Firm Productivity and Learning in the Digital Economy: Evidence from Cloud Computing*

James Brand† Mert Demirer‡ Connor Finucane§ Avner A. Kreps¶

August 20, 2024

Preliminary Draft: Please do not cite or distribute without consent of the authors.

Abstract

Computing technologies have become critical inputs to production in the modern firm. However, there is little large-scale evidence on how efficiently firms use these technologies. In this paper, we study firm productivity and learning in cloud computing by leveraging CPU utilization data from over one billion virtual machines used by nearly 100,000 firms. We find large and persistent heterogeneity in compute productivity, both across and within firms, similar to canonical results in the literature. More productive firms respond better to demand fluctuations, use a wider variety of specialized machines, and show higher attentiveness to resource utilization. Notably, productivity is dynamic as firms learn to be more productive over time. New cloud adopters improve their productivity by more than 30% in their first year and reach the productivity level of experienced firms within four years. In our counterfactual calculations, we estimate that raising all firms to the 80th percentile of productivity would reduce aggregate electricity usage by 17%.

JEL: D24, L86

Keywords: Productivity, cloud computing, learning, computation

*We thank Dan Ackerberg, Vivek Bhattacharya, Noman Bashir, Jeffrey Campbell, Ambar La Forgia, Sonia Jaffe, Bob Gibbons, Matthew Grennan, Patrick Hummel, Gaston Illanes, Donald Ngwe, Rob Porter, Devesh Raval, Chad Syverson and Neil Thomson for helpful conversations, comments and suggestions. Aaron Banks provided excellent research assistance.

†Microsoft, jamesbrand@microsoft.com

‡MIT Sloan, mdemirer@mit.edu

§Microsoft, cfinucane@microsoft.com

¶Northwestern University, avner@u.northwestern.edu

1 Introduction

Firm productivity has long been a central focus in economic research, with an extensive literature investigating its dispersion, persistence, and underlying drivers (Syverson, 2004; Foster et al., 2008). Production is a dynamic process that often requires firms to adapt to new technologies and, in recent years, the digital revolution has dramatically transformed production across all sectors (Agrawal et al., 2018; Goldfarb and Tucker, 2019). With the acceleration of digitization and the emergence of AI, firms are increasingly integrating digital technologies into production, relying heavily on computation and data (Brynjolfsson and McElheran, 2016; McElheran et al., 2024).

While an extensive literature explores the rise of the digital economy and the impact of IT on firm productivity (Brynjolfsson and Hitt, 2003; Bartel et al., 2007; Bloom et al., 2012), there is a notable gap in our understanding of how productively firms use these new technologies. Studies of IT adoption often view IT as a complementary technology, helping firms improve existing modes of production (Brynjolfsson and Milgrom, 2013). However, the rise of the digital economy emphasizes the importance of treating IT itself as a mode of production, making IT productivity an object of interest.

A key reason for the limited evidence on emerging inputs is the prevalence of total factor productivity (TFP) in empirical and methodological studies of productivity. Most studies focus on manufacturing sectors and often use plant-level TFP as the main metric of productivity (Bartelsman and Doms, 2000; Syverson, 2011). However, TFP is inherently a 'residual' measure, quantifying output unexplained by observed inputs (Solow, 1957). The black-box nature of TFP limits insight into the mechanisms underlying firm productivity, which is particularly important when studying emerging technologies.

In this paper, we develop and study new measures of firm productivity in the digital economy using data from a large cloud computing provider. Our analysis draws on high-frequency CPU utilization data from over 1 billion virtual machines (VMs) used by nearly 100,000 firms across various industries and countries. Using this dataset, we construct firm- and division-level compute productivity measures that quantify the extent to which organizations could perform the same computing tasks with fewer compute inputs. We use our measures to analyze dispersion in compute productivity both across and within firms, as well as the process by which firms learn to use computing more productively.

Our compute productivity measure tracks how efficiently firms utilize computing resources, similar to other factor productivity measures such as labor productivity (Baily et al., 2001; Bandiera et al., 2009). While it directly informs us about productivity in the digital economy, it also offers unique advantages that can advance our understanding of

firms and firm productivity beyond digital technologies. First, as a quantity-based measure attributable to a specific aspect of production, it allows for precise quantification of the economic resources used in production, such as IT hardware and electricity. Second, it enables high-frequency tracking of 'machine-level' production, offering detailed insights into the mechanisms driving productivity dispersion and learning processes. Lastly, it allows for the estimation of productivity for individual divisions within firms, improving our understanding of intra-firm productivity dynamics.

We begin our paper by summarizing the concept of productivity in the context of computing. In computing, the productivity of resources, such as server hardware, is commonly measured by their utilization rate. Achieving high utilization has been a persistent challenge for firms, both historically in on-premise computing and in the current cloud era (Whitney et al., 2014). Compared to on-premise computing, which requires firms to make periodic capital investments, cloud computing lets firms rent compute resources on demand, eliminating the need to maintain excess capacity. To take full advantage of this flexibility, firms must monitor usage and efficiently deploy compute resources to match varying workloads. Existing evidence highlights the difficulty of this task: industry surveys and utilization studies demonstrate persistent self-reported compute underutilization on the cloud, which is often attributed to monitoring problems within firms, a lack of organizational adaptability, and the new skills required to use the cloud effectively (Cortez et al., 2017; Tirmazi et al., 2020; Everman et al., 2022; Flexera, 2023).

In cloud computing, virtual machines (VMs) function as the fundamental unit of production for firms. Typically, a firm deploys many VMs simultaneously to meet varying computing needs, such as managing website traffic, hosting internal applications, or running product simulations. This process can be managed manually or automatically through tools like autoscaling, which dynamically adjusts compute resources within seconds based on workload. Cloud providers offer a vast menu of VMs with different characteristics and sizes, making choosing the right VM a nontrivial task.

Cloud computing pricing mainly follows a linear model, where firms pay a fixed per-minute price for each deployed VM. This price scales with the VM's capacity and is paid regardless of how much the firm actually uses the VM. With these characteristics, cloud computing represents an input with truly marginal cost, enabling firms to scale flexibly with negligible dynamic frictions and adjustment costs (Pindyck, 1986; Asker et al., 2014).

Our empirical analysis relies on a firm-level compute productivity measure constructed using individual VM utilization data collected at 5-minute intervals. This measure aims to quantify the firm's resource usage relative to that of a cost-minimizing firm with the same compute output. Following industry practices, we base our productivity measure

on peak CPU utilization, which measures how many computations a machine makes as a percentage of its capacity.

Our measure incorporates two sources of inefficient usage: idle and overprovisioned VMs. We define a VM as *idle* if the firm never uses it and *overprovisioned* if its peak compute load would fit within the capacity of a smaller VM with the same characteristics. The underlying idea is that the firm could eliminate idle VMs or downsize overprovisioned ones without impacting their computing usage, thus saving resources and money. Both of these forms of inefficiency are commonly discussed in the industry, and our definitions of them are similar to productivity measurements by cloud platforms and in the operating systems literature (Weinman, 2011; Islam et al., 2012). We measure peak utilization over a seven-day period at the VM-level, ensuring that we do not misinterpret potential low average utilization due to compute demand volatility as inefficiency. By aggregating both sources of VM-level inefficiency, we obtain monthly firm- and division-level productivity measures.

Using this measure, we first document new empirical facts about productivity in computing. Our findings reveal significant dispersion in compute productivity across firms that is persistent in short (1-month) and long-term (5-year) horizons. Controlling for industry and month, we find that firms at the 90th percentile of the distribution are 3.2 times more efficient than those at the 10th percentile. There is also substantial within-firm productivity dispersion: nearly 40% of the cross-division variance in productivity is within the firm. The levels of dispersion and persistence in compute productivity are comparable to findings in other productivity studies (Syverson, 2011; Cunningham et al., 2023).

We then explore factors contributing to productivity differences, starting with observable characteristics. Our dataset includes not only standard firm-level observables such as industry and size but also detailed VM characteristics, including machine types, memory, and operating systems. We find that these observables have limited explanatory power for productivity differences. Industry and firm size account for less than 1% of productivity dispersion, and even the most detailed VM characteristics explain less than 15% of variation. This points to the role of unobserved heterogeneity in compute productivity, again mirroring common findings in the literature (Fox and Smeets, 2011; Metcalfe et al., 2023).

Using our data, we go beyond dispersion and ask what makes firms more productive in computing. Our analysis establishes three key patterns that differentiate more from less productive firms (above and below industry-level median productivity). First, more productive firms are better able to adjust to fluctuations in workload. We analyze resource consumption on weekends, during which there is a substantial decline in total compute demand, and find that high-productivity firms handle these demand fluctuations with

significantly greater efficiency, reducing their provisioned VMs by 30% more than low-productivity firms. Second, we find that more productive firms use more specialized machines and are better able to match the machine to the job, while less productive firms are much more likely to put all of their jobs on a single type of VM. Finally, we demonstrate that more productive firms exhibit better monitoring of whether workloads are active and shut down idle workloads faster. More productive firms are more than twice as likely to shut down a machine on the day it becomes idle than less productive firms.

After documenting significant variation in productivity across firms, we shift our focus to changes in productivity over time. Learning is a natural focus in this context, given that cloud computing is a relatively new and rapidly evolving technology requiring new skills and organizational investments. Our analysis asks whether firms improve their productivity as they accumulate experience with cloud computing. To study this, we analyze the productivity trajectories of firm cohorts over time, considering both short-term and long-term learning.

We find that firms that adopt cloud technology improve their productivity consistently and substantially over time. Firms with one year of experience are 32% more productive than their initial productivity on average. The rate of improvement slows after the initial year, with firms exhibiting roughly 45% higher productivity by the end of their fourth year and making no improvements thereafter. The extended time to reach steady-state productivity perhaps reflects the need to invest in complementary technologies or practices, as widely documented in the literature (Bresnahan et al., 2002; Tambe et al., 2012).

While there is a large literature on learning-by-doing (Benkard, 2000; Levitt et al., 2013), our data allow us to explore the detailed mechanisms behind *how* firms learn. Do firms reallocate resources to more productive use cases? Do they experiment with different types of VMs to find the best fit for their needs? Or do they simply become more efficient with the products they already use? To answer these questions, we decompose monthly productivity growth at the firm level using the method of Foster et al. (2001), analyzing the contributions of division-level reallocation, entry, and exit.

We find that firm-level learning masks substantial within-firm productivity dynamics. First, we observe significant productivity growth at the division level, persisting even beyond the firm's fourth year, suggesting that individual divisions continue to improve their productivity even when firm-level learning plateaus. The flattening in firms' productivity comes from firms deploying new divisions to the cloud, which themselves start relatively unproductive and need to learn to use the cloud. Therefore, while firms do have less productive divisions exiting the cloud, our results demonstrate that learning at the firm level does not come from reallocation. Underscoring these results is the fact that experienced

divisions cannot fully transfer their learning to new divisions in the same firm; while new divisions in more productive firms do start out as more productive than new divisions in less productive firms, new divisions of experienced firms do not tend to start more efficient or learn quicker than new firms.

In the final part of our paper, we conduct simple counterfactual calculations to quantify the aggregate impact of productivity dispersion in computing on economic resources. We estimate how much compute resources and electricity would be saved if all firms were to achieve a given benchmark productivity level. To perform this analysis, we first estimate the relationship between electricity consumption and CPU utilization using a separate dataset with information on electricity usage and utilization. We find that idle machines consume 50% of their full-load electricity, and every one percentage point (pp) increase in utilization leads to a 0.5 pp increase in electricity consumption. This analysis highlights the importance of estimating the machine-level 'production function' to accurately analyze the economic implications of productivity dispersion.

Our results reveal significant aggregate implications of productivity dispersion in computing. If the productivity of all firms below the 80th percentile rose to that level, the economy would reduce compute resources by 22% and electricity consumption by 17%. The difference between these two figures underscores a nonlinear relationship between productivity distribution and underlying resource consumption.

We take steps to ensure our method accurately captures compute productivity without picking up potential confounding factors. First, we intentionally estimate a conservative measure by considering peak utilization over a seven-day period[1] and excluding complex overprovisioning scenarios, like consolidating multiple jobs onto fewer VMs. While these choices may overestimate the average productivity level, they ensure our measures reflect genuine inefficiency. Second, we show that our results remain similar when controlling for a rich set of VM characteristics and that dispersion in compute productivity is not simply explained by firm-level observables. Third, we extensively review industry literature to demonstrate that our measure closely aligns with how firms themselves measure inefficiency in practice. Fourth, we repeat productivity estimation with publicly available utilization data from various cloud providers and find similar productivity dispersion. Fifth, we conduct robustness checks using other utilization metrics in computing (memory and networking) and find similar results.

We nevertheless acknowledge some limitations of our study. First, our approach does not capture all forms of compute inefficiency, such as inefficiently written code or ineffi-

[1]That is, for instance, a VM that runs at 90% utilization for several hours but is otherwise idle is classified as efficient.

ciently run data centers. While important, such inefficiencies are fundamentally different, as they pertain to changing the production process rather than resource deployment. Second, our dataset is limited to compute inputs, as we do not observe other firm inputs or output. We believe this is a worthwhile trade-off given the extensive research on IT's impact on various firm-level measures (spending, capital, labor, and revenue) while micro-analysis of IT usage remains scarce.

Contribution to the Literature. First and foremost, this paper contributes to the large literature on firm productivity (e.g., Syverson, 2004; Bloom and Van Reenen, 2007; Hsieh and Klenow, 2009; see Syverson, 2011 for a review). This body of research has documented significant dispersion in firm productivity within narrow industries and analyzed the drivers of productivity differences. Our paper extends this literature by estimating the firm productivity in computing, an input that is increasingly important for modern firms. We document several empirical facts about compute productivity that parallel findings in the broader productivity literature. Moreover, we observe productivity at an extremely granular level, enabling us to quantify the link between productivity and resource consumption and analyze underlying mechanisms in detail.

Our paper also contributes to studies on the effect of IT investment on firm performance (Brynjolfsson and Hitt, 2003; Bartel et al., 2007; Bloom et al., 2012; Brynjolfsson and McElheran, 2016; Brynjolfsson et al., 2023). This literature studies the impact of IT inputs on several firm outcomes, including productivity, profit, and firm growth. Other papers in this literature have focused on case studies, understanding the mechanisms of IT's effect on firms (Baker and Hubbard, 2004; Miller and Tucker, 2011). A key insight from this literature is that IT performance depends on complementary investments and organizational structure (Bresnahan et al., 2002). Our work extends this literature by shifting focus from the benefits of IT adoption to the efficient use of these technologies.

By demonstrating firm learning in compute productivity, our paper contributes to the empirical literature on learning-by-doing (Benkard, 2000; Thornton and Thompson, 2001; Kellogg, 2011; Levitt et al., 2013; Hendel and Spiegel, 2014; Tadelis et al., 2023). This literature typically analyzes a single firm or a small number of firms in a narrow industry, showing productivity improvements with firm experience. Our study differs by providing large-scale evidence from a diverse set of firms across various industries and focusing on learning in an emerging technology.

Finally, we contribute to the recent but growing literature on the economics of cloud computing (Jin and McElheran, 2017; Jin, 2022; DeStefano et al., 2023; Demirer et al., 2024; Lu et al., 2024) by quantifying productivity dispersion and learning on the cloud.

2 The Role of Computing in Firm Production

This section provides an overview of firms' use of computing technologies, with a focus on cloud computing. Additional details on cloud computing are provided in Appendix A.

2.1 Background on Computing

Computing has become a fundamental input for firms across all industries. Along with storage and networking, computing is an essential component of firms' IT infrastructure. At a fundamental level, computing enables firms to process data or perform specific tasks requiring calculations using a combination of hardware and software. Firms use computing in various ways, depending on their industry, size, and specific needs.

One can categorize firms' use of computing into production, development, and administrative purposes. In production, firms rely extensively on compute resources like servers and data centers to deliver digital services such as streaming, online banking, mobile applications, and websites (Greenstein, 2020). Additionally, compute is a complementary input in producing non-digital services, facilitating functions such as payment processing, customer relationship management, inventory management, logistics optimization, and predictive analytics (Zolas et al., 2021).

Firms that produce software applications rely heavily on compute resources throughout the product development cycle. In the non-digital context, with the rise of computer-aided design (CAD), manufacturing firms use compute resources to design and test products before physical prototyping and production (Leigh et al., 2020). Additionally, firms employ computing technology for various administrative functions, such as human resources, finance, sales, and personnel communications.

2.2 Background on Cloud Computing

Traditionally, computing is done on servers purchased and maintained by individual firms, known as "on-premise" computing. More recently, however, advancements in server technology and fast networking have given rise to cloud computing, which allows firms to access IT services remotely over the Internet. In cloud computing, the physical resources are owned and maintained by cloud providers, and firms have on-demand access to these resources via a rental market.[2] Cloud computing is one of the most rapidly adopted technologies by firms in recent years, with nearly 80% of firms using at least one IT function on the cloud as of 2018 (Kalyani et al., 2021; Zolas et al., 2021).

[2]A third form of IT infrastructure, occupying a middle ground between on-premise and cloud computing, is colocation, in which firms rent space in a data center to house their own servers and networking equipment.

The services offered by cloud platforms fall into three categories: software as a service (SaaS), platforms as a service (PaaS), and infrastructure as a service (IaaS). From SaaS to IaaS, each type progressively increases the user's responsibility for managing the underlying infrastructure. SaaS products (e.g., Microsoft Office 365, Google Workspace, Dropbox) abstract away all the complexities of the underlying infrastructure, enabling users to focus entirely on using the application. PaaS products (e.g., Salesforce Platform, Google App Engine) provide tools and environments for development without requiring users to manage the infrastructure.

IaaS, which is the focus of our paper, refers to fundamental components of IT infrastructure like computing, storage, and networking, which are managed by cloud users directly. This direct management allows users to maintain granular control over their IT environments, customizing them according to specific needs and requirements. In this way, IaaS serves as a modern alternative to on-premise IT, offering similar levels of control with the added benefits of scalability and reduced physical infrastructure costs (Jin and McElheran, 2017).[3]

2.2.1 Virtual Machines

VMs are the primary compute resources that firms use when running workloads in the cloud and can be considered the primary production unit in computing. VMs enable a single server to run multiple isolated operating systems or applications, each with its own dedicated CPU, memory, and storage. This technology, known as virtualization, allows cloud providers to partition the same physical machine into separate resources and allocate them to different firms independently. VMs represent the most widely used IaaS products and are offered across all major cloud platforms.[4,5]

Firms typically use multiple VMs simultaneously to support their operations, often running tens or even hundreds of VMs concurrently. For instance, a medium-sized firm might maintain a cluster of 50-100 VMs distributed across various functions: web servers, application servers, databases, and development environments. Firms can deploy and manage VMs either manually or automatically. VMs can be manually provisioned through cloud providers' websites or interfaces. This process typically involves logging into the platform

[3]IaaS accounts for a large fraction of cloud computing revenue — over 20% of the 2022 total of $545.8 billion. IDC— Worldwide Public Cloud Services Revenues.

[4]Many readers may have interacted with a virtual or remote desktop supported by an underlying VM.

[5]Cloud providers also offer a range of more sophisticated products like 'serverless' computing, which charges users only for the resources they use, and 'container' services, which allow for isolated application development. These products are less commonly used and often require specialized knowledge and specific operational contexts, so they are not suitable for all types of applications. We provide an overview of these cloud products in Appendix A.3.1.

Figure 1: Sample of VM Choices on Amazon Web Services

Instance name	On-Demand hourly rate	vCPU	Memory	Storage	Network performance
t4g.nano	$0.0042	2	0.5 GiB	EBS Only	Up to 5 Gigabit
t4g.micro	$0.0084	2	1 GiB	EBS Only	Up to 5 Gigabit
t4g.small	$0.0168	2	2 GiB	EBS Only	Up to 5 Gigabit
t4g.medium	$0.0336	2	4 GiB	EBS Only	Up to 5 Gigabit
t4g.large	$0.0672	2	8 GiB	EBS Only	Up to 5 Gigabit
t4g.xlarge	$0.1344	4	16 GiB	EBS Only	Up to 5 Gigabit

Notes: This figure presents a sample of VM choices from one family of VMs available on Amazon Web Services (AWS). The table outlines various VM instance types along with their associated on-demand hourly rates, virtual CPUs (vCPUs), memory, storage, and network characteristics.

and selecting the desired VM specifications. VMs can also be deployed automatically using scripts or infrastructure as code (IaC) tools. This approach allows developers and IT teams to pre-define VM configurations in code, which can be executed to provision VMs in response to changing demand. Once requested, VMs are typically deployed in only a few seconds (Nguyen and Lebre, 2017; Tirmazi et al., 2020).[6]

When selecting a VM in the cloud, firms can choose between many configurations of CPU, memory, storage, and networking designed for different use cases. For example, Amazon Web Services (AWS) offers a menu of VM options even within a single "instance type," as shown in Figure 1. In this family, the storage and network options are fixed, but customers can choose between 2 or 4 CPUs (cores) and CPU-to-memory ratios ranging from 4:1 to 1:4. In choosing VM types, firms must consider factors such as price, memory, and storage, and evaluate them against their need. Firms face a vast menu of VM configurations; AWS advertises over 750 different types of VM instances on its EC2 public cloud.[7]

The pricing of VM instances typically depends on several factors, including the instance type, hardware model, operating system, and geographic region (Hummel and Schwarz, 2022). More powerful instances with higher specifications generally cost more. However, for a given VM model, firms are usually charged per unit of time at a fixed rate regardless of the intensity, purpose, or actual utilization of the VM.[8]

[6]We provide more details on VMs and VM deployment in Appendix A.1. For technically oriented readers, we provide descriptions of VM deployment for two real-world applications in web development and machine learning in Appendix A.2.

[7]AWS EC2— Overview.

[8]Cloud providers offer discounts if firms commit to using cloud services over a specific period of time (typically one year or three years). These discounts are called "reserved instance" or "committed use" discounts, depending on the provider. These discounts are applied to the list price and are the same across customers except for very large ones. For more details, see Appendix A.3.3.

In cloud computing, VMs can be categorized in different ways.[9] A VM *type* refers to a specific configuration tailored for particular workloads, such as compute- or memory-intensive. A VM *family or series* refers to a specific grouping of VMs characterized by particular performance characteristics, hardware configurations, or intended use cases; see Table OA-2 for examples of VM series offered by different cloud providers. A VM *configuration* is the exact combination of machine characteristics for a given machine series, which includes memory, cores, and operating system.

For a given VM type, computing capacity is primarily determined by the number of cores—independent processing units that execute tasks simultaneously. More cores allow for more concurrent computations, increasing overall capacity. The product of the number of cores and the duration of the VM's use gives *core-hours*, the unit of computation resource we use throughout the paper.

2.3 The Economics of Cloud versus On-Premise Computing

The advent of cloud computing shifted the economics of compute provisioning dramatically, reducing the fixed costs of acquiring and maintaining compute hardware and increasing the size and importance of marginal costs (Etro, 2015). Traditional on-premise IT involves hosting physical servers in data centers located within an organization's facilities. In this paradigm, computing capacity is a capital expenditure: firms make periodic investments under uncertainty and maintain servers that depreciate over time (Pindyck, 1986). This leads to a classical peak-load problem: firms need to provision enough capacity to handle peak loads, which typically leads to underutilization during off-peak periods (Brown and Johnson, 1969; Carlton, 1977). Indeed, Whitney et al. (2014) estimate that utilization rates of on-premise servers are as low as 12%. As a result, firms face a tradeoff between the costs of maintaining excess capacity and the risks of insufficient capacity to meet demand.

Cloud computing flips this paradigm by shifting computing from a capital expenditure to a variable cost. Firms using the cloud no longer need to worry about capacity planning or resource constraints, as capacity is always available on demand.[10] This allows firms to scale their compute resources based on varying requirements. Thus, firms can accommodate sudden spikes in traffic, handle increased workloads during peak periods, or scale down

[9] Appendix B.2 contains more details on each of these categories.

[10] In practice, firms initially request a quota, and the capacity is available up to the quota requested by the firm. Except for a few specialized VMs, firms can choose a quota that is high enough for their computing needs. Cloud providers can offer enough on-demand capacity using spot instances, in which VMs are available at a significant discount but can be reclaimed with short notice. We exclude spot instances from our sample, which account for a small share of the IaaS market. For details, see Appendix A.3.2.

when resources are no longer needed without the need to invest in and maintain hardware.

In summary, linear pricing, instant scalability, and minimal transaction and adjustment costs make computing a truly variable input, free from adjustment costs or dynamic frictions (Asker et al., 2014). This makes cloud computing an ideal environment for studying productivity, as we can abstract away from dynamic considerations and many of the frictions associated with dynamic inputs that could justify inefficiencies.

2.4 Drivers of Productivity in Cloud Computing

There is widespread acknowledgment in the industry that while cloud computing brings numerous benefits, it also introduces new challenges that firms must manage for efficient use. Industry sources and economic theory highlight two primary challenges: organizational monitoring frictions and the new skills required for cloud computing.

First, cloud computing can exacerbate latent monitoring problems within firms. Real-time monitoring of the costs and benefits of compute usage is far more important in the cloud than in on-premise computing because the marginal cost of compute usage is positive. Engineers and data scientists who provision VMs may not naturally have an incentive to be cost-conscious, creating principal-agent problems for the firm. Organizational inertia can make it hard to fix this monitoring problem quickly — as firms transition to cloud computing, efficiency requires new management practices and complementary investment, referred to as 'digital capital' by Tambe et al. (2020), that may be difficult or take time to implement.[11] For example, one industry survey notes that 81% of respondents say that "their development teams are embracing the cloud and other technologies faster than the rest of the organization can adopt and manage them" (Couchbase, 2022). According to an employee of a cloud cost optimization startup we interviewed, even when management would like to cut costs, it is challenging to get engineers to take action because there is a lack of incentive for full engagement, and large companies, in particular, are subject to operational inertia that hinder full internalization of organizational objectives.

Second, cloud computing requires new skills. Chief among these is choosing a VM and using various tools available in the cloud. This requires adaptation and learning on the part of employees of firms that shift to the cloud. Indeed, one industry report says that finding the best match for a workload in a cloud provider's inventory is "easier said than done," with many teams "simply choos[ing] instances they know and have used before," which tends to "underutiliz[e] other resources that they have paid for" (CAST AI, 2024). Another report states that many firms either "do not have the skills they need to manage

[11] These challenges were also observed in the previous major shift in computing, the transition from mainframe to client/server architectures, as documented by Bresnahan et al. (1996).

their database infrastructure in-house, or they are using resources that could create greater value if used elsewhere in the business" (Couchbase, 2022).

The challenge of optimizing cloud spending has led to the development of a plethora of tools to help firms become more efficient. First, cloud platforms themselves offer customers various ways to view and manage their efficiency, often proposing steps to save money and alerting customers when costs exceed anticipated levels.[12] Autoscaling and load balancing are the most important first-party efficiency tools, allowing customers to set rules for shutting off underutilized VMs and deploying new VMs during high utilization periods.[13] For example, an engineer managing web traffic for an online retailer may use autoscaling to scale up computing capacity during promotions and scale down afterward. We provide an overview of first-party tools in Appendix A.1.2 and Table OA-3.

There are also several third-party tools to help cloud users optimize their costs. This includes a large and fast-growing market of cloud cost optimization consultants who analyze firms' cloud usage and recommend ways in which firms can become more efficient. These consultants are typically more deeply engaged with the organization and provide more tailored recommendations than first-party tools do. This market was worth $17.6 billion in 2022 and is forecasted to reach over $80 billion by 2030.[14] There are also open-source best practices collected in a framework called FinOps, short for Financial Operations, to help firms manage their cloud resources efficiently.[15]

These tools underscore that while the cloud provides flexibility to firms, it brings its own set of impediments to achieving full efficiency. These challenges are nontrivial but tractable, as there are many tools that can help firms improve their productivity. As such, it is natural to think that productivity on the cloud may be heterogeneous and that firms may enjoy gains from experience, two patterns that we explore empirically below.

3 Data and Summary Statistics

This section introduces the datasets used in our analysis and highlights key summary statistics. We provide a more detailed description of the data in Appendix B.

[12] AWS— Reporting and Cost Optimization; Azure— Deployment Optimizer; Google Cloud—Cost Management.

[13] See Figure OA-8 for an illustration of load balancer.

[14] Yahoo Finance— Global Cloud Computing Report. For an example of the tool provided by one of these startups, see Figure OA-7.

[15] The FinOps Foundation is a non-profit organization hosted at the Linux Foundation. It provides a community, resources, and best practices for professionals and organizations to manage cloud costs effectively.

3.1 CPU Utilization Data

Our primary dataset contains detailed information on the CPU utilization of a large random sample of VMs from a global cloud provider. CPU utilization measures how much of a computer's processing capacity is currently in use relative to the maximum load it can handle, and it is a critical metric in assessing a computer's performance efficiency (Mason et al., 2018). Computing systems typically record CPU utilization statistics at 5 or 10-minute intervals. To make this data more manageable, we aggregate it to the VM-day level, recording the CDF of CPU utilization every day while the VM is active.[16]

The CPU utilization data are available intermittently between 2017 and 2023, with varying durations each year. Our earliest data are from 2017, covering approximately 60 days, while in 2018 and 2019, the collection period was about 30 days each year. No data is available for 2020 and 2021, but we have a consecutive 12 months of data across 2022 and 2023.[17] Although this intermittent data may limit some analyses, it still allows us to estimate productivity changes for up to six years and to track short-term changes in productivity.

For each VM in our data, we observe its duration, the anonymized firm ID that uses the VM, and the anonymized division ID for multi-division firms. A "division" in our data collects all users that share an administrative structure for oversight of the VMs and a payment/billing contract with the cloud provider. These divisions may correspond to product teams or functional divisions within the company, though no further information is available. As such, we will refer to these as "units" in the rest of the paper.

We also observe various attributes of the VMs, including machine type, a machine series ID that provides information on the hardware manufacturer and series, operating system, memory, and number of cores. Our data also specify the region of the data center that hosts the VM (EU, US, and others) and an anonymized data center ID.

3.2 Firm and Firm-Division Level Data

Although our CPU utilization data is an unbalanced panel, we have a balanced monthly panel at the firm and unit levels. These data cover the period from 2017 to mid-2023 and

[16] Additionally, we have VM-day level memory and network utilization data (maximum, 90th percentile, and average) for one-month periods in 2022 and 2023. Memory utilization indicates the amount of RAM being used by a system, while network utilization measures the data transfer usage relative to the total available bandwidth. While we analyze this data as a robustness check, we primarily focus on CPU utilization to measure computing efficiency, as it directly correlates with resource consumption (power use), is more easily observed by engineers, and is more often the binding resource constraint in cloud environments (Mason et al., 2018).

[17] The company stored this data sporadically for independent reasons and made it available to us for research.

include each firm's and unit's normalized monthly total computation in core-hours. With these data, we can track firms' entry/exit into or out of the cloud and changes in compute usage over time.[18] In addition, at the firm level, we observe the region of the firm (EU, US, or other), whether the firm is multinational, its industry classification, and quartiles of a size measure, which proxies the number of employees. At the unit level, we observe the industry classification and the region of usage.

3.3 Publicly Available Cloud Data

We supplement our main dataset with publicly available CPU utilization data from various cloud providers, such as Google and Microsoft. These datasets provide additional information not available in our main data and allow us to validate our findings in other cloud computing environments. One of these datasets is the 2019 Power Traces from Google Cloud, which records 5-minute electricity consumption from 55 power domains within a data center and CPU utilization for all VMs hosted on servers connected to these power domains. This dataset allows us to estimate the relationship between CPU utilization and electricity consumption, quantifying resource usage by underprovisioned VMs. Further details of public cloud datasets can be found in Appendix B.6.

3.4 Sample Construction and Summary Statistics

We use sampling to reduce the data size and eliminate the inclusion of confidential information. First, we trim many firms that fall below certain thresholds for the magnitude and consistency of VM usage and sample firms on the right tail of the total usage distribution using a sampling rate. Then, we randomly sample a percentage of VM-days at a fixed sampling rate for every remaining firm.[19] This allows us to obtain a representative sample of VMs for each firm. In addition to this initial data processing, we perform a few data cleaning steps to remove very short VMs and firms with few observations, as detailed in Appendix B.5.

