Music Generator

By

Terry Wang, Rima Mittal, Joshua Goldberg

Supervisor: Yuri Balasanov

A Capstone Project

Submitted to the University of Chicago in partial fulfillment
of the requirements for the degree of

Master of Science in Analytics

Graham School of Continuing Liberal and Professional Studies

March, 2020

The Capstone Project committee for Terry Wang, Rima Mittal, Joshua Goldberg
Certifies that this is the approved version of the following capstone project report:

Music Generator

Approved by Supervising Committee:

____

Yuri Balasanov

Abstract

Maximum 40 to 50 words. An abstract is a concise description of your project. It should include very brief description of the problem, purpose, method, key results, and conclusions. Note you may want to write the abstract after writing the report.

Keywords: Include 6 to 10 keywords on the same page as the abstract. Select keywords that would help a researcher retrieve your report.

NOTE: Do not use "#" or "##" symbols to start new sections in the abstract section, as one typically would in other r markdown documents. Doing so will result in generating a table of contents entry *prior* to the Introduction, which is not desirable.

Executive Summary

The executive summary is a maximum one page, double spaced summary of your report aimed at informing someone, who does not read the entire report, about your project. The executive summary is an extended version of the abstract with more space allocated to the key findings of the project and the conclusions and recommendations. You may want to write this section after writing the report.

Second Paragraph.

Third Paragraph.

NOTE: Like the abstract, do not use “#” or “##” symbols to start new sections in the executive summary section. Doing so will result in generating a table table of contents entry *prior* to the Introduction, which is not desirable.

Table of Contents

List of Figures

List of Tables

Introduction

We built a deep learning model to help individuals generate interesting music ideas. The model is a GAN-based generator. The data used to train the generator are 130,000 midi files across many different genres of music.

Problem Statement

In 1950s, experimental music composers wrote music using randomized statistical modelling. In 1990s, David Bowie built the Verbasizer, which implemented a random re-ordering of group of words and sentences to produce potentially significant lyrical combinations. Nowadays, there is an entire industry built around AI generated music including Flow Machines, IBM Watson Beat and Google's NSynth.

Our purpose is to build an effective platform to help artists in their creative process and assist them in music composition. Note that artists do not have to be professionally designated. Any individual interested in creating music should find the platform useful and empowering. The first concrete step we need to take to achieve that goal is to build a music generation model that is able to generate music that is similar to what a human would compose in the style of music the model is trained on.

Research Purpose

The purpose of this project is to develop a deep learning based music generator (instrumental music) which has a good understanding of the language of music and is able to generate human-like outputs, ultimately helping musicians and non-musicians alike to develop and refine musical ideas, and facilitate them in the music composition process. We divided the project into two parts, Generation and Validation.

By doing this iteratively, the weights and parameters of the model will hopefully converge to a point at which the music generated is satisfactory to the user, who can use the output of the model, or save the weights for future use. In a sense, we are democratizing deep learning to non-technical users wishing to generate new music ideas in a novel way. Moreover, we expect the algorithmic process to able to explore more music combinations than any individual, and thus provide a more holistic and creative approach to music creation.

Variables and Scope

Our data consists of 130,000 midi files curated by a Reddit user, with a size of 3.65GB uncompressed. The collection has a great amount of variety representing all major genres in classical, jazz, metal, pop, etc. These midi files can be parsed by python packages such as midi and music21 into numpy arrays that preserves the note duration information, and our model will be trained on them.

Writing Tips

- Develop an outline of what you are going to write for each section before you start writing
- Good writing follows a general format: introduction, body, and conclusion. This rule applies to the entire project, to each section, and to each paragraph
- Each paragraph presents only one idea and follows the **MEAL** rule
  - **M**ain: start the paragraph with the *Main* idea
  - **E**xplain the concepts and define the terms
  - **A**nalyze the idea
  - **L**ink: conclude the paragraph and *Link* it to the next paragraph
- Make sure to define any concepts or terms that reader might be unfamiliar with right before or after they are used

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Spacing and Markdown

Be careful with your spacing in *Markdown* documents. While white-space largely is ignored, it does at times give *Markdown* signals as to how to proceed. As a habit, try to keep everything left aligned whenever possible, especially as you type a new paragraph. In other words, there is no need to indent basic text in the Rmd document (in fact, it might cause your text to do funny things if you do).

