Queen Mary University of London

CS4FN

institute of CODING

Improving Computing Activities Using Semantic Waves

Paul Curzon

Semantic Waves is a theory by **Karl Maton** (2013), part of Legitimation Code Theory. See link below for how it is being applied to Computer Science

teachinglondoncomputing.org/semantic-waves/

Twitter: @cs4fn

What are semantic waves?

- An educational theory by Maton (2013):
  - a simple but powerful theory of **how to teach concepts**.
  - successfully applied across MANY disciplines.
- We have applied it to Computing
- We draw heavily on his work, and diagrams are adapted for computing from his papers

Find out more at: legitimationcodetheory.com/

K. Maton. 2013. Making semantic waves: a key to cumulative knowledge-building. Linguistics and Education 24, 8-22 (2013).

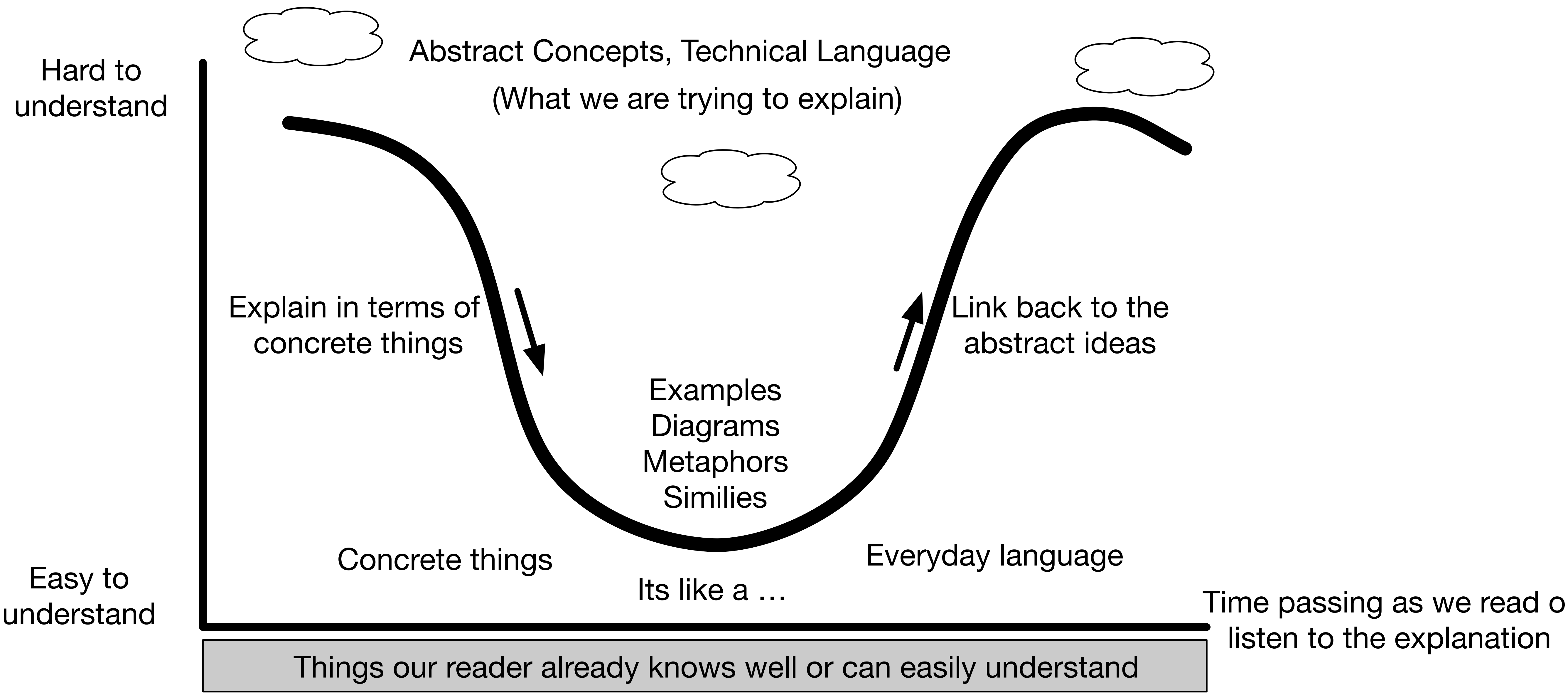
What can semantic waves do?

- A way to
  - think about what a good explanation / learning experience is
    - whether written, multimedia or spoken.
  - think about why metaphor and unplugged teaching works (and why sometimes they might not).
  - evaluate lesson plans /online resources,
  - teach students how to write good explanations.

Mastery of a subject

- To master a subject you must both
  - Master the **technical language**, and
  - Deeply **understand the abstract concepts**
- eg a phrase like “a while loop” has precise, deeply packed technical meanings to an expert programmer
- The secret to good teaching is to link
  - **abstract, technical** concepts and language to ..
  - **concrete, everyday** concepts and language

A good learning experience follows a wave pattern



Queen Mary University of London

CS4FN

institute of CODING

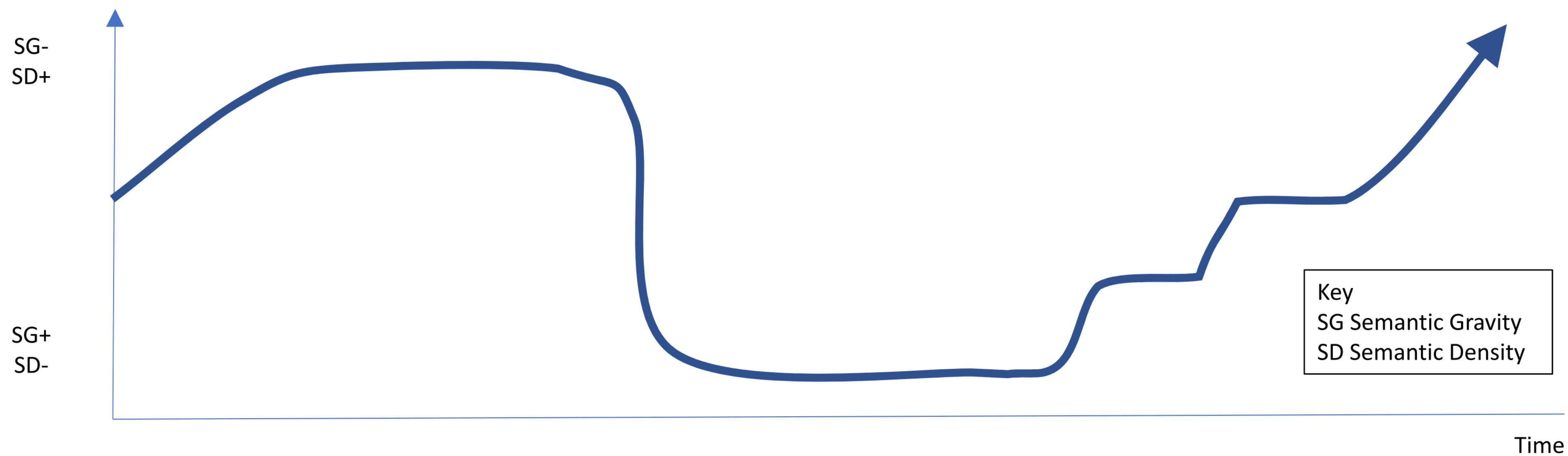
Analysis of an unplugged activity (crazy characters)

This example references Barefoot's 'Crazy Characters' activity and how Karl Maton and Jane Waite used Semantic Waves to analyse it.

