The Macroeconomic Effect of AI: Sizing the Software Engineering Channel*

Alex Blumenfeld **Jonathon Hazell** **Chen Lian** **Andreas Schaab**

March 25, 2026

Abstract

What is the macroeconomic effect of AI? One important channel is that AI will increase the productivity of software engineering, which this paper measures using asset prices. Asset prices capitalize not only current productivity gains due to AI, but also a forecast of future gains expected by the market. We estimate a cross-sectional relationship between the exposure of firms' stock returns to an AI index, and their share of costs in software engineering; and develop a model that maps this relationship into software engineering productivity and GDP. Under our baseline calibration, news about AI from November 2022 to December 2025 implies the following forecast by markets: a cumulative increase in software engineering productivity equivalent to a permanent increase of 28.8 percent per year; and a cumulative GDP increase equivalent to an increase of 0.53 percent per year.

*Blumenfeld: UC Berkeley; alexblumenfeld@berkeley.edu. Hazell: London School of Economics; j.hazell@lse.ac.uk. Lian: UC Berkeley and NBER; chen_lian@berkeley.edu. Schaab: UC Berkeley and NBER; schaab@berkeley.edu. We thank Maarten De Ridder and Eben Lazarus for valuable comments and suggestions; and Sven van Holten and Nicolas Tokay for outstanding research assistance.

1 Introduction

The rapid advance of artificial intelligence has prompted intense debate about its macroeconomic consequences. Forecasts of AI's effect on GDP span an enormous range, from modest gains of roughly one percent over a decade (Acemoglu, 2025) to transformative increases of seven percent or more in global output (Briggs and Kodnani, 2023). This uncertainty reflects a fundamental measurement problem: AI's economic effects operate through many channels—task automation, new product creation, scientific discovery. The aggregate impact of each channel is difficult to observe directly—not the least because many of AI's advances have yet to play out. At the same time, the stakes of this question are high. Whether AI will meaningfully accelerate productivity growth or leave only a modest footprint shapes decisions ranging from monetary policy to corporate investment to workforce training.

Among the channels through which AI may affect the macroeconomy, software engineering (SWE) stands out as both concrete and potentially large. Software engineers produce and maintain the digital infrastructure that underpins operations across virtually every industry, and randomized trials find that developers using AI coding assistants complete individual tasks 20 to 55 percent faster (Peng et al., 2023; Cui et al., 2025). However, these experimental estimates face two limitations as a basis for sizing the macroeconomic impact. First, they measure the effect of today's tools and cannot speak to future AI advances. Second, they capture speed-ups on isolated tasks, not the overall productivity of a software engineer whose job bundles many complementary activities. Because total productivity is disproportionately constrained by the least-improvable bottleneck tasks (Jones and Tonetti, 2026), even large gains on individual tasks can translate into modest gains in a software engineer's overall productivity.

This paper takes a financial-markets approach that addresses both limitations, using forward-looking asset prices to infer the total effect of AI on software engineering productivity and, through it, on GDP. Because stock prices capitalize the entire expected future path of AI-driven productivity gains—not only the improvements delivered by today's tools but also anticipated future advances—they provide a cumulative measure of AI's impact on software engineering productivity and on GDP through the software engineering channel. And because stock prices reflect markets' assessments of total software engineering productivity inclusive of all bottleneck complementarities, they sidestep the difficult aggregation from task-level speed-ups to overall software engineering productivity gains.

The central idea is that if AI primarily raises the productivity of software engineers, firms that rely more heavily on software engineers should benefit disproportionately from AI advances. In particular, an AI-induced increase in software engineering productivity lowers the effective cost of software engineering labor. Firms with higher software engineering cost shares experience a larger reduction

in their unit costs and capture outsized profit gains. This differential profit exposure is capitalized into stock prices, generating a cross-sectional pattern: firms with higher software engineering cost shares should exhibit higher stock return sensitivity to an AI index. The magnitude of this cross-sectional relationship—how steeply stock return sensitivity to the AI index rises with software engineering cost share—is informative about the size of the underlying AI-induced cumulative software engineering productivity growth per unit increase in the AI index.

Specifically, our approach to sizing the GDP impact of AI through the software engineering channel combines an empirical step with a model-based step. The empirical step estimates how firms' exposure to AI advances rises with their software engineering cost share; the structural step maps this cross-sectional gradient into a cumulative increase in software engineering productivity and into GDP growth.

The empirical step proceeds in two stages. In the first stage, we estimate each firm's stock return sensitivity to an AI index, controlling for standard risk factors. In the second stage, we regress these sensitivities on firms' software engineering cost shares, controlling for industry fixed effects and firm-level controls, to obtain the cross-sectional slope δ. A key advantage of this cross-sectional approach is that it isolates the software engineering productivity channel—even though movements in the AI index can reflect also changes in overall productivity or discount-rates. Because these channels affect all firms uniformly, they do not generate cross-sectional profit variation correlated with software engineering cost shares. Only SWE-augmenting productivity shocks create a differential cost advantage for SWE-intensive firms, generating profit variation proportional to firms' software engineering cost shares. The cross-sectional regression thus projects firms' AI-index exposures onto their software engineering cost shares, retaining only the component driven by AI-induced software engineering productivity growth.

Although the cross-sectional slope isolates the software engineering productivity channel, translating it into an implied productivity gain requires a structural model. The slope reflects both the magnitude of the underlying AI-induced software engineering productivity growth per unit movement in the AI index and the structural parameters that govern how strongly a given software engineering productivity increase translates into cross-sectional profit differences. To disentangle these, we develop a model that inverts the chain from software engineering productivity to cost reductions to differential profits to the observed slope δ. Two key structural parameters govern this inversion: the elasticity of substitution across product varieties ε, which determines how aggressively consumers shift expenditure toward firms that lower their prices, and the substitution elasticity σ between SWE and non-SWE labor, which controls how much firms shift their input mix toward cheaper software engineers. Together, these parameters determine how much a given software engineering productivity increase translates into differential profit gains across firms with different software engineering cost

shares. Inverting this mapping recovers the AI-induced software engineering productivity growth per unit movement in the AI index. Multiplying by the total return of the AI index, over a given period, yields the cumulative AI-induced software engineering productivity growth implied by the AI news in that period. Finally, the aggregate economy-wide software engineering cost share $\bar{\gamma}$ converts the cumulative software engineering productivity growth into the corresponding cumulative GDP impact through the software engineering channel.

We study data from the start of 2022 to the end of 2025. Our data combine stock returns from CRSP, firm-level accounting variables from Compustat, and employee-level occupation data from Revelio Labs, which provide granular information on each firm's software engineering workforce. We proxy for the AI index using the ROBO Global Artificial Intelligence ETF (THNQ), and control for standard risk factors using the Fama-French three-factor model. The key firm-level regressor is the software engineering wage-bill share: the fraction of a firm's total payroll attributable to software engineers, constructed from Revelio's occupation and compensation data.

We aim to measure firms who might benefit from software engineering as an input into production—as opposed to firms that sell output related to software engineering (e.g. firms selling software-as-a-service). Therefore following Acemoglu et al. (2022), we drop all firms in software producing industries (NAICS sectors 51 and 54); we also drop firms who are part of the semiconductor supply chain. In our sample there are firms across various industries that use software engineering intensively, including: online marketplaces (e.g. eBay), high end electronics manufacturers (e.g. Sonos), and travel booking platforms (e.g. Expedia or Airbnb).

In our baseline specification with NAICS3 industry fixed effects and size controls, the second-stage regression yields an estimated slope of $\tilde{\delta} \approx 1.8$ on the software engineering wage-bill share: a one-percentage-point increase in a firm's software engineering payroll share is associated with a 1.8 percentage-point increase in its loading on the AI index. Because the model is formulated in terms of the software engineering cost share γ_j, we convert to the cost-share basis by dividing by the aggregate labor cost share ($\alpha_L = 0.312$ from KLEMS), giving a cost-share slope of $\delta \approx 5.8$. Year-by-year regressions show that the positive relationship between software engineering intensity and AI-index exposure is present in every year from 2022 through 2025. The result is robust and stable across a range of additional specifications, which account for many possible confounders; and we also consider specifications with fine industry fixed effects (NAICS4 and NAICS6 level). This stability suggests limited scope for unobserved heterogeneity, other than the software engineering share, to explain the cross sectional relationship (Oster, 2019).

To convert δ into a productivity and GDP estimate, we invert the structural mapping from the model. Under our baseline calibration ($\varepsilon = 5$, $\sigma = 1$), news about AI from November 2022 to December 2025 implies a market-implied cumulative increase in software engineering productivity equivalent

to a permanent increase of 28.8 percent per year, within the range of the 20–55 percent task-level speed-ups found in the experimental literature (Peng et al., 2023; Cui et al., 2025). Two offsetting forces can explain this alignment. On the one hand, complementarities between tasks imply that total productivity gains should be smaller than task-level gains (Jones and Tonetti, 2026): when a software engineer's workflow combines many complementary activities, output is constrained by the least-improvable bottleneck tasks, so even large gains on individual tasks translate into modest gains in overall productivity. On the other hand, our estimate reflects the annualized cumulative effect of the entire future path of software engineering productivity growth, so it capitalizes anticipated future AI advances and should therefore exceed the gains delivered by today's tools alone. Through the software channel, the market-implied cumulative increase in GDP is equivalent to a permanent increase of 0.53 percent per year. This number is bounded above due to the small aggregate software engineering cost share $\bar{\gamma} = 0.0185$. These estimates are moderately sensitive to the within-firm substitution elasticity σ but stable across demand elasticities $\varepsilon \geq 2$.

We embed this empirical strategy in a richer environment with intermediate inputs, an input-output network, and aggregate shocks that can shift the relative demand composition across goods. This extension serves two purposes. First, it formalizes the identification assumptions behind our approach to inferring the macroeconomic impact of AI through software engineering productivity. The key concern is that an aggregate shock correlated with the AI index might systematically shift consumer demand toward SWE-intensive goods, generating profit differentials proportional to firms' software engineering cost shares that mimic a true productivity shock. Our identification therefore requires a single condition—that aggregate shocks do not systematically shift demand toward SWE-intensive goods conditional on industry fixed effects and other firm-level controls. Under this condition, our two-stage regression isolates the software engineering productivity channel from confounding macro shocks, and the reduced-form slope δ can be cleanly inverted into AI-induced software engineering productivity change and, via Hulten's theorem, into GDP.

Second, the extended model allows a precise quantitative evaluation that accounts for cost-savings propagation through intermediate-input linkages. The input-output network lowers the structural gradient δ^S because upstream software engineering cost advantages partially diffuse to downstream sectors, attenuating the cross-sectional spread. But the quantitative effect is modest: for the news about AI from November 2022 to December 2025, the annualized cumulative GDP impact through the software engineering channel changes only marginally, confirming that the dominant channel is the within-firm cost reduction from cheaper effective software engineering labor rather than the propagation of cost savings through the production network.

The remainder of the paper is organized as follows. Section 2 presents the two-stage empirical strategy and develops a simple model that maps the cross-sectional slope into GDP. Section 3 de-

scribes the data. Section 4 reports the empirical results and a back-of-the-envelope GDP calculation. Section 5 extends the framework to a quantitative environment with input-output linkages, formalizes identification, and provides the full quantitative analysis.

Literature review. A growing literature constructs occupation-level AI exposure measures that classify tasks by their susceptibility to automation (Eloundou et al., 2024; Felten, Raj and Seamans, 2021; Webb, 2020; Acemoglu et al., 2022; Cazzaniga et al., 2024). Building on these measures, existing estimates of AI's aggregate impact on GDP span a wide range. Acemoglu (2025) bounds the TFP effect at 0.53–0.66% over a decade; Aghion and Bunel (2024) estimate 0.68% per year using a task-based framework with broader assumptions (see also Briggs and Kodnani, 2023; Nordhaus, 2021; Trammell and Korinek, 2025; Filippucci, Gal and Schief, 2024). Yet these top-down calculations rely on task-exposure measures that say little about how quickly firms adopt AI, how task-level gains aggregate to firm-level productivity, or how AI capabilities will evolve in the future.

Randomized experiments find large task-level speed-ups from AI coding assistants (Peng et al., 2023; Paradis et al., 2024; Cui et al., 2025; Gambacorta et al., 2024), with similar gains documented in customer support (Brynjolfsson, Li and Raymond, 2024), professional writing (Noy and Zhang, 2023), and management consulting (Dell'Acqua et al., 2023). However, translating task-level gains into aggregate output is difficult. Acemoglu et al. (2022) find that AI-exposed establishments reduce non-AI hiring, yet detect no aggregate employment or wage effects at the occupation or industry level, illustrating the gap between micro-level adoption and macro-level consequences. Trammell (2026) shows that within workflows, automating early tasks can have little effect when later tasks or handoffs remain bottlenecks, and Becker et al. (2025) find that frontier AI tools slowed experienced open-source developers in mature codebases. Jones and Tonetti (2026) provide a theoretical benchmark for why such aggregation losses can be severe: in their model the economy's production structure features complementary tasks—"weak links"—so that even fully automating all current software tasks yields a small GDP effect. Consistent with these aggregation challenges, Bick, Blandin and Deming (2025) show that generative AI diffused rapidly by late 2024 but still delivered only modest average time savings in the aggregate, despite high adoption rates. Our paper takes a different route: because asset prices capitalize not only current AI-driven productivity gains but also anticipated future advances, and already embed expectations about adoption speed and task-to-job aggregation, we can sidestep these difficulties. Specifically, we estimate the macroeconomic importance of one concrete channel—software engineering (SWE)—from cross-sectional stock returns. Our baseline implies an annualized GDP impact of approximately 0.53% through the software engineering channel.

A growing literature uses stock market data to measure the economic value of AI at the firm level. Eisfeldt, Schubert and Zhang (2023) show that firms with high workforce exposure to generative AI

gained roughly 5% in market value relative to low-exposure firms after ChatGPT's release. Hampole et al. (2025) find that task-level AI substitution is strong but within-occupation task reallocation mutes firm-level employment effects. Babina et al. (2024) show that AI-investing firms grow faster and innovate more. These papers use broad, task-based exposure measures; our paper instead isolates a single productivity channel—software engineering—and uses a structural production-network model to map the cross-sectional stock return slope into an aggregate GDP effect. Andrews and Farboodi (2025) use bond yield responses to AI model releases to assess whether markets price in transformative growth, and Bertomeu et al. (2025) exploit Italy's temporary ChatGPT ban for causal identification of AI's firm-level value (see also Kogan et al., 2017; Eisfeldt and Schubert, 2024; Qian, Peng and Li, 2025; Ante and Saggu, 2025). We differ from this work, by focusing on a single channel—software engineering.

Finally, our work connects to the theoretical literature on AI, automation, and growth. The task-based framework of Acemoglu and Restrepo (2019) provides the foundation for analyzing how automation displaces workers from existing tasks while new tasks reinstate labor demand, with Acemoglu and Restrepo (2022) showing that task displacement accounts for 50–70% of US wage structure changes over four decades. Growth-theoretic models show that AI's long-run impact depends critically on whether it can automate the research process itself (Aghion, Jones and Jones, 2019; Korinek and Suh, 2024). Recent work emphasizes that AI-driven changes in task composition interact with workers' comparative advantage to produce heterogeneous labor market effects even within highly exposed occupations (Freund and Mann, 2025; Althoff and Reichardt, 2026), and that AI may augment workers' capabilities rather than automate their tasks (Agrawal, McHale and Oettl, 2026). We focus on how AI affects the productivity of a single, crucial, input—namely software; and do not focus on how AI affects the assignment of workers to tasks more broadly.

2 Sizing the Macroeconomic Impact of AI via Software Engineering

This section presents our approach to sizing the macroeconomic impact of AI through the software engineering channel. We first describe the empirical strategy that estimates the cross-sectional relationship between firms' stock return exposure to an AI index and their software engineer intensity. We then develop a model that provides the theoretical foundation for this empirical approach, showing how to map the estimated cross-sectional slope into an implied change in cumulative software engineer productivity and, ultimately, a cumulative GDP impact. The model in this section is deliberately simple for illustrative purposes; Section 5 extends the framework to a richer environment with input-output linkages and demand composition shifts, which allows us to formally characterize the conditions under which our approach identifies the macroeconomic impact of AI and to provide a

full quantitative analysis.

2.1 Empirical Strategy

Our empirical strategy proceeds in two stages. In the first stage, following the spirit of Fama and MacBeth (1973), we estimate each firm's stock return sensitivity to an AI index, controlling for standard risk factors, to obtain a firm-level AI beta. In the second stage, we regress these estimated betas on firms' software engineer payroll shares, controlling for industry fixed effects and other firm-level controls, to estimate the cross-sectional slope $\tilde{\delta}$.

First stage: firm-level AI betas. We start by estimating each firm's sensitivity, or "beta", with respect to an AI-focused exchange traded fund. For each firm j in the regression sample, which we will describe in section 3, we run the following time-series regression using weekly data:

$$R_{jt} - R_t^f = \alpha_j + \beta_j \left(R_t^{\text{AI}} - R_t^f\right) + \sum_i \psi_{ji} F_{it} + \epsilon_{jt}, \tag{1}$$

where $R_{jt} = d\log V_{jt}$ is firm j's stock return, R_t^{AI} is the return on the AI index, R_t^f is the risk-free rate, and F_{it} are control factors. In the baseline specification, the controls are the Fama-French three factors (Fama and French, 1993): the excess market return, SMB (small minus big), and HML (high minus low book-to-market). The coefficient β_j measures firm j's AI-factor loading: a one percentage point increase in the AI index excess return is associated with a β_j percentage point movement in the firm's excess return. The control factors absorb each stock's exposure to broad market forces, such as the market risk premium, size, and value, so that β_j isolates the firm's sensitivity to AI-specific news.

Second stage: explaining cross-sectional betas. The cross-sectional model relates estimated firm betas to software-engineering intensity:

$$\hat{\beta}_j = \tilde{\delta} \cdot \tilde{\gamma}_j + \Gamma \cdot X_j + u_j, \tag{2}$$

where $\hat{\beta}_j$ is the first-stage estimate of firm j's AI beta, $\tilde{\gamma}_j$ is the software engineering payroll share (our empirical proxy for the model's cost share γ_j), and X_j denotes the full vector of second-stage controls. In the baseline specification, X_j includes size controls together with a full set of industry indicators. The coefficient $\tilde{\delta}$ measures by how many percentage points the AI beta increases when a firm's software engineering payroll share rises by one percentage point, conditional on this common control set X_j. Throughout, the focus is on firms that use software as an input into production. Therefore in what follows we will exclude firms who either sell software, or form part of the semiconductor supply chain.

We use the software engineering payroll share $\tilde{\gamma}_j$ rather than the software engineering cost share γ_j because the firm-level payroll share is directly constructable from Revelio's individual-level com-

pensation data, whereas the cost share requires total firm-level costs that are not readily available for the same set of firms. The two can be related by a proportionality factor: $\tilde{\gamma}_j = \gamma_j / \alpha_L$, where α_L is the aggregate labor share of total costs, so the estimated coefficient satisfies $\tilde{\delta} = \delta \alpha_L$. From 2020 KLEMS data, the aggregate labor cost share is $\alpha_L = 0.312$ (with capital and intermediate inputs accounting for 25.2% and 43.6%, respectively), giving $\delta = \tilde{\delta} / \alpha_L \approx 3.2\tilde{\delta}$. We report $\tilde{\delta}$ in the regression tables and convert to the model's cost-share slope δ for all quantitative exercises below.

The cross-sectional slope δ is the key empirical input to the structural model developed next. To translate δ into an aggregate GDP impact, we need a model that provides a structural interpretation of why firms with higher software engineer cost shares have higher AI betas, and that maps the magnitude of this relationship into an implied change in software engineer productivity.

2.2 From Cross-Sectional Estimates to GDP Impact via a Model

We now develop a simple general equilibrium model to map the cross-sectional slope δ into implied aggregate productivity and GDP effects. To ease exposition and directly deliver closed-form expressions for our quantities of interest, we use a tractable parametric environment without intermediate inputs and with symmetric constant elasticity of substitution demand weights across goods—meaning aggregate shocks do not shift the composition of demand for different goods. Section 5 introduces intermediate inputs and input-output linkages, relaxes demand symmetry, develops the role of demand composition in order to discuss omitted variable bias, and provides a full quantification.

2.2.1 Environment

Time is discrete, indexed by $t \in \{0, 1, \ldots\}$. The economy comprises J firms, indexed by j, each producing a distinct good. A representative household owns all firms and financial markets are complete. There are three aggregate shocks: S_t, A_t, and m_t. S_t is a scalar that denotes the productivity of software engineers (SWE). A_t is a Hicks-neutral aggregate productivity shock. m_t is an aggregate discount-rate shock. The aggregate shocks may be arbitrarily correlated.

Households. The representative household is endowed with two factors, supplied inelastically: software engineers in quantity s and non-SWE labor in quantity L. Household consumption preferences are

$$\mathbb{E}_0\Big[\sum_{t=0}^{\infty} \beta^t u(c_t; m_t)\Big],$$

where period utility u may depend on m_t, capturing discount-rate shocks and shifts in the marginal utility of wealth. The consumption bundle is a standard CES aggregate with symmetric weights,

$$c_t = \Big[\sum_j c_{jt}^{\frac{\varepsilon-1}{\varepsilon}} \Big]^{\frac{\varepsilon}{\varepsilon-1}},$$

so that aggregate shocks do not alter the relative demand for different goods. Cost minimization implies

$$c_{jt} = \Big(\frac{p_{jt}}{P_t}\Big)^{-\varepsilon} c_t \qquad \text{and} \qquad P_t = \Big[\sum_j p_{jt}^{1-\varepsilon} \Big]^{\frac{1}{1-\varepsilon}},$$

where P_t is the consumer price index and p_{jt} denotes the price of good j. We use nominal expenditure as the numeraire, setting $P_t c_t = 1$ for all t. The household's stochastic discount factor (SDF) is

$$M_{t,t+1} = \beta \frac{u_c(c_{t+1}; m_{t+1})}{u_c(c_t; m_t)}.$$

Production. Each firm j produces gross output using a constant returns to scale CES production technology,

$$y_{jt} = A_t \Big[\phi_j^{1/\sigma} (S_t s_{jt})^{(\sigma-1)/\sigma} + (1-\phi_j)^{1/\sigma} L_{jt}^{(\sigma-1)/\sigma} \Big]^{\sigma/(\sigma-1)}, \tag{3}$$

where s_{jt} and L_{jt} are firm j's employment of SWE and non-SWE labor, and $\sigma > 0$ is the elasticity of substitution. The parameter $\phi_j \in (0,1)$ governs software engineering intensity and varies across firms. The scalar A_t acts as a Hicks-neutral shifter common to all firms, while S_t specifically augments SWE labor. The total cost of firm j is $\mathscr{C}_{jt} = w_t s_{jt} + W_t L_{jt}$, and its unit cost is

$$mc_{jt} = \frac{1}{A_t} \Big[\phi_j (w_t/S_t)^{1-\sigma} + (1-\phi_j) W_t^{1-\sigma} \Big]^{1/(1-\sigma)},$$

where w_t/S_t is the effective software engineering wage. Firms face CES demand and charge a constant markup $p_{jt} = \frac{\varepsilon}{\varepsilon-1} mc_{jt}$. Per-period profits are a constant fraction of nominal revenue, $\pi_{jt} = \frac{1}{\varepsilon} p_{jt} y_{jt}$.

