'Easy geom recipes' evaluation and feedback report

***Abstract*—This report describes the new 'Easy geom recipes' https://evamaerey.github.io/easy-geom-recipes/ learning resource and reactions to it based on an 8-expert survey and focus group evaluation conducted in May 2025.**

I. Introduction

The resource 'Easy geom recipes' is an how-to-first educational resource designed to gently bring newcomers into the powerful world of ggplot2 layer extension – and extension more generally.

This work is motivated by the belief that layer extension could bring great value to many ggplot2 users (e.g. data scientists, educators, students) who either haven't recognized its power or who may believe the space to be too difficult to enter.

By going through computationally simple, step-by-step extension cases which also showcase pay-offs — e.g. concise, readable code where group- and panel-wise compute happens 'for free' — newcomers should 1) more fully appreciate extension's power 2) gain motivation for further theoretical and how-to learning, and 3) realize that extension is a within-reach tool, usable in-script (no package building or maintenance required!).

At its core, 'Easy geom recipes' rethinks who appropriate ggplot2 extenders might be. Prerequisites are not, for example, being comfortable with OOP and being a developer with package building know-how. Rather this point-of-entry's requisite is having a pretty good feel for ggplot2's API – ggplot2 fluency – and a sense of how new functionality could add fluency when 'base' ggplot2 capabilities are at their limits.

Specifically, the recipes demonstrate how useful geom_*() layer functions can be created by creating new Stat objects: a Stat-to-geom_*() point of entry. This approach is 1) powerful (Stats encapsulate compute); 2) familiar (geom_* prefixed functions dominate layer usage among most ggplot2 users); 3) informative (introduces new concepts/functions ggproto OOP inheratance, `layer()` usage); and 4) friendly (it can be demonstrated with simple and familiar computational examples like means or row numbers).

The approach emphasizes application and practice, providing a series of simple, easy-to-follow examples as well as companion exercises, with with intent to foster internalization and retention through repeated practice.

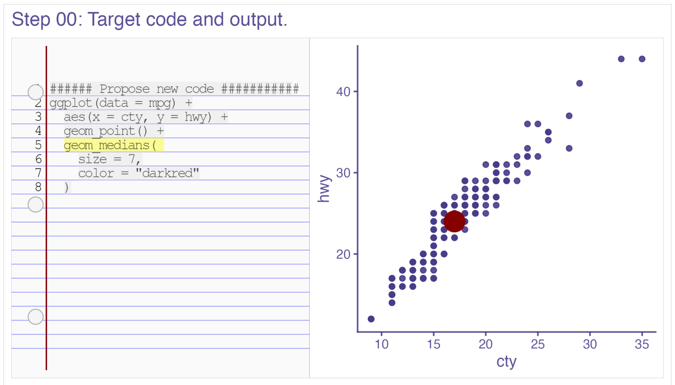
In early 2023, a bare-bones how-to tutorial was evaluated by post-secondary statistics educators in a downloadable .Rmd format. This report evaluates a new version using quarto and WebR.

Using Quarto and WebR means that greater explanation can accompany the how-to steps in collapsed call-outs and {webr} chunks that make the exercises easier to try in-situ on the hosting website.

The recipes have the limited goal of giving newcomers tastes of success and expose foundational mechanisms and concepts in layer extension. More layer extension 'moves' are explored in other resources.

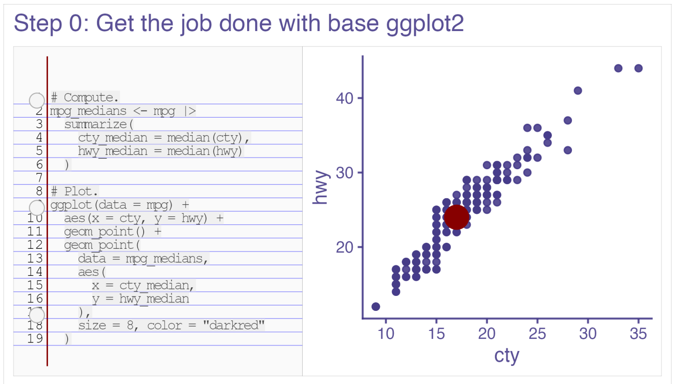
'Easy Geom Recipes' show newcomers how to extend layers with a approachable formula (a 'new-classic' introduction):

- Step 00. Define your aspiration, i.e. identify your code-output goal
- Step 0. Complete the task without extension.
- Step 1. Define Compute. Test.
- Step 2. Define Stat Object. Test. (We also note that an 'early exit' can be taken here. It may be preferable to just use the Stat as an argument in a geom_* function in an in-script setting.)
- Step 3. Define User-facing function(s) geom_* and stat_*. (hint: use geom_point and stat_identity definitions as a model)
- Step 4. Test 'target code' from Step 00. Enjoy!



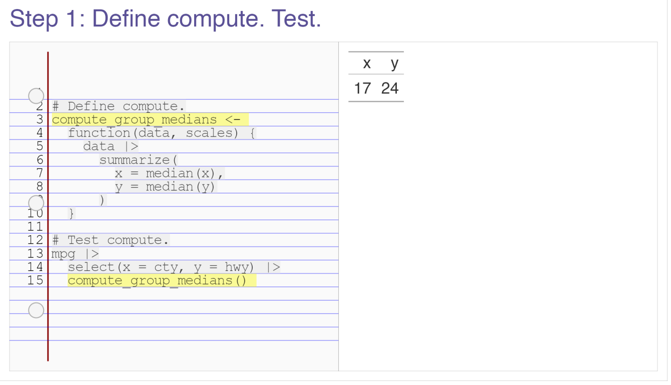
Step 00: Target code and output.

```
###### Propose new code ###########
ggplot(data = mpg) +
  aes(x = cty, y = hwy) +
  geom_point() +
  geom_medians(
    size = 7,
    color = "darkred"
  )
```



Step 0: Get the job done with base ggplot2

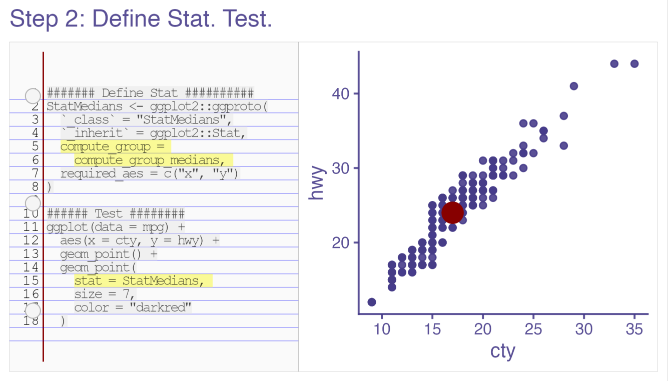
```
# Compute.
mpg_medians <- mpg |>
  summarize(
    cty_median = median(cty),
    hwy_median = median(hwy)
  )

# Plot.
ggplot(data = mpg) +
  aes(x = cty, y = hwy) +
  geom_point() +
  geom_point(
    data = mpg_medians,
    aes(
      x = cty_median,
      y = hwy_median
    ),
    size = 8, color = "darkred"
  )
```



Step 1: Define compute. Test.

