Lecture for mathematics teacher students

in the course mathematical modelling and problem solving
given by Philip Gerlee (autumn 2016)

Dag Wedelin, 2015-2016

models and modelling

What is mathematical modelling?

“a convenient way to represent reality so that we more easily can draw conclusions about it“

<u>simplification</u>

The first step of applied mathematical problem solving

Can be starting point of:

- a scientific theory
- an engineering method
- a software system

Some modelling experiences

Real problems are typically *ill-structured*!

Delineate, simplify, assume!

iterative and intuitive.

Models are wrong and useful

Also very simple models can be useful!

Conclusions depend on the choice of model.

A model or a theory?

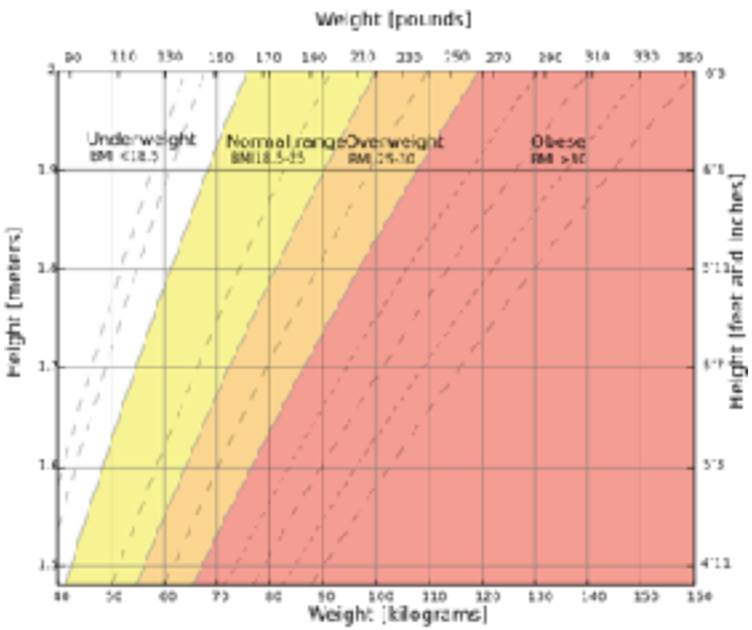
<u>Model</u>: intentional simplification often used for specific purpose. Not perfect but useful!

<u>Theory</u>: Often more general and comprehensive. A model that is assumed to be”correct”.

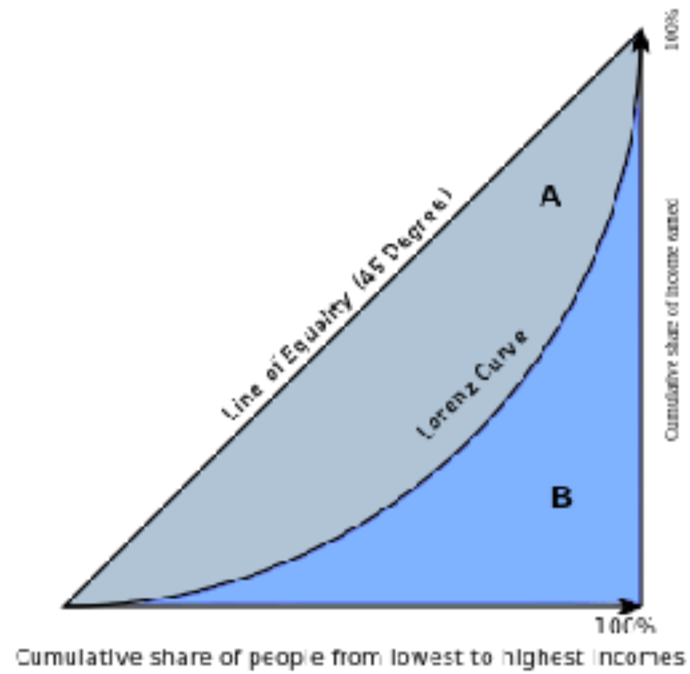
However, by modelling you may sometimes discover theories! (kepler, surprise)

Descriptive and prescriptive models

BMI = m/h^2

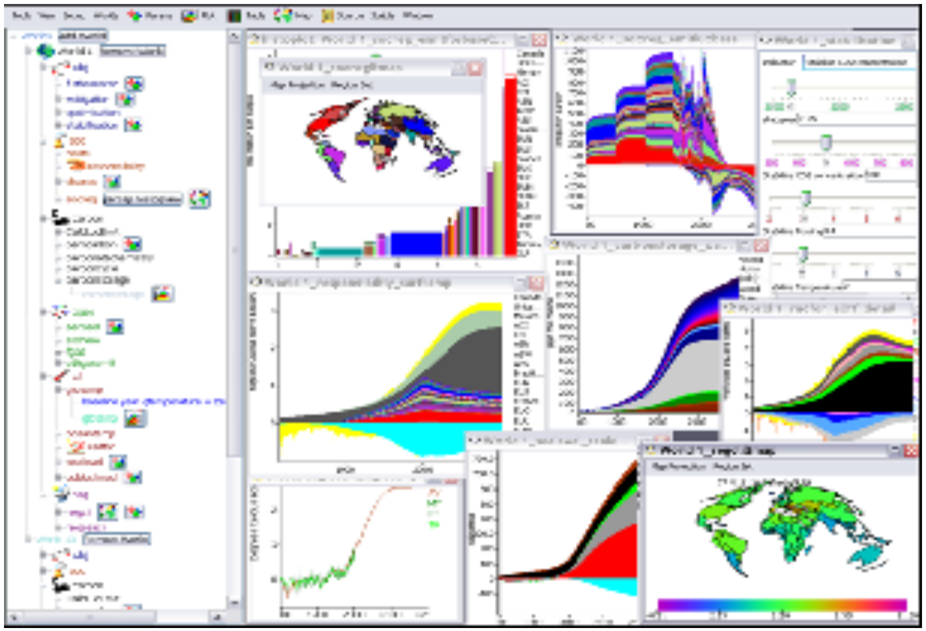


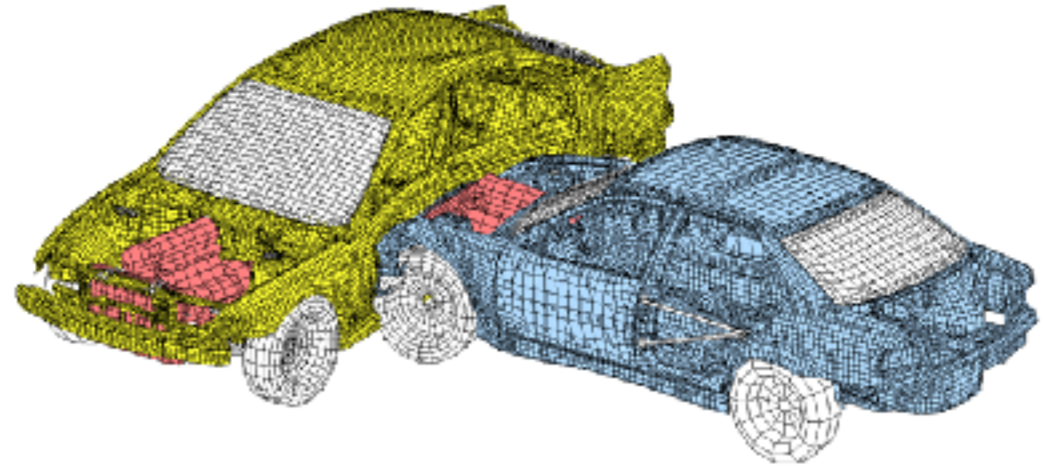
GINI coefficient = A/(A+B)