Table 1 presents summary statistics as observed in 2019, the middle of our sample. Panel A shows the industry categories based on 1-digit SIC codes, highlighting the predominant sectors such as services and IT/Software, which account for 36.20% and 23.15% of our sample, respectively. Although smaller in share, we observe non-digital industries such as Manufacturing, Transportation, and Communications. Column (2) shows the percentage

[18] We observe these data from one cloud provider only, and therefore are unable to tell if a firm used a different cloud provider beforehand. As such, our measure of cloud experience will be a lower bound on firms' actual cloud experience.

[19] For confidentiality reasons, we do not reveal the sampling rate, but it is between 70 and 100 percent.

Table 1: Distribution of Industries, Firm Regions, and VM Statistics in Mid-Sample (2019)

	Share (%) (1)	Multi-unit (%) (2)	Cloud Experience (3)
	Panel A. Industry Category (1-digit SIC)		
Services	36.20	26.72	1.97
IT/Software	23.15	43.06	2.73
Retail Trade	12.29	32.28	2.00
Manufacturing	9.11	43.27	2.24
Public Administration	7.48	47.20	2.52
Transportation and Communications	6.10	44.11	2.38
Finance, Insurance, and Real Estate	4.54	48.33	2.25
Other	1.12	32.62	1.96
	Panel B. Firm Region		
Other	41.12	29.54	1.90
US	31.22	26.24	1.97
EU	21.43	29.77	1.93
Multinational	6.22	59.81	2.65
	Panel C. VM Statistics		
	Mean	SD	Mode
Duration (days)	2.52	13.75	1
Number of cores	7.88	12.04	4
Share downsizable	0.72	0.45	1

Notes: This table reports summary statistics for industries, regions, and VM statistics in our sample for June 2019, which is the midpoint of our sample. Industries are defined as the ten divisions classified by SIC codes, with the exception of software firms, which are carved out of the services division. Column (1) reports the unweighted shares based on the number of firms. Column (2) reports the number of firms with multiple units in June 2019, and Column (3) reports the average number of years since the firm first used cloud computing. Panel C provides unweighted summary statistics for VMs that were created during a week in 2022, including their duration, number of cores, and share of downsizable VMs.

of firms with multiple units using cloud computing. This percentage ranges from 25% to 45% across industries, indicating that for many firms in our sample, we observe the productivity of multiple divisions. Column (3) shows the average number of years firms have been using cloud computing. Given that cloud computing is a relatively recent technology, the average experience is only a few years, with little variation across industries.

Panel B details the geographical distribution of firms, with 31.22% located in the US, 21.43% in the EU, and 6.22% classified as multinational. As expected, multinational firms are more likely to have multiple units and have, on average, 0.7 years more cloud computing experience than domestic firms. Panel C presents descriptive statistics at the VM level. The

Table 2: Example CPU Usage for a Hypothetical Firm

Job	Capacity	Duration	Actual Input Use	Peak Util.	Peak Load	Efficient Usage	Efficient Input Use	Efficiency
	[a]	[b]	[c] = [a] × [b]	[d]	[e] = [a] × [d]		[f]	[g] = [f] ÷ [c]
Job 1	2-core	10h	20 ch	0%	0 cores	**Eliminate**	0 ch	0%
Job 2	4-core	5h	20 ch	25%	1 core	**Downsize**	10 ch	50%
Job 3	8-core	10h	80 ch	75%	6 cores	**Maintain**	80 ch	100%
Total			**120 ch**				**90 ch**	**75%**

Notes: Table 2 presents a breakdown of CPU usage for three different jobs within a hypothetical firm. Job 1, with a 2-core capacity and minimal utilization, suggests idleness (0% peak utilization), leading to the recommendation to eliminate this job. Job 2, with a 4-core capacity used at 25% peak utilization, indicates overprovisioning, and the advice is to downsize. In contrast, Job 3 shows optimal use of an 8-core capacity at 75% peak utilization, which is maintained as efficiently used. The total CPU input across all jobs accumulates to 120 core-hours with an actual efficient use of 90 core-hours, reflecting an overall efficiency of 75%.

average lifespan of a VM is 2.52 days, but there is significant heterogeneity. Similarly, we observe large variations in the number of cores, reflecting that firms use VMs with varying capacities. Finally, more than 70% of the VMs in the data are downsizable, meaning firms have the opportunity to choose smaller capacity VMs if they overprovision.[20]

4 Productivity in Computing

This section defines how we use our data to measure compute productivity. This productivity measure is based on CPU utilization and aims to quantify the share of compute resources firms use productively, taking into account the constraints firms face in provisioning VMs. We first describe how we construct our productivity measure and then argue that it is consistent with industry practice and has first-order importance to understanding productivity in computing. A more formal description of the construction of our measure is in Appendix C.

4.1 Measuring Compute Productivity

To motivate and illustrate our measure, consider the usage pattern of a hypothetical firm documented in Table 2. Suppose there are three sizes of VMs available for the firm's use case: 2-core, 4-core, and 8-core. The firm runs three jobs: Job 1 on a 2-core machine, Job 2 on a 4-core machine, and Job 3 on an 8-core machine. Jobs 1 and 3 last 10 hours, and Job 2 lasts 5 hours. Therefore, the total computing resource the firm pays for — the firm's total input use — is $2 \times 10 + 4 \times 5 + 8 \times 10 = 120$ core-hours.

[20] We provide a more detailed definition of whether a VM is downsizable in Section 4.

Now suppose that we observe the following utilization patterns. On Job 1, the firm did not actually utilize the machine at all; the peak load for Job 1 was 0 cores. On Job 2, the firm did use the machine, but at most 25% of the computing capacity was used at any given moment. Therefore, the peak load of Job 2 was $25\% \times 4 = 1$ core. Finally, on Job 3, the peak utilization was 75%, meaning the peak load of the job was $75\% \times 8 = 6$ cores. These loads define the cores that were needed to perform the observed workloads and can be used to determine computing efficiency.

What would a perfectly cost-minimizing firm have done if it had the same computing needs and faced the same set of VMs it could provision? First, it would eliminate Job 1, which does not result in any computing output for the firm. By doing so, the firm can avoid paying for 20 core-hours of input. Second, given that Job 2 only requires a capacity of 1 core at peak, the firm would downsize Job 2 to a 2-core machine, reducing the total input usage for Job 2 from 20 core-hours to 10 core-hours. Finally, since Job 3 requires a peak capacity of 6 cores, it cannot be downsized (the next smallest available machine is 4 cores); the cost-minimizing firm would provision the same 8-core machine for the job and use the same 80 core hours of input. Overall, therefore, a cost-minimizing firm would have only used and paid for 90 core-hours, while this firm actually used 120 core-hours. As such, we conclude that this firm could have used 75% as much input as it actually did to get the same output.

Our measure of compute productivity generalizes the logic of this example. We assign each job j run by firm i on day t a productivity $\omega_{ijt} \in [0, 1]$, where, at a high level,

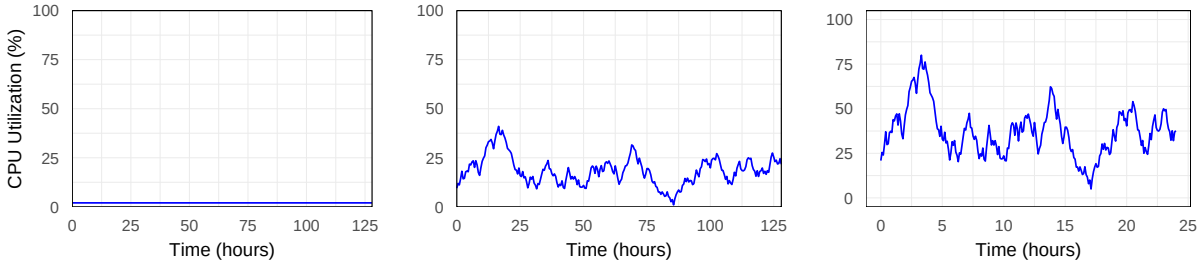
$$\omega_{ijt} = \frac{\text{Minimum number of core-hours needed for job}}{\text{Actual core-hours used for job}}$$

This formula essentially measures (the inverse of) the share of resources that are wasted when running job j. To determine the minimum number of core hours needed for job j, we use its peak utilization over a seven-day period.[21] The focus on peak utilization ensures our measures remain robust to various concerns that might explain low utilization. For instance, our approach does not mark as inefficient low average utilization due to fluctuating demand, nor does it credit very short-term potential savings from briefly turning off a VM. We take peak utilization to be the 95th percentile CPU usage over a seven-day period, following the recommendations made by cloud providers, as well as in the computing literature (Reiss et al., 2012; Cortez et al., 2017).[22]

[21] For VMs shorter than seven days, we use the peak utilization over the life of the VM.

[22] This measurement is less sensitive to measurement errors or spikes stemming from random events like software updates than the maximum CPU utilization.

Figure 2: CPU Usage Patterns



(a) Idle VM **(b)** Overprovisioned VM **(c)** Properly Provisioned VM

Notes: This figure illustrates the CPU usage patterns of three different types of VM usage. Panel (a) shows the CPU utilization of an idle VM, maintaining a constant utilization near 0% throughout the duration. Panel (b) shows an overprovisioned VM where the peak utilization only reaches about 40%. Panel (c) shows a properly provisioned VM with peak utilization above 75%.

Using our methodology, we identify two distinct sources of inefficient VMs: idleness and overprovisioning. Job j is *idle* if the peak CPU utilization for the job is under 10% of the capacity of the VM the firm chose. This amount of utilization is explained by the CPU's background processes rather than any actual CPU usage by the user. Because an idle job does not have any output, the minimum number of core-hours needed for the job is zero, hence $\omega_{ijt} = 0$.

Job j is *overprovisioned* if it is not idle, but would have only reached a peak utilization of 90% or less on a VM that has fewer cores but is otherwise similar.[23] In this case, the minimum number of cores to run the job is that of the smallest such VM that fits the job's peak load. If such a smaller substitute exists, we call the VM *downsizable*, which we define as there existing a configuration with fewer cores but the same machine type, memory, data center, and operating system. Typically, cores scale in powers of two, so an overprovisioned VM will often be resized to a VM with half the number of cores, in which case $\omega_{ijt} = 0.5$.[24] Finally, if a job is neither idle nor overprovisioned, it is *properly provisioned*. In this case, $\omega_{ijt} = 1$.

Figure 2 is another illustration of our measure. Each panel displays the CPU utilization of a job over time. Panel (a) is an idle job. The firm is paying for over 125 hours of computing power even though it actually did not use the VM it provisioned. This idle

[23] By "similar," we mean that the alternative VM has the same machine type, memory, data center, and operating system. We focus on these four characteristics because VMs that share these characteristics are readily substitutable. We discuss this further and demonstrate that our analyses are robust to alternative definitions in Appendix E.5.

[24] In some cases, it is possible to downsize to a quarter of the number of cores, i.e., if the peak utilization is 20% and a VM with one-fourth of the number of cores is available. In this case $\omega_{ijt} = 0.25$. Although we omit this category from this section for brevity, as it is rarely observed in the data, we include it in our productivity calculations. See Appendix D.1 for the details of the productivity calculation procedure.

VM cannot be reallocated to another firm by the cloud provider and still consumes 50% of the electricity of a fully utilized machine. Panel (b) is a potentially overprovisioned job. Although the VM was continuously used for the job, it could have also fit on a substitute VM with half the number of cores (hence, generally, half the cost). If such a VM exists, this job would be marked as overprovisioned and have a productivity of 0.5. Finally, panel (c) reflects a properly provisioned job. Although not all of the capacity of the VM is used — indeed, most of the time, the job would have fit on a smaller VM — the peak CPU utilization on the chosen VM goes to around 75%, which means the peak load of the job would not be able to fit on a smaller VM.

As in the example in Table 2, we can aggregate the job-level productivity to an overall productivity for the firm by taking a weighted average across jobs, weighted by the core-hours of each job. The overall productivity level in month m for firm i with jobs J_{im} is:

$$\omega_{im} = \frac{\sum_{j \in J_{im}} \sum_t \omega_{ijt} ch_{ijt}}{\sum_{j \in J_{im}} \sum_t ch_{ijt}} \tag{1}$$

where ch_{ijt} is the core-hours of job j on day t. ω_{im} can be interpreted as the total amount of resources that a perfectly cost-minimizing firm would have used to produce the same output as firm i in month m, divided by the total amount of resources that firm i actually used in month m — in other words, a direct measure of firm i's output divided by firm i's input.[25] Using the same procedure, we also estimate unit-month level productivity for a within-firm analysis and define idleness and overprovisioning productivity measures to decompose overall compute productivity, as described in Appendix D.1.

One potential concern with this method is that there might be structural factors that affect the efficiency of each job. For example, it might be more costly to hit capacity when running an external-facing web server than running an internal-facing analytics model, making the first use case likelier to be overprovisioned. While our measure represents a "physical" measure of productivity in the sense that it is units of output divided by units of input, it may also be desirable to have other productivity measures that control for productivity differences inherent to certain aspects of the job.

To do so, we use fixed-effect regressions to control for the impact of job- and time-specific factors on productivity. Letting ω_{ijt} be the productivity for job j run by firm i that

[25] We also note that, like most single-factor productivity measures, our measure is isomorphic to TFP for firms whose production function is Leontief in computing. A formal derivation is in Appendix C.2.

is active on day t, we estimate the following:

$$\omega_{ijt} = \omega_{im} + Z'_{jt}\beta + \epsilon_{ijt} \tag{2}$$

where Z_{jt} includes job- and time-specific factors such as the type of the machine and the day of the week (utilization tends to be lower on weekends). The resulting firm-month fixed effect ω_{im} is our estimate of firm i's productivity in month m, controlling for the factors in Z_{jt}. We weight by the number of core-hours of each machine on each day; therefore, if Z_{jt} were not included, the resulting estimates of ω_{im} would be numerically equivalent to the core-hour-weighted average outlined above. We also run this using just a firm fixed effect ω_i rather than firm-month fixed effects at the firm level.[26]

While our measure of compute productivity is novel to the economics literature, it is in line with measures used in industry as well as the operating systems literature and bears similarity to other previously developed capacity utilization-based productivity measures. First, cloud providers' definitions of efficiency on the cloud are similar to our own; for example, Microsoft Azure's developer-facing API generates a "resize SKU" recommendation if the 95th percentile CPU of a job would be under 80% on a less expensive VM, nearly identical to our definition of overprovisioning,[27] while AWS and Google Cloud Platform have their own similar definitions of idleness and overprovisioning.[28] Second, the cloud optimization startups described in Section 2 also focus on idleness and overprovisioning; one cloud optimization startup states that the "best practices for optimizing cloud costs" are to "identify underutilized resources, detect idle resources, [and] rightsize cloud resources (select the best type and size of instances)."[29] Third, similar metrics have also been used to measure productivity in the operating systems and IT literature (Weinman, 2011; Islam et al., 2012; Folkerts et al., 2013). Finally, the idea behind our measure — that the extent to which firms utilize the capacity they paid for is a dimension of firms' productivity — has been used to study productivity in myriad other contexts in economics (Hubbard, 2003; Braguinsky et al., 2015; Butters, 2020, for example).[30]

Nevertheless, there are three caveats to point out about our measure. First, we do not

[26] In these fixed effect regressions, we can only compare the fixed effects of firms within a connected set (Abowd et al., 1999; Metcalfe et al., 2023). We find that across these specifications, there is either one connected set or that the largest connected set covers more than 99% of the firms.

[27] Azure— How-to Guides.

[28] AWS— API Reference; Google Cloud— Reduce Overprovisioned Instances; Google Cloud— Identify Idle Instances.

[29] CASTAI— Cloud Cost Optimization.

[30] The key difference between our measure and those previously developed in the literature is that these papers tend to consider capacity as fixed in the short run and load as variable, while in our context, we think of load as fixed and the provisioned computing capacity as variable. Nevertheless, the common source of inefficiency is a mismatch between load and capacity.

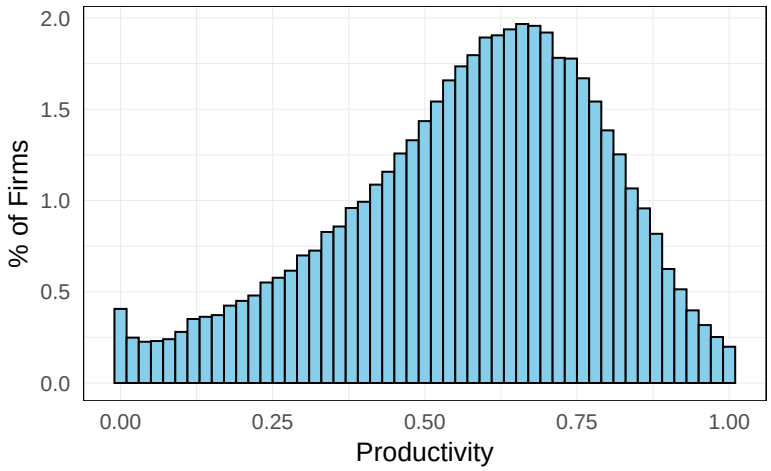
measure all forms of computing inefficiency that could exist. For example, inefficient code can consume excess computing power. While we focus on productivity heterogeneity that comes just from provisioning decisions, it is possible that some properly provisioned jobs based on our measure would be overprovisioned if the code were improved. Therefore, our measure should be viewed as a productivity measure conditional on code efficiency (and all other factors besides the choice of VM). To the extent that provisioning and coding skills are positively correlated, then we will understate the degree of productivity dispersion and the amount of learning. Second, we do not capture the underutilization of other dimensions of the VM (memory and networking) as granularly as we do for CPU utilization. We have limited information on memory and network utilization; in Appendix E.1, we run robustness checks using the information we have. Further, as previously mentioned, inefficiency in terms of CPU utilization tends to be the kind most cited both in the industry and in the literature and has the most implications for counterfactual energy usage and emissions. Finally, our measure does not account for more elaborate efficiency improvements, such as consolidating multiple VMs with 70% peak CPU usage into a smaller number of more fully utilized VMs. While theoretically possible, such improvements may not be practical and would require additional assumptions.

4.2 Comparison of Compute Productivity with TFP

As the productivity literature has overwhelmingly focused on TFP, it is important to discuss the relative strengths and weaknesses of our productivity measure. TFP is typically defined as a residual in production function estimates after accounting for the contributions of measured inputs. Therefore, as the "unexplained" part of the output, it can correspond to many factors such as technology and management, offering limited insight into specific mechanisms (Solow, 1957). Moreover, TFP estimation faces well-documented measurement challenges, including conflation of physical productivity with prices due to using revenue data (Foster et al., 2008), reliance on aggregated inputs (Orr, 2022), measurement errors (Collard-Wexler and De Loecker, 2016), and lack of high-frequency data.

While narrower in scope compared to TFP, our productivity measure is attributable to a single input with a clear interpretation. It is a physical productivity measure that can quantify the productivity of actual physical resources used in firm production, such as compute resources and electricity. Additionally, by observing productivity at the minimal unit of production (VM), we can study the precise mechanism underlying efficiencies. These advantages make it possible to study aspects of productivity that are hard to analyze using TFP, thereby allowing us to complement existing evidence on firm productivity.

Figure 3: Dispersion of Firm Compute Productivity



Notes: This figure shows the distribution of firm-level compute productivity, estimated using the entire sample, weighting each VM by its core hour as shown in Equation (1). The x-axis represents productivity levels ranging from 0 to 1, while the y-axis shows the percentage of firms. Each observation corresponds to a firm, and the histogram bars reflect the unweighted distribution of firms across different productivity intervals. The corresponding figures conditional on different sets of control variables are reported in Figure OA-11.

5 Empirical Facts on Compute Productivity

In this section, we first present findings on the dispersion and persistence of compute productivity and compare them with the existing productivity literature. Then, we analyze the role of within-firm dispersion in productivity heterogeneity.

5.1 Dispersion and Persistence of Compute Productivity

Our first set of empirical results documents significant heterogeneity in compute productivity across firms. In Figure 3, we plot the raw histogram of all firms' compute productivity estimated on the entire sample, and Table 3 reports several statistics. Unlike traditional productivity measures, our metric ranges from zero to one, zero indicating consistently idle VMs and one indicating optimal VM provisioning with no idleness or overprovisioning. First, we observe that the median firm has a productivity of 0.63, meaning that it utilizes only around 60% of the provisioned resources productively. However, there is substantial dispersion in compute productivity. Firms in the 90th percentile of the productivity distribution are 3.2 times more efficient than those in the 10th percentile. This large dispersion highlights that some firms are more capable of utilizing compute resources effectively.

We next analyze productivity dispersion within firms. While there is recent literature

Table 3: Dispersion and Persistence of Compute Productivity

	No Control (1)	Industry (2)	Time (3)	Industry/Time (4)
	Panel A. Dispersion			
Dispersion:				
Mean	0.602	-	-	-
Median	0.626	-	-	-
10-90th perc ratio	3.186	3.204	3.248	3.274
Inter Quartile Range	1.669	1.673	1.679	1.687
R^2	0.000	0.011	0.003	0.016
Within-Firm Decomp. (%):				
Between-firm	28.152	27.806	39.140	39.002
Within-firm	71.848	72.194	60.860	60.998
Within-Firm-Between-Region Decomp. (%):				
Between-region	5.599	5.717	17.764	8.643
Within-region	94.401	94.283	82.236	91.357
	Panel B. Persistence (AR(1) Coefficients)			
1-month persistence:				
Productivity	0.933	0.932	0.933	0.933
	(0.000)	(0.001)	(0.000)	(0.001)
Idleness Productivity	0.934	0.933	0.934	0.932
	(0.000)	(0.001)	(0.000)	(0.001)
Overprovisioning Productivity	0.911	0.916	0.908	0.912
	(0.000)	(0.001)	(0.000)	(0.001)
1-year persistence:				
Productivity	0.645	0.632	0.646	0.633
	(0.002)	(0.003)	(0.002)	(0.003)
Idleness Productivity	0.659	0.642	0.658	0.637
	(0.002)	(0.003)	(0.002)	(0.003)
Overprovisioning Productivity	0.599	0.595	0.563	0.558
	(0.002)	(0.003)	(0.003)	(0.003)
5-years persistence:				
Productivity	0.321	0.301	0.322	0.302
	(0.007)	(0.008)	(0.007)	(0.008)
Idleness Productivity	0.332	0.313	0.333	0.313
	(0.007)	(0.008)	(0.007)	(0.008)
Overprovisioning Productivity	0.104	0.100	0.104	0.101
	(0.006)	(0.007)	(0.006)	(0.007)

Notes: This table reports the dispersion and persistence of productivity measures with different sets of controls. Panel A presents statistics on the distribution of productivity, as well as the composition of productivity dispersion. Panel B shows the persistence of productivity measures with 1-month, 1-year, and 5-year AR(1) coefficients, with standard errors in parentheses. Productivity measures are obtained using Equation (1) separately for overall productivity, idleness, and overprovisioning productivity. Further details of the estimation and visualization of persistence are provided in Appendix D.2 and in Figure OA-5. Standard errors clustered at the firm level are in parentheses.

on within-firm productivity dispersion, most studies are limited to either a single firm or a single productivity measure in one industry. Table 3 presents the results of a within- and between-firm productivity decomposition. We find that within-firm dispersion explains close to 80% of total productivity dispersion in the economy, suggesting that different units in the same firm can exhibit large compute productivity heterogeneity. This large within-firm heterogeneity is consistent with other within-firm heterogeneity results in the literature (Kehrig and Vincent, 2019; Orr, 2022; Haltiwanger et al., 2022).

Using our data, we can go beyond unit-level productivity dispersion and analyze within- versus between-region decomposition; in other words, whether the units of the same firm in the same region (US, EU, and others) have similar productivity levels to other units. This decomposition analyzes only multinational firms that have units in different geographies. We see that region explains around 6% of productivity dispersion. Although this percentage is small, it still demonstrates that the region of a unit, even within the same firm, accounts for a non-negligible share of unit productivity.

We next examine the persistence of productivity over time. Panel B of Table 3 reports AR(1) coefficients across different specifications for 1-month, 1-year, and 5-year horizons. We find large persistence in short-term productivity: AR(1) coefficients are 0.933 and 0.645 for 1-month and 1-year, respectively. While this persistence declines over longer horizons, it remains significant, with a 5-year AR(1) coefficient of 0.321. We also find that both idleness and overprovisioning are persistent in all time horizons, although overprovisioning exhibits less persistence than idleness.

It is important to examine whether simple observable factors can explain the productivity dispersion we observe. In Columns (2-4) of Table 3, we present the same results after controlling for industry, time, and industry-by-time fixed effects. We find that observable factors explain only a small fraction of the variation in firm-level productivity, giving consistent results across different control specifications.[31] These results point to the role of unobserved heterogeneity in compute productivity, again mirroring common findings in the productivity literature (Fox and Smeets, 2011).[32]

Dispersion and persistence of firm productivity are some of the canonical results in the productivity literature. Many studies have documented similar levels of dispersion across firms, primarily relying on TFP estimates from manufacturing industries. Syverson (2004) found that the average 90-10th percentile ratio in the US manufacturing sector is 2.45, which is slightly smaller than our estimates. Other estimates from the literature are

[31] We report the productivity dispersion by industry in Figure OA-3.

[32] Murciano-Goroff et al. (2024) find similar evidence in the digital economy by showing that observable characteristics explain little variation in firms' propensity to use software with known vulnerabilities.

Table 4: Productivity Differences Across Firm Characteristics

	Overall Prod. (1)	Idleness Prod. (2)	Overprov Prod. (3)
	Panel A: Firm Size (relative to 1st quartile)		
2nd quartile	0.010	-0.009	0.061
	(0.003)	(0.002)	(0.005)
3rd quartile	0.029	-0.007	0.127
	(0.003)	(0.002)	(0.006)
4th quartile	-0.037	-0.060	0.075
	(0.003)	(0.002)	(0.005)
	Panel B: Region (relative to EU)		
US firms	0.052	0.049	0.033
	(0.005)	(0.003)	(0.002)
Industry FE	X	X	X
Time FE	X	X	X

Notes: This table reports the productivity differences across firm characteristics. Panel A presents the average productivity estimates for firms in different size quartiles relative to the 1st quartile, expressed in percentage terms. Panel B shows the differences between US firms and EU firms. Columns (1) through (3) report the results for overall productivity, idleness productivity, and overprovisioning productivity, respectively. Each coefficient is estimated from a firm-month level regression, where the outcome variable is the level of productivity specified in columns, with the control variables specified in the bottom panel table. The construction of firm-month level productivity estimates is described in Section 4 and Appendix D.1. Standard errors clustered at the firm level are reported in parentheses.

an interquartile range of 1.76 in US retail (Foster et al., 2006) and a 90-10 ratio of 5 in Chinese and Indian manufacturing industries (Hsieh and Klenow, 2009). There are also estimates of productivity dispersion focusing on a specific input like us. For example, Fox and Smeets (2011) measures labor productivity using Danish matched employer-employee data and finds a 90-10 ratio of 3.36, while Davis et al. (2008) reports a 90-10 ratio of 7.3 in electricity productivity in the US Census. Overall, our findings on productivity and its persistence align with the broad evidence of productivity differences in the literature and complement them by showing that persistent productivity differences continue to exist in the digital economy.

5.2 Productivity Differences by Firm Characteristics

In this section, we analyze how productivity varies with important firm characteristics such as firm size and region.

There is no clear ex-ante relationship between firm size and compute productivity based on our discussions of drivers of compute productivity in Section 2.4. While larger firms may have more resources for complementary investments and skill development,

they might also face greater organizational frictions. Our findings, presented in Panel A of Table 4, confirm this intuition. We see no clear pattern in the relationship between firm size and productivity. Firms in the 2th and 3rd quartiles of the size distribution are slightly more productive than the smallest firms, while the largest firms actually exhibit a lower productivity level than the smallest firms. However, when examining the components of productivity, we observe that firms in the largest quartile exhibit 6.0% lower idleness productivity but 7.5% higher overprovisioning productivity relative to those in the smallest quartile. This pattern may reflect greater organizational frictions and monitoring challenges in larger firms, potentially resulting in more idle machines, while their expertise in IT leads to higher overprovisioning productivity.

In Panel B, we report the productivity differences between US and EU firms. We find that US firms are consistently more productive than EU firms, showing a 3% to 5% higher productivity across all categories. This result is consistent with the findings of Bloom et al. (2012), which shows that American firms are better at utilizing IT.

5.3 Robustness Checks and Ruling Out Alternative Explanations

Several alternative explanations could potentially explain the observed dispersion in compute productivity. The most important ones are demand volatility and risk aversion: seemingly inefficient firms may purposefully maintain idle and over-provisioned VMs because they are worried about rare usage spikes that do not materialize. Although we think there are many tools to help firms automatically scale their capacity up and down discussed in Section 2, and not fully taking advantage of these tools should be viewed as inefficiency in and of itself, we conduct two exercises to show that this concern does not explain our results. First, in Appendix E.4, we estimate various measures of firm load volatility and show that they explain less than 1% of the variance in firm productivity. Second, we estimate the probability of firms hitting VM capacity (stockouts) and show that high-productivity firms do not have higher stockout rates than low-productive firms.

A second explanation for productivity dispersion could be that firms use computing for different purposes, which could inherently have different productivity patterns and explain the dispersion in firm productivity. While we have shown that industry has little explanatory power, we can go further by using machine types to control for specific use cases, as different use cases are often represented by observable job characteristics, such as machine type or memory requirements. In Appendix E.3, we estimate firm productivity by including machine characteristics such as memory and machine type, as well as more granular VM configuration IDs. We find that while these factors have some explanatory

power in explaining productivity, they account for only a small portion of the variation.[33]

Beyond these empirical tests, many of our results in the paper suggest that simple mechanical relationships are not driving productivity estimates. The empirical facts about compute productivity are primarily consistent with the findings of the broader productivity literature. Moreover, we showed that productivity dispersion exists conditional on firm and even within-firm unit — to the extent that jobs in the same unit within the same firm tend to be for more similar use cases; this demonstrates that the use case itself does not drive the dispersion we observe in the data. Finally, as we will discuss in Section 7, there is significant short-term learning at the firm and unit level, suggesting that productivity is not simply explained by firm-level characteristics.

Our other robustness checks, described in Appendix E and reported in Appendix H, demonstrate the robustness and external validity of our results. First, we repeat our analysis with different downsizability definitions and peak utilization calculations over a period of 1, 3, and 15 days and find that our results are robust to these different choices. Second, we find that firms with below-median productivity are 60.0% more likely to exit the cloud than those with above-median productivity, again similar to other results in the productivity literature (Foster et al., 2016). Third, we analyze utilization in VMs along other dimensions, such as memory and network utilization, and find that these different measures of utilization are positively correlated with CPU utilization, meaning jobs we identify as inefficient in terms of CPU utilization are also likelier to be inefficient in terms of memory or network utilization. Fourth, we show that idleness and overprovisioning productivity, which are potentially generated by different mechanisms, are positively correlated. Finally, we analyze dispersion in compute productivity using publicly available datasets from Google Cloud and Microsoft Azure and find similar levels of dispersion in those datasets as shown above.[34]

6 Mechanisms: What Do More Productive Firms Do?

We have now seen that firms differ dramatically in the efficiency with which they use VMs, that these differences are persistent, and that they are not simply explained by observables. However, the question remains: What are the specific actions being taken

[33] As reported in Figure OA-1, machine type and memory explain 4.5% and 7% of the variation, respectively. Even controlling for the precise configuration of a VM explains only 24.1% of the observed variation in productivity. This suggests that approximately three-quarters of the variation in productivity occurs among firms that use identical VMs.

[34] Another potential concern is that we have a selected sample of firms, specifically the customers of our data provider. However, the relevant population for this study is the sample of firms that use cloud computing. Conditional on this sample, our understanding of the cloud industry suggests minimal selection bias, as cloud providers offer largely similar services, which we also confirm with publicly available datasets.

by high-productivity firms that make them more efficient users of computing? Our data allows us to observe both the computing load firms face and their high-frequency VM allocation choices, enabling us to identify some of the factors that distinguish highly productive firms from their less efficient counterparts.

In this section, we analyze three factors: responsiveness to changing demand conditions, the variety of VM types used by firms, and the propensity to turn off idle VMs. These mechanisms are not only inherently interesting but also shed light on the drivers of productivity discussed in Section 2.4. Better responsiveness to demand and more appropriate VM choices likely indicate a firm's ability to predict demand accurately and adjust VM quantity/type in response to changing conditions. Additionally, detecting and shutting down idle VMs may reflect better monitoring mechanisms within the organization. To study these factors, we perform an out-of-sample exercise in which we first classify firms as 'high' or 'low' productivity based on their productivity in 2022 relative to their industry's median productivity and then analyze the differences between them in 2023.