You may wish to PAUSE and read the links before continuing…

https://bit.ly/BarefootCrazyCharacters
https://bit.ly/SemanticCrazyCharacters

Analysis of an unplugged activity (crazy characters)



Lesson Plan Steps	Explain you are going to use a new word – can they listen out?	Share the learning intention.	Say you are going to use the algorithm now.	Read out your steps and learners draw the crazy characters. Model adding extra detail.	Ask pupils to show what they have drawn. I didn't expect that.	How could you change that?	Ask what the algorithm was. Explain what an algorithm is
Semantic Profile Notes	**Signalling** A signal that a high is coming on the semantic profile.	**Concept Introduction** This is what you are going to learn about.	**Connecting** Connecting the theory to the concrete.	**Concrete activity** Practical activity with high semantic gravity. Learners are adding knowledge if the meaning is connected. The extra detail adds flow.	**Counter expectancy** Alternative options are introduced, increasing density.	**Staged return** Density increases as context is reduced	**Packing** Develop/reveal the definition and pack the concept.

Queen Mary University of London

CS4FN

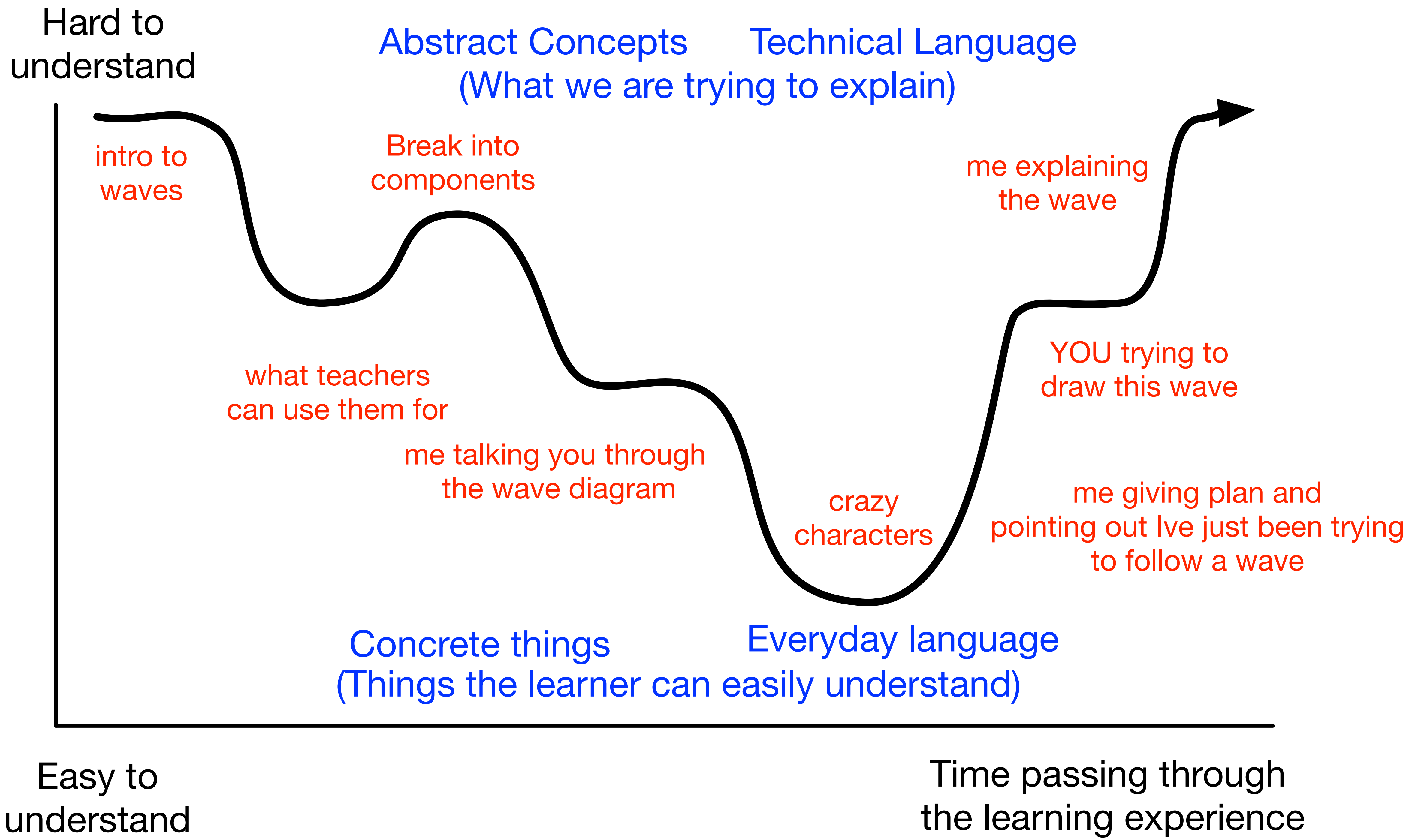
institute of CODING

Sketch the profile of the talk so far

The lesson plan of the talk so far:

1. Introduced the term semantic wave - a theory by Maton
2. Outline some practical things a teacher can do with it
3. Break mastery into component parts
4. Explain it in terms of a diagram
5. Discuss Crazy Characters activity
6. Give plan, Point out I’ve been following some kind of wave
7. I’m now getting you to apply it to an example.

