What I've Learned from Users

September 2022

I recently told applicants to Y Combinator that the best advice I could give for getting in, per word, was

> Explain what you've learned from users.

That tests a lot of things: whether you're paying attention to users, how well you understand them, and even how much they need what you're making.

Afterward I asked myself the same question. What have I learned from YC's users, the startups we've funded?

The first thing that came to mind was that most startups have the same problems. No two have exactly the same problems, but it's surprising how much the problems remain the same, regardless of what they're making. Once you've advised 100 startups all doing different things, you rarely encounter problems you haven't seen before.

This fact is one of the things that makes YC work. But I didn't know it when we started YC. I only had a few data points: our own startup, and those started by friends. It was a surprise to me how often the same problems recur in different forms. Many later stage investors might never realize this, because later stage investors might not advise 100 startups in their whole career, but a YC partner will get this much experience in the first year or two.

That's one advantage of funding large numbers of early stage companies rather than smaller numbers of later-stage ones. You get a lot of data. Not just because you're looking at more companies, but also because more goes wrong.

But knowing (nearly) all the problems startups can encounter doesn't mean that advising them can be automated, or reduced to a formula. There's no substitute for individual office hours with a YC partner. Each startup is unique, which means they have to be advised by specific partners who know them well. [1]

We learned that the hard way, in the notorious "batch that broke YC" in the summer of 2012. Up till that point we treated the partners as a pool. When a startup requested office hours, they got the next available slot posted by any partner. That meant every partner had to know every startup. This worked fine up to 60 startups, but when the batch grew to 80, everything broke. The founders probably didn't realize anything was wrong, but the partners were confused and unhappy because halfway through the batch they still didn't know all the companies yet. [2]

At first I was puzzled. How could things be fine at 60 startups and broken at 80? It was only a third more. Then I realized what had happened. We were using an $O(n^2)$ algorithm. So of course it blew up.

The solution we adopted was the classic one in these situations. We sharded the batch into smaller groups of startups, each overseen by a dedicated group of partners. That fixed the problem, and has worked fine ever since. But the batch that broke YC was a powerful demonstration of how individualized the process of advising startups has to be.

Another related surprise is how bad founders can be at realizing what their problems are. Founders will sometimes come in to talk about some problem, and we'll discover another much bigger one in the course of the conversation. For example (and this case is all too common), founders will come in to talk about the difficulties they're having raising money, and after digging into their situation, it turns out the reason is that the company is doing badly, and investors can tell. Or founders will come in worried that they still haven't cracked the problem of user acquisition, and the reason turns out to be that their product isn't good enough. There have been times when I've asked "Would you use this yourself, if you hadn't built it?" and the founders, on thinking about it, said "No." Well, there's the reason you're having trouble getting users.

Often founders know what their problems are, but not their relative importance. [3] They'll come in to talk about three problems they're worrying about. One is of moderate importance, one doesn't matter at all, and one will kill the company if it isn't addressed immediately. It's like watching one of those horror movies where the heroine is deeply upset that her boyfriend cheated on her, and only mildly curious about the door that's mysteriously ajar. You want to say: never mind about your boyfriend, think about that door! Fortunately in office hours you can. So while startups still die with some regularity, it's rarely because they wandered into a room containing a murderer. The YC partners can warn them where the murderers are.

Not that founders listen. That was another big surprise: how often founders don't listen to us. A couple weeks ago I talked to a partner who had been working for YC for a couple batches and was starting to see the pattern. "They come back a year later," she said, "and say 'We wish we'd listened to you.'"

It took me a long time to figure out why founders don't listen. At first I thought it was mere stubbornness. That's part of the reason, but another and probably more important reason is that so much about startups is counterintuitive. And when you tell someone something counterintuitive, what it sounds to them is wrong. So the reason founders don't listen to us is that they don't *believe* us. At least not till experience teaches them otherwise. [4]

The reason startups are so counterintuitive is that they're so different from most people's other experiences. No one knows what it's like except those who've done it. Which is why YC partners should usually have been founders themselves. But strangely enough, the counterintuitiveness of startups turns out to be another of the things that make YC work. If it weren't counterintuitive, founders wouldn't need our advice about how to do it.

Focus is doubly important for early stage startups, because not

only do they have a hundred different problems, they don't have anyone to work on them except the founders. If the founders focus on things that don't matter, there's no one focusing on the things that do. So the essence of what happens at YC is to figure out which problems matter most, then cook up ideas for solving them — ideally at a resolution of a week or less — and then try those ideas and measure how well they worked. The focus is on action, with measurable, near-term results.