We define firm j's software engineering cost share, $\gamma_{jt} = (w_t s_{jt})/\mathscr{C}_{jt}$. In this CES structure, $\gamma_{jt} = \phi_j (w_t/(S_t \cdot mc_{jt} A_t))^{1-\sigma}$, which is strictly increasing in ϕ_j. The value of firm j is the present discounted value of its profit stream,

$$V_{jt} = \mathbb{E}_t \Big[\sum_{\tau=0}^{\infty} M_{t,t+\tau} \pi_{j,t+\tau} \Big]. \tag{4}$$

Market clearing. Goods market clearing implies $y_{jt} = c_{jt}$ and labor market clearing requires $\sum_j s_{jt} = s$ and $\sum_j L_{jt} = L$. We define firm j's Domar weight or sales share as $\lambda_{jt} = p_{jt} y_{jt}$. The aggregate Domar-weighted software engineering cost share is $\bar{\gamma}_t = \sum_j \lambda_{jt} \gamma_{jt}$.

2.2.2 First-Order Responses to Aggregate Shocks

We characterize the first-order response of the economy to a perturbation $(d\log S_t, d\log A_t, d\log m_t)$. We drop t subscripts when referring to steady-state values.

Under CES demand with constant markups, a firm's profit response is driven by changes in its unit cost relative to the aggregate price index: $d\log\pi_{jt} = (1-\varepsilon)(d\log mc_{jt} - d\log P_t)$. Applying Shephard's lemma, a firm's marginal cost moves according to its factor cost shares: $d\log mc_{jt} = -d\log A_t + \gamma_j(d\log w_t - d\log S_t) + (1-\gamma_j)d\log W_t$. Because the aggregate price index P_t simply aggregates these individual costs using Domar weights λ_j, it follows that $d\log P_t = -d\log A_t + \bar{\gamma}(d\log w_t - d\log S_t) + (1-\bar{\gamma})d\log W_t$, where $\bar{\gamma} = \sum_j \lambda_j\gamma_j$ is the Domar-weighted aggregate software engineering cost share. Taking the difference and substituting into the profit equation yields the relative profit response:

$$d\log\pi_{jt} = (\varepsilon-1)(\gamma_j - \bar{\gamma})\big(d\log S_t - (d\log w_t - d\log W_t)\big). \tag{5}$$

Equation (5) isolates the product-market force driving our results. The term $d\log S_t - (d\log w_t - d\log W_t)$ captures the net reduction in the effective cost of software engineering labor relative to other labor, which can come from either a direct increase in software engineering productivity $d\log S_t$ or a decline in the relative software engineering wage $d\log w_t - d\log W_t$. When this effective cost falls, firms with a higher-than-average reliance on software engineers ($\gamma_j > \bar{\gamma}$) experience a relative marginal cost decline. Since demand is elastic ($\varepsilon > 1$), consumers substitute toward these newly cheaper goods, amplifying the cost advantage into outsized profit gains.

Crucially, the Hicks-neutral productivity shock A_t drops out of relative unit costs entirely. Because A_t scales all production frontiers uniformly without distorting relative marginal products, both factor prices must adjust equally in general equilibrium ($d\log w_t = d\log W_t$) to clear labor markets. This leaves relative costs—and therefore relative profits—unchanged. Similarly, the discount-rate shock m_t shifts the stochastic discount factor uniformly across firms but does not affect cross-sectional relative profits.

In contrast, S_t is a SWE-augmenting shock. An increase in S_t effectively multiplies the aggregate supply of software engineers, which creates a differential cost advantage for software-engineering-intensive firms. With inelastic factor supplies, the relative effective software engineering wage $\log w_t - \log S_t$ must fall in general equilibrium to clear the labor market. The size of this adjustment is

$$d\log w_t - d\log S_t - d\log W_t = -\frac{1}{\Sigma}d\log S_t, \qquad \text{with} \quad \Sigma = \sigma + (\varepsilon-\sigma)\frac{\sigma_s^2}{\bar{\gamma}(1-\bar{\gamma})},$$

where $\sigma_s^2 = \sum_j \lambda_j(\gamma_j - \bar{\gamma})^2$ is the cross-sectional variance of software engineering cost shares and Σ is the aggregate elasticity of substitution, combining within-firm input substitution and between-firm consumer reallocation. If firms can easily substitute toward software engineers (high σ), or if

consumers aggressively shift expenditure toward SWE-intensive goods (high ε acting on large cross-sectional dispersion σ_s^2), the effective software engineering wage falls less in general equilibrium, which dampens the cross-sectional profit spread.

Substituting this general equilibrium factor-price adjustment back into (5) yields

$$d\log\pi_{jt} = \frac{\varepsilon-1}{\Sigma}(\gamma_j-\bar{\gamma})d\log S_t = \delta^S(\gamma_j-\bar{\gamma})d\log S_t,$$

where $\delta^S \equiv \frac{\varepsilon-1}{\Sigma}$ is the structural cross-sectional gradient of the profit elasticity with respect to the software engineering cost share. Because financial markets are forward-looking, firm values capitalize not just the contemporaneous profit response but the entire discounted stream of future changes. To state the result compactly, define the normalized cumulative software engineering productivity change,

$$\mathscr{S}_t \equiv (1-\beta)\mathbb{E}_t\Big[\sum_{k=0}^{\infty}\beta^k d\log S_{t+k}\Big],$$

If the shock is permanent ($\mathbb{E}_t[d\log S_{t+k}] = d\log S_t$ for all $k \geq 0$), then $\mathscr{S}_t = d\log S_t$: the normalized cumulative change coincides with the per-period productivity change. We summarize the firm-value response in the following lemma.

Lemma 1 (Firm Value Response)**.** *To first order, the firm value response is*

$$d\log V_{jt} = \delta^S\gamma_j\mathscr{S}_t + \mu_t, \tag{6}$$

where μ_t is firm-invariant, collecting responses common to all firms.

2.2.3 Inversion and GDP Impact

We now derive the model counterpart of the empirical cross-sectional slope δ. Let F_t denote a vector of control factors (e.g., the Fama-French market, SMB, and HML factors), and define the residualized excess AI index return $\tilde{R}_t^{\text{AI}}$ as the population residual from projecting $R_t^{\text{AI}} - R_t^f$ onto F_t. In population, the two-stage procedure recovers

$$\beta_j = \frac{\text{Cov}_t(d\log V_{jt}, \tilde{R}_t^{\text{AI}})}{\text{Var}_t(\tilde{R}_t^{\text{AI}})}, \qquad \delta = \frac{\text{Cov}_j(\beta_j, \gamma_j)}{\text{Var}_j(\gamma_j)}.$$

Lemma 1 implies $\beta_j = \beta_0 + \kappa\delta^S\gamma_j$, where β_0 is firm-invariant and κ is the population projection coefficient of $\mathscr{S}_t$ onto the residualized AI index return:

$$\kappa \equiv \frac{\text{Cov}_t(\mathscr{S}_t, \tilde{R}_t^{\text{AI}})}{\text{Var}_t(\tilde{R}_t^{\text{AI}})}.$$

It follows that $\delta = \kappa\delta^S$. Because Hicks-neutral and discount-rate shocks do not generate cross-sectional variation correlated with γ_j, the slope δ isolates the software engineering productivity channel. Movements in the AI index may also capitalize other aggregate shocks, but these common effects are ab-

sorbed into the firm-invariant intercept β_0 and are not reflected in δ. Moreover, because asset prices are forward-looking, δ captures the normalized cumulative software engineering productivity change $\mathscr{S}_t$, including current and future gains as priced by financial markets, including anticipated future advances in AI. However, the expression $\delta = \kappa\delta^S$ shows that the observed slope δ conflates these productivity gains (captured by κ) with the structural gradient δ^S, which governs how strongly a given productivity shock translates into cross-sectional profit differences. Recovering the implied software engineering productivity gains and, ultimately, the GDP impact of AI through the software engineering channel therefore requires inverting δ through the structural gradient.

Proposition 1 (Financial-Market-Implied Software Engineering Productivity Change)**.** *To first order, the best linear predictor of the normalized cumulative software engineering productivity change given a movement in the AI index is*

$$\mathbb{E}[\mathscr{S}_t \mid \tilde{R}_t^{AI}] = \kappa \cdot \tilde{R}_t^{AI}, \qquad \textit{where} \quad \kappa = \frac{\delta}{\delta^S} = \frac{\Sigma}{\varepsilon - 1} \cdot \delta. \tag{7}$$

Propositions 1 and 2 report the normalized cumulative software engineering productivity change without restricting the time-series dynamics of $d\log S_{t+k}$. As an illustrative special case, suppose the shock is permanent: $\mathbb{E}_t[d\log S_{t+k}] = d\log S_t$ for all $k \geq 0$. Then $\mathscr{S}_t = d\log S_t$, i.e. the normalized cumulative change coincides with the per-period productivity change, and κ directly gives the market-implied per-period productivity change per unit of AI index return:

$$\mathbb{E}[d\log S_t \mid \tilde{R}_t^{\mathrm{AI}}] = \kappa \cdot \tilde{R}_t^{\mathrm{AI}} = \frac{\Sigma}{\varepsilon - 1} \cdot \delta \cdot \tilde{R}_t^{\mathrm{AI}}. \tag{8}$$

Finally, we translate the implied software engineering productivity gains into aggregate output effects. Hulten's theorem applies period by period: $d\log Y_t = d\log A_t + \bar{\gamma}\, d\log S_t$, where the discount-rate shock m_t does not affect real output under inelastic labor supply. Because our cross-sectional strategy isolates only the S_t component, define the normalized cumulative GDP change through the software engineering channel,

$$\mathscr{Y}_t \equiv (1-\beta)\mathbb{E}_t\Big[\sum_{k=0}^{\infty} \beta^k d\log Y_{t+k}\big|_S\Big],$$

in parallel to $\mathscr{S}_t$.

Proposition 2 (GDP Impact of AI through the Software Engineering Channel)**.** *Since $d\log Y_{t+k}|_S = \bar{\gamma}\, d\log S_{t+k}$ period by period,*

$$\mathscr{Y}_t = \bar{\gamma}\mathscr{S}_t, \tag{9}$$

where $\bar{\gamma} = \sum_j \lambda_j \gamma_j$ is the Domar-weighted aggregate software engineering cost share. The best linear predictor of the normalized cumulative GDP change through the software engineering channel given a

movement in the AI index is

$$\mathbb{E}[\mathscr{Y}_t \mid \tilde{R}_t^{AI}] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{AI} = \frac{\bar{\gamma}\Sigma}{\varepsilon - 1} \cdot \delta \cdot \tilde{R}_t^{AI}. \tag{10}$$

Equation (10) provides a transparent sufficient-statistic formula for the GDP impact of AI through the software engineering productivity channel. The formula decomposes the multiplier into three components:

$$\bar{\gamma}\kappa \quad = \quad \underbrace{\delta}_{\substack{\text{observed} \\ \text{cross-sectional slope}}} \quad \times \quad \underbrace{\frac{\Sigma}{\varepsilon - 1}}_{\substack{\text{inversion:} \\ \text{index return} \to \mathscr{S}}} \quad \times \quad \underbrace{\bar{\gamma}}_{\substack{\text{Hulten weight:} \\ \mathscr{S} \to \mathscr{Y}}}$$

The first factor is the observed cross-sectional slope from the data. The second converts asset-price exposures into normalized cumulative software engineering productivity change: a more elastic product market (ε) amplifies relative profit differences, so a smaller underlying shock is needed to rationalize a given δ; conversely, higher aggregate substitutability (Σ) dampens wage effects, so a larger shock is required. The third factor translates software engineering productivity into GDP via the aggregate cost share $\bar{\gamma}$.

As an illustrative special case, if the shock is permanent then $\mathscr{S}_t = d\log S_t$ and $\mathscr{Y}_t = d\log Y_t|_S$, i.e. the normalized cumulative changes coincide with their per-period counterparts:

$$d\log Y_t\big|_S = \bar{\gamma}\, d\log S_t = \mathscr{Y}_t, \tag{11}$$

and the best linear predictor is

$$\mathbb{E}[\mathscr{Y}_t \mid \tilde{R}_t^{\mathrm{AI}}] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\mathrm{AI}} = \frac{\bar{\gamma}\Sigma}{\varepsilon - 1} \cdot \delta \cdot \tilde{R}_t^{\mathrm{AI}}. \tag{12}$$

The multiplier κ converts a unit movement in the residualized AI index into the implied normalized cumulative software engineering productivity change. Multiplying κ by the total residualized AI-index return $\tilde{R}_t^{\mathrm{AI}}$ over a given period therefore yields the cumulative AI-induced software engineering productivity change implied by the AI news in that period. Similarly, multiplying the GDP multiplier $\bar{\gamma}\kappa$ by $\tilde{R}_t^{\mathrm{AI}}$ yields the corresponding normalized cumulative GDP change through the software engineering channel.

The simple model presented above delivers a transparent sufficient-statistic formula, but it leaves open whether the cross-sectional slope δ cleanly identifies the software engineering productivity channel in a richer environment. The main concern is that an aggregate shock correlated with the AI index might systematically shift consumer demand toward SWE-intensive goods, generating profit differentials proportional to firms' software engineering cost shares that mimic a true productivity shock. Section 5 embeds our empirical strategy in a general equilibrium environment with intermediate inputs, an input-output network, and aggregate shocks that can shift the relative demand composition across goods. There we show that identification requires a single condition: that aggregate

shocks do not systematically shift demand toward SWE-intensive goods conditional on industry fixed effects and other firm-level controls. Under this condition, our two-stage regression isolates the software engineering productivity channel from confounding macro shocks, and the reduced-form slope δ can be cleanly inverted into AI-induced software engineering productivity change and, via Hulten's theorem, into GDP.

3 Data

Our core data sources are resume history data from Revelio Labs (Revelio Labs, 2026), financial statement variables from Compustat (S&P Global Market Intelligence, 2025), historical data on the Fama-French factors from Kenneth French's website, and stock and exchange traded fund (ETF) price data from CRSP (Center for Research in Security Prices, 2025).

Revelio Labs aggregates publicly available professional profiles from platforms such as LinkedIn, to construct a comprehensive, individual-level dataset covering workers' employment histories, job titles, employers, and compensation (Revelio Labs, 2026). Each record includes a standardized occupation classification based on O*NET codes (e.g., 15-1252.00, "Software Developers"), which Revelio then aggregates into approximately ten broad role categories such as "Software Engineer," "Data Scientist," "Sales," and "Operations." Each record also includes an imputed measure of total compensation, which combines base salary and variable pay such as bonuses but excludes stock options. Compensation is not observed directly but is instead predicted by Revelio using a model that takes as inputs features of the job such as role, seniority, and firm. The dataset also provides individual-level sampling weights that adjust for the fact that workers in some occupations are more likely to have public professional profiles than others. We use these data to construct the key firm-level regressor in our analysis: the share of total payroll attributable to software engineers.

Compustat North America, maintained by S&P Global Market Intelligence, provides standardized financial statement variables drawn from firms' annual and quarterly public filings with the SEC (S&P Global Market Intelligence, 2025). We use the Compustat annual fundamentals file to obtain firm-level measures of total assets, net sales, operating profitability, investment, leverage, and book-to-market ratios. These variables will serve as controls in various regressions to come. We also use the Compustat-CRSP link table to merge accounting data with stock return data at the firm level.

The Fama-French factor returns, available from Kenneth French's data library, provide daily and weekly time series of the standard risk factors used in asset pricing (Fama and French, 1993, 2015). In our baseline results, we will make use of the three factors of Fama and French (1993): the excess market return (MKT), the size factor (SMB), and the value factor (HML). In robustness exercises, we will also make use of the momentum factor (UMD) and the profitability factor (RMW) from the five-

factor model of Fama and French (2015). The excess market return (MKT) is the return on a broad value-weighted portfolio of US equities minus the risk-free rate, proxied by the one-month Treasury bill rate. The size factor (SMB, "Small Minus Big") is constructed by sorting firms into groups based on market capitalization and taking the difference in returns between portfolios of small-cap and large-cap stocks. The value factor (HML, "High Minus Low") sorts firms by their ratio of book equity to market equity and takes the difference in returns between high book-to-market (value) firms and low book-to-market (growth) firms. The momentum factor (UMD, "Up Minus Down") sorts firms on their cumulative returns over the prior two to twelve months and takes the difference between past winners and past losers. The profitability factor (RMW, "Robust Minus Weak") sorts firms on operating profitability and takes the difference in returns between highly profitable and less profitable firms. These factors are designed to capture exposure to broad market forces and to well-documented cross-sectional return premia.

The Center for Research in Security Prices (CRSP) US Stock and US Index databases provide historical security-level data on prices, returns, shares outstanding, and market capitalization for all securities listed on major US exchanges (Center for Research in Security Prices, 2025). We use weekly holding-period returns from CRSP for the firm-level time-series regressions in our first stage. CRSP also provides returns on exchange-traded funds, which we use to obtain the return series for the AI-related ETFs that serve as our aggregate AI index. We link CRSP security data to Compustat firm identifiers using the CRSP-Compustat merged database.

In the main results, we use THNQ, the ROBO Global Artificial Intelligence ETF, as our measure of the stock market returns of AI-intensive firms. THNQ tracks the ROBO Global Artificial Intelligence Index, which selects constituents based on their revenue exposure to artificial intelligence and classifies them into two broad categories: firms providing AI infrastructure (such as cloud computing, semiconductors, and data centers) and firms developing or deploying AI applications (such as healthcare AI, autonomous systems, and business analytics). The index uses a modified equal-weight methodology, in which holdings are assigned to scoring tiers based on the depth of their AI revenue exposure and then approximately equally weighted within each tier, giving higher aggregate weight to firms with more direct AI exposure. The index typically contains around 65 to 75 holdings and is rebalanced quarterly. Appendix Table B.1 lists the twenty largest holdings and their weights in the index.

We also run robustness checks that replace THNQ with AIQ, the Global X Artificial Intelligence and Technology ETF, to ensure that our results are not driven by ROBO Global's particular choices regarding which firms to label as representative of innovations in AI. AIQ tracks the Indxx Artificial Intelligence and Big Data Index, which selects firms that derive a meaningful share of their revenue from AI, big data, or related technology segments. Unlike THNQ, AIQ uses a capped market-capitalization weighting scheme, in which individual holdings are subject to concentration caps of approximately 3

to 4.5 percent at each rebalance. This gives the index a heavier tilt toward large-cap technology and semiconductor firms relative to THNQ. The index typically contains around 85 to 90 holdings and is reconstituted semi-annually. Appendix Table B.2 lists the twenty largest holdings and their index weights.

Our main measure of the importance of software engineers at a firm is the share of total compensation in Revelio that goes to all workers in the "Software Engineer" category, specifically during 2021, the year before our stock-price panel begins.[1] We compute the total number of workers in each occupation at a firm using Revelio's individual-level weights (addressing measurement error from some occupations being more likely to have public profiles). Since compensation is imputed, we will also report results using raw employment shares of software engineers rather than payroll shares.

3.1 Sample Selection

Table 1 shows how we arrive at our main analysis sample. The time period is from the start of 2022 to the end of 2025. We start with the initial CRSP-Compustat merge (row 1). We filter to Compustat firms with strictly positive assets and sales (row 2). In row 3, we make industry restrictions. We drop NAICS sectors 52 (Finance and Insurance) and 53 (Real Estate and Rental and Leasing).

The focus of the paper is on firms that use software as an input to production, not firms that sell software products. Therefore we drop the latter group. As in Acemoglu et al. (2022), in Row 3 we also exclude firms in NAICS sectors 51 (Information) and 54 (Professional, Scientific, and Technical Services), as many firms in these sectors are producers of AI rather than users. In Row 4, we drop firms classified by Claude as being involved in the development or production of semiconductors. These firms' stock returns will likely covary with returns on the AI index because they are integral suppliers of hardware required to run AI models, as opposed to being exposed to AI innovations through the software channel.[2] In row 5 we retain only firms that match to Revelio. In row 6 we compute average assets and average sales from 2017 to 2021, ignoring missing years, and restrict to firms with strictly positive average assets and sales over this period. In row 7, we use a balanced panel of firms with stock price data for all weeks of 2022 through 2025. Finally, in row 8 we drop the bottom 5% of firms ranked by total employment Revelio (sample selection is a concern for these firms, because they may have few resumes posted online). At this point, we have the final dataset.

The Table also lists the percentiles of 2021 sales. Even as we make the sample restrictions, the distribution of firm sales remains reasonably similar. Appendix Figure C.1 shows the distribution of firms across NAICS2 sectors after the NAICS exclusions and in our cleaned final sample; the distributions

[1] Revelio's "Software Engineer" role category aggregates 125 O*NET occupation codes. Appendix Table B.3 lists the twenty largest by compensation share.

[2] Appendix Table B.4 summarizes the firms that are dropped by this restriction.

are similar in each.

We present some simple statistics about our main regressor, the firm level payroll share of software engineers. Table 2 shows the top 20 firms with the highest software engineering intensity according to our measure. There seem mostly to be two business models in common among the firms with a high share of software engineers, namely: hi-tech manufacturing (e.g. Sonos, Arlo) and e-commerce platforms (e.g. eBay, Expedia). Figure C.2 shows the full distribution of software engineer payroll shares; for reference, in our main sample, there are 196 firms with zero software engineers observed in our dataset in 2021.

4 Results

This section presents our main empirical results concerning the cross sectional relationship between a firm's software engineering share, and the loading of its stock return on the aggregate AI index. We find that a 1 pp increase in the software engineering share of payroll correlates with around a 2 pp increase in the stock's sensitivity to the AI index. These estimates imply a roughly 28.8% increase in software engineer productivity, which in turn leads to a GDP increase of 0.53%.

4.1 Baseline Estimates

We start with our baseline estimates of $\tilde{\delta}$, estimated from equations (1) and (2). That is, in a first stage, we estimate the "beta" of each stock with respect to the AI index (after controlling for other aggregate risk factors). In a second stage, we regress these betas on the firm's software engineering share, as well as size and industry controls. Let us add some other details about the regression. In the baseline, the regression is unweighted. Standard errors heteroskedasticity robust. Our firm size controls are indicators for quintiles of average assets and sales between 2017 and 2021. The industry controls are either at the NAICS 2- or 3-digit level.

As we have discussed, we make an identification assumption to size the software channel from the cross section of asset prices. At the firm level, and conditional on our controls, the software engineering cost share does not correlate with other determinants of how stock prices comove with the AI index. Later on, we will present various tests designed to probe this assumption, as well as a range of other robustness tests.

Figure C.3 shows the distribution of our first-stage betas for the specification with NAICS3 fixed effects, and Table B.5 shows summary statistics. This measure of firm-level loadings on our chosen AI index is centered near zero, and 18.8% of firms have AI betas that are significant at the 5% level.

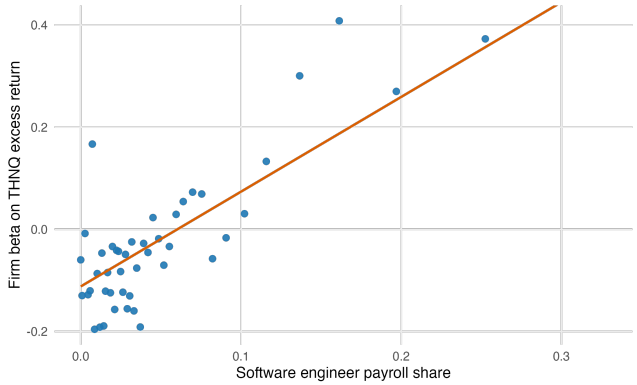
Table 3 reports estimates of the baseline regression. In the first column, we present estimates of

the regression of the AI index beta on the software engineering share. The coefficient $\tilde{\delta} = 1.8$ means that a 1 pp increase in a firm's software engineering payroll share leads to a 1.8 pp increase in their AI beta. That is, after a 1 pp increase in the AI index, the firm's return is 1.8 pp higher. The result is stable across specifications. As we add size controls in column 2, 2 digit industry fixed effects in column 3, or 3 digit industry fixed effects in column 4, the estimates remain at around 1.8, and continue to be statistically significant and precisely estimated. In the final column, we add a control for firms' exposure to task displacement due to ChatGPT, developed by Eisfeldt, Schubert and Zhang (2023). Again, the results do not change.