A course grained version

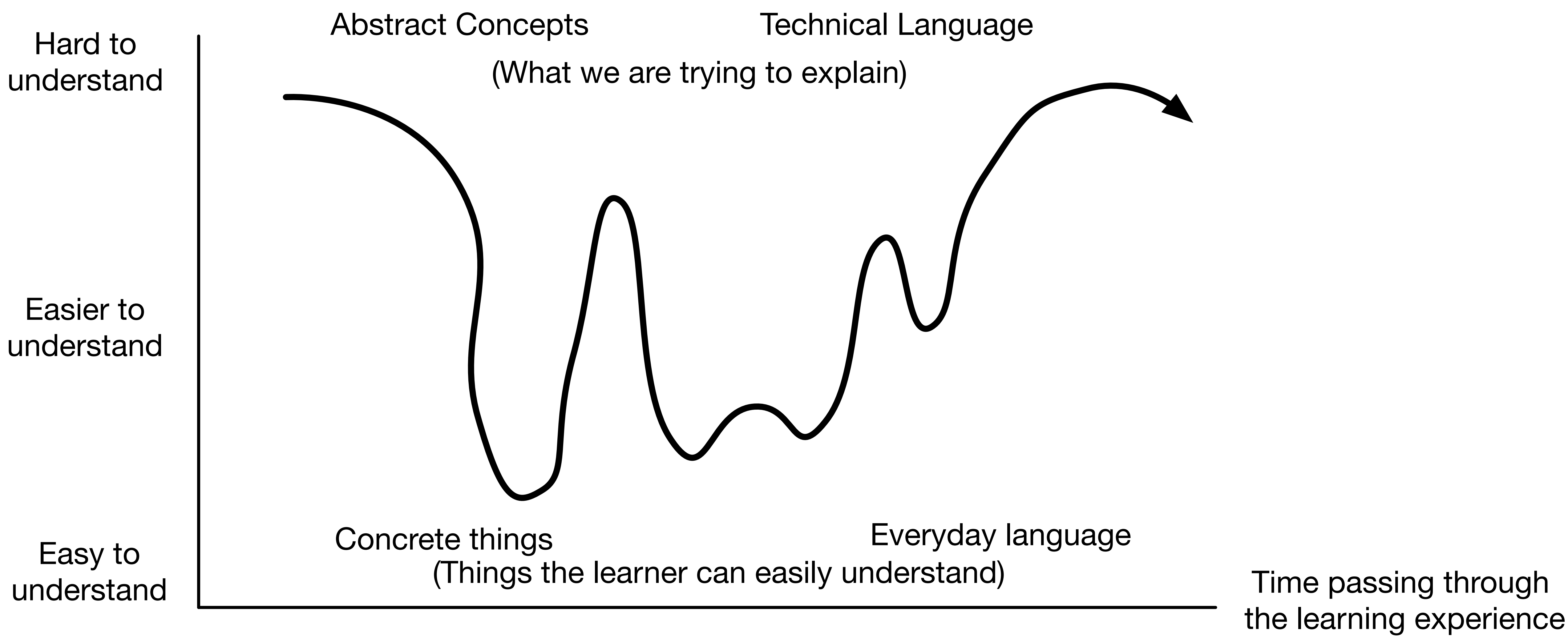


Queen Mary University of London

CS4FN

institute of CODING

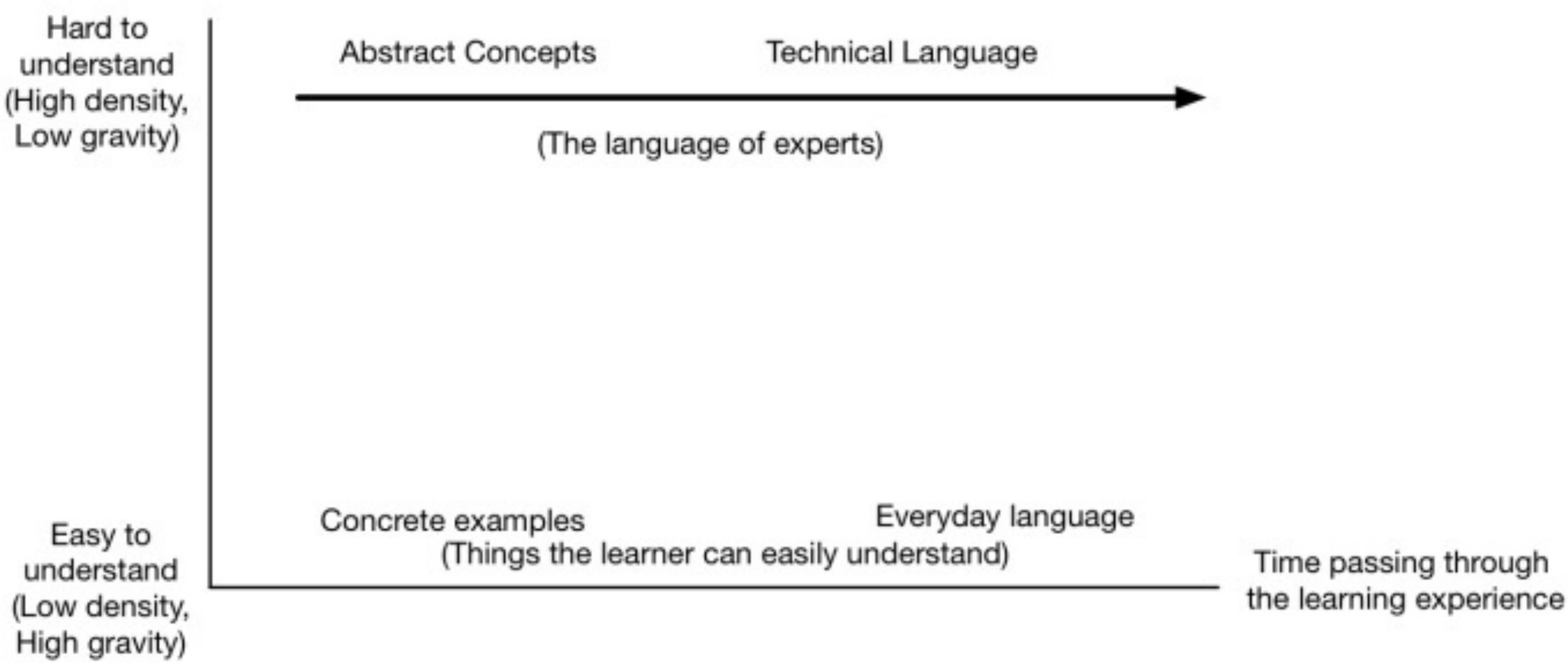
Good explanations of complex concepts have waves within waves



How to provide a bad learning experience …

Flatlining your students

(eg a lot of wikipedia…stereotypical university lectures)



Queen Mary University of London

CS4FN

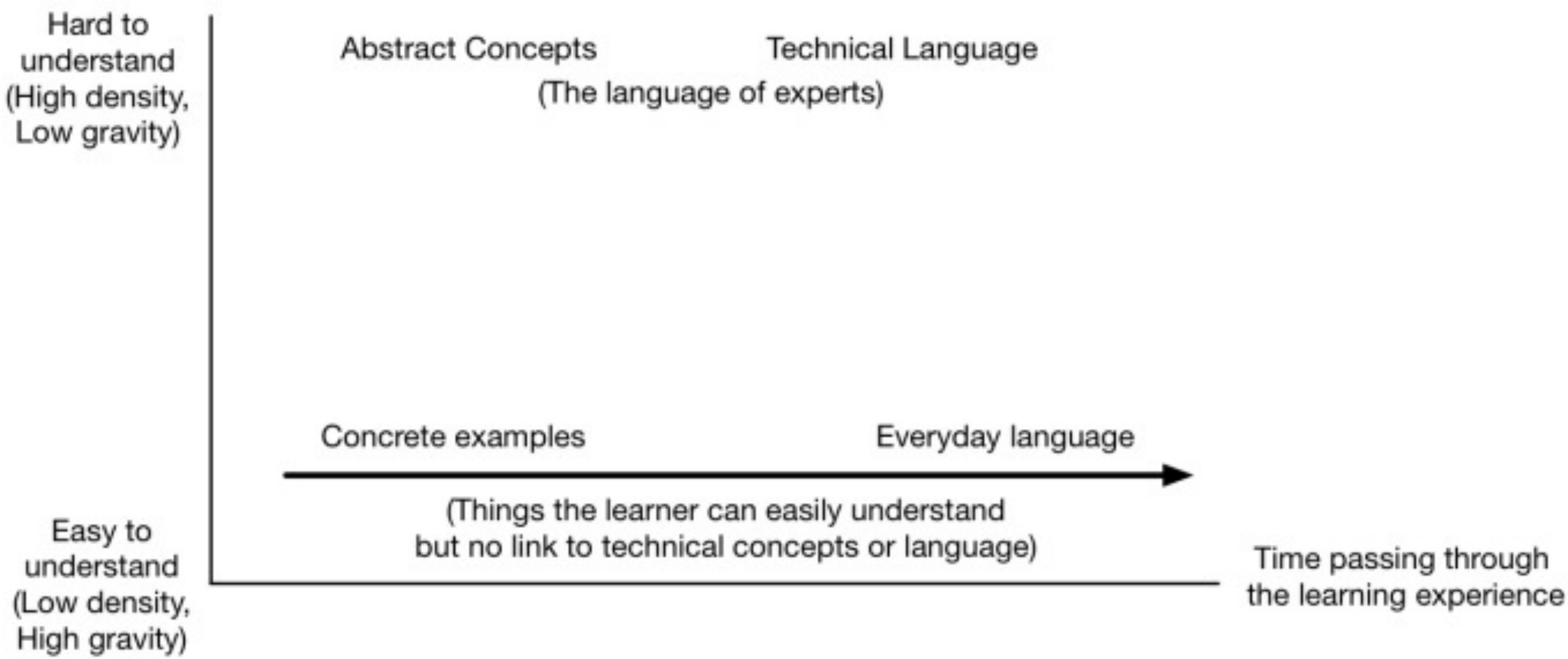
institute of CODING

For example…

“The ***while* construct** consists of a **block** of **code** and a **condition**/**expression**. The **condition/expression** is **evaluated**, and if the **condition/expression** is ***true***, the **code** within the **block** is **executed**. This repeats until the **condition/expression** becomes **false**.” -Wikipedia.

