$ and some $i \leq \inf \mathcal{S}$: so for all $i \in \mathcal{S}$, A_i is weakly lower and ℓ_i strictly lower than in allocation (8).

Likewise, $f > \ell I$ would require a positive-measure task set $\mathcal{S} \subset (I, 1]$ such that $L_i < \frac{\ell - f}{1-I} \left(< \frac{f}{I}\right)$ for all $i \in \mathcal{S}$ and some $i \leq \inf \mathcal{S}$, leaving both A_i and L_i strictly lower than in allocation (8) with $f \leq \ell I$.

Since allocation (8) with $f \leq \ell I$ yields output

$$\left(1 + \alpha \frac{f}{I}\right) \frac{\ell - f}{1 - I},$$

it only remains to find the $f \in [0, \ell I]$ that maximizes the above given I. □

Note that if $\alpha > 1$, $f(I) > 0$ for all I.

How learning changes automation's impact on output. In a standard task-based model, the elasticity of output to increases in automatability—more precisely,

$$-\frac{d \ln(Y)}{d \ln(1 - I)}$$

—equals 1 (in the long run, or immediately given unlimited capital).

Here, from equation (7), we see that when I is low, it is not efficient to use capital at all, because performing the automatable tasks gives the worker much more context on the remainder of the workflow and performing them costs the worker only a small fraction of her time. When $I > \alpha\ell$, on the other hand, it is costly enough to perform the automatable tasks in the workflow manually (in non-negligible quantities) that it is efficient to fully automate them. The elasticity of output to increases in automatability thus equals 0 for low $I < \frac{\alpha\ell}{1+2\alpha}$ and 1 only for $I > \alpha\ell$. Substituting the middle row of (7) into (6) and differentiating reveals that the elasticity rises monotonically from 0 to 1 across the intermediate range.[4]

Specialization and multiple workflows. The model of this section offers a simple explanation for why it may be efficient for each worker to perform a suite of tasks.

It also offers an explanation for why, at least while $I < \alpha\ell$, wages might increase in hours worked. If $I = 0$, for example, a worker working ℓ hours per week (or with ℓ hours of cumulative experience, on the model's "experience"-based interpretation) spreads these hours equally across all tasks within the workflow, enjoying productivity $1 + \alpha\ell$ at each task and thus producing

$$\ell + \alpha\ell^2$$

units of output.

With only one workflow, the model predicts that each worker will perform *all* tasks, as long as $I < \alpha\ell$. This is of course unrealistic, and fails to capture our third desideratum: gains from specialization. Suppose instead therefore that there are W symmetric workflows and N individuals, with $W \gg N$. Let Y_w denote the output of workflow w, and again for simplicity, let output $Y(Y_1, ..., Y_W)$ be symmetric and Leontief in each Y_w. Then, fixing each individual's labor supply at 1, it is uniquely efficient for each individual to perform W/N workflows, dedicating N/W units of labor to each.

[4]One might intuit that the elasticity exceeds 1 for intermediate I, since $f(I)$ falls rapidly through this range. This is incorrect: though the fraction of the workforce assigned to tasks $(I, 1 - I]$ rises rapidly, these gains are largely offset by the rapid decrease in learning.

Absent automation, the output of each task, and aggregate output, then equals

$$\frac{N}{W} + \alpha\left(\frac{N}{W}\right)^2,$$

so aggregate output per person equals

$$\frac{1}{W} + \alpha\frac{N}{W^2}.$$

This increases in N because a larger population allows each worker to specialize in fewer workflows.

Of course, if for all w tasks wi with $i \leq I$ are automatable, the optimal allocation of individuals to workflows does not change. The optimal allocation of each individual's work within her workflows is still characterized by (5) and (7), and output still equals (6), with $\ell = N/W$.

3 A richer task space

Motivation. The model of the previous section offers a simple explanation for our three "puzzles", but it fails in two important ways. First, it predicts that it is efficient for each worker to perform an arbitrary set of workflows, rather than a related set. Second, it predicts that when $I > \alpha$, there is no longer any benefit at all to assigning each worker a related cluster of tasks, as in a model without learning.

Both these limitations are artifacts of the simplifying assumption that tasks are either entirely necessary for learning (if they are downstream in the same workflow) or entirely irrelevant (if they are in different workflows). This section weakens that assumption. Qualitatively, the central point is that:

- Insofar as we learn more by doing a given task (or related tasks) more than by doing more distant tasks, it is efficient to specialize and reap the resulting increasing returns.
- Insofar as there are diminishing returns to learning from a given task in isolation, specialization can be counterproductive. This is true regardless of the state of automation, but it means that feasible automation can be inefficient by requiring us to specialize in inefficient ways.