6.1 Responsiveness to Demand Changes

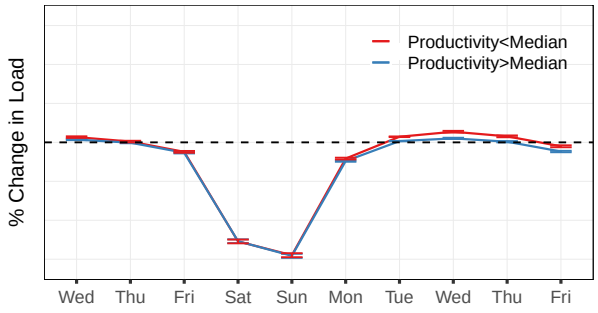
To analyze the differences in responses to demand changes between high and low-productivity firms, we employ an event study approach. Our regression model is specified as follows:

$$\log(y_{it}) = \sum_{k=-4}^{5} \beta_k^H D_{ik}^H + \sum_{k=-4}^{5} \beta_k^L D_{ik}^L + \alpha_i + \gamma_t + \epsilon_{it} \tag{3}$$

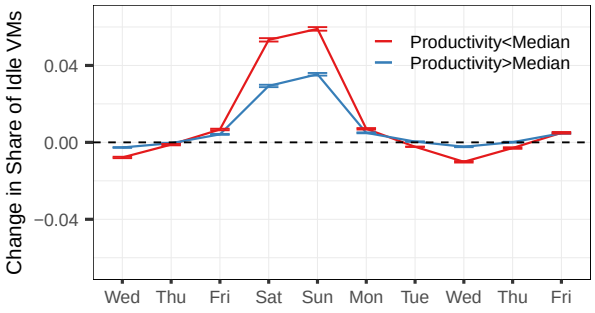
In this equation, y_{it} represents the outcome variable for firm i on day t. We include firm fixed effects (α_i) to account for time-invariant firm characteristics and day fixed effects (γ_t) to control for any overall time trends or seasonality. The key variables of interest are two sets of dummy variables, D_{ik}^H and D_{ik}^L, which represent each day from Tuesday to the following Friday in a two-week cycle for high and low-productivity firms, respectively. The coefficients β_k^H and β_k^L capture the day-of-the-week effects for each group. This specification allows us to compare how high and low-productivity firms adjust their compute resources over the course of a week while controlling for firm-specific and time-specific factors. We estimate this regression for three outcome variables: total compute load, the share of idle VMs, and the share of total provisioned resources.

Figure 4 plots the coefficients separately for high-productivity and low-productivity firms. First, in Figure 4(a), we show that high- and low-productivity firms face nearly identical shifts in demand on the weekends. With this in mind, Figures 4(b) and 4(c) then plot how these two groups change compute inputs in response to this load reduction.

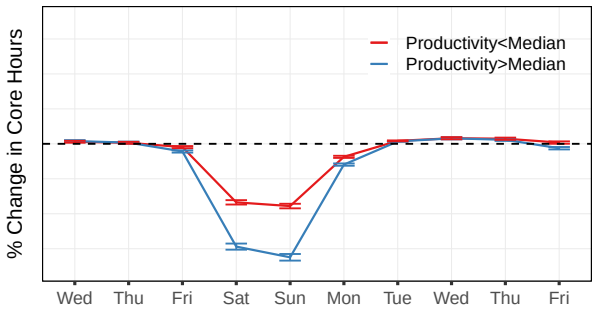
Figure 4: Firm Responses to Weekend Demand by Productivity Level



(a) Change in Load



(b) Share of Idle Machines



(c) Change in Provisioned Resources

Notes: These figures show coefficient estimates from equation (3) with three outcome variables listed in the text, reporting the percentage changes in compute load, the share of idle VMs, and provisioned resources for firms above and below the median productivity level. The productivity levels are estimated using the 2022 sample, whereas the regressions are estimated using the 2023 sample. The estimates are calculated using firm-day level data, constructed using an analog of equation (1) for daily frequency controlling for firm-fixed effects and day-fixed effects. Error bars represent 95% confidence intervals, clustered at the firm level. The y-axis of Panel (a) and (c) are obfuscated for confidentiality reasons.

We find that high-productivity firms respond to weekend-induced demand fluctuations better than low-productivity firms for both idleness and provisioning. On weekends, low-productivity firms see an average 6% increase in idle VMs, compared to only a 3% increase for high-productivity firms. Figure 4(c) shows a very similar pattern for total provisioned VMs, with high-productivity firms reducing total compute resources significantly more than low-productivity firms.[35]

This analysis suggests that firms' ability to anticipate and adapt to demand fluctuations significantly influences their productivity. This finding not only informs our compute productivity analysis but also, to the best of our knowledge, provides the first large-scale

[35]One potential concern with our interpretation could be that not all load changes are demand-driven, as employees might use compute resources for other purposes, such as product development. To address this concern, we repeat this exercise, focusing on software firms whose compute loads are more likely to be driven by customer demand. We find similar results for this subset of firms.

Table 5: Usage Patterns of More Versus Less Productive Firms

Dependent variable	prod >median	% of mean	prod >median	% of mean
HHI of usage across VM series	-0.095	-16.0	-0.068	-11.4
	(0.002)		(0.002)	
1(all usage on one VM series)	-0.119	-54.9	-0.093	-42.9
	(0.002)		(0.002)	
Number of VM series used	1.291	29.2	0.785	17.8
	(0.029)		(0.026)	
Cohort quarter/firm size FE			X	X
Industry/region FE			X	X

Notes: All rows display the coefficient of a regression of the dependent variable on an indicator for whether the firm is more productive than the median firm, along with the ratio between the coefficient and the mean of the dependent variable. The left set of columns includes the raw difference between the groups, while the right set controls for sign-up cohort quarter fixed effects, industry (2-digit SIC code) fixed effects, region fixed effects, and firm size quartile fixed effects. Productivity levels are estimated using the 2022 sample, while the regressions are estimated using the 2023 sample. Standard errors clustered at the firm level are in parentheses.

evidence that firms differ in their abilities to adjust to high-frequency changes in market conditions. In doing so, we contribute to the broader literature on firms' responsiveness to various shocks (Pozzi and Schivardi, 2016; David and Joseph, 2017; Cooper et al., 2024).

6.2 Usage of a Variety of VM Types

As discussed above, firms have access to many different types of VMs. Each VM type is specialized for different workloads, and a firm's selection of various machine types likely indicates its level of sophistication in cloud computing resource allocation. With this idea in mind, we investigate the machine choices of high and low-productivity firms and test whether more productive firms tend to use a wider array of more specialized VM series. To do so, we estimate the following at the firm level:

$$y_i = \beta D_i^H + Z_i'\gamma + \epsilon_i \tag{4}$$

The outcome variables, y_i, include measures of dispersion of firms' usage on different machine series, including a machine series HHI (the sum of the squared usage shares on each machine series for each firm), an indicator for whether the firm only uses one machine series, and the number of machine series the firm uses. The coefficient of interest, β, multiplies an indicator for whether the firm's productivity is higher than the median productivity in its industry. Finally, Z_i includes firm-level controls such as firm size,

industry, region, and cohort quarter fixed effects.

Table 5 displays the results. We consistently find that more productive firms tend to spread out their usage across a wider array of VM series. This is true both unconditionally and controlling for firm-level covariates. For example, we find that high-productivity firms have a VM series HHI that is 0.068 lower than low-productivity firms — more than 11% of the mean VM series HHI across firms — indicating that more productive spread their usage out among a wider array of VM types. They also use more VM series and are drastically less likely to put all their usage on one VM series. These results suggest important differences in the sophistication between high- and low-productivity firms; high-productivity firms are more aware of the full menu of VMs offered by cloud providers and are better able to take advantage of specialized VM series on different jobs.

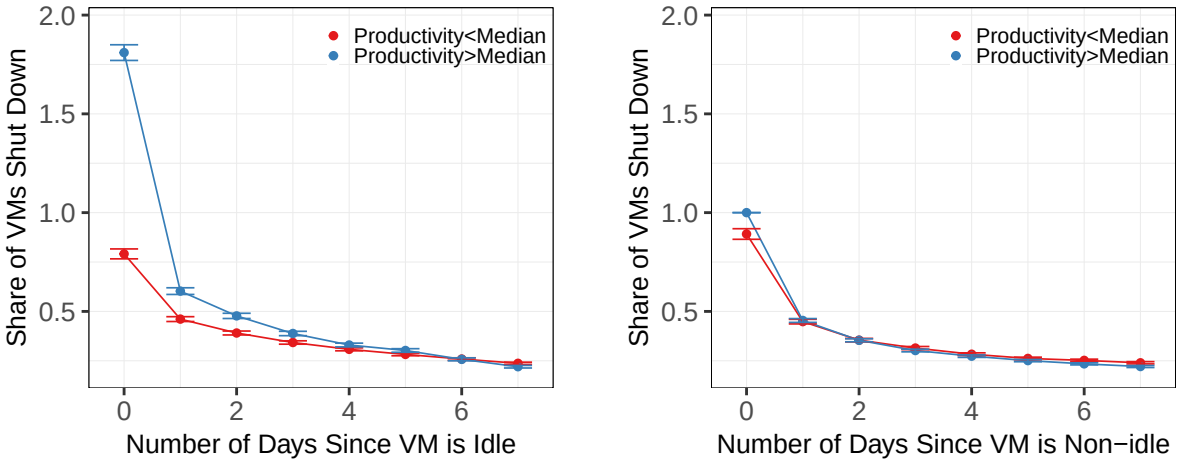
6.3 Turning Off Idle Machines

One factor plausibly contributing to dispersion in compute productivity is a difference in monitoring capabilities. When there are mistakes and resources are left idle, which can happen at both high- and low-productivity firms, economic theory would predict that firms that are better at monitoring — either because they can better align the incentives of software engineers and managers or because managers can better observe and control the actions of engineers — should be better able to catch idle VMs and shut them down before they use up too many resources.

To investigate this, we examine the probability and speed with which low- and high-productivity firms detect and are able to shut down idle VMs. Figure 5(a) displays the probability that a VM that is idle today is shut down today, tomorrow, or on any day out of the next week, broken out by whether the firm is more or less productive than the median, only including VMs of longer than one day. To provide a comparison, Figure 5(b) plots the probability that a VM that is *not* idle today is shut down for both groups. The probability that a high productivity firm shuts down a non-idle VM on day 0 is normalized to 1. Overall, while high- and low-productivity firms have a remarkably similar propensity to shut down non-idle VMs, high-productivity firms are more than twice as likely as low-productivity firms to identify and shut down an idle VM on the day it becomes idle. Even if the VM is not shut down on that day, high-productivity firms are more likely to detect it and shut it down on any given day within the next week. In fact, less productive firms are slightly *less* likely to shut down an idle VM than a non-idle VM, suggesting monitoring problems inhibiting the ability to address idle resources.

In summary, we take the results of this section as suggestive of important differences in

Figure 5: Shutdown Probabilities of Idle VMs by Productivity Level



(a) Idle Machines **(b)** Non-Idle Machines

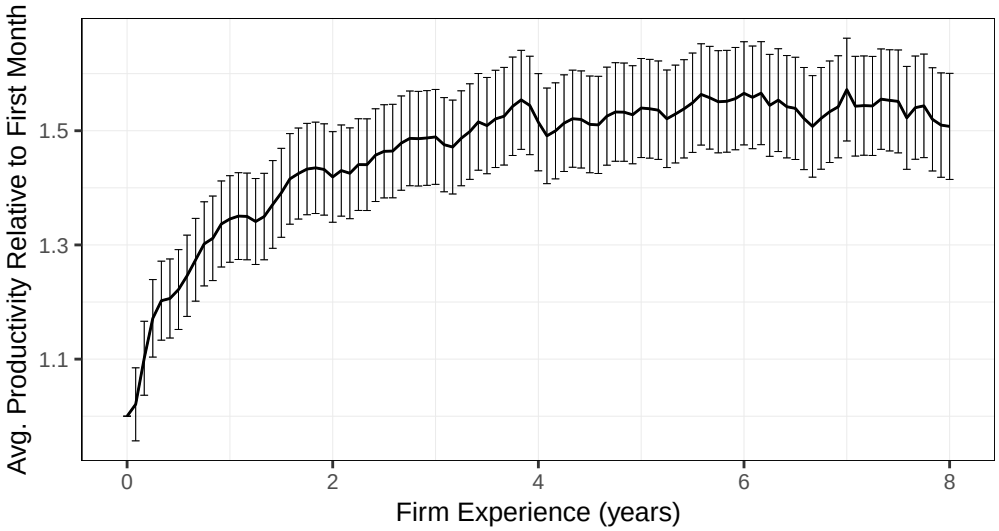
Notes: The left panel represents the probability that an idle VM is shut down in a given number of days, while the right panel represents the probability that a non-idle VM is shut down in a given number of days. The red lines represent less productive firms, while the blue lines represent more productive firms. The shutdown probability of non-idle machines on day 0 for more productive firms is normalized to 1. The productivity levels are estimated using 2022 data, while the probability of shutting down idle and non-idle machines is estimated using 2023 data. The crossbars are 95% confidence intervals, with standard errors clustered by firm. Only includes VMs that last longer than one day and that end before the end of our sample period.

the behavior of more and less productive firms. The more productive firms utilize fewer resources per unit of computing output both in a steady state and in response to changes in demand, use a wider array of more specialized machines, and are better able to monitor and shut down idle resources. Although we will return to exploring the magnitude of these real-world impacts in more detail in Section 8, we now move to discuss one potential *cause* of the dispersion we see here: firms learning to use new technology.

7 Learning: How Firm Productivity Changes Over Time

In this section, we investigate the role of learning: firms becoming more productive as they use the cloud more. There is reason to believe that learning could play a role in explaining productivity dispersion *ex-ante*. Cloud computing is a relatively new technology, and new technologies take time to diffuse with potential learning processes (Arrow, 1962; Rosen, 1972; Chari and Hopenhayn, 1991). It is, however, unclear *ex-ante* how quickly firms learn and what they do to improve their productivity. Do they shift resources to more productive use cases? Do they experiment with different kinds of products and find the product that best fits the use case? Or do they simply learn to use the products they already use more productively?

Figure 6: Productivity Against Firm Experience in July-September 2022



Notes: This figure illustrates the average productivity level as a function of firm experience, measured in years since the firm first began using cloud services. The productivity level in the initial month (month 0) is normalized to 1. The figure reports estimates without controls. The crossbars indicate the 95% confidence intervals. The analysis is based on data from July-September 2022.

To answer these questions, we present two sets of results. First, we document that firms indeed learn to be more productive over time — there is a strong relationship between a firm's overall productivity and its experience using the cloud. This relationship holds both cross-sectionally (at a given point in time, more experienced firms are more productive) and over time (cohorts of firms that adopt the cloud at the same time get better over time).

Second, we demonstrate *how* firms learn. We decompose firms' productivity growth and find that just about all firm-level productivity growth is from units within the firm getting more productive, with a significantly smaller amount coming from allocative efficiency — firms reducing the usage of less productive units. We further decompose this within-unit productivity growth to see how units become more productive through their choice of machines. While much of the within-unit productivity growth occurs within the same type of machine, suggesting that firms do better at provisioning the kind of machine they originally chose, we find that units are constantly trying new kinds of VMs, ramping up the usage of those that perform well, and stopping to use those that perform poorly, suggesting that experimentation also plays a role in explaining why firms learn.

7.1 Learning at the Firm Level

At any given point in time, more experienced firms are substantially more productive than less experienced firms. Figure 6 plots the length of time each firm has used the

cloud as of one quarter, July through September 2022, against productivity in that quarter. The average productivity of firms in their first month of usage is normalized to 1. The difference between new firms and firms that have used the cloud for just one year is stark: one-year-old firms are roughly 40% more productive than brand-new firms. Older firms still tend to be more productive than newer firms, but the slope levels out after that first year. Four-year-old firms are roughly 55% more productive than brand-new firms, and receive fairly minimal productivity gains after that. The same patterns hold when controlling for firm- and job-specific factors, as shown in Appendix H.

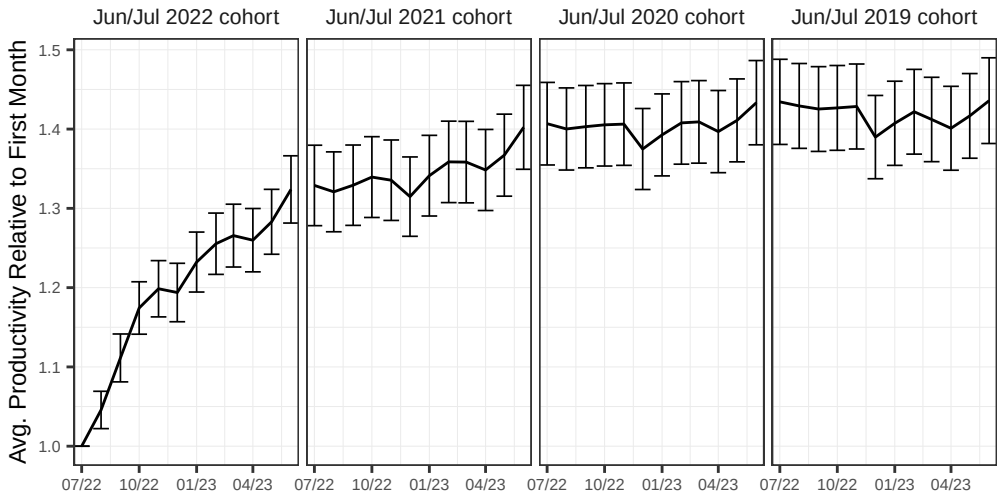
While suggestive, this cross-sectional relationship between experience and productivity does not itself establish the existence of learning. We might be worried about two forms of selection. First, there might be selection based on cloud adoption: firms that adopted the cloud earlier did so because they knew they would be more productive. Second, there might be survivorship bias: if the least productive firms tend to exit, then the most experienced firms will tend to be more productive on average, regardless of whether learning occurs.

To control for both forms of selection, we focus on the productivity growth of individual cohorts over time, conditional on survival to the end of our sample. Figure 7 plots the productivity over the second half of 2022 and the first half of 2023 for four cohorts of firms: those that adopted the cloud in June-July 2022, 2021, 2020, and 2019. We find that firms' productivity dynamics over time exhibit extremely similar patterns to the patterns across firms at a point in time.[36] Firms that started in the middle of 2022 had their productivity increase by more than 30% in their first year of usage. Their productivity at the end of their first year was similar to the productivity of the 2021 cohort at the beginning of their second year, suggesting that productivity dynamics are not largely driven by differences in cohort adoption.

Over the next several years, the pace of learning slows down substantially. Nevertheless, the cohorts still exhibit substantially similar learning dynamics, as the productivity of each cohort at the end of the year is nearly identical to the productivity of the cohort from the year prior at the start of the year. Learning eventually plateaus at around 40-50% more productive than the first month. The flat productivity level of older cohorts also suggests that learning is not driven by aggregate productivity trends in the cloud industry; if this

[36]Figure 7 still incorporates some cross-sectional variation in productivity across different cohorts. Using our limited data from 2017-2019, we are able to characterize learning for the cohort of firms that started using the cloud in 2017 over six years in a way that only uses within-firm variation over time. This analysis is limited by the sparsity of our data; in addition the development and introduction of a wider variety of more powerful VMs over the six-year time horizon may also bias these results. Nevertheless, we still find clear evidence of learning among this cohort of firms. See Figure OA-6 in Appendix C.

Figure 7: Productivity by Cohort Over Time



Notes: This figure displays productivity estimates for different cohorts during the period between July 2022 and June 2023. It reports estimates for four distinct cohorts, each cohort reflecting productivity relative to their experience levels: 0-1, 1-2, 2-3, and 3-4 years. The average productivity for the June-July 2022 cohort in July 2022 is normalized to 1. To be included in the analysis, a firm must have had nonzero usage every month from July 2022 to June 2023. Error bars indicate the 95% confidence intervals, with standard errors clustered by firm. Only includes a balanced panel of firms that used an average of at least 50 VM-days per month.

were the case, then older cohorts would be getting more productive, too.

We can decompose learning into initial productivity to investigate the extent to which aggregate learning comes from more or less initially productive firms. Table 6 breaks out the June-July 2022 cohort's productivity in each quarter by the productivity quintile in the first quarter, along with the overall average productivity and the ratio between the 90th and 10th percentile productivity in each quarter. Table 6 demonstrates that learning is primarily driven by the bottom of the distribution, improving their productivity relative to the top, which reduces the dispersion of the overall productivity distribution over time. Indeed, while the top quintiles are substantially more productive than the bottom quintiles initially, productivity is relatively flat for the top two quintiles (after an initial drop-off for the top quintile, which could be attributed to mean reversion after a favorable initial draw) but increases significantly for the less initially productive firms. However, this is not a full reversion to the mean; the firms that are the most productive initially remain the most productive after a year. These results indicate that productivity dispersion declines substantially over the first year, as the 90-10 ratio declines from 35.3 to 7.9 within one year.

While the results of the previous subsection establish that new firms increase their productivity over time, they do not shed light on what exactly firms do to learn. In the next two subsections, we investigate how firms learn.

Table 6: Short-Term Learning Average Productivity by Quintile

	Productivity Quintile in 2022Q3						
Quarter	1	2	3	4	5	Full Sample	90-10 ratio
2022Q3	0.09	0.56	1.00	1.38	1.97	1.00	35.3
2022Q4	0.41	0.82	1.13	1.46	1.85	1.13	10.3
2023Q1	0.56	0.92	1.21	1.49	1.77	1.19	8.2
2023Q2	0.65	0.99	1.24	1.50	1.71	1.22	7.9

Notes: This table reports the average productivity values for each quintile and the full sample at quarterly intervals. The "Full Sample" column represents the average across all quintiles for each quarter. The average productivity for the full sample in 2022Q3 is normalized to 1. Only includes a balanced panel of firms that entered in June-July 2022 and used an average of at least 50 VM-days per month.

7.2 Decomposing Within-Firm Learning

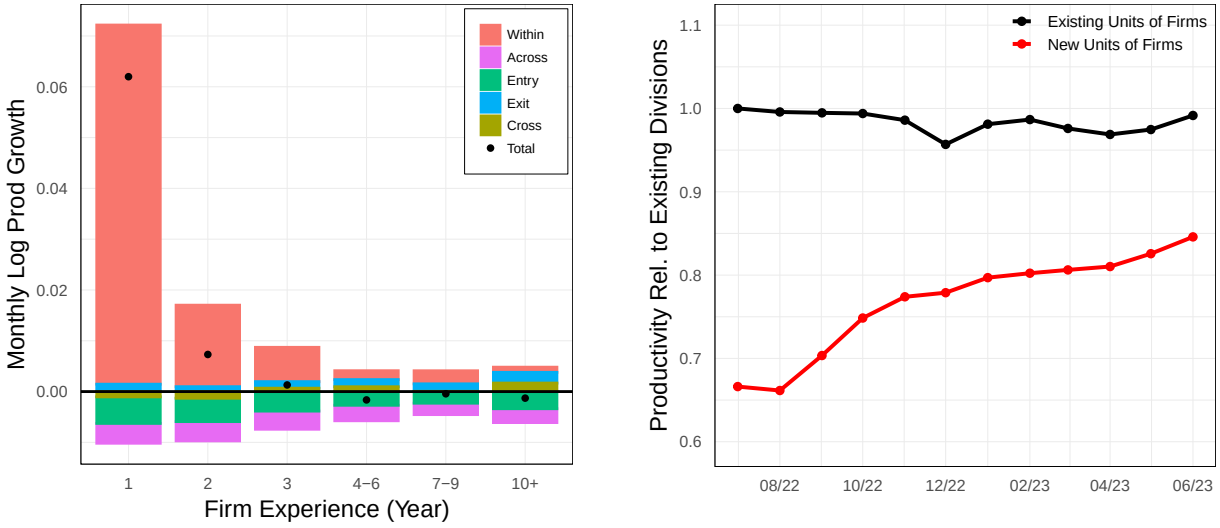
We begin by studying whether learning happens across or within units. On one hand, learning might be driven by firms devoting more resources to more productive units (across-unit learning); on the other, units themselves might be getting more productive (within-unit learning). To quantify these mechanisms, we decompose monthly productivity growth at the firm level using the method outlined in Foster et al. (2001). Firm i has units K_{im} in month m; each unit $k \in K_{im}$ has monthly productivity ω_{km}. We can write firm i's productivity in month m as the core-hour weighted average productivity of each of its units; that is, letting s_{km} be the core-hour share of unit k in month m,

$$\omega_{im} = \sum_{k \in K_{im}} s_{km} \omega_{km}$$

We are interested in decomposing $\Delta\omega_{im} = \omega_{im} - \omega_{i(m-1)}$, the change in productivity for firm i from month $m-1$ to month m. For ease of notation, label $m = 1$. Let $S_{i1} = K_{i0} \cap K_{i1}$ be the set of units in the firm that used the cloud in both month 0 and month 1; $X_{i1} = K_{i0} \setminus K_{i1}$ the set of units that stopped using the cloud in month 0; and $E_{i1} = K_{i1} \setminus K_{i0}$ the set of units that started using the cloud in month 1. We can decompose $\Delta\omega_{i1}$ as follows:

$$\begin{aligned} \Delta\omega_{i1} = & \overbrace{\sum_{k \in S_{i1}} s_{k0}(\omega_{k1} - \omega_{k0})}^{\text{=Within}} + \overbrace{\sum_{k \in S_{i1}} (s_{k1} - s_{k0})(\omega_{k0} - \omega_{i0})}^{\text{=Across}} + \overbrace{\sum_{k \in S_{i1}} (s_{k1} - s_{k0})(\omega_{k1} - \omega_{k0})}^{\text{=Cross}} \\ & + \underbrace{\sum_{k \in E_{i1}} s_{k1}(\omega_{k1} - \omega_{i0})}_{\text{=Entry}} + \underbrace{\sum_{k \in X_{i1}} s_{k0}(\omega_{i0} - \omega_{k0})}_{\text{=Exit}} \end{aligned} \tag{5}$$

Figure 8: Decomposition of Firm Learning Within and Across Units



(a) Within-firm Learning Decomposition

(b) New vs Existing Unit Productivity

Notes: Panel (a) figure presents the decomposition of monthly log productivity growth by firms' years of experience in cloud computing, displaying five components: within-firm, across-firm, entry, exit, and cross effects shown in Equation (5). The x-axis represents the firm's cloud experience in years, while the y-axis shows the monthly log productivity growth. The black dots indicate the average month-to-month productivity growth at the firm-level, whereas each bar represents a component of the decomposition. Panel (b) plots the average productivity of the unit(s) that joined in June-July 2022 against the average productivity of the older units in the same firm. The productivity of existing units in July 2022 is normalized to 1. The details of the estimation procedure are provided in Appendix D.4.

The first term of the decomposition, Within, reflects the productivity growth coming from within-unit learning. The second term, Across, reflects the productivity growth from the compositional effect of the reallocation of resources across units. It would be negative if the units whose resource share increased were less productive than the firm's average productivity, $\omega_{k0} < \omega_{i0}$. The third term, Cross, represents the correlation between within-firm productivity growth and within-firm resource share growth. The fourth term, Entry, represents the contribution of units that are new to the cloud and would be positive if entering units were more productive than the firm average. Finally, Exit reflects the contribution of units that stop using the cloud and would be positive if exiting units are less productive than average.

Fixing two months, we take this decomposition for each firm and then average the terms across firms, separated by the experience group of each firm. Figure 8(a) plots these average decomposition terms by firms' years of experience in cloud computing. The black dots represent the overall average monthly productivity growth for firms of each experience group. From this, we see that the flattening-out of firm productivity

over time masks substantial dynamics in firm productivity. In the first year, the within component is positive, and the across component is negative, suggesting that units within firms themselves learn rather than firms learning how better to allocate resources across units. The negative across term is likely due to newer units having greater month-to-month usage growth while being less productive. The entry term remains negative regardless of how experienced the firm is, meaning that units that join the cloud tend to be less productive than the firm. This suggests that firms cannot fully transmit whatever one unit learns about using the cloud to other units. Finally, the exit term is positive but small — relatively few units stop using the cloud once they start using it, but those that do tend to be less productive than the average unit within their firm.

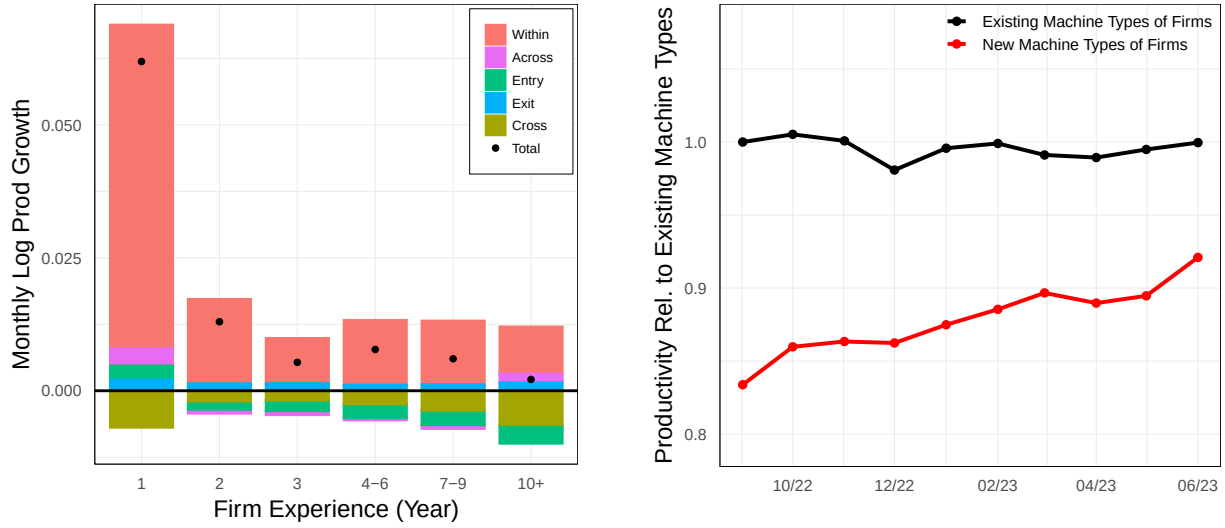
Other patterns in the data confirm that much of the firm-level learning happens within units. In Figure 8(b), we plot the productivity of new units in the firm relative to existing units, normalizing the productivity of existing units in July 2022 to zero. New units start out with 35% lower productivity than other units in the same firm but close roughly two-thirds of that gap in their first year. This is similar in magnitude to the overall pace of firm-level learning shown in Figures 6 and 7, suggesting that new units of experienced firms do not seem to learn any differently than new firms. In Figure OA-2 reported in the Appendix, we examine within-firm knowledge transfer more by looking at whether new units at experienced firms learn differently depending on the firm's productivity. We find that new units at more productive firms tend to be more productive overall (the levels of the lines are different) but that the rate of productivity growth is invariant to the firm's productivity level (the slopes are the same). Therefore, our analysis suggests no evidence for within-firm learning transfer.

Overall, we are still left with somewhat of a puzzle: most of the firm-level learning happens from individual units learning, and within-firm knowledge transfers do not seem to be driving this. This raises the obvious next question of how individual units learn, which we explore in the next subsection.

7.3 How Units Learn at the Machine Level

In this subsection, we further decompose learning within a unit to see how units become more productive at the machine level. We take a similar approach to the last section, utilizing the decomposition of productivity growth from Foster et al. (2001) shown in equation (5). Instead of decomposing within-firm productivity growth into within- and across-unit components, we will decompose within-unit productivity growth into within- and across-machine series terms. This will help us learn about changing machine usage

Figure 9: Decomposition of Within-Unit Learning Within and Across Machine Series



(a) Within-unit Learning Decomposition

(b) New vs Existing VM Productivity

Notes: Panel (a) presents the decomposition of monthly log unit-level productivity growth, displaying five components: within-firm, across-firm, entry, exit, and cross effects shown in Equation (5). The x-axis represents the firm's cloud experience in years, while the y-axis shows the monthly log productivity growth of the unit. The black dots indicate the average month-to-month productivity growth at the unit level, whereas each bar represents a component of the decomposition. Panel (b) plots the average productivity of the machine series that firms start using in August-September 2022 against the average productivity of the older machine in the same firm. The productivity of existing VMs in September 2022 is normalized to 1.

patterns as units become more productive.