- Is this something about buildings, spies, guillotines, people being happy or unhappy, and/or lie detectors?…
- If you didn’t know what a while construct was before you still won’t unless you are already have mastery of everything in red.

More ways than one to flatline… how unplugged goes wrong



Queen Mary University of London

CS4FN

institute of CODING

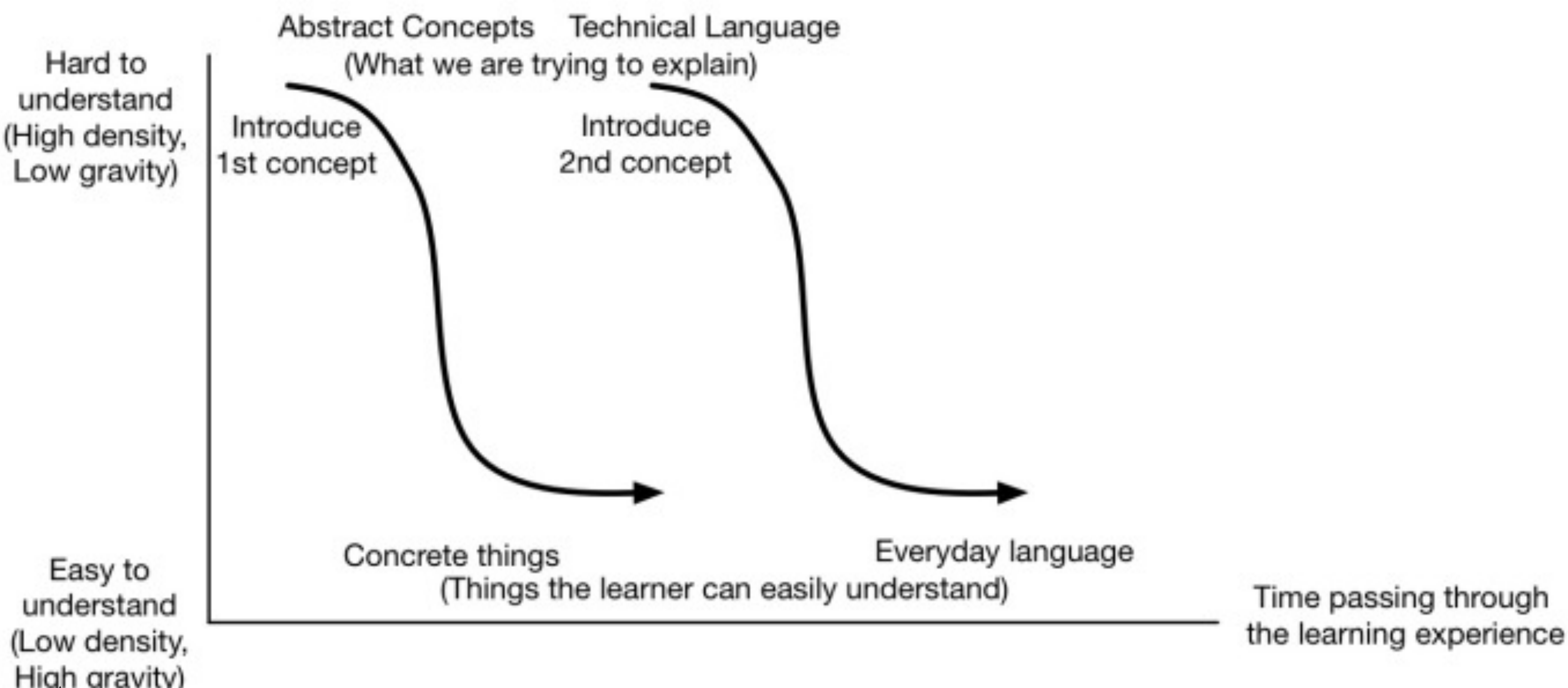
For example, you all know about cooking already so I'll use a cooking example to explain

"Recipes have structure. They start with a name. They have an ingredient list first. Then the main part of the recipe is a series of instructions to follow. Instructions can tell you to keep doing something eg 'stir the mixture until it is smooth'"

- Weren't we learning about computing not cookery???? What has this got to do with anything?

- Done like this the recipe analogy (which is a good analogy if used well) explains nothing.

Another way to get it wrong… never repacking the concepts



Queen Mary University of London

CS4FN

institute of CODING

Understanding what an algorithm is

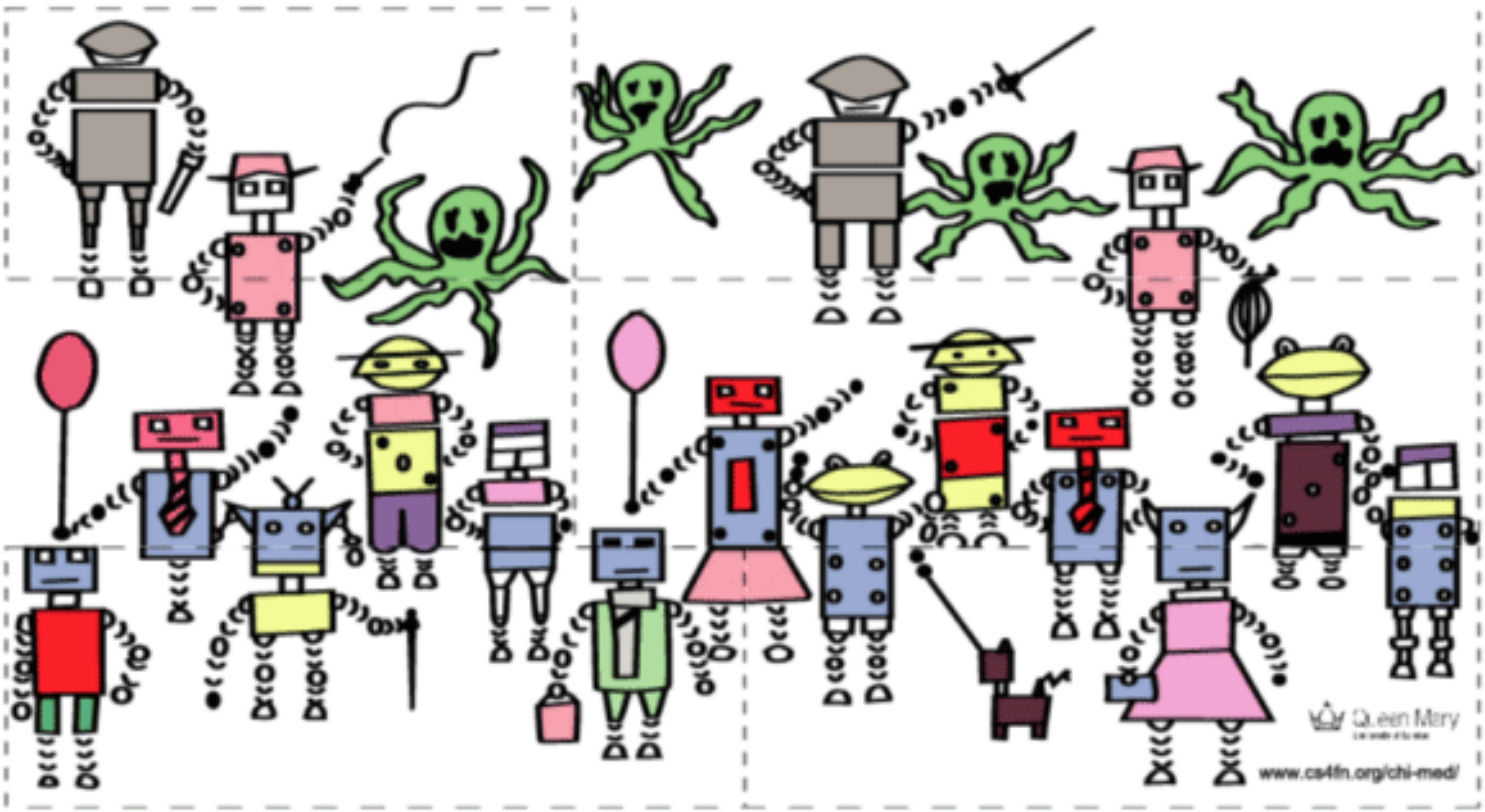
By the end

- You will be able to explain the concept of **an algorithm**

But we will use a magic trick to do this

Let's explore an example unplugged activity: Teleporting Robot

How many robots?
www.cs4fn.org/magic/



To do the trick…

1. Build the jigsaw with the smaller pieces on the **left**.
2. Count the robots and remember how many there are
3. Mix up the pieces
4. Rebuild the jigsaw with the smaller pieces on the **right**.
5. Count the robots

A robot has disappeared

You too can do magic ...