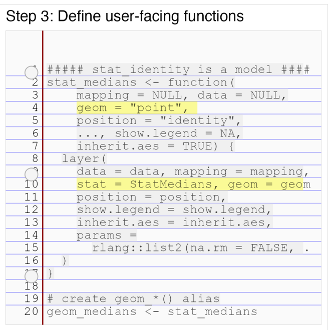
```
# Define compute.
compute_group_medians <-
  function(data, scales) {
    data |>
      summarize(
        x = median(x),
        y = median(y)
      )
  }

# Test compute.
mpg |>
  select(x = cty, y = hwy) |>
  compute_group_medians()
```

x	y
17	24



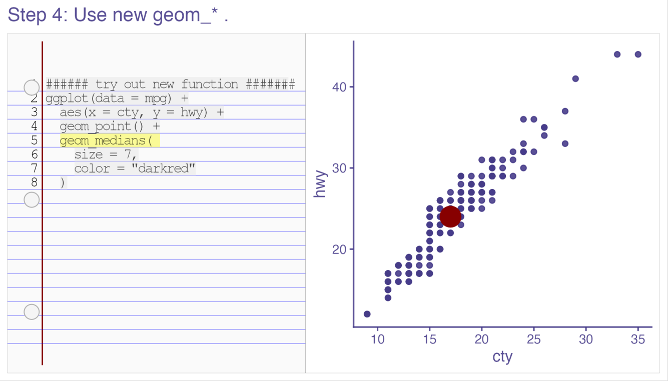
Step 2: Define Stat. Test.

```
####### Define Stat ##########
StatMedians <- ggplot2::ggproto(
  `_class` = "StatMedians",
  `_inherit` = ggplot2::Stat,
  compute_group =
    compute_group_medians,
  required_aes = c("x", "y")
)

###### Test ########
ggplot(data = mpg) +
  aes(x = cty, y = hwy) +
  geom_point() +
  geom_point(
    stat = StatMedians,
    size = 7,
    color = "darkred"
  )
```



Step 3: Define user-facing functions

```
##### stat_identity is a model ####
stat_medians <- function(
    mapping = NULL, data = NULL,
    geom = "point",
    position = "identity",
    ..., show.legend = NA,
    inherit.aes = TRUE) {
  layer(
    data = data, mapping = mapping,
    stat = StatMedians, geom = geom
    position = position,
    show.legend = show.legend,
    inherit.aes = inherit.aes,
    params =
      rlang::list2(na.rm = FALSE, .
  )
}

# create geom_*() alias
geom_medians <- stat_medians
```



Step 4: Use new geom_* .

```
###### try out new function #######
ggplot(data = mpg) +
  aes(x = cty, y = hwy) +
  geom_point() +
  geom_medians(
    size = 7,
    color = "darkred"
  )
```

II. Departure from other educational materials: Stat-defined geom_*() functions as the point-of-entry for layer extension

A noticeable feature of 'Easy geom recipes' is that create use Stat-defined geom_* functions as a point of entry, which merits some discussion.

Experienced ggplot2 extension developers know that among the three main mechanisms to defining new layer behavior —Stats, Geoms, and positions —- creating a Stat is the simplest.

The recipes acknowledge this technical reality but also recognize that most ggplot2 users feel more comfortable with geom_* functions than stat_* functions, making geom-prefixed layer functions the preferred interface. Focus group participant Participant A for example commented about hardly ever using stat-prefixed layer functions:

> I've used ggplot for a very long time… Conceptually I get [the difference between stat_*() functions and geom_()*s functions.] But like … I would not put + stat_*() anything. That's not something I would naturally do after using the gg platform… for 10 years.

Given ggplot2 users' comfort, the recipes take a pragmatic approach of creating new layers that are geom-prefixed functions.

Naming conventions

Common naming convention matches geom_* functions to their internal Geom objects (e.g., geom_rect uses GeomRect). A results of creating new geom_*()s that are Stat-based is non-standard naming. This could surprise more advanced users, who have a greater tendancy to mix and match ggproto objects.

However, many successful geom_* functions break with the naming convention. For instance, geom_count() uses GeomPoint and StatSum internally, while geom_jitter() combines GeomPoint with StatIdentity and position_jitter()—not a GeomJitter as the name might suggest. Similarly, in the extension ecosystem, ggbump::geom_bump() is built from StatBump and GeomLine.

The fact that these functions are non-conventionally named hasn't hindered users from specifying millions of plots with geom_jitter, geom_count and geom_bump.

Thus, for sake of user comfort, we think this non-standard naming is merited.

Naming conventions, continued

Among more experience extenders in the focus group recognized difficulty in layer-naming conventions.

> **Participant H** I just think that it's weird to have two things that live at the [same level], because ultimately they all filter down to layer. And it's really a layer that you're creating.

> **Participant D** There's a sense in which also, like, you feel that maybe everything should just be 'layer_'. But the problem is that nobody actually does that in practice. And so, you know, you would be teaching something, and it would be kind of a ggplot variant that nobody else really uses.

For even more participant feedback about this topic, see the feedback section "Why not start with Geom-based geom_* functions".

Existing educational materials *do* model the new Stat to new geom_*() approach. For example, `geom_chull()` combines `GeomPolygonHollow` and `StatChull`, while `geom_spring()` uses `StatSpring` and `GeomPath` before the case study progresses to defining `GeomSpring`. And in the springs case study, Thomas Lin Pedersen notes user preference for the geom_*() constructor:

> Stat objects are almost always paired with a geom_*() constructor because most ggplot2 users are accustomed to adding geoms, not stats, when building up a plot.

But gleaning the lesson to define Stat-based `geom_*()`s from existing materials can feel circuitous. The `geom_chull()` example from the extension vignette primarily serves to demonstrate how `GeomPolygonHollow` is created from `GeomPolygon`. The `StatSpring`-based geom_spring() appears mid-case-study rather than as the final goal. 'Easy geom recipes' embraces Stat-based geom_* functions as an end unto themselves.