This doesn't imply that founders should rush forward regardless of the consequences. If you correct course at a high enough frequency, you can be simultaneously decisive at a micro scale and tentative at a macro scale. The result is a somewhat winding path, but executed very rapidly, like the path a running back takes downfield. And in practice there's less backtracking than you might expect. Founders usually guess right about which direction to run in, especially if they have someone experienced like a YC partner to bounce their hypotheses off. And when they guess wrong, they notice fast, because they'll talk about the results at office hours the next week. [5]

A small improvement in navigational ability can make you a lot faster, because it has a double effect: the path is shorter, and you can travel faster along it when you're more certain it's the right one. That's where a lot of YC's value lies, in helping founders get an extra increment of focus that lets them move faster. And since moving fast is the essence of a startup, YC in effect makes startups more startup-like.

Speed defines startups. Focus enables speed. YC improves focus.

Why are founders uncertain about what to do? Partly because startups almost by definition are doing something new, which means no one knows how to do it yet, or in most cases even what "it" is. Partly because startups are so counterintuitive generally. And partly because many founders, especially young and ambitious ones, have been trained to win the wrong way. That took me years to figure out. The educational system in most countries trains you to win by hacking the test instead of actually doing whatever it's supposed to measure. But that stops working when you start a startup. So part of what YC does is to retrain founders to stop trying to hack the test. (It takes a surprisingly long time. A year in, you still see them reverting to their old habits.)

YC is not simply more experienced founders passing on their knowledge. It's more like specialization than apprenticeship. The knowledge of the YC partners and the founders have different shapes: It wouldn't be worthwhile for a founder to acquire the encyclopedic knowledge of startup problems that a YC partner has, just as it wouldn't be worthwhile for a YC partner to acquire the depth of domain knowledge that a founder has. That's why it can still be valuable for an experienced founder to do YC, just as it can still be valuable for an experienced athlete to have a coach.

The other big thing YC gives founders is colleagues, and this may be even more important than the advice of partners. If you look at history, great work clusters around certain places and institutions: Florence in the late 15th century, the University of Göttingen in the late 19th, *The New Yorker* under Ross, Bell Labs, Xerox PARC. However good you are, good colleagues make you

better. Indeed, very ambitious people probably need colleagues more than anyone else, because they're so starved for them in everyday life.

Whether or not YC manages one day to be listed alongside those famous clusters, it won't be for lack of trying. We were very aware of this historical phenomenon and deliberately designed YC to be one. By this point it's not bragging to say that it's the biggest cluster of great startup founders. Even people trying to attack YC concede that.

Colleagues and startup founders are two of the most powerful forces in the world, so you'd expect it to have a big effect to combine them. Before YC, to the extent people thought about the question at all, most assumed they couldn't be combined — that loneliness was the price of independence. That was how it felt to us when we started our own startup in Boston in the 1990s. We had a handful of older people we could go to for advice (of varying quality), but no peers. There was no one we could commiserate with about the misbehavior of investors, or speculate with about the future of technology. I often tell founders to make something they themselves want, and YC is certainly that: it was designed to be exactly what we wanted when we were starting a startup.

One thing we wanted was to be able to get seed funding without having to make the rounds of random rich people. That has become a commodity now, at least in the US. But great colleagues can never become a commodity, because the fact that they cluster in some places means they're proportionally absent from the rest.

Something magical happens where they do cluster though. The energy in the room at a YC dinner is like nothing else I've experienced. We would have been happy just to have one or two other startups to talk to. When you have a whole roomful it's another thing entirely.

YC founders aren't just inspired by one another. They also help one another. That's the happiest thing I've learned about startup founders: how generous they can be in helping one another. We noticed this in the first batch and consciously designed YC to magnify it. The result is something far more intense than, say, a university. Between the partners, the alumni, and their batchmates, founders are surrounded by people who want to help them, and can.

Notes

[1] This is why I've never liked it when people refer to YC as a "bootcamp." It's intense like a bootcamp, but the opposite in

structure. Instead of everyone doing the same thing, they're each talking to YC partners to figure out what their specific startup needs.

[2] When I say the summer 2012 batch was broken, I mean it felt to the partners that something was wrong. Things weren't yet so broken that the startups had a worse experience. In fact that batch did unusually well.

[3] This situation reminds me of the research showing that people are much better at answering questions than they are at judging how accurate their answers are. The two phenomena feel very similar.

[4] The Airbnbs were particularly good at listening — partly because they were flexible and disciplined, but also because they'd had such a rough time during the preceding year. They were ready to listen.

[5] The optimal unit of decisiveness depends on how long it takes to get results, and that depends on the type of problem you're solving. When you're negotiating with investors, it could be a couple days, whereas if you're building hardware it could be months.

Thanks to Trevor Blackwell, Jessica Livingston, Harj Taggar, and Garry Tan for reading drafts of this.