- You can download the pdf, print and cut out the jigsaw (or see the puzzle online) at
  - **https://bit.ly/TeleportingRobot**
- **You can do the trick, even without knowing why it works:**
  - Just carefully follow the steps.

Self-working Tricks

- Magicians call a trick like this a “**self-working**” trick
- Follow the steps in the right order and the trick just works
  - Even if you have no idea what you are doing
- Computer Scientists call it an **algorithm**
  - I wrote it in English for a human to follow

Programs

- I wrote the trick's "algorithm" in English for a human to follow
- With programs we write algorithms in a programming language for a computer to follow
  - Computers have no idea what they are doing
  - They must have precise steps they can follow blindly
  - They can only follow algorithms
  - They have to always work

Exercise:
Explain “algorithm”

- Explain what is meant by an algorithm illustrating your answer with an example

What is an algorithm?

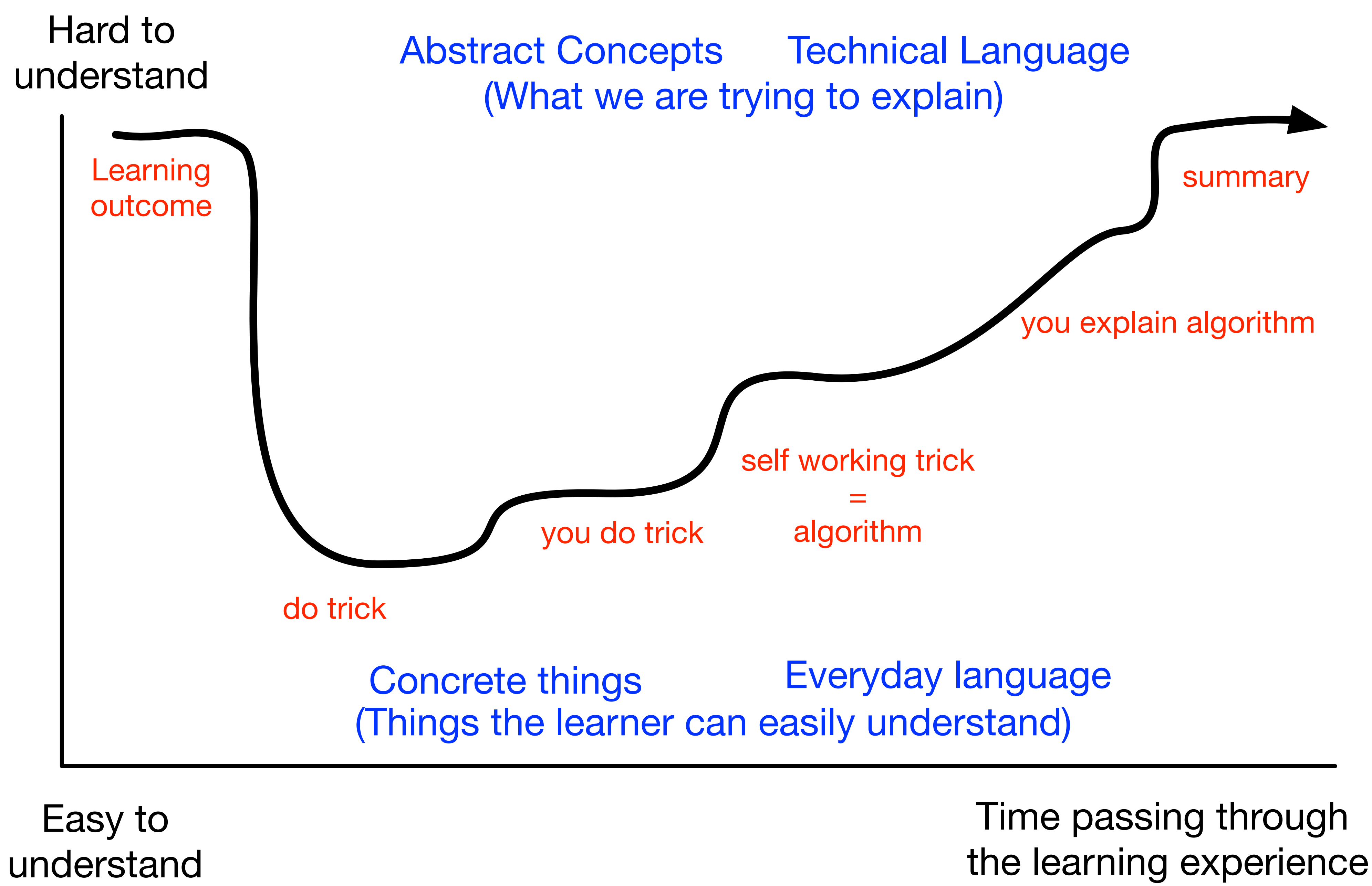
- Precise instructions of how to do some specific thing
- Must be followed exactly and in the specified order.
- If done so they guarantee to do that thing correctly.

- A computational agent should be able to blindly follow the instructions
- Without ANY understanding of what they are doing or why.
- The result should still be obtained.