Figure 9(a) displays the results of this decomposition. Units are broken out by the firm's cloud experience. The black dotted line reflects the total within-unit component and, therefore, will be equal to the "within" term of Figure 8(a). It reveals that units' machine choices are not static; they are constantly trying new machines and getting more productive at the machines series they provision. The entry term is substantially negative, reflecting the fact that even well-established units try new machines that they are initially less productive at using. Units discard the less productive machines (the exit term is positive) and get substantially more productive with the machines they retain (the within term is positive). Meanwhile, the cross-term is negative, reflecting that as firms add new machines, the share of their existing machines with increasing productivity declines, leading to a negative correlation between productivity and share changes.

Figure 9(b) provides deeper insight into the within-unit component by comparing the productivity levels of machines that unit firms have experience with versus new machines they adopt. The data reveals that firms are about 15% less productive with newly adopted machines compared to their existing ones. This 15% productivity gap is notably smaller

than the 50% difference observed between new and old units, indicating that units are able to leverage some of their existing expertise when adopting new machines, but there's still a learning curve at the firm level.

Overall, the patterns in within-unit usage demonstrated by Figure 9 reveal a dynamic picture of experimentation-driven learning. Aggregate productivity gains are a compositional effect of productivity declines due to experimentation with new machines, and productivity increases from learning to use the new machines efficiently and stopping the use of the less productive ones that were tried. With these results, we provide empirical evidence for the mechanisms by which productivity growth and learning take place within firms, mirroring the experimentation-driven productivity dispersion observed by Foster et al. (2018) in the context of innovative industries.

8 Counterfactual Resource Calculations: Electricity and VM

This section implements counterfactual scenarios using back-of-the-envelope calculations to quantify the aggregate implications of productivity dispersion. Specifically, we estimate how much compute resources and electricity usage would be saved if all firms reached a given benchmark productivity level. We note that this counterfactual analysis is a partial equilibrium exercise and does not consider many important factors, such as resource constraints of the cloud provider, cloud prices, or firms' endogenous responses to productivity changes. We view this section as an accounting exercise to analyze the aggregate impacts of productivity dispersion rather than a full modeling of the cloud industry.

8.1 Estimating the Electricity Consumption and Utilization

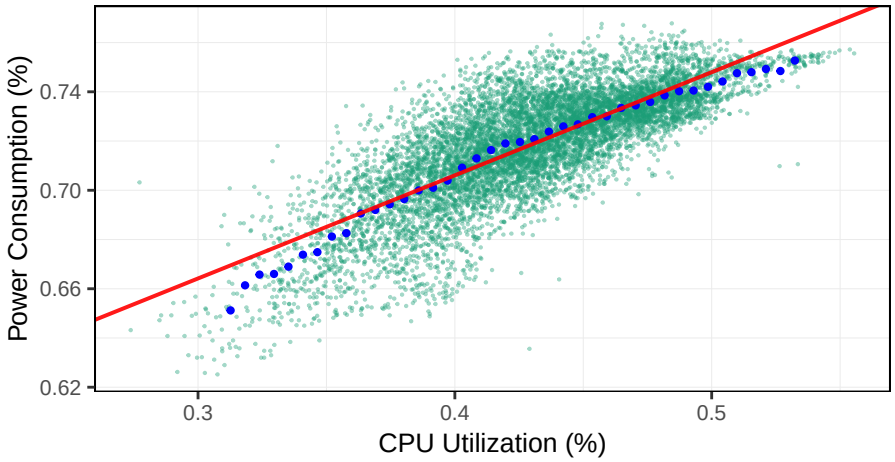
One unique aspect of computing servers is that they consume a significant amount of electricity even when idle because they need to maintain essential functions such as operating systems, network connections, and cooling systems.[37] The relationship between utilization and electricity consumption is crucial for calculating the overall resource impact of productivity dispersion, as various types of inefficiencies (idleness and overprovisioning) result in different levels of inefficient use of electricity.[38]

We estimate the relationship between utilization and consumption using a public dataset from Google Cloud. These data include the 5-minute power consumption read-

[37] Additionally, peripheral components like hard drives and power supplies continue to draw power, and the servers must remain in a ready state for quick activation.

[38] There is experimental literature in computer science showing that the relationship between electricity consumption and idleness can vary, and idle resources can still consume significant electricity (Kansal et al., 2010; Waßmann et al., 2013).

Figure 10: Relationship Between Power and CPU Utilization



Notes: This figure shows the estimated relationship between CPU utilization and power consumption for VMs. The scatter plot shows individual data points representing power consumption (%) as a function of CPU utilization (%) in 5-minute intervals. Blue points show binned scatter points, and the red line shows the regression line.

ings of 57 power domains and the CPU utilization of all VMs hosted on physical servers connected to these power domains. To use these data, we aggregate the utilization of individual VMs to calculate the average utilization of all VMs connected to each power domain. This gives us panel data on relative power consumption and CPU utilization at a 5-minute frequency.[39] Estimating electricity consumption through CPU utilization is a common approach in the computing literature (Möbius et al., 2013). The details of this estimation procedure can be found in Appendix D.5.

We run a simple regression using these data to estimate the relationship between relative power consumption and utilization. Figure 10 presents the results of this regression along with a binscatter plot. The plot reveals several interesting findings. First, the range of average utilization across VMs is limited, with the average utilization rarely exceeding 50% or dropping below 30%. Second, the regression line has a slope of 0.5, indicating that a one percentage point (pp) increase in utilization leads to a 0.5pp increase in power consumption. Finally, by extrapolating the regression line to 0% utilization, we estimate that idle machines consume approximately 50% of their maximum power. Given these

[39]This analysis does not measure power consumption at the VM level, the level of analysis in our study. However, the aggregate changes we analyze should be well-approximated by changes at the power domain level. Moreover, the experimental literature using single servers is consistent with our results (Kansal et al., 2010; Waßmann et al., 2013).

estimates, we assume the following relationship between power and utilization:

$$\mathbb{E}[p_{vt}] = (0.5 + 0.5u_{vt})k_v^{max}$$

where k_v^{max} denotes the power consumption when VM v is utilized at 100% and u_{vt} denotes the utilization of machine v at time t. We further assume that $k_v^{max} = kc_v$, where c_v is the number of cores of machine v. This assumption is reasonable because the computation power of a machine typically increases linearly with the number of cores.

8.2 Counterfactual Resource Calculations

This section provides an overview of the calculation of counterfactual compute resources (core-hours) and electricity, while Appendix D.6 provides the details.

We ask how much total core-hours would be saved if all firms below the benchmark productivity level, $\bar{\omega}_{it}$, reached that level. More formally, we write the counterfactual productivity as:

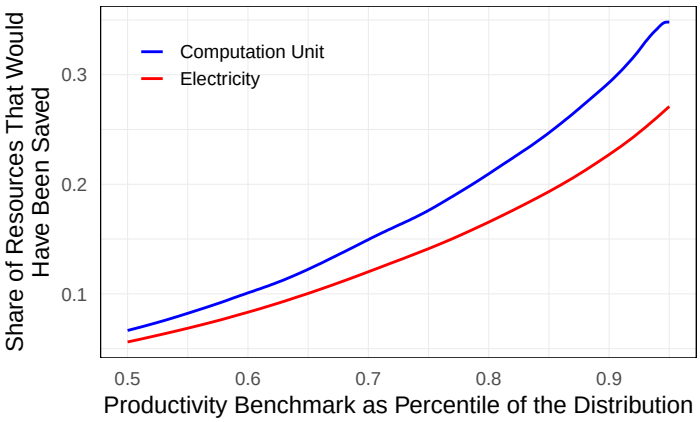
$$\omega_{it}^c = \bar{\omega}_{it} \cdot 1(\omega_{it} < \bar{\omega}_{it}) + \omega_{it} \cdot 1(\omega_{it} \geq \bar{\omega}_{it}) \tag{6}$$

where ω_{it}^c denotes the counterfactual productivity level. The calculation of counterfactual core-hours consumed is relatively straightforward because the calculations do not depend on how firms improve their productivity. Specifically, whether firms reduce idleness or overprovisioning to achieve the benchmark productivity does not affect the total resource savings. Therefore, we calculate the number of core hours each firm saves to reach the benchmark productivity level and then aggregate them at the economy level.

The second counterfactual calculates potential electricity savings due to improved firm productivity. For this counterfactual, we use the relationship between electricity consumption and VM utilization estimated in Equation (6). However, this calculation requires assumptions about how firms improve their productivity because whether a firm reduces idleness or overprovisioning affects the changes in electricity consumption.

To see this, assume a firm has one idle VM and one overprovisioned VM, both with two cores, and that maximum electricity consumption is two units per VM. The productivity of this firm is 25%, and its total electricity consumption is 1 + 1.5 = 2.5 units. Suppose the benchmark productivity is 50%. The firm can achieve this productivity level either by eliminating the idle machine or rightsizing the overprovisioned machine. Eliminating the idle machine would save 1 unit of electricity while rightsizing the overprovisioned machine would save 0.5 units of electricity. Therefore, we must take a stance on how firms

Figure 11: Resource Savings Under Counterfactual Productivity Increase



Notes: This figure illustrates the potential resource savings in computation units and power consumption under a counterfactual increase in productivity. The x-axis represents the productivity benchmark as a percentile of the distribution, while the y-axis shows the percentage of resources that would have been saved. The blue line shows the percent change in core-hours, and the red line shows the percent change in power consumption.

reach the benchmark productivity level.

To make progress, we assume that core-hour savings from each mechanism are proportional to the potential productivity improvement from each mechanism. For example, if a firm has 50% of its machines idle and 30% overprovisioned, the ratio of eliminated idle VMs to eliminate overprovisioned VMs to achieve the benchmark productivity will be 5/3. Under these assumptions, using the calculations detailed in Appendix D.7, we can estimate the total electricity savings if all firms reach the benchmark productivity level.

Figure 11 shows the results from these calculations. The x-axis reports the benchmark productivity level in terms of the firm-level productivity distribution from the 50th to the 100th percentile, whereas the y-axis shows the percent of resource savings. First, even simply elevating all firms to the median level of productivity leads to a 7% decline in compute resources and a 5% decline in electricity. At the 80th percentile, the corresponding numbers are 22% and 17%. These results suggest that firm inefficiencies in computing have significant aggregate implications for productive resources in the economy.

The results also reveal an interesting nonlinear relationship. Total resource savings increase non-linearly with the benchmark productivity and compute resource savings are larger than electricity savings, with the gap increasing as the benchmark productivity level rises. This points to the importance of mapping productivity measures to real-world resource use in production to understand the implications of productivity dispersion.

9 Concluding Remarks

One of the robust findings emerging from firm studies is the high degree of dispersion in various outcomes, including productivity, markups, and labor shares (Syverson, 2011; Van Reenen, 2018). In this paper, we show that similar differences exist in how productively firms use emerging technologies by analyzing evidence from cloud computing.

Our study uses CPU utilization data from over 1 billion VMs employed by nearly 100,000 firms. We develop a novel compute productivity measure and show substantial dispersion in productivity across and within firms. To better understand this dispersion, we study the specific practices that separate high- from low-productivity firms, finding that more productive firms are better at handling compute demand fluctuations, use a wider variety of more specialized machines, and are more attentive to shutting off idle machines. Finally, we study learning in the cloud and find that new firms improve substantially in their first year on the cloud and attain a stable productivity level within four years. Our analysis of within-firm learning demonstrates that firms face barriers to transferring knowledge across divisions.

Our results have several implications for the broader productivity literature. First, although our productivity measure is derived from far more granular and specific data than is typical, our estimates of productivity dispersion and persistence corroborate the general magnitudes seen in other studies. However, we also find that the dynamics of productivity within firms over time are complex—units within a firm may continue to experiment and become more efficient even when firm-level productivity appears to plateau or stabilize. Second, we highlight that organizational frictions such as imperfect monitoring can drive this dispersion, emphasizing the importance of accounting for such frictions in models of firm behavior. Finally, our learning results demonstrate that within-firm productivity growth plays an important role in the evolution of aggregate productivity in industries adopting a new technology, suggesting that the full economic value of cloud computing technology is still being realized.

References

Abowd, J. M., F. Kramarz, and D. N. Margolis (1999). High Wage Workers and High Wage Firms. *Econometrica 67*(2), 251–333.

Agrawal, A., J. Gans, and A. Goldfarb (2018). *Prediction Machines: The Simple Economics of Artificial Intelligence*. Harvard Business Review Press.

Arrow, K. J. (1962). The Economic Implications of Learning by Doing. *The Review of Economic Studies 29*(3), 155–173.

Asker, J., A. Collard-Wexler, and J. De Loecker (2014). Dynamic Inputs and Resource (Mis) Allocation. *Journal of Political Economy 122*(5), 1013–1063.

Athavale, J., M. Yoda, and Y. Joshi (2018). Thermal Modeling of Data Centers for Control and Energy Usage Optimization. In E. M. Sparrow, J. P. Abraham, and J. M. Gorman (Eds.), *Advances in Heat Transfer*, Volume 50, pp. 123–186.

Baily, M. N., E. J. Bartelsman, and J. Haltiwanger (2001). Labor productivity: structural change and cyclical dynamics. *Review of Economics and Statistics 83*(3), 420–433.

Baker, G. P. and T. N. Hubbard (2004). Contractibility and Asset Ownership: On-Board Computers and Governance in U.S. Trucking. *The Quarterly Journal of Economics 119*(4), 1443–1479.

Bandiera, O., I. Barankay, and I. Rasul (2009). Social connections and incentives in the workplace: Evidence from personnel data. *Econometrica 77*(4), 1047–1094.

Bartel, A., C. Ichniowski, and K. Shaw (2007). How Does Information Technology Affect Productivity? Plant-Level Comparisons of Product Innovation, Process Improvement, and Worker Skills. *The Quarterly Journal of Economics 122*(4), 1721–1758.

Bartelsman, E. J. and M. Doms (2000). Understanding Productivity: Lessons from Longitudinal Microdata. *Journal of Economic Literature 38*(3), 569–594.

Benkard, C. L. (2000). Learning and Forgetting: The Dynamics of Aircraft Production. *American Economic Review 90*(4), 1034–1054.

Bloom, N., R. Sadun, and J. V. Reenen (2012). Americans Do IT Better: US Multinationals and the Productivity Miracle. *American Economic Review 102*(1), 167–201.

Bloom, N. and J. Van Reenen (2007). Measuring and Explaining Management Practices Across Firms and Countries. *The Quarterly Journal of Economics 122*(4), 1351–1408.

Braguinsky, S., A. Ohyama, T. Okazaki, and C. Syverson (2015). Acquisitions, Productivity, and Profitability: Evidence from the Japanese Cotton Spinning Industry. *American Economic Review 105*(7), 2086–2119.

Bresnahan, T., S. Greenstein, D. Brownstone, and K. Flamm (1996). Technical Progress and Co-invention in Computing and in the Uses of Computers. *Brookings Papers on Economic Activity. Microeconomics 1996*, 1–83.

Bresnahan, T. F., E. Brynjolfsson, and L. M. Hitt (2002). Information Technology, Workplace Organization, and the Demand for Skilled Labor: Firm-Level Evidence. *The Quarterly Journal of Economics 117*(1), 339–376.

Brown, G. and M. B. Johnson (1969). Public Utility Pricing and Output Under Risk. *American Economic Review 59*(1), 119–128.

Brynjolfsson, E. and L. M. Hitt (2003). Computing Productivity: Firm-Level Evidence. *Review of Economics and Statistics 85*(4), 793–808.

Brynjolfsson, E., W. Jin, and X. Wang (2023). Information Technology, Firm Size, and Industrial Concentration. *National Bureau of Economic Research, No. 31065*.

Brynjolfsson, E. and K. McElheran (2016). The Rapid Adoption of Data-Driven Decision-Making. *American Economic Review 106*(5), 133–139.

Brynjolfsson, E. and P. Milgrom (2013). Complementarity in Organizations. In *The Handbook of Organizational Economics*, pp. 11–55. Princeton University Press Princeton, NJ.

Butters, R. A. (2020). Demand Volatility, Adjustment Costs, and Productivity: An Examination of Capacity Utilization in Hotels and Airlines. *American Economic Journal: Microeconomics 12*(4), 1–44.

Carlton, D. W. (1977). Peak Load Pricing with Stochastic Demand. *American Economic Review 67*(5), 1006–1010.

CAST AI (2024). 2024 Kubernetes Cost Benchmark Report.

Chari, V. V. and H. Hopenhayn (1991). Vintage Human Capital, Growth, and the Diffusion of New Technology. *Journal of Political Economy 99*(6), 1142–1165.

Collard-Wexler, A. and J. De Loecker (2016). Production Function Estimation with Measurement Error in Inputs. *Economic Research Initiatives at Duke, No. 226*.

Cooper, R., J. C. Haltiwanger, and J. Willis (2024). Declining Responsiveness at the Establishment Level: Sources and Productivity Implications. *National Bureau of Economic Research, No. 32130*.

Cortez, E., A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini (2017). Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pp. 153–167.

Couchbase (2022). Couchbase Cloud Evolution Report 2022.

Cunningham, C., L. Foster, C. Grim, J. Haltiwanger, S. W. Pabilonia, J. Stewart, and Z. Wolf (2023). Dispersion in Dispersion: Measuring Establishment-Level Differences in Productivity. *Review of Income and Wealth 69*(4), 999–1032.

David, B. and V. Joseph (2017). Shocks vs. Responsiveness: What Drives Time-Varying Dispersion? *Journal of Political Economy*.

Davis, S. J., C. Grim, and J. Haltiwanger (2008). Productivity Dispersion and Input Prices: The Case of Electricity. *US Census Bureau Center for Economic Studies Paper No. CES-WP-08-33*.

Demirer, M., D. J. J. Hernández, D. Li, and S. Peng (2024). Data, Privacy Laws and Firm Production: Evidence from the GDPR. *National Bureau of Economic Research, No. 32146*.

DeStefano, T., R. Kneller, and J. Timmis (2023). Cloud Computing and Firm Growth. *The Review of Economics and Statistics*, 1–47.

Diaconu, C., M. Aulbach, F. Faerber, P. Frey, M. Grund, A. Kemper, T. Neumann, and M. Thiele (2013). Hyrise: A Main Memory Hybrid Storage Engine. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 195–206. Association for Computing Machinery.

Etro, F. (2015). The Economics of Cloud Computing. In *Cloud Technology: Concepts, Methodologies, Tools, and Applications*, pp. 2135–2148.

Everman, B., M. Gao, and Z. Zong (2022). Evaluating and Reducing Cloud Waste and Cost—A Data-Driven Case Study from Azure Workloads. *Sustainable Computing: Informatics and Systems 35*.

Flexera (2023). 2023 State of the Cloud Report.

Folkerts, E., A. Alexandrov, K. Sachs, A. Iosup, V. Markl, and C. Tosun (2013). Benchmarking in the Cloud: What it Should, Can, and Cannot Be. In *Selected Topics in Performance Evaluation and Benchmarking: 4th TPCTC Revised Selected Papers 4*, pp. 173–188.

Foster, L., C. Grim, and J. Haltiwanger (2016). Reallocation in the Great Recession: Cleansing or Not? *Journal of Labor Economics 34*(S1), S293–S331.

Foster, L., C. Grim, J. C. Haltiwanger, and Z. Wolf (2018). *Innovation, Productivity Dispersion, and Productivity Growth*. Number 24420. National Bureau of Economic Research.

Foster, L., J. Haltiwanger, and C. J. Krizan (2006). Market Selection, Reallocation, and Restructuring in the US Retail Trade Sector in the 1990s. *The Review of Economics and Statistics 88*(4), 748–758.

Foster, L., J. Haltiwanger, and C. Syverson (2008). Reallocation, Firm Turnover, and Efficiency: Selection on Productivity or Profitability? *American Economic Review 98*(1), 394–425.

Foster, L., J. C. Haltiwanger, and C. J. Krizan (2001). Aggregate Productivity Growth: Lessons from Microeconomic Evidence. In *New Developments in Productivity Analysis*, pp. 303–372. University of Chicago Press.

Fox, J. T. and V. Smeets (2011). Does Input Quality Drive Measured Differences in Firm Productivity? *International Economic Review 52*(4), 961–989.

Goldfarb, A. and C. Tucker (2019). Digital Economics. *Journal of Economic Literature 57*(1), 3–43.

Google (2019). Cluster Power Data 2019. Last accessed on 2024-06-24.

Greenstein, S. (2020). Digital Infrastructure. In *Economic Analysis and Infrastructure Investment*, pp. 409–447. University of Chicago Press.

Greenstein, S. and T. P. Fang (2020). Where the Cloud Rests: The Location Strategies of Data Centers. *Harvard Business School Working Paper, No. 21-042*.

Gregg, B. (2014). *Systems Performance: Enterprise and the Cloud*. Pearson Education.

Haltiwanger, J. C., H. R. Hyatt, and J. Spletzer (2022). Industries, Mega Firms, and Increasing Inequality. *National Bureau of Economic Research, No. 29920*.

Hendel, I. and Y. Spiegel (2014). Small Steps for Workers, a Giant Leap for Productivity. *American Economic Journal: Applied Economics 6*(1), 73–90.

Hsieh, C.-T. and P. J. Klenow (2009). Misallocation and Manufacturing TFP in China and India. *The Quarterly Journal of Economics 124*(4), 1403–1448.

Hubbard, T. N. (2003). Information, Decisions, and Productivity: On-Board Computers and Capacity Utilization in Trucking. *American Economic Review 93*(4), 1328–1353.

Hummel, P. and M. Schwarz (2022). Efficient capacity provisioning for firms with multiple locations: The case of the public cloud. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pp. 1018–1039.

Husain Bohra, A. E. and V. Chaudhary (2010). VMeter: Power Modelling for Virtualized Clouds. In *2010 IEEE IPDPSW*, pp. 1–8.

Islam, S., K. Lee, A. Fekete, and A. Liu (2012). How a Consumer Can Measure Elasticity for Cloud Platforms. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, pp. 85–96.

Jiang, Z., C. Lu, Y. Cai, Z. Jiang, and C. Ma (2013). VPower: Metering Power Consumption of VM. In *2013 IEEE 4th International Conference on Software Engineering and Service Science*, pp. 483–486.

Jin, W. (2022). Cloud Adoption and Firm Performance: Evidence from Labor Demand. *Available at SSRN 4082436*.

Jin, W. and K. McElheran (2017). Economies Before Scale: Survival and Performance of Young Plants in the Age of Cloud Computing. *Rotman School of Management Working Paper, No. 3112901*.

Kalyani, A., N. Bloom, M. Carvalho, T. A. Hassan, J. Lerner, and A. Tahoun (2021). The Diffusion of New Technologies. *National Bureau of Economic Research, No. 28999*.

Kansal, A., F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya (2010). Virtual Machine Power Metering and Provisioning. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, pp. 39–50.

Kehrig, M. and N. Vincent (2019). Good Dispersion, Bad Dispersion. *National Bureau of Economic Research, No. 25923*.

Kellogg, R. (2011). Learning by Drilling: Interfirm Learning and Relationship Persistence in the Texas Oilpatch. *The Quarterly Journal of Economics 126*(4), 1961–2004.

Leigh, N. G., B. Kraft, and H. Lee (2020). Robots, Skill Demand and Manufacturing in US Regional Labour Markets. *Cambridge Journal of Regions, Economy and Society 13*(1), 77–97.

Levitt, S., J. List, and C. Syverson (2013). Toward an Understanding of Learning by Doing: Evidence from an Automobile Assembly Plant. *Journal of Political Economy 121*(4), 643–681.

Lu, Y., G. M. Phillips, and J. Yang (2024). The Impact of Cloud Computing and AI on Industry Dynamics and Firm Financing. *Available at SSRN 4480570*.

Mason, K., M. Duggan, E. Barrett, J. Duggan, and E. Howley (2018). Predicting Host CPU Utilization in the Cloud Using Evolutionary Neural Networks. *Future Generation Computer Systems 86*, 162–173.

McElheran, K., J. F. Li, E. Brynjolfsson, Z. Kroff, E. Dinlersoz, L. Foster, and N. Zolas (2024). AI Adoption in America: Who, What, and Where. *Journal of Economics & Management Strategy 33*(2), 375–415.

Melitz, M. J. and S. Polanec (2015). Dynamic Olley-Pakes Productivity Decomposition with Entry and Exit. *The Rand journal of Economics 46*(2), 362–375.

Metcalfe, R. D., A. B. Sollaci, and C. Syverson (2023). Managers and Productivity in Retail. *National Bureau of Economic Research, No. 31192*.

Microsoft Azure (2019). Azure Public Dataset V2. Last accessed on 2024-06-25.

Miller, A. R. and C. E. Tucker (2011). Can Health Care Information Technology Save Babies? *Journal of Political Economy 119*(2), 289–324.

Möbius, C., W. Dargie, and A. Schill (2013). Power Consumption Estimation Models for Processors, Virtual Machines, and Servers. *IEEE Transactions on Parallel and Distributed Systems 25*(6), 1600–1614.

Murciano-Goroff, R., R. Zhuo, and S. Greenstein (2024). Navigating Software Vulnerabilities: Eighteen Years of Evidence from Medium and Large US Organizations. *National Bureau of Economic Research, No. 32696*.

Nguyen, T. L. and A. Lebre (2017). Virtual Machine Boot Time Model. In *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, pp. 430–437. IEEE.

Orr, S. (2022). Within-Firm Productivity Dispersion: Estimates and Implications. *Journal of Political Economy 130*(11), 2771–2828.

Osei-Opoku, E., R. Regaieg, and M. Koubaa (2020). An Accurate Power Consumption Model for Cloud Computing Data Centres. *International Journal of Engineering Applied Sciences and Technology 04*, 395–399.

Pindyck, R. S. (1986). Irreversible Investment, Capacity Choice, and the Value of the Firm. *National Bureau of Economic Research, No. 1980*.

Pozzi, A. and F. Schivardi (2016). Demand or Productivity: What Determines Firm Growth? *The RAND Journal of Economics 47*(3), 608–630.

Radovanovic, A., B. Chen, S. Talukdar, B. Roy, A. Duarte, and M. Shahbazi (2022). Power Modeling for Effective Datacenter Planning and Compute Management. *IEEE Transactions on Smart Grid 13*(2), 1611–1621.

Reiss, C., A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch (2012). Towards Understanding Heterogeneous Clouds at Scale: Google Trace Analysis. *Intel Science and Technology Center for Cloud Computing*.

Rosen, S. (1972). Learning by Experience as Joint Production. *The Quarterly Journal of Economics 86*(3), 366–382.

Singh, R., M. A. Qureshi, and K. Annamalai (2015). A Brief Overview of Recent Developments in Thermal Management in Microelectronics. *Journal of Electronic Packaging 137*(4).

Solow, R. M. (1957). Technical Change and the Aggregate Production Function. *The Review of Economics and Statistics 39*(3), 312–320.

Syverson, C. (2004). Product Substitutability and Productivity Dispersion. *Review of Economics and Statistics 86*(2), 534–550.

Syverson, C. (2011). What Determines Productivity? *Journal of Economic literature 49*(2), 326–365.

Tadelis, S., C. Hooton, U. Manjeer, D. Deisenroth, N. Wernerfelt, N. Dadson, and L. Greenbaum (2023). Learning, Sophistication, and the Returns to Advertising: Implications for Differences in Firm Performance. *National Bureau of Economic Research, No. 31201*.

Tambe, P., L. Hitt, D. Rock, and E. Brynjolfsson (2020). Digital Capital and Superstar Firms. *National Bureau of Economic Research, No. 28285*.

Tambe, P., L. M. Hitt, and E. Brynjolfsson (2012). The Extroverted Firm: How External Information Practices Affect Innovation and Productivity. *Management Science 58*(5), 843–859.

Thornton, R. A. and P. Thompson (2001). Learning from Experience and Learning from Others: An Exploration of Learning and Spillovers in Wartime Shipbuilding. *American Economic Review 91*(5), 1350–1368.

Tirmazi, M., A. Barker, N. Deng, M. E. Haque, Z. G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes (2020). Borg: the Next Generation. In *Proceedings of the Fifteenth European Conference on Computer Systems*, pp. 1–14.

Van Reenen, J. (2018). Increasing differences between firms: Market power and the macro-economy.

Veni, T. and S. M. S. Bhanu (2016). Prediction Model for Virtual Machine Power Consumption in Cloud Environments. *Procedia Computer Science 87*, 122–127.

Verma, A., L. Pedrosa, M. R. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes (2015). Large-Scale Cluster Management at Google with Borg. In *Proceedings of the European Conference on Computer Systems (EuroSys)*.

Waßmann, I., D. Versick, and D. Tavangarian (2013). Energy Consumption Estimation of Virtual Machines. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pp. 1151–1156.

Weinman, J. (2011). Time is Money: the Value of On-Demand.

Whitney, J., P. Delforge, et al. (2014). Data Center Efficiency Assessment. *Issue Paper on NRDC*.

Wilkes, J., C. Reiss, N. Deng, M. E. Haque, and M. Tirmazi (2020). Google Cluster-Usage Traces V3. Last accessed on 2024-06-24.

Zolas, N., Z. Kroff, E. Brynjolfsson, K. McElheran, D. N. Beede, C. Buffington, N. Goldschlag, L. Foster, and E. Dinlersoz (2021). Advanced Technologies Adoption and Use by US Firms: Evidence from the Annual Business Survey. *National Bureau of Economic Research, No. 28290*.

Firm Productivity and Learning in Computing

James Brand Mert Demirer
Connor Finucane Avner A. Kreps

Appendix - For Online Publication

Contents

A Institutional Details

A.1 Details of VM Deployment

In this section, we provide more details about VMs.

A.1.1 VM Creation and Selection

All cloud providers offer a browser-based platform with step-by-step instructions for generating a VM. These instructions are typically designed for first-time VM creation and may not be relevant for firms' daily operations. However, in this section, we outline the VM selection steps to explain the various components involved. In the next subsection, we provide an overview of the tools commonly used by firms for VM management and operations.

Account Creation: Before deploying VMs, user accounts or service accounts need to be created and configured. This involves setting up the necessary permissions and roles to ensure that users or services have the appropriate access levels to manage and interact with the VMs.

Resource Allocation: During deployment, specific amounts of CPU, memory, and storage are determined for the VM.

Network Configuration: VMs need to be connected to the appropriate virtual networks. This involves assigning IP addresses, configuring network interfaces, and setting up any necessary VLANs (virtual local access networks).

Storage Provisioning: VMs require storage for the operating system, applications, and data. This can involve allocating space on local disks, SANs (storage area networks), or network-attached storage systems.

Security Settings: Security configurations are applied during deployment. This includes setting up firewalls, defining access controls, and implementing any required encryption.

Region Choice: The user selects the geographic region of the data center for the VM. This choice can impact performance due to latency and varying redundancy policies.

These steps are required when creating a VM for the first time. In practice, developers can use VM images, which are templates that store information such as the OS, security settings, and pre-installed software, allowing users to create VMs without redefining these parameters. Organizations often maintain a library of standardized images for various purposes, streamlining the VM deployment process.

A.1.2 Available Tools For Flexibility

As we argue in Section 2.4, there are many available tools in cloud computing that can reduce potential frictions firms might face when deploying VMs. In this section, we list some of these tools and provide a brief description.

Cloning a VM: Cloning technology allows users to create new VMs from the running state of an existing VM. The cloned VM is identical to the source VM and can be created quickly from a specific point in time. This method makes it efficient to deploy large-scale applications and enables the creation of numerous VMs on a single host. All major cloud providers offer some form of this cloning capability.

Auto Shutdown: In cloud computing, users have the capability to schedule or automatically shut down VMs to help manage costs and optimize resource usage. This functionality allows users to define specific times for VMs to stop and start, eliminating the need for constant monitoring.

Live Migration: Live migration allows users to move running VMs from one physical host to another without downtime. This technology ensures that applications continue to operate during maintenance, load balancing, or hardware upgrades.

Automatic Redundancy and Fault Tolerance: Cloud providers offer built-in automatic redundancy and fault tolerance to ensure high availability and reliability of services. These features include distributing data and workloads across multiple servers, data centers, or geographic regions to prevent single points of failure. In case of hardware or software failures, automatic failover mechanisms can redirect traffic to healthy instances, minimizing downtime and maintaining service continuity.

