A Details of the User Study

We proposed an online web-based system for the user study. Our study was approved by our institution's Institutional Review Board (*IRB*). The pipeline of the user study is shown in Figure 1. Before the user study began, all participants will be redirected to the online study system designed by us. In the very beginning, we recorded the participant's Prolific ID and provided them with the completion code at the very last of the study to trace every participant and improve the quality of the answers. The participants are required to complete all mini-tests to improve the experiment quality. The 72 tasks and the questionnaire are the subsequent procedures. Note that all participants are divided into two groups for a strict counter-balance order to mitigate the learning effects, as shown in the two paths (i.e., Task Sequence 1 and Task Sequence 2) in Figure 1. All choices and questions are required for the completion. Each participant will get a completion code after they answer all open questions in the questionnaire.

Note that to avoid the participants comparing the line slopes via the arrows' vertical positions, we adjust the arrow directions the same as the target line segments to mitigate the effects. To remind each participant to perform the study carefully, we terminate the test if the participant failed one of the mini-tests at the beginning of the study (Figure 1(B)). In addition, we insert a warning page between the mini-test and the formal 36 studies to inform the participants the study and timer will begin and ask them to finish each task as quickly as possible (Figure 1(A)).

B Usage of Our Open-source Package for Intercept Graph

To support the easy implementation of *Intercept Graph*, we develop a JavaScript function and make it open-public via the URL: https://www.npmjs.com/package/interceptgraph. The open-sourced package supports the accurate and smooth comparison of state changes for the general public.

We provide a useful API for the quick implementation of *Intercept Graph*. Specifically, we define two parameters to build the chart:

- ***svg_id:*** the identification of the SVG container to draw *Intercept Graph*. The parameter only accepts the *id* of the DOM element. Note that the width and height of the SVG should be specified within the tag using *width* and *height* parameters.

- ***src:*** the data source to be rendered. The parameter only accept *CSV* data file as input. Using the path name to the *CSV* file for the parameter. Note that the *CSV* data file should be formatted as follows: 1) no headers for the dataset; 2) the order of the data field: *name*, *data series 1*, *data series 2*; 3) use "," for separating the attributes.

An example of the usage of the package is shown in Figure 2.

```
<script>
    interceptgraph_build('#interceptGraph_SVGContainer', path_to_dataFile)
</script>

return(
    <div>
        <svg id="interceptGraph_SVGContainer" width='600px' height='600px'/>
    </div>
)
```

Fig. 2: An example is to call the rendering function to draw an Intercept Graph conveniently and quickly. The user needs to mount the node of the SVG element first, and then render the chart into the SVG element. The size of the SVG element is required to be specified in advance instead of including it in a style sheet.

C Generated Datasets for Metric Evaluation

In Section 6, we use randomly-generated datasets to evaluate the metrics, i.e., line crossings and intensity ratio. For the dataset generation, we define three scale levels (i.e., small scale, medium scale, and large scale) and 81 normal distributions to perform the metric evaluation. In this section, we showcase some of the charts (i.e., our approach: *Intercept Graph*, baseline approach: slope graph) driven by the datasets we generated in advance, as shown in Figure 3.

The procedure for dataset generation is as follows: first, we defined three scale levels (i.e., the small-scale, the medium-scale, and the large-scale) of the number of data items. For each scale level, an increment of 5 data items was set to cover a wider range. Assume that the axis ranges of the two data series were from 0 to 100. For each *randomly-generated* dataset, we tried to generate all possible distributions for the given number of data. We applied the Gaussian Distribution to the data points generation and set three types of means and three types of standard deviations to the data series (i.e., d_1 and d_2), respectively. We, therefore, generated 1215 randomly-generated datasets in total: 3 scale levels with 5 scale numbers per level $\times$ 9 Gaussian Distributions of data series d_1 $\times$ 9 Gaussian Distributions of data series d_2.

We evaluate *Intercept Graph* in comparison with the slope graph that *Intercept Graph* can effectively mitigate the line crossings compared to the slope graph, especially for the datasets with a large number of data items (e.g., over a medium scale of 200). Meanwhile, *Intercept Graph* can magnify the ratio of the length of two state changes, making it more intuitive to tell apart the relationship between two state changes.