Participant E, noted the usefulness of a more straightforward pathways to the Stat-based geom_*() functions in the focus group:

> When you're building your first extension package, you tend not to jump into super complicated extensions straight away. Going through the vignette, there's a lot of stuff to read through that maybe isn't relevant. You're reading through material thinking, 'Is this something I need to know to build [my extension]?' … It gives you a lot of information you don't necessarily need, and as a new learner, that can be a bit overwhelming."

Still, creating a geom_*() user-facing function as well as a stat_*() from a Stat may feel confusing. Participant A commented, 'Like I've used ggplat for a very long time I do not know, did not know, still don't know the difference exactly between stats and geoms other than geom is the thing I would use, right? [but] I kind of left that section like, … 'why did we do a Stat and not a Geom, like, I … the tutorial starts with, you're gonna make a geom_* But I made a Stat.

Participant A

For more feedback on how 'Easy geom recipes' complements existing resources see the, "Complement to existing resources" section.

III. Easiness

Beyond making Stat-based geom_* functions a primary goal, 'easy recipes' progresses in small, manageable steps: Step 0 begins with an existing plot where extension might be helpful; Step 1 isolates the required computation and prepares it as a function; Step 2 defines a Stat based on that computation; and Step 3 creates the user-facing constructor function. Each step models testing methods to verify successful completion.

The examples are deliberately simple−−computing means and indexes in the first exercises. While these computations aren't exciting and don't produce especially eye-catching visualizations, they should be familiar to ggplot2 users. This familiarity should allow learners to focus on extension mechanics and not be distracted by complex or unfamiliar mathematical concepts.

Also, using tidyverse pipelines for data manipulation, given its intuitiveness and readability, means learners can focus more of their attention on understanding the extension mechanism.

IV. How-to-first, accompanied by explanation

In the new quarto-webr framework, explanation accompanies the coding steps via quarto call-outs. Users can engage with theory and explanation based on the level of their curiosity.

I believe collapsible call-out makes the tutorial resonate for very different audiences and people with different learning styles. There are people that thrive in a taste-first environment, who might say of their experience 'Show me the mechanism and what it delivers first, then I'll have a reason and footholds to engage with theory'. See Participant B's discussion below. And other learners thrive in an explanation-accompanied environment, who might say "I need to have the mechanics embedded in explanation.” See Participant C’s discussion.

> **Participant B** Some of the [callout] notes addressed things that I didn’t understand fully, but they weren’t a barrier to completing the exercises. … I mostly benefited from reading [the callouts] after going further in the recipes. But that’s probably because I just didn’t notice those things at first; the reason why those things were important became clearer as I proceeded through the content.

> **Participant C** In general, the “you may have noticed” collapsed callouts felt like they should be toggled open by default or in an always-open callout. Especially while reading the first recipe, I felt that I needed all of the additional context they provided to understand the provided code.

For more participant feedback on callouts, see the ‘Managing Callouts’ section.

V. Participant Profile

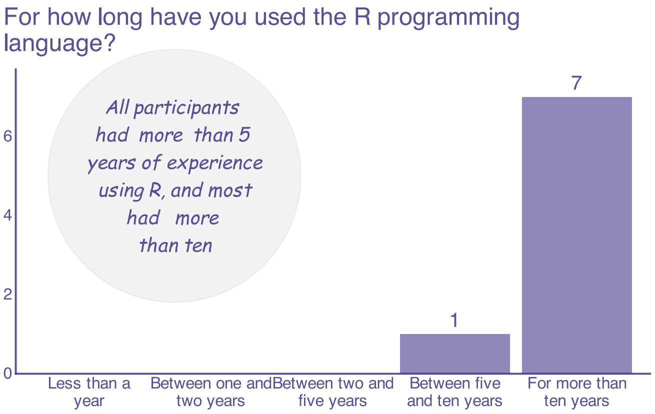
This year, May 6th, feedback on the quarto/webr version was sought from eight data and data-visualization professionals.

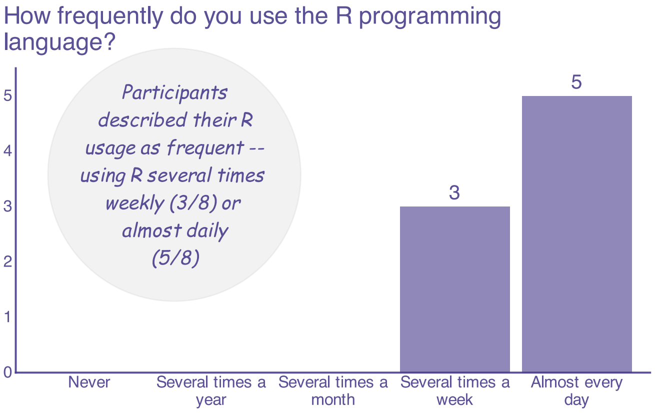
- Participant A, Software Engineer;
- Participant B, Senior Instructor in Statistics;
- Participant C, Software Developer;
- Participant D, Associate Professor in Statistics;
- Participant E, Data Visualization Professional;
- Participant F, Lecturer in University Statistics;
- Participant G, Product Marketing;
- Participant H, Professor;

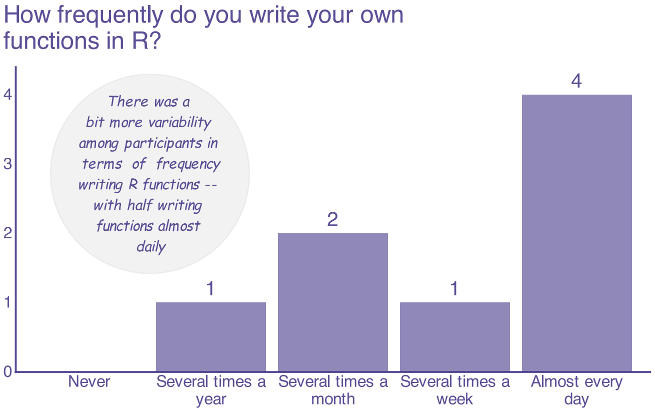
Participants were asked to complete the tutorial and survey before attending a 1-hour virtual focus group. The focus group was recorded to preserve detail of responses. The following sections provide an overview of their background in R, ggplot2, OOP and education.