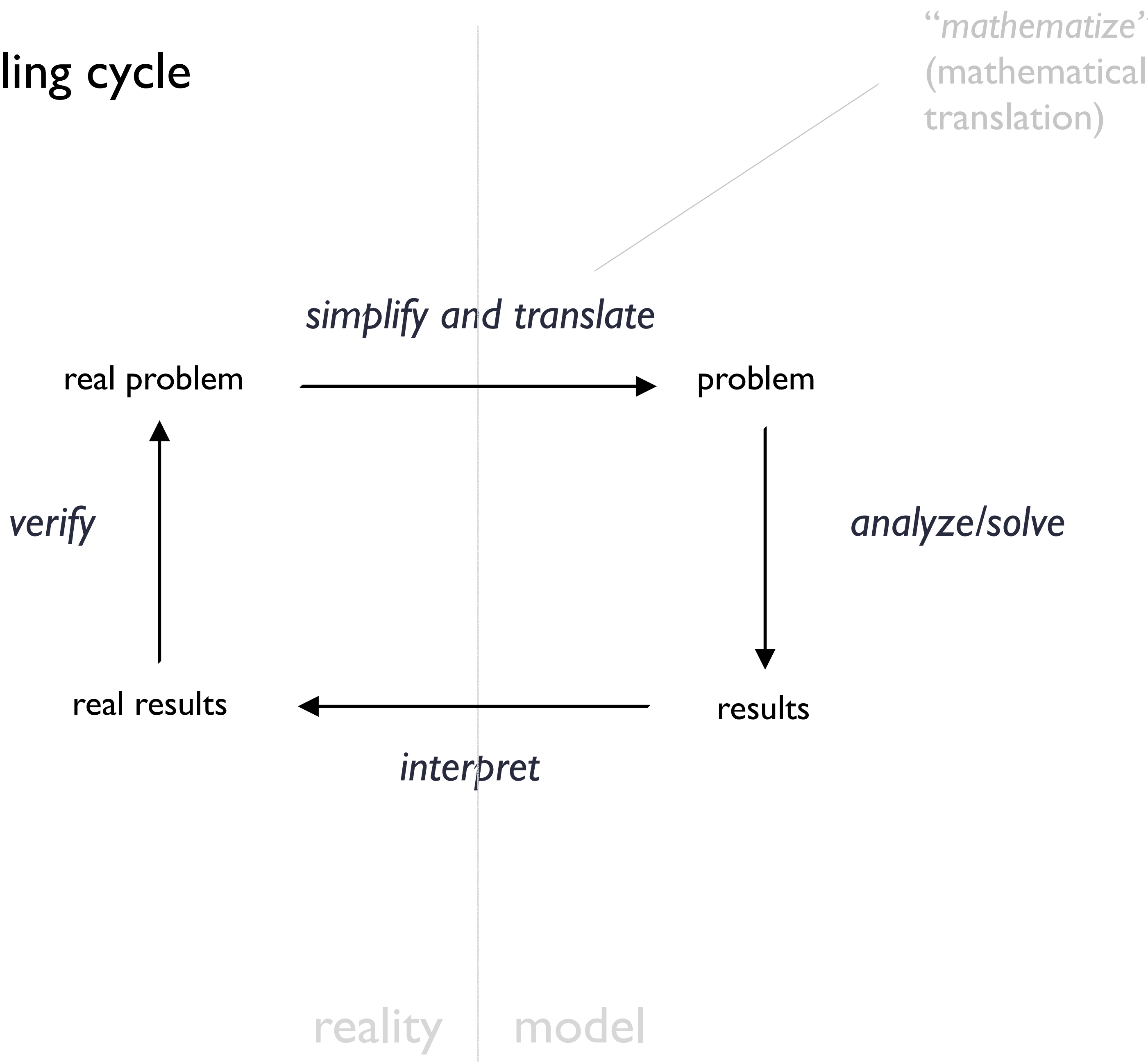
These models are artifacts, based on ideas of what we think are sensible concepts!

We can scale up the math!