This point can be illustrated simply in a two-dimensional task space.

Model. There is a unit interval of workflows w, each of which contains a unit interval of tasks wi. As in Section 2, output is Leontief in tasks (or equivalently, in the output Y_w of each workflow, which is Leontief in the output of each of its tasks). There are N workers, indexed by n. ℓ^n_{wi} denotes the density of n's performance of task wi and L^n_w denotes the density of n's performance of tasks in workflow w. Worker n exogenously supplies one unit of labor:

$$\int_0^1 \int_0^1 \ell^n_{wi} di\, dw = \int_0^1 L^n_w\, dw = 1. \tag{9}$$

Now, however, let the productivity of worker n at task wi equal

$$A^n_{wi} = A^n_w\big(1 + \alpha \inf_{j \le i} A^n_w \ell^n_{wj}\big), \quad \alpha > 0; \tag{10}$$

$$A^n_w = g\Big(\int_0^w f(w - v) L^n_v\, dv\Big), \tag{11}$$

where $f(\cdot)$ is positive and decreasing—so that more distant workflows $v \ll w$ contribute less to productivity at w—and $g(\cdot)$ is increasing.

Specialization into intervals of workflows. A^n_w faces no diminishing returns in each L^n_v (except insofar as $g(\cdot)$ may exhibit diminishing returns to learning collectively), let alone the extreme case of Leontief returns maintained within a workflow. As a result, since $f(\cdot)$ is decreasing, efficiency requires partitioning the set of workflows into N equal intervals, each performed exclusively by a single worker. Let $n \in \{1, ..., N\}$ denote the worker assigned workflows

$$w \in \Big[\frac{n-1}{N}, \frac{n}{N}\Big].$$

In the absence of automation, optimal L^n_w decreases in w through this range, offsetting the fact that higher-indexed workflows benefit from more learning via the performance of lower-indexed workflows.

For illustration, suppose

$$f(x) = x^{-\beta}, \quad g(x) = x^{\frac{\beta}{1-2\beta}}, \quad \beta \in \left(0, \frac{1}{2}\right). \tag{12}$$

Then an allocation $\{L^n_w\}$ from $\underline{w}$ to $\overline{w}$ equalizes $A^n_w L^n_w$, and thus Y_w, across the range if

$$L^n_w = m(w - \underline{w})^{-\beta}, \quad m > 0 \tag{13}$$

(letting $A^n_{\underline{w}} L^n_{\underline{w}} \equiv \lim_{w \to \underline{w}^+} A^n_w L^n_w$). The missing coefficient m in (13) follows from the constraint

$$\int_{\underline{w}}^{\overline{w}} L^n_w dw = 1$$
$$\implies m = (1-\beta)(\overline{w} - \underline{w})^{\beta-1}. \tag{14}$$

Substituting (12)–(14) into (11) yields

$$\begin{aligned} A^n_w &= \left(m \int_{\underline{w}}^{w} \big((w-v)(v-\underline{w})\big)^{-\beta} dv\right)^{\frac{\beta}{1-2\beta}} \\ &= \left(m \frac{\big(\Gamma(1-\beta)\big)^2}{\Gamma(2-2\beta)} (w-\underline{w})^{1-2\beta}\right)^{\frac{\beta}{1-2\beta}} \\ &\propto (w-\underline{w})^{\beta}, \end{aligned}$$

which confirms that $A^n_w L^n_w$ is independent of w across the range of workflows n performs.

When the population doubles, each worker performs half as many workflows, so A^n_w increases for each workflow w she still performs. In the example just above, the elasticity of productivity to population is $\frac{\beta(1-\beta)}{1-2\beta}$. To see this, by (12)–(14), halving $\overline{w} - \underline{w}$ (fixing $\underline{w}$) multiplies L^n_w by $2^{1-\beta}$ for each w in the new range. This in turn multiplies A^n_w by $2^{\frac{\beta(1-\beta)}{1-2\beta}}$.

Within a workflow, its effective labor $A^n_w L^n_w$ is allocated equally across tasks.