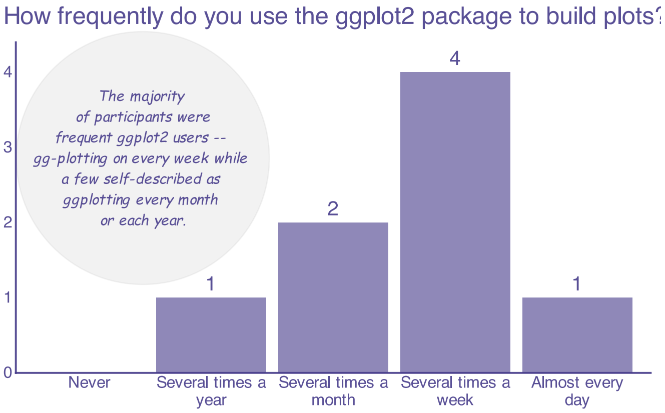
A. Participant experience as R and ggplot2 users

The eight participants have a wealth of experience with R and R programming, but somewhat more varied experience with ggplot2 and quite varied experience with ggplot2 extension.

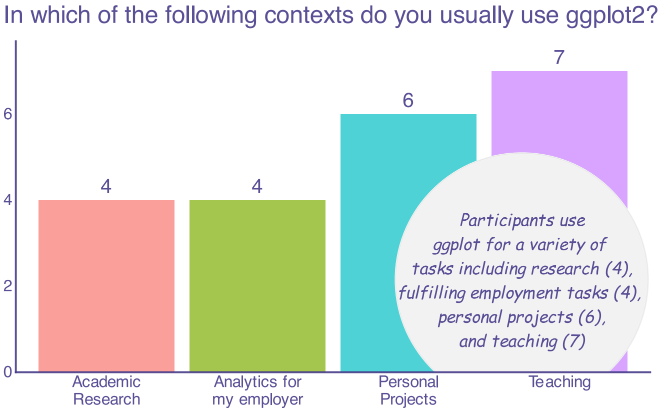







How frequently do you use the ggplot2 package to build plots?

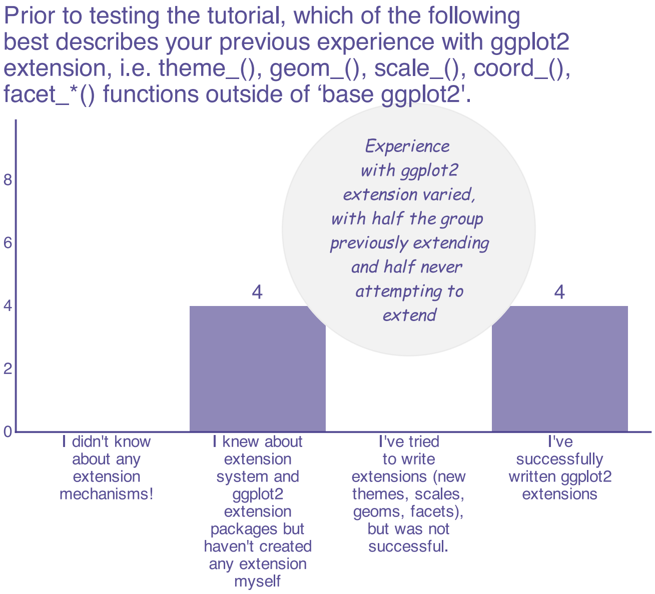
The majority of participants were frequent ggplot2 users -- gg-plotting on every week while a few self-described as ggplotting every month or each year.


In which of the following contexts do you usually use ggplot2?

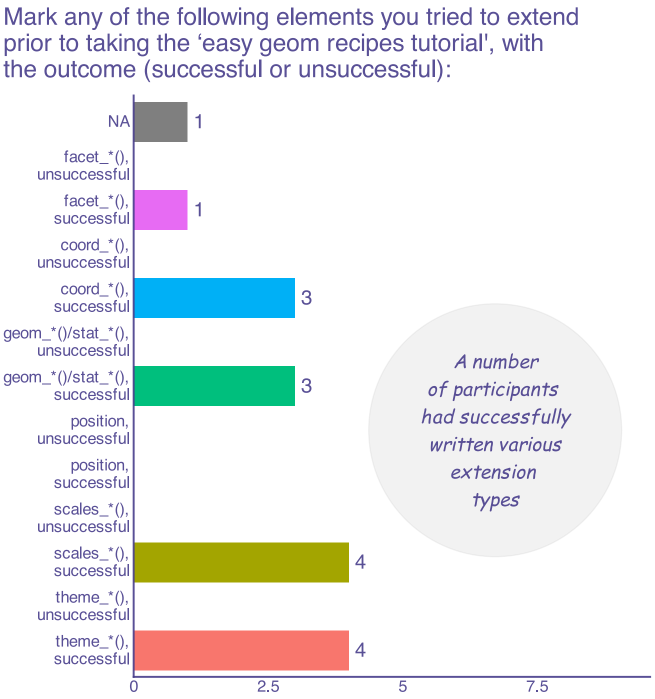
Participants use ggplot for a variety of tasks including research (4), fulfilling employment tasks (4), personal projects (6), and teaching (7)

B. Prior experience with ggplot2 extension and object oriented programming

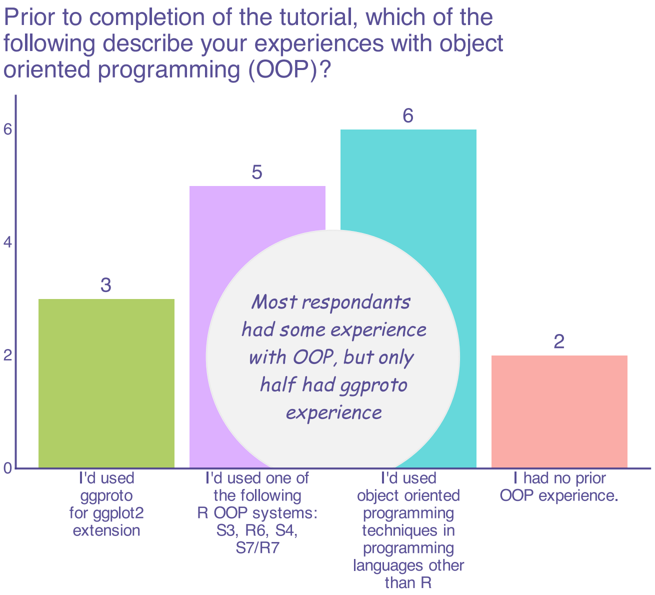
The group had a range of experiences when it came to ggplot2 extension and object oriented programming (OOP).


Prior to testing the tutorial, which of the following best describes your previous experience with ggplot2 extension, i.e. theme_(), geom_(), scale_(), coord_(), facet_*() functions outside of 'base ggplot2'.