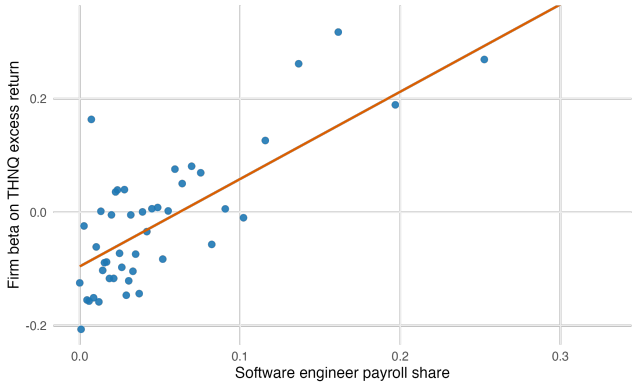
We visualize the result with binned scatter plots. Specifically, Figure 1 plots 50 bins of the software engineering share, on the x axis; against the firm's sensitivity to the AI index, on the y axis. The top-left panel has no controls (i.e. column 1 of Table 3). The top-right panel controls for NAICS3 and size fixed effects (i.e. our baseline specification, in column 4 of the table). The bottom panel adds controls for task displacement due to ChatGPT (i.e. column 5 of the table). In each case, there is a clear positive relationship between software engineering share and sensitivity to the AI index.

The estimated coefficients remain similar as we consider an extensive range of robustness exercises, as Table 4 documents. The first row is the baseline specification. The second row drops microcap firms, defined as firms with market capitalization below 1 $billion at the end of 2021. The third row uses instead of our baseline AI index, THNQ, the alternative index (AIQ). As we discussed, AIQ has broader coverage, meaning it is more likely to include all AI stocks, but may also contain other technology stocks too. The fourth and fifth rows add in additional controls in the first stage regression, which estimates the sensitivity of each firm to the AI index. These controls absorb other time series confounders that might correlate with AI shocks and affect firm values. In particular we control for the excess return of the ETF QQQ (which tracks the NASDAQ 100) in row 4, which absorbs the exposure of each shock to general technology shocks. In row 5 we control for 2 more Fama-French factors, momentum and profitability. Following Fama and French (2015), these factors absorb additional time series shocks that affect firms heterogeneously. Row 6 weights the regression by size (end of 2021 market capitalization) and row 7 controls for market capitalization. Row 8 weights by the software engineering payroll share, while row 9 uses instead raw employment shares of software engineers. Rows 10-13 control for additional characteristics that affect firms' sensitivity to aggregate shocks (book-to-market, operating profitability, investment and leverage). Row 14 controls for region, via the location of the firm's headquarters. Row 15 restricts the sample to US-incorporated firms only. Rows 16-17 control for the most granular possible industry fixed effects, at the 4 and 6 digit level. Finally, row 18 addresses potential concerns about serial correlation in the first-stage time series and sampling uncertainty in the cross section by reporting bootstrap standard errors: calendar weeks are resampled using a moving-block bootstrap with block length 6 weeks, firms are resampled with re-

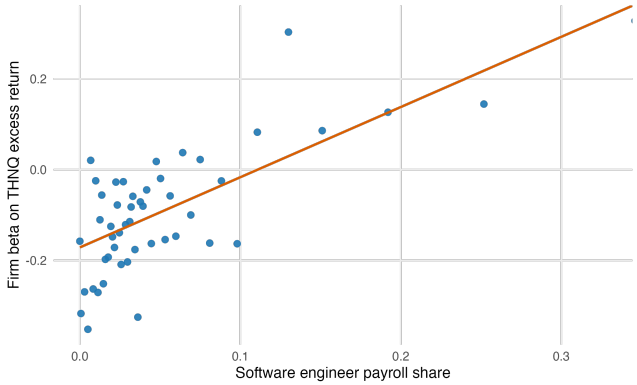
Figure 1: Software engineer payroll share vs. THNQ beta



(a) No controls



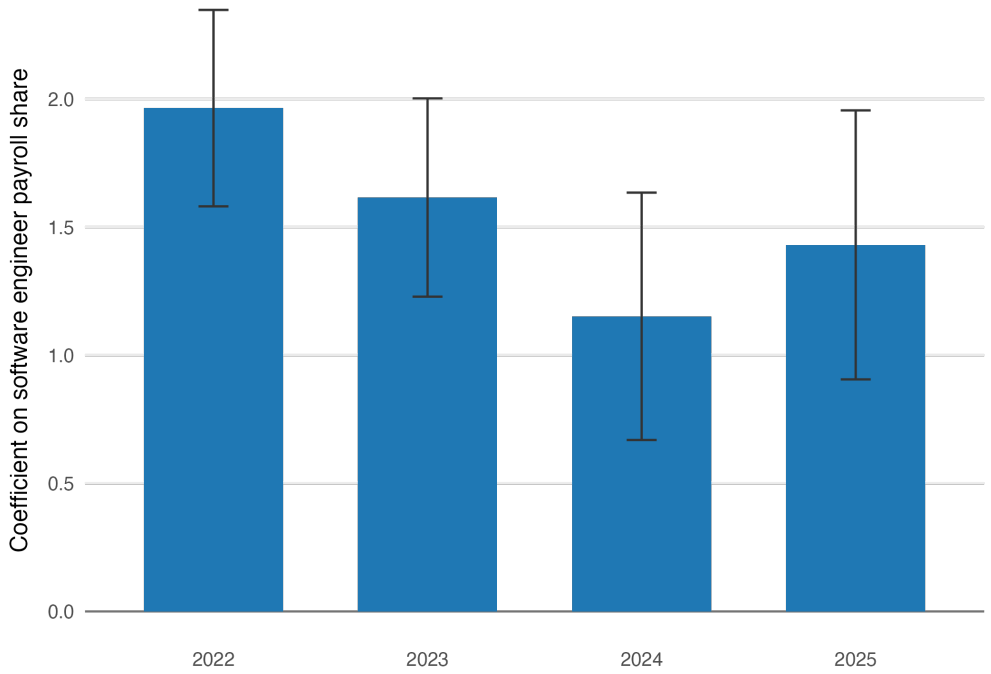
(b) NAICS3 and size fixed effects



(c) GenAI task exposure controls

Notes: Each panel plots 50 equal-sized bins of the software engineer payroll share (x-axis) against the firm's estimated beta on THNQ excess returns (y-axis). Panel (a) includes no controls. Panel (b) controls for NAICS3 industry and firm size quintile fixed effects. Panel (c) additionally controls for quintiles of the generative AI task exposure measure of Eisfeldt, Schubert and Zhang (2023). The fitted line is the OLS slope from the corresponding column of Table 3.

Figure 2: Year-by-year second-stage coefficient estimates



Notes: Each point reports the estimated coefficient on the software engineer payroll share from the baseline NAICS3 fixed-effects second-stage specification, where the first-stage AI betas are estimated using data from a single calendar year only. Whiskers show 95% confidence intervals based on HC3 standard errors.

placement, and both stages are re-estimated in each iteration. The bootstrap standard error (0.42) is larger than the baseline HC3 standard error (0.23), but the coefficient remains statistically significant.

It is reassuring that the coefficients are stable as we consider various permutations. As Oster (2019) points out, stability of the coefficients as we add observed heterogeneity to the controls is informative. It suggests that unobserved heterogeneity is unlikely to bias the estimates.

Our baseline regression studies a continuous measure of software engineering intensity. We now study a discrete measure, by discretizing into five quintiles of software engineering. Table 5 reports the results. As the table shows, the AI sensitivity is higher in the top quintile and to some extent the fourth quintile. That is, the top tail of software engineering intensity drives the results. In Appendix Table B.6, we show this result is not driven by outliers (as is apparent from Figure 1). We also study the stability of our estimates, by estimating $\tilde{\delta}$ separately for every year. Figure 2 plots the coefficients. The estimates are similar in every year.

4.2 Quantifying the GDP Impact of AI through the Software Engineering Channel

The goal of this subsection is to translate the cross-sectional slope δ from the regression of firm-level AI-index betas β_j on software engineering cost shares γ_j into the aggregate GDP impact of AI through

the software engineering channel. Proposition 2 provides the bridge: the best linear predictor of the normalized cumulative GDP change through the software engineering channel given a movement in the residualized AI index is

$$\mathbb{E}[\mathscr{Y}_t \mid \tilde{R}_t^{\mathrm{AI}}] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\mathrm{AI}},$$

where $\bar{\gamma}$ is the Domar-weighted aggregate software engineering cost share and κ is the multiplier from Proposition 1. All that remains is to express $\bar{\gamma}\kappa$ in terms of estimable quantities and to plug in empirical values. By equation (10),

$$\bar{\gamma}\kappa = \frac{\bar{\gamma}\Sigma}{\varepsilon - 1} \cdot \delta.$$

Assuming further Cobb-Douglas within-firm substitution ($\sigma = 1$), the aggregate substitution elasticity becomes $\Sigma = 1 + (\varepsilon - 1)\sigma_s^2/[\bar{\gamma}(1-\bar{\gamma})]$, and the expression reduces to

$$\bar{\gamma}\kappa = \left[\frac{\bar{\gamma}}{\varepsilon - 1} + \frac{\sigma_s^2}{1-\bar{\gamma}}\right] \cdot \delta,$$

where the bracketed expression decomposes into the direct Hulten effect and the between-firm reallocation correction. A key advantage of working with the normalized cumulative change is that the formula depends only on the demand elasticity ε, the cross-sectional dispersion σ_s^2, and the aggregate software engineering cost share $\bar{\gamma}$, but not on the discount factor β. Table 6 collects all parameter values and sources. We now discuss the calibration of each input.

The demand elasticity ε. The demand elasticity governs the product-market force: how strongly a relative cost advantage translates into relative profit gains. Standard values in the production-network literature range from $\varepsilon = 1$ (Cobb-Douglas, as in Long and Plosser, 1983 and Acemoglu et al., 2012) to $\varepsilon \approx 4$–6 (trade-elasticity estimates from Broda and Weinstein, 2006). The lower bound $\varepsilon = 1$ is ruled out by our identification strategy: at $\varepsilon = 1$, the AI shock generates no cross-sectional spread in profits ($\delta^S = 0$), and the method has no identifying variation. Our baseline is $\varepsilon = 5$, following Broda and Weinstein (2006), which corresponds to a markup of $\mu = \varepsilon/(\varepsilon - 1) = 1.25$. We explore $\varepsilon \in \{2, 3, 5\}$ in the sensitivity analysis.

The within-firm substitution elasticity σ. The parameter σ governs substitution between software engineering and non-SWE labor within the firm. In the small-$\bar{\gamma}$ limit, the GDP multiplier is approximately proportional to σ, making this the most influential structural parameter.

The canonical estimate from Katz and Murphy (1992) for college-versus-non-college substitution is $\sigma \approx 1.4$, reinforced by Ciccone and Peri (2005) at $\sigma \approx 1.5$ and Krusell et al. (2000) at $\sigma \approx 1.67$ in a capital-skill complementarity framework. However, Havránek et al. (2024) argue that after correcting for publication bias and CES-specific attenuation, the literature is consistent with $\sigma \leq 1$. Three caveats apply to any direct mapping from these estimates. First, SWE versus non-SWE is a finer partition than

college versus non-college: the within-skilled substitutability between software engineers and other college workers may differ from the standard across-skill-group estimate. Second, sector-level and aggregate elasticities can diverge substantially through within-plant versus between-plant substitution channels, as shown by Oberfield and Raval (2021). Third, short-run estimates at business-cycle frequencies may understate long-run substitutability.

Given this uncertainty, we set $\sigma = 1$ (Cobb-Douglas within the labor bundle) as the baseline and explore $\sigma \in \{0.5, 1.0, 1.5, 2.0\}$.

Data-based objects. The remaining inputs to the GDP formula are directly computable from publicly available data. The data directly yield the software engineering output share γ_j^y, defined as the ratio of software engineering compensation in sector j to sectoral gross output. We construct γ_j^y using 2021 data from two sources. From the BLS Occupational Employment and Wage Statistics (OEWS), we obtain industry-by-occupation employment and wage cells for the detailed software-production occupation family (SOC codes 15-1251 through 15-1255), restricting to private-sector establishments. Because OEWS reports wages rather than total compensation, we convert to a compensation proxy using industry-specific multipliers from the BLS Employer Costs for Employee Compensation (ECEC). From the BEA/BLS KLEMS production accounts, we obtain sectoral gross output, value added, and labor compensation for 61 private-sector industries (excluding federal and state-and-local government). The Domar weights are sectoral gross output divided by private-sector GDP, where private GDP is the sum of sectoral value added over all private KLEMS industries. The Domar-weighted mean output share is $\bar{\gamma}^y = \sum_j \lambda_j \gamma_j^y = 0.0148$, and the Domar-weighted cross-sectional variance of output shares is $\sigma_s^{2,y} = \sum_j \lambda_j (\gamma_j^y - \bar{\gamma}^y)^2 = 0.000962$.

Because firms charge a constant markup $p_{jt} y_{jt} = \frac{\varepsilon}{\varepsilon-1} \mathscr{C}_{jt}$, the model's cost share $\gamma_j = w_t s_{jt} / \mathscr{C}_{jt}$ is related to the output share by $\gamma_j = \frac{\varepsilon}{\varepsilon-1} \gamma_j^y$. The aggregate cost-share objects therefore depend on ε: $\bar{\gamma} = \frac{\varepsilon}{\varepsilon-1} \bar{\gamma}^y$ and $\sigma_s^2 = (\frac{\varepsilon}{\varepsilon-1})^2 \sigma_s^{2,y}$. At the baseline $\varepsilon = 5$, this gives $\bar{\gamma} = 0.0185$ and $\sigma_s^2 = 0.0015$. In practice, software engineering cost shares are small and concentrated: most sectors have γ_j close to zero, while a handful of technology-intensive sectors have much higher values. This implies $\sigma_s^2 \ll \bar{\gamma}(1-\bar{\gamma})$, so that the between-firm reallocation channel is quantitatively modest.

Our second-stage regression yields $\tilde{\delta} = 1.8$ (column 4 of Table 1, the specification with NAICS3 fixed effects and size controls). Converting from the payroll-share to the cost-share basis as described in Section 2.1, the cost-share slope is $\delta = \tilde{\delta} / \alpha_L = 1.8/0.312 \approx 5.8$. From November 2022 (the launch of ChatGPT 3.5) to December 2025, the cumulative residualized AI-index return (after Fama-French projection) is $\tilde{R}_t^{\text{AI}} = 14.9\%$.

Back-of-the-envelope calculation. At the baseline ($\varepsilon = 5$, $\sigma = 1$), the cost-share objects are $\bar{\gamma} = \frac{5}{4} \times 0.0148 = 0.0185$ and $\sigma_s^2 = (\frac{5}{4})^2 \times 0.000962 = 0.0015$. The aggregate substitution elasticity is $\Sigma \approx 1.33$.

Proposition 1 gives the multiplier as

$$\kappa = \frac{\Sigma}{\varepsilon - 1} \cdot \delta = \frac{1.33}{4} \times 5.8 \approx 1.93.$$

Next, take the cumulative residualized AI-index return of $\tilde{R}_t^{\text{AI}} = 14.9\%$. Multiplying by κ gives the market-implied normalized cumulative software engineering productivity change:

$$\mathbb{E}[\mathscr{S}_t \mid \tilde{R}_t^{\text{AI}}] = \kappa \cdot \tilde{R}_t^{\text{AI}} \approx 1.93 \times 0.149 \approx 28.8\%.$$

The corresponding normalized cumulative GDP change through the software engineering channel is

$$\mathbb{E}[\mathscr{Y}_t \mid \tilde{R}_t^{\text{AI}}] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}} \approx 0.0185 \times 28.8\% \approx 0.53\%.$$

If the software engineering productivity shock is permanent ($\mathbb{E}_t[d\log S_{t+k}] = d\log S_t$ for all $k \geq 0$), then $\mathscr{S}_t = d\log S_t$ and $\mathscr{Y}_t = d\log Y_t|_S$, so the normalized cumulative changes coincide with the per-period changes: a 28.8% annual software engineering productivity increase and a 0.53% annual GDP contribution through the software engineering channel.

To benchmark these estimates against direct evidence, note that the experimental literature finds AI coding assistants raise task-level productivity by 20–55%: Peng et al. (2023) find a 56% reduction in task completion time, Paradis et al. (2024) report 21–26%, Cui et al. (2025) estimate a 26% throughput increase, and Gambacorta et al. (2024) find a 55% increase in output.[3] Two offsetting forces place our 28.8% estimate in the same range as these task-level figures. First, our estimate is forward-looking: $\mathscr{S}_t$ reflects not only today's AI-driven productivity gains but also anticipated future advances in AI models, workflows, and adoption, which pushes our estimate *above* the gains delivered by current tools alone. Second, weak-link complementarities push in the opposite direction: because a developer's workflow bundles many complementary tasks, total productivity is constrained by the least-improvable steps, so task-level speed-ups overstate overall gains (Jones and Tonetti, 2026). That these two forces roughly offset, leaving our estimate in the same ballpark as the experimental evidence, provides a useful cross-check on the magnitudes.

Robustness. Table 7 reports the structural gradient δ^S, the multiplier $\kappa = \delta/\delta^S$, and the GDP multiplier $\bar{\gamma}\kappa$ across a grid of (σ, ε). The results illustrate two key features. First, the GDP impact is bounded by the aggregate software engineering cost share $\bar{\gamma}$: the first-order Hulten effect limits the GDP contribution of the software engineering channel regardless of how large the software engineering productivity gain might be. Second, κ is moderately sensitive to σ but relatively stable across demand elasticities $\varepsilon \geq 2$, reflecting the approximate cancellation of $(\varepsilon - 1)$ between the numerator of δ^S and the product-market force in the profit elasticity.

[3] The only study reporting a negative effect is Becker et al. (2025), who find that experienced open-source developers were 19% *slower* with AI on tasks in mature codebases they knew deeply.

5 Identification and Quantitative Analysis

This section extends the stylized framework of Section 2.2 to a rich, quantitative general equilibrium environment. We depart from the simple model in two key dimensions. First, we introduce intermediate inputs and an input-output (IO) network, which allows shocks to propagate through supply chains. Second, we generalize the other aggregate shocks into a vector Z_t that not only acts as a Hicks-neutral productivity shifter but also as a demand shifter $\omega_j(Z_t)$, explicitly allowing for aggregate shocks that alter the relative demand composition across goods.

This extended framework serves a dual purpose. First, it formalizes the identification assumptions behind our empirical strategy. We demonstrate that, as long as the other aggregate shocks do not systematically shift demand toward SWE-intensive goods, our cross-sectional approach perfectly isolates the software engineering productivity channel from confounding macro shocks, even in a rich network economy. Second, by accounting for input-output linkages and value-added shares, it allows for a precise quantitative evaluation of the macroeconomic GDP impact of AI.

5.1 Environment

Time is discrete, indexed by $t \in \{0, 1, \ldots\}$. The economy comprises J firms, indexed by j and k, each producing a distinct good. A representative household owns all firms and financial markets are complete.

The economy is driven by three types of shocks. S_t is a scalar that denotes the productivity of software engineers (SWE). $Z_t = (Z_{1t}, \ldots, Z_{Nt})$ is a vector that collects all other aggregate shocks, generalizing the Hicks-neutral and discount-rate shocks of the simple model while additionally allowing for arbitrary cross-good demand shifts. Finally, $\xi_t = (\xi_{1t}, \ldots, \xi_{Jt})$ is a vector of firm-specific idiosyncratic shocks. The scalar S_t and the vector Z_t may be arbitrarily correlated.

Households. The representative household is endowed with two factors, supplied inelastically: software engineers in quantity s and non-SWE labor in quantity L. Household consumption preferences are

$$\mathbb{E}_0 \sum_{t=0}^{\infty} \beta^t u(c_t; Z_t),$$

where period utility u may depend on components of Z_t, capturing preference shocks and shifts in the marginal utility of wealth (analogous to the discount-rate shock m_t in Section 2.2). The consumption bundle is

$$c_t = \left[\sum_j \omega_j(Z_t)^{\frac{1}{\varepsilon}} c_{jt}^{\frac{\varepsilon-1}{\varepsilon}} \right]^{\frac{\varepsilon}{\varepsilon-1}},$$

where the CES share parameters $\omega_j(Z_t)$ may depend on Z_t. Unlike the symmetric demand in the simple model, this structure explicitly allows aggregate shocks to shift the relative demand composition across different goods. Cost minimization implies

$$c_{jt} = \omega_j(Z_t)\left(\frac{p_{jt}}{P_t}\right)^{-\varepsilon} c_t \qquad \text{and} \qquad P_t = \left[\sum_j \omega_j(Z_t)\, p_{jt}^{1-\varepsilon}\right]^{\frac{1}{1-\varepsilon}},$$

where P_t is the consumer price index and p_{jt} denotes the price of good j. We use nominal expenditure as the numeraire, setting $P_t c_t = 1$ for all t. The household's stochastic discount factor (SDF) is

$$M_{t,t+1} = \beta \frac{u_c(c_{t+1}; Z_{t+1})}{u_c(c_t; Z_t)},$$

which depends on Z_t both directly (through preferences) and indirectly (through equilibrium consumption).

Production. Each firm j produces gross output using a nested production technology,

$$y_{jt} = A_t F_j(h_{jt}, x_{jt}; \xi_{jt}), \tag{13}$$

where h_{jt} is a labor aggregate, $x_{jt} = (x_{j1,t}, \ldots, x_{jJ,t})$ is the vector of intermediate inputs purchased from other firms, and ξ_{jt} is a firm-specific shock. The labor aggregate combines software engineering and non-SWE labor according to

$$h_{jt} = H_j(S_t s_{jt}, L_{jt}), \tag{14}$$

where s_{jt} and L_{jt} are firm j's employment of SWE and non-SWE labor. Both F_j and H_j exhibit constant returns to scale in their arguments and may vary across firms.[4]

The shock S_t enters as strictly factor-augmenting in (14), scaling the effective quantity of software engineering labor exactly as in Section 2.2. At the margin, a one percent increase in S_t is equivalent to a one percent increase in software engineering employment at every firm. The scalar $A_t = f(Z_t)$ acts as a Hicks-neutral shifter driven by the aggregate shock vector Z_t: it multiplies the entire production function, reducing all firms' unit costs proportionally. Crucially, Hicks-neutrality restricts the *within-firm* input margin — aggregate shocks in Z_t do not distort relative marginal products across inputs — but it does not restrict cross-sectional heterogeneity. Different firms may still respond differently to A_t through variation in their production functions $F_j(\cdot)$ and their positions within the input-output network. The firm-specific shock ξ_{jt} is unrestricted.