R Markdown Basics

Lists

It's easy to create a list. It can be unordered like

- Item 1
- Item 2

or it can be ordered like

1. Item 1
2. Item 2

Notice that I intentionally mislabeled Item 2 as number 4. *Markdown* automatically figures this out! You can put any numbers in the list and it will create the list. Check it out below.

To create a sub-list, just indent the values a bit (at least four spaces or a tab). (Here's one case where indentation is key!)

1. Item 1
2. Item 2
3. Item 3
    - Item 3a
    - Item 3b

Line breaks

Make sure to add white space between lines if you'd like to start a new paragraph. Look at what happens below in the outputted document if you don't:

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph. This should be a new paragraph.

Now for the correct way:

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph.

This should be a new paragraph.

Background

This section provides a literature review that informs the reader in the following ways:

- Provides background information so the reader may understand related concepts
- Identifies and defines concepts and variables involved and ummarizes what is known
- Motivates the research and highlights its importance within the literature described

Introduction

- Define the focus: e.g., pricing models based on emerging market conditions
- Introduce the thesis: e.g., existing models are ex-post and there is a need for ex-ante

Body

- Elaborate on the thesis
- Review literature to substantiate your assumptions and build support for the thesis
- Resolve conflicting findings to the thesis and formulate alternative theses
- Discuss why the alternatives are not favored and Identify knowledge gaps
- Identify and motivate the contribution of the research

Conclusion

- Summarize the evidence in support of your thesis
- Summarize why alternative theses are not favored
- State the research question and hypotheses

Code chunks

When you click the **Knit** button above a document will be generated that includes both content as well as the output of any embedded code chunks within the document. You can embed a code chunk as demonstrated below.

The `sleep` data is a built-in **R** dataset (`Student-t` (1908) *The probable error of the mean*. Biometrika, 6, 20). It shows the effect of two soporific drugs (increase in hours of sleep compared to control) on 10 patients.

Use the `kable` function to create quick, decent looking tables. Other packages for higher quality tables are xtable and stargazar. These are especially recommended for working with LaTeX output.

```r
data(sleep)
sleep2 <- data.frame(sleep[1:10, 1], sleep[11:20, 1])
colnames(sleep2) <- c("extra_sleep_drug1", "extra_sleep_drug2")
kable(sleep2, row.names = TRUE, caption = "Sleep Data",
                format = "latex", longtable = TRUE)
```

Table 1: Sleep Data

	extra_sleep_drug1	extra_sleep_drug2
1	0.7	1.9
2	-1.6	0.8
3	-0.2	1.1
4	-1.2	0.1
5	-0.1	-0.1
6	3.4	4.4
7	3.7	5.5
8	0.8	1.6
9	0.0	4.6
10	2.0	3.4

Linked tables and List of Tables

We linked Table 1 above to the `List of Tables` section following the table of contents. For this to work properly, two steps are required.

1. The code chunk of the table must be given a label argument. ie; `{r sleep, args}`.
2. Define a caption in the `kable` function (`caption = "Sleep Data"`) call. The caption will be passed to the **`List of Tables`** section and is required for an entry to appear.

More than R: Other Languages

R Markdown supports the following languages:

```
names(knitr::knit_engines$get())
```

```
 [1] "awk"         "bash"        "coffee"      "gawk"        "groovy"
 [6] "haskell"     "lein"        "mysql"       "node"        "octave"
[11] "perl"        "psql"        "Rscript"     "ruby"        "sas"
[16] "scala"       "sed"         "sh"          "stata"       "zsh"
[21] "highlight"   "Rcpp"        "tikz"        "dot"         "c"
[26] "fortran"     "fortran95"   "asy"         "cat"         "asis"
[31] "stan"        "block"       "block2"      "js"          "css"
[36] "sql"         "go"          "python"      "julia"       "sass"
[41] "scss"        "theorem"     "lemma"       "corollary"   "proposition"
[46] "conjecture"  "definition"  "example"     "exercise"    "proof"
[51] "remark"      "solution"
```