Experience with ggplot2 extension varied, with half the group previously extending and half never attempting to extend


Mark any of the following elements you tried to extend prior to taking the 'easy geom recipes tutorial', with the outcome (successful or unsuccessful):

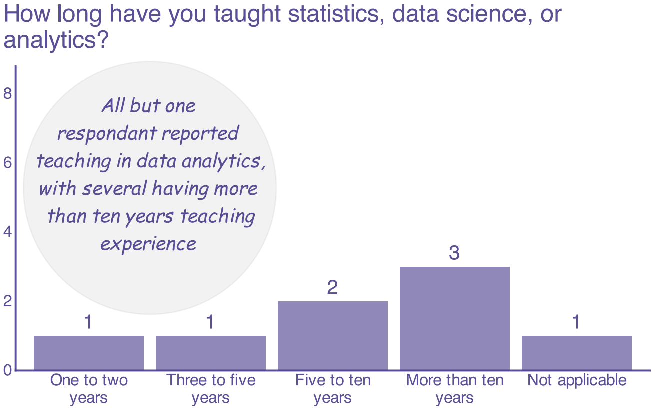
A number of participants had successfully written various extension types


Prior to completion of the tutorial, which of the following describe your experiences with object oriented programming (OOP)?

Most respondants had some experience with OOP, but only half had ggproto experience

C. Teaching profile

Most participant had at least some teaching experience and so could speak to the potential accessibility for data science students.


How long have you taught statistics, data science, or analytics?

VI. Feedback on the tutorial

A. Accessibility

In the focus group, there was much discussion about the accessibility of the recipes. Overall, the feedback was that the tutorial was quite accessible.

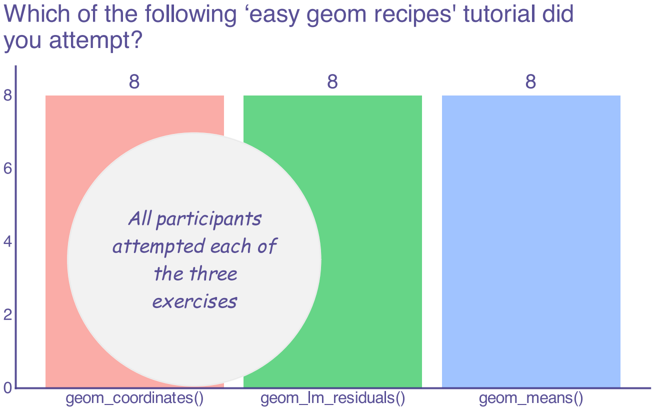
> **Participant D** I really liked your resource because it felt like a concise kind of collection of topics. You have to do this. Do this, do this, and then boom. You have a Stat and a geom_*.

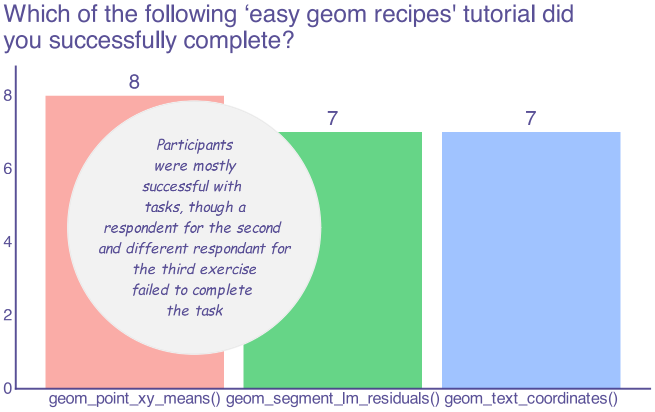
> **Participant E** The tutorial … takes you through: do this 1st and then this and then this. I think that's really helpful. Is really nice rather than having to sort of dig through [material]… When you're building your 1st extension package, you tend not to jump into the … super complicated extension straight away, or build a simpler extension that maybe only adds a couple of things. -

> **Participant C** So this was my 1st time poking at any sort of extension… For the most part, I felt like I was able to interpret everything that I was seeing readily enough.

B.

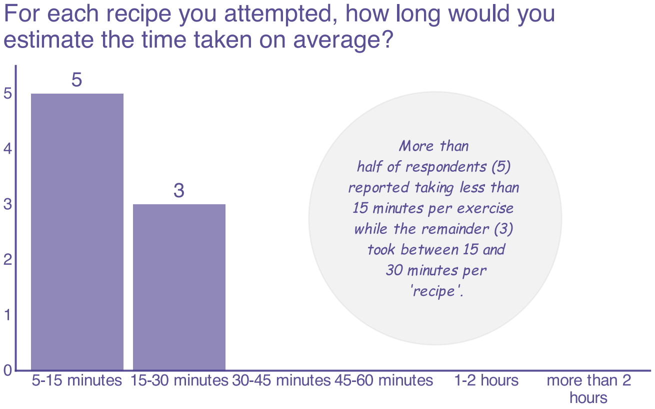
Several survey questions asked about clarity of the recipes and participants success with them and this was a topic of discussion in the focus group. Overall, attempt and completion rates of the tutorial exercises was quite high for the group, with some difficulty being noted for exercise #2 and #3 noted among a few individuals. This merits careful follow-up and probable revision

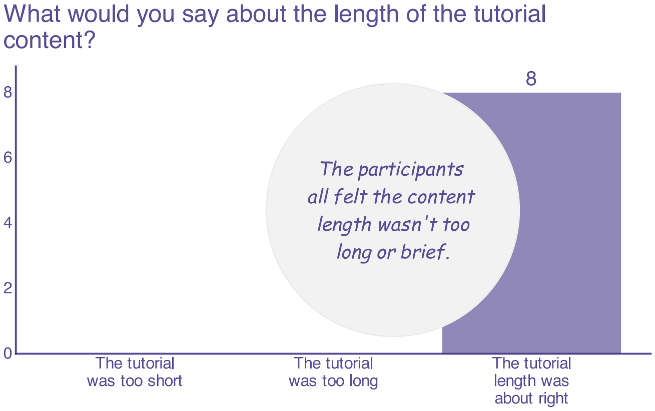

Which of the following 'easy geom recipes' tutorial did you attempt?


Which of the following 'easy geom recipes' tutorial did you successfully complete?

