Visual Embellishments in Bar Charts

Claire Pang

December 16, 2018

1 Paper implemented

Drew Skau, Lane Harrison, Robert Kosara, "An evaluation of the impact of visual embellishments in bar charts, Computer Graphics Forum (Proceedings EuroVis), vol. 34, no. 3, pp. 221–230, 2015.

2 Abstract

An implementation of the above paper, which evaluates how embellishments affect data communication. I implement the same user study by deploying on Amazon Mechanical Turk, perform data cleaning and statistical analysis, and compare my results to those found in the original paper.

3 Goal of paper

The main goal of this paper is to evaluate how common embellishments (often referred to as "chart junk") affect the communication of the data. These extraneous embellishments are typically used when creating infographics and software programs allow designers to easily add these embellishments. To understand how embellished charts are perceived, user studies testing 7 common embellishments are conducted.

4 User Study Replication

4.1 Bug fixes and scope of study

The code for this user study was available on the authors' github [https://github.com/dwskau/bar-chart-embellishment], however it had bugs when it came to displaying the D3 visualizations. The bug was that the fill for some of the visualizations was "#0000" rather than "#000000", causing some of the visualizations to not appear. The user study was based off experimentr, so I just had to figure out how to run the study on my computer. The user study consists of relative and absolute questions:

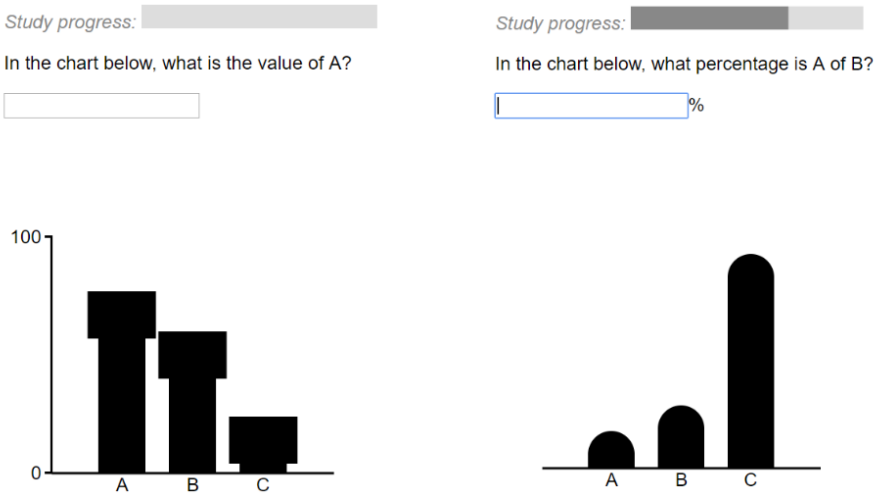


Figure 1: Example of questions asked in the study. **Left:** example of an absolute question. **Right:** example of a relative question.

- Relative: Comparing the value of two bars, where the answer should be the % of one bar in comparison to the other (Figure 1 - left).
- Absolute: Comparing the absolute value of a bar, where the answer should be a number 1-100 in comparison to the "100" tick mark (Figure 1 - right).

In total, abstract versions of 7 common embellishments (Figure 2) found in infographics were tested (including the baseline, a basic bar chart). There were 5 absolute questions/embellishment, and 8 relative questions/embellishment. This adds up to a total of 91 questions, all of which I included in my user study.

4.2 Experimentr, Redis, and deploying on vgl-userstudy

After that getting the study to run on my computer, I spent most of my time testing the data collection with redis and understanding the data that was pulled. For each survey submission, the data was saved 3 times in redis, where all the data was the same except with a different postID and user feedback (comments left at the end). I could not figure out why that occurred, but figured it was nothing to worry about as long as I got the data. The next step was to deploy it on vgl-userstudy (one of USF CS's servers). I used tool *tmux* so that the study would stay running even when I closed the ssh window.