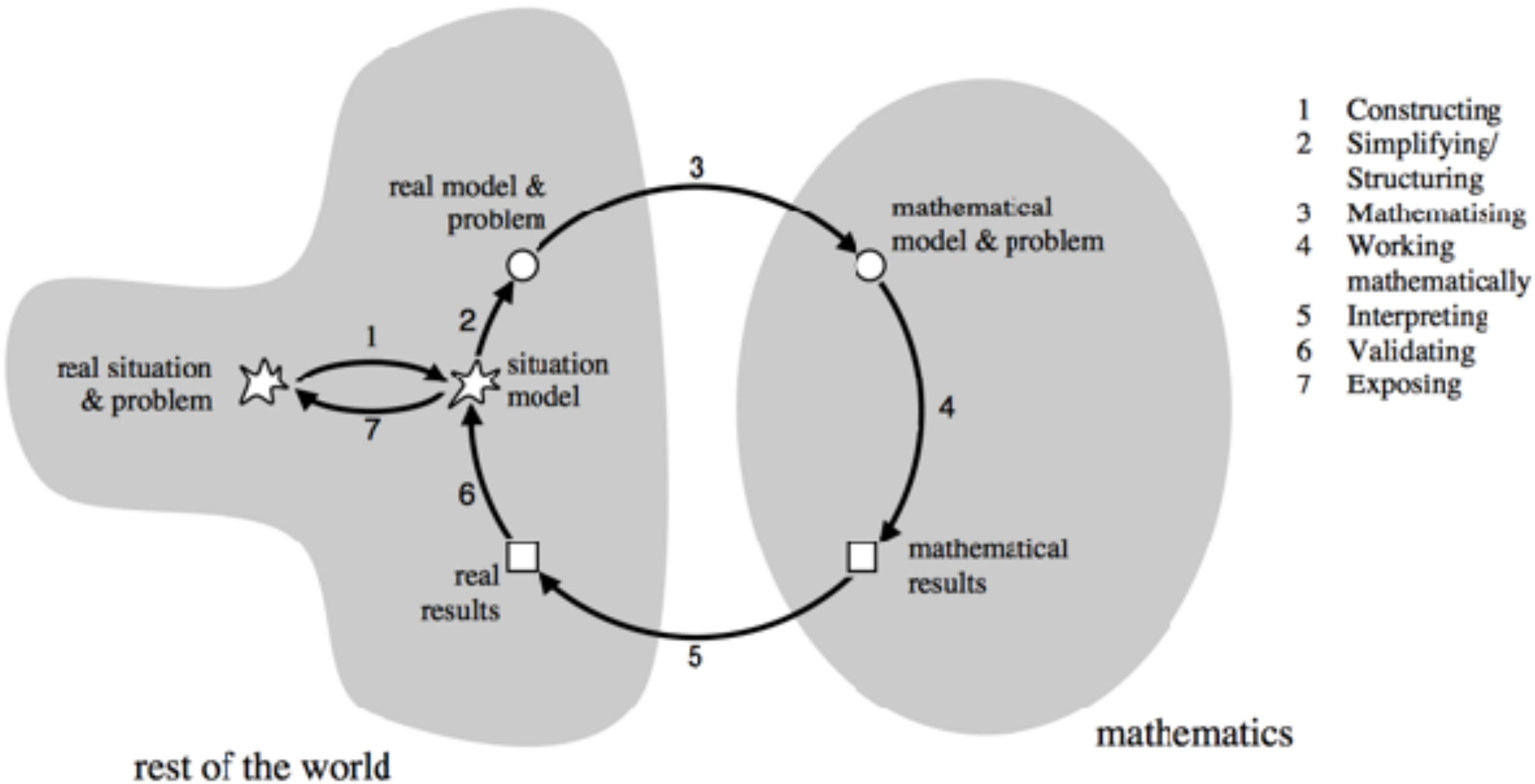




Modelling cycle



More elaborate modelling cycle



Blum and Leiß (2005)

First, the problem situation has to be understood by the problem solver, that is a *situation model* has to be constructed. Then the situation has to be simplified, structured and made more precise, leading to a *real model* of the situation…

problem solving

Why problem solving?

It's the ***variation***!

(Problems come in infinite variations - it will never be sufficient to learn a finite set of given methods)

What is problem solving?

"You are engaged in problem solving when you are trying to achieve something, and you do not know a straightforward way to do so."

(A. Schoenfeld)

Polya control strategy in four "phases"

1 understand the problem

2 make a plan (select the approach)

3 carry out the plan

4 look back

Each phase is associated with some standard questions and a set of heuristic strategies.

Some similarity to the modeling cycle (think about!)

Does not highlight the explorative nature of non-trivial problem solving

Heuristic strategies (Polya and earlier)

simplify, specialize, make example, split in parts, draw figure, find related problem, consider extreme cases, guess and check, systematically test …

"rules of discovery and invention" (Polya)

There are quite a few more…

Heuristics are not algorithms, but rather general descriptions of what you commonly do when you solve problems!

"to understand a problem better or to make progress towards its solution" (Schoenfeld)

Think about some heuristics and how we have used them in different problems! (also the modeling cycle…)

Some insights of problem solving

<u>It is normal to be stuck and that your ideas don't work!</u>

<u>Since you are searching you are in some sense always "on the right track"!</u>

<u>Search for small steps towards the solution!</u>

<u>Investigate the problem</u>: draw figure, make examples, simple cases, ...

<u>Explore ways forward</u>: ask the right question, create subgoal, solve simpler problem, test cases, iteratively improve, work backwards, ...

Be aware of what you are doing (meta-cognitive awareness)

Solve problems from scratch and use knowledge to accelerate your own problem solving.

Always do something, never stop! Work actively with pencil and paper!

You can develop several roles within yourself!

solve!

ask questions!

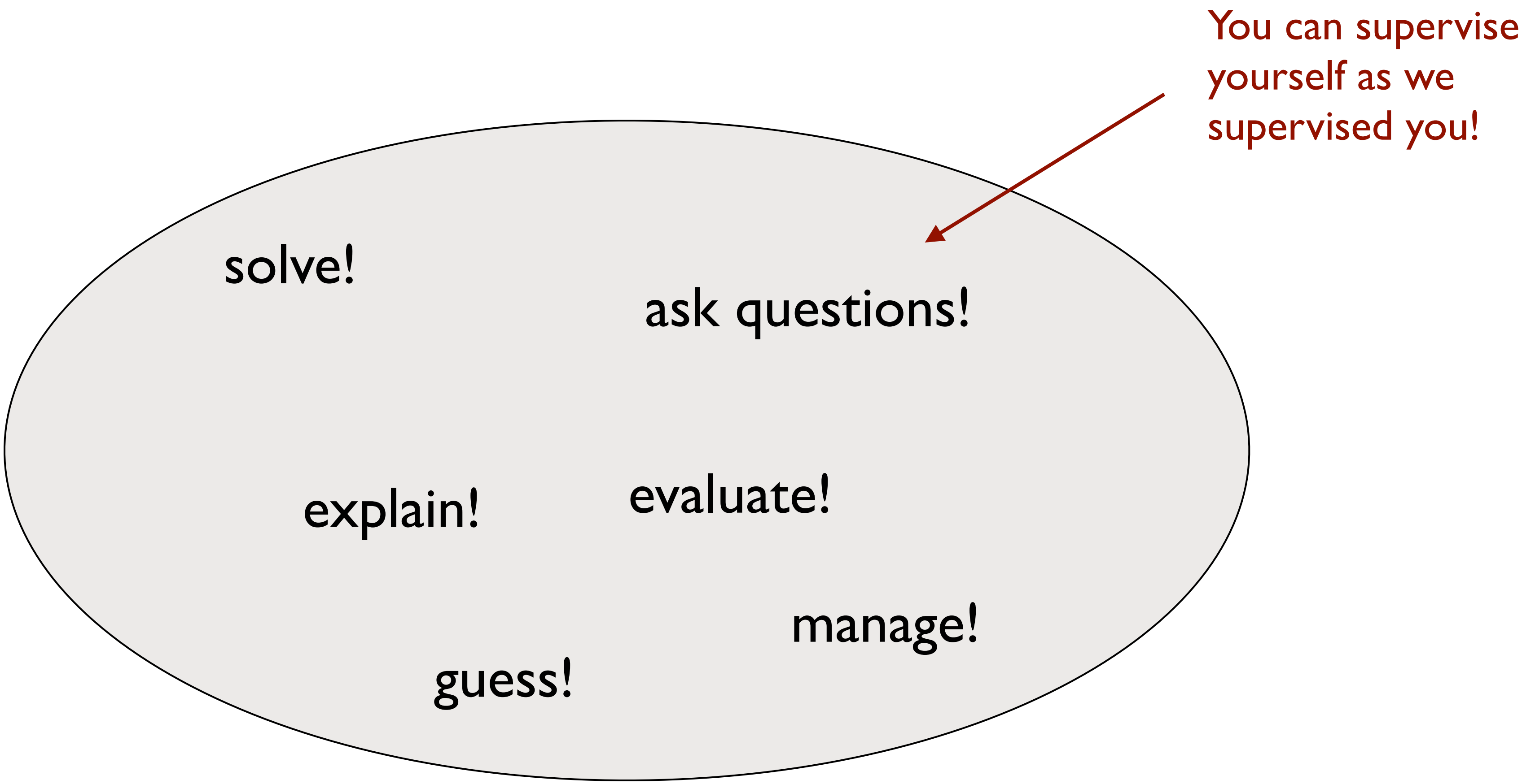
explain!

evaluate!

manage!

guess!