Autoscaling: In cloud computing, autoscaling refers to the automatic adjustment of compute resources based on the current demand. This capability allows cloud environments to dynamically allocate or deallocate resources such as CPU, memory, and storage to applications or services as needed. When demand increases, autoscaling provisions additional instances to handle the load, ensuring that applications can maintain performance levels. Conversely, when demand decreases, it reduces the number of active instances, scaling down the resources in use.

The process of autoscaling involves automatic monitoring of the performance metrics and resource utilization of applications. Firms can set autoscaling rules and policies based on their specific requirements, such as thresholds for CPU usage, memory consumption, or network traffic. For example, if CPU utilization exceeds a certain threshold, the autoscaling mechanism adds more instances to distribute the load. Similarly, if resource usage falls below a specified level, it scales back the instances. These rules can be configured through

cloud provider dashboards or APIs, allowing firms to tailor the autoscaling behavior to their application needs.

Load Balancers: Load balancers distribute incoming network traffic across multiple servers or instances. By evenly distributing the load, load balancers ensure that no single server becomes overwhelmed, which helps maintain application performance and reliability. They monitor the health of instances and redirect traffic away from any that are failing or underperforming, ensuring continuous availability. Load balancers can handle various types of traffic, including HTTP/HTTPS, TCP, and UDP, making them versatile for different types of applications and services. Load balancers complement autoscaling by working together to optimize resource utilization and application performance. While autoscaling dynamically adjusts the number of active instances based on demand, load balancers distribute the incoming traffic among these instances.

Cost Monitoring Tools: Cloud providers offer a range of tools to help users monitor and manage their resource utilization and costs. For example, AWS CloudWatch provides comprehensive monitoring for resources and applications across AWS environments. It includes advanced visualization tools, automated alarms, and integration with other AWS services to help identify and manage idle resources. Similarly, Google Cloud Operations, formerly known as Stackdriver, offers integrated monitoring, logging, and tracing for applications and systems on Google Cloud. It includes real-time log management and metrics observability to detect and manage idle resources. These tools aid in optimizing resource usage and controlling cloud spending.

A.2 Details of VM Deployment: Two Examples

In this section, we provide two examples of deploying IT resources in cloud computing to help readers understand its use cases. While this section is quite technical, it offers important insights about the day-to-day work of software developers in cloud environments.

A.2.1 "Create, Read, Update, Delete" Application

The first toy example we consider is the deployment of a simple CRUD (Create, Read, Update, Delete) application. This refers to a general class of applications that provide a user interface for the typical operations involved in persistent storage. A common example of this application is a blog or message board. In this example "Create" would correspond to a user making a new post or comment, "Read" could correspond to the functionality of listing all posts and comments corresponding to some filter, "Update" corresponds to editing posts or profile information, and "Delete" corresponds to the deletion of posts or comments. The basic components of the architecture of this application includes a

web server, which is accessible by users on the public internet, and a database server, which is typically only accessible by the web server itself. The web server's responsibilities include authenticating users, producing HTML that provides the information and features available to the user, and issuing control commands and queries to the database based on the user's request.

We describe specific deployment scenarios of this application type on the Microsoft Azure cloud. For the first example, we consider hosting this application only in a single region, with a fixed resource footprint, and with manual processes to deploy resources. To complete this deployment we:

1. Create a Resource Group (RG), a logical group which will contain all of the resources for this deployment.

2. Create a Virtual Network (VNet) and create a subnet within this VNet.

3. Select and provision an appropriate VM to run the web application based on our application's requirements, budget, and account quota. Additionally, we must:

   (a) Select the proper region and operating system for the application.

   (b) Add the VM's network interface to the appropriate subnet within the VNet.

   (c) Create a Network Security Group (NSG) and add rules that restrict the incoming traffic to SSH/RDP and HTTPS traffic from internet IP Addresses.

   (d) Give the VM a static IP address. It is also possible to give it a human-readable alias using Azure DNS.

4. Create an Azure Database for PostgreSQL to serve as the persistent storage back-end for the application.

   (a) Similar to the web server VM, we must choose an appropriate virtual core count, virtual memory amount, storage size, storage scale rule, and storage performance tier based on our requirements and budget.

   (b) We will place the database in the same region as our web server.

   (c) We will disallow public IP access to the database and integrate it into the existing VNet and the same or new subnet.

At this point our resources are established and we can install the application onto the web server and complete the connection between the application and the database to be

able to service user requests using the persistent storage. We can monitor the health and utilization of the instances using Azure Monitor.

The above architecture leans towards using IaaS solutions (Infrastructure as a Service) and is not capable of scaling horizontally. It is a simple deployment method that can be hard to maintain, and the architecture will likely be insufficient to handle dynamic loads. Since we can only control the size of a single instance for the web server and database, respectively, it will be challenging to avoid being under or over-provisioned, and we will incur downtime in the application if we need to scale instances up or down. A common way to handle this is by using IaaS offerings for the web server, such as a Virtual Machine Scale Set (VMSS) and Load Balancer (LB). The VMSS is a collection of identical VMs in a single region. With this deployment method, we can define conditions on the pool of VMs that will trigger custom scale-up and scale-down actions. These rules or conditions are defined by statistics on the time series of instance-level counters such as CPU, Network, and Disk Utilization. The LB can be configured with a front-end IP address to accept user traffic and then given the VMSS as a back-end pool to distribute requests over. This architecture allows us to scale horizontally instead of vertically, increasing our ability to handle dynamic loads, making us less likely to be under or over-provisioned at any given time, and decreasing our application's expected downtime. In this design, it is possible to use reserved instance purchases to reduce costs on the compute hosting the application. We can monitor request traffic, instance utilization, and allocation rates to develop an estimate of a lower bound on the capacity we require to host the application.

Scaling the database using only IaaS offerings would require a lot of engineering effort, and on modern cloud platforms, it is much more common to use a managed database service. In the above example, we could use Azure SQL Server, or if we switch away from a relational database, we could consider Cosmos DB. The choice of which type of database to use is technical and heavily depends on the data model and transaction requirements of the application. Generally, Cosmos DB offers more scaling options but is a non-relational/NoSQL database. Fortunately, both options allow for horizontal scaling for read replicas. This means that we can apply similar scaling procedures to service the read requests from the web application servers.

This redesigned architecture requires more engineering effort to build and maintain but is significantly more capable, reliable, and cost-effective once we are presented with a nonnegligible amount of user traffic. While this design is more complex to implement, the jump in complexity is far less than if we were to implement this without modern cloud services and tools. This is especially true when we consider the complexity that is abstracted away from us by the managed database services. If we wanted to use PaaS

offerings to host the application, we could consider container-based services like Azure Kubernetes Service (AKS) or serverless options like Azure Functions. Since the focus of this paper is on IaaS resources, we do not include details on how to redesign around these services.

A.2.2 Data Analytics and Machine Learning

Another common use of modern public cloud infrastructure is building pipelines for data analytics and machine learning use cases. Since many of these use cases can be implemented purely as PaaS services, we will focus on a sample architecture that favors IaaS resources for batch training of a custom machine learning model or a numerical simulation. In these settings, it is common to have a highly parallelizable workload that consists of iterating over a set of parameters or hyperparameters for an underlying model or simulation, where for each parameter setting, we wish to construct the model or simulation according to the specified parameters, and then train the model or execute the simulation and store the relevant results from the process. We could implement a system like this using Azure Blob Storage or Azure Data Lake Storage (ADLS) to store the model or simulation parameter configurations and training/simulation results, Azure Batch to acquire the required capacity and allocate jobs to nodes, and then either a single dedicated VM or cluster of dedicated VMs that orchestrate the Azure Batch node pool and collate results from the storage. The exact steps to provision these resources and develop the code to orchestrate and execute the jobs is involved, but we can describe an outline of the process and architecture:

1. Provision a storage account and create a container that will store model/simulation result blobs and another to store blobs defining the model/simulation parameters.[40]

2. Provision an Azure Container Registry, construct a Docker image, which defines the functionality of the model or simulation given the parameter data, and write that Docker image into the container registry.[41]

3. Given an estimate of model training time and the desired parameters to iterate over, estimate the number of worker nodes needed to complete the entire batch job in a reasonable time period.

[40] A blob, short for Binary Large OBject, is a collection of binary data stored as a single entity in a database management system. In the context of cloud storage, blobs are used to store large amounts of unstructured data such as text, images, videos, or, in this case, model/simulation results and parameters.

[41] Docker is a platform that uses OS-level virtualization to deliver software in packages called containers. These containers are lightweight, standalone, and executable packages of software that include everything needed to run an application: code, runtime, system tools, system libraries, and settings.

4. From the orchestration nodes, construct the parameter data structures, write them to storage, and then define jobs using the Azure Batch API, which points to the parameters in storage and the image stored in the container registry.

5. Start the job and monitor the job status for task-level failures from the orchestration nodes.

6. Detect when the job completes and collate the results to do model parameter selection or generate simulation reports on the orchestration nodes.

This architecture still leverages some PaaS services to manage persistent storage and container images. Azure Batch is capable of acquiring very large amounts of capacity (tens of thousands of instances) relatively quickly and efficiently, allocating work to the acquired nodes.

All of the examples listed above comprise only a small fraction of all of the use cases of modern public clouds like Azure. However, even among this small sample, we see significant variability in terms of the PaaS services coupled with our deployed VMs and the potential utilization patterns of the deployed VMs themselves. The latter utilization variability by use case is relevant to the VM counter data examined in this paper. For example, in the CRUD application case, we are subject to an uncontrollable and variable amount of user requests in the future and must design our architecture to handle both the expected and unexpected variability in this load. In the model training pipeline, we can learn more about the workload ahead of time. Instead, we need to focus on ensuring that the worker pool is sized correctly and that we efficiently pack jobs onto the nodes. Another substantial difference between these two applications is that the CRUD application is more likely to be what is known as an "IO bound" workload, while the machine learning pipeline is "GPU bound" or "CPU bound" depending on the model or simulation type. CRUD application VMs spend the majority of their time communicating with the database layer or waiting on responses from users, often at unpredictable intervals. Meanwhile, the ML pipeline VMs will only reach out to storage predictably at the beginning and end of their job run. In the case of the ML pipeline, it is much easier to achieve a very high CPU utilization rate than with instances in the CRUD application. If we are looking at all of the counter metrics, we will see that instances in different layers of the architectures are more likely to achieve high utilization rates in different metrics. In some cases, it may not be possible to achieve the same average CPU utilization rates in the CRUD application that we could achieve in the ML pipeline while maintaining reasonable disk, memory, and network utilization rates. Nevertheless, firms still have the ability and incentive to match

the capacity of the VM they pay for to the maximum requirements of their workload as closely as possible for both types of workloads.

A.3 Details on Cloud Computing

A.3.1 Non-IaaS Cloud Computing

As we mention in the text, there are three types of cloud products: SaaS, PaaS, and IaaS. SaaS is fundamentally different from PaaS and IaaS because it has broader coverage (including email, office products, etc.) and is more consumer-facing than B2B products. Therefore, we restrict our discussion to PaaS and IaaS in this section. While this paper primarily focuses on IaaS and VMs, which are more common than PaaS, PaaS could be a substitute for IaaS in some cases. We provide some discussion of PaaS services in this section.

Non-IaaS cloud computing encompasses a range of services that abstract and automate the underlying infrastructure, allowing users to focus more on application development and deployment rather than managing physical or virtual hardware. These services include containers and container orchestration, serverless computing, and Code as a Service. Containers package applications with their dependencies, ensuring consistent and reliable performance across different environments, while container orchestration tools manage the deployment, scaling, and operation of these containers. Serverless computing, also known as Function as a Service (FaaS), enables developers to run code without provisioning or managing servers, as the cloud provider handles the infrastructure, scaling, and execution of code in response to events. PaaS offers a managed platform that includes the operating system, development frameworks, and other tools needed to build and deploy applications, abstracting the underlying infrastructure to allow for faster and easier application development.

Despite the convenience of non-IaaS services, IaaS remains more common because it provides a high degree of flexibility and control, allowing users to tailor the infrastructure to their specific needs. This level of control is important for complex, custom applications and enterprise environments that require precise configurations and optimizations. Additionally, IaaS can accommodate a wide range of workloads, from legacy applications to modern microservices, making it more versatile than PaaS services.

A.3.2 Resource Availability in Cloud Computing

In cloud computing, firms typically request a quota that specifies the maximum capacity they may need at any given time. Once this quota is established, users can access the

resources up to that limit whenever they require them. Except for a few specialized VMs, firms can adjust or increase their quota anytime. Even though the quota cannot guarantee the immediate availability of resources, the high reliability of cloud services ensures that firms rarely face situations where their resource requests cannot be fulfilled.

Cloud computing providers ensure sufficient capacity by making significant investments in their data centers. While predicting the needs of individual firms can be challenging, cloud providers can more accurately forecast aggregate demand across all users. By leveraging the law of large numbers along with the relative uncorrelatedness of variation in firms' workloads, they can predict overall load requirements more effectively and invest accordingly to maintain sufficient infrastructure.

However, aggregate demand is still volatile, and cloud providers need to make short-term adjustments to the available capacity. Cloud providers use a mechanism known as spot instances to manage this volatility. Spot instances allow providers to rent out unused capacity at steep discounts, offering a cost-effective option for firms with flexible workloads. However, these instances come with the caveat that the cloud provider can reclaim the resources at any time. Essentially, cloud providers overinvest in infrastructure to ensure they can meet peak demand and then monetize the excess capacity through the spot market.

A.3.3 Pricing in Cloud Computing

Cloud computing pricing is primarily designed to be on-demand, allowing users to pay only for the resources they consume. This flexible pricing model is often linear, meaning that costs scale directly with usage—whether it's computing power, storage, or bandwidth. Users are billed based on the amount of resources used over a given period, making it straightforward to predict and manage costs for various workloads.

However, beyond this simple linear model, cloud providers offer more elaborate pricing mechanisms to cater to different needs and usage patterns. For instance, reserved instances allow users to commit to a certain level of resource usage over a longer term in exchange for a lower rate, which can be beneficial for predictable, steady workloads. Additionally, spot instances are available at a significantly reduced cost but come with the trade-off that the cloud provider can reclaim the resources with little notice, making them ideal for non-critical or flexible tasks. Despite these varied pricing strategies, cloud costs can still be thought of as variable costs for firms, as they are still directly tied to the level of resource consumption.

B Data Appendix

B.1 CPU Utilization Data

CPU utilization is a fundamental metric in computing that quantifies the workload on a computer's central processing unit. It is typically expressed as a percentage, representing the proportion of time the CPU spends executing non-idle tasks relative to its total available processing time (Gregg, 2014).

The most common method for measuring CPU utilization relies on system counters provided by the operating system. These counters continuously track the CPU's state, recording the time spent in various modes such as user mode (executing application code), system mode (executing kernel-level operations), and idle mode. By sampling these counters at regular intervals, typically every few milliseconds, the operating system can calculate the percentage of time the CPU spends in non-idle states (Gregg, 2014). This data is then aggregated over longer periods (e.g., seconds or minutes) to provide a meaningful representation of CPU usage.

The raw data we have access to are aggregations of counter readings at the 5-minute level, taking the maximum utilization reading in each 5-minute interval. Therefore, for each VM, at a 5-minute interval, we have the maximum CPU utilization. Since we have data on more than 1 billion VMs, the data at this granularity is not manageable, and we further aggregate this data to the VM-day level by calculating the inverse CDF of max CPU utilization in 5% intervals. In particular, for each 5% increment, we calculate the number of counters under 5, 10, . . ., 95% utilization. We also record the maximum CPU utilization and the total number of hours the VM is running during that day. This sample forms the primary dataset for all analyses conducted in the paper.

B.2 Virtual Machine Data

Together with the information on CPU utilization of VMs, we also collect information on the important characteristics of VMs to understand their usage patterns and performance. Our data includes the data center of the VM, which is anonymized for privacy and security reasons. However, we observe the geographical region of the data center, categorized into US, EU, and Other. This helps identify the geographical location of the firm and whether firms and units run jobs outside of their domestic country. We also observed the series to which each VM belongs. Cloud providers group VMs of similar sizes, hardware, and features into the same families, typically referred to as machine series or instance types,

depending on the cloud provider.[42] This variable is anonymized, and we only observe a unique identifier for confidentiality reasons.

Another important piece of information is the VM type, which categorizes VMs based on their primary purpose, such as general-purpose, compute-optimized, memory-optimized, and storage-optimized. We observe the actual values of these variables, allowing us to analyze the types of VMs used by each firm. In addition, we observe other key VM characteristics, including their operating system (Linux or Windows), memory, and number of cores.

These VM characteristics include the following set of VM groupings, which we use throughout the text.

1. The *VM type*, as described above, categorizes VMs based on their primary purpose. It takes the values of general purpose, compute optimized, memory optimized, storage optimized, HPC (high-performance computing), or GPU (graphics processing unit).

2. The *VM series* is a more granular indicator of the architecture of the VM. Each VM series is defined by a combination of hardware such as processing chip, software components, and certain proportions or features such as the menu of available memory per CPU combinations. Generally speaking, VMs that are in the same VM series will only differ according to the data center they are physically located in, size attributes such as cores or memory, and their operating system; the VM series defines all other attributes of the VM.

3. The *VM configuration* is a unique combination of a VM series, data center, operating system, number of cores, and amount of memory. This variable represents the exact hardware and software that the VM user chooses, and is the most granular variable that classifies a VM.

B.3 Firm and Unit Level Data

There are two ID variables associated with the creator of each VM in our CPU utilization data. Each VM is associated with a unit ID. The unit ID collects all users that share a system administrative structure for oversight of the VMs and a payment/billing contract with the cloud provider. Each unit ID is then associated with a higher-level firm ID. All unit IDs whose users are part of the same directory will be part of the same firm ID.

Although our CPU utilization data are intermittent throughout 2017-2023, we have unit-month and firm-month level panel datasets that cover the entire sample period from

[42] For examples of machine series from top providers, see these links: Google Cloud— Machine Families, AWS EC2— Instance Types

2017 until mid-2023. These panel datasets contain normalized statistics on the cloud usage of each unit and firm in each month, including the number of VMs deployed, the number of active VM days, a measure of cloud spend, and the total number of hours and core-hours across all VMs on the cloud. The datasets also include whether the firm used any reserved instances and the share of the firm's total usage of PaaS products.

In addition to these panels, we also have datasets with information about each unit and firm in our sample. We observe the industry of each firm, which we then map to SIC codes. Each firm is also associated with potentially multiple billing addresses; while we do not observe the billing addresses themselves, we observe indicators for whether the firm is associated with billing addresses in the US, EU, or another region of the world, as well as an indicator for whether the firm is multinational (has billing addresses in multiple countries). We also construct a "usage region" for each firm and unit based on the locations of the data centers in which the firm or unit had the most compute usage. Since network latency increases with the geographic distance between the user and the data center, cloud users are more likely to choose data centers in the same region, making the region that the firm or unit had the most usage a useful proxy for the operational location of the firm or unit. Finally, our data also include quartiles within region, industry, and year of an independent measure of firm size for each firm.

B.4 Sampling Details

In addition to aggregating our data to the VM-day level used in this paper, we take multiple other steps to reduce the data to a manageable size and to fully obfuscate firm identities. With our VM-day dataset, we perform three such sampling steps. First, we filter out the firms that fall below fixed thresholds for total usage and consistency of usage. Second, we thin the right tail of the distribution by sampling the top few percentiles of firms (measured by total usage) at an undisclosed sample rate. This sampling allows us to maintain a sample that is representative of a wide array of firms while also reducing the size of the data further and eliminating the possibility of identifying large firms through the combination of our data with public information and from producing sensitive aggregate statistics about the cloud platform itself. Finally, we sample VM-day observations at a fixed undisclosed rate within the firm that is between 70% and 100%.

B.5 Cleaning Steps for VM-day Data

In addition to the initial cleaning and sampling steps applied to the raw data, we implement additional filtering steps to the VM data to remove firms with low usage and short-duration

VMs.

First, we remove VMs with a duration of less than 20 minutes. These VMs are likely used by firms to initiate another job, such as testing configurations or starting batch processing jobs. They may also be part of a scale set that has been activated only briefly to handle temporary spikes in demand. These short-duration VMs account for a negligible share of total core hours. Additionally, we exclude a very small number of VMs associated with operating systems other than Linux and Windows or that are missing information such as memory or VM configuration.

At the firm level, we remove firms that are inactive for more than 80% of the months and those with less than 1,000 hours, 500 core hours, or 200 VM-days of usage. This step ensures that each firm has a sufficient sample size to estimate its productivity accurately. These cleaning steps affect only a negligible fraction of firms, accounting for less than 1% of those in the raw data.

B.6 Public Cloud and Compute Data

There are a number of publicly available *traces* detailing utilization information pertaining to both cloud and cluster environments. A trace is a dataset containing detailed information about the utilization and performance of compute resources. These traces often provide data on users and compute hardware as well.

These traces are collected from real-world compute environments. Providers of these compute resources make these traces publicly available for research, analysis, and educational purposes. Below, we describe three usage traces. The first dataset we describe was released by Google Cloud (GC) and is a trace of a high-performance computing cluster. The second dataset is a public *power* trace provided by GC (Google, 2019) that corresponds to the public GC cluster trace. The third was released by Microsoft Azure (Azure) and is a trace of a cloud computing environment.

As in the main analysis, the key variables in these datasets pertain to both resources provisioned and utilized. Each of the datasets described in this section is a trace of cluster information about users' resources, compute activity and the networks within which their VMs run workloads.

B.6.1 Google Cloud Platform Cluster Trace

The contents of this trace allow for the close study of job scheduling and cluster management (Verma et al., 2015; Tirmazi et al., 2020). However, our main interest in these data is to utilize information provided on resource utilization to measure the compute productivity

of cloud computing users.

The GC trace was sampled in May 2019 and pertains to eight clusters utilizing Google's *Borg* cluster manager. These clusters are located in data centers in New York and Chicago in North America, Helsinki and Brussels in Europe, and Singapore in Asia. The users of the clusters traced in these data are Google engineers and services.

The unit of observation in the usage component of the trace data is a *task*. Tasks are processes that originate from programs running as part of jobs submitted to the cluster manager by users. The tasks detailed in this trace are either the result of jobs run by Google engineers, or they are run within reserved resources available to Google services used by internal or external users (Wilkes et al., 2020). Tasks are executed either within resource allocations (similar to a VM) or directly on machines. The data also contains task-related event information. For example, task-related events include when the task is submitted to the scheduler, when the task completes, as well as auxiliary events that occur during the task's runtime or if it fails. Observations are recorded every five minutes. Hence, these data are an unbalanced panel of usage and event information of processes executed on the a cluster.

On the user end, these data contain original resource requests and usage, and user-configured constraints on the requested resources. The resources users can request are memory (RAM) and CPU cores. Requested and used memory is measured in bytes and then reported after being normalized by a constant factor. CPU requests are measured in internal "Google Compute Units" (GCUs) (Wilkes et al., 2020), which are similar to CPU cores but enable the comparison of compute hardware across machines. Similar to memory, the GCU measurements are reported after being normalized by a constant factor. CPU usage is measured in CPU-seconds and reported after being normalized. The data also contains information about machine attributes, machine availability, obfuscated user and job identifiers, and variables that track events, missing data, and reasons for task failures.

B.6.2 Google Cloud Platform Power Trace

The content of this dataset is similar to the Google trades data described above, with the addition of product information on power consumption, which allows us to measure the relationship between CPU utilization and power consumption.

A typical data center draws power from its local grid. These facilities also maintain a set of large generators that ensure continuous operations during power outages or other disruptions to the primary grid power supply. Once power is drawn into the data center, it is transformed down into a power distribution unit (PDU). PDUs are the main distributor

of power to both IT and non-IT resources in a data center. They manage the flow of power to equipment and monitor environmental factors like temperature and humidity. PDUs often also provide a layer of redundancy in order to increase uptime and reliability. On the data center floor, PDUs manage power supplied to clusters and their supporting IT equipment, as well non-IT equipment such as cooling resources (Radovanovic et al., 2022).

Modern cluster computing produces large amounts of heat. Therefore, measuring power consumption requires accounting for the composite power supply to both IT (e.g., servers) and non-IT (e.g., coolers) resources on the data center floor (Singh et al., 2015; Athavale et al., 2018). The power trace provided by GC incorporates both IT and non-IT power demands. This dataset includes power utilization levels of 55 PDUs, which manage the power supply to each of the clusters in the GC cluster trace. The power for each cluster is managed by multiple PDUs. The data include two key variables: total power utilization and production power utilization.

Total power utilization, measured in 5-minute intervals, indicates the percentage of available power capacity consumed through a PDU for all IT and non-IT equipment, including coolers. Similarly, production power utilization is an estimated measure provided by GC that details the power consumption attributable to production workloads, including power consumed by non-IT equipment.

B.6.3 Microsoft Azure Trace

Azure publicly provides a number of cloud traces. We focus on a 2019 trace containing a representative subset of first-party Azure VM workloads (Microsoft Azure, 2019). These are VM workloads used by Microsoft's own services and are drawn from a single geographical region. An analysis of this trace can be found in Diaconu et al. (2013).

The dataset is 235GB and consists of 198 files. It covers 30 consecutive days and includes over two million VMs. The dataset comprises unbalanced panel data with 5-minute VM CPU utilization readings. Nearly 1.25 billion VM CPU utilization readings are recorded from over five thousand subscriptions.

This trace includes the following key variables: sanitized user, VM, and deployment IDs; timestamp in seconds (recorded every 5 minutes); indicators for when VMs were created and deleted; count of VMs created; deployment size; maximum, minimum, average, and 95th percentile of CPU utilization; VM virtual core count; and VM memory utilization in GBs.

C Productivity Measurement Details

C.1 Details of Measuring Cloud Productivity

As in the main text, index firms by i, jobs by j, and days by t. Firm i assigns job j, which runs for h_{ijt} hours on day t, to VM v_{ij}. Each VM v is defined by a tuple $(c(v), x(v))$, where $c(v) \in C \subset \mathbb{N}$ is the number of cores of VM v and $x(v) \in X$ are the VM's characteristics, which includes the VM's machine type, memory, data center, and operating system.

On day t, we observe n_{ijt} snapshots of CPU utilization $\{u_{ijst}\}_{s=1}^{n_{ijt}}$. By multiplying the utilization with the capacity of the chosen machine, we get n_{ijt} snapshots of the load of each job: $\{\ell_{ijst}\}_{s=1}^{n_{ijt}}$, where $\ell_{ijst} := u_{ijst}c(v_{ij})$. We assume that the load for each job is exogenous — that is, we take it as given that each firm must use the exact same amount of computing power that we observe them using in the data.

On each day, there is a set of VMs available for the firm to choose from. Let V_t be the set of VMs available on day t, and $V_t(x) = \{v \in V_t : x(v) = x\}$ be the set of VMs available on day t that have characteristics x. We also define the outside option "VM" v_0 as the 0-core VM that represents not running a job. Similar to the load, we assume that the characteristics v_t of a VM are exogenous, and we take it as given that the firm chose these correctly. Therefore, we infer the choice set of firm i for job j on day t to be $V_{ijt} = V_t(x(v_{ij})) \cup \{v_0\}$.

We compare the firm's provisioning decision v_{ijt} with the decision of a hypothetical cost-minimizing firm v^*_{ijt}. To do so, we need to model the optimal provisioning process. For our baseline analysis, we assume the following:

Assumption 1. *The cost-minimizing firm provisions based solely on the peak load of the VM, which we take to be the 95th percentile load over the period in which the VM is being provisioned.*

Assumption 2. *If a VM has a peak utilization of under 10% over a given time period, then the firm does not receive any benefit from that job over that time period.*

Assumption 3. *The cost-minimizing firm will downsize a machine only if the peak utilization on that machine would be less than 90%.*

Assumption 4. *After the initial provisioning decision, it is only worthwhile for a firm to change its provisioning decision if a VM will be improperly provisioned over a seven-day period or longer.*

As discussed in the main text, Assumption 1 is relatively standard, both in industry and literature definitions of improper provisioning. Assumption 2 is justified by a 10% peak CPU utilization being explainable by background processes of the CPU and not by

any foreground processes run by the user. Assumption 3 comports with the rightsizing recommendations given by cloud providers to their clients. Finally, Assumption 4 is justified by firms facing sufficiently high switching costs from reconfiguring a job to a new type of VM.[43]

Let $\mathcal{T}_t^k = \{t, t+1, \ldots, t+k-1\}$ be defined as the set of k consecutive days starting with day t. Let $\overline{\ell}_{ij}(\mathcal{T}_t^k)$ be the peak utilization over $\mathcal{T}_t^k$:

$$\overline{\ell}_{ij}(\mathcal{T}_t^k) = \max\left\{\ell : \frac{\sum_{r=t}^{t+k-1}\sum_{s=1}^{n_{ijr}} \mathbf{1}(\ell > \ell_{ijsr})}{\sum_{r=t}^{t+k-1} n_{ijr}} \leq 0.95\right\} \tag{7}$$

Define the peak utilization $\overline{u}_{ij}(\mathcal{T}_t^k)$ analogously. Let T be the length of job j in days and, for ease of exposition, relabel the days so that job j lasts from day 1 to day T.

First suppose $T \geq 7$. Given our assumption, the cost-minimizing firm's decision on each day $t = 1, \ldots, T$ solves:

$$\begin{aligned} v_{ijt}^* &= \arg\min_{v \in V_{ijt}} c(v) \\ \text{s.t.} \quad \min_{r \in \{\max\{1,t-6\},\ldots,\min\{t,T-6\}\}} & \overline{\ell}_{ij}(\mathcal{T}_{t-r}^7)\mathbf{1}(\overline{u}_{ij}(\mathcal{T}_{t-r}^7) \leq 0.1) \leq c(v) - 0.1 \cdot \mathbf{1}(v \neq v_{ij}) \end{aligned} \tag{8}$$

It is easiest to interpret the constraint of (8) in words. The left-hand side of the constraint searches over all of the sets of seven consecutive days that include day t. If day t is part of a seven-day stretch in which the peak utilization is under 0.1, then it is idle, the left-hand side of the constraint will evaluate to zero, and any VM will cover the load over those seven days. In this case, the cost-minimizing firm will choose to deprovision the job, i.e., select $v_{ijt}^* = v_0$ for those seven days. Otherwise, firm i will take the smallest VM that will cover the peak utilization of job j over a seven-day stretch that includes day t. This will always include as a possibility the actual VM that the firm chose, v_{ij}, but could include a smaller VM if the VM is downsizable (there exists a smaller VM with the same characteristics) and the peak load over the seven-day period is small enough to be covered by this smaller VM with at most a peak utilization of 90%. If this indeed is the case, then $v_{ijt}^* \neq v_{ij}$ and we say that job j is overprovisioned over those seven days. In practice, because the number of cores in a given VM nearly always scales by powers of two, a VM will be overprovisioned if a smaller VM exists and its 95th percentile CPU utilization over a seven-day period is under 45%.

[43] In practice, some major cloud providers have processes to reprovision running jobs to VMs of different sizes without any interruption in service. For example, see Amazon Web Services, "Resizing clusters," available at https://docs.aws.amazon.com/redshift/latest/mgmt/rs-resize-tutorial.html, accessed on June 5, 2024. Thus, we view this assumption as conservative.

For VMs that are shorter than seven days — $T < 7$ — we evaluate only the initial provisioning decision and do so over the entire length of the VM. That is, the cost-minimizing firm's provisioning decision solves

$$v^*_{ijt} = \arg\min_{v \in V_{ijt}} c(v) \text{ s.t. } \overline{\ell}_{ij}(\mathcal{T}_1^T)\mathbf{1}(\overline{u}_{ij}(\mathcal{T}_1^T) \leq 0.1) \leq c(v) - 0.1 \cdot \mathbf{1}(v \neq v_{ij})$$

As discussed in the main text, the final productivity measure ω_{ijt} is the ratio between resource usage of the cost-minimizing firm and firm i's actual resource usage on job j on the day t: $\omega_{ijt} = c(v^*_{ijt})/c(v_{ij})$.

C.2 Microfoundation of Compute Productivity as Rescaled TFP

We note that while ω_{ijt} is fundamentally a measure of how effectively firm i solves a cost minimization problem, holding output fixed, it also has an interpretation as a more traditional total factor productivity measure if the production function is Leontief in computing.