4.3 Deploying on Mechanical Turk

Once I tested that the data collection worked on the lab machines too, I had USF Professor Alark Joshi help me put my survey on Mechanical Turk. The

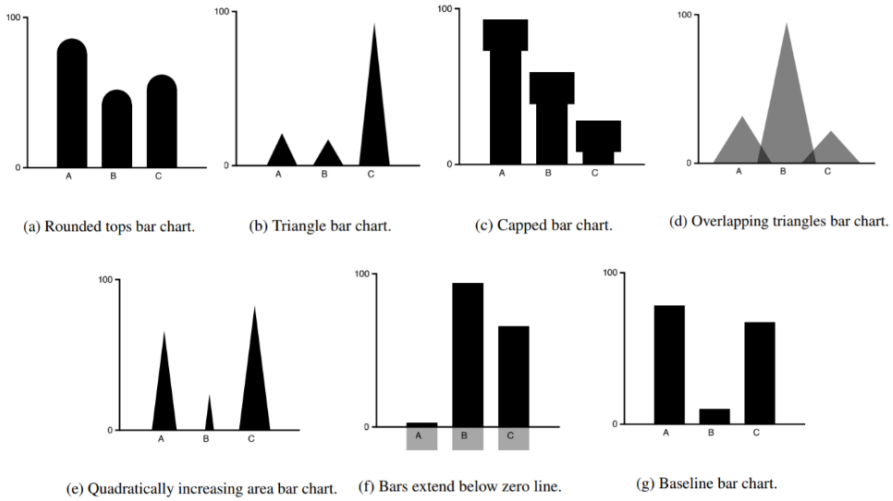


Figure 2: The 7 different embellishments tested in the study.

paper had 100 participants, paying them each $2.00. We decided to also use 100 participants, but due to the budget, we only paid $1.00 per participant. The users were given a link to my user study hosted on vgl-userstudy, and then they were expected to submit the unique postID (generated by experimentr) on Mechanical Turk, which was given upon survey completion.

4.4 Contacting the authors

Throughout the entire replication process, I did not contact the authors since I was able to successfully run the survey and understood the data that was being collected. The statistical analyses that were conducted were described the paper in enough detail, with the statistics they derived included in a table.

4.5 Data Cleaning

After collecting all the data from Mechanical Turk, I had to do data cleaning. The data that was collected from redis was in a terrible format. First, it was stored as in JSON format, so I used an online tool (any of the results by searching JSON to CSV on Google will do) to convert the JSON to CSV.
Each survey submission/postID (3 per submission as mentioned in section 4.2) was one row, with multiple columns. There were 1618 columns, often with extraneous information that I didn't need for analysis. Each question asked in the survey had 10 associated columns, labeled like "ans_trial.10", "log_error.10", etc. The "10" refers to the question number, so for each question, these columns were named accordingly.
The cleaned data that the authors posted on their Github was structured where

each row referred to a question, therefore I wrote a R script to convert my data to mimick that structure, keeping only the data I needed for analysis (and only kept one copy of each user submission, determined by the postID submitted on Mechnical Turk). Therefore, my CSV file went from 300 x 1618 to 9100 x 14.

4.6 Approving/rejecting participants

After I converted the data to a more readable format, I was able to look at the data and check the quality. 7 out of the 100 participants were rejected. 1 user completed only half the survey, meaning they likely used the "enter" button to skip over a majority of the questions to get the postID. The other participants were rejected for the following reasons:

- Giving repetitive answers (e.g., only putting "11" or "55") for every question
- Sequential answers (such as 10, 20, 30, ...)
- Wildly inaccurate answers that were not even plausible. The maximum value in both the relative and absolute case was 100, but they had multiple answers over 100.
- Using the formula $(given - correct)/correct$, if over 75% of their answers were more than 50% incorrect

For the last condition, I tried to minimize my rejections from this by making the error threshold really high because it was sensitive to magnitude. For example, if the correct answer was 5, and they put 9, the error answer percent difference would be .8, or 80%. For users satisfying this condition, I manually looked over all their answers to see if their answers were somewhat plausible before rejecting. I thought that approving/rejecting the participants was the hardest part of the replication process because I could not tell if users actually tried during the survey or not. Although the average error rate was high overall, I only rejected participants I had definitive evidence against.