You can develop several roles within yourself!



why is this a challenge for students?

My starting point

Many software engineering and and computer science students do not understand where and how to use mathematics.

Not even the mathematics they already know!

```
import java.awt.*;

public abstract class Animation
  extends java.applet.Applet
    implements java.lang.Runnable {

    protected Dimension d;  // bitmap size
    protected Image im;  // extra image for drawing
    protected Graphics offscreen; // the offscreen bitmap to draw in
    protected int delay = 100; // in milliseconds
    protected Thread animationThread;

    final public void init() {
    d = getSize();
    im = createImage(d.width, d.height);
    offscreen = im.getGraphics();
    initAnimator();
    }

    //final public void paint(Graphics g) {update(g);}

    final public void update(Graphics g) {
    paintAnimator(offscreen);  // first draw offscreen to reduce flicker
    g.drawImage(im, 0, 0, this);  // then put on screen
    }

    // To be implemented in subclass that does the actual drawing
    protected void initAnimator() {} // init for drawing routines
    abstract protected void paintAnimator(Graphics g); // the actual drawing
will be here

    public void setDelay(int d) {delay=d;}

    public void start() {
    animationThread = new Thread(this);
    animationThread.start();
    }

    public void stop() {
    animationThread = null;
```

Solving *well-structured* problems in school

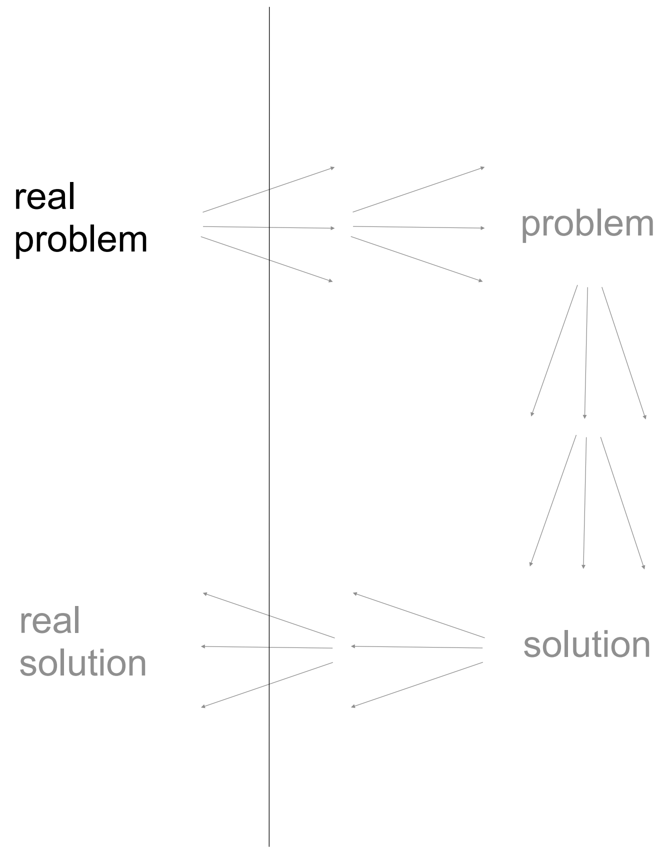
problem

known method

solution

no modelling!
no problem solving!

Solving *ill-structured* real-world problems



modelling and
problem solving!

What are the main challenges for students early in the course? (research project 2013)

Some modelling difficulties…

However, the main challenges are in problem solving:

1. <u>understanding the problem</u>

   "The first week we really didn't spend much time on trying to understand the problem and what the problem was. We just dove in and that kind of backfired sometimes…"

2. <u>exploring alternatives</u>

   "I don't think we would have got stuck like we did if we … tried to see different possibilities of getting forward, and not getting to an answer directly."

- *I initially found exercise 2 very hard and wasted lots of time trying to solve it. It was mainly because I kept trying to solve it using one approach the entire time, but in the end the approach proved to be all wrong. To avoid this the next time, I will work on becoming less stubborn, and more open to trying new approaches.*

- *I also wasted some time one exercise 3 because I had completely misunderstood the problem (I thought the problem was basically the shortest path problem.). So I ended up wasting lots of time on the solving the wrong problem. ... I learned from this that you should truly understand the problem before you start solving it.*

Student in module 3, 2014

Attitudes and expectations are also very important

Expect to try things out

Have open mind

Courage, patience, dare to use common sense

Self-confidence

"You need to have some courage to dare to try different things and not just sit there and wait…and if it doesn't work you must be prepared to discard it and try another one"

What is needed to be a good problem solver? (Schoenfeld)

Resources (knowledge of different kinds)

Heuristics (“tricks of the trade”)

Self-regulation (monitoring and control, “self-awareness”)

Appropriate attitudes and expectations (your “belief system”)

What is needed to be a good problem solver? (Schoenfeld)

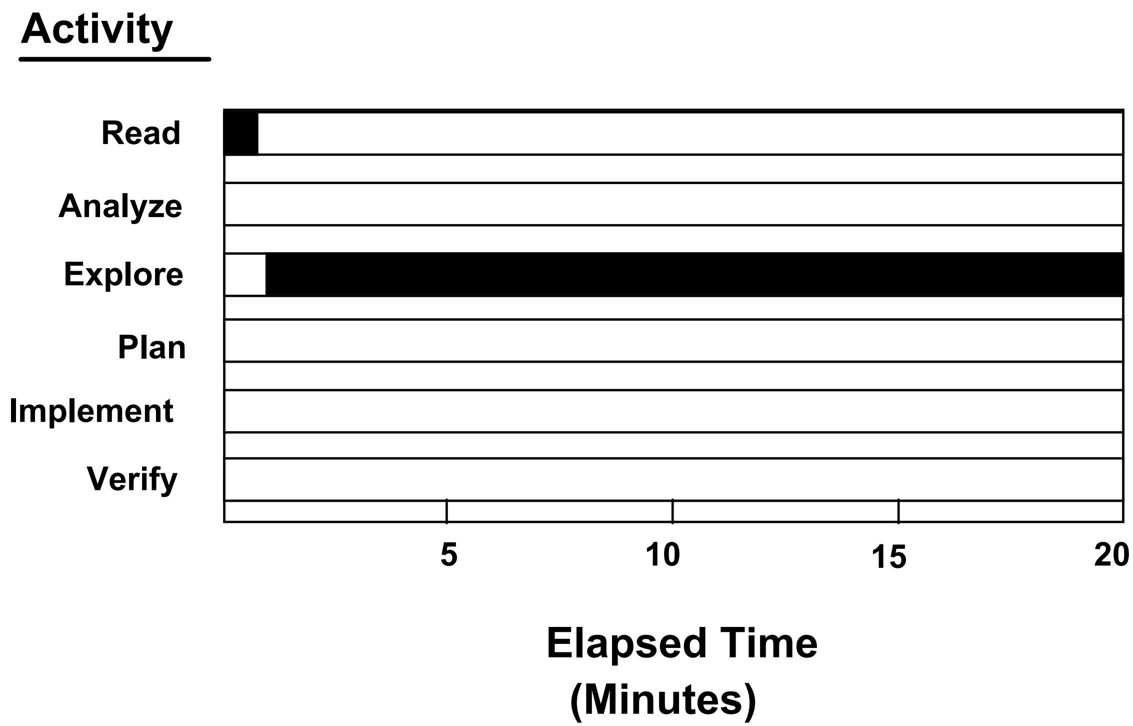
Resources (knowledge of different kinds)