The total cost of firm j is $\mathscr{C}_{jt} = w_t s_{jt} + W_t L_{jt} + \sum_k p_{kt} x_{jk,t}$, and its unit cost is $mc_{jt} = \mathscr{C}_{jt}/y_{jt}$. Con-

[4] The nested structure (13)–(14) embeds weak separability of the two labor types from intermediate inputs: conditional on the labor aggregate h_{jt}, the marginal rate of substitution between any two intermediates does not depend on (s_{jt}, L_{jt}), and vice versa. It rules out asymmetric complementarities — for instance, software engineering labor being more complementary with some intermediate inputs than non-software-engineering labor.

stant returns and the nested structure of (13)–(14) have sharp implications for unit costs: A_t enters as a proportional cost reduction common to all firms (Hicks-neutrality), while S_t appears exclusively through the effective software engineering wage w_t/S_t (factor-augmenting). Firms face CES demand and charge a constant markup $p_{jt} = \frac{\varepsilon}{\varepsilon-1} mc_{jt}$. Consistent with the simple model, per-period profits are a constant fraction of nominal revenue, $\pi_{jt} = \frac{1}{\varepsilon} p_{jt} y_{jt}$.

We define firm j's cost shares for software engineering, other labor, and intermediate inputs as

$$\gamma_{jt} = \frac{w_t s_{jt}}{\mathscr{C}_{jt}}, \qquad \theta_{jt} = \frac{W_t L_{jt}}{\mathscr{C}_{jt}}, \qquad \text{and} \qquad \Omega_{jk,t} = \frac{p_{kt} x_{jk,t}}{\mathscr{C}_{jt}},$$

which sum to unity, $\gamma_{jt} + \theta_{jt} + \sum_k \Omega_{jk,t} = 1$. We define the value-added share as $\alpha_{jt} = \gamma_{jt} + \theta_{jt}$.

The value of firm j is the present discounted value of its profit stream,

$$V_{jt} = \mathbb{E}_t\Big[\sum_{\tau=0}^{\infty} M_{t,t+\tau} \pi_{j,t+\tau}\Big]. \tag{15}$$

Because profits depend on the full sequence of future shocks, V_{jt} is a nonlinear function of the current state (S_t, Z_t, ξ_t) and the stochastic structure governing their evolution.

Market clearing and equilibrium. Goods market clearing requires that gross output equals the sum of final consumption and intermediate demand,

$$y_{jt} = c_{jt} + \sum_k x_{kj,t}.$$

Labor market clearing requires $\sum_j s_{jt} = s$ and $\sum_j L_{jt} = L$. We continue to use nominal expenditure as the numeraire, setting $P_t c_t = 1$ for all t. We assume that firms are small relative to the AI index (granularity), so that individual idiosyncratic shocks ξ_{jt} are diversified away in V_t^{AI}.

Taking as given stochastic processes $\{S_t, Z_t, \xi_{jt}\}_{jt}$, a competitive equilibrium comprises quantities $\{y_{jt}, x_{jt}, c_{jt}, s_{jt}, L_{jt}\}_{jt}$ and prices $\{w_t, W_t, p_{jt}\}_{jt}$ such that households and firms optimize, and markets clear.

Let Ω_t denote the $J \times J$ matrix of intermediate-input cost shares and $\Lambda_t = (I - \Omega_t)^{-1}$ the Leontief inverse. The identity $\gamma_{jt} + \theta_{jt} + \sum_k \Omega_{jk,t} = 1$ implies $\Lambda_t \alpha_t = \mathbf{1}$, where α_t is the vector of value-added shares. We define firm j's Domar weight or sales share λ_{jt} as the ratio of its sales to total final expenditures, $\frac{p_{jt} y_{jt}}{\sum_j p_{jt} c_{jt}}$. Under our numeraire normalization, this conveniently implies $\lambda_{jt} = p_{jt} y_{jt}$.

Finally, we denote by Σ_j the matrix of firm j's Allen-Uzawa substitution elasticities, which govern input substitution in response to relative price changes. The entry σ_j^{hk} measures the elasticity of substitution between the labor aggregate h_j and intermediate input k in the outer production function F_j. The entry $\sigma_j^{k\ell}$ measures the elasticity between intermediates k and ℓ. These are the Allen-Uzawa elasticities, defined via the cost function as $\sigma_j^{hk} = \frac{\mathscr{C}_j \cdot \partial^2 \mathscr{C}_j / (\partial h_j^* \partial p_k)}{(\partial \mathscr{C}_j / \partial h_j^*) \cdot (\partial \mathscr{C}_j / \partial p_k)}$, where $\mathscr{C}_j$ is firm j's outer cost function evaluated at input prices $(h_j^*, p_1, \ldots, p_J)$, and analogously for the intermediate-intermediate

pairs. We also denote by σ_j^H the elasticity of substitution between software engineering and non-SWE labor in the inner function H_j. Together, the collection $\Sigma_j = (\sigma_j^H, \{\sigma_j^{hk}\}_k, \{\sigma_j^{k\ell}\}_{k,\ell})$ fully summarizes the substitution parameters relevant to firm j's cost structure.

5.2 First-Order Responses to Shocks

We characterize the first-order response of firm values to a perturbation $(d\log S_t, d\log Z_t, d\log \xi_t)$ at date t. We drop t subscripts when referring to steady-state values. Because financial markets are forward-looking, firm values capitalize not just the contemporaneous shock but the entire discounted stream of future changes. As in Section 2.2, define the normalized cumulative software engineering productivity change,

$$\mathscr{S}_t \equiv (1-\beta)\mathbb{E}_t\Big[\sum_{\tau=0}^{\infty}\beta^\tau d\log S_{t+\tau}\Big],$$

as in Section 2.2. If the shock is permanent, then $\mathscr{S}_t = d\log S_t$: the normalized cumulative change coincides with the per-period productivity change. Define $\mathscr{Z}_{nt}$ analogously for each component of the aggregate shock vector Z_t. These normalized cumulative changes summarize the entire expected future path of each shock without restricting its time-series dynamics.

Lemma 2 (Equilibrium Responses)**.** *To first order:*

(i) The profit response of firm j is

$$d\log \pi_{jt} = b_j d\log S_t + \sum_n b_{jn}^Z d\log Z_{nt} + \sum_k b_{jk}^\xi d\log \xi_{kt},$$

where the elasticities b_j, b_{jn}^Z, and b_{jk}^ξ depend on cost shares, the IO matrix Ω, the demand elasticity ε, and substitution elasticities. Appendix A.4 constructs these objects.

(ii) The firm-value response is $d\log V_{jt} = (1-\beta)\mathbb{E}_t\sum_{\tau=0}^{\infty}\beta^\tau(d\log\pi_{j,t+\tau} + d\log M_{t,t+\tau})$. Since the elasticities b_j, b_{jn}^Z, and b_{jk}^ξ are evaluated at steady state, the linearization applies identically at every future horizon. Substituting the period-by-period profit response from part (i) and recognizing the normalized cumulative change definitions yields

$$d\log V_{jt} = b_j\mathscr{S}_t + \sum_n b_{jn}^Z\mathscr{Z}_{nt} + \mu_t, \tag{16}$$

where μ_t is firm-invariant, collecting all present-value responses symmetric across firms.[5]

The economics behind the identification result developed in Section 5.3 is a factor-price equalization result. Because A_t scales all firms' production frontiers without distorting relative marginal

[5] Idiosyncratic shocks ξ_{kt} contribute analogous NPV terms, but these are diversified away in aggregate portfolios and play no role in our identification strategy.

products, both factor prices respond equally ($d\log w_t = d\log W_t$) and the within-value-added factor composition γ_{jt}/α_{jt} is undistorted. Components of Z_t also affect profits through demand weights $\omega_j(Z_t)$ and through the stochastic discount factor. The SDF channel is firm-invariant and absorbed by μ_t. The demand-weight channel generates cross-sectional variation in b_{jn}^Z, but this variation runs through ω_j—which goods consumers shift toward—not through within-firm factor composition. This is why ω_j appears as an argument of g_n in our formal identification proofs.

By contrast, S_t changes relative factor prices because it is SWE-augmenting rather than Hicks-neutral. The effective software engineering wage w_t/S_t falls as S_t rises, creating a differential cost advantage for software-engineering-intensive firms. The resulting cross-sectional variation in b_j is correlated with γ_j, and this asymmetry is the foundation of the identification strategy developed in Section 5.3.

AI index and control factors. We also define an AI stock index with price V_t^{AI} and a vector of control factors $F_t = (F_{1t}, F_{2t}, \ldots)$ — including e.g. market return, Fama-French HML, SMB. Both represent aggregate portfolios of firm values, and we assume that they are unaffected by idiosyncratic shocks ξ_{jt}. Since these portfolios capitalize the same normalized cumulative shocks as individual firms, they admit representations in terms of $\mathscr{S}_t$ and $\mathscr{Z}_{nt}$. To first order, the AI index satisfies

$$R_t^{\text{AI}} = \phi^S \mathscr{S}_t + \sum_n \phi_n^Z \mathscr{Z}_{nt}, \tag{17}$$

where ϕ^S and ϕ_n^Z are the index's loadings on the normalized cumulative fundamental aggregate shocks. The only substantive requirement on the index is $\phi^S \neq 0$: it loads on AI productivity. Nonzero ϕ_n^Z is expected and allowed — the index may respond to other aggregate shocks. Similarly, the control factors satisfy

$$F_{it} = \psi_i^S \mathscr{S}_t + \sum_n \psi_{in}^Z \mathscr{Z}_{nt}, \tag{18}$$

with unrestricted loadings ψ_i^S and ψ_{in}^Z. Finally, the risk-free rate admits the representation

$$R_t^f = \chi^S \mathscr{S}_t + \sum_n \chi_n^Z \mathscr{Z}_{nt}, \tag{19}$$

with loadings χ^S and χ_n^Z on the normalized cumulative shocks.

5.3 Identification, Inversion, and GDP Impact

This section bridges our quantitative general equilibrium model and the two-stage empirical regression analysis introduced in Section 2.1. Building on the intuition developed in the simple model (Section 2.2), we formally establish that our two-stage empirical procedure can recover AI's impact on GDP through the software productivity channel in a realistic economy with input-output linkages,

unobserved macroeconomic shocks, and cross-sectional demand shifts.

We structure the analysis in three steps. First, we provide a structural characterization of the empirical second-stage coefficient δ. Second, we establish the identification conditions under which our cross-sectional regression successfully purges confounding effects from other aggregate shocks and from AI impacts operating through non-software-engineering channels. Finally, we invert this structural mapping to translate observed asset-price movements into implied AI productivity gains, and apply Hulten's theorem to evaluate their impact on aggregate GDP.

5.3.1 A Structural Characterization of the Second-Stage Coefficient δ

To connect the model to the data, we replicate our two-stage empirical procedure from Section 2.1 within the structural model.

Recall that our empirical strategy begins with the residualized excess AI index return, $\tilde{R}_t^{\text{AI}}$. This is the component of the AI index return $(R_t^{\text{AI}} - R_t^f)$ that is orthogonal to our control factors F_t. Using the factor representations (17)–(19), this residualized index can be expressed as a linear combination of fundamental shocks:

$$\tilde{\phi}^S = (\phi^S - \chi^S) - \sum_i \hat{\beta}_i \psi_i^S \qquad \text{and} \qquad \tilde{\phi}_n^Z = (\phi_n^Z - \chi_n^Z) - \sum_i \hat{\beta}_i \psi_{in}^Z, \tag{20}$$

where $\hat{\beta}_i$ are the population projection coefficients of the AI index on the control factors.

First stage: Firm-level betas on the AI index (β_j). In the first stage of our empirical approach, we estimate each firm's AI beta by regressing its excess return on the residualized AI index return. The population counterpart to this coefficient is $\beta_j = \text{Cov}_t(d\log V_{jt} - R_t^f, \tilde{R}_t^{\text{AI}})/\text{Var}_t(\tilde{R}_t^{\text{AI}})$. By substituting the firm-value response from Equation (16), we can decompose firm j's beta as:

$$\beta_j = \beta_0 + \kappa b_j + \sum_n \kappa_n^Z b_{jn}^Z. \tag{21}$$

Here, β_0 is a firm-invariant term that absorbs symmetric responses to the stochastic discount factor and the risk-free rate. The parameters κ and κ_n^Z are the population projection coefficients of the normalized cumulative shocks onto the residualized index:

$$\kappa = \frac{\text{Cov}_t(\mathscr{S}_t, \tilde{R}_t^{\text{AI}})}{\text{Var}_t(\tilde{R}_t^{\text{AI}})} \qquad \text{and} \qquad \kappa_n^Z = \frac{\text{Cov}_t(\mathscr{Z}_{nt}, \tilde{R}_t^{\text{AI}})}{\text{Var}_t(\tilde{R}_t^{\text{AI}})}.$$

Intuitively, κ, the object of interest, measures how much a one-unit increase in the residualized AI index return predicts an increase in the normalized cumulative software engineering productivity change ($\mathscr{S}_t$). On the other hand, κ_n^Z measures how much a one-unit increase in the residualized AI index return predicts an increase in the normalized cumulative change of the other aggregate shocks ($\mathscr{Z}_{nt}$).

Second stage: The cross-sectional slope (δ). In our second empirical stage, we regress these estimated betas (β_j) on firms' software engineering cost shares (γ_j), conditional on a set of controls X_j. The population coefficient for this cross-sectional slope is:

$$\delta = \frac{\text{Cov}_j(\beta_j, \gamma_j | X_j)}{\text{Var}_j(\gamma_j | X_j)}.$$

Substituting our expression for β_j from Equation (21) into this covariance yields:

$$\delta = \underbrace{\kappa \delta^S}_{\text{Signal}} + \underbrace{\sum_n \kappa_n^Z \delta_n^Z}_{\text{Contamination}}, \tag{22}$$

where we define the structural cross-sectional gradients as:

$$\delta^S = \frac{\text{Cov}_j(b_j, \gamma_j | X_j)}{\text{Var}_j(\gamma_j | X_j)} \qquad \text{and} \qquad \delta_n^Z = \frac{\text{Cov}_j(b_{jn}^Z, \gamma_j | X_j)}{\text{Var}_j(\gamma_j | X_j)}.$$

Equation (22) transparently illustrates our empirical challenge. The first term is the *signal*: the component of δ that reflects the impact of AI through the software engineering productivity channel, as characterized in Section 4.2. Specifically, δ^S captures the cross-sectional gradient between a firm's software-engineering-driven profit elasticity (b_j) and its software engineering intensity (γ_j), which can be used to recover the object of interest, κ, the coefficient in the best linear predictor of the normalized cumulative software engineering productivity change given the residualized AI index. The subsequent terms represent *contamination*: the cross-sectional covariance between non-software-engineering profit elasticities (b_{jn}^Z) and software engineering intensity. To use our cross-sectional estimate δ to recover κ and hence the GDP impact of AI through the software engineering channel, our strategy requires these contamination terms to vanish (i.e., $\delta_n^Z = 0$ for all n).

5.3.2 Identification and Inversion

Equation (22) poses a classic identification problem: the reduced-form estimate δ conflates the software engineering productivity signal with contamination from other aggregate shocks. Recovering κ, and hence the GDP impact of AI through the software engineering channel, requires conditions under which the contamination terms vanish ($\delta_n^Z = 0$ for all n).

In the stylized model of Section 2.2, identification held trivially because the other aggregate shocks were assumed to be Hicks-neutral on the supply side and did not shift relative demand across goods, leaving relative factor prices—and thus the relative costs of software-engineering-intensive firms—undistorted.

In our extended environment, however, the other aggregate shocks Z_t can be arbitrarily complex and explicitly shift the relative demand composition across goods via $\omega_j(Z_t)$. The primary threat to our identification strategy is that an aggregate shock Z_{nt} might systematically shift consumer demand

toward SWE-intensive goods. Such a demand shift would disproportionately increase the demand for software engineering labor, driving up the software engineering wage and causing marginal costs to change in proportion to γ_j. This demand-driven mechanism would generate profit differentials proportional to γ_j, exactly as a true software engineering productivity shock does, making the two empirically indistinguishable.

To formally rule out this specific threat, we impose the following condition on firm-specific exposure to aggregate demand shocks:

Assumption 1. *(Conditional factor-neutral demand composition) For each aggregate shock Z_{nt}, the steady-state demand-weight elasticities $\omega'_{jn} = \partial \log \omega_j / \partial \log Z_n|_{ss}$ satisfy*

$$Cov_j(\lambda_j \gamma_j, \omega'_{jn} \mid X_j) = 0,$$

where $\lambda_j \gamma_j$ is the Domar-weighted software engineering cost share of firm j, and X_j is the vector of controls.

Assumption 1 guarantees that when an aggregate shock Z_{nt} hits the economy, any resulting shifts in consumer preferences do not systematically favor software-intensive products conditional on controls X_j, which include industry fixed effects. A broad macroeconomic shock that uniformly lifts demand satisfies this condition trivially, as does a shock that shifts demand across goods but in a way that is conditionally orthogonal to their software engineering cost shares. Since X_j includes industry fixed effects, the assumption is also satisfied by any shock that shifts demand across industries but not differentially toward high-γ_j firms within an industry. The assumption would only be violated by a confounding shock that, conditional on X_j, shifts demand toward firms with high software engineering cost shares within the same industry—for example, a sudden shift in consumer tastes specifically toward software-heavy products within an industry that coincides with, but is entirely distinct from, actual AI-driven software engineering productivity growth.

With this condition in place, we state our main theoretical identification result.

Proposition 3 (Identification and Inversion)**.** *Suppose that for any two firms j and k with identical controls $X_j = X_k$ (which include industry fixed effects), their production functions and baseline consumer demand weights are identical: $F_j(\cdot) = F_k(\cdot)$ and $\omega_j(\cdot) = \omega_k(\cdot)$. Then, under Assumption 1, a firm's profit exposure to any aggregate shock Z_{nt} is uncorrelated with its software engineering cost share γ_j conditional on X_j:*

$$Cov_j\left(b^Z_{jn}, \gamma_j \middle| X_j\right) = 0.$$

Consequently, $\delta^Z_n = 0$ for all n, and the decomposition in Equation (22) collapses to a pure reflection of

the AI's impact through software engineering productivity, uniquely identifying

$$\kappa = \frac{\delta}{\delta^S}. \tag{23}$$

Proposition 3 establishes the conditions under which our empirical strategy can isolate the macroeconomic impact of AI through software engineering productivity. The economic mechanism underlying this result relies on the principle of factor-price equalization discussed in Section 2.2. Because the aggregate shocks Z_t act as Hicks-neutral shifters within the production network, they do not distort the relative marginal products of software engineering versus non-SWE labor. Consequently, both factor wages move in lockstep, and the resulting profit differences remain perfectly orthogonal to software engineering intensity conditional on X_j. By contrast, the SWE-augmenting shock S_t fundamentally lowers the effective software engineering wage, granting a definitive structural cost advantage to software-engineering-intensive firms that our cross-sectional regression slope δ reliably detects.

In practice, IO linkages (Ω_j), baseline demand weights (ω_j), and substitution elasticities cluster tightly by industry. Including industry fixed effects as part of X_j thus effectively controls for these objects and neutralizes the contamination terms. These controls close off both channels through which an aggregate shock Z_{nt} could generate a spurious factor-price wedge: Hicks-neutrality prevents supply-side distortions conditional on X_j, while Assumption 1 prevents demand-side distortions.

With κ identified, the best linear predictor of the normalized cumulative software engineering productivity change given the AI index is

$$\mathbb{E}[\mathscr{S}_t \mid \tilde{R}_t^{\text{AI}}] = \kappa \tilde{R}_t^{\text{AI}} = \frac{\delta}{\delta^S} \tilde{R}_t^{\text{AI}}. \tag{24}$$

We obtain δ directly from our empirical regressions. The structural gradient δ^S encapsulates how strongly a given productivity shock translates into cross-sectional profit differences based on demand and substitution elasticities: it converts observed asset-price exposures into implied productivity gains.

Crucially, our procedure does not require the AI index to be a flawless, "pure" measure of AI-driven software engineering productivity S_t. The index only needs to possess some genuine exposure to software engineering productivity ($\phi^S \neq 0$). Even if the AI index responds heavily to other aggregate shocks ($\phi_n^Z \neq 0$), Proposition 3 ensures this noise does not systematically bias our structural estimate of δ. It merely reduces the signal-to-noise ratio κ, which naturally widens standard errors but creates no structural bias.

5.3.3 GDP Impact

We conclude by translating the recovered normalized cumulative software engineering productivity change $\mathscr{S}_t$ into an aggregate macroeconomic impact on real output. To first order, this mapping

follows directly from Hulten's theorem. Define the normalized cumulative GDP change through the software engineering channel,

$$\mathscr{Y}_t \equiv (1-\beta)\mathbb{E}_t\Big[\sum_{k=0}^{\infty}\beta^k d\log Y_{t+k}\big|_S\Big],$$

in parallel to $\mathscr{S}_t$. The result highlights the Domar-weighted mean software engineering cost share across the economy,

$$\bar{\gamma} = \sum_j \lambda_j \gamma_j,$$

as the critical sufficient statistic for assessing the aggregate macroeconomic software productivity impact of AI.

Proposition 4 (GDP impact of AI)**.** *Under the conditions of Proposition 3, since* $d\log Y_{t+k}|_S = \bar{\gamma}\, d\log S_{t+k}$ *period by period,*

$$\mathscr{Y}_t = \bar{\gamma}\mathscr{S}_t. \tag{25}$$

The best linear predictor of the normalized cumulative GDP change through the software engineering channel given the residualized AI-index return is

$$\mathbb{E}[\mathscr{Y}_t \mid \tilde{R}_t^{AI}] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{AI}. \tag{26}$$

Equation (26) delivers a highly transparent sufficient-statistic formula. The expected macroeconomic impact of an AI-driven stock market movement can be calculated directly from the residualized AI index return, scaled by a composite multiplier $\bar{\gamma}\kappa$. This multiplier cleanly decomposes into three distinct conceptual steps:

$$\bar{\gamma}\kappa \quad = \quad \underbrace{\delta}_{\substack{\text{observed}\\ \text{cross-sectional slope}}} \quad \times \quad \underbrace{\frac{1}{\delta^S}}_{\substack{\text{inversion:}\\ \text{index return}\to\mathscr{S}}} \quad \times \quad \underbrace{\bar{\gamma}}_{\substack{\text{Hulten weight:}\\ \mathscr{S}\to\mathscr{Y}}}$$

This breakdown exactly mirrors the logical sequence of our paper. First, we rely on the *data* to provide the cross-sectional slope (δ). Second, we use the *model inversion* to translate that financial asset-pricing moment into unobserved structural productivity gains ($1/\delta^S$). Third, we apply the *macroeconomic Hulten weight* ($\bar{\gamma}$) to aggregate those granular productivity gains into a top-line GDP effect. Consequently, a one percent increase in the residualized AI index return maps to a predicted $\bar{\gamma}\kappa$ percent increase in the normalized cumulative GDP change through the software engineering channel.

The economic intuition behind $\bar{\gamma}$ is straightforward: it represents the total share of aggregate economic costs attributable to software engineers, fully accounting for the extensive web of upstream and downstream intermediate input linkages. At present, $\bar{\gamma}$ remains small relative to the overall econ-

omy. Because the first-order GDP effect is bounded by this cost share, the macroeconomic impact of AI operating strictly through the software engineer productivity channel is correspondingly bounded.

Demand shocks versus supply shocks. This result underscores why Assumption 1 (ruling out confounding demand shocks) is so critical. A true software engineering productivity shock S_t physically expands the economy's production possibility frontier, delivering genuine GDP growth equal to $\bar{\gamma} d \log S_t$ period by period. Conversely, a pure shift in consumer demand composition merely reallocates existing output across the frontier without increasing total capacity. Due to the envelope theorem and constant markups, such a reallocation generates exactly zero first-order increase in aggregate GDP. If demand-composition contamination had falsely inflated our measured δ, our inversion formula would incorrectly interpret expenditure reshuffling as structural productivity growth, thereby overstating AI's real macroeconomic impact. By ensuring δ reflects only productivity, our strategy guarantees that the measured GDP effect represents true technological expansion.