4.7 Statistical Analysis

The code on Github did not include an analysis script, which required me to write my own in R. However, the tests that they performed and their results were included in the paper. Log error was used in the p-value calculations, and was automatically generated within the included user study code. The paper performed 6 Mann-Whitney Wilcoxon tests, comparing all the embellishments to the baseline.
The reason why Mann-Whitney Wilcoxon tests were done was because the data is not normally distributed, which I also checked for in my analysis. The paper also mentions that they performed a Bonferroni correction to address the problem of multiple comparisons. I used a method (*p.adjust*) in R to apply the Bonferroni correction. I have never used the Bonferroni correction

before and the corrected p-values seem extremely adjusted (unlike the ones from the paper), therefore I have provided both the Wilcoxon test p-value and the corrected p-values in my results (Figure 3 and 4). The full code and results are appended at the end of this report.

4.8 My results

4.8.1 Absolute judgments

The major findings from the paper were:

- Quadratic bars performed significantly worse than baseline
- Extended and capped performed similarly to the baseline

My results:

- *Same:* Quadratic also performed significantly worse than baseline with an unadjusted p-value of .007.
- *Same:* Extended and capped also performed similarly to the baseline, with unadjusted p-values greater than 0.87.
- *Different:* My mean error and standard deviation in all cases were higher.

Embellishment	Mean	SD	p-value
baseline	1.41	1.85	
capped	1.41	1.68	0.50
overlapping	1.64	1.76	0.048
quadratic	1.70	1.78	0.007*
rounded	1.67	1.77	0.009
triangle	1.70	1.68	0.012
extended	1.45	1.68	0.562

embellishment	mean	sd	p_value	p_val bonf
baseline	1.741984	1.878694	NA	NA
capped	1.819811	1.752839	0.8720448	1.0000000
overlapping	2.063182	1.873572	0.0114749	0.0688495
quadratic	2.022934	1.871716	0.0070324	0.0421945
rounded	1.844422	2.043567	0.2956269	1.0000000
triangle	1.962231	1.811785	0.2035150	1.0000000
zero	1.804515	1.965258	0.9146440	1.0000000

Figure 3: P-values in the **absolute** case. All calculations done using log error. **Left:** Paper results, significant values * at $\alpha = 0.0083$. **Right:** My results. "p_value" indicates the p-value from the Wilcoxon test, and "p_val bonf" indicates the p-value after applying the Bonferroni correction.

4.8.2 Relative judgments

The major findings from the paper were:

- In all except extended, baseline performed significantly better. Extended was the only one that performed similarly to baseline.
- Quadratic had the highest error rate overall.

My results:

- *Different:* My mean and standard deviation in all cases were higher.
- *Same:* Quadratic had the highest error rate overall.
- *Different:* The only case that performed significantly worse than the baseline was quadratic.
- *Same:* Extended also did perform similarly to baseline, with mean error being 2.59 (baseline) and 2.58 (extended).

Embellishment	Mean	SD	p-value
baseline	1.43	1.85	
capped	1.70	1.68	0.0013^*
overlapping	1.82	1.76	$< 0.001^*$
quadratic	2.33	1.78	$< 0.001^*$
rounded	1.86	1.77	$< 0.001^*$
triangle	1.85	1.68	$< 0.001^*$
extended	1.59	1.66	0.097

embellishment	mean	sd	p_value	p_val bonf
baseline	2.597021	2.262353	NA	NA
capped	2.706159	2.137560	0.4061049	1.0000000
overlapping	2.779224	2.141845	0.1671329	1.0000000
quadratic	3.043939	1.955588	0.0000414	0.0002483
rounded	2.826405	2.005906	0.0554253	0.3325515
triangle	2.864869	2.041002	0.0184212	0.1105271
zero	2.589071	2.117536	0.4962398	1.0000000

Figure 4: P-values in the **relative** case. All calculations done using log error. **Left:** Paper results, significant values * at $\alpha = 0.0083$. **Right:** My results. "p_value" indicates the p-value from the Wilcoxon test, and "p_val bonf" indicates the p-value after applying the Bonferroni correction.

4.8.3 Overall results

Notable observations from the paper is that the quadratic chart performed the worst overall, even compared to similar techniques such as the triangle or overlapping. This is also confirmed with my results. Additionally, all charts performed worse than the baseline (with the exception of extended). This was also confirmed with my results, where extended performed the same as the baseline or only slightly worse.

The paper did not talk about demographics, therefore I will not compare my demographics.

