How to believe your models

Louise Pryor

Outline

- Setting the scene
- Specifying the model
- Implementing and testing the model
- Parameters and data
- Believing the results

What do you use models for?

- Reserving
- Pricing
- Capital requirements
- Reporting
- ...

How do you build them?

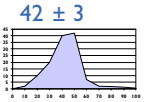
- Pencil/paper/calculator
- Spreadsheet(s)
- Specialist modelling package
- Purpose built system

- Combination

What do you expect from your models?

- The Answer(s)
- Range(s)
- Distribution(s)
- Qualitative relationships
- ...

42 ± 3



$x > y$

$x \propto \frac{1}{y}$

Who cares?

- You (professional pride)
- Your boss
- Board (Sox...)
- Shareholders
- FSA

- Marketing and sales folk
- ...

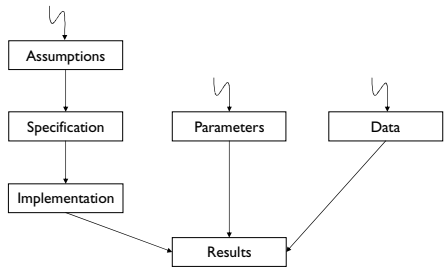
What could go wrong?

- You're using the wrong model
  - The specification is wrong
- The implementation is wrong
  - It doesn't do what you want it to do
- The wrong data or parameters are used
  - Garbage In, Garbage Out

Proportionality

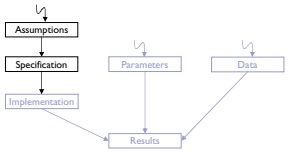
- How much do you care?
  - What is the model being used for?
- How complex is the model?
  - There is less to go wrong with simple models

The links in the chain



Outline

- Setting the scene
- Specifying the model
- Implementing and testing the model
- Parameters and data
- Believing the results



Specification

What the model is meant to do

- Is the model appropriate for what you will be using it for?
  - What limitations are there on the theory (if any) you are using?
  - What assumptions are you making?
- Is the specification complete
  - Every case should be covered
  - It should be specific enough to test against

Specification sign offs

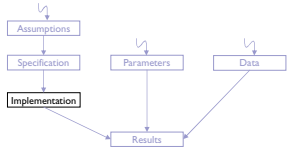
- Who should sign off on the specification?
  - Someone who has the authority
  - Someone who knows enough about the details
- What if a signed-off specification has obvious errors in it?
- Consider having two levels of specification
  - The general principles; signed off at a high level
  - Detailed enough to implement from
- Proportionality

Reviewing

- Reviewing should happen throughout the process
- It's necessary but not sufficient
  - You can't expect to pick everything up at the review stage
  - The earlier you find problems the easier it is to fix them
- Who should review?
  - More or less experienced in general/particular?

Outline

- Setting the scene
- Specifying the model
- Implementing and testing the model
- Parameters and data
- Believing the results



Implementation

Where things can really go wrong

- You are writing software
  - Spreadsheet
  - Specialist modelling package
- Writing software is hard
- Software error rates are high
  - Lawrence & Lee: average 7% of unique formulae, maximum 22%, 100% of spreadsheets had errors

Managing the risk

Good systems and controls (process)

- Thorough reviewing
  - Find errors earlier (easier to fix)
- Thorough testing
  - Need specification to test against
  - Keep testing records
  - Need version control
- Change control
  - Keep changes small and documented

Reviewing vs Testing

Reviewing	Testing
Inspect	Run the code
Manual or automatic (depends on software)	Manual or automatic
Automated: syntactic	Automated: semantic
Depth depends on the reviewer's understanding	Breadth depends on the execution paths chosen

When to test

- Unit testing individual components
- System testing as a whole
- Regression testing new against old
- Acceptance testing by user

- All are more difficult with spreadsheets

Change is a part of life

It's a continuously iterative process

- Implementation raises questions about the specification
- Testing raises questions about the implementation and specification
- When a model is used you see how it could be improved

Preventing problems

Be aware

- Don't be over confident
- Don't trust yourself or others
- "It's only a small change – it won't affect anything else"

Use version control

- Easy to back out of changes
- Know where you are

Having confidence in the implementation

Audit trail of

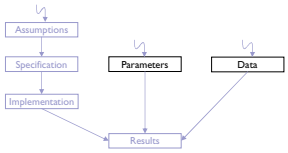
- Changes (with versions they appear in)
- Tests (with versions tested)

Documentation

- For future maintainers
- So you know what you think it's doing

Outline

- Setting the scene
- Specifying the model
- Implementing and testing the model
- Parameters and data
- Believing the results



Parameters

Where have the parameters come from and why?

- Have they been signed off, and if so by whom?
  - Are the principles right?
  - How about the details?
  - Was the sign off a full review?
- Are they the results of some calibration process?
  - Do you believe the calibration process?
  - Specification, implementation, parameters, data...

Data

Where has the data come from?

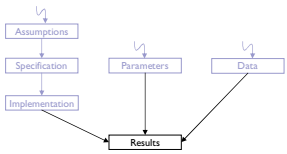
- Extracted from back office system?
  - Do you have confidence in the extraction process?
  - Specification, implementation, assumptions, data...
- Some other source?
  - Do you believe it is what it says it is?

Is there a sign off on data?

- Who signed off?
- Was it a full review?

Outline

- Setting the scene
- Specifying the model
- Implementing and testing the model
- Parameters and data
- Believing the results



Results

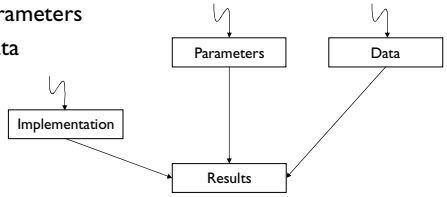
Do the results come from where you think they do?

- What version of the implementation was used?
  - What is its specification (including changes)?
  - What tests has it passed?
  - Has everything been reviewed or signed off? By whom?
- Which set of parameters was used?
  - Signed off by?
  - What changes?
- What data set was used?

Audit trail

Trace the results back to known

- Implementation
- Parameters
- Data



Believing your models

Know what you are doing

- Appropriate theory, correct model, reasonable assumptions
- Be aware of the risks

Do it properly

- Good process

Demonstrate that you've done it

- Good audit trails