5.4 Quantification

This subsection extends the back-of-the-envelope calculation of Section 4.2 by incorporating the input-output network developed above. The annualized GDP impact formula and the Hulten result (Proposition 4) carry over unchanged; the only difference is that the structural gradient δ^S must now be computed numerically because intermediate-input linkages break the closed-form expression available in the no-IO case. As in Section 4.2, the normalized cumulative multiplier is $\kappa = \delta/\delta^S$, which depends only on the demand elasticity ε, the substitution elasticities, and the cross-sectional distribution of cost shares — not on the discount factor or on the time-series dynamics of the shocks. We adopt the same baseline calibration as in Section 4.2 (see Table 6); the IO extension introduces two additional elasticities, ζ and θ, required away from Cobb-Douglas outer production functions.

With intermediate inputs, δ^S cannot be computed in closed form. The numerical computation proceeds in four steps, following the structure of Section 5.3. First, assemble the IO cost-share matrix Ω from the BEA Make-Use Tables and the Leontief inverse $\Lambda = (I - \Omega)^{-1}$. Second, construct the revenue-response matrix $\mathscr{B}$ from the CES demand structure. Third, solve the 2×2 factor-market-clearing system for the general-equilibrium factor-price response. Fourth, compute the profit-elasticity vector b and project onto γ_j controlling for α_j.

Under Cobb-Douglas outer production functions (F_j has unit elasticity between the labor aggregate and intermediates), the IO computation requires no additional calibrated parameters beyond the data-based objects in Table 6: the IO matrix, value-added shares, and expenditure weights pin down $\mathscr{B}$ and the factor-market-clearing system. Away from Cobb-Douglas, the standard dimensional reduction following Atalay (2017) and Baqaee and Farhi (2019) imposes a nested CES structure on F_j:

a top nest between the labor aggregate h_j and a materials composite m_j with elasticity ζ (the value-added–intermediates elasticity), and a bottom nest across individual intermediates within m_j with elasticity θ (the across-intermediates elasticity). This reduces the parameter space from $J(J+1)/2$ Allen-Uzawa elasticities per sector to two numbers.

The literature places ζ relatively precisely at 0.5–0.7, with Atalay (2017) estimating $\zeta \approx 0.6$ using military-spending instruments and Peter and Ruane (2023) confirming the same value using Indian tariff reforms. The across-intermediates elasticity θ is more uncertain: the short-run estimates of Atalay (2017) yield $\theta < 0.2$ (strong complementarity), while the long-run estimates of Peter and Ruane (2023) give $\theta \approx 2$ (substitutes). Miranda-Pinto and Young (2022) allow sector-specific θ_j and find most estimates in $[0,1]$. For the present application, the relevant horizon is ambiguous, since asset prices capitalize a mix of short-run and long-run effects. Our recommended approach is to use $\zeta = 0.6$ and vary θ over $[0.2, 2.0]$ as a robustness exercise. In practice, the IO extension has a modest quantitative effect on δ^S relative to the no-IO formula, because the dominant channel — the within-firm cost reduction from cheaper effective software engineering labor — operates through σ and ε rather than through the IO network.

Back-of-the-envelope calculation. Under Cobb-Douglas outer production functions at the baseline ($\varepsilon = 5$, $\sigma = 1$), the structural gradient is $\delta^S \approx 3.01$, the same as the closed-form expression in Section 4.2. The normalized cumulative multiplier is

$$\kappa = \frac{\delta}{\delta^S} = \frac{5.8}{3.01} \approx 1.93,$$

From November 2022 to December 2025, the cumulative residualized AI-index return is $\tilde{R}_t^{\text{AI}} = 14.9\%$. Multiplying by κ gives a market-implied normalized cumulative software engineering productivity change from news about AI from November 2022 to December 2025 of $\kappa \cdot \tilde{R}_t^{\text{AI}} \approx 28.8\%$, and a corresponding normalized cumulative GDP change through the software engineering channel of $\bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}} \approx 0.53\%$, comparable to the no-IO model of Section 4.2. This confirms that the dominant channel is the within-firm cost reduction from cheaper effective software engineering labor, not the propagation of cost savings through intermediate-input linkages.

Table 8 reports the structural gradient δ^S, the normalized cumulative multiplier $\kappa = \delta/\delta^S$, and the normalized cumulative GDP change $\bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}}$ across the same (σ, ε) grid as Table 7, now incorporating the IO network under Cobb-Douglas outer production functions. The results illustrate the same two key features identified in Section 4.2. First, the annualized GDP impact ranges from roughly 0.35% at the most conservative calibration ($\sigma = 0.5$, $\varepsilon = 5$) to 5.1% at the most aggressive ($\sigma = 2$, $\varepsilon = 2$), with the baseline ($\sigma = 1$, $\varepsilon = 5$) delivering 0.53%. Second, the within-firm substitution elasticity σ remains the most influential structural parameter, because higher substitutability allows software engineering

productivity gains to diffuse more broadly through the firm's cost structure. The demand elasticity ε has a comparatively moderate effect for $\varepsilon \geq 3$, reflecting the approximate cancellation of $(\varepsilon - 1)$ between δ^S and the product-market force in the profit elasticity.

Robustness. The IO-specific elasticities ζ and θ have limited influence on the results. At the baseline ($\sigma = 1, \varepsilon = 5$), varying the value-added–intermediates elasticity over $\zeta \in [0.5, 0.7]$ changes the annualized GDP impact by only a few basis points, and varying the across-intermediates elasticity over $\theta \in [0.2, 2.0]$ has a similarly small effect. This insensitivity arises because the dominant channel — the within-firm cost reduction from cheaper effective software engineering labor — operates through σ and ε rather than through the IO network.

6 Conclusion

This paper uses asset prices to infer the macroeconomic impact of AI through the software engineering channel. By exploiting the cross-sectional relationship between firms' AI-index betas and their software engineering cost shares, and inverting this relationship through a structural production-network model, we estimate that news about AI from November 2022 to December 2025 implies an annualized cumulative AI-induced software engineering productivity increase of roughly 28.8 percent, and a corresponding annualized cumulative GDP impact through the software engineering channel of approximately 0.53 percent.

Our approach naturally extends to other occupation groups through which AI may affect the macroeconomy, by applying the same cross-sectional identification strategy with occupation-specific cost shares. Beyond these level effects on productivity, an increase in software engineering productivity may also raise the rate of innovation—by lowering the cost of building and iterating on new software products—potentially generating dynamic gains that compound over time. Sizing such growth-rate effects is beyond the scope of this paper but is an important direction for future work.

Tables

Table 1: Sample selection

		Percentiles of 2021 Sales				
Filter	Remaining Firms	P10	P25	P50	P75	P90
Initial CRSP dataset (active GVKEY links)	5,724	9.3	69.9	498.0	2,655.7	11,272.1
Compustat merge, requiring AT>0 and SALE>0	5,140	9.3	69.9	498.0	2,655.7	11,272.1
Drop NAICS 52/53/51/54	3,285	6.0	50.0	570.4	3,201.5	12,786.2
Drop semiconductor-related firms	3,147	5.5	47.4	540.1	3,028.0	12,365.9
Revelio match	2,138	6.7	62.7	610.1	3,320.7	12,377.8
Average sales/assets filter	1,908	9.1	85.6	709.7	3,533.0	12,707.1
Balanced panel	1,896	9.6	89.9	718.1	3,564.0	12,732.3
Revelio size trim	1,794	10.9	103.9	808.8	3,803.8	13,280.0

Notes: This table documents the progression of the analysis sample as sequential filters are applied. Each row reports the number of firms remaining after the indicated filter, along with selected percentiles of the cross-sectional distribution of 2021 net sales (Compustat variable `sale`, in millions of USD). The initial dataset consists of all CRSP securities with an active GVKEY link to Compustat. The Compustat merge retains firms with strictly positive total assets and sales. The NAICS filter drops Finance and Insurance (52), Real Estate (53), Information (51), and Professional/Technical Services (54). The semiconductor filter removes firms involved in semiconductor development or production. The Revelio match retains firms with workforce data in 2021. The average sales/assets filter requires strictly positive average assets and sales over 2017–2021 and strictly positive sales in each year from 2022 to 2024. The balanced panel retains firms with stock price data for all weeks of 2022 through 2025. The Revelio size trim drops the bottom 5% of firms by total Revelio employment.

Table 2: Twenty firms with the highest software engineer payroll shares

Rank	Firm	NAICS2	Market Cap (2021)
1	Jumia Technologies AG	44-45	$0.9 B
2	Innoviz Technologies Ltd	31-33	$0.9 B
3	Full Truck Alliance Co Ltd	48-49	$0.7 B
4	Ubiquiti Inc	31-33	$19.2 B
5	PDD Holdings Inc	44-45	$25.8 B
6	Cimpress plc	31-33	$1.9 B
7	Energous Corp	31-33	$0.1 B
8	Mastech Digital Inc	56	$0.2 B
9	Mogu Inc	44-45	$0.0 B
10	Ciena Corp	31-33	$11.9 B
11	Calix Inc	31-33	$5.1 B
12	Arlo Technologies Inc	31-33	$0.9 B
13	Cambium Networks Corp	31-33	$0.7 B
14	Sonos Inc	31-33	$3.8 B
15	Extreme Networks Inc	31-33	$2.0 B
16	Comtech Telecommunications	31-33	$0.6 B
17	eBay Inc	44-45	$39.5 B
18	Harmonic Inc	31-33	$1.2 B
19	Motorola Solutions Inc	31-33	$45.9 B
20	Expedia Group Inc	56	$27.1 B

Notes: This table reports the 20 firms with the highest software engineer payroll shares in the analysis sample. The software engineer payroll share is the fraction of total firm compensation in Revelio attributable to workers in the "Software Engineer" role category, measured in 2021. Market capitalization is end-of-2021, reported in billions of U.S. dollars. NAICS2 is the two-digit North American Industry Classification System code.

Table 3: Regression of THNQ beta on software engineer payroll share (continuous)

Dependent Variable:	Beta on THNQ excess return				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
Software engineer payroll share	1.85***	1.96***	1.98***	1.54***	1.55***
	(0.21)	(0.21)	(0.21)	(0.23)	(0.34)
Fixed-effects					
Quintile of average 2017-2021 assets		Yes	Yes	Yes	Yes
Quintile of average 2017-2021 sales		Yes	Yes	Yes	Yes
NAICS2			Yes		
NAICS3				Yes	Yes
Quintile of GenAI task exposure (Eisfeldt, Schubert and Zhang, 2023)					Yes
Fit statistics					
Observations	1,794	1,794	1,794	1,787	1,103
R^2	0.06	0.10	0.12	0.16	0.24
Within R^2		0.07	0.07	0.03	0.04

HC3 standard-errors in parentheses

*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Notes: The dependent variable is the firm-level beta on THNQ excess returns, estimated in a first-stage weekly regression of firm excess returns on THNQ excess returns with FF3 controls over the 2022-2025 balanced panel. The key regressor is the software engineer payroll share, measured using Revelio and defined as the share of firm payroll attributable to software engineer occupations. Columns differ only in the controls and fixed effects indicated in the table. HC3 standard errors are reported in parentheses. Observations are firms.

Table 4: Robustness checks

		Coefficient on software engineer share measure	Std. Error	Observations
1.	Baseline	1.54	0.23	1787
2.	Drop micro-cap firms	1.74	0.34	994
3.	Alternate AI index: AIQ	1.67	0.37	1787
4.	NASDAQ control in first stage	1.54	0.24	1787
5.	FF5 factors in first stage	1.28	0.27	1787
6.	Weight by end-of-2021 market cap	1.43	0.49	1787
7.	Control for log(end-of-2021 market cap)	1.68	0.24	1787
8.	Weight by software engineer payroll share	1.21	0.31	1591
9.	Use software engineer employment share	1.44	0.24	1787
10.	Control for book-to-market	1.52	0.24	1665
11.	Control for operating profitability	1.44	0.27	1359
12.	Control for capex/assets ratio	1.59	0.23	1749
13.	Control for book leverage	1.63	0.23	1746
14.	HQ state FE	1.20	0.26	1610
15.	US-incorporated firms only	1.16	0.24	1571
16.	NAICS4 FE	1.53	0.27	1699
17.	NAICS6 FE	1.47	0.34	1433
18.	Bootstrap SE (block + firm resample)	1.54	0.42	1787

Notes: Each row reports the estimated coefficient on the software engineer share measure from a one-at-a-time departure from the baseline specification (row 1), which uses NAICS3 fixed effects, size quintile controls, and HC3 standard errors. Row 2 drops firms with end-of-2021 market capitalization below $1 billion. Row 3 replaces the baseline AI index (THNQ) with the alternative index AIQ. Rows 4–5 add first-stage controls: row 4 adds the excess return on QQQ (NASDAQ 100) to absorb general technology shocks; row 5 replaces the Fama-French three-factor model with the five-factor model, adding momentum (UMD) and profitability (RMW). Row 6 weights the second-stage regression by end-of-2021 market capitalization; row 7 adds log market capitalization as a control. Row 8 weights by the software engineer payroll share; row 9 replaces the payroll share with the raw employment share of software engineers. Rows 10–13 add firm-level controls: book-to-market ratio, operating profitability, capital expenditure to assets ratio, and book leverage, respectively. Row 14 adds fixed effects for the U.S. state in which the firm is headquartered. Row 15 restricts the sample to U.S.-incorporated firms. Rows 16–17 replace NAICS3 fixed effects with finer industry controls at the NAICS4 and NAICS6 level. Row 18 reports bootstrap standard errors from a moving-block bootstrap (block length 6 weeks) with firm resampling, re-estimating both stages in each iteration. Unless otherwise noted, the dependent variable, software-intensity measure, and core controls are unchanged from the baseline. Observation counts vary only when the specification changes the estimation sample.

Table 5: Regression of THNQ beta on software engineer payroll share quintiles

Dependent Variable:	Beta on THNQ excess return				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
Quintile 1	-0.02	0.05	0.05	0.05	-0.02
	(0.06)	(0.06)	(0.06)	(0.06)	(0.08)
Quintile 2	-0.04	0.09	0.09*	0.07	0.00
	(0.05)	(0.06)	(0.06)	(0.06)	(0.08)
Quintile 3	-0.04	0.10*	0.11**	0.06	0.01
	(0.05)	(0.05)	(0.06)	(0.06)	(0.08)
Quintile 4	0.07	0.21***	0.23***	0.14**	0.07
	(0.05)	(0.06)	(0.06)	(0.06)	(0.08)
Quintile 5	0.27***	0.37***	0.38***	0.27***	0.21**
	(0.06)	(0.06)	(0.06)	(0.06)	(0.08)
Fixed-effects					
Quintile of average 2017-2021 assets		Yes	Yes	Yes	Yes
Quintile of average 2017-2021 sales		Yes	Yes	Yes	Yes
NAICS2			Yes		
NAICS3				Yes	Yes
Quintile of GenAI task exposure (Eisfeldt, Schubert and Zhang, 2023)					Yes
Fit statistics					
Observations	1,794	1,794	1,794	1,787	1,103
R^2	0.04	0.09	0.11	0.15	0.23
Within R^2		0.05	0.06	0.02	0.03

HC3 standard-errors in parentheses

*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Notes: Quintile indicators are constructed from the same software engineer payroll-share measure used in the continuous specification. Firms with positive software engineer payroll share are partitioned into quintiles, and coefficients are reported relative to the omitted zero-share group. Columns retain the same sequence of controls and fixed effects as in the continuous table. HC3 standard errors are reported in parentheses. Observations are firms.

Table 6: Calibrated parameter values

Parameter	Description	Baseline	Range	Source
Data-based objects				
$\bar{\gamma}^y$	Domar-weighted mean software engineering output share	0.0148	—	BLS OEWS + BEA
$\sigma_s^{2,y}$	Cross-sectional variance of γ_j^y	0.000962	—	BLS OEWS + BEA
$\tilde{\delta}$	Cross-sectional slope (payroll-share basis)	1.8	—	Table 3, col. 4
δ	Cross-sectional slope (cost-share basis)	5.8	—	$\tilde{\delta}/\alpha_L$; KLEMS $\alpha_L = 0.312$
Ω	IO cost-share matrix	—	—	BEA Make-Use
λ_j	Domar weights	—	—	BEA gross output
Calibrated parameters				
ε	Demand elasticity	5	$\{2,3,5\}$	Broda and Weinstein (2006)
σ	SWE–non-SWE substitution elast.	1.0	$\{0.5,1,1.5,2\}$	Katz and Murphy (1992); Havránek et al. (2024)
β	Annual discount factor	0.96	—	Standard
IO extension (non-Cobb-Douglas outer function)				
ζ	VA–intermediates elasticity	0.6	$[0.5,0.7]$	Atalay (2017)
θ	Across-intermediates elasticity	0.5	$[0.2,2.0]$	Atalay (2017); Peter and Ruane (2023)

Notes: "Data-based objects" are directly computable from publicly available data; the cost-share counterparts used in the model are $\bar{\gamma} = \frac{\varepsilon}{\varepsilon-1}\bar{\gamma}^y$ and $\sigma_s^2 = (\frac{\varepsilon}{\varepsilon-1})^2\sigma_s^{2,y}$. "Calibrated parameters" are taken from the literature or varied in sensitivity analysis. The IO extension parameters are required only away from Cobb-Douglas outer production functions.

Table 7: GDP multiplier sensitivity (simple model, no IO)

		δ^S			κ			$\bar{\gamma}\kappa$ $(\times 10^2)$		
		$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$
σ	0.5	1.43	2.67	4.58	4.07	2.17	1.27	12.03	4.82	2.34
	1.0	0.88	1.67	3.01	6.58	3.48	1.93	19.47	7.72	3.57
	1.5	0.64	1.21	2.24	9.09	4.78	2.60	26.90	10.62	4.80
	2.0	0.50	0.95	1.78	11.60	6.09	3.26	34.34	13.52	6.03

Notes: Data-based inputs: $\bar{\gamma}^y = 0.0148$, $\sigma_s^{2,y} = 0.000962$, $\delta = 5.8$; cost-share objects $\bar{\gamma} = \frac{\varepsilon}{\varepsilon-1}\bar{\gamma}^y$ and $\sigma_s^2 = (\frac{\varepsilon}{\varepsilon-1})^2\sigma_s^{2,y}$ vary with ε. δ^S is the structural cross-sectional gradient. $\kappa = \delta/\delta^S$ is the implied annualized cumulative software engineering productivity growth per unit of AI-index return. $\bar{\gamma}\kappa$ is the GDP multiplier, reported in percentage points $(\times 10^2)$. Each cell uses the no-IO formula from Section 2.2.

Table 8: GDP impact sensitivity (IO model, Cobb-Douglas outer)

		δ^S			κ			$\bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}}$ (%)		
		$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$	$\varepsilon = 2$	$\varepsilon = 3$	$\varepsilon = 5$
σ	0.5	1.43	2.67	4.58	4.07	2.17	1.27	1.79	0.72	0.35
	1.0	0.88	1.67	3.01	6.58	3.48	1.93	2.90	1.15	0.53
	1.5	0.64	1.21	2.24	9.09	4.78	2.60	4.01	1.58	0.72
	2.0	0.50	0.95	1.78	11.60	6.09	3.26	5.12	2.01	0.90

Notes: Data-based inputs: $\bar{\gamma}^y = 0.0148$, $\sigma_s^{2,y} = 0.000962$, $\delta = 5.8$, $\tilde{R}_t^{\text{AI}} = 14.9\%$; cost-share objects $\bar{\gamma} = \frac{\varepsilon}{\varepsilon-1}\bar{\gamma}^y$ and $\sigma_s^2 = (\frac{\varepsilon}{\varepsilon-1})^2\sigma_s^{2,y}$ vary with ε. δ^S is the structural cross-sectional gradient. $\kappa = \delta/\delta^S$ is the annualized cumulative software engineering productivity growth per unit of AI-index return. $\bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}}$ is the annualized cumulative GDP growth through the software engineering channel, reported in percent. Each cell uses the IO-augmented computation from Section 5.3 under Cobb-Douglas outer production functions.

References

Acemoglu, Daron. 2025. "The simple macroeconomics of AI." *Economic Policy*, 40(121): 13–58.

Acemoglu, Daron, and Pascual Restrepo. 2019. "Automation and new tasks: how technology displaces and reinstates labor." *Journal of Economic Perspectives*, 33(2): 3–30.

Acemoglu, Daron, and Pascual Restrepo. 2022. "Tasks, automation, and the rise in U.S. wage inequality." *Econometrica*, 90(5): 1973–2016.

Acemoglu, Daron, David Autor, Jonathon Hazell, and Pascual Restrepo. 2022. "Artificial intelligence and jobs: evidence from online vacancies." *Journal of Labor Economics*, 40(S1): S293–S340.

Acemoglu, Daron, Vasco Carvalho, Asuman Ozdaglar, and Alireza Tahbaz-Salehi. 2012. "The network origins of aggregate fluctuations." *Econometrica*, 80(5): 1977–2016.

Aghion, Philippe, and Simon Bunel. 2024. "AI and growth: where do we stand?" *Working Paper*.

Aghion, Philippe, Benjamin Jones, and Charles Jones. 2019. "Artificial intelligence and economic growth." In *The Economics of Artificial Intelligence: An Agenda*. 237–282. University of Chicago Press.

Agrawal, Ajay, John McHale, and Alexander Oettl. 2026. "Enhancing worker productivity without automating tasks: a different approach to AI and the task-based model." *NBER Working Paper*.

Althoff, Lukas, and Hugo Reichardt. 2026. "Task-specific technical change and comparative advantage." *Working Paper*.

Andrews, Isaiah, and Maryam Farboodi. 2025. "Do markets believe in transformative AI?" *NBER Working Paper*.

Ante, Lennart, and Aman Saggu. 2025. "Quantifying a firm's AI engagement: constructing objective, data-driven AI stock indices using 10-K filings." *Technological Forecasting and Social Change*.

Atalay, Enghin. 2017. "How important are sectoral shocks?" *American Economic Journal: Macroeconomics*, 9(4): 254–280.

Babina, Tania, Anastassia Fedyk, Alex He, and James Hodson. 2024. "Artificial intelligence, firm growth, and product innovation." *Journal of Financial Economics*, 151: 103745.

Baqaee, David, and Emmanuel Farhi. 2019. "The macroeconomic impact of microeconomic shocks: beyond Hulten's theorem." *Econometrica*, 87(4): 1155–1203.

Becker, Kevin, Ege Rush, Elijah Barnes, and David Rein. 2025. "Are AI assistants making experienced developers slower? Evidence from a randomized controlled trial." *Working Paper*.

Bertomeu, Jeremy, Yibin Lin, Yuqi Liu, and Florian Ni. 2025. "The impact of generative AI on information processing: evidence from the ban of ChatGPT in Italy." *Journal of Accounting and Economics*.

Bick, Alexander, Adam Blandin, and David Deming. 2025. "The rapid adoption of generative AI." *NBER Working Paper*.

Briggs, Joseph, and Devesh Kodnani. 2023. "The potentially large effects of artificial intelligence on economic growth." *Working Paper*.

Broda, Christian, and David Weinstein. 2006. "Globalization and the gains from variety." *Quarterly Journal of Economics*, 121(2): 541–585.