Suppose that firm i produces a single product sold at exogenous price p. For ease of exposition, suppose the firm only uses computing for one job j. Let the firm's production function at a given moment in time s be $f_{is}(\ell_{ijst}, z) = \min\{\ell_{ijst}, g_{is}(z)\}$, z are other inputs that are assumed fixed in the short run and $g_{is}(z)$ reflects the amount of computing input that firm i can turn into output at moment s. Assume the price of computing power is linear in the amount of computing power used, and normalize the per-core-hour price to 1. Assume that each moment s is h hours long. Then the firm solves:

$$\max_{\{\ell_{ijst}\}_s, v} \sum_s (p f_{is}(\ell_{ijst}, z) - hc(v)) \quad \text{s.t.} \quad v \text{ satisfies the constraint of (8)} \tag{9}$$

If p is high enough such that the firm does not want to "waste" any fixed input z, then the profit-maximizing firm will choose to set $\ell_{ijst} = g_{is}(z)$ at each moment. In this case, the first part of the maximization problem becomes a constant and the problem can be rewritten as

$$\max_v \sum_s -hc(v) \quad \text{s.t.} \quad v \text{ satisfies the constraint of (8)} \tag{10}$$

which is equivalent to the cost minimization problem in (8). Under these assumptions,

firm i's profits on day t are given by

$$\sum_s p f_{is}(\ell_{ijst}, z) - hc(v_{ij}) = \sum_s p f_{is}(\ell_{ijst}, z) - \frac{hc(v^*_{ijt})}{\omega_{ijt}} \tag{11}$$

where the equality is by definition of ω_{ijt}. The typical TFP would enter as a multiplier of the production function f_{is}; that is, the profit function would be $\sum_s pA_{ijt} f_{is}(\ell_{ijst}, z) - hc(v^*_{ijt})$, and A_{ijt} is TFP. From the formulation in (11), it is clear that the profit function using TFP and the production function using our cost productivity measure are the same up to a rescaling. As such, under these conditions, our productivity measure ω_{ijt} is simply a linear transformation of a TFP measure, where the coefficient is the productivity of the most productive (cost-minimizing) firm.

D Estimation Details

D.1 Details of Productivity Calculation

We use the following procedure to implement the measures represented in Section 4. First, for each possible VM configuration (a combination of VM series, data center, operating system, memory, and cores) a firm could choose, we evaluate whether that configuration is downsizable on each day (another VM configuration of the same type, data center, operating system, and memory is available on that day). We also evaluate whether the configuration is *twice downsizable*, which is defined as there existing a machine that the configuration could be downsized to that is itself downsizable. As discussed in the main text, cores scale in powers of two; therefore, if a VM is twice downsizable, this means there exists a VM with the same machine type, data center, operating system, and memory that has a quarter of the number of cores.

Second, for each VM on each day, using the daily inverse utilization CDF, we compute the peak (95th percentile) CPU utilization for all seven-day streaks that include that day. For VMs that last for fewer than seven days, we compute the peak CPU utilization over the life of the VM. We then assign the productivity measure at the VM-day level using the following hierarchical definition:

1. If a VM-day is part of a seven-day streak with a peak CPU utilization lower than 10%, it is idle and assigned a value of 0.

2. Else if a VM-day is part of a seven-day streak with a peak CPU utilization lower than 20% AND the VM configuration is twice downsizable, it is overprovisioned, with the correct configuration being a VM a quarter of the size, and assigned a value of 0.25.

3. Else if a VM-day is part of a seven-day streak with a peak CPU utilization lower than 45% AND the VM configuration is downsizable, it is overprovisioned, with the correct configuration being a VM half the size, and assigned a value of 0.5.

4. Else a VM-day is properly provisioned and assigned a value of 1.

We also define alternative independent variables that decompose productivity from idleness and from overprovisioning separately. The idleness variable equals 1 if and only if the main productivity dependent variable is equal to 0, while the overprovisioning variable equals 1 if and only if the main productivity dependent variable is greater than 0, but less than 1. To remove the negative mechanical correlation between these two variables, we estimate a firm's overprovisioning inefficiency excluding all idle observations;

that is, overprovisioning inefficiency will be the share of VMs that are overprovisioned, conditional on not being idle.

Once we have defined a dependent variable, we then regress this dependent variable on a fixed effect that is either at the firm, firm-month, firm-month-VM profile, unit, unit-month, or unit-month-VM profile level.[44] We weight by core-hours in order to properly account for the resources used by each VM on each day. Our baseline estimates include no controls, meaning that the resulting fixed effects simply represent the weighted average of the dependent variable; these are the productivity estimates used throughout the main text of the paper. We also regress these accounting for day-of-week fixed effects and an indicator for whether there is a holiday in the region on the given date; day-of-week plus holiday plus machine type fixed effects; day-of-week plus holiday plus machine type interacted with data center region fixed effects; and day-of-week plus holiday plus VM profile fixed effects.[45] Results with these alternative levels of controls are located in Appendix H.

In these regressions, a location normalization is to be made — one can add and subtract a constant from two different fixed effects and arrive at the exact estimates for all units. Our normalization is to make the average productivity according to each of these fixed effect regressions to be the average productivity in the averages without controls. Finally, for all the alternative controls, we verify that the controls form a connected set, and that therefore the fixed effects resulting from the estimation procedure are directly interpretable and comparable with one another.

D.2 Details of Dispersion and Persistence Estimation

This section provides the details of the estimations presented in Table 3.

In our dispersion analysis, we use the firm-month level productivity estimates as detailed in the prior section. Column (1) presents statistics calculated from the raw firm-month level data without any controls. For columns (2-4), we compute the same statistics within each group in the control variables (industry, month, and industry-by-month). After calculating these statistics for each group, we then take the weighted average across the groups, using the number of firms in each group as weights. This weighting approach ensures that smaller industries with fewer firms do not disproportionately influence the aggregate statistics.

In the decomposition analysis in Panel B, the goal is to estimate the variance explained by within and between-firm heterogeneity for the first analysis and within-region across-

[44] A VM profile is a unique combination of VM series, data center, and operating system.
[45] For the firm-month-VM profile and unit-month-VM profile regressions, the final three sets of controls are extraneous because they are nested by VM profile.

region analysis for the within-firm for the second analysis. To do the within-between firm decomposition, we aim to estimate the following decomposition.

$$\text{Var}(\omega_{im}^{k} - \bar{\omega}_{m}) = \text{Var}(\omega_{im}^{k} - \bar{\omega}_{im}) + \text{Var}(\bar{\omega}_{im} - \bar{\omega}_{m})$$

where i denotes firm, k denotes unit, and m denotes month. We further decompose within-firm dispersion into within-firm between-region and between-firm within-region components using:

$$= \text{Var}(\omega_{im}^{kr} - \bar{\omega}_{im}^{r}) + \text{Var}(\bar{\omega}_{im}^{r} - \bar{\omega}_{im}) + \text{Var}(\bar{\omega}_{im} - \bar{\omega}_{m}^{r}) + \text{Var}(\bar{\omega}_{m}^{r} - \bar{\omega}_{m})$$

where r denotes region. For this analysis, we only use multinational firms, which are firms that have units in multiple geographic regions, classified as US, EU, and domestic.

To achieve these decompositions, we use a regression framework and obtain the R^2 from those regressions. Specifically, for the within-firm decomposition, we regress unit-level productivity on firm fixed effects and take the R^2 from that regression as the between-firm component. We repeat the same exercise while including the controls reported in Columns (2-4) of Table 3. For these specifications, we first run the fully saturated regression with the control variables and record the R_0^2. Then, we include firm fixed effects by interacting them with the control variables and record the resulting R^2 as R_1^2. To find the within-firm variation, we calculate $R_1^2/(1 - R_0^2)$, which quantifies the share of variance explained by firm fixed effects after controlling for the specified set of control variables.

The calculation of within- and between-region decomposition is similar. We restrict the sample to the multi-national firms and estimate the contribution of region-fixed effects with or without control variables in the specification.

For persistence results, we use the month-firm level data and regress the productivity on 1-month, 1-year, and 5-year lagged values separately for productivity, idleness productivity, and overprovisioning productivity. Results in Columns (2-4) run these regressions by controlling for the corresponding control variable.

D.3 Details of Learning Estimation

In our learning analysis, we make the following sampling restrictions. First, we remove all firms with an average of less than 50 VM days per month. Second, for the learning analyses that are based on within-cohort variation in productivity over July 2022-June 2023 (including Figure 7 and Table 6), we limit to a balanced panel of firms, i.e., firms that have usage in each month from July 2022 to June 2023.

In all figures in this section, we normalize the productivity estimate for firms in each month by dividing by the productivity estimate of firms in their first month. We then compute the standard errors of the ratio between the productivity estimate of firms in a given month and the productivity estimate of firms in their first month using the delta method. In particular, suppose that $\overline{\omega}_t$ is the expected productivity of firms that are t months old, and σ_t is the standard error. Using the delta method, a first-order approximation of σ_t^{norm}, the standard error of $\overline{\omega}_t/\overline{\omega}_0$, is:

$$\sigma_t^{\text{norm}} \approx \frac{1}{\overline{\omega}_0}\sqrt{\sigma_t^2 - \frac{2\overline{\omega}_t}{\overline{\omega}_0}\text{cov}(\overline{\omega}_0, \overline{\omega}_t) + \frac{\overline{\omega}_t^2}{\overline{\omega}_0^2}\sigma_0^2} \tag{12}$$

To compute an estimated $\widehat{\sigma}_t^{\text{norm}}$, we plug in the estimated average productivities, along with the estimated variances and covariances from the coefficient covariance matrix of regression of productivity on firm experience indicators. In this regression, standard errors are clustered by the firm; this implies a nonzero covariance across months in Figure 7 and Table 6 that is estimated from the data. In Figure 6, the unit of observation is the firm, and therefore, the standard errors are computed using the empirical standard deviations, and the covariance in the estimates is assumed to be zero. The exception is when $t = 0$, in which case we know that $\text{cov}(\overline{\omega}_0, \overline{\omega}_0) = \sigma_0^2$, and therefore this expression simplifies to $\sigma_0^{\text{norm}} = 0$.

D.4 Details of Learning Decomposition Analysis

In our learning decomposition, we restrict our sample to the period from July 2022 to June 2023 to be able to calculate month-to-month productivity growth. Following Melitz and Polanec (2015), we decompose the log productivity change into five components specified in the main text. In some rare cases, firm or unit productivity in a given month is zero. For those months, we set productivity to 0.01 so that we do not drop those observations when we take the logarithm.

In rare cases where an account or machine does not have any usage in a given month, we do not treat those months as entry and exit, but we impute the productivity of that machine series or unit from the previous month, and we set its core hours to zero.

In this decomposition, we first implement the corresponding decomposition at the firm or unit level, depending on the specification, and then average each component at the unit level without weighting by firm experience. The firm experience is measured as of the end of the sample, June 2023.

In calculating Figure 8(b), we first subset the data to firms that are more than three years

old and have an account that began using cloud computing in July 2022 and continued usage through June 2023. These new accounts constitute our sample of new units within experienced firms. We then subset the units of these firms that started using cloud computing after July 2019, ensuring that these units have at least three years of experience by July 2022. We calculate the average productivity of these groups for each month from July 2022 to June 2023 and report the productivity levels by normalizing them relative to the productivity level of new units in July 2022.

For Figure 9(b), we employ a similar approach with one key distinction. Unlike for firms and accounts, we lack separate data on when a firm began using a particular machine. Instead, we infer this information from the usage data. As our 2022 data begins in July, we identify new machines as those first used by a firm in August. This method could introduce a minor error if a firm had previously used a machine before July 2022 but skipped usage in July. However, such cases are rare, and if they occur, they make our results more conservative. After identifying the first use date of a machine, we proceed with the analysis as described in the previous paragraph.

D.5 Measuring Relationship Between Utilization and Electricity

We combine the public cluster usage and power data provided by Google Cloud (GC) to estimate the basic relationship between cloud resources and power utilization. A description of the datasets we use can be found in Appendix B.6. First, we describe the aggregation of the GC cluster data used in the analysis of Section 8.1. Then, in , we describe the simple regression we estimate.

D.5.1 Calculating Utilization by PDU

The GC cluster trace denotes each of the 8 clusters contained in the data by a through h. When compressed, the full size of the cluster trace alone is nearly 2.6 terabytes. Therefore, aggregating and merging the cluster data with the power trace would be computationally expensive. Instead, we focus on cluster a.

In Appendix B.6, we described the task as the unit of observation in the usage data of the cluster trace. Each task in the cluster trace is identified by an index relative to a *collection ID*. A collection is either a set of resources where jobs executing tasks run or are stand-alone jobs submitted directly to the scheduler to run on a machine. Hence, the unique usage observation is identified by the pairing (collection ID, task ID).

Each PDU is uniquely associated with a cluster. Moreover, each PDU supplies power to a specific subset of machines within a cluster. Every task is scheduled on a single machine. As noted in Appendix B.6, CPU utilization (in terms of GCUs- see Appendix

B.6) is reported after being normalized. However, the normalizing factor is the same across all observations. Hence, for each 5-minute interval t and PDU p, we compute CPU utilization at the PDU level as

$$U_{pt} = \frac{\frac{1}{c}}{\frac{1}{c}} \cdot \frac{\sum_{m_{jt} \in \mathcal{M}(p,t)} \sum_{i \in m_{jt}} u_{i,m_{jt}}}{\sum_{m_{jt} \in \mathcal{M}(p,t)} \sum_{i \in m_{jt}} r_{i,m_{jt}}}, \tag{13}$$

where c is the resource specific normalizing factor, $\mathcal{M}(p,t)$ denotes the set of active machines belonging to PDU p at time t, i indexes the pair (collection, task) at t, and $u_{i,m_{jt}}$ and $r_{i,m_{jt}}$ are the used and requested CPU resources of i, respectively. Since c enters the reported measures linearly, it gets canceled out in the computation of U_{pt} to yield a genuine CPU utilization measure in percentage terms. While one would expect utilization to be less than or equal to 100%, the Borg cluster manager allows tasks to utilize available CPU capacity so long as the machine executing the task is not overloaded (Tirmazi et al., 2020). For this reason, utilization can exceed 100%.

D.5.2 Estimating Idle Power Consumtpion

In the computer science and electrical engineering literature, researchers have estimated the relationship between CPU utilization and power consumption through a combination of regression analysis and experimental methods. Experimental studies utilize machines with fixed characteristics and controlled computing environments to generate data on CPU utilization and power consumption.

These studies employ various metering techniques to accurately measure these factors (Kansal et al., 2010; Waßmann et al., 2013; Jiang et al., 2013). Power consumption is modeled as a linear function of CPU utilization and then estimated on the experimental data (Husain Bohra and Chaudhary, 2010; Jiang et al., 2013; Osei-Opoku et al., 2020). These are conventional techniques for estimating the relationship between consumption and utilization; however, Veni and Bhanu (2016) note that non-linear models are better suited for robust power consumption prediction across workloads when one wants to capture better the interaction between CPU utilization and features such as disk space unavailable in our sample.

Since we are primarily interested in the relationship between power consumption and CPU utilization, the regression in Figure 10 is specified as

$$power_{pt} = \alpha_{pt} + \beta \cdot U_{pt} + \varepsilon_{pt},$$

where $power_{pt}$ is the total power utilization of PDU p at time t, α_{pt} represents power consumption when VMs are utilizing no CPU, β is the effect of a percentage increase of CPU utilization on consumption, and ε_{pt} is an error term. The parameters of this regression are estimated via ordinary least squares. As noted above, we estimate $\beta_{pt} = 0.5$, which predicts a 0.5 percentage point (pp) increase in power consumption from a 1pp increase in CPU utilization.

In real-world cluster traces, we rarely observe VMs that consistently use 0% CPU. For example, in our sample from the GC trace, CPU utilization seldom falls outside the range of 30%-50%. Additionally, the completely idle VMs are mostly short-lived, likely created for testing the provisioning or scaling of VMs (Cortez et al., 2017). Therefore, the constant term in the linear model of power consumption is used to extrapolate the level of consumption at 0% CPU. In our regression, we estimate a value of α_{pt} corresponding to 50%. This value is consistent with the experimental literature discussed above.

D.6 Counterfactual Resource Calculations

In this analysis, we calculate the total core-hours that would have been saved if all firms below the benchmark productivity level ω_{it} reached the productivity level $\bar{\omega}_{it}$. For this, we denote the counterfactual productivity:

$$\omega_{it}^{c} = \bar{\omega}_{it} \cdot 1(\omega_{it} < \bar{\omega}_{it}) + \omega_{it} \cdot 1(\omega_{it} \geq \bar{\omega}_{it}) \tag{14}$$

We ask what would be the total core hours needed if all firms had a productivity of ω_{it}^{c}. This calculation is relatively straightforward because it does not depend on whether firms increase productivity through idleness or overprovisioning; the core hours that would be saved will be the same regardless of the mechanism of improvements.

We use $s_{im} = \sum_{j \in J_{im}} c(v_{ij}) h_{ijt}$ to denote the total core hours used by firm i in month m. We categorize these core hours into three types of machine utilization:

$$s_{im} = s_{im}^{i} + s_{im}^{o} + s_{im}^{p} \tag{15}$$

Here s_{im}^{i}, s_{im}^{o}, and s_{im}^{p} denote idle, over-provisioned, and productive core hours, respectively. The output from these different types of machines is denoted by $y_{im}^{i} = 0$, $y_{im}^{o} = 0.5 s_{im}^{o}$, and $y_{im}^{p} = s_{im}^{p}$; thus, $y_{im} = y_{im}^{i} + y_{im}^{o} + y_{im}^{p}$ represents the total output. Let s_{it}^{c} denote the total core-hours needed for firms to produce the same output with the

counterfactual productivity ω_{it}^c. Therefore, we have:

$$\omega_{it} = \frac{y_{im}}{s_{im}}, \qquad \omega_{it}^c = \frac{y_{im}}{s_{im}^c} \tag{16}$$

By taking the ratio, we can find s_{it}^c as:

$$s_{it}^c = s_{it} \frac{\omega_{it}^c}{\omega_{it}} \tag{17}$$

This calculation shows that the mechanism by which firms achieve efficiency gains does not matter since we are counting only core-hours that are used anymore in the counterfactual scenario.

By aggregating firm-level counterfactual core-hours, we can calculate s_t^c as:

$$s_t^c = \sum_i s_{im} \frac{\omega_{im}^c}{\omega_{im}} \tag{18}$$

By further aggregating these over time

$$s^c = \sum_t s_t^c, \qquad s = \sum_t s_t$$

The total resource savings in the economy is the ratio between counterfactual and factual resources:

$$\Delta s = \frac{s - s^c}{s}. \tag{19}$$

D.7 Counterfactual Electricity Calculations

This section calculates how much electricity would have been saved in the counterfactual.

Let s_{ijmt} denote the core-hour for VM j, used by firm i, in month m and let $u_{ijmt} \in [0, 1]$ denote the utilization at time (5-min interval) t. Based on the relationship between utilization, We assume that power consumption takes the following form:

$$p_{imtj} = (0.5 + 0.5u_{imtj})k_m^{max}$$

where k_j^{max} represents the power consumption when machine j is utilized at 100%. This power utilization assumes that when the machine is idle, the power consumption is 50% of maximum power and then increases linearly with utilization. This assumption is based

on the relationship between power and utilization that we estimated in Section 8.1.

We further assume that $k_j^{max} = kc_j$, where c_j is the number of cores of machine j, and we normalize $k = 1$. This assumption is reasonable because the computation power required for a machine typically increases linearly with the number of cores. This functional form is particularly convenient because additivity is preserved under integration, meaning that the core-hours of a machine is a sufficient statistic. In particular, the power consumption of VM m during its duration t_m is given by:

$$\int_t p_{imtj}\, dj = \int_t (0.5 + 0.5u_{imtj})c_{im}\, dt = 0.5c_{imt}T_{imt} + 0.5c_{imt}T_{imt} \int_t u_{imt}\, dt \tag{20}$$

$$= 0.5(1 + \bar{u}_{imt})s_{imt} \tag{21}$$

where $\bar{u}_{imt}$ is the average utilization of machine m, T_{imt} is the duration of VM, and $s_{imt} = c_{imt}T_{imt}$ is the total core-hours of machine m. Furthermore, we can aggregate this at the firm level as follows:

$$p_{it} = \sum_{m(it)} 0.5(1 + \bar{u}_{imt})s_{imt} = 0.5 \sum_{m(it)} s_{imt} + 0.5 \sum_{m(it)} \bar{u}_{imt}s_{imt} \tag{22}$$

$$= 0.5s_{it} + 0.5\bar{u}_{it}s_{it} \tag{23}$$

where u_{it} is the firm i's utilization in month t. This form suggests that a firm's total power requirement depends on the number of core hours they use and the average utilization in a given month. This makes counterfactual power calculations tricky because whether firms improve idleness or overprovisioning will affect both core-hours and the average utilization.

To make progress, we introduce additional notation to separate efficiency gains from changes in idleness and overprovisioning. Let $s_{it}^{i,c}$, $s_{it}^{o,c}$, and $s_{it}^{p,c}$ denote the counterfactual idle, overprovisioned, and productive core hours respectively, and $\Delta s_{it}^{i} = s_{it}^{i} - s_{it}^{i,c}$, similarly for other utilization types. We have that:

$$\frac{s_{it}^{p} + 0.5s_{it}^{o}}{s_{it}} = \omega_{it}, \qquad \frac{s_{it}^{p,c} + 0.5s_{it}^{o,c}}{s_{it}^{c}} = \omega_{it}^{c} \tag{24}$$

Moreover,

$$s_{it}^{p} + 0.5s_{it}^{o} = s_{it}^{p,c} + 0.5s_{it}^{o,c} = y_{it} \tag{25}$$

since we condition on the actual output in counterfactual calculations. This implies that:

$$0.5\Delta s^p_{it} = -\Delta s^o_{it}$$

so an X core-hours reduction in overprovisioned machines should add $X/2$ core-hours of productive VM. Using Equations (24) and (25), we also obtain:

$$\frac{s^p_{it} + 0.5 s^o_{it}}{s_{it}} = \omega_{it}, \qquad \frac{s^{p,c}_{it} + 0.5 s^{oc}_{it}}{s_{it} - 0.5\Delta s^o_{it} - \Delta s^i_{it}} = \omega^c_{it}.$$

where the denominator in the second equation specifies the total core hours used in the counterfactual. This gives:

$$0.5\Delta s^o_{it} - \Delta s^i_{it} = s_{it}\left(\frac{\omega^c_{it} - \omega_{it}}{\omega^c_{it}}\right)$$

This provides an equation, but two unknowns Δs^o_{it} and Δs^i_{it}. So we need another assumption to pin down Δs^o_{it} and Δs^i_{it} separately. For this, we make the following assumption:

$$\frac{s^i_{it} - \Delta s^i_{it}}{s^i_{it}} = \frac{s^o_{it} - \Delta s^o_{it}}{s^o_{it}} \qquad \Longrightarrow \qquad \frac{\Delta s^i_{it}}{s^i_{it}} = \frac{\Delta s^o_{it}}{s^o_{it}}. \tag{26}$$

The underlying idea behind this assumption is that core-hour savings from each mechanism are proportional to the initial waste from each mechanism. Without additional information, this assumption seems reasonable and assumes that firms split efforts equally between different mechanisms.

Now, under the assumption given in Equation 26, one can compute Δs^i_{it} and Δs^o_{it} as follows:

$$\Delta s^i_{it} = s_{it}\left(\frac{\omega^c_{it} - \omega_{it}}{\omega^c_{it}}\right)\left(\frac{s^i_{it}}{s^i_{it} + 0.5 s^o_{it}}\right), \qquad \Delta s^o_{it} = s^i_{it}\frac{s^o_{it}}{s^i_{it}} \tag{27}$$

With this, we know the counterfactual distribution of idle, overprovisioned, and productive machines. We can calculate average counterfactual utilization of firm i at time t, u^c_{it} as:

$$\bar{u}^c_{it} = \bar{u}_{it}\frac{s_{it}}{s^c_{it}}$$

Thus, we can calculate both factual and counterfactual firm-level power requirements:

$$p_{it}^c = s_{it}^c + 0.5\bar{u}_{it}^c s_{it}^c, \qquad p^c = \sum_{it} p_{it}^c,$$

The total power saving in the economy is given by:

$$\Delta p = \frac{p - p^c}{p}.$$

E Robustness Checks

E.1 Robustness to Other Utilization Measures

In cloud computing, network and memory utilization are commonly monitored alongside CPU utilization to measure the performance and efficiency of virtual machines (VMs) and other resources.

Network utilization refers to the amount of data being transferred in and out of a VM or across the cloud infrastructure. High network utilization reflects significant data traffic, while low utilization indicates minimal use of the available bandwidth. Memory utilization measures the amount of allocated memory actively being used by a VM. High memory utilization suggests that a VM is using most of its allocated memory, whereas low utilization indicates that the memory allocation may exceed the needs of the workload.

We focus on CPU utilization because it is the most relevant metric in the industry and the most resource-intensive component of computing infrastructure. CPUs typically consume the majority of power in servers, making their efficient use crucial for minimizing energy consumption and operational costs. Additionally, CPU utilization is a standard measure of performance and efficiency in cloud computing, as it directly reflects how well the processing power is being used. By concentrating on CPU utilization, we align our analysis with industry practices and address the most significant aspect of resource management in cloud environments.

Still, one potential concern with the analysis is that while some firms may show high efficiency based on their CPU utilization, they might be less efficient in their use of memory and network resources, leading to wasted resources in other areas of computing. To address this, we conduct a robustness check to ensure that network or memory utilization does not undermine our CPU utilization results.

We have limited data on memory and network utilization for one-month periods in 2022 and 2023. Using this data, we estimate the correlation between CPU utilization and other utilization measures. The direction of this correlation is unclear beforehand. Some jobs may be memory-intensive, using more memory and less CPU, while others may be compute-intensive, relying more on CPU than memory. This variation could result in a negative correlation between these utilization measures. Conversely, if a job is truly idle, it would likely use neither memory nor CPU, potentially generating a positive correlation between the two measures.

In Figure OA-15, we report the correlation between utilization measures and find that CPU utilization is positively correlated with both network and memory utilization. This

suggests that the firms identified as inefficient in terms of CPU utilization also tend to be inefficient in other dimensions of computing resource utilization.

E.2 Robustness to Time Period of Utilization Measurement

As mentioned in Section 4.1, and detailed further in Appendices C.1 and D.1, we define our productivity and inefficiency using the peak VM utilization over a seven-day period. Doing so ensures that our measure is conservative with respect to the potential costs that short-term provisioning changes represent, particularly given the predictable volatility in load associated with days of the week. In addition, it is consistent with the internal measures of cloud providers.

However, one might be concerned about the sensitivity of our results to this choice. To investigate this, we re-estimate our productivity measures using different periods over which we calculate peak productivity: one-day, three-day, and 15-day. We then repeat our analyses using these alternative productivity measures to ensure that our main results are robust to these alternative productivity definitions.

E.3 Robustness to Machine Controls

As explained in Section 4.1, we first estimate the productivity of individual VMs based on their idleness and overprovisioning and then aggregate these measures at the firm level at different frequencies. In our main specification, we treat all machines the same and simply sum up the VM-level efficiencies to the firm level. One potential concern with this approach is that VMs have different machine characteristics that could affect utilization. We believe focusing on peak utilization mitigates this concern, as peak utilization is less sensitive to machine type. For instance, a memory-intensive or network-intensive workload will naturally have lower average utilization; for example, the CPU will spend idle time waiting for data transfer, and this should not affect peak utilization. However, we still provide several robustness checks to show that our results are not driven by different machine characteristics.

To account for these differences, we aggregate VM-level productivity to firm-level using a weighted regression by controlling for several job characteristics as follows:

$$\omega_{ijt} = \omega_{im} + \beta Z_{jt} + \varepsilon_{ijt}$$

This regression essentially estimates firm-level fixed effects by accounting for systematic productivity differences between machine types in different control bins.

Our first control includes day of the week and holiday fixed effects to account for

temporal variations in productivity differences. For example, if less productive firms, for unrelated reasons, run their jobs on weekends and jobs on weekends are systematically less productive, this specification will accommodate that.

Our second control adds product-fixed effects by interacting day and holiday fixed effects with product machines. We then gradually add more controls, including machine and region-by-machine fixed effects. These machine characteristics address potential differences due to hardware specifications.

In these fixed effect regressions, we can only compare the fixed effects of firms within a connected set (Abowd et al., 1999; Metcalfe et al., 2023). This means firms must be linked directly or indirectly in the graph of firm-to-machine characteristics. In our setting, due to the high number and variety of machines used by firms, we either have all firms in one connected set, or we have one large connected set that covers more than 99% of the firms and a few small connected sets that cover firms using only a few specialized VMs. This allows us to compare almost all firms, even if we control for detailed machine characteristics.

E.4 Robustness to Load Volatility Measures

One important identification threat in our paper is that inefficiency is rational because firms maintain idle capacity when facing a volatile workload to reduce the probability of hitting capacity. Even though we argued that due to the nature of cloud computing and the available tools, there is no reason for firms to maintain idle machines, we still analyze whether productivity measures are correlated with important load volatility outcomes and the probability of hitting capacity. For these reasons, we define the following measures used in the paper. For all measures, we define both the unweighted version and the weigher version weighted by core huts.

- **Standard Deviation:** Measures the dispersion of resource usage from its mean, providing insight into the variability of VM utilization.
- **Coefficient of Variation (CoV):** Calculated as the ratio of standard deviation to mean (sd/mean), offering a standardized measure of dispersion that allows comparison of variability across VMs with different average utilization levels.
- **Fourth Moment:** Calculated by summing the fourth power of deviations from the mean across all utilization bins. This measure is particularly sensitive to extreme values in the distribution.

- **Kurtosis:** Derived from the fourth moment, calculated as (fourth_moment / sd^4) - 3. Quantifies the heaviness of the tails of the resource usage distribution relative to a normal distribution. Higher values indicate more frequent extreme fluctuations in VM utilization.
- **Tail Event:** Defined as usage exceeding a threshold set at (mean + 4*sd). This identifies extreme usage events that are significantly above the average utilization.
- **Tail Event Probability:** Represents the likelihood of extreme resource usage events, calculated as the proportion of time VM utilization exceeds the tail threshold.

We also calculate the load faced by the firm at the VM and at the daily level, which quantifies the aggregate compute demand, assuming constant code efficiency and IT infrastructure. To compute this load, we integrate the area under the CPU utilization curve.

E.5 Robustness: Measurement of Downsizability

An important aspect of measuring compute productivity is the concept of downsizability: identifying alternative VMs that a firm could select if a VM is overprovisioned. When discussing downsizability in VMs, it is important first to establish the criteria for an appropriate substitute with fewer cores. A good substitute VM should maintain equivalent performance across all specifications, except having a reduced number of CPU cores.

Several key factors should be considered when defining a substitute VM. These are primarily memory, machine type, operating system, region, and data center. For example, the memory capacity should remain the same or be higher to ensure that the job can run in the alternative VM. The operating system should also remain the same to maintain software compatibility. Another but less clear dimension is the machine type. Machines are different in many dimensions, including type, manufacturer, and series. In principle, the same job can be run on different hardware versions and even on hardware from different manufacturers. However, firms might prefer to maintain the same machine types for consistency, performance predictability, and ease of management.

Another critical factor when considering VM downsizing is a geographical region or data center. The location of the data center might be important as firms tend to choose data centers close to their customers or employees to reduce latency (Greenstein and Fang, 2020). Moreover, firms might prefer to use a particular data center because their data is stored there. Finally, regulatory compliance and data sovereignty requirements can dictate the need for specific regional or data center locations.

Considering these factors and the nuanced nature of downsizability, we define various levels of downsizability. In each measure, we maintain the constraint that the alternative VM should have the same memory and operating system while allowing for variations in machine type and location.

- "Data Center-Machine Series-OS-Memory" Downsizing: This is the most restrictive measure, requiring VMs to be in the same data center, machine type, and series, with identical OS and memory.
- "Region-Machine Series-OS-Memory" Downsizing: This variation relaxes the data center requirement to the regional level while maintaining other restrictions.
- "Region-Machine Type-OS-Memory" Downsizing: This measure allows for different machine series within the same region, type, OS, and memory specifications.
- "Region-OS-Memory" Downsizing: This variation permits downsizing across different machine types within the same region, maintaining OS and memory consistency.
- "OS-Memory Downsizing": The least restrictive measure, allowing downsizing across different regions, only requiring the same OS and memory specifications.

In our baseline specification, we choose "Region-Machine Series-OS-Memory Downsizing" to balance the need for consistent performance with the flexibility of using different machine types within the same region. However, we also conduct robustness checks using other downsizability measures to ensure the reliability of our findings.