Heuristics ("tricks of the trade")

Self-regulation (monitoring and control, "self-awareness")

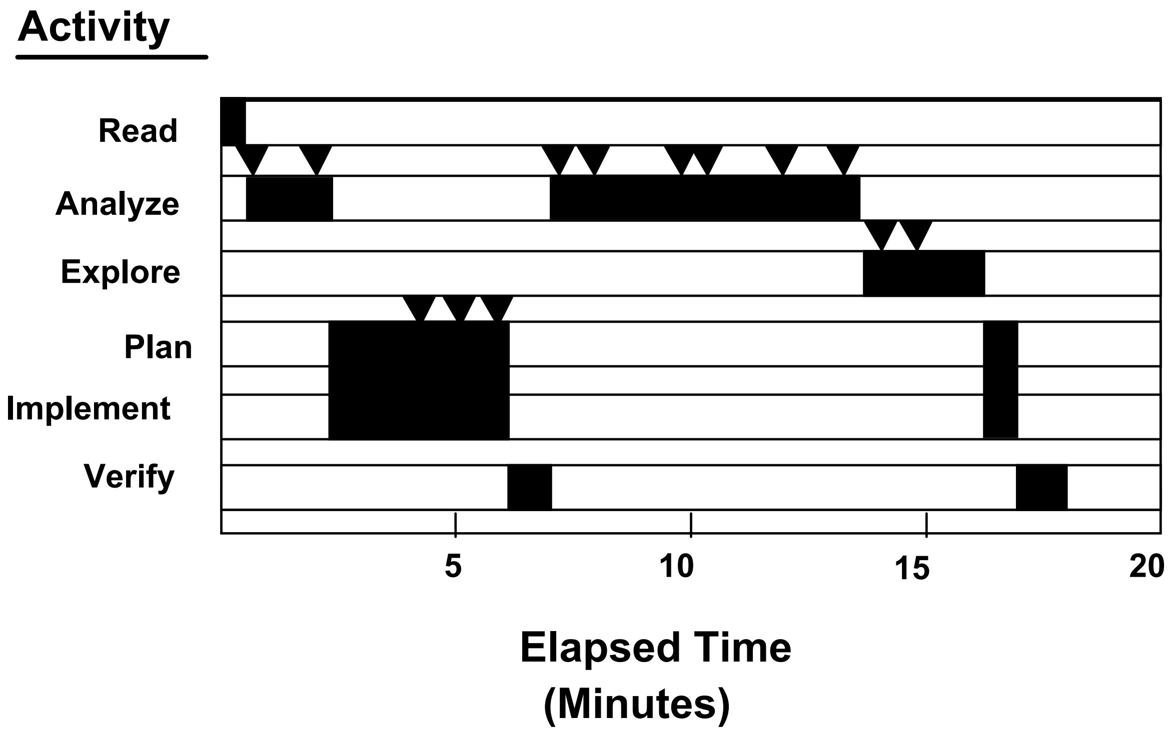
Appropriate attitudes and expectations (your "belief system")

METACOGNITION!



’)

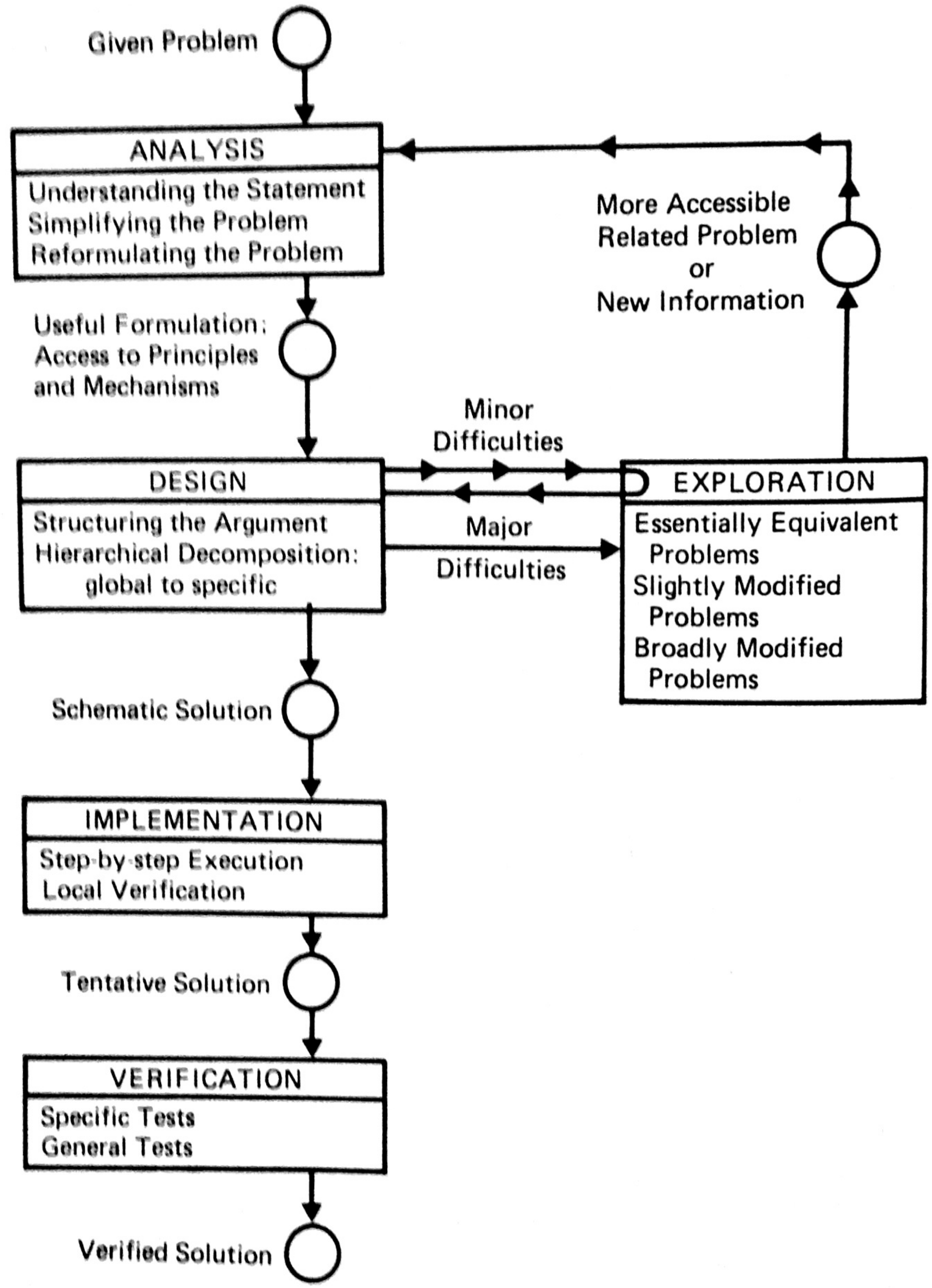
Time-line graph of a typical student attempt to solve a non-standard problem.