Exercise:
Sketch the profile

- Sketch the profile of the activity we just did

My sketch of the profile



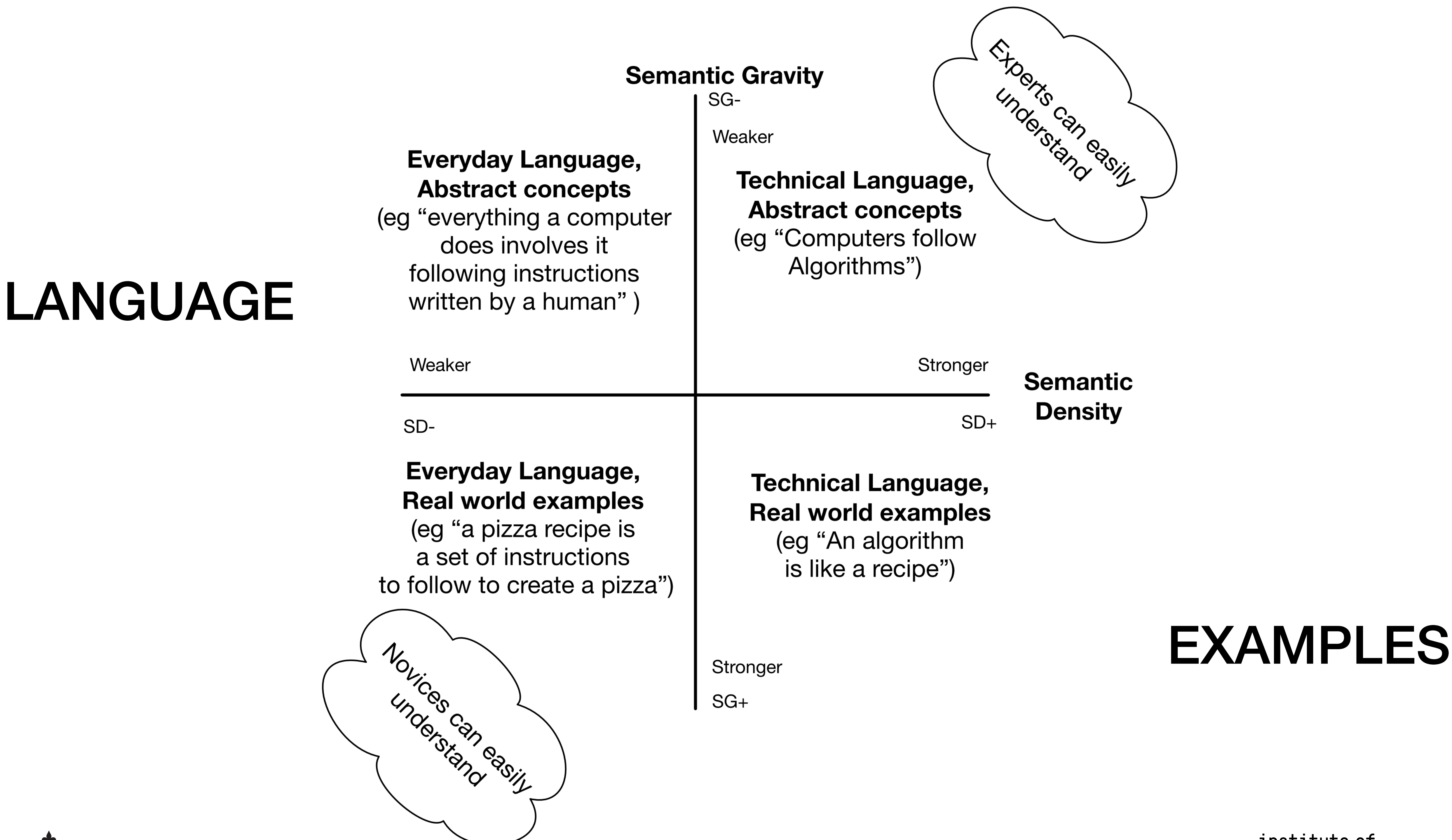
Queen Mary University of London

CS4FN

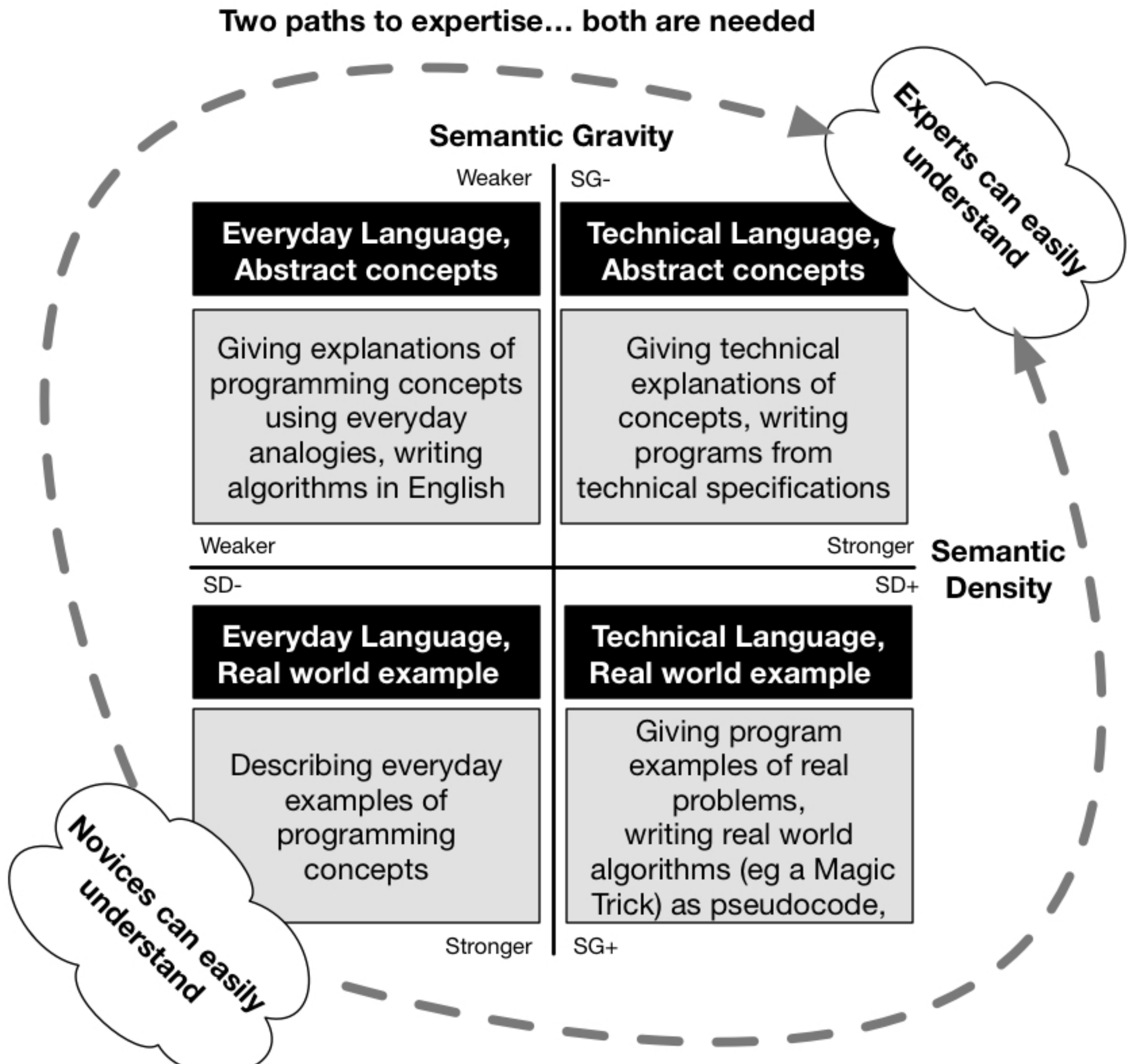
institute of CODING

Semantic Density: how technical the language

Semantic Gravity: how abstract/concrete the examples



Two paths to expertise



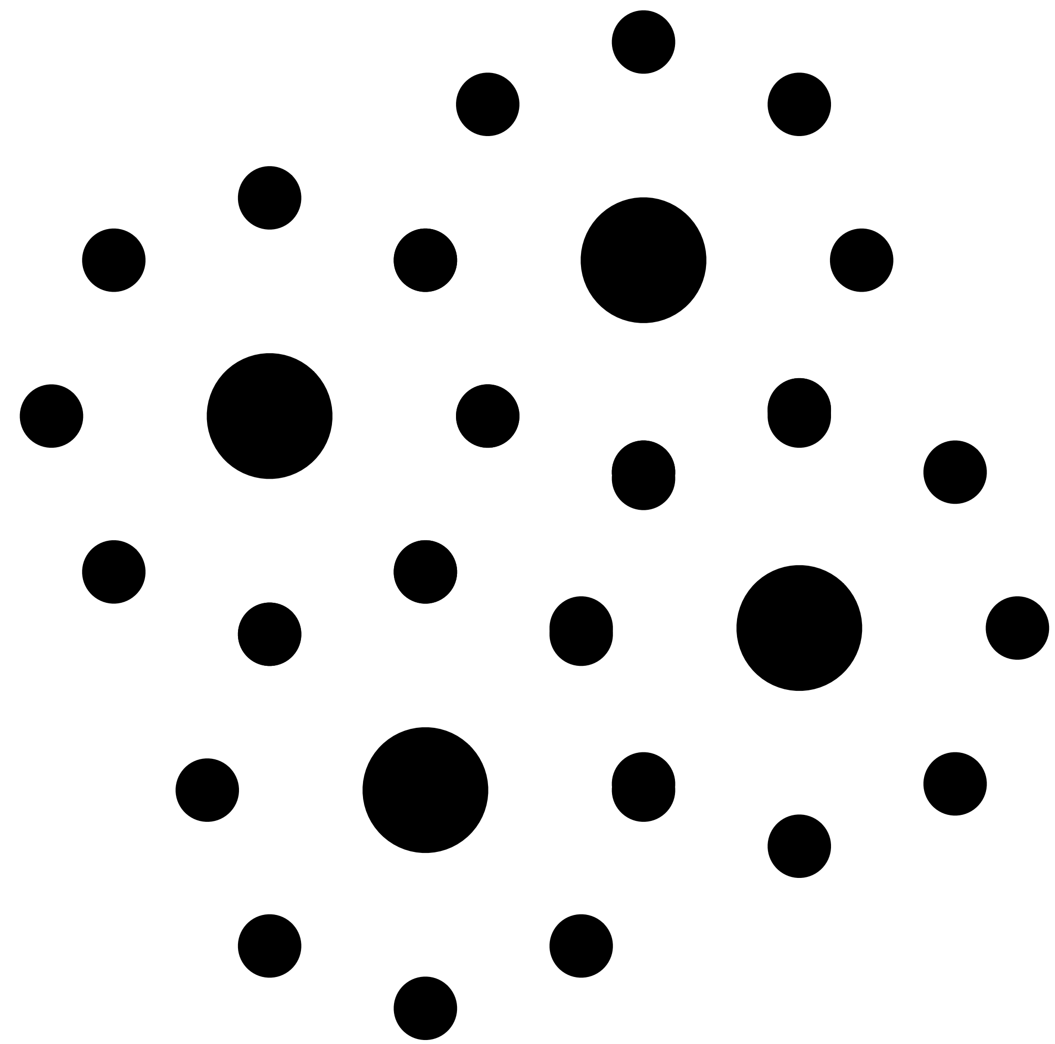
Adapted to apply to programming from Maton (2013)

Optical Illusion Art:
A fun coding activity?

- Abstract art such as the Op Art of Bridget Riley can add a fun twist to
  - learning to code
  - learning about vector (v bitmap) graphics
- Drawing sequences of simple shapes that create optical illusions playing with our perception

Riley 1

1. Circle, Black, 24, 20, 4, 4
2. Circle, Black, 19, 8, 4, 4
3. Circle, Black, 12, 25, 4, 4
4. Circle, Black, 7, 13, 4, 4
5. Circle, Black, 20, 2, 2, 2