Automation. Suppose that, across all workflows, tasks wi with $i \leq I$ are automatable. Then it is easy to verify that the results of this section and the previous section are essentially unchanged. It is still optimal to assign each worker an interval of workflows and to allocate $L^n (= 1)$ so that $A^n_w L^n_w$ is equalized across the interval. The allocation of $A^n_w L^n_w$ across workflow w that equalizes $A^n_{wi} \ell^n_{wi}$ across i is given by $\ell^n_{wi} =$ (5) (with L^n_w in place of ℓ) and

$$f(I) = \begin{cases} L^n_w I, & I \leq \frac{\alpha A^n_w L^n_w}{1+2\alpha}; \\ \frac{\alpha L^n_w - I}{2\alpha}, & \frac{\alpha A^n_w L^n_w}{1+2\alpha} \leq I \leq \alpha A^n_w L^n_w; \\ 0, & I \geq \alpha A^n_w L^n_w \end{cases}$$

(i.e. essentially (7) with $A^n_w L^n_w$ in place of ℓ when defining the thresholds). The impact on output of increasing I is precisely as described in Section 2.

If instead tasks wi with $w \leq W$ are automatable, for some $W \in (0, 1)$, these are fully automated, since it is always more efficient to allocate a population across a narrower range of workflows. A proportional decrease to $1 - W$ functions like a proportional increase to population, as described above.

4 Machine learning by doing

We will now briefly explore the third and last implication of (within- and cross-task) learning for automation: that once automation is sufficiently advanced, if machines can learn by doing as humans can, growth will rise by *more* than in a standard task-based model.

We have been assuming for simplicity that output is Leontief across tasks and that, while $I < 1$, output is bottlenecked by labor and not by the tasks capital can perform. In our setting, before full automation, capital's productivity at tasks wi with $i < I$ is irrelevant. In particular, it is irrelevant whether we model capital in the traditional, learning-free way or as learning from the tasks it performs. "Machine learning by doing"[5] thus has interesting implications here only after full automation, so we will consider its implications only in this case. It will hopefully be clear that similar implications can emerge more continuously in a model in which capital partially substitutes

[5] I.e. some process by which the data gathered while doing work—including human evaluations—improves subsequent performance. This may include both continuous learning and simply the use of data gathered on the job in training future AI models.

for labor on the margin while work remains only partially automated.

In a standard task-based model, output after full automation is AK. Once the labor bottleneck is fully relieved, output grows exponentially in the absence of further technological development (given non-negligible saving), and new learning can speed growth only by increasing the replication rate of capital. Here, suppose that capital's productivity at workflow w (A_w) and at task wi (A_{wi}) is given by (10)–(11), with K and k_{wj} in place of L^n and ℓ^n_{wj}. Instead of the unit labor supply constraint faced by each individual in (9), capital faces the constraint

$$\int_0^1 \int_0^1 k_{wi} di \, dw = \int_0^1 K_w \, dw = K.$$

Then output exhibits increasing returns in capital. In particular, using the functional forms above, and recalling that in this setting $[\underline{w}, \overline{w}] = [0, 1]$, we have

$$\begin{aligned} K_w &= (1-\beta) w^{-\beta} \\ \implies A_w &\propto K^{\frac{\beta}{1-2\beta}} w^{\beta}. \end{aligned}$$

The elasticity of productivity to capital is $\frac{\beta}{1-2\beta}$, even greater than the $\frac{\beta(1-\beta)}{1-2\beta}$ elasticity of productivity to population absent automation. This is due to the assumption that, unlike in the human case, every unit of capital can learn from the experience of every other unit. Put another way, when capital performs all workflows, the optimal allocation of a doubled capital stock doubles the density of capital on each workflow. By contrast, a doubled population must divide itself across workflows more finely, so that the mind performing each task benefits from a narrower range of learning.

5 Concluding remarks

Eloundou et al. (2024) evaluate the extent to which GPT-4, perhaps accompanied by necessary "wrapper" software, can automate each task in the O*NET database. At face value, the analysis implies that a lightly augmented GPT-4 could automate 38% of the wage bill.[6] Since GPT-4 does not seem on track to increase output by anything

[6]Following Acemoglu (2025), I am weighting each task by its occupation's share of the 2019–2022 wage bill.

close to 38%, recent attempts to quantify the GDP impact of LLMs (Acemoglu, 2025; Aghion and Bunel, 2024) have produced more intuitively reasonable figures largely by assuming, somewhat ad hoc,

- that tasks judged less than "75% automatable" (Eloundou et al. categories 0–2) generate no labor savings at all and
- that for tasks judged at least "75% automatable" (Eloundou et al. categories 3 and 4), their labor savings in practice will permanently equal their judged automatability multiplied by recent empirical estimates of time savings on a handful of tasks in real-world settings.