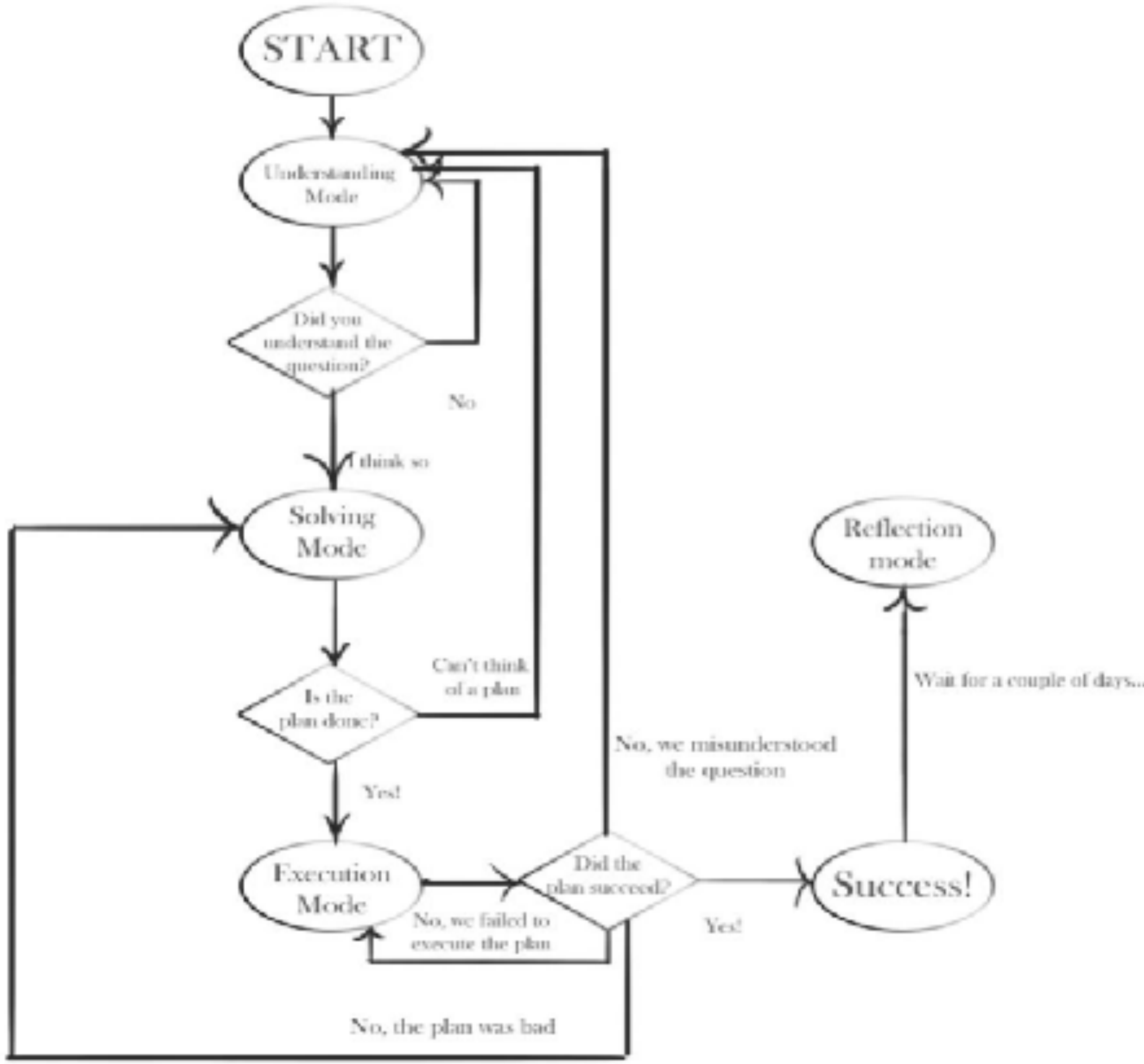
Time-line graph of a mathematician working a difficult problem

Schoenfeld control strategy

Makes the <u>exploration</u> more visible!



A student version! (from 2013)



Other student versions!

- Förstå problemet, vad som efterfrågas och hur svaret kan se ut
- Beskriv problemet med matematiska termer
- Välj modell och de variabler och restriktioner som behövs (planera hur man skall ta sig an problemet)
- Gå konstant framåt genom att lösa mindre delar av problemet, enklare eller specialfall
- Kontrollera om slutsatsen man nått är vad som efterfrågas och om det är rimligt
- Kan man nå någon djupare insikt utifrån vad man kommit fram till

---

Förstå!
Förenkla!
Gör om, gör rätt!

Initial steps (figure out the problem)

- Explain what the problem is.
- Understand by drawing figures.
- Attempt to set up the problem.
- Constants, variables, equations, and so on.

Explanation (examinate and determine the problem components and scope)

- Discuss the problem further. ○ Possibly narrowing the scope or broadening the possibilities.
- Explore possibilities. This can possibly lead to simplifying the problem.
- Make necessary assumptions, either to enable progress or to restrict the scope of the problem.
- (Create necessary placeholder figures or write full figure texts.)

Result (find a method toward a solution)

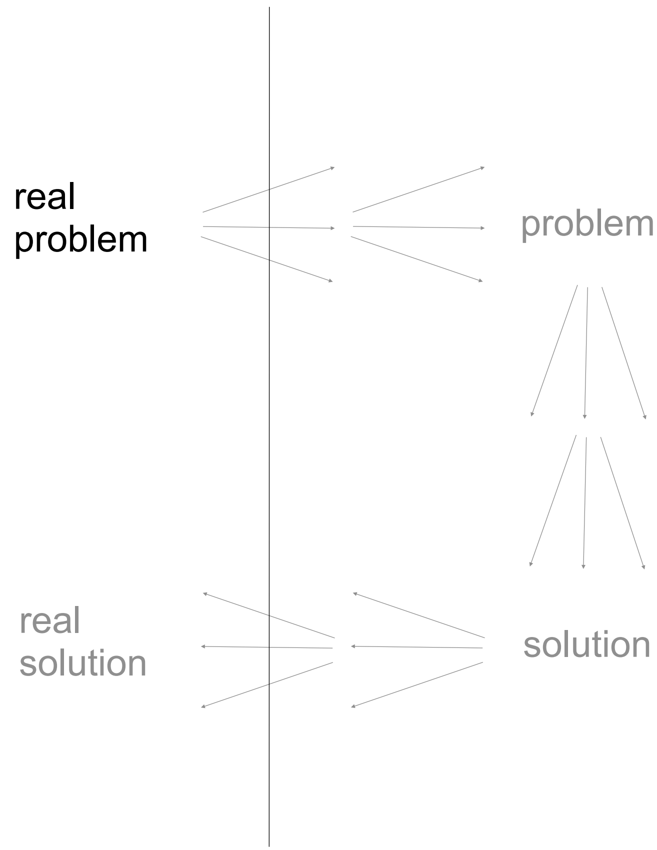
- Create figures.
- Use tools.
- Write explanations and do calculations (with tools).
- Argue for or prove the correctness of the procedure.
- (Create necessary placeholder figures or write full figure texts.)