6. Circle, Black, 15, 4, 2, 2
7. Circle, Black, 25, 4, 2, 2
8. Circle, Black, 8, 7, 2, 2
9. Circle, Black, 3, 9, 2, 2
10. Circle, Black, 13, 9, 2, 2
11. Circle, Black, 27, 9, 2, 2
12. Circle, Black, 1, 14, 2, 2
13. Circle, Black, 15, 14, 2, 2
14. Circle, Black, 25, 14, 2, 2
15. Circle, Black, 20, 16, 2, 2
16. Circle, Black, 30, 16, 2, 2
17. Circle, Black, 3, 19, 2, 2
18. Circle, Black, 13, 19, 2, 2
19. Circle, Black, 8, 21, 2, 2
20. Circle, Black, 18, 21, 2, 2
21. Circle, Black, 32, 21, 2, 2
22. Circle, Black, 6, 26, 2, 2
23. Circle, Black, 20, 26, 2, 2
24. Circle, Black, 30, 26, 2, 2
25. Circle, Black, 25, 28, 2, 2
26. Circle, Black, 8, 31, 2, 2
27. Circle, Black, 18, 31, 2, 2
28. Circle, Black, 13, 33, 2, 2



cs4fn White Disks, inspired by Riley's White Disks

How not to teach programming…

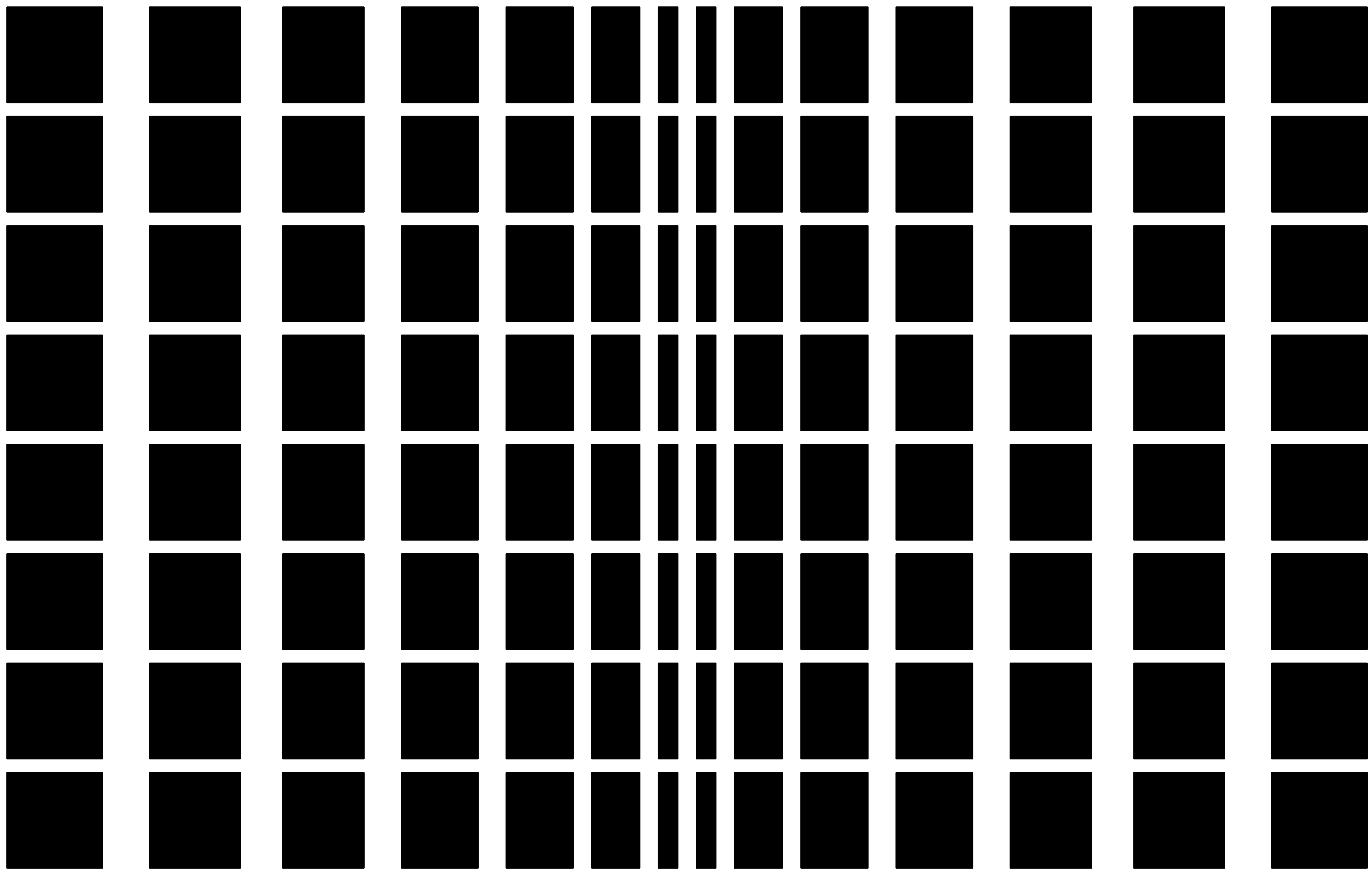
- We are going to learn to program.Type in the following code, then run it…What does it do?