Of these, if you use **`python`**, first load `reticulate`.

```
library(reticulate)
pyPath <- py_discover_config()
use_python(pyPath$python_versions[1])
```

Create a `conda` environment to store packages, install, and load one.

```
conda_create("r-reticulate")
py_install("numpy")
numpy <- import("numpy")
```

Define a python code chunk with arguments and run code: `{python, eval=FALSE, etc}`

```
import numpy as np
print(np.reshape(np.arange(1,25), (4,3,2), "F"))
```

Please see the Reticulate: R interface to Python for more specifics.

Let's not forget about C++/Rcpp enables compilation of C++ into R functions.

```
#include <Rcpp.h>
using namespace Rcpp;
// [[Rcpp::export]]
NumericVector humanPercent(NumericVector x) {
  return x * 100;
}
```

```
humanPercent(x=0.0274)
```

Please see Rcpp for Seamless R and C++ Integration, which is a very mature project and provides many examples.

Including plots

You can also embed plots. For example, here is a way to use the base **R** graphics package to produce a plot using the built-in `pressure` dataset. In this case, the `echo=FALSE` parameter was added to the code chunk to prevent printing the code.

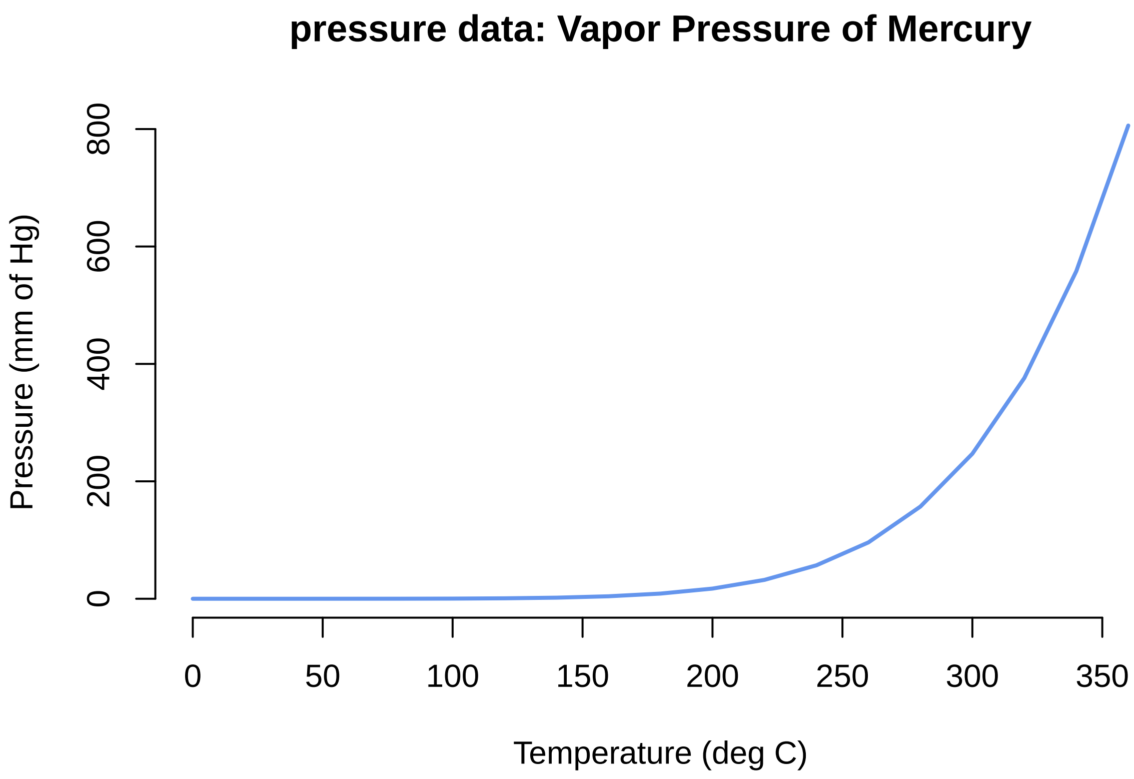

Figure 1: Pressure Plot

This is Figure 1: pressure data. Internally, we also labeled the code chunk `pressureplot` so we can call its contents and print them in the Appendix. see Appendix for the code that generated this plot. There are many arguments which govern the behavior of your code chunks. The creator of `knitr` has a many notes on this `http://yihui.name/knitr/options/`.