Final steps

- Draw conclusions, answering the problem.
- Correct by rereading the texts.
- Simplify the reasoning, if possible.

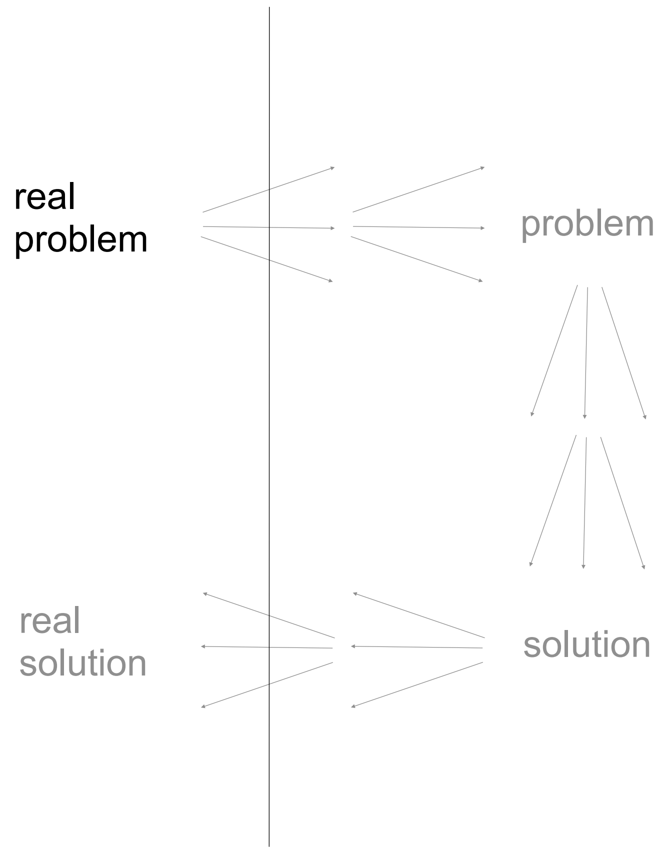
how the course works

Solving *ill-structured* real-world problems



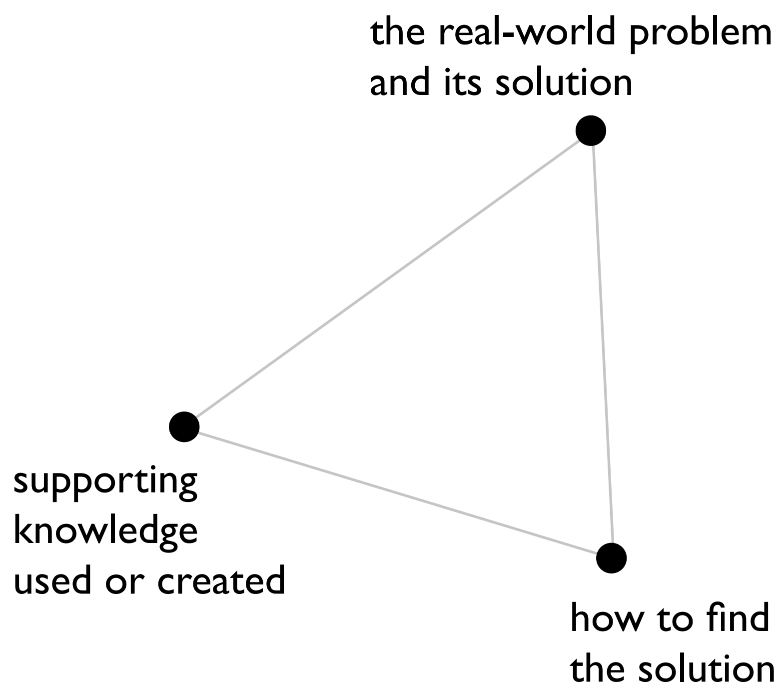
Training problems should contain realistic challenges

The problem should <u>not</u> be easy to understand!



The problem should be difficult to solve

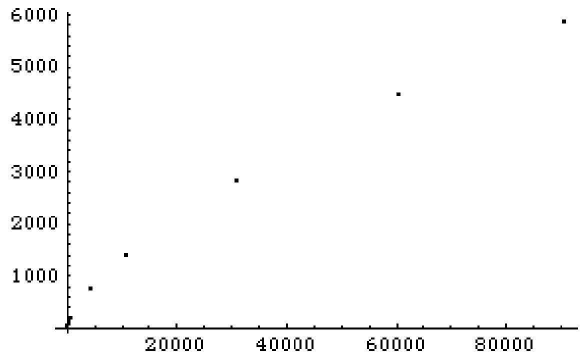
Every problem is a meaningful _case_ with several learning opportunities!