E.6 Correlation Between Idleness and Over-provisioning Productivity

In this robustness check, we analyze the relationship between idleness and overprovisioning productivity. A positive relationship between these two productivity measures would suggest that inefficiency is not driven by a particular mechanism that generates only one type of inefficiency. For example, one explanation for inefficiency could be that a firm's workload is volatile for a given job, so firms overprovision to ensure they can meet additional demand. However, this explanation is less likely to account for idleness because an efficient firm could easily manage volatility across VMs using available tools.

For this robustness check, we regress overprovisioning productivity on idleness productivity using firm-month-level data, controlling for industry and time fixed effects. This regression yields a coefficient of 0.064 with a standard error (clustered by firm) of 0.004. This suggests that firms with idle machines tend to have overprovisioned machines. The

result provides suggestive evidence that underlying firm-level factors drive both compute productivity measures rather than simply mechanical explanations.

E.7 Relationship between Productivity and Firm Exit

There is a large literature showing that less productive firms are more likely to exit. In this section, we test this hypothesis by investigating the relationship between firm productivity and the probability that firms leave the cloud.

For this exercise, we use data from 2022 and 2023. We classify each firm as "high" or "low" productivity based on whether they are above or below median productivity in their industry in 2022. Then, we look at the exit probability for these groups from January 2023 to June 2023. Our data ends in June, so to be conservative, we consider a firm to have exited if we do not see any VM deployment one month before the sample period ends. We find that low-productivity firms are 60% more likely to leave the cloud than high-productivity firms.

E.8 External Validity: Dispersion in Public Traces

Although the data used in our main analysis comprises a large compute trace from a global cloud provider, one concern might be that our analysis lacks external validity. To the extent that public compute traces are characteristic of computing environments outside of our sample, we show that our results on productivity dispersion summarized in Figure OA-16 hold out-of-sample.

To do this, we use the public compute traces described in Appendix B. These public traces differ from our data in both structure and duration. Therefore, for each trace used below, we describe the construction of the sample we used to analyze productivity.

For both traces, we either directly applied the assumptions used in cleaning our own data or selected observations that resembled the operative objects of our analysis as closely as possible. These steps ensure that the analysis of productivity in these public traces serves as a valid test of our findings.

E.8.1 Azure Cloud Trace

The 2019 Microsoft Azure (henceforth, Azure) data traces one month of virtual machine (VM) utilization readings and characteristics. For utilization, we observe the average and maximum CPU utilization over every five-minute interval within the VM's lifetime. For characteristics, we observe requested cores and memory (in gigabytes), as well as the timestamp of when the VM was created and deleted. We also observe a "machine category"

variable that describes whether a VM is “delay insensitive," “interactive," or “unknown".

As expected, we observe core and memory request levels that mainly scale by a factor of two. For cores, we observe request levels of 2, 4, 8, 24, and 30. For memory, we observe request levels of 2, 4, 8, 32, 64, and 70. As in our main analysis, we define downsizeability in terms of cores for a given set of machine characteristics. In particular, for a VM's given level of memory request and machine category, the VM is downsizeable if another with the same memory and category exists with half as many cores. By definition, no VM with two cores is downsizeable.

Also, as in our main analysis, we consider productivity based on the 95^{th} percentile of max CPU utilization. Unlike our data, the Azure trace only contains obfuscated user identifiers. Hence, we analyze the users' dispersion of productivity.

Before computing productivity, we clean the trace data in accordance with the sampling done on our main data. We drop all users observed to have less than 10 VMs throughout the duration of the trace, resulting in a loss of 2.8% of users. We also remove VMs with a lifetime of less than 20 minutes, resulting in a more significant loss of about 39% of the available VMs in the data. This finalized sample contains information on the resource usage of 3,503 users with 1,632,952 VMs.

Next, we define for VM j of user i on day t a productivity measure $\omega_{ijt} \in [0, 1]$. This definition follows that of ω_{ijt} in Section 4. Using this, we can aggregate the VM-level productivity to a user-day-level productivity. The productivity for user i on day t with jobs J_{it} is given by

$$\omega_{it} = \frac{\sum_{j \in J_{it}} \omega_{ijt} ch_{ijt}}{\sum_{j \in J_{it}} ch_{ijt}}, \tag{28}$$

where ch_{ijt} is the core-hours of job j on day t. Unfortunately, the Azure trace only contains timestamps in seconds, and it is unknown when the trace began. Hence, we have no mapping between timestamps and particular dates. Thus, we determine the days where observations fall to be the modulus of the timestamp divided by the number of seconds in a day.

Appendix Figure OA-16(a) plots the distribution of user-day level productivity throughout the duration of the Azure Trace. There is a significant mass at the one-half productivity level, representing the days when users needed only half of the cores they provisioned for their VMs. This distribution is broadly similar to that derived from our sample in OA-16. Moreover, the distribution of productivity is skewed more toward idleness than in OA-16, bolstering the observation that overprovisioning is pervasive in cloud computing.

E.8.2 Google Cluster Trace

Similar to the Azure trace, the cluster data provided by Google (henceforth, GC) traces one month of CPU usage and characteristics from May 2019. Whereas the Azure trace concerned genuine VMs, the cluster trace concerns cluster usage by jobs submitted by Google engineers and services (see Appendix section B). We use the detailed information in the trace provided by GC to focus on jobs resembling the structure of VMs as closely as possible.

As described in Appendix Section B, the task is the fundamental unit of observation in the usage data in this trace. Tasks are executed as instances of jobs that either run independently and directly on a physical machine or as part of a *alloc set*, which represent sets of fixed resources. In the GC trace, jobs and alloc sets are referred to as *collections*. For each collection, we observe whether auto-scaling is enabled and whether, if enabled, it is constrained. For this data, we take our VM-like objects as collections without vertical scaling. Henceforth, we will refer to these collections as VMs. We focus on these VMs because the provisioning decisions for CPU and memory are made by the user rather than the cluster manager. In this way, this subset of collections helps us focus on compute resources most similar to our sample and enables us to focus on user-made provisioning decisions.

The GC trace contains observations of nearly 5.2 million collections. Only about 360,000 (6.8%) of these have vertical scaling disabled. Since these VMs run on a cluster, they start using compute resources when scheduled on a machine. Thus, we consider the lifetime of the VM as beginning at the scheduled time. We remove all collections that do not have an explicit scheduling event. This preserves 99.5% of the collections. We also know exactly when the trace started and ended, so we do not deal with the timestamp-to-date conversion difficulties as with the Azure trace.

Some VMs have multiple scheduling events observed. These cases correspond either to a VM that failed and was restarted or a VM that was booted from a machine due to a higher-priority VM needing to be scheduled. For these kinds of VMs, we take the minimum scheduling event observed as the start time of the VM. As in our main analysis, we remove VMs with a lifetime of less than 20 minutes and all users with less than 10 VMs. This results in a sample of 24,909 VMs. While this is a small subset of all available collections, it was created so that the VM-like objects we analyze resemble the VMs in our data as much as possible.

In our data and the Azure trace, we observe core and memory requests that scale by a factor of two. This is not the case in this trace. Since our VMs are cluster jobs, they

are able to request memory in terms of bytes rather than just gigabytes, and therefore, requests can differ at a much more granular level. Given the availability of provisioning resources at such a granular level, users are able to provision VMs efficiently at virtually any level of compute usage. Thus, we consider all VMs to be downsizeable, and hence, the productivity of a given VM will be solely based on its CPU utilization.

Another aspect of the setting in the GC trace is that the cluster manager used by GC's clusters allows VMs to use available CPU resources on the physical machine from which it is drawing compute so long as that machine is not at capacity. From the analysis in Section D.7, we know that the utilization of machines in the GC trace is well below capacity throughout the sampling period. As a result, virtually all VMs will not see their workload throttled from hitting 100% utilization of their requested resources. This fact, along with the assumed downsizeability of all VMs, should attenuate the inefficiencies observed in the more traditional VM settings of our data and the Azure trace. Indeed, in Appendix Figure OA-16(b), we see that productivity is skewed toward the right, but there is still a large dispersion in the productivity of users.

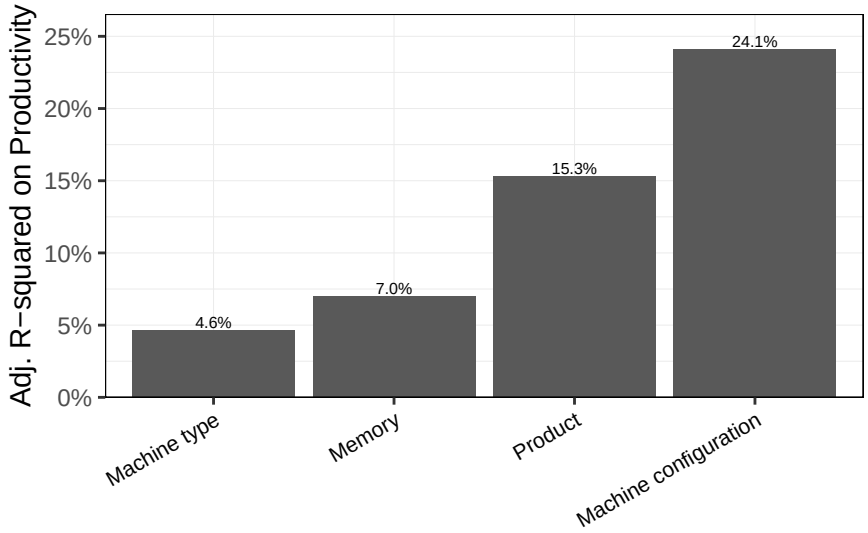
As above, we define the productivity of the VM j of user i on the day t by the measure ω_{ijt}. In this case, ω_{ijt} is zero for VMs with CPU utilization under 10%, one-half for VMs with between 10% and 45% utilization, and one for VMs with greater than 45% utilization. As normal, we consider the 95$^{\text{th}}$ percentile of maximum CPU utilization. With this, we aggregate VM level productivity to the user-day level using a similar notation as above:

$$\omega_{it} = \frac{\sum_{j \in J_{it}} \omega_{ijt} ch_{ijt}}{\sum_{j \in J_{it}} ch_{ijt}}. \tag{29}$$

Appendix Figure OA-16(b) shows that productivity is skewed toward efficient provisioning. This is likely due to the opportunity for granular provisioning, as well as the ability of VMs to have CPU utilization over 100%, as discussed above. With these caveats, we still observe a similar pattern in the dispersion of productivity by the user to our main analysis and the analysis of the Azure trace. In particular, we observe a large mass at the productivity level of one-half where users were able to reduce their CPU requests by 50% and still maintain their workloads.

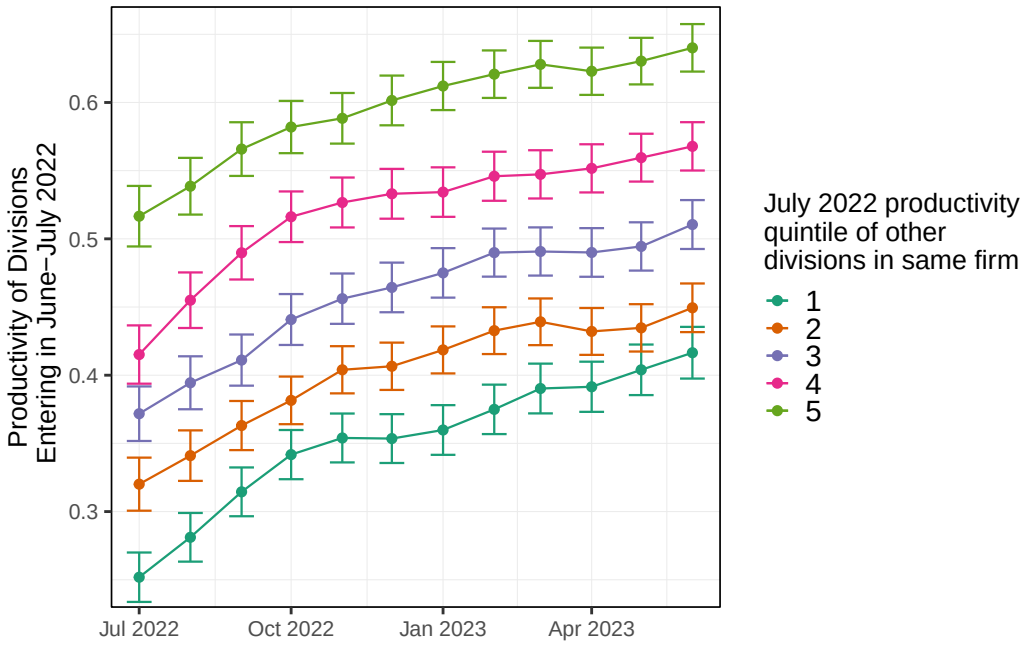
F Additional Figures

Figure OA-1: Explanatory Power of Virtual Machine Characteristics



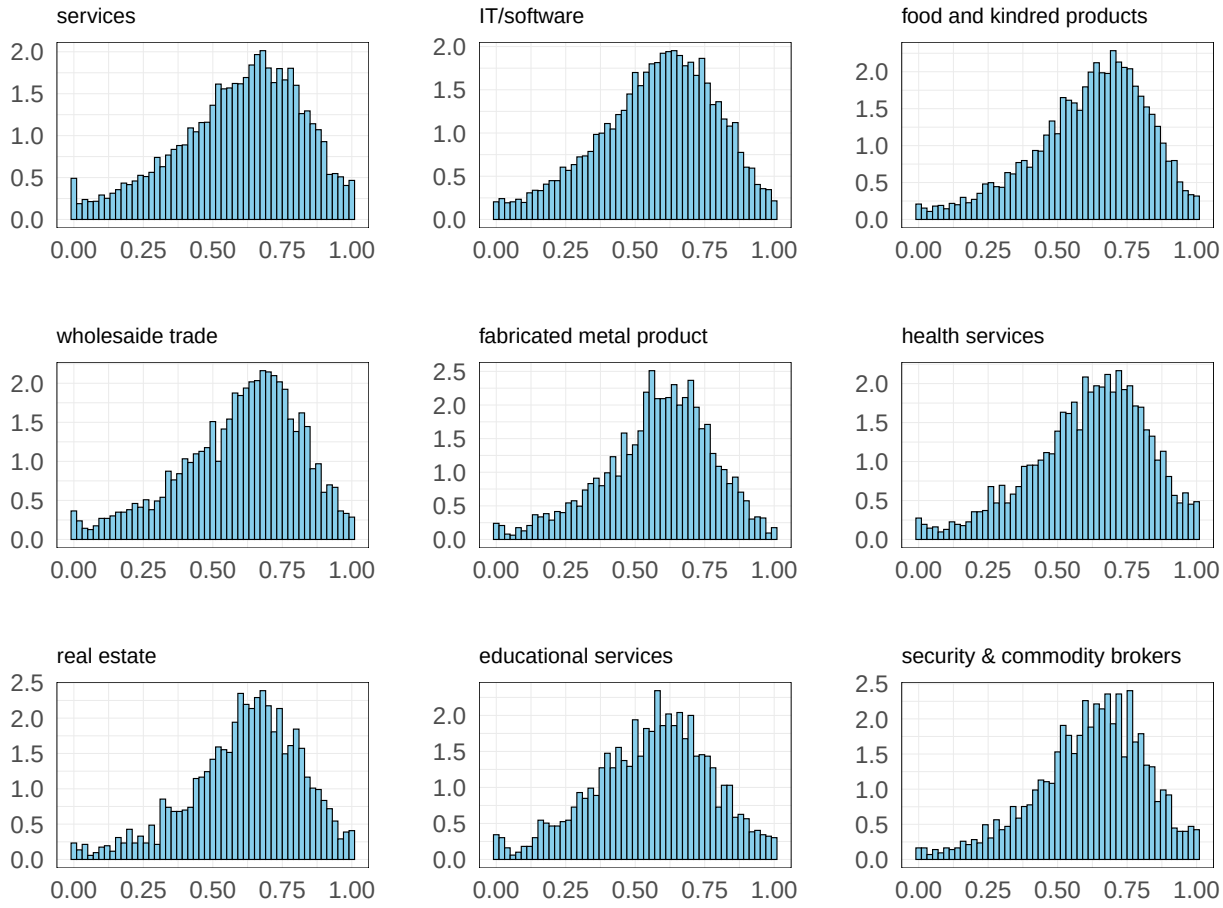
Notes: Adjusted R^2 from the regression of VM-day level productivity on increasingly detailed levels of fixed effects. Fixed effects included in bars to the right always nest the fixed effects used in preceding (left) bars.

Figure OA-2: Within-Firm Transfer



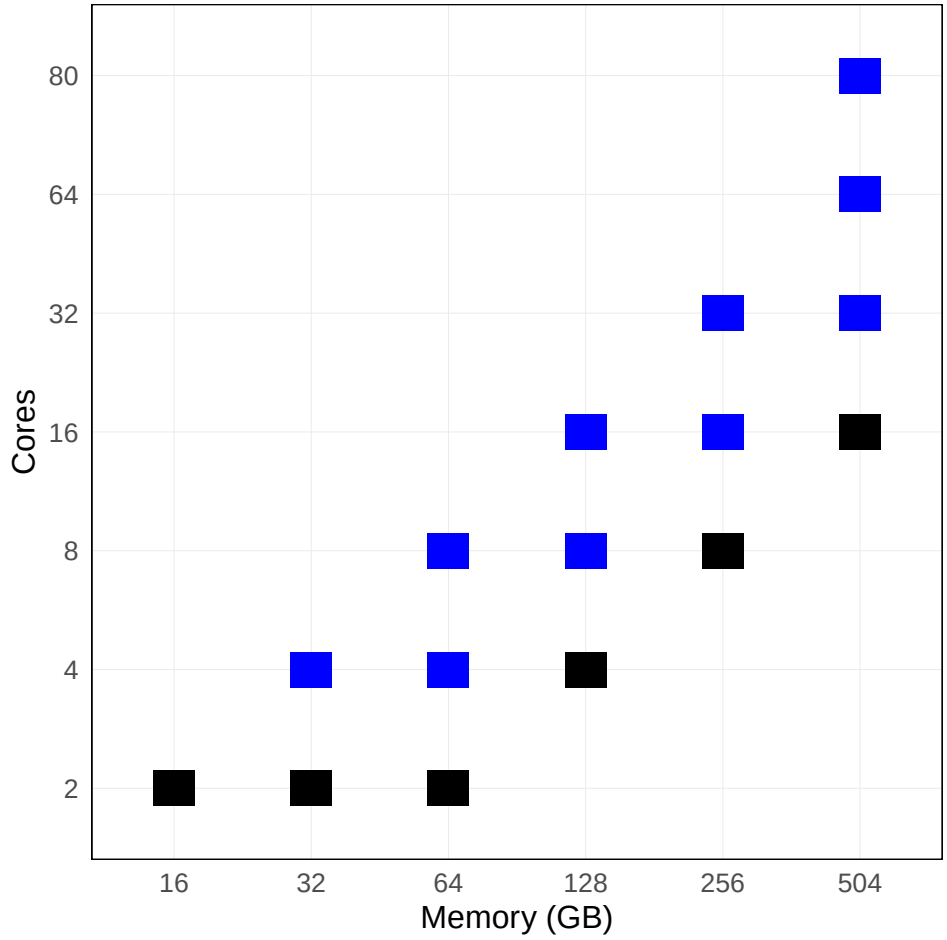
Notes: In panel (b), we group firms into five evenly sized groups based on the productivity of existing units in July 2022 and then plot the average productivity of the new units over time.

Figure OA-3: Productivity Dispersion by Industry



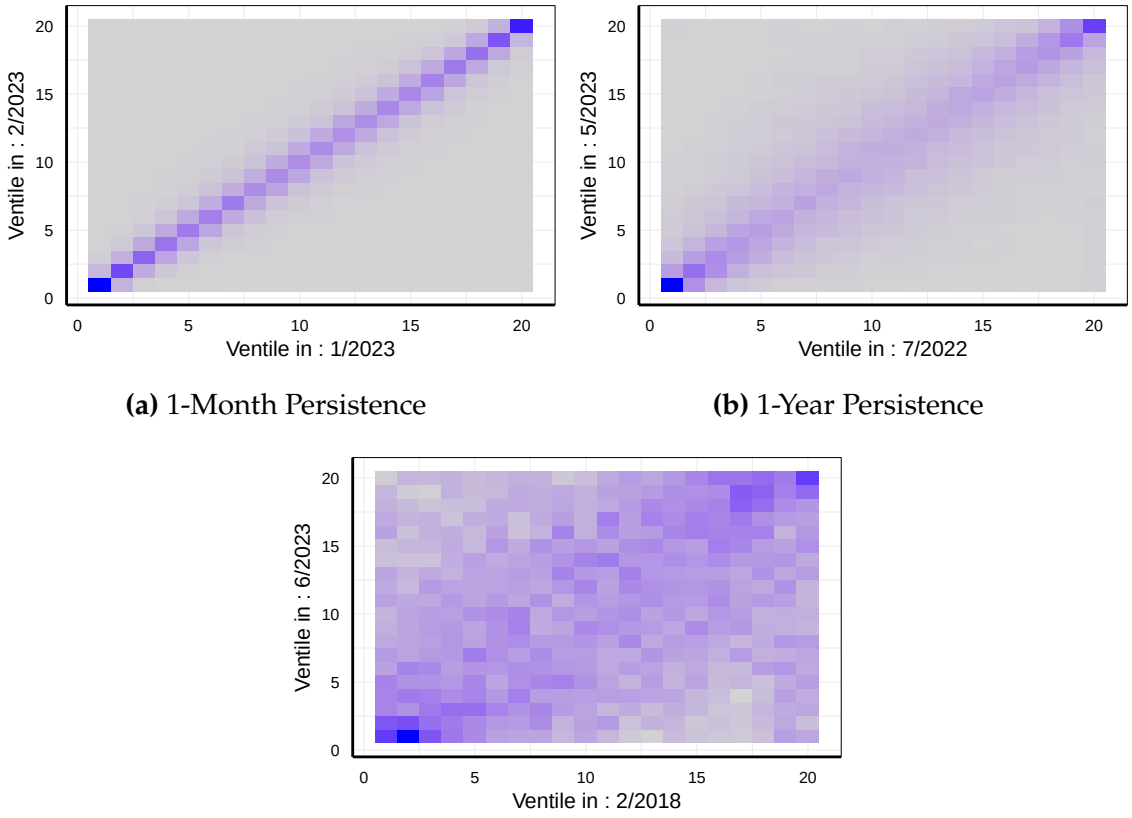
Notes: This figure displays the productivity distribution by industry, calculated in the same way as in Figure 3. Industry classification is based on 1-digit SIC codes, which are converted from the provider's internal industry classification.

Figure OA-4: Downsizability Example: Memory and Core Combinations



Notes: This figure represents one specific VM family example from our data, where we list all combinations of available memory (GB) and cores within this family. Each point represents a specific VM configuration. The green-colored points indicate downsizable machines, where an alternative VM exists with the same memory capacity but fewer cores.

Figure OA-5: Persistence of Productivity in the Short, Medium, and Long Run



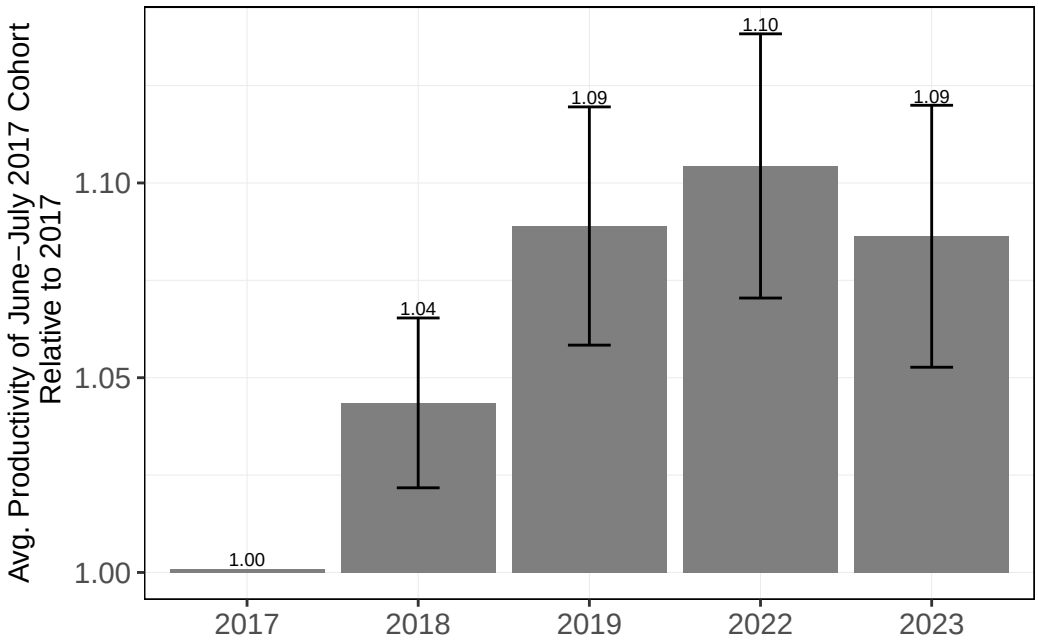
(a) 1-Month Persistence

(b) 1-Year Persistence

(c) 5-Year Persistence

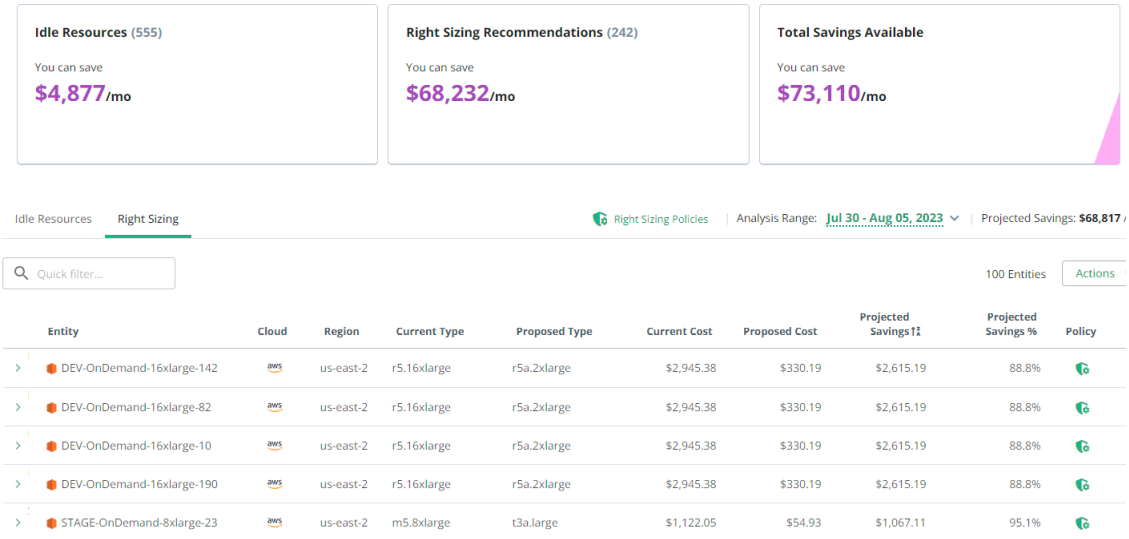
Notes: Presents heatmaps illustrating productivity persistence across three different time horizons: (a) 1-Month, (b) 1-Year, and (c) 5-Year. Each heatmap's axes are divided into 20 equally sized bins, representing the ventilates of the productivity distribution. The x-axis shows the ventile at the start of the period, while the y-axis shows the ventile at the end of the period. Each cell's color intensity corresponds to the frequency of firms moving from one ventile to another over the specified time horizon. Panel (a) depicts 1-month persistence from January 2023 to February 2023, panel (b) shows 1-year persistence from July 2022 to May 2023, and panel (c) illustrates 5-year persistence from February 2018 to June 2023.

Figure OA-6: Productivity of Firms Joining in June-July 2017 Over Six Years

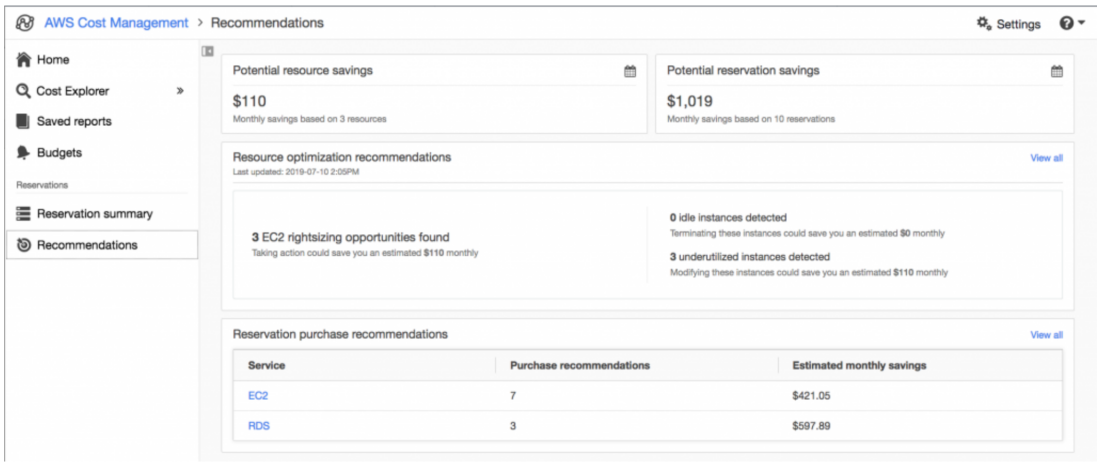


Notes: The average productivity in 2017 is normalized to 1. The crossbars represent the 95% confidence interval. Standard errors are clustered by firm To be included, a firm must have had its first usage in June-July 2017 and had usage in or after May 2023.

Figure OA-7: Idleness and Overprovisioning Detection Tools from Industry



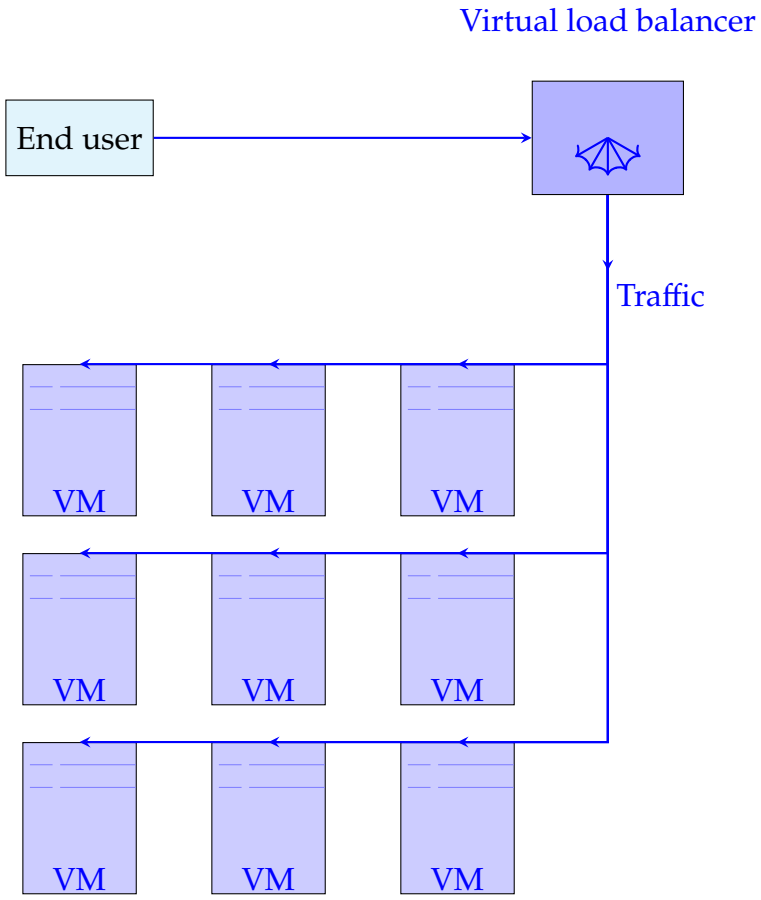
(a) Virtana Cost Management Tool



(b) AWS

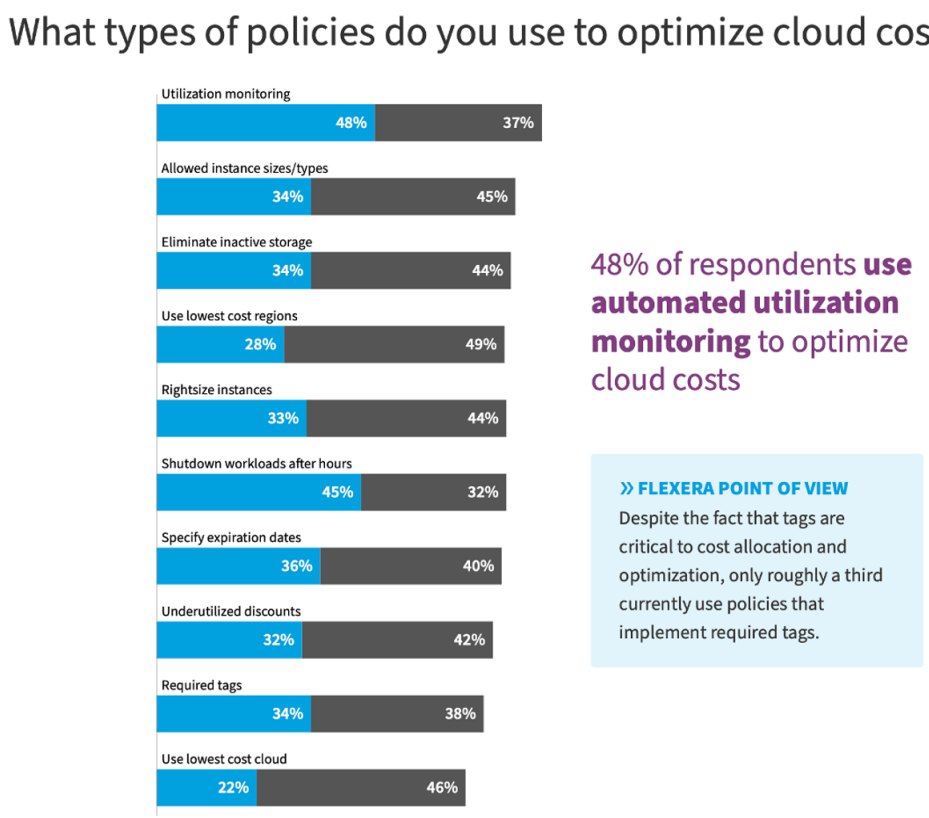
Notes: This figure presents two examples of tools for detecting idleness and overprovisioning in cloud computing. Panel (a) shows a dashboard from cloud optimization startup Virtana, taken from https://www.virtana.com/products/cloud-cost-management while Panel (b) displays the cost management interface of AWS obtain from https://aws.amazon.com/blogs/aws-cloud-financial-management/launch-resource-optimization-recommendations/.