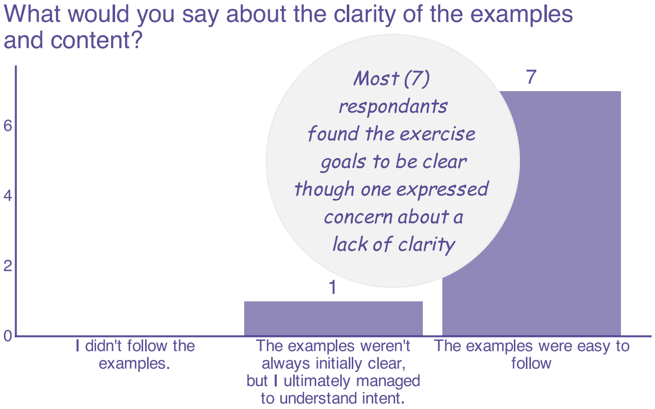
> My only problem was when trying the Step 2: Write Stat part of the exercise in Recipe #2. I got an error ("Error: Problem while setting up geom.") after doing what I believed I needed to do during the exercise. I didn't have this problem when completing the exercises for the other two recipes.

Perhaps with some small revisions and/or additional hints, the recipes can reliably deliver 'easiness' to the most ggplot2 users.


For each recipe you attempted, how long would you estimate the time taken on average?


What would you say about the length of the tutorial content?

The participants all felt the content length wasn't too long or brief.


What would you say about the clarity of the examples and content?

Most (7) respondants found the exercise goals to be clear though one expressed concern about a lack of clarity

C. *Complement to existing materials*

In the focus group, participants that *had* done layer extension reflected on how using the tutorial differed from their prior experience with other materials.

> **Participant D:** Yeah, I have found myself doing … quite a bit [in guiding graduate students in extension] over the past few years. My [graduate students] are probably more oriented towards computational aspects or statistical aspects as opposed to communication. But … I don't have … a go-to collection of resources to say, like, 'Here's how you do this', you know. Here's how you extend ggplot2.

> **Participant D:** Github is a great … teaching resource that you can learn by yourself. And at that stage most of my students are, you know, they're Phd students, and they're capable of self learning. So I can kind of point them to [other extension packages]. They'll look over it. They'll come and we'll … talk through it. But of course, the simpler the resource the better.

> **Participant E** I definitely think [the tutorial] would have helped [me learn extension]. I took a similar approach of [Participant D's graduate students]: I want to build an extension package. So I'm going to go and find another extension package that exists that is not too dissimilar from the thing I want to build and adapt from that. And … you are sort of searching through Github repositories and [are] trying to understand codes that other people have written, and it's not necessarily well documented or explained… [And authors] don't write code in a linear fashion. So you don't know from looking at the final product which bit you need to do first.

One responded reported doing the tutorial while also consulting other materials!

> **Participant F** I found myself just going back and forth between the book and the tutorial, because, like well, you know, to get more information. But you could say, that's a benefit, right? I wouldn't have done it in the 1st place, if the [easy recipes] weren't there.

D. *Packaging versus in-script use of Stats*

The recipes demonstrate how to test new Stats created in Step 2 using syntax like `geom_point(stat = StatMeans)`. It's also suggested that this be an 'early-exit' for users that are just interested in using the functionality themselves, instead of writing the user-facing constructor function which requires much scaffolding code.

In the focus group, there was some discussion about making it clear why one would proceed to Step 3, writing the user-facing constructor function. The answer to proceed or not revolved around the intent to package the new layer capabilities.

> **Participant B** The motivation for writing custom Stats for ggplot was well demonstrated. The user-facing functions part was interesting, but it strikes me as something more of interest to people developing packages to extend ggplot.

> **Participant E** I think there needs to be a little bit more motivation for what's the point of step 3. Because, like you get to the end of Step 2, you have `geom_point(stat = StatMedians)`, and that feels like essentially as easy as `geom point`, right? …. What's the motivation to then go and create this stat_ [or geom_] function? … Because I think if I was… a new user and I didn't want to make a package, and I was [creating] something just for me. I'd get to the end of Step 2, and that's reasonably clean, and it kind of works for my use case. So … a little bit more motivation as to why build a stat_ function [would be helpful].

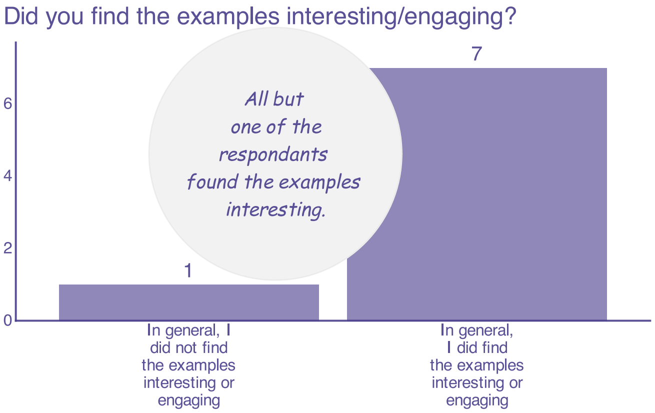
But others in the group appreciated the full demonstration, with an eye to package building.

> **Participant C** My reflex is like, if I were [create] a package, I would want it to have the geom_ [function provided], ... rather than people having to type [in the Stat] out themselves... So that's sort of my entry point into motivating step 3 [user-facing functions]. And I think I would ultimately want to have access to that information.

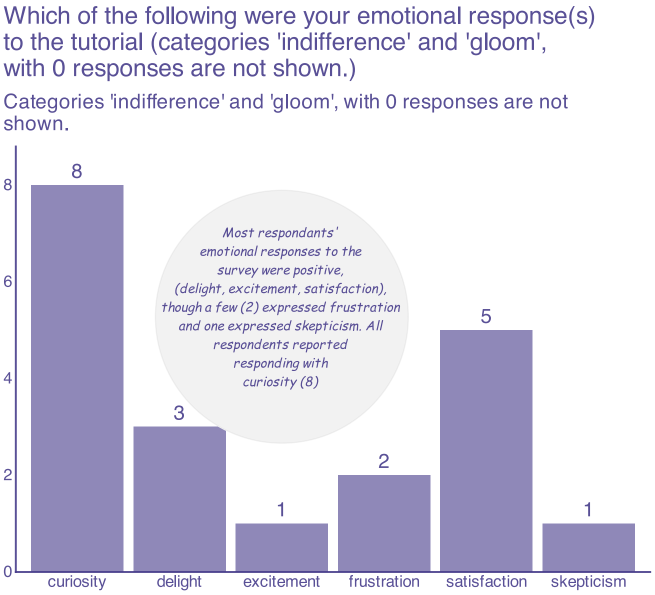
From a pedegogical perspective, writing the constructor function also illuminates how familiar functions in ggplot2 are defined. Writing the constructor function, even if they opt for the 'early exit', connects the learner to what they already know (i.e. they should come away thinking, 'Oh, geom_smooth is defined by both a Stat and a Geom').