- “Copy code” is thought to be a poor way to teach programming

- What is the semantic profile of such an activity?

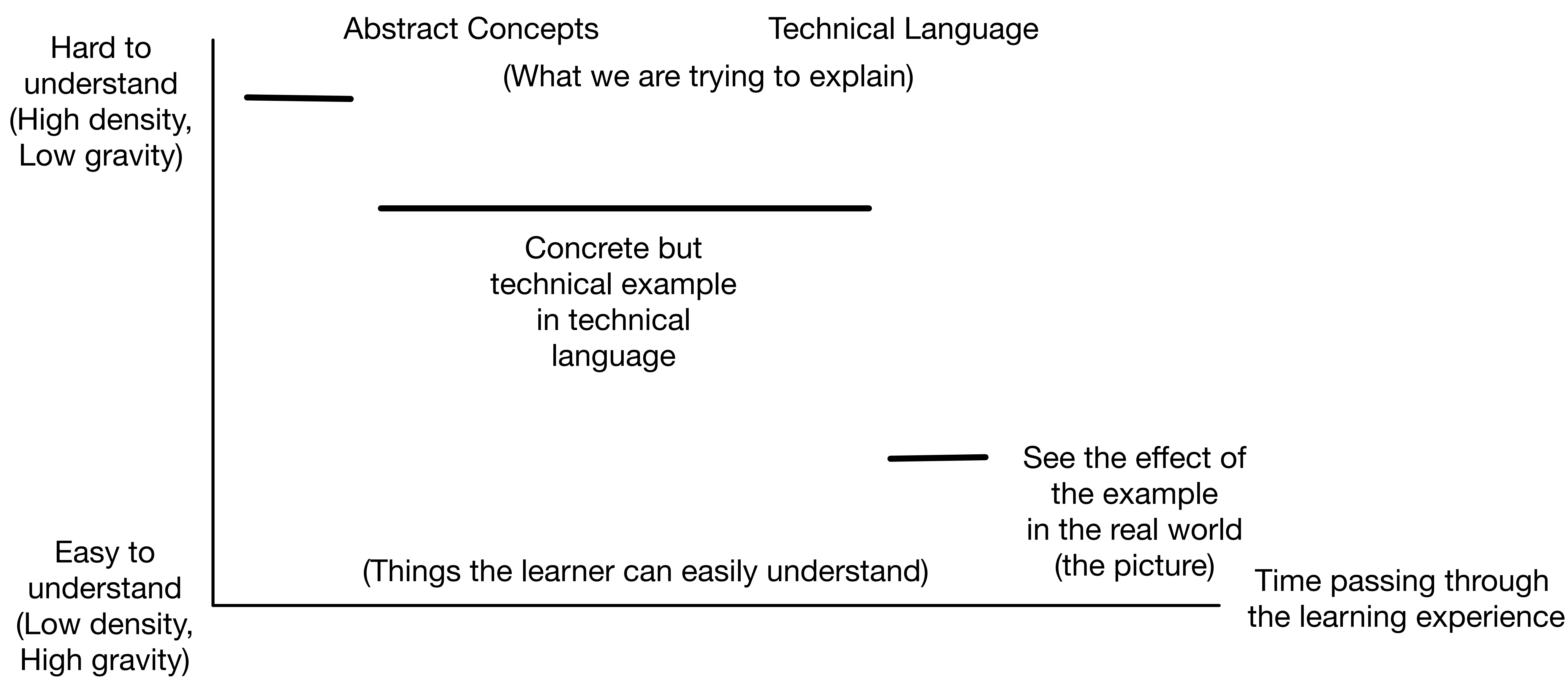
```
RileyStrip:
1.  Rectangle, Black, 0, 0, 2, 2
2.  Rectangle, Black, 3, 0, 1.9, 2
3.  Rectangle, Black, 5.8, 0, 1.7, 2
4.  Rectangle, Black, 8.3, 0, 1.6, 2
5.  Rectangle, Black, 10.5, 0, 1.4, 2
6.  Rectangle, Black, 12.3, 0, 1, 2
7.  Rectangle, Black, 13.7, 0, 0.4, 2
8.  Rectangle, Black, 14.5, 0, 0.4, 2
9.  Rectangle, Black, 15.3, 0, 1, 2
10. Rectangle, Black, 16.7, 0, 1.4, 2
11. Rectangle, Black, 18.7, 0, 1.6, 2
12. Rectangle, Black, 21.1, 0, 1.7, 2
13. Rectangle, Black, 23.7, 0, 1.9, 2
14. Rectangle, Black, 26.6, 0, 2, 2

1.  RileyStrip, 1, 1
2.  RileyStrip, 3.3, 1
3.  RileyStrip, 5.6, 1
4.  RileyStrip, 7.9, 1
5.  RileyStrip, 10.2, 1
6.  RileyStrip, 12.5, 1
7.  RileyStrip, 14.8, 1
8.  RileyStrip, 17.1, 1
```

What the program produces



cs4fn Curves From Rectangles: inspired by Riley's images

Down escalator with no unpacking or repacking



Queen Mary University of London

CS4FN

institute of CODING

How can we do better?

A better semantic wave? A better learning experience?

- Do it unplugged first: On squared paper, follow these steps to draw this picture
  - Let’s look at what you just did…
- Use a pseudocode first
- Have them create their own Op Art image/instructions… perhaps using other shapes
- Give the ready-typed in code. Ask them to read it and predict what it might do.
- Ask them to make changes and predict what will happen.

RileyStrip:

1. Rectangle, Black, 0, 0, 2, 2
2. Rectangle, Black, 3, 0, 1.9, 2
3. Rectangle, Black, 5.8, 0, 1.7, 2
4. Rectangle, Black, 8.3, 0, 1.6, 2
5. Rectangle, Black, 10.5, 0, 1.4, 2
6. Rectangle, Black, 12.3, 0, 1, 2
7. Rectangle, Black, 13.7, 0, 0.4, 2
8. Rectangle, Black, 14.5, 0, 0.4, 2
9. Rectangle, Black, 15.3, 0, 1, 2
10. Rectangle, Black, 16.7, 0, 1.4, 2
11. Rectangle, Black, 18.7, 0, 1.6, 2
12. Rectangle, Black, 21.1, 0, 1.7, 2
13. Rectangle, Black, 23.7, 0, 1.9, 2
14. Rectangle, Black, 26.6, 0, 2, 2

1. RileyStrip, 1, 1
2. RileyStrip, 3.3, 1
3. RileyStrip, 5.6, 1
4. RileyStrip, 7.9, 1
5. RileyStrip, 10.2, 1
6. RileyStrip, 12.5, 1
7. RileyStrip, 14.8, 1
8. RileyStrip, 17.1, 1

Improving activities

- Check your lesson plan (and subparts) follows a wave
  - avoid flatlining your students
  - aim for waves within waves
- Include BOTH unpacking and repacking activities
  - linking examples / everyday language to the technical language / abstract concepts.
- Make sure **the students do the repacking** (not just you)
- Make use of both routes to Mastery
  - via concrete examples AND via everyday language

Semantic wave analysis can improve our teaching

Thinking about Semantic Waves applied to an activity or the way that it is delivered can …

- Help us see why a particular delivery of an activity might not work well
- Why other deliveries may be an improvement
- How both good and bad can be improved

Thank You

- See our web page on Semantic Waves in Computing
  teachinglondoncomputing.org/**semantic-waves**

Queen Mary University of London

CS4FN

institute of CODING