Brynjolfsson, Erik, Danielle Li, and Lindsey Raymond. 2024. "Generative AI at work." *Quarterly Journal of Economics*, 139(4): 1721–1784.

Cazzaniga, Mauro, Florence Jaumotte, Longji Li, Giovanni Melina, Augustus Panton, Carlo Pizzinelli, Emma Rockall, and Marina Tavares. 2024. "Gen-AI: artificial intelligence and the future of work." *IMF Staff Discussion Note*.

Center for Research in Security Prices. 2025. "CRSP US Stock Databases Guide." Documentation.

Ciccone, Antonio, and Giovanni Peri. 2005. "Long-run substitutability between more and less educated workers: evidence from U.S. states, 1950–1990." *Review of Economics and Statistics*, 87(4): 652–663.

Cui, Zheyuan, Mert Demirer, Sonia Jaffe, Leon Musolff, Sida Peng, and Tobias Salz. 2025. "The effects of generative AI on high-skilled work: evidence from three field experiments with software developers." *Management Science*.

Dell'Acqua, Fabrizio, Edward McFowland, Ethan Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Krayer, François Candelon, and Karim Lakhani. 2023. "Navigating the jagged technological frontier: field experimental evidence of the effects of AI on knowledge worker productivity and quality." *Working Paper*.

Eisfeldt, Andrea, and Gregor Schubert. 2024. "AI and finance." *NBER Working Paper*.

Eisfeldt, Andrea, Gregor Schubert, and Miao Zhang. 2023. "Generative AI and firm values." *NBER Working Paper*.

Eloundou, Tyna, Sam Manning, Pamela Mishkin, and Daniel Rock. 2024. "GPTs are GPTs: an early look at the labor market impact potential of large language models." *Science*, 384(6702): eadj0998.

Fama, Eugene, and Kenneth French. 1993. "Common risk factors in the returns on stocks and bonds." *Journal of Financial Economics*, 33(1): 3–56.

Fama, Eugene, and Kenneth French. 2015. "A five-factor asset pricing model." *Journal of Financial Economics*, 116(1): 1–22.

Fama, Eugene F., and James D. MacBeth. 1973. "Risk, Return, and Equilibrium: Empirical Tests." *Journal of Political Economy*, 81(3): 607–636.

Felten, Edward, Manav Raj, and Robert Seamans. 2021. "Occupational, industry, and geographic exposure to artificial intelligence: a novel dataset and its potential uses." *Strategic Management Journal*, 42(12): 2195–2217.

Filippucci, Francesco, Peter Gal, and Matthias Schief. 2024. "Miracle or myth? Assessing the macroeconomic productivity gains from artificial intelligence." *OECD Artificial Intelligence Papers*.

Freund, Lukas, and Lukas Mann. 2025. "Job transformation, specialization, and the labor market effects of AI." *Working Paper*.

Gambacorta, Leonardo, Han Qiu, Yi Shan, and Daniel Rees. 2024. "The impact of artificial intelligence on productivity, distribution and power: evidence from a field experiment." *Working Paper*.

Hampole, Manish, Dimitris Papanikolaou, Lawrence Schmidt, and Bryan Seegmiller. 2025. "Artificial intelligence and the labor market." *NBER Working Paper*.

Havránek, Tomáš, Zuzana Iršová, Lubica Laslopová, and Olesia Zeynalova. 2024. "Publication and attenuation biases in measuring skill substitution." *Review of Economics and Statistics*, 106(5): 1187–1200.

Hulten, Charles. 1978. "Growth accounting with intermediate inputs." *Review of Economic Studies*, 45(3): 511–518.

Jones, Charles, and Christopher Tonetti. 2026. "The allocation of resources to generative AI." *Working Paper*.

Katz, Lawrence, and Kevin Murphy. 1992. "Changes in relative wages, 1963–1987: supply and demand factors." *Quarterly Journal of Economics*, 107(1): 35–78.

Kogan, Leonid, Dimitris Papanikolaou, Amit Seru, and Noah Stoffman. 2017. "Technological innovation, resource allocation, and growth." *Quarterly Journal of Economics*, 132(2): 665–712.

Korinek, Anton, and Donghyun Suh. 2024. "Scenarios for the transition to AGI." *NBER Working Paper*.

Krusell, Per, Lee Ohanian, José-Víctor Ríos-Rull, and Giovanni Violante. 2000. "Capital-skill complementarity and inequality: a macroeconomic analysis." *Econometrica*, 68(5): 1029–1053.

Long, John, and Charles Plosser. 1983. "Real business cycles." *Journal of Political Economy*, 91(1): 39–69.

Miranda-Pinto, Jorge, and Eric Young. 2022. "Flexibility and frictions in multisector models." *American Economic Journal: Macroeconomics*, 14(3): 450–480.

Nordhaus, William. 2021. "Are we approaching an economic singularity? Information technology and the future of economic growth." *American Economic Journal: Macroeconomics*, 13(1): 299–332.

Noy, Shakked, and Whitney Zhang. 2023. "Experimental evidence on the productivity effects of generative artificial intelligence." *Science*, 381(6654): 187–192.

Oberfield, Ezra, and Devesh Raval. 2021. "Micro data and macro technology." *Econometrica*, 89(2): 703–732.

Oster, Emily. 2019. "Unobservable selection and coefficient stability: Theory and evidence." *Journal of Business and Economic Statistics*, 37(2): 187–204.

Paradis, Ziv, Albert Zhu, Jessica Klukas, Ambar Mushtaq, Amelia Wattenberger, Kevin Yang, Yun-Nung Zhang, and Elie Bursztein. 2024. "Large language models for code generation in enterprise settings: a controlled experiment." *Working Paper*.

Peng, Sida, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. "The impact of AI on developer productivity: evidence from GitHub Copilot." *Working Paper*.

Peter, Alessandra, and Cian Ruane. 2023. "The aggregate importance of intermediate input substitutability." *NBER Working Paper*.

Qian, Cuili, Mike Peng, and Siyu Li. 2025. "The impact of generative AI announcements on suppliers: evidence from the stock market." *Production and Operations Management*.

Revelio Labs. 2026. "Revelio Labs Data Dictionary." Documentation: https://www.data-dictionary.reveliolabs.com/.

S&P Global Market Intelligence. 2025. "Compustat North America: Data Guide." Documentation.

Trammell, Philip. 2026. "Workflows and automation." *Working Paper*.

Trammell, Philip, and Anton Korinek. 2025. "Economic growth under transformative AI." *Annual Review of Economics*.

Webb, Michael. 2020. "The impact of artificial intelligence on the labor market." *Working Paper*.

Supplemental Appendix

A Proofs

A.1 Proof of Lemma 1

Proof. The proof has three steps: relative profits given factor prices, the general-equilibrium factor-price response to the software engineering shock, and valuation.

Step 1: relative profits. Since firms charge a constant markup, profits are a constant fraction of revenue. Under CES demand and the numeraire $P_t c_t = 1$,

$$\pi_{jt} = \frac{1}{\varepsilon} p_{jt} y_{jt} = \frac{1}{\varepsilon} p_{jt}^{1-\varepsilon} P_t^{\varepsilon-1},$$

so

$$d\log \pi_{jt} = (1-\varepsilon)\left(d\log p_{jt} - d\log P_t\right) = (1-\varepsilon)\left(d\log mc_{jt} - d\log P_t\right).$$

By Shephard's lemma applied to the CES unit-cost function,

$$d\log mc_{jt} = -d\log A_t + \gamma_j\left(d\log w_t - d\log S_t\right) + \left(1-\gamma_j\right) d\log W_t.$$

Differentiating the CES price index and using the fact that, without intermediate inputs, the expenditure share equals the Domar weight $\lambda_j = p_j y_j$, gives

$$d\log P_t = \sum_j \lambda_j d\log p_{jt} = -d\log A_t + \bar{\gamma}\left(d\log w_t - d\log S_t\right) + \left(1-\bar{\gamma}\right) d\log W_t.$$

Subtracting the aggregate price response from the firm-level response yields

$$d\log \pi_{jt} = (\varepsilon-1)\left(\gamma_j - \bar{\gamma}\right)\left(d\log S_t - \left(d\log w_t - d\log W_t\right)\right). \tag{A.1}$$

Step 2: factor prices. Define

$$x_t \equiv d\log S_t - \left(d\log w_t - d\log W_t\right),$$

the change in the effective software engineering cost advantage relative to non-SWE labor. We derive x_t from labor-market clearing.

Because there are no intermediate inputs, total revenue equals final expenditure and $\sum_j \lambda_j = 1$. Since markups are constant, total cost is the constant fraction $\frac{\varepsilon-1}{\varepsilon}$ of revenue. Hence

$$w_t s = \left(\frac{\varepsilon-1}{\varepsilon}\right)\sum_j \lambda_{jt}\gamma_{jt} \qquad \text{and} \qquad W_t L = \left(\frac{\varepsilon-1}{\varepsilon}\right)\sum_j \lambda_{jt}\left(1-\gamma_{jt}\right).$$

Using fixed factor supplies and writing $\bar{\gamma}_t = \sum_j \lambda_{jt}\gamma_{jt}$, we obtain

$$d\log w_t = \frac{d\bar{\gamma}_t}{\bar{\gamma}} \qquad \text{and} \qquad d\log W_t = -\frac{d\bar{\gamma}_t}{1-\bar{\gamma}}.$$

It remains to compute $d\bar{\gamma}_t$. First, CES demand implies

$$d\log\lambda_{jt} = (1-\varepsilon)\left(d\log p_{jt} - d\log P_t\right) = (\varepsilon-1)\left(\gamma_j - \bar{\gamma}\right)x_t,$$

where the second equality uses (A.1). Second, the CES cost-share formula implies

$$\log\left(\frac{\gamma_j}{1-\gamma_j}\right) = \log\left(\frac{\phi_j}{1-\phi_j}\right) + (1-\sigma)\left(\log w_t - \log S_t - \log W_t\right),$$

so

$$d\gamma_{jt} = (\sigma-1)\gamma_j\left(1-\gamma_j\right)x_t.$$

Therefore, using $d\lambda_{jt} = \lambda_j d\log\lambda_{jt}$ to convert the log differential to the absolute differential,

$$d\bar{\gamma}_t = \sum_j \gamma_j d\lambda_{jt} + \sum_j \lambda_j d\gamma_{jt} = (\varepsilon-1)\sum_j \lambda_j\gamma_j\left(\gamma_j-\bar{\gamma}\right)x_t + (\sigma-1)\sum_j \lambda_j\gamma_j\left(1-\gamma_j\right)x_t.$$

Using

$$\sum_j \lambda_j\gamma_j\left(\gamma_j-\bar{\gamma}\right) = \sigma_s^2 \qquad \text{and} \qquad \sum_j \lambda_j\gamma_j\left(1-\gamma_j\right) = \bar{\gamma}\left(1-\bar{\gamma}\right) - \sigma_s^2,$$

we obtain

$$d\bar{\gamma}_t = \left[(\sigma-1)\bar{\gamma}\left(1-\bar{\gamma}\right) + (\varepsilon-\sigma)\sigma_s^2\right]x_t.$$

Substituting into the two labor-market-clearing equations gives

$$d\log w_t - d\log W_t = \frac{d\bar{\gamma}_t}{\bar{\gamma}\left(1-\bar{\gamma}\right)} = \left[(\sigma-1) + (\varepsilon-\sigma)\frac{\sigma_s^2}{\bar{\gamma}\left(1-\bar{\gamma}\right)}\right]x_t.$$

Using the definition of x_t,

$$x_t = d\log S_t - \left(d\log w_t - d\log W_t\right) = d\log S_t - \left[(\sigma-1) + (\varepsilon-\sigma)\frac{\sigma_s^2}{\bar{\gamma}\left(1-\bar{\gamma}\right)}\right]x_t.$$

Hence

$$x_t = \frac{1}{\Sigma}d\log S_t, \qquad \text{where} \qquad \Sigma = \sigma + (\varepsilon-\sigma)\frac{\sigma_s^2}{\bar{\gamma}\left(1-\bar{\gamma}\right)}.$$

Substituting back into (A.1) yields the period-by-period relative-profit response

$$d\log\pi_{jt} = \delta^S\left(\gamma_j - \bar{\gamma}\right)d\log S_t, \qquad \text{where} \qquad \delta^S \equiv \frac{\varepsilon-1}{\Sigma}. \tag{A.2}$$

Step 3: valuation and the NPV bridge. The same linearization applies at every future horizon. Therefore, for each $k \geq 0$,

$$d\log\pi_{j,t+k} = \delta^S\left(\gamma_j - \bar{\gamma}\right)d\log S_{t+k}.$$

Note that this expression has no firm-invariant component: as shown in Step 1, the Hicks-neutral shock A_t cancels from relative unit costs entirely, and the discount-rate shock m_t does not enter profits directly. Log-linearizing firm value to first order gives

$$d\log V_{jt} = (1-\beta)\,\mathbb{E}_t\left[\sum_{k=0}^{\infty}\beta^k\left(d\log\pi_{j,t+k} + d\log M_{t,t+k}\right)\right].$$

Substituting the expression above, recognizing $\mathscr{S}_t = (1-\beta)\mathbb{E}_t[\sum_{k=0}^{\infty}\beta^k d\log S_{t+k}]$, and combining the firm-invariant terms yields

$$d\log V_{jt} = \delta^S\gamma_j\mathscr{S}_t + \mu_t, \tag{A.3}$$

where

$$\mu_t \equiv (1-\beta)\,\mathbb{E}_t\left[\sum_{k=0}^{\infty}\beta^k d\log M_{t,t+k}\right] - \delta^S\bar{\gamma}\mathscr{S}_t$$

collects all firm-invariant terms: the present-value nominal stochastic discount factor effect and the cross-sectional mean profit response. This is (6). □

A.2 Proof of Proposition 1

Proof. From the valuation step in the proof of Lemma 1, firm value admits the representation

$$d\log V_{jt} = \delta^S\gamma_j\mathscr{S}_t + \mu_t,$$

where μ_t is common across firms. Projecting $d\log V_{jt}$ on the residualized AI-index return gives

$$\beta_j = \frac{\mathrm{Cov}_t\left(d\log V_{jt}, \tilde{R}_t^{\mathrm{AI}}\right)}{\mathrm{Var}_t\left(\tilde{R}_t^{\mathrm{AI}}\right)} = \beta_0 + \delta^S\kappa\gamma_j,$$

where

$$\beta_0 \equiv \frac{\mathrm{Cov}_t\left(\mu_t, \tilde{R}_t^{\mathrm{AI}}\right)}{\mathrm{Var}_t\left(\tilde{R}_t^{\mathrm{AI}}\right)}$$

is firm-invariant and

$$\kappa \equiv \frac{\mathrm{Cov}_t\left(\mathscr{S}_t, \tilde{R}_t^{\mathrm{AI}}\right)}{\mathrm{Var}_t\left(\tilde{R}_t^{\mathrm{AI}}\right)}$$

is the population linear-projection coefficient of $\mathscr{S}_t$ on $\tilde{R}_t^{\mathrm{AI}}$. Taking the cross-sectional slope on γ_j,

$$\delta = \frac{\mathrm{Cov}_j\left(\beta_j, \gamma_j\right)}{\mathrm{Var}_j\left(\gamma_j\right)} = \delta^S\kappa.$$

Rearranging,

$$\kappa = \frac{\delta}{\delta^S} = \frac{\Sigma}{\varepsilon - 1}\cdot\delta.$$

Because the proposition is stated in terms of the best linear predictor, the projection formula is exactly

$$\mathbb{E}\left[\mathscr{S}_t \mid \tilde{R}_t^{\mathrm{AI}}\right] = \kappa\cdot\tilde{R}_t^{\mathrm{AI}},$$

which is (7). □

A.3 Proof of Proposition 2

Proof. We prove the annualized cumulative statement first and then derive the flow special case under a permanent software engineering productivity shock.

Annualized NPV of GDP growth. At each horizon k, a pure software engineering productivity shock affects firm j's unit cost as

$$\frac{\partial \log mc_j}{\partial \log S} = -\gamma_j.$$

Thus, to first order, $d\log S_{t+k}$ is equivalent to a firm-specific Hicks-neutral TFP shock of size $\gamma_j d\log S_{t+k}$. Applying Hulten's theorem period by period,

$$d\log Y_{t+k}\big|_S = \sum_j \lambda_j \gamma_j d\log S_{t+k} = \bar{\gamma} d\log S_{t+k}.$$

Multiplying by $(1-\beta)\beta^k$, summing over k, and taking $\mathbb{E}_t$ gives

$$\mathscr{Y}_t = (1-\beta)\mathbb{E}_t\left[\sum_{k=0}^{\infty} \beta^k d\log Y_{t+k}\big|_S\right] = \bar{\gamma}\mathscr{S}_t,$$

which is (9).

Conditional prediction. Using Proposition 1,

$$\mathbb{E}\left[\mathscr{S}_t \mid \tilde{R}_t^{\mathrm{AI}}\right] = \kappa \cdot \tilde{R}_t^{\mathrm{AI}}.$$

Multiplying by $\bar{\gamma}$,

$$\mathbb{E}\left[\mathscr{Y}_t \mid \tilde{R}_t^{\mathrm{AI}}\right] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\mathrm{AI}} = \frac{\bar{\gamma}\Sigma}{\varepsilon - 1} \cdot \delta \cdot \tilde{R}_t^{\mathrm{AI}},$$

which is (10).

Flow special case under a permanent shock. If the software engineering productivity shock is permanent ($\mathbb{E}_t[d\log S_{t+k}] = d\log S_t$ for all $k \geq 0$), then $\mathscr{S}_t = d\log S_t$ and $\mathscr{Y}_t = d\log Y_t|_S$. The period-by-period Hulten argument gives

$$d\log Y_t\big|_S = \bar{\gamma} d\log S_t = \mathscr{Y}_t,$$

which is (11). Combining this with (8) yields

$$\mathbb{E}\left[\mathscr{Y}_t \mid \tilde{R}_t^{\mathrm{AI}}\right] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\mathrm{AI}} = \frac{\bar{\gamma}\Sigma}{\varepsilon - 1} \cdot \delta \cdot \tilde{R}_t^{\mathrm{AI}},$$

which is (12). □

A.4 Proof of Lemma 2 Part (i)

For the proofs, we introduce notation $\tilde{\gamma}_{jt} = \gamma_{jt}/\alpha_{jt}$ for the within-value-added software engineering cost share. Since $\pi_{jt} = \frac{1}{\varepsilon} p_{jt} y_{jt}$ with constant ε, we have $d\log \pi_{jt} = d\log\left(p_{jt} y_{jt}\right)$. We derive the profit response in three stages: prices given factor prices, factor prices in general equilibrium, and assembly.

Step 1: prices. With constant markups, $d\log p_{jt} = d\log mc_{jt}$. By Shephard's lemma applied to the nested cost function dual to (13)–(14):

$$d\log mc_{jt} = -d\log A_t + \gamma_j d\log \tilde{w}_t + \theta_j d\log W_t + \sum_k \Omega_{jk} d\log p_{kt} + e_j d\log \xi_{jt},$$

where $\tilde{w}_t = w_t/S_t$ is the effective software engineering wage and $e_j = \partial \log mc_j / \partial \log \xi_j$ is the idiosyncratic cost-function elasticity. The cost-share coefficients follow from the chain rule applied to the nested structure: the labor aggregate H_j has unit cost $h_j^*(\tilde{w}_t, W_t)$ with shares $\tilde{\gamma}_j$ on effective SWE and $1-\tilde{\gamma}_j$ on non-SWE, so that $\gamma_j = \alpha_j \tilde{\gamma}_j$ and $\theta_j = \alpha_j\left(1-\tilde{\gamma}_j\right)$ after composing with the outer cost function.

Since $d\log p_{jt} = d\log mc_{jt}$, this is a linear system in the price vector with input — output feedback through $\sum_k \Omega_{jk} d\log p_{kt}$. Applying the Leontief inverse $\Lambda = (I-\Omega)^{-1}$ element by element:

$$d\log p_{jt} = \sum_k \Lambda_{jk}\left[\gamma_k d\log \tilde{w}_t + \theta_k d\log W_t - d\log A_t + e_k d\log \xi_{kt}\right].$$

Define firm j's *total factor contents*:

$$\bar{\gamma}_j \equiv \sum_k \Lambda_{jk}\gamma_k \qquad \text{and} \qquad \bar{\theta}_j \equiv \sum_k \Lambda_{jk}\theta_k.$$

These measure firm j's direct and indirect (upstream) exposure to each labor type, weighted by IO linkages. Note that $\bar{\gamma}_j + \bar{\theta}_j = \sum_k \Lambda_{jk}\alpha_k = 1$ by the identity $\Lambda\alpha = \mathbf{1}$: firm j's price is fully accounted for by upstream labor costs after tracing through the IO chain. The price response is

$$d\log p_{jt} = \bar{\gamma}_j d\log \tilde{w}_t + \left(1-\bar{\gamma}_j\right) d\log W_t - \left(\sum_k \Lambda_{jk}\right) d\log A_t + \sum_k \Lambda_{jk} e_k d\log \xi_{kt}.$$

The aggregate price index satisfies $d\log P_t = \sum_j f_j d\log p_{jt} + \frac{1}{1-\varepsilon}\sum_j f_j d\log \omega_j(Z_t)$, where $f_j = p_j c_j$ is the final expenditure share. Defining the cost-based centralities $v_k = \sum_j f_j \Lambda_{jk}$:

$$d\log P_t = \bar{\gamma}^c d\log \tilde{w}_t + \left(1-\bar{\gamma}^c\right) d\log W_t - \left(\sum_k v_k\right) d\log A_t + \sum_k v_k e_k d\log \xi_{kt} + \frac{1}{1-\varepsilon}\sum_j f_j d\log \omega_j(Z_t),$$

where $\bar{\gamma}^c = \sum_k v_k \gamma_k$ is the cost-based Domar-weighted aggregate software engineering share. In the presence of constant markups ($\mu > 1$), the cost-based centralities v_k differ from the revenue-based Domar weights λ_k; with perfect competition ($\mu = 1$), $v_k = \lambda_k$ and $\bar{\gamma}^c = \bar{\gamma}$.

Step 2: firm revenue. Firm j's revenue $R_{jt} = p_{jt}y_{jt}$ is final demand plus intermediate demand,

$$\begin{aligned} R_{jt} &= p_{jt}c_{jt} + \sum_k p_{jt}x_{kj,t} \\ &= f_{jt} + \frac{1}{\mu}\sum_k \Omega_{kj,t}R_{kt}, \end{aligned}$$

where $f_{jt} = p_{jt}c_{jt} = \omega_j(Z_t)\, p_{jt}^{1-\varepsilon} P_t^{\varepsilon-1}$ is final consumer expenditure on good j, and $\Omega_{kj,t}R_{kt}/\mu = \Omega_{kj,t}\mathscr{C}_{kt}$ is firm k's intermediate-input spending on good j (by Shephard's lemma and constant markups). The second term is the source of the demand feedback: when firm k's revenue rises, its total cost R_{kt}/μ rises proportionally, and a share Ω_{kj} of that increased spending falls on firm j.

Define $\alpha_j^f = f_j/R_j$ (the share of firm j's steady-state revenue from final demand) and $\mathscr{S}_{jk} = \Omega_{kj}R_k/\left(\mu R_j\right)$ (the share from firm k's intermediate demand). These exhaust revenue: $\alpha_j^f + \sum_k \mathscr{S}_{jk} = 1$. Log-linearizing yields

$$d\log R_{jt} = \alpha_j^f\, d\log f_{jt} + \sum_k \mathscr{S}_{jk}\left(d\log R_{kt} + d\log\Omega_{kj,t}\right).$$

The three pieces on the right-hand side have distinct economic content.