R Markdown Tables, Graphics, References, and Labels

In addition to the tables that can be automatically generated from a data frame in **R** that you saw in R Markdown Basics using the `kable` function, you can also create tables using *pandoc*. (More information is available at `http://pandoc.org/README.html#tables`.) This might be useful if you don't have values specifically stored in **R**, but you'd like to display them in table form. Below is an example. Pay careful attention to the alignment in the table and hyphens to create the rows and columns.

Table 2: Correlation of Inheritance Factors for Parents and Child

Factors	Correlation between Parents & Child	Inherited
Education	-0.49	Yes
Socio-Economic Status	0.28	Slight
Income	0.08	No
Family Size	0.18	Slight

We can also create a link to the table like so: Table 2.
To create the link, place "Table `\@ref(tab:sleep)`", using the label argument defined in step 1. This can be helpful to reference in other parts of the document.

Inline code

If you'd like to put the results of your analysis directly into your discussion, add inline code like this:

> The `cos` of 2π is "r cos(2*pi)".
>
> The `cos` of 2π is 1.

Another example would be the direct calculation of the standard deviation:

> The standard deviation of `speed` in `cars` is "r sd(cars$speed)".
>
> The standard deviation of `speed` in `cars` is 5.2876444.

One last neat feature is the use of the `ifelse` conditional statement which can be used to output text depending on the result of an **R** calculation:

> r ifelse(sd(cars$speed) < 6, "The standard deviation is less than 6.", "The standard deviation is equal to or greater than 6.")
>
> The standard deviation is less than 6.

Note the use of > here, which signifies a quotation environment that will be indented.

As you see with `$2 \pi$` above, mathematics can be added by surrounding the mathematical text with dollar signs. More examples of this are in Math Examples.

Figures

If your capstone has a lot of figures, *R Markdown* might behave better for you than that other word processor. One perk is that it will automatically number the figures accordingly in each chapter. You'll also be able to create a label for each figure, add a caption, and then reference the figure in a way similar to what we saw with tables earlier. If you label your figures, you can move the figures around and *R Markdown* will automatically adjust the numbering for you. No need for you to remember! So that you don't have to get too far into LaTeX to do this, a couple **R** functions have been created for you to assist. You'll see their use below.

In the **R** chunk below, we will load in a picture stored as `phoenix-logo.png` in our main directory. We then give it the caption of "Phoenix logo", the label of "phoenixlogo", and specify that this is a figure. Make note of the different **R** chunk options that are given in the R Markdown file (not shown in the knitted document).

```r
include_graphics(path = "figure/phoenix-logo.jpg")
```

Figure 2: Phoenix logo

Here is a reference to the Phoenix logo: Figure 2. Note the use of the `fig:` code here. By naming the **R** chunk that contains the figure, we can then reference that figure later as done in the first sentence here. We can also specify the caption for the figure via the R chunk option `fig.cap`.

Next, we will explore the use of the `out.extra` chunk option, which can be used to shrink or expand an image loaded from a file by specifying `"scale= "`. Here we use the mathematical graph stored in the "subdivision.pdf" file.

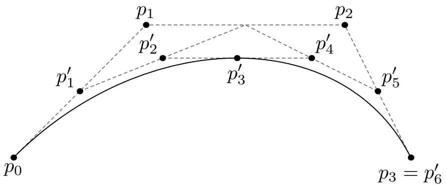

Figure 3: Subdiv. graph

Here is a reference to this image: Figure 3. Note that `echo=FALSE` is specified so that the **R** code is hidden in the document.

More Figure Stuff

Lastly, we will explore how to rotate and enlarge figures using the `out.extra` chunk option. (Currently this only works in the PDF version of the book.)