Figure OA-8: Representation of Load Balancer

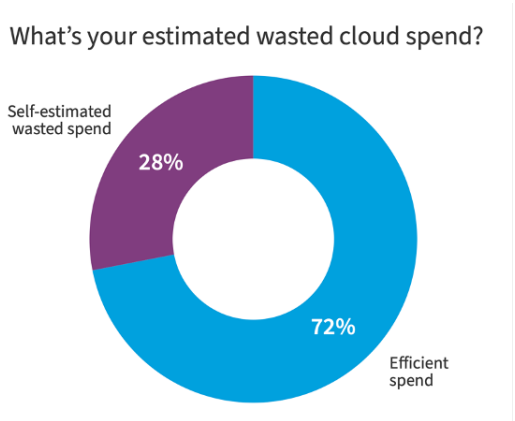


Notes: This figure illustrates a virtual load-balancing architecture in cloud computing. It depicts the flow of traffic from end users through a virtual load balancer, which then distributes the requests across multiple virtual machines (VMs). The load balancer directs traffic (indicated by a blue arrow) to three rows of VMs, each row containing three VMs. This architecture is designed to optimize resource utilization and improve system performance by efficiently distributing incoming requests across available compute resources.

Figure OA-9: Surveys About Cloud Utilization



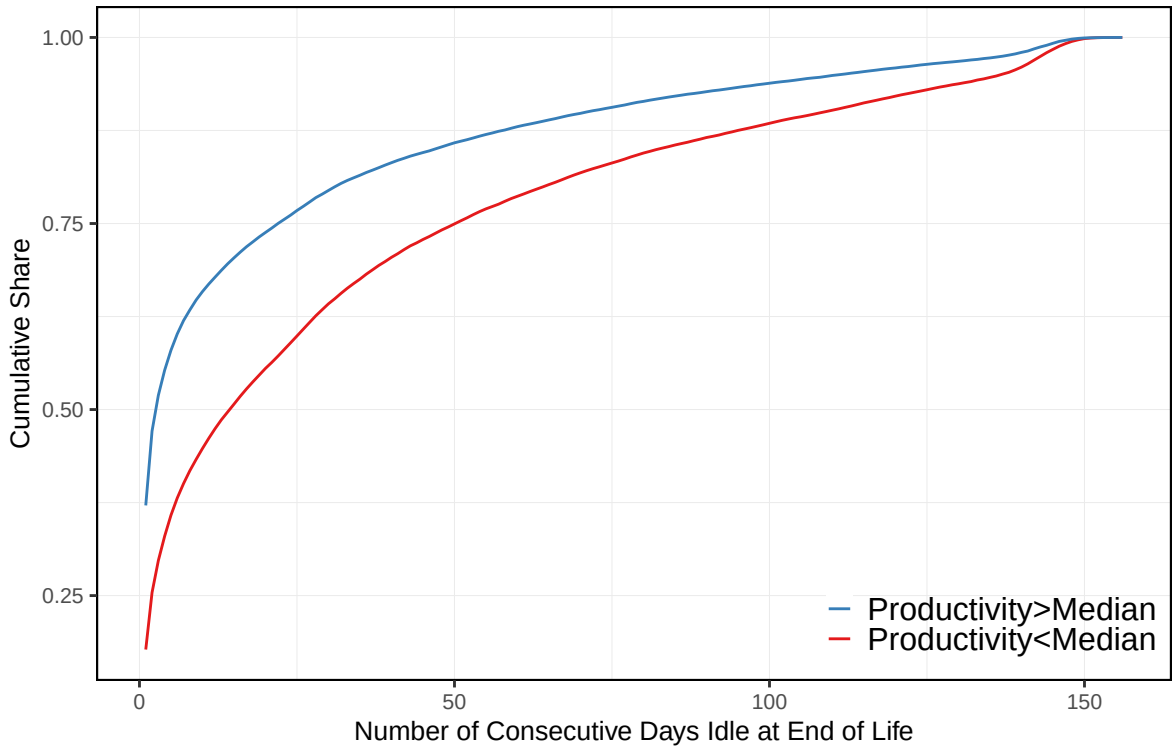
(a) Survey Response to a Question about Cloud Optimization tools



(b) Survey Response to a Questions about Spending

Notes: This figure is a screenshot from a survey conducted by Flexera titled "State of the Cloud" (Flexera, 2023). It shows the responses to two questions asked in the survey. Panel (a) illustrates the types of policies companies use to optimize cloud costs, while Panel (b) displays the respondents' estimates of their wasted cloud spend.

Figure OA-10: Number of Days Idle at the end of VM-High and Low Productive Firms



Notes: Firms are categorized as high or low productivity according to 2022 data. Then, for all multi-day VMs that end with at least one consecutive idle day in a row and that end before the end of our sample, we plot the CDF of the number of consecutive days each VM is idle at the end of its life for low and high productivity firms.

G Additional Tables

Table OA-1: Comparison of Virtual Machine Types Offered by Major Cloud Providers

VM Type	AWS	Azure	GCP
General Purpose	M5, T3	B-series, Dsv3-series	E2, N1, N2, N2D
Compute Optimized	C5	Fsv2-series	C2
Memory Optimized	R5, X1	Esv3-series, Mv2-series	M1, M2
Storage Optimized	I3, D2	Lsv2-series	-
GPU/Accelerated Computing	P3, G4	NC-series, NV-series	A2
High Performance Compute	-	H-series	-
Shared-core	-	-	f1-micro, g1-small

Notes: This table summarizes the main types of virtual machines offered by Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP). The categories are general and may not be exhaustive. Each provider offers multiple sizes and variants within each type. "-" indicates that the provider doesn't have a direct equivalent or the information wasn't specified in the given context.

Table OA-2: Virtual Machine (VM) Types, Key Considerations, and Ideal Applications

VM Type	**Key Considerations**	**Ideal For**
General Purpose	Cost-effective, balanced CPU, memory, temporary storage	Web servers, application servers, development environments, small to medium databases
Compute Optimized	High core counts, faster CPUs	Scientific computing, HPC, video editing, simulations
Memory Optimized	Large RAM capacities	Databases, caching layers, in-memory analytics
Storage Optimized	Local SSDs, high I/O performance	Large databases, data warehousing, Big Data analytics, real-time applications
GPU	Diverse GPU types and configurations	Machine learning, deep learning, video editing, scientific simulations
High-Performance	Exceptionally high compute power, massive memory, ultra-fast storage	Scientific modeling, simulations, weather forecasting

Notes: Source: https://www.cloudoptimo.com/blog/the-ultimate-guide-to-choosing-the-right-azure-virtual-machine/.

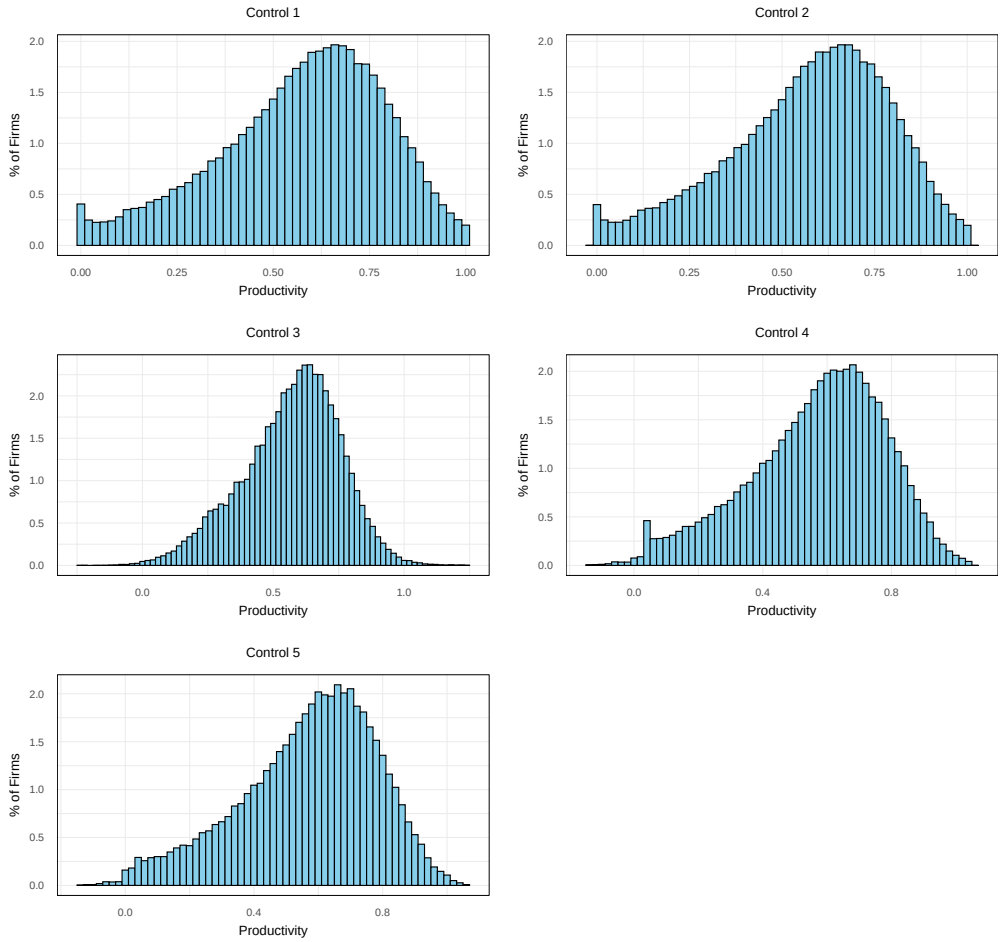
Table OA-3: Cost Optimization Tools by Cloud Provider

Cloud Provider	Cost Optimization Tool	Description
AWS	AWS Cost Explorer	Interface to view costs, usage, and ROI for AWS services, with data for the past 13 months and forecasting capabilities.
	AWS Budgets	Allows setting and enforcing budgets for AWS services, with notifications when budgets are exceeded or reached.
	AWS Trusted Advisor	Provides automated recommendations for cost optimization, including EC2 reserved instance optimization and idle resource identification.
	Amazon CloudWatch	Monitoring service that can set alarms based on metrics, commonly used for cost optimization by identifying underutilized resources.
	AWS Instance Scheduler	Automates starting and stopping of EC2 and RDS instances based on defined schedules to save costs.
	AWS Pricing Calculator	Estimates the cost of use cases on AWS, helping to model solutions and explore pricing points before deployment.
Azure	Azure Cost Management and Billing	Provides cost analysis, budgeting, and recommendations for cost optimization, integrated with Azure portal.
	Azure Advisor	Offers personalized best practices and recommendations to optimize Azure resources, including cost optimization.
	Azure Pricing Calculator	Helps estimate costs for Azure services and solutions, allowing users to model and forecast expenses before deployment.
GCP	Google Cloud Cost Management	Includes tools for cost visibility, budgeting, and recommendations to optimize cloud spending.
	Google Cloud Pricing Calculator	Estimates costs for Google Cloud services, allowing users to model and forecast expenses before deployment.
	Google Cloud Recommender	Provides recommendations for cost optimization, including rightsizing VM instances and identifying idle resources.
	Google Cloud Budgets and Alerts	Allows setting budgets and receiving alerts when costs exceed predefined thresholds, integrated with Google Cloud Console.

Notes: This table summarizes the main cost optimization tools offered by Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).

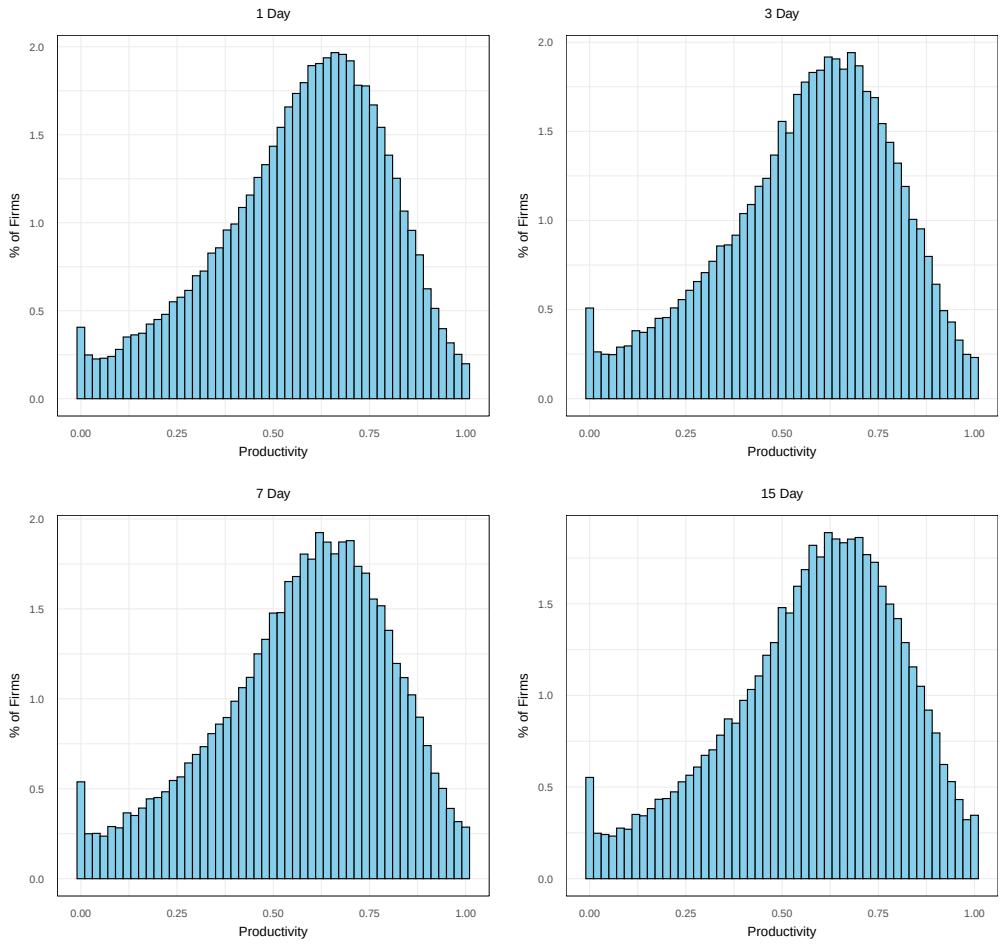
H Robustness Results

Figure OA-11: Robustness: Dispersion Histogram With Different Machine Controls



Notes: This figure presents the distribution of firm-level productivity estimates using productivity measures that control for different machine characteristics, as detailed in Section E.3. The histograms show the dispersion in productivity under various control specifications, including the day of the week, holiday, product ID, and machine type.

Figure OA-12: Robustness: Dispersion Histogram With Different Peak Definition (1/3/15 days)



Notes: This figure presents the distribution of firm-level productivity estimates using productivity measures that control for different days of measurement of peak utilization as detailed in Appendix E.2.

Table OA-4: Robustness: Dispersion and Persistence of Productivity with Controls

	No Controls (1)	Day Holiday (2)	Day, Holiday, Product (3)	Day Holiday Machine Char. (4)	Day Region Machine Char. (5)
			Panel A. Dispersion		
Dispersion:					
Mean	0.602	-	-	-	-
Median	0.626	-	-	-	-
10-90th perc ratio	3.186	3.186	2.870	3.142	3.118
Inter Quartile Range	1.669	1.669	1.601	1.654	1.649
Within-Firm:					
Within-firm	28.152	28.163	24.824	28.081	28.060
Between-firm, within-industry	71.848	71.837	75.176	71.919	71.940
Within-Firm-Between-Region:					
Within-region	5.599	5.611	5.432	5.545	5.602
Between-region, within-industry	94.401	94.389	94.568	94.455	94.398
			Panel B. Persistence (AR(1) Coefficients)		
1-month persistence:					
Productivity	0.933	0.933	0.918	0.930	0.930
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Idleness	0.934	0.934	0.920	0.933	0.933
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Overprovisioning	0.911	0.911	0.900	0.907	0.906
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
1-year persistence:					
Productivity	0.645	0.645	0.594	0.640	0.640
	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)
Idleness	0.659	0.659	0.614	0.659	0.659
	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)
Overprovisioning	0.599	0.598	0.549	0.585	0.584
	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)
5-year persistence:					
Productivity	0.321	0.321	0.259	0.338	0.338
	(0.004)	(0.004)	(0.004)	(0.004)	(0.004)
Idleness	0.332	0.332	0.243	0.333	0.333
	(0.004)	(0.004)	(0.004)	(0.004)	(0.004)
Overprovisioning	0.104	0.104	0.140	0.120	0.116
	(0.003)	(0.003)	(0.004)	(0.003)	(0.003)

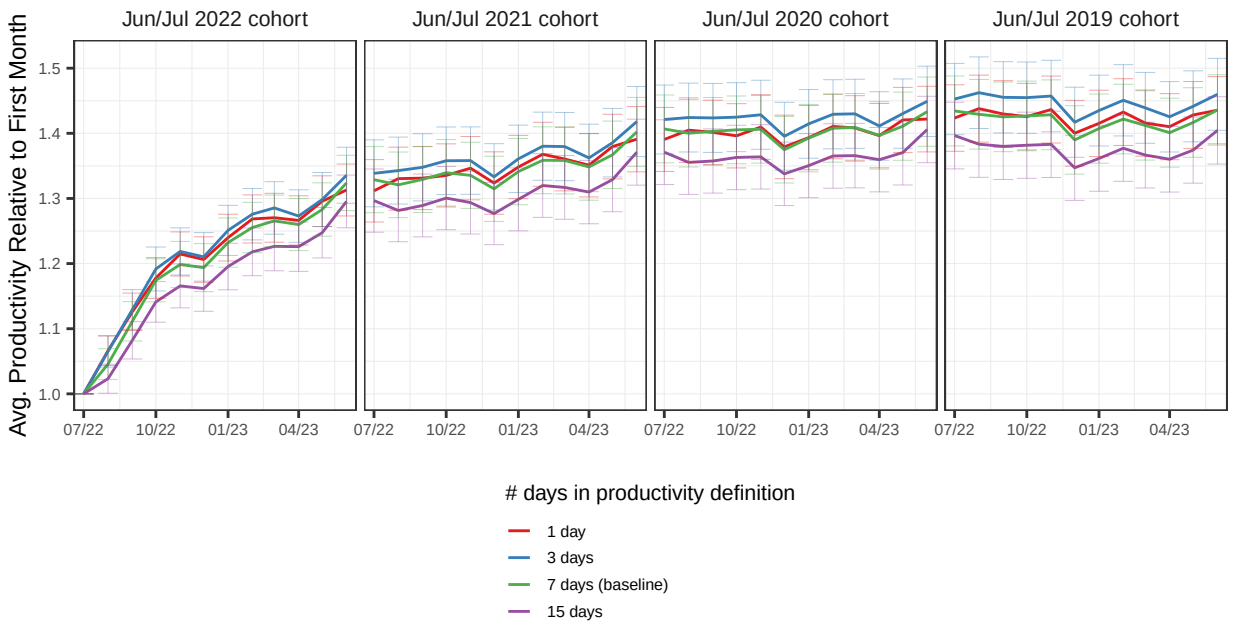
Notes: This table reports the dispersion and persistence of productivity measures across different specifications. Panel A presents the dispersion of compute productivity. Panel B shows the persistence of productivity, idleness, and overprovisioning measures with 1-month, 3-month, and 5-month autoregressive (AR(1)) coefficients, including their standard errors in parentheses. Specifications are (1) raw, (2) day-of-week and holiday controls, (3) day-of-week, holiday, and product ID controls, (4) day-of-week, holiday, and machine type controls, (5) day-of-week, holiday, data center region, and machine type controls.

Table OA-5: Robustness: Dispersion and Persistence of Productivity Peak Def.

	1 Day (1)	3 Days (2)	7 Days (3)	15 Days (4)
		Panel A. Dispersion		
Dispersion:				
Mean	0.600	-	-	-
Median	0.625	-	-	-
10-90th perc ratio	3.027	3.187	3.186	3.154
Inter Quartile Range	1.650	1.672	1.669	1.660
Within-Firm:				
Within-firm	28.620	28.213	28.152	28.012
Between-firm, within-industry	71.380	71.787	71.848	71.988
Within-Firm-Between-Region:				
Within-region	5.712	5.625	5.599	5.566
Between-region, within-industry	94.288	94.375	94.401	94.434
		Panel B. Persistence (AR(1) Coefficients)		
1-month persistence:				
Productivity	0.939	0.936	0.933	0.932
	(0.000)	(0.000)	(0.000)	(0.000)
Idleness	0.934	0.934	0.934	0.934
	(0.000)	(0.000)	(0.000)	(0.000)
Overprovisioning	0.911	0.911	0.911	0.911
	(0.000)	(0.000)	(0.000)	(0.000)
1-year persistence:				
Productivity	0.658	0.652	0.645	0.641
	(0.002)	(0.002)	(0.002)	(0.002)
Idleness	0.659	0.659	0.659	0.659
	(0.002)	(0.002)	(0.002)	(0.002)
Overprovisioning	0.599	0.599	0.599	0.599
	(0.002)	(0.002)	(0.002)	(0.002)
5-year persistence:				
Productivity	0.325	0.321	0.321	0.314
	(0.004)	(0.004)	(0.004)	(0.004)
Idleness	0.332	0.332	0.332	0.332
	(0.004)	(0.004)	(0.004)	(0.004)
Overprovisioning	0.104	0.104	0.104	0.104
	(0.003)	(0.003)	(0.003)	(0.003)

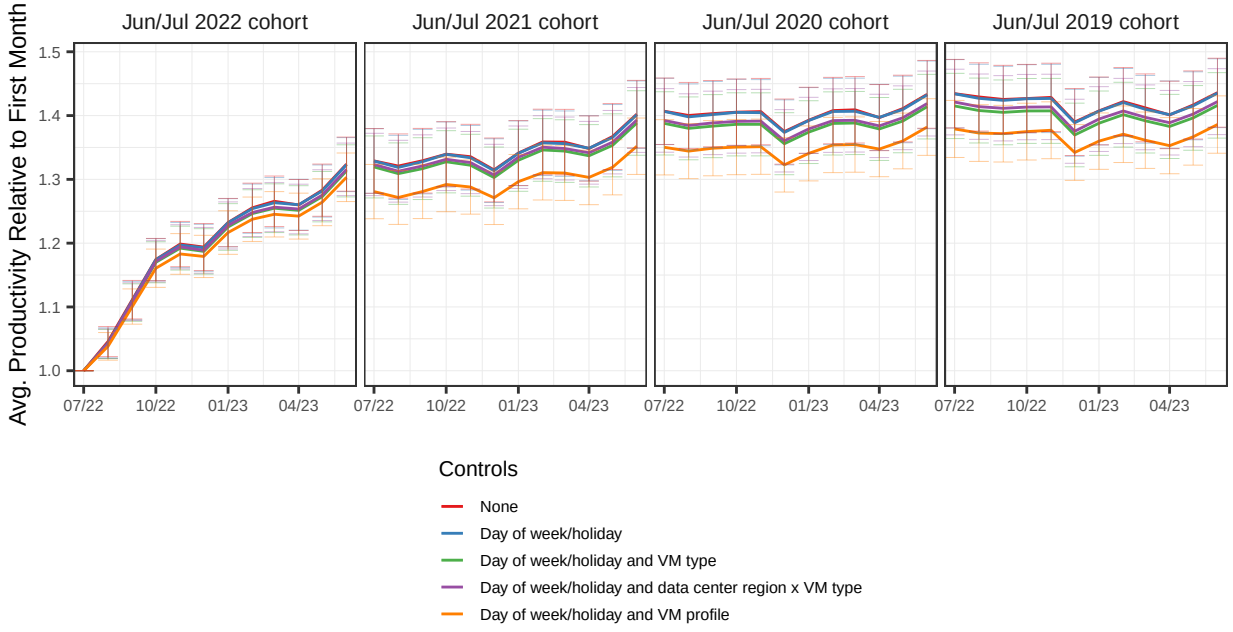
Notes: This table reports the dispersion and persistence of productivity measures. Panel A presents the dispersion of compute productivity across different productivity measures. Panel B shows the persistence of productivity, idleness, and overprovisioning measures with 1-month, 3-month, and 5-month autoregressive (AR(1)) coefficients, including their standard errors in parentheses.

Figure OA-13: Robustness: Productivity by Cohort Over Time - Different Days



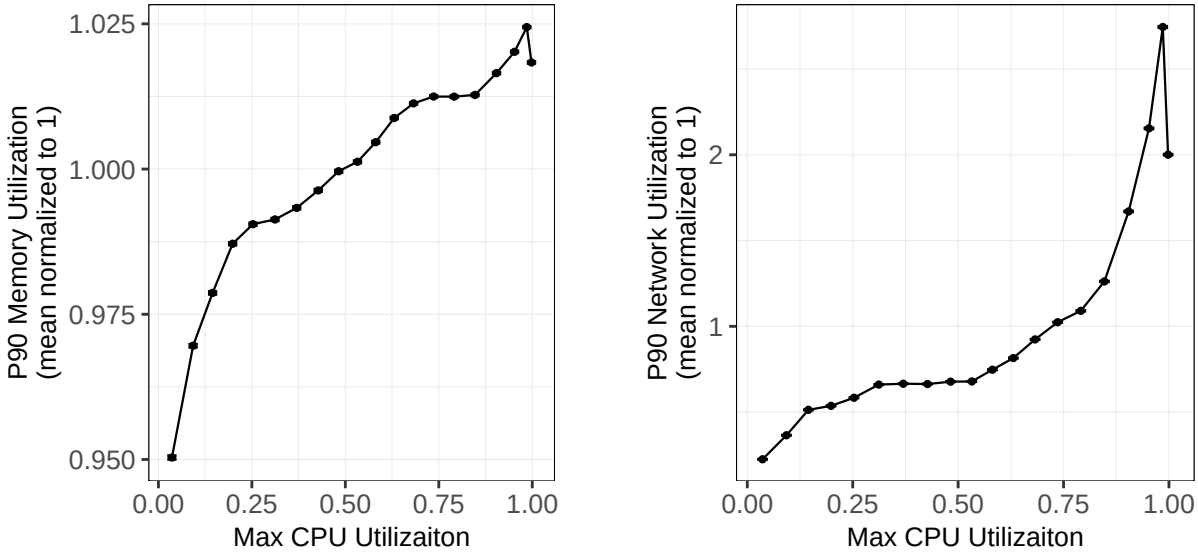
Notes: This figure shows learning analysis of Figure 7 using a different number of days in the definition of productivity.

Figure OA-14: Robustness: Productivity by Cohort Over Time - Controls



Notes: This figure shows the learning analysis of Figure 7 using different control variables in the productivity estimation.

Figure OA-15: Robustness: Correlation of CPU Utilization with Other Measures

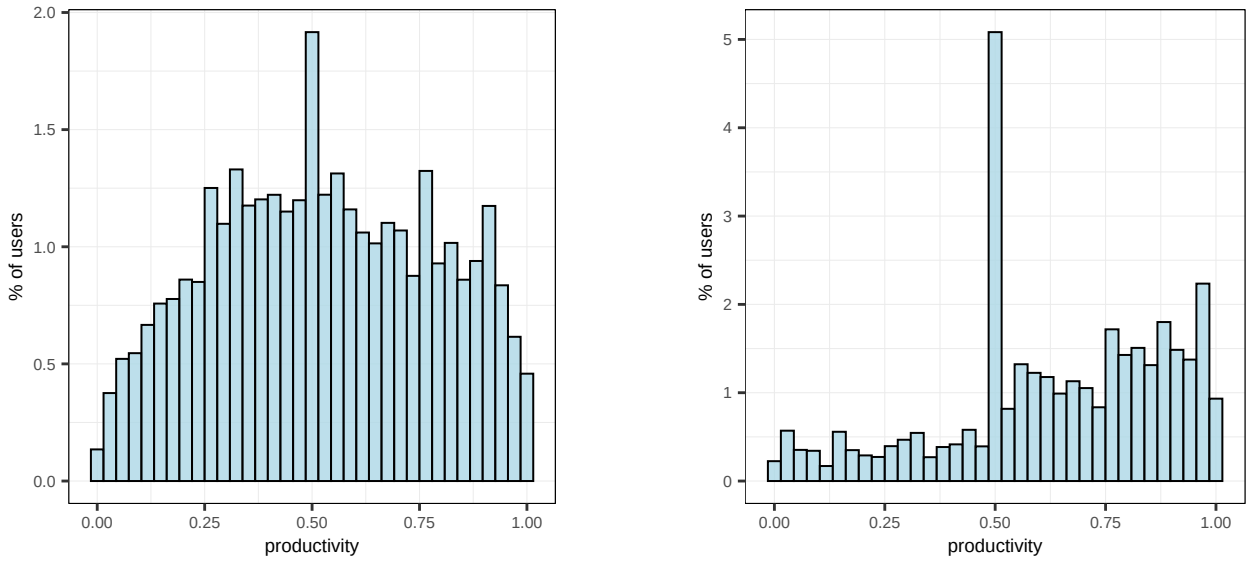


(a) CPU and Memory Utilization

(b) CPU and Network Utilization

Notes: This figure illustrates the correlation between CPU utilization correlations and other resource measures. For each figure, we divide VM-days by the max CPU utilization into twenty equally sized bins, then plot the average max CPU utilization of VMs in that bin against the max memory utilization and 90th percentile network utilization of VMs in that bin. The average max memory and the average 90th percentile network across all bins is normalized to 1.

Figure OA-16: Dispersion of User Compute Productivity



(a) Azure **(b)** GCP

Notes: These figures illustrate the distribution of user-day level compute productivity, estimated using the entire sample, weighting each VM by its core hour as shown in Equations (28) and (29) for Azure and GCP respectively. The x-axis represents productivity levels ranging from 0 to 1, while the y-axis shows the percentage of firms. Each observation corresponds to a user, and the histogram bars reflect the unweighted distribution of firms across different productivity intervals.