E. *Engagement and other attitudes*

Most attitudes in the survey were positive, and most respondents found the examples reasonably engaging.


Did you find the examples interesting/engaging?

All but one of the respondants found the examples interesting.

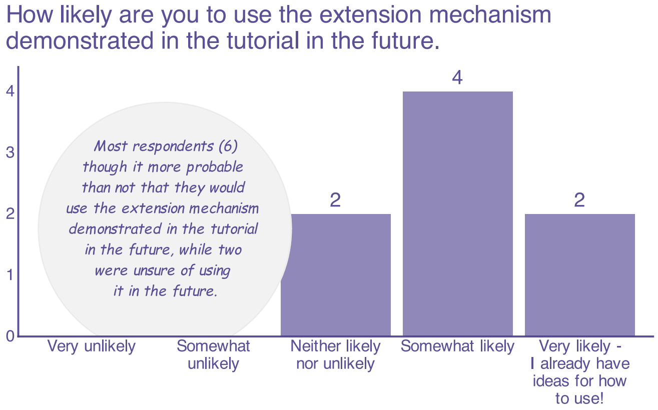

Which of the following were your emotional response(s) to the tutorial (categories 'indifference' and 'gloom', with 0 responses are not shown.)

Categories 'indifference' and 'gloom', with 0 responses are not shown.

Most respondants' emotional responses to the survey were positive, (delight, excitement, satisfaction), though a few (2) expressed frustration and one expressed skepticism. All respondents reported responding with curiosity (8)

F. *Prospects for future personal and educational use*

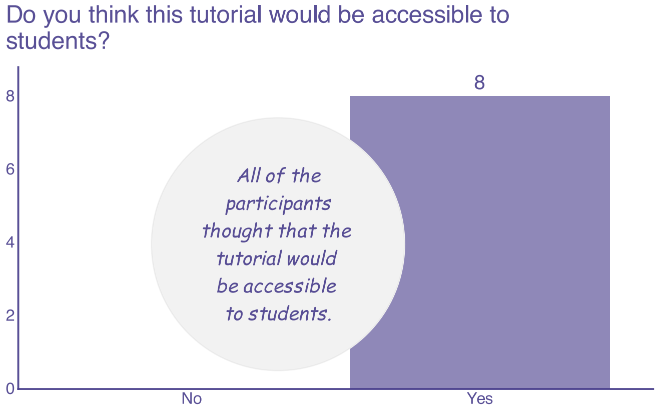
The participants had varied feedback about their likelihood of future use.


How likely are you to use the extension mechanism demonstrated in the tutorial in the future.

Most respondents (6) though it more probable than not that they would use the extension mechanism demonstrated in the tutorial in the future, while two were unsure of using it in the future.

One participant gave very specific feedback about his likelihood of use – indicating his plans to teach the material with the tutorial in the near future

> **Participant D** I intend to include it at the end of my data visualization component for at least one of my courses! ... I think that undergraduate students could understand it pretty easily, at least after the introduction to data visualization that I give them, which is very ggplot 2 focused... Now, it would come at the end, and it would be in a situation where you know you do have a kind of redundancy going on: So you do the same kind of computation every single time. And wow! Wouldn't it be nice if you could, you know, bake that into the tool itself? And then, hey, you get this *additional stuff* – you know you can do group-wise computations, and you don't have to do [the computations again] manually. And you can do this faceting, and you don't do [the computations] manually. So I really liked it for that.


Do you think this tutorial would be accessible to students?

All of the participants thought that the tutorial would be accessible to students.

G. *Call-out management*

I was especially interested to hear feedback the use of callouts for adding more detailed information about the tutorial steps. Generally, the callouts were appreciated, but respondents felt they could be better managed.

Several respondents felt the callouts should be toggled open by default or the content indicated in the title.

> **Participant C** In general, the "you may have noticed" collapsed callouts felt like they should be toggled open by default or in an always-open callout. Especially while reading the first recipe, I felt that I needed all of the additional context they provided to understand the provided code.

> **Participant H** I found the many tips and "you may have noticed" expandable sections a bit confusing, as I never knew before clicking on them whether I would find them useful or not, so I felt the need to click on all of them, but that disrupted the flow.

But others indicated that being able to at first ignore the call outs was helpful given their background. Further along in the recipes, it was easier to make sense of them.

> **Participant B** Some of the [callout] notes addressed things that I didn't understand fully, but they weren't a barrier to completing the exercises. ... I mostly benefited from reading [the callouts] after going further in the recipes. But that's probably because I just didn't notice those things at first; the reason why those things were important became clearer as I proceeded through the content.

Another comment was that I could do a better job indicating that very similar information would be available in all three exercises.

> **Participant G** Another thing that I ran into is like after the 1st tutorial. When I got to the second and the 3rd since some of the text was repeated, I was like kind of scrolling through really quickly. And then I realized, like, Oh, I missed something important. And so maybe like some other way of using a call out boxes to be like, you know. kind of highlighting that it's different than what folks may have seen before. I think like it's a it's a great benefit right? That the process is so structured that you go like step one step, 2, step 3. But if there are other like kind of key things to to know seeing like if Quarto can help kind of in that way.

H. *Structure verses prose*

Another topic in the focus group was regarding the formulaic approach, with one respondent questioning it:

> **Participant A** I think 'recipes' is both a good place to end up but also an overly-constraining format for the first few tutorials. I'd really encourage you to break out of the recipe structure for the first one or two.

At the same time, others hinted that clearly breaking out the steps was helpful:

> **Participant C** Very satisfying that the same basic steps all still applied from recipe to recipe. -

> **Participant B** [It was] sort of that 'monkey see monkey do'... But at this level that kind of is a good thing to be doing, because just I have no context for extension. So doing that over the course of 3 tutorials. It's like, okay, I'm getting the hang of this....

I. *Interleaving demonstration and practice*

One respondent suggested going back and forth between example and practice instead of leaving the practice until the end. This seems worth considering and asking others about this idea.

> **Participant A** I ... would have liked to have the practice sections interleaved with the learning. The 'Your Turn' examples were perfect for reinforcing what I had just read, but it was a lot more work to read the whole tutorial, keep it all in my head, and then try to practice.

J. *Target audience and prerequisites*

Respondents were concerned with the question of being clear about who the target audience was. I believe that fairly new ggplot2 users could grasp the material covered in the tutorial, especially with the guidance of more experienced user.