5 Discussion

The reason for the differing results is likely due to the Mechanical Turk setup. The actual study paid each participant $2.00, whereas in my study, participants were only paid $1.00. Higher paid participants may be more motivated to spend more time and actually take the survey. Additionally, none

of the participants in my survey were Master Turkers, which means it's unlikely they were dedicated towards giving correct answers.
The rejections in the actual survey may also have been more strict. The maximum error for the 94 participants in their survey was 45%. I only excluded participants who had done something definitively wrong, which means that my restrictions were not as strict. This means that my pool of participants could include those who sped through the survey or did not actually try to give the right answers.
In terms of completion time, the average for the paper was 19 minutes 11 seconds. The completion time, according to the Mechanical Turk submissions (which may not be accurate, as it may not account for the actual time spent taking the survey), was 25 minutes and 16 seconds. According to the histogram generated with my R script, most of the users fell below 20 minutes, and there were even submissions for 10 minutes or less. Although it is likely that the ones who completed in less than 15 minutes sped through the survey, I did not find definitive enough evidence to exclude them from the study.

6 Future Work

Although I completed the entire scope of the study as in the paper, there is still more research that can be done on this topic. This type of study can be continued, by testing other types of embellishments. Other perceptual studies can also be done, as this focuses primarily on how people perceive very basic representations of embellishments. Embellishments in addition to the very simplistic versions, such as in the real infographics examples given in the paper, may affect perception differently. I would also improve the interface of the survey so that there would be error checking, such as requiring users to submit an answer for each question.

Visual Embellishments in Bar Charts

Claire Pang

Setup

Functions to generate a data frame with mean, standard deviation, and p-values for each embellishment type.

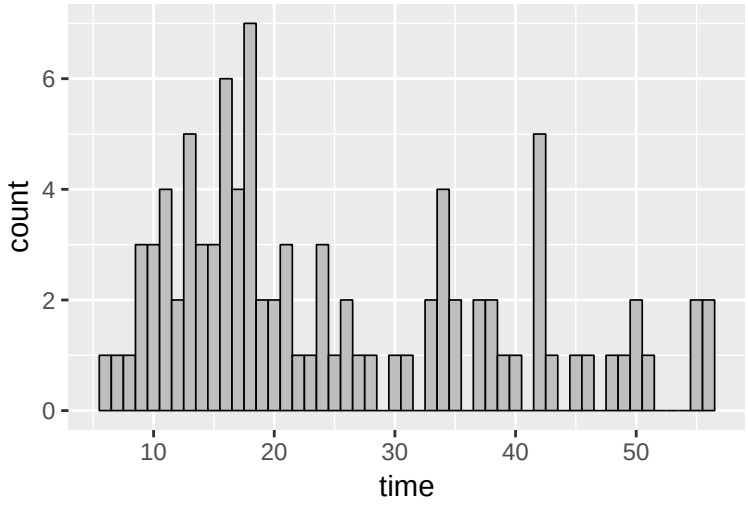
```
setwd("C:/Users/ciz4c/Desktop/bar/mturk")
mtdata <- read.csv("processed-edit.csv", header = T, stringsAsFactors = F)
userdata <- read.csv("CP-Batch_3451164_batch_results.csv", header = T,
    stringsAsFactors = F)
library(ggpubr)
library(knitr)

df_abs <- data.frame(embellishment = character(), mean = numeric(), sd = numeric(),
    p_value = numeric())

addmetrics_abs <- function(embellishment, data) {
    if (embellishment == "baseline") {
        df_abs <<- rbind(df_abs, data.frame(embellishment = embellishment,
            mean = mean(data), sd = sd(data), p_value = NA))
    } else {
        df_abs <<- rbind(df_abs, data.frame(embellishment = embellishment,
            mean = mean(data), sd = sd(data), p_value = wilcox.test(baseline_abs,
                data)$p.value))
    }
}
df_rel <- data.frame(embellishment = character(), mean = numeric(), sd = numeric(),
    p_value = numeric())

addmetrics_rel <- function(embellishment, data) {
    if (embellishment == "baseline") {
        df_rel <<- rbind(df_rel, data.frame(embellishment = embellishment,
            mean = mean(data), sd = sd(data), p_value = NA))
    } else {
        df_rel <<- rbind(df_rel, data.frame(embellishment = embellishment,
            mean = mean(data), sd = sd(data), p_value = wilcox.test(baseline_rel,
                data)$p.value))
    }
}
```