First, we have final demand $d\log f_{jt} = d\log\omega_j + (1-\varepsilon)\left(d\log p_{jt} - d\log P_t\right)$, from CES demand under the numeraire $P_t c_t = 1$. When firm j's price falls relative to the aggregate, consumers substitute toward good j, raising j's final-demand revenue. The strength of this reallocation is governed by the demand elasticity ε.

Second, we have intermediate-demand revenue feedback. The $d\log R_{kt}$ terms capture the demand multiplier through the IO network. When firm k's revenue rises, k's spending on all inputs rises proportionally (at rate $1/\mu$ per unit of revenue), which raises the revenue of k's suppliers, which raises their input spending, and so on.

Third, we have input substitution. The $d\log\Omega_{kj,t}$ terms capture how firm k's input expenditure shares shift in response to relative price changes. Because the outer cost function is constant returns, twice differentiable, and depends on input prices only through relative prices, the cost-share responses are linear in local price changes. Hence there exist steady-state coefficients η_{kj}^h and $\eta_{kj\ell}$, determined by the cost shares and the Allen-Uzawa elasticities of firm k, such that

$$d\log\Omega_{kj,t} = \eta_{kj}^h\, d\log h_k^* + \sum_\ell \eta_{kj\ell}\, d\log p_{\ell t}, \qquad \eta_{kj}^h + \sum_\ell \eta_{kj\ell} = 0,$$

where

$$d\log h_k^* = \tilde{\gamma}_k\, d\log\tilde{w}_t + \left(1-\tilde{\gamma}_k\right) d\log W_t$$

is the log change in the unit cost of firm k's labor aggregate (by Shephard's lemma on H_k). The zero-sum property $\eta_{kj}^h + \sum_\ell \eta_{kj\ell} = 0$ ensures that a uniform scaling of all input prices leaves expenditure shares unchanged, so that the Cobb-Douglas special case ($\sigma = 1$ for all substitution elasticities) cor-

rectly yields constant shares. Two forces are at work. First, when the labor aggregate becomes cheaper relative to intermediates (because S_t rises, lowering $\tilde{w}_t$), firm k substitutes toward labor and away from intermediates. Second, when some intermediates become relatively cheaper than others (because their upstream software engineering content $\bar{\gamma}_\ell$ differs), firm k shifts its input mix. Both effects are linear in factor prices and goods prices, hence linear in factor prices and shocks via Step 1.

The $d\log R_{kt}$ terms make this a linear fixed point in the revenue vector, exactly parallel to the price system in Step 1. There, Ω_{jk} created upstream cost feedback and we inverted $(I-\Omega)$ to get Λ_{jk}. Here, $\mathscr{S}_{jk}$ creates downstream demand feedback. Define $\Psi = (I-\mathscr{S})^{-1}$, the demand-side Leontief inverse: Ψ_{jk} measures the total effect on firm j's revenue from a unit demand impulse at firm k, after tracing through all rounds of intermediate-demand propagation. Rearranging and inverting:

$$\begin{aligned} d\log\pi_{jt} &= d\log R_{jt} \\ &= \sum_k \Psi_{jk}\left[\alpha_k^f d\log\omega_k + \alpha_k^f(1-\varepsilon)\left(d\log p_{kt} - d\log P_t\right) + \sum_m \mathscr{S}_{km} d\log\Omega_{mk,t}\right]. \end{aligned}$$

Every object inside the brackets is known from Step 1 (goods prices as functions of factor prices and shocks), so profits are linear in $\left(d\log\tilde{w}_t, d\log W_t, d\log A_t, d\log\omega_k, d\log\xi_{kt}\right)$.

Step 3: wage responses. *Factor market clearing.* With inelastic supplies s and L, total software engineering compensation satisfies $w_t s = \sum_j \gamma_{jt} R_{jt}/\mu$. Log-linearizing with $d\log s = 0$:

$$d\log w_t = \sum_j \nu_j\left(d\log\gamma_{jt} + d\log R_{jt}\right),$$

where $\nu_j = \gamma_j R_j / \sum_k \gamma_k R_k$ is firm j's share of total steady-state software engineering compensation. The analogous condition for non-SWE labor gives

$$d\log W_t = \sum_j \nu_j^L\left(d\log\theta_{jt} + d\log R_{jt}\right),$$

with $\nu_j^L = \theta_j R_j / \sum_k \theta_k R_k$. Each equation says: since factor supply is fixed, the factor price must adjust until the demand for that factor (reflected in cost shares times revenue) is unchanged.

The term $d\log R_{jt}$ is linear in factor prices and exogenous shocks by Steps 1–2. It remains to determine the cost-share responses $d\log\gamma_{jt}$ and $d\log\theta_{jt}$.

Cost-share responses. Since $\gamma_j = \alpha_j\tilde{\gamma}_j$, we have $d\log\gamma_{jt} = d\log\alpha_{jt} + d\log\tilde{\gamma}_{jt}$. The two pieces reflect substitution at different levels of the production structure.

The within-value-added piece captures substitution between software engineering and non-SWE labor inside H_j. Applying the cost-share elasticity formula to the inner function:

$$d\log\tilde{\gamma}_{jt} = \left(1-\tilde{\gamma}_j\right)\left(1-\sigma_j^H\right)\left(d\log\tilde{w}_t - d\log W_t\right),$$

where σ_j^H is the elasticity of substitution in H_j between software engineering and non-SWE labor. This

elasticity is distinct from the Allen-Uzawa elasticities σ_m^{hk} and $\sigma_m^{k\ell}$ of the outer function that appeared in Step 2: σ_j^H governs substitution *within* the labor aggregate, while σ_m^{hk} governs substitution *between* the labor aggregate and intermediates. When $\sigma_j^H > 1$, a fall in the relative software engineering wage shifts expenditure toward software engineering, raising $\tilde{\gamma}_j$.

The value-added share piece captures substitution between the labor aggregate and intermediates inside F_j. Applying the cost-share elasticity formula to the outer function:

$$d\log\alpha_{jt} = \sum_k \Omega_{jk}\left(\sigma_j^{hk} - 1\right)\left(d\log p_{kt} - \tilde{\gamma}_j\, d\log\tilde{w}_t - \left(1-\tilde{\gamma}_j\right) d\log W_t\right),$$

where the bracketed term is the log change in the price of intermediate k relative to the unit cost of firm j's labor aggregate. When $\sigma_j^{hk} > 1$ and intermediate k becomes relatively cheaper, firm j substitutes toward k and away from labor, lowering α_j.

Combining: $d\log\gamma_{jt} = d\log\alpha_{jt} + d\log\tilde{\gamma}_{jt}$ is linear in factor prices and goods prices. Since goods prices are linear in factor prices (Step 1), $d\log\gamma_{jt}$ is linear in $\left(d\log\tilde{w}_t, d\log W_t\right)$ and the exogenous shocks. The response $d\log\theta_{jt}$ is constructed analogously, with $d\log\tilde{\theta}_{jt}$ capturing the internal substitution:

$$d\log\tilde{\theta}_{jt} \equiv d\log\left(1-\tilde{\gamma}_{jt}\right) = \tilde{\gamma}_j\left(\sigma_j^H - 1\right)\left(d\log\tilde{w}_t - d\log W_t\right),$$

and the same $d\log\alpha_{jt}$ in the outer piece (since $\theta_j = \alpha_j\left(1-\tilde{\gamma}_j\right)$, the value-added share response is common to both).

The 2×2 *system.* Substituting the cost-share expressions into the factor-market-clearing conditions, using $d\log\tilde{w}_t = d\log w_t - d\log S_t$:

$$\begin{aligned}
d\log w_t = \sum_j \nu_j \Big[& \left(1-\tilde{\gamma}_j\right)\left(1-\sigma_j^H\right)\left(d\log\tilde{w}_t - d\log W_t\right) \\
& + \sum_k \Omega_{jk}\left(\sigma_j^{hk}-1\right)\left(d\log p_{kt} - \tilde{\gamma}_j\, d\log\tilde{w}_t - \left(1-\tilde{\gamma}_j\right) d\log W_t\right) \\
& + d\log R_{jt}\Big], \\
d\log W_t = \sum_j \nu_j^L \Big[& \tilde{\gamma}_j\left(\sigma_j^H - 1\right)\left(d\log\tilde{w}_t - d\log W_t\right) \\
& + \sum_k \Omega_{jk}\left(\sigma_j^{hk}-1\right)\left(d\log p_{kt} - \tilde{\gamma}_j\, d\log\tilde{w}_t - \left(1-\tilde{\gamma}_j\right) d\log W_t\right) \\
& + d\log R_{jt}\Big],
\end{aligned}$$

where $d\log p_{kt}$ and $d\log R_{jt}$ are linear functions of $\left(d\log\tilde{w}_t, d\log W_t\right)$ and the exogenous shocks, as determined in Steps 1–2. The first line of each equation is inner substitution (within the labor aggregate), the second is outer substitution (between the labor aggregate and intermediates), and the third is the revenue channel. Together, these two equations form a 2×2 linear system in $\left(d\log\tilde{w}_t, d\log W_t\right)$

with forcing from $\left(d\log S_t, d\log A_t, d\log\omega_k, d\log\xi_{kt}\right)$.

When S_t is the only active shock, $d\log\tilde{w}_t = d\log w_t - d\log S_t$ introduces a mechanical $-d\log S_t$ into the effective software engineering wage, which feeds into both wage equations directly through the $d\log\tilde{w}_t$ substitution terms. Assuming this 2×2 system is nonsingular, solving (by Cramer's rule or substitution) yields factor prices as linear functions of the shock.

Step 4: assembly. The 2×2 system from Step 3 determines factor prices as linear functions of the exogenous shocks. Write the solution as

$$d\log\tilde{w}_t = \gamma_1^S d\log S_t + \sum_n \gamma_{1n}^Z d\log Z_{nt} + \sum_k \gamma_{1k}^\xi d\log\xi_{kt}$$

and analogously for $d\log W_t$ with coefficients γ_2^S, γ_{2n}^Z, γ_{2k}^ξ. Each γ coefficient is a known function of the steady-state objects enumerated in Steps 1–3: cost shares $\left(\gamma_j, \theta_j, \Omega_{jk}\right)$, the supply-side Leontief inverse Λ_{jk}, the demand-side Leontief inverse Ψ_{jk}, final-demand shares α_j^f, factor-compensation weights $\left(\nu_j, \nu_j^L\right)$, the demand elasticity ε, the inner substitution elasticities σ_j^H, and the outer Allen-Uzawa elasticities $\left(\sigma_j^{hk}, \sigma_j^{k\ell}\right)$.

Substituting the factor-price solution into Step 1 gives goods prices as linear functions of the shocks. Substituting goods prices into Step 2 gives profits as linear functions of the shocks. The result is the linear decomposition stated in the lemma:

$$d\log\pi_{jt} = b_j d\log S_t + \sum_n b_{jn}^Z d\log Z_{nt} + \sum_k b_{jk}^\xi d\log\xi_{kt}.$$

Each coefficient is obtained by tracing the corresponding shock through the full chain: exogenous shock → factor prices (Step 3) → goods prices (Step 1) → revenue and profits (Step 2), plus any direct effects that bypass factor prices (the Hicks-neutral cost reduction $-d\log A_t$, the demand-weight shifts $d\log\omega_j$, and the idiosyncratic cost shocks $e_j d\log\xi_{jt}$).

To make the dependence on observables explicit, consider the AI shock $d\log S_t$ in isolation. From Step 2, firm j's profit response given factor prices is

$$d\log\pi_{jt} = \sum_k \Psi_{jk}\Big[\alpha_k^f(1-\varepsilon)\left(\bar{\gamma}_k - \bar{\gamma}^c\right)\left(d\log\tilde{w}_t - d\log W_t\right) + \sum_m \mathscr{S}_{km} d\log\Omega_{mk,t}\Big],$$

where we have substituted $d\log p_{kt} - d\log P_t = \left(\bar{\gamma}_k - \bar{\gamma}^c\right)\left(d\log\tilde{w}_t - d\log W_t\right) + \ldots$ from Step 1, using $\bar{\gamma}_k + \bar{\theta}_k = 1$ and $\bar{\gamma}^c + \bar{\theta}^c = 1$ so that the two factor-price terms collapse into the single relative wage $d\log\tilde{w}_t - d\log W_t$. The input-substitution terms $d\log\Omega_{mk,t}$ are also proportional to factor prices (Step 2). Substituting the factor-price solution $d\log\tilde{w}_t = \gamma_1^S d\log S_t$, $d\log W_t = \gamma_2^S d\log S_t$ gives b_j in closed form. The cross-sectional variation in b_j is driven by the total factor contents $\bar{\gamma}_k$ (entering through goods prices) and the demand-network weights Ψ_{jk} (entering through revenue propagation): firms

with high upstream software engineering content, or whose downstream customers have high software engineering content, benefit most from the AI shock.

The elasticities b^Z_{jn} and b^ξ_{jk} are constructed by the same substitution with the factor-price response to $d\log Z_{nt}$ or $d\log\xi_{kt}$, plus the direct effects of $d\log A_t$, $d\log\omega_j$, and $d\log\xi_{jt}$ that enter Step 2 without passing through factor prices.

A.5 Proof of Lemma 2 Part (ii)

Proof. At steady state, $d\log V_{jt} = (1-\beta)\,\mathbb{E}_t\left[\sum_{\tau=0}^{\infty}\beta^\tau\left(d\log\pi_{j,t+\tau} + d\log M_{t,t+\tau}\right)\right]$. Since the profit elasticities b_j, b^Z_{jn}, and b^ξ_{jk} are partial derivatives evaluated at steady state, the same linearization applies at every future horizon τ:

$$d\log\pi_{j,t+\tau} = b_j\, d\log S_{t+\tau} + \sum_n b^Z_{jn}\, d\log Z_{n,t+\tau} + \sum_k b^\xi_{jk}\, d\log\xi_{k,t+\tau}.$$

Substituting into the firm-value expression and factoring out the constant elasticities,

$$d\log V_{jt} = b_j \underbrace{(1-\beta)\,\mathbb{E}_t\Big[\sum_{\tau=0}^{\infty}\beta^\tau\, d\log S_{t+\tau}\Big]}_{\mathscr{S}_t} + \sum_n b^Z_{jn}\underbrace{(1-\beta)\,\mathbb{E}_t\Big[\sum_{\tau=0}^{\infty}\beta^\tau\, d\log Z_{n,t+\tau}\Big]}_{\mathscr{Z}_{nt}} + \sum_k b^\xi_{jk}\underbrace{(1-\beta)\,\mathbb{E}_t\Big[\sum_{\tau=0}^{\infty}\beta^\tau\, d\log\xi_{k,t+\tau}\Big]}_{\mathscr{E}_{kt}}$$

where $\mathscr{E}_{kt}$ is the annualized NPV of the idiosyncratic shock to firm k. The stochastic discount factor component μ_t depends on aggregate shocks but not on firm identity, hence it is symmetric across firms. Note that while the idiosyncratic terms $\sum_k b^\xi_{jk}\mathscr{E}_{kt}$ remain in the individual firm valuation (acting as the unobserved firm-level error u_j in the cross-sectional regression), they are diversified away in aggregate portfolios by the granularity assumption. □

A.6 Proof of Proposition 3

Because the profit response is first-order, the coefficients b_j and b^Z_{jn} are partial derivatives evaluated at steady state:

$$b_j = \left.\frac{\partial\log\pi_j}{\partial\log S}\right|_{\text{ss}}, \qquad b^Z_{jn} = \left.\frac{\partial\log\pi_j}{\partial\log Z_n}\right|_{\text{ss}}.$$

Each coefficient is computed holding all other shocks at zero. The profit response to a joint perturbation is then the linear combination $b_j d\log S_t + \sum_n b^Z_{jn} d\log Z_{nt} + \ldots$ by linearity. This means we can characterize b^Z_{jn} by studying a pure $d\log Z_{nt}$ perturbation with $d\log S_t = 0$ and $d\log\xi_{kt} = 0$ for all k. The equal-factor-price result does not need to hold for arbitrary joint perturbations — it only needs to hold for the Z_n-only perturbation, because that is the perturbation that pins the coefficient b^Z_{jn}.

We proceed by tracing the Z_n shock through each stage of the equilibrium characterization in Appendix A.4, asking at each stage: where does $\tilde{\gamma}_j$ appear, and where does it drop out?

The shock Z_n affects the economy through two channels: the Hicks-neutral shifter ($d\log A_t = \bar{z}_n d\log Z_{nt}$, where $\bar{z}_n = \partial \log f / \partial \log Z_n$) and the demand weights ($d\log \omega_j = \omega'_{jn} d\log Z_{nt}$). We treat each in turn.

The Hicks-neutral channel. *Equal factor prices.* We claim that, under a pure Z_n perturbation with $d\log S_t = 0$, the equilibrium satisfies $d\log w_t = d\log W_t$. Conjecture this and write the common value as $c \cdot d\log Z_{nt}$. Since $d\log S_t = 0$, the effective software engineering wage satisfies $d\log \tilde{w}_t = d\log w_t = d\log W_t = c \cdot d\log Z_{nt}$, so the effective wage ratio $\tilde{w}_t / W_t$ is unchanged. By the structure of H_j, the within-value-added software engineering share $\tilde{\gamma}_j$ depends only on this ratio, so

$$d\log \tilde{\gamma}_{jt} = 0 \qquad \text{for all } j.$$

We now verify the conjecture by tracing through the three stages of the equilibrium characterization.

Stage 1: goods prices. The unit cost of firm j's labor aggregate is $h_j^*(\tilde{w}_t, W_t)$, which is homogeneous of degree one in $(\tilde{w}, W)$. Under the conjecture, $d\log h_j^* = \tilde{\gamma}_j \cdot c + (1 - \tilde{\gamma}_j) \cdot c = c$, uniformly across firms. Substituting into the unit cost equation from the proof of Lemma 2:

$$d\log mc_{jt} = -\bar{z}_n d\log Z_{nt} + \alpha_j \cdot c d\log Z_{nt} + \sum_k \Omega_{jk} d\log p_{kt}.$$

The software engineering cost share γ_j and non-SWE cost share θ_j appear only through their sum α_j. The within-value-added software engineering share $\tilde{\gamma}_j$ has dropped out. Solving the IO price system via the Leontief inverse and using $\Lambda\alpha = \mathbf{1}$:

$$d\log p_{jt} = \left[c - \bar{z}_n \sum_k \Lambda_{jk} \right] d\log Z_{nt}.$$

Prices depend on the IO structure through $\sum_k \Lambda_{jk}$ but not on $\tilde{\gamma}_j$.

Stage 2: revenue. Revenue depends on goods prices, the demand network (Ψ_{jk}, α_j^f), and input substitution ($d\log \Omega_{mk,t}$). Since goods prices do not depend on $\tilde{\gamma}_j$ (shown above), the final-demand and intermediate-demand-feedback channels in the revenue equation carry no $\tilde{\gamma}_j$ dependence. For the input-substitution terms, recall from the proof of Lemma 2 that

$$d\log \Omega_{mk,t} = \eta^h_{mk} d\log h^*_m + \sum_\ell \eta_{mk\ell} d\log p_{\ell t},$$

where $d\log h^*_m = \tilde{\gamma}_m d\log \tilde{w}_t + (1 - \tilde{\gamma}_m) d\log W_t$. Under the conjecture, $d\log h^*_m = c \cdot d\log Z_{nt}$ regardless of $\tilde{\gamma}_m$. The remaining terms depend on goods prices, which we showed do not depend on $\tilde{\gamma}_\ell$. Therefore $d\log \Omega_{mk,t}$ does not depend on $\tilde{\gamma}_m$, and hence revenue $d\log R_{jt}$ does not depend on $\tilde{\gamma}_j$.

Stage 3: verification. It remains to verify that both factor-market-clearing conditions yield the same scalar c. From the proof of Lemma 2, the software engineering condition is

$$d\log w_t = \sum_j \nu_j \left(d\log \gamma_{jt} + d\log R_{jt} \right),$$

and the non-SWE condition replaces ν_j with ν_j^L and γ_j with θ_j. Since $\gamma_j = \alpha_j\tilde{\gamma}_j$ and $\theta_j = \alpha_j\left(1-\tilde{\gamma}_j\right)$, we have $d\log\gamma_{jt} = d\log\alpha_{jt} + d\log\tilde{\gamma}_{jt}$ and $d\log\theta_{jt} = d\log\alpha_{jt} + d\log\left(1-\tilde{\gamma}_{jt}\right)$. Under the conjecture, $d\log\tilde{\gamma}_{jt} = 0$ for all j (shown above), so $d\log\gamma_{jt} - d\log\theta_{jt} = 0$: the cost-share responses are symmetric in the two labor types. The value-added share response $d\log\alpha_{jt}$ is common to both and depends on goods prices and outer elasticities (neither of which involves $\tilde{\gamma}_j$). The revenue response $d\log R_{jt}$ is also common. For the two equations to reduce to the same scalar c, we require $\sum_j \nu_j\left(d\log\alpha_{jt} + d\log R_{jt}\right) = \sum_j \nu_j^L\left(d\log\alpha_{jt} + d\log R_{jt}\right)$. This is an unconditional aggregate condition: it requires that the macro shock Z_n does not systematically shift unconditional aggregate expenditure toward software-engineering-intensive sectors relative to non-software-engineering-intensive sectors. Assumption 1 guarantees orthogonality conditional on X_j, which suffices for the demand-weight channel treated below; however, the unconditional balance here is a mild additional requirement, satisfied whenever the production structure is sufficiently symmetric across sectors or, more generally, whenever the equilibrium responses $d\log\alpha_{jt} + d\log R_{jt}$ (which are common within X_j groups) do not systematically covary with the between-group difference in software engineering compensation weights $\nu_j - \nu_j^L$. Under this condition, the system reduces to a single scalar equation, which pins c and confirms the conjecture.

The demand-weight channel. The shock Z_n may also shift expenditure weights $\omega_j(Z_t)$. From the revenue equation in the proof of Lemma 2, the demand-weight shift $d\log\omega_j = \omega'_{jn}d\log Z_{nt}$ enters final demand and propagates through the demand network via Ψ_{jk}. This generates cross-sectional variation in b_{jn}^Z through ω'_{jn} — reflecting which goods consumers shift toward — but this variation depends on ω'_{jn} and the demand network, not on within-firm factor composition $\tilde{\gamma}_j$.

What b_{jn}^Z depends on. Combining both channels, the firm-specific variation in b_{jn}^Z runs through the following objects and no others:

(i) the value-added share α_j and the IO cost shares $\Omega_{j1},\ldots,\Omega_{jJ}$ (which determine $\sum_k \Lambda_{jk}$, Ψ_{jk}, and α_j^f);

(ii) the baseline demand weight ω_j and the demand-weight elasticity ω'_{jn};

(iii) the outer Allen-Uzawa elasticities σ_j^{hk} and $\sigma_j^{k\ell}$.

In particular, b_{jn}^Z does not depend on the within-value-added software engineering share $\tilde{\gamma}_j$ or the inner substitution elasticity σ_j^H. We may therefore write

$$b_{jn}^Z = g_n\left(\alpha_j, \Omega_{j1},\ldots,\Omega_{jJ}, \omega_j, \omega'_{jn}, \sigma_j^{h\cdot}, \sigma_j^{\cdot\cdot}\right)$$

for some function g_n that does not take $\tilde{\gamma}_j$ or σ_j^H as arguments.