Many things can be learned from a single problem

idea of curve fitting

a story about Kepler



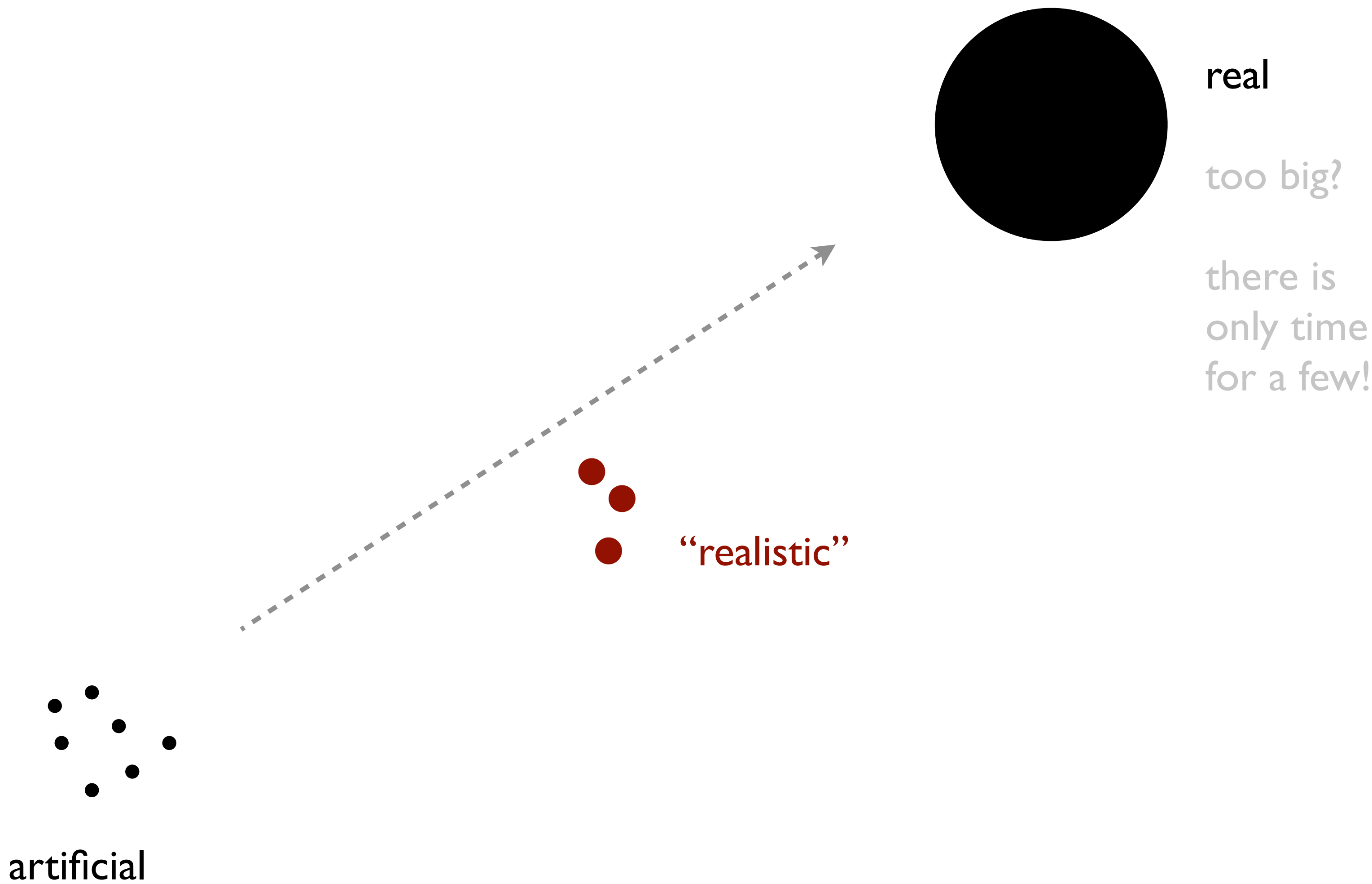
evaluate quality of solution

recall basic mathematical functions

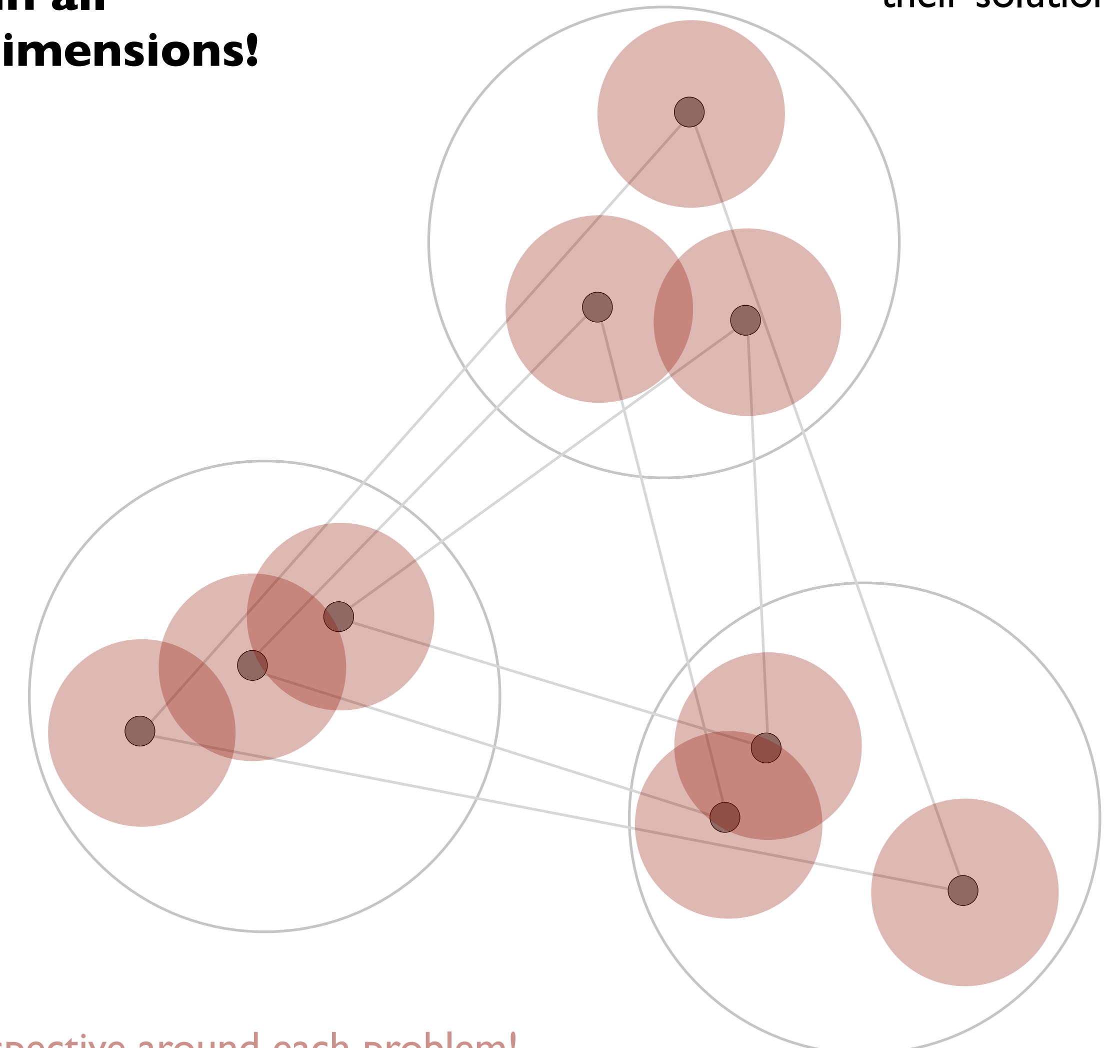
develop your own problem solving strategy

prepare for least squares method

Use many small "realistic" problems



Create a significant <u>variation</u> in all learning dimensions!



also create a perspective around each problem!

Fundamental idea

Learn to handle variation in unknown future problems by practising to handle variation with problems that we have now!

A cognitive apprenticeship learning environment

Authentic problems - learning is based on real practice

Make expert thinking visible - teacher shows good ways to think

Make student thinking visible - enables feedback and alignment with expert thinking

Cognitive apprenticeship can be applied in many areas when complex skills need to be taught, and to convey the experience of the teacher.

The course was originally designed intuitively based on experience, but we have found that cognitive apprenticeship is useful to explain how it works.

Necessary to focus on process -
not on correct solutions!

this is really important!

some comments at the end

Mathematical thinking

Mathematical reasoning

clearly define and draw conclusions in small steps

Problem solving

what to do when you don't know what to do

Modelling

linking the real world and the abstract world

Don’t be a follower!

Take control of your own thinking!

Trust your own judgement!

Focus on problems, not on solutions!

Create your own problems!

For you as a software engineer (IT, CS-programme,…)

Your ability to use mathematics and mathematical thinking as a problem solving tool is an important part of your capacity as engineers!

Many other ”IT-people” do not have that.

For you as a teacher of mathematics

Modelling and problem solving are at the heart of applying mathematical thinking in practice!

Thank you!