Completion time

```
# Get approved users
approved <- which(userdata$Approve == "x")
time <- userdata$WorkTimeInSeconds[approved]
time <- time/60
paste0("The average completion time is ", mean(time))
```

```
## [1] "The average completion time is 25.273476702509"
```

```
ggplot() + aes(time) + geom_histogram(binwidth = 1, size = 0.3, colour = "black",
    fill = "gray")
```

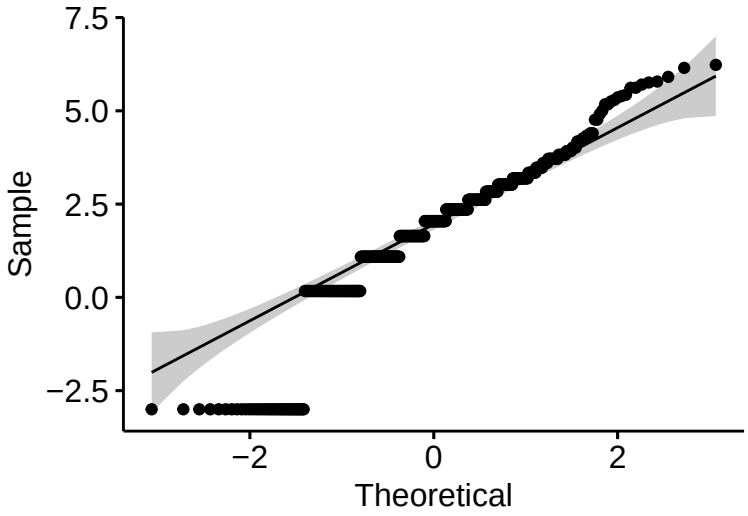


Checking for normality

```
# get data relating to baseline
baseline_idx <- which(mtdata$chart_embellishment == "baseline" & mtdata$question_type ==
    "absolute")
baseline_abs <- mtdata$log_error[baseline_idx]
# A QQ Plot to do a basic check for normality
```

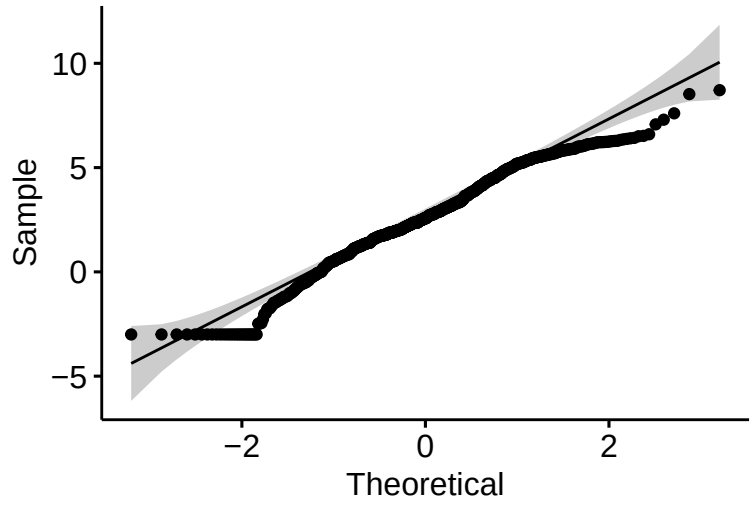
Absolute

```
ggqqplot(baseline_abs)
```



Relative

```
baseline_idx <- which(mtdata$chart_embellishment == "baseline" & mtdata$question_type ==
    "relative")
baseline_rel <- mtdata$log_error[baseline_idx]
ggqqplot(baseline_rel)
```



Based on the two QQ plots above, most of the points do not lie along the line. Therefore, we can assume that this data is NOT normally distributed. Thus, we must use the Mann Whitney-Wilcoxon tests to generate p-values for samples that are not normally distributed.