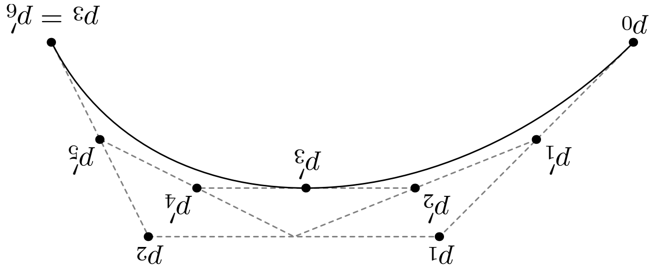

Figure 4: A Larger Figure, Flipped Upside Down

As another example, here is a reference: Figure 4.

Footnotes and Endnotes

You might want to footnote something.[1] The footnote will be in a smaller font and placed appropriately. Endnotes work in much the same way.

[1] footnote text

Methodology

The methodology section may include the following subsections:

- Data
- Descriptive analyses
- Modeling Framework

Data

General description of the data and source(s). This includes limitations and delimitations; units of analysis; time window for aggregation and modeling (if applicable); and validation and development samples.

Let's revisit the work of E.W. Crampton's November 22, 1946 paper titled *The Growth of the Odontoblasts of the Incisor Tooth as a Criterion of the Vitamin C Intake of the Guinea Pig*. It is one of the data sets available within in the R statistical programming environment.

From the description file (`?ToothGrowth`):

> The response is the length of odontoblasts (cells responsible for tooth growth) in 60 guinea pigs. Each animal received one of three dose levels of vitamin C (0.5, 1, and 2 mg/day) by one of two delivery methods, orange juice (OJ) or ascorbic acid (a form of vitamin C, coded VC).

Variables:

- *len*: *numeric*. Tooth length
- *supp*: *factor*. Supplement type (VC or OJ).
- *dose*: *numeric*. Dose in milligrams/day

```
data(ToothGrowth)
```

Descriptive analyses

Identification of the methodologies that would provide a view into the data. These may include highlighting the relationships that are important for the research, validating assumptions with respect to important metrics, and providing evidence that substantiates the way data is analyzed.

```
growth_short <- rbind(ToothGrowth[1:7,], ToothGrowth[26:34,], ToothGrowth[53:60,])
kable(growth_short, align = "r", format = "latex",
      longtable = TRUE, caption = "Tooth Growth")
```

Table 3: Tooth Growth

	len	supp	dose
1	4.2	VC	0.5
2	11.5	VC	0.5
3	7.3	VC	0.5
4	5.8	VC	0.5
5	6.4	VC	0.5
6	10.0	VC	0.5
7	11.2	VC	0.5
26	32.5	VC	2.0
27	26.7	VC	2.0
28	21.5	VC	2.0
29	23.3	VC	2.0
30	29.5	VC	2.0
31	15.2	OJ	0.5
32	21.5	OJ	0.5
33	17.6	OJ	0.5
34	9.7	OJ	0.5
53	22.4	OJ	2.0
54	24.5	OJ	2.0
55	24.8	OJ	2.0
56	30.9	OJ	2.0
57	26.4	OJ	2.0
58	27.3	OJ	2.0
59	29.4	OJ	2.0
60	23.0	OJ	2.0

Table: 3: Head, center, and tail of *Tooth Growth* data.

Modeling Framework

Justification of model(s) selected. Identification of dependent and independent variables, per model as well as variable transformations. Feature extraction (if applicable). Discussion of model(s) functional form. Assumptions of model(s) and ways of insuring that assumptions are observed or tested. Assessing model(s) performance and validation.