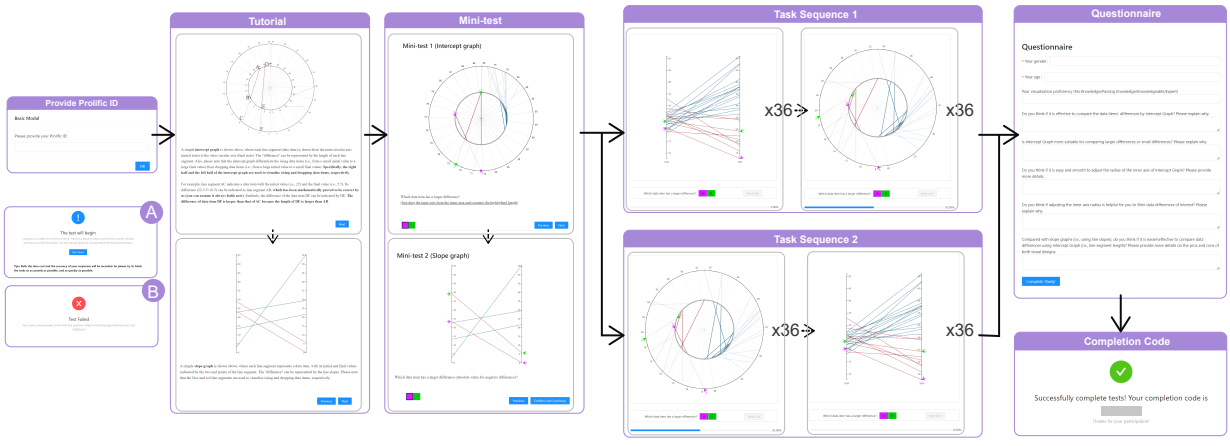

Fig. 1: The pipeline of the system for user study, consisting of the screenshots of the system interfaces. Each participant is required to complete all tasks and the questionnaire to receive the completion code at the very last of the study. Each participant was compensated with US $3.28.

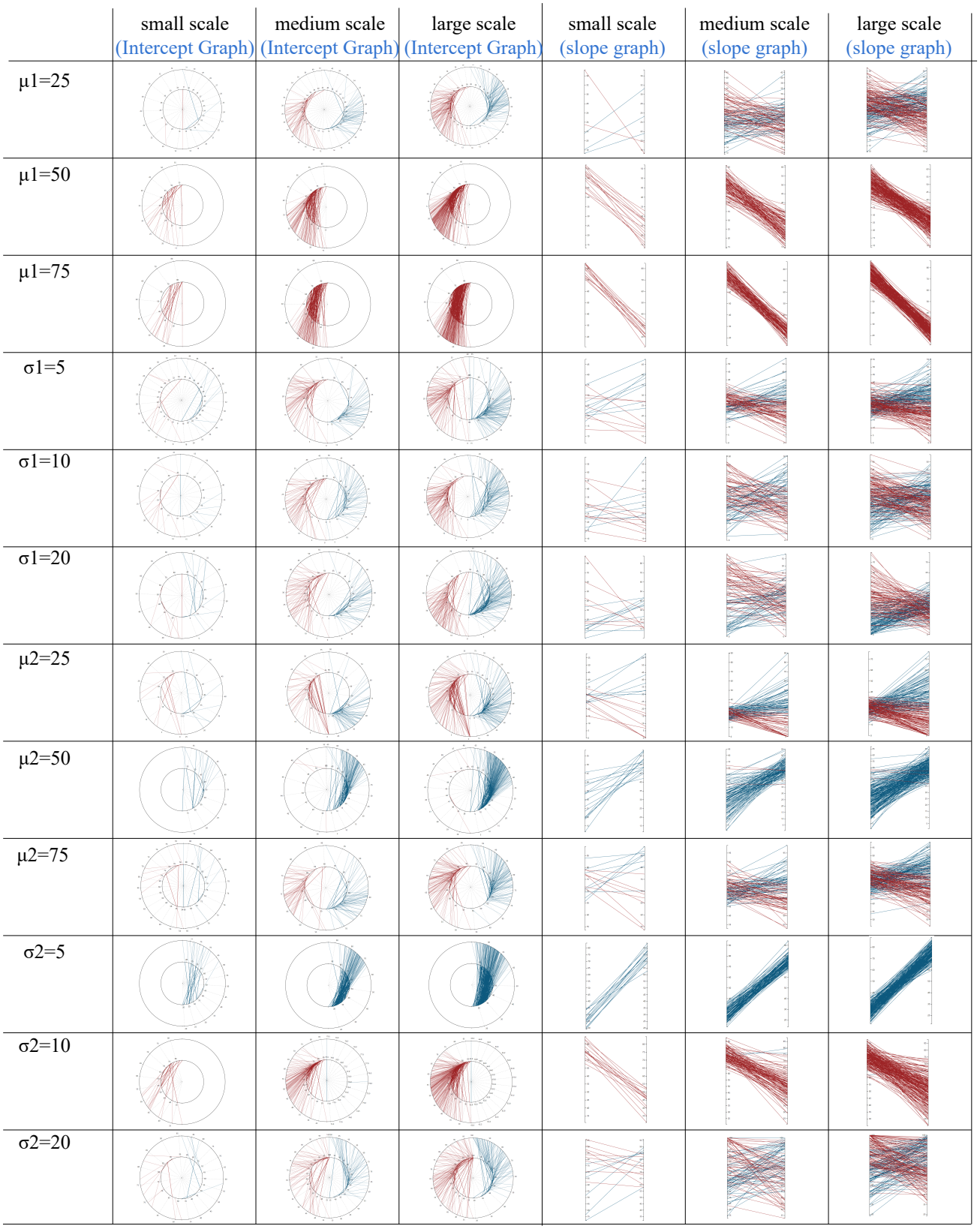

Fig. 3: The comparison of *Intercept Graph* and slope graph. We use the same data to evaluate *Intercept Graph* and the slope graph in each experiment. The Gaussian Distribution shown in the chart is a part of the whole set (i.e., 12 out of 81). μ_1, σ_1 are for Gaussian Distribution of data series 1, while μ_2, σ_2 are for Gaussian Distribution of data series 2. The other distribution parameters in each row are set as the first value in each parameter set, as shown in Table 1.