In each embellishment case, we will compare against the "baseline" results.

```
addmetrics_abs("baseline", baseline_abs)
addmetrics_rel("baseline", baseline_rel)

# get data relating to capped
capped_idx <- which(mtdata$chart_embellishment == "capped" & mtdata$question_type ==
    "absolute")
capped_abs <- mtdata$log_error[capped_idx]
addmetrics_abs("capped", capped_abs)
capped_idx <- which(mtdata$chart_embellishment == "capped" & mtdata$question_type ==
    "relative")
capped_rel <- mtdata$log_error[capped_idx]
addmetrics_rel("capped", capped_rel)

# get data relating to overlapping
overlapping_idx <- which(mtdata$chart_embellishment == "overlapping" &
    mtdata$question_type == "absolute")
overlapping_abs <- mtdata$log_error[overlapping_idx]
addmetrics_abs("overlapping", overlapping_abs)
overlapping_idx <- which(mtdata$chart_embellishment == "overlapping" &
    mtdata$question_type == "relative")
overlapping_rel <- mtdata$log_error[overlapping_idx]
addmetrics_rel("overlapping", overlapping_rel)

# get data relating to quadratic
quadratic_idx <- which(mtdata$chart_embellishment == "quadratic" & mtdata$question_type ==
    "absolute")
```

```
quadratic_abs <- mtdata$log_error[quadratic_idx]
addmetrics_abs("quadratic", quadratic_abs)
quadratic_idx <- which(mtdata$chart_embellishment == "quadratic" & mtdata$question_type ==
    "relative")
quadratic_rel <- mtdata$log_error[quadratic_idx]
addmetrics_rel("quadratic", quadratic_rel)

# get data relating to rounded
rounded_idx <- which(mtdata$chart_embellishment == "rounded" & mtdata$question_type ==
    "absolute")
rounded_abs <- mtdata$log_error[rounded_idx]
addmetrics_abs("rounded", rounded_abs)
rounded_idx <- which(mtdata$chart_embellishment == "rounded" & mtdata$question_type ==
    "relative")
rounded_rel <- mtdata$log_error[rounded_idx]
addmetrics_rel("rounded", rounded_rel)

# get data relating to triangle
triangle_idx <- which(mtdata$chart_embellishment == "triangle" & mtdata$question_type ==
    "absolute")
triangle_abs <- mtdata$log_error[triangle_idx]
addmetrics_abs("triangle", triangle_abs)
triangle_idx <- which(mtdata$chart_embellishment == "triangle" & mtdata$question_type ==
    "relative")
triangle_rel <- mtdata$log_error[triangle_idx]
addmetrics_rel("triangle", triangle_rel)

# get data relating to zero
zero_idx <- which(mtdata$chart_embellishment == "zero" & mtdata$question_type ==
    "absolute")
zero_abs <- mtdata$log_error[zero_idx]
addmetrics_abs("zero", zero_abs)
zero_idx <- which(mtdata$chart_embellishment == "zero" & mtdata$question_type ==
    "relative")
zero_rel <- mtdata$log_error[zero_idx]
addmetrics_rel("zero", zero_rel)
```

Bonferroni correction

```
df_abs <- cbind(df_abs, p.adjust(df_abs$p_value, method = "bonf"))
colnames(df_abs) <- c("embellishment", "mean", "sd", "p_value", "p_val bonf")
df_rel <- cbind(df_rel, p.adjust(df_rel$p_value, method = "bonf"))
colnames(df_rel) <- c("embellishment", "mean", "sd", "p_value", "p_val bonf")
```

Results

Absolute

```
kable(df_abs)
```

embellishment	mean	sd	p_value	p_val bonf
baseline	1.741984	1.878694	NA	NA
capped	1.819811	1.752839	0.8720448	1.0000000
overlapping	2.063182	1.873572	0.0114749	0.0688495
quadratic	2.022934	1.871716	0.0070324	0.0421945
rounded	1.844422	2.043567	0.2956269	1.0000000
triangle	1.962231	1.811785	0.2035150	1.0000000
zero	1.804515	1.965258	0.9146440	1.0000000

Relative

```
kable(df_rel)
```

embellishment	mean	sd	p_value	p_val bonf
baseline	2.597021	2.262353	NA	NA
capped	2.706159	2.137560	0.4061049	1.0000000
overlapping	2.779224	2.141845	0.1671329	1.0000000
quadratic	3.043939	1.955588	0.0000414	0.0002483
rounded	2.826405	2.005906	0.0554253	0.3325515
triangle	2.864869	2.041002	0.0184212	0.1105271
zero	2.589071	2.117536	0.4962398	1.0000000