One criteria could be how comfortable *Steps 00*, defining the desired syntax and behavior, and *Step 0*, get the job done with 'base ggplot2, are for people attempting the tutorial. If those Steps do *not* feel comfortable yet, it might not be appropriate for the user to proceed. Clear messaging to that effect should be included.

> **Participant G** It would be helpful to have a list of prerequisites or resources in the intro.

> **Participant H** I do think it's a valid question, like, who is the audience – who should learn it?

Participant G Since there's a structured series of steps to the tutorials, I would have liked to see the "whole game" at the beginning (à la "R Packages" https://r-pkgs.org/whole-game.html) that walks through the steps generally (define compute, define new stat, etc.). It could be linked into the intro...

K. *Why not write wrapper functions?*

Participant H When you have to make the plot over and over again, my 1st approach would probably not be an extension. My 1st approach would be [to] just write a function that makes the type of plot that I want.

Participant D If you write a wrapper function of a ggplot plot, then you can very easily find yourself in situations where you need metaprogramming tools. and that that leads you to kind of like a whole different route of embracing things and complexity that that I I don't think you want to introduce. And at that point it's probably easier to have these kind of like local versions of of stats and geoms that accomplishes the kind of task that you want. Without even bundling it into some kind of package.

Participant D So like a clean example of that where you say like 'Hey, okay, you know, I want to [just write a wrapper function]. But oh, now I want to change variables, and now you're dead from the ggplot side and your wrapper. So if you could find a clean example where you want to do the same thing many times, but to do it functionally with a wrapper would require you to use tidy evaluation or some kind of Meta programming non-syntactic stuff. That'd be a really really nice example.

Participant E So if they're kind of making, you know the same sort of custom chart over and over again, then this is an ideal solution, you know.

L. *Straight to Geom-based geom_*() functions?*

There was some discussion from a more seasoned extender actually defining a Geom as the start point.

Participant H I think the examples were too boring honestly. And you just need to do something totally crazy like, write a geom that draws stars, or something like that, like something that obviously is not currently possible. And it's different. And it looks visually interesting. And then nobody will ask, well, why am I just not using geom_point? Well, because it's not that we're doing something totally different.

But in the interest of being a gentle introduction, I creating a Stat that defines the new behavior is appropriate. And being able to simply baking compute into a layer is something that the target audience is likely to find new and useful.

Participant F Wouldn't [creating a Geom] require more complex code...? I never tried to make an extension – I did not want to deal with grid.

This participant might be reflecting on *ggplot2 extension vignette* that states: 'It's harder to create a new geom than a new stat because you also need to know some grid.'

M. *General skepticism*

Participant H In practice, there's so many pitfalls and confusing things. The problem is, the data-flow through ggplot is so complex and confusing... Yes, in theory it's easy ... you just make a new stat. And then in practice, there's so many pitfalls and confusing things.

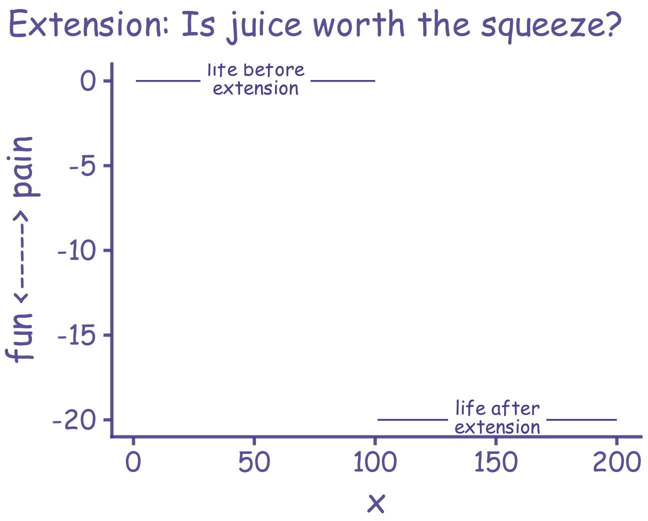
N. *Sundry*

There were many other valuable, detailed comments ranging from syntax formatting, data manipulation tooling, introduction to WebR coding chunks, load time for code with WebR. These are not reported here, but will be addressed individually.

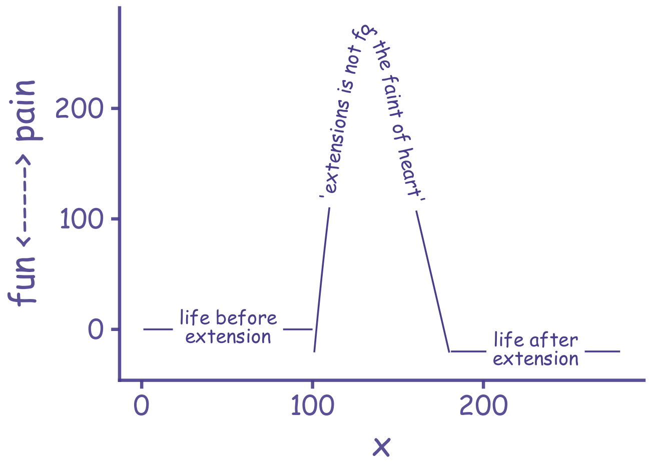
VII. Looking ahead

We are incredibly grateful to all the reviewers of tutorial. Incorporating feedback will certainly make for an improved experience for anyone using the tutorial in the future.

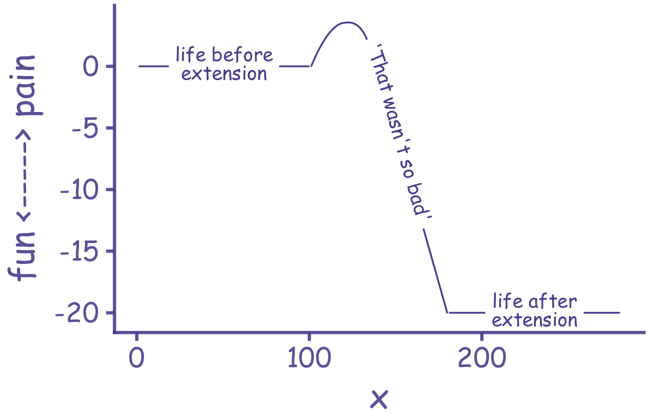
After the 'Easy geom recipes' introduction, learners should know the basics of layer extension and have a strong footing for further learning.



Juice is *not* worth the squeeze



Juice *is* worth the squeeze*



* more often than you might think