This section briefly summarizes some of the considerations and steps that should be taken when validating that visualizations show what is intended.

Four levels of visualization design

Visualization design consist of four nested levels [?, 68]:

1. Domain situation, where it is considered whether or not a problem could be solved or even partially solved by visualization
2. Data/task abstraction, where the data that contain the (partial) solution is found, and abstracted from the domain.
3. The third level, where visualization idioms and interaction methods are chosen.
4. The algorithm level, where the algorithms behind the visualization idioms are constructed.

Splitting visualizations up in four parts like this will make things easier when analysing, and if something doesn't work, finding the root of the problem will be easier as well. There are two ways of going through this model, first being down-stream, where the designer try to solve a visualization problem that exist for a target group. In this case, there are usually already visualizations that will suit the problem. The other is up-stream, where the objective is usually to create some new visualization, starting at the algorithm, or idiom step, working up to test practical use of the visualization.

Visualization validation

When a visualization is made, it is time to validate it. There are several threats to a visualization's validity, neatly summarized in "Visualization analysis and Design" [?, 75]

- *Wrong problem*: You misunderstood their needs.
- *Wrong abstraction*: You're showing them the wrong thing.
- *Wrong idiom*: The way you show them doesn't work.
- *Wrong algorithm*: Your code is too slow.

Wrong problem: this is most likely to happen with a down-stream design, where there is some sort of miscommunication between the designer and the people who need the visualization. This will almost guaranteed cause problems in the data abstraction step as well, so if something seems to be wrong there, it would be a good idea to check the problem step as well.

To solve this, one should first contact and interview the target group if possible, however if the target group is too broad, or very general they might not know what they need themselves, in which case validating everything else first, and then looking at adoption rates, checking if people actually start using the visualization.

Wrong abstraction: This occurs if the data selected simply does not have what is needed to solve the given problem. A common way to validate the correctness of the abstraction, is getting target users to use the visualization system, and see if the user base find it useful.

Wrong idiom: This happens if the visualization technique isn't suitable for the given dataset or simply doesn't lower the cognitive load for the user compared to looking at the raw data. If something is wrong at this level, using a different visualization technique should be considered, if the designer is working down-stream. Otherwise double-checking the visual encoding/interaction is recommended, followed by getting people to test the visualization for different purposes, and adjusting based on their feedback.

Wrong algorithm: The major problem here is, if visualization would take seconds, minutes or even more, where a user would expect a result in less than a second.

A good first step, is to analyze the complexity of the algorithm, and see if any apparent improvements can be made. After that, it is worth taking a look at the input data to check if it can be pre-processed, to improve the visualization loading times and responsiveness even further. Then there are other cases, where the problem is the memory usage. In these cases it should be considered to reduce memory usage where possible, even if it will reduce performance.