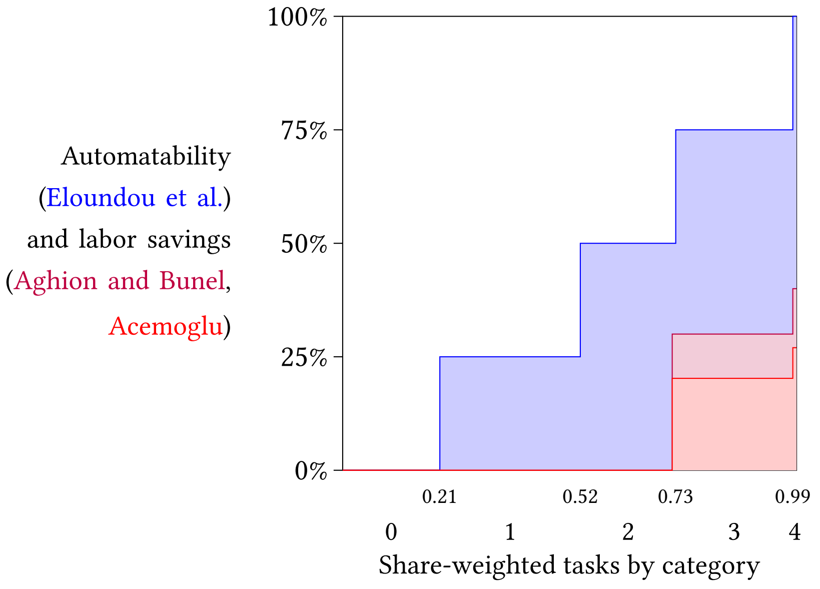

Figure 1: Distribution of "automatability" (Eloundou et al.) and "labor cost savings" (Acemoglu, Aghion and Bunel) across share-weighted tasks

After these and other adjustments, the impact on GDP, even of systems that can truly automate 38% of wage-adjusted tasks in isolation, is estimated to be a few percent at most.

These adjustments perhaps produce reasonable estimates of the impacts of LLMs on GDP today and in the very near future. However, because they permanently multiply task-level estimates of automatability by a factor of 0.27–0.4, in line with recent estimates of labor savings even on *fully automatable tasks*, they have the strange implication that full automation would only yield 27–40% labor savings!

By contrast suppose that, as in the model of this paper, the lack of full labor savings on fully automatable tasks is due to the fact that some labor on task A remains necessary for the worker to understand how to integrate its output with the not-yet-automated task B (and indeed that for tasks with sufficiently low automatability, perhaps it is not worth automating at all, as the authors above assume). Then labor savings per automatable task will rise to 1 as more tasks become automatable. As AI advances, the GDP impact of AI will rise, perhaps quickly, to that suggested by a plain reading of Eloundou et al. or beyond.

References

Acemoglu, Daron, "The Simple Macroeconomics of AI," *Economic Policy*, 2025, *40* (121), 13–58.

— **and Pascual Restrepo**, "The Race Between Man and Machine: Implications of Technology for Growth, Factor Shares, and Employment," *American Economic Review*, 2018, *108* (6), 1488–1542.

Aghion, Philippe and Simon Bunel, "AI and Growth: Where Do We Stand?," 2024. Working paper.

— **, Benjamin F. Jones, and Charles I. Jones**, "Artificial Intelligence and Economic Growth," in Ajay Agrawal, Joshua Gans, and Avi Goldfarb, eds., *The Economics of Artificial Intelligence: An Agenda*, National Bureau of Economic Research, 2019, pp. 237–282.

Ameriks, John, Joseph Briggs, Andrew Caplin, Minjoon Lee, Matthew D. Shapiro, and Christopher Tonetti, "Older Americans Would Work Longer if Jobs Were Flexible," *American Economic Journal: Macroeconomics*, 2020, *12* (1), 174–209.

Coase, Ronald H., "The Nature of the Firm," *Economica*, 1937, *4* (16), 386–405.

Eloundou, Tyna, Sam Manning, Pamela Mishkin, and Daniel Rock, "GPTs are GPTs: Labor market impact potential of LLMs," *Science*, 2024, *384* (6702), 1306–8.

Jarosch, Gregor, Laura Pilossoph, and Anthony Swaminathan, "Should Friday be the New Saturday? Hours Worked and Hours Wanted," 2025. NBER working paper 33577.

Korinek, Anton and Donghyun Suh, "Scenarios for the Transition to AGI," 2024. NBER working paper 32255.

Prescott, Edward C., Richard Rogerson, and Johanna Wallenius, "Lifetime Aggregate Labor Supply with Endogenous Workweek Length," *Review of Economic Dynamics*, 2009, *12* (1), 23–36.

Zeira, Joseph, "Workers, Machines, and Economic Growth," *Quarterly Journal of Economics*, 1998, *113* (4), 1091–1117.