The identification result. Since $\gamma_j = \alpha_j \tilde{\gamma}_j$, the cross-sectional variation in γ_j has two sources: variation in α_j (value-added share) and variation in $\tilde{\gamma}_j$ (within-value-added software engineering intensity). By the premise of Proposition 3, conditioning on the controls X_j fixes the observable production structure and baseline demand weights: for any two firms j and k with $X_j = X_k$, we have identical $\left(\alpha_j, \Omega_{j\cdot}, \omega_j, \sigma_j^{h\cdot}, \sigma_j^{\cdot\cdot}\right)$. After conditioning on X_j, the residual variation in γ_j comes entirely from $\tilde{\gamma}_j$. But the supply-side components of g_n have no variation along this dimension. Furthermore, by Assumption 1, the demand-weight elasticity ω'_{jn} is orthogonal to γ_j conditional on X_j. Since λ_j is determined by the observables absorbed in X_j, this ensures the demand-weight shifts do not systematically correlate with γ_j conditional on X_j. Therefore, b_{jn}^Z is uncorrelated with γ_j conditional on X_j:

$$\mathrm{Cov}_j\left(b_{jn}^Z, \gamma_j \mid X_j\right) = 0 \qquad \text{for all } n.$$

Consequently, $\delta_n^Z = \frac{\mathrm{Cov}_j\left(b_{jn}^Z, \gamma_j | X_j\right)}{\mathrm{Var}_j(\gamma_j | X_j)} = 0$ for all n. The master equation (22) reduces to $\delta = \kappa \delta^S$. Rearranging yields Equation (23):

$$\kappa = \frac{\delta}{\delta^S}.$$

This concludes the proof.

A.7 Proof of Proposition 4

Proof. Part (a). Define real GDP as the consumption aggregate $Y_t = c_t$. Under our numeraire ($P_t c_t = 1$), real GDP growth is

$$d\log Y_t = -d\log P_t.$$

We derive $d\log P_t$ in response to $d\log S_t$, holding (Z_t, ξ_t) fixed.

Step 1: the AI shock as an equivalent firm-level TFP shock. By Shephard's lemma applied to the unit cost function of firm j,

$$\frac{\partial \log mc_j}{\partial \log S} = -\gamma_j,$$

since S enters only through the effective software engineering wage w/S, and γ_j is the cost share of software engineering labor. A one percent increase in S therefore reduces firm j's unit cost by γ_j percent, holding factor prices and goods prices fixed. This is equivalent, to first order, to a Hicks-neutral TFP increase of $\gamma_j d\log S_t$ at firm j.

Step 2: Hulten's theorem. By the envelope theorem, the first-order GDP response to firm-level Hicks-neutral TFP shocks $\{d\log A_j^{\mathrm{TFP}}\}_{j=1}^J$ is

$$d\log Y_t = \sum_j \lambda_j d\log A_j^{\mathrm{TFP}}, \tag{A.4}$$

where $\lambda_j = p_j y_j$ are the Domar weights under our numeraire.[6]

Step 3: combining. Substituting $d\log A_j^{\text{TFP}} = \gamma_j d\log S_{t+k}$ into (A.4):

$$d\log Y_{t+k}\big|_S = \sum_j \lambda_j \gamma_j d\log S_{t+k} = \bar{\gamma} d\log S_{t+k}.$$

This extends Hulten's theorem to factor-augmenting productivity shocks: the first-order GDP elasticity with respect to S_t equals the Domar-weighted cost share of the factor that S_t augments. Multiplying by $(1-\beta)\beta^k$, summing over k, and taking $\mathbb{E}_t$ gives

$$\mathscr{Y}_t = (1-\beta)\mathbb{E}_t\Big[\sum_{k=0}^{\infty} \beta^k d\log Y_{t+k}\big|_S\Big] = \bar{\gamma}\mathscr{S}_t,$$

which is (25).

Part (b). From Proposition 3, $\kappa = \text{Cov}_t\left(\mathscr{S}_t, \tilde{R}_t^{\text{AI}}\right)/\text{Var}_t\left(\tilde{R}_t^{\text{AI}}\right) = \delta/\delta^S$, so the best linear predictor of $\mathscr{S}_t$ given the residualized index return is $\mathbb{E}\left[\mathscr{S}_t \mid \tilde{R}_t^{\text{AI}}\right] = \kappa \tilde{R}_t^{\text{AI}}$. Since $\mathscr{Y}_t = \bar{\gamma}\mathscr{S}_t$ by Part (a), linearity of the conditional expectation gives

$$\mathbb{E}\left[\mathscr{Y}_t \mid \tilde{R}_t^{\text{AI}}\right] = \bar{\gamma}\kappa \cdot \tilde{R}_t^{\text{AI}} = \frac{\bar{\gamma}}{\delta^S} \cdot \delta \cdot \tilde{R}_t^{\text{AI}}.$$

□

[6] This is the standard Hulten (1978) result. The first-order effect equals the Domar weight because the reallocation of factors and intermediates across firms in response to a small TFP shock has only a second-order effect on GDP — a consequence of the envelope theorem applied to each firm's cost-minimization problem. With constant markups, the markup wedge distorts the allocation relative to the social optimum, but this distortion is a level effect that does not affect the first-order comparative static. See Baqaee and Farhi (2019) for a general treatment.

B Appendix Tables

Table B.1: Twenty largest holdings in the THNQ ETF

Rank	Company	Portfolio Weight (%)
1	Nebius Group (NBIS)	2.90
2	Lumentum Holdings (LITE)	2.74
3	Cloudflare (NET)	2.56
4	Alphabet (GOOGL)	2.44
5	Advanced Micro Devices (AMD)	2.36
6	Datadog (DDOG)	2.36
7	Lam Research (LRCX)	2.34
8	Ambarella (AMBA)	2.34
9	Pure Storage (PSTG)	2.32
10	Alibaba Group (BABA)	2.28
11	Amazon.com (AMZN)	2.26
12	ASML Holding (ASML)	2.25
13	Analog Devices (ADI)	2.24
14	Tempus AI (TEM)	2.22
15	Palo Alto Networks (PANW)	2.21
16	Meta Platforms (META)	2.17
17	NVIDIA (NVDA)	2.14
18	JD.com (JD)	2.14
19	TSMC (2330.TW)	2.14
20	Infineon Technologies (IFX)	2.12
Top 20 Total		45.83

Notes: This table reports the twenty largest holdings in the THNQ ETF by portfolio weight as of March 24, 2026. The narrow range of weights (2.12% to 2.90%) reflects the fund's modified equal-weight methodology. The fund held 56 securities in total on this date. Source: Stock Analysis (stockanalysis.com/etf/thnq/holdings/).

Table B.2: Twenty largest holdings in the AIQ ETF

Rank	Company	Portfolio Weight (%)
1	SK hynix (000660.KRX)	4.34
2	Samsung Electronics (005930.KRX)	4.26
3	Netflix (NFLX)	3.65
4	Micron Technology (MU)	3.54
5	Cisco Systems (CSCO)	3.47
6	TSMC (TSM)	3.34
7	Apple (AAPL)	3.32
8	Broadcom (AVGO)	3.17
9	NVIDIA (NVDA)	3.11
10	Meta Platforms (META)	3.05
11	Alphabet (GOOGL)	3.03
12	Palantir Technologies (PLTR)	3.02
13	Amazon.com (AMZN)	2.91
14	Microsoft (MSFT)	2.81
15	Tencent Holdings (0700.HK)	2.81
16	Oracle (ORCL)	2.79
17	Tesla (TSLA)	2.72
18	AMD (AMD)	2.64
19	IBM (IBM)	2.33
20	Alibaba Group (BABA)	2.30
Top 20 Total		63.61

Notes: This table reports the twenty largest holdings in the AIQ ETF by portfolio weight as of March 24, 2026. The wider range of weights (2.30% to 4.34%) relative to THNQ reflects AIQ's capped market-capitalization weighting scheme. The fund held 84 securities in total on this date. Source: Stock Analysis (stockanalysis.com/etf/aiq/holdings/).

Table B.3: O*NET codes in Revelio's "Software Engineer" category

O*NET Code	Occupation Title	Compensation Share (%)
15-1252.00	Software Developers	33.50
15-1299.08	Computer Systems Engineers/Architects	10.66
15-1254.00	Web Developers	5.78
17-2199.06	Microsystems Engineers	5.06
15-1253.00	Software Quality Assurance Analysts and Testers	3.92
15-1255.00	Web and Digital Interface Designers	3.44
15-2051.00	Data Scientists	3.34
11-9041.00	Architectural and Engineering Managers	3.04
27-1029.00	Designers, All Other	2.81
15-1243.00	Database Architects	2.49
15-1241.00	Computer Network Architects	2.37
15-1299.09	Information Technology Project Managers	2.21
15-1299.07	Blockchain Engineers	1.70
15-1221.00	Computer and Information Research Scientists	1.62
15-1251.00	Computer Programmers	1.43
41-9031.00	Sales Engineers	1.20
15-1299.05	Information Security Engineers	1.12
27-1014.00	Special Effects Artists and Animators	1.00
27-1011.00	Art Directors	0.91
17-2072.01	Radio Frequency Identification Device Specialists	0.89
Top 20 Total		87.39

Notes: This table lists the twenty largest O*NET occupation codes, ranked by compensation share, among the 125 codes that Revelio Labs classifies under the "Software Engineer" role category. The compensation share is each occupation's percentage of the total weighted compensation summed across all workers in all 125 software engineer occupation codes in 2021, where the weighting uses Revelio's individual-level sampling weights. These twenty occupations collectively account for 87.4% of total software engineer compensation. Source: Revelio Labs (Revelio Labs, 2026).

Table B.4: Semiconductor-related firms excluded from the analysis sample

Panel A: NAICS6 composition

NAICS6 Industry	Firms
334413: Semiconductor and Related Device Manufacturing	60
333242: Semiconductor Machinery Manufacturing	7
333310: Commercial and Service Industry Machinery Manufacturing	2
334513: Instruments for Measuring/Controlling Industrial Process Variables	2
334515: Instruments for Measuring/Testing Electricity and Electrical Signals	2
333314: Missing / Unknown	1
334112: Computer Storage Device Manufacturing	1
334511: Search, Detection, Navigation, and Guidance Systems Manufacturing	1
Missing: Missing / Unknown	1

Panel B: 25 largest firms by market capitalization

Firm	Market Cap
Texas Instruments Inc	$174.1 B
Applied Materials Inc	$139.5 B
KLA Corp	$64.8 B
STMicroelectronics NV	$44.5 B
GlobalFoundries Inc	$34.6 B
ON Semiconductor Corp	$29.4 B
Skyworks Solutions Inc	$25.5 B
Enphase Energy Inc	$24.5 B
Monolithic Power Systems Inc	$22.8 B
Entegris Inc	$18.8 B
Qorvo Inc	$17.0 B
SolarEdge Technologies Inc	$14.8 B
Wolfspeed Inc	$13.0 B
Synaptics Inc	$11.4 B
MKS Instruments Inc	$9.7 B

Continued on next page

Table B.4: Semiconductor-related firms excluded from the analysis sample (continued)

Universal Display Corp	$7.9 B
Allegro MicroSystems Inc	$6.9 B
SiTime Corp	$6.1 B
Amkor Technology Inc	$6.1 B
MaxLinear Inc	$5.8 B
Semtech Corp	$5.7 B
Power Integrations Inc	$5.6 B
MACOM Technology Solutions Holdings Inc	$5.5 B
Cirrus Logic Inc	$5.3 B
Diodes Inc	$4.9 B

Notes: Panel A reports the NAICS6 composition of the 77 firms classified as semiconductor-related. Panel B reports the 25 largest of these firms ranked by end-of-2021 market capitalization (in billions of U.S. dollars). Firms are classified as semiconductor-related using Claude based on their business description and primary industry.

Table B.5: Summary statistics for firm-level AI betas

Statistic	Value
N	1794
Mean	-0.018
SD	0.525
Min	-2.975
P10	-0.511
P25	-0.304
Median	-0.077
P75	0.202
P90	0.596
Max	2.920
Share > 0	0.419

Notes: Betas are firm-level coefficients on weekly THNQ excess returns from the baseline Fama-French three-factor first-stage specification over the balanced 2022–2025 panel. Each observation is a unique firm (GVKEY). *Share > 0* is the fraction of firms with a positive beta.

Table B.6: Sensitivity of the baseline coefficient to outlier treatment

Dependent Variable:	Beta on THNQ excess return				
Specification	Baseline	Drop top 1%	Drop top 5%	Winsorize 99/1	Winsorize 95/5
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
Software engineer payroll share	1.54***	1.55***	2.03***	1.61***	2.07***
	(0.23)	(0.27)	(0.38)	(0.25)	(0.30)
Fixed-effects					
NAICS3	Yes	Yes	Yes	Yes	Yes
Quintile of average 2017-2021 assets	Yes	Yes	Yes	Yes	Yes
Quintile of average 2017-2021 sales	Yes	Yes	Yes	Yes	Yes
Fit statistics					
Observations	1,787	1,769	1,699	1,787	1,787
R^2	0.16	0.16	0.15	0.16	0.16
Within R^2	0.03	0.02	0.02	0.03	0.03

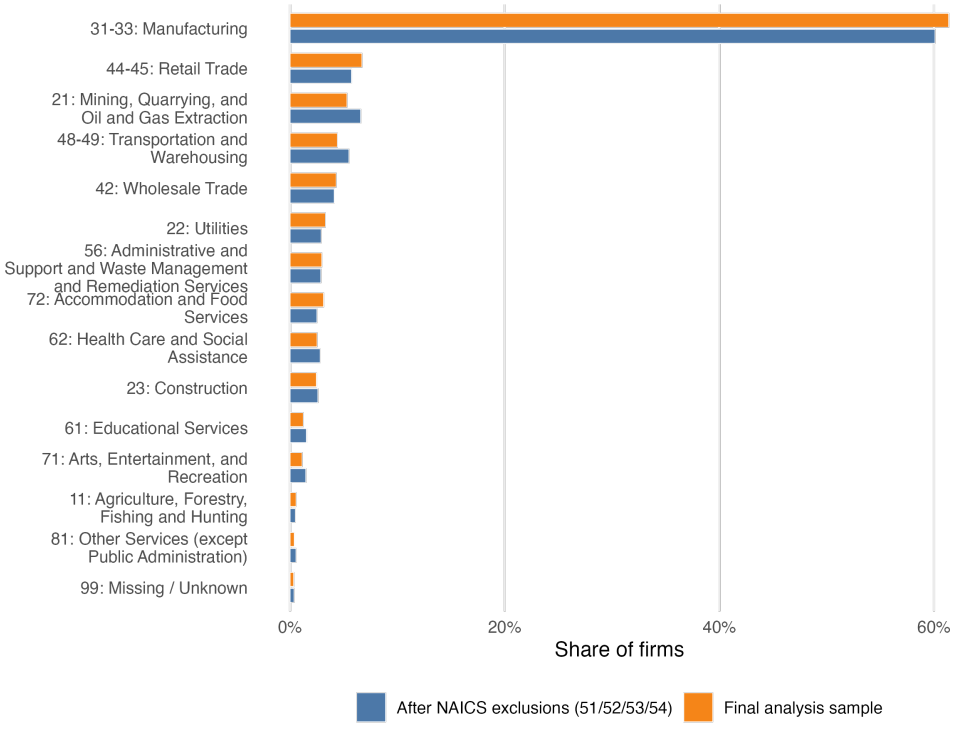
HC3 standard-errors in parentheses

*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

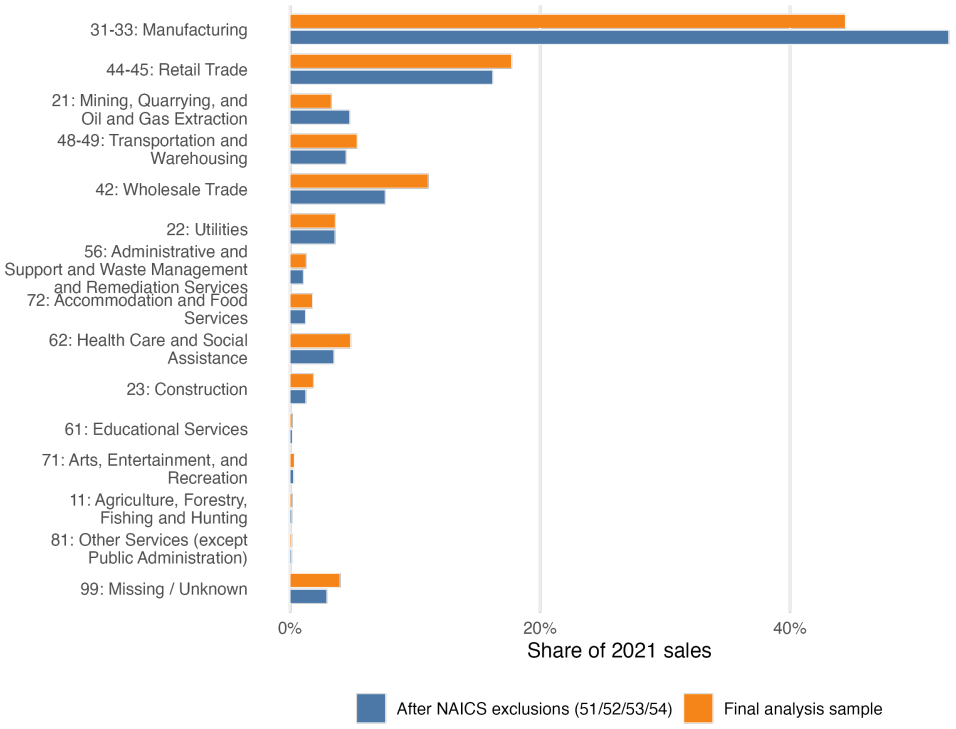
Notes: This table starts from the baseline NAICS3 fixed-effects specification and varies only the treatment of the software engineer payroll-share regressor. "Drop top 1%" and "Drop top 5%" exclude firms above the corresponding percentile of the continuous share measure in the baseline estimation sample. "Winsorize 99/1" and "Winsorize 95/5" cap the regressor at the indicated lower and upper percentiles while leaving the estimation sample unchanged. All other controls, fixed effects, and the dependent variable are held fixed across columns. HC3 standard errors are reported in parentheses. Observations are firms.

C Appendix Figures

Figure C.1: Sectoral composition of the analysis sample



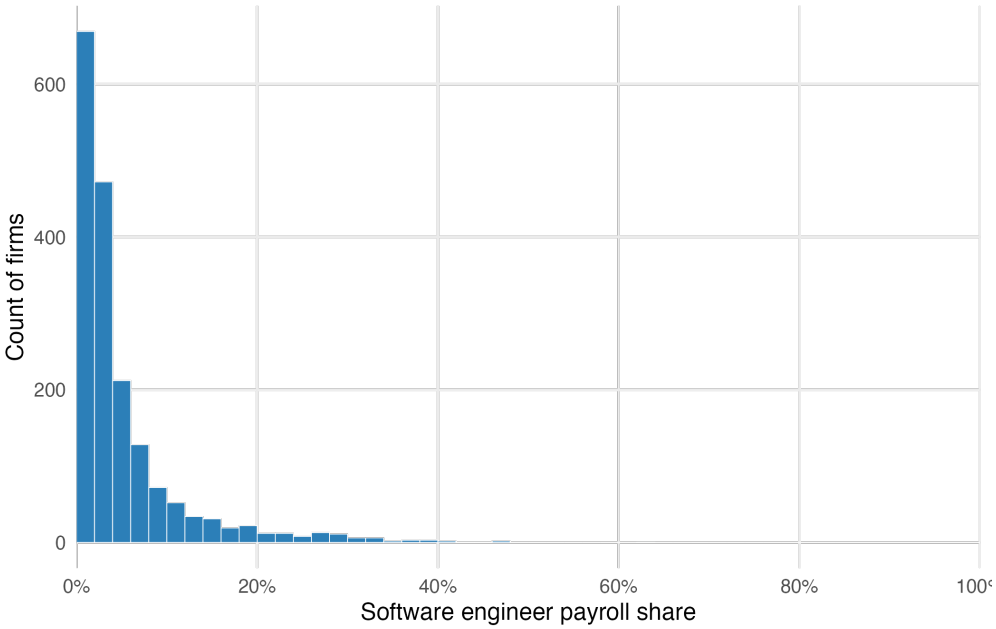
(a) Share of firms



(b) Share of 2021 sales

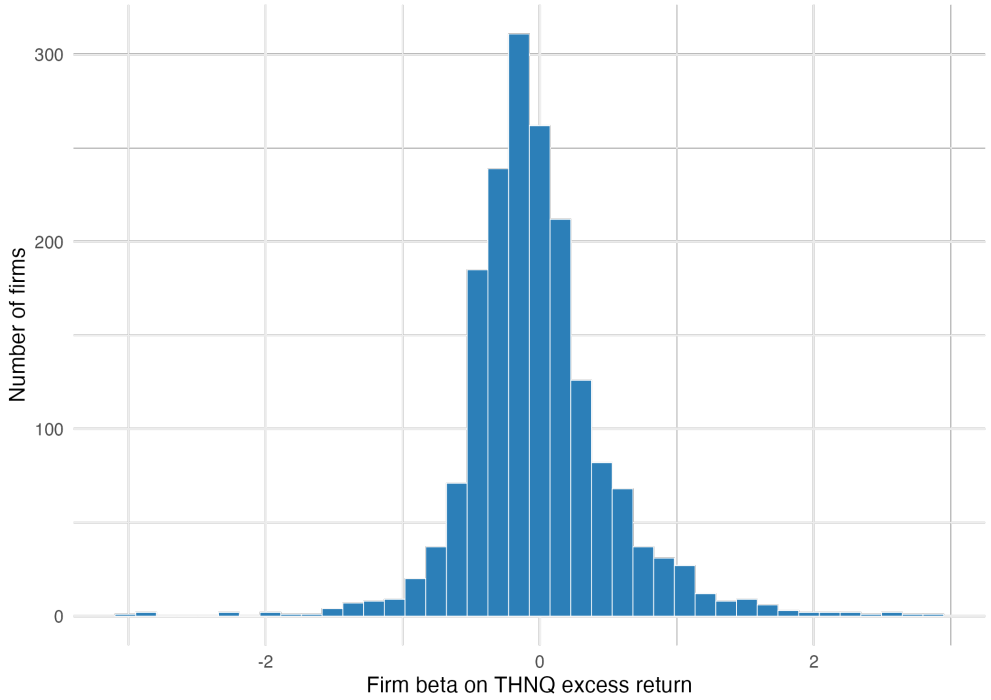
Notes: Panel (a) shows the share of firms in each NAICS2 sector; panel (b) shows the share of aggregate 2021 sales. The light bars show the distribution after dropping NAICS sectors 51, 52, 53, and 54. The dark bars show the distribution in the final analysis sample after all remaining filters (semiconductor exclusion, Revelio match, balanced panel, and size trim).

Figure C.2: Distribution of software engineer payroll shares



Notes: This figure shows the histogram of the software engineer payroll share across firms in the final analysis sample, measured in 2021. The payroll share is the fraction of total firm compensation attributable to workers classified under Revelio's "Software Engineer" role category. The distribution has a mass point at zero (196 firms) and a long right tail.

Figure C.3: Distribution of firm-level AI betas (baseline specification)



Notes: This figure shows the histogram of estimated firm-level AI betas from the first-stage regression of weekly firm excess returns on THNQ excess returns, controlling for the Fama-French three factors (MKT, SMB, HML), over the balanced 2022–2025 panel. Each observation is a unique firm.