Transform `dose` into a `factor`. Only three dosage levels are present.

```
data(ToothGrowth)
colnames(ToothGrowth) <- c("length", "supplement", "dose")
ToothGrowth$dose <- as.factor(ToothGrowth$dose)
```

We are most interested in discovering which treatment leads to the optimal tooth growth. In this vein, we use `aggregate` function to transform our data and compute the average tooth `length` by both `supplement` type and `dose` size.

```
groupedTooth <- aggregate(ToothGrowth, by=ToothGrowth[,2:3], FUN=mean)[,1:3]

kable(groupedTooth, align = "r", caption = "Average tooth length",
      format = "latex", longtable = TRUE)
```

Table 4: Average tooth length

supplement	dose	length
OJ	0.5	13.23
VC	0.5	7.98
OJ	1	22.70
VC	1	16.77
OJ	2	26.06
VC	2	26.14

Math and Science notation

TEX is the best way to typeset mathematics. Donald Knuth designed TEX when he got frustrated at how long it was taking the typesetters to finish his book, which contained a lot of mathematics. One nice feature of *R Markdown* is its ability to read LATEX code directly.

Get around math mode's automatic italicizing in LaTeX by using the argument `$\mathrm{formula here}$`, with your formula inside the curly brackets. (Notice the use of the backticks here which enclose text that acts as code.)

So, $\mathrm{Fe_2^{2+}Cr_2O_4}$ is written `$\mathrm{Fe_2^{2+}Cr_2O_4}$`.

The command below does what you'd expect: it forces the current line/paragraph to not indent. See below and examples of commonly used symbols:

Exponent or Superscript written as `$x^2$` becomes x^2

Subscript written as `$x_1$` becomes x_1

Infinity written as `$\infty$` becomes ∞

alpha written as `$\alpha$` becomes α

beta written as `$\beta$` becomes β

delta written as `$\delta$` becomes δ

epsilon written as `$\epsilon$` becomes ϵ

sigma written `$\sum_{i=1}^n f(x)$` becomes $\sum_{i=1}^n f(x)$

Math Examples

An Ordinary Least Squares model, from *Introductory Econometrics, 6th edition* by Jeffrey M. Wooldridge, page 27.

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

An infinite distributed lag (IDL) time series model, by Wooldridge, page 633.

$$y_t = \alpha + \delta_0 z_t + \delta_1 z_{t-1} + \delta_2 z_{t-2} \ldots + \epsilon_t$$

A vector autoregressive (VAR) model, by Wooldridge, page 657.

$$y_t = \delta_0 + \alpha_1 y_{t-1} + \gamma_1 z_{t-1} + \alpha_2 y_{t-2} + \gamma_2 z_{t-2} \ldots,$$

Determinant of a square matrix:

$$\det \begin{vmatrix} c_0 & c_1 & c_2 & \dots & c_n \\ c_1 & c_2 & c_3 & \dots & c_{n+1} \\ c_2 & c_3 & c_4 & \dots & c_{n+2} \\ \vdots & \vdots & \vdots & & \vdots \\ c_n & c_{n+1} & c_{n+2} & \dots & c_{2n} \end{vmatrix} > 0$$

A regularization problem solved by Jerome Friedman, Trevor Hastie, Rob Tibshirani and Noah Simon, implemented in the R package `glmnet`.

$$\min_{\beta_0,\beta} \frac{1}{N} \sum_{i=1}^{N} w_i l(y_i, \beta_0 + \beta^T x_i) + \lambda \left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1\right]$$

From Lapidus and Pindar, Numerical Solution of Partial Differential Equations in Science and Engineering, page 54.

$$\int_t \left\{ \sum_{j=1}^{3} T_j \left(\frac{d\phi_j}{dt} + k\phi_j \right) - kT_e \right\} w_i(t)\; dt = 0, \qquad\qquad i = 1,2,3.$$

From Lapidus and Pindar, page 145.

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f\big(\xi,\eta,\zeta\big) = \sum_{k=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} w_i w_j w_k f\big(\xi,\eta,\zeta\big).$$

Additional R Markdown and bookdown resources

- *Bookdown* Online Book - `https://bookdown.org/yihui/bookdown/`
- *Markdown* Info Sheet - `https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet`
- *R Markdown* Reference Guide - `https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf`

Findings

Should be organized as follows:

- Results of descriptive analyses
- Modeling results
- Results of model performance and validation

Results of descriptive analyses

```r
kable(summary(ToothGrowth), align = "r", caption = "Summary of ToothGrowth data",
      format = "latex", longtable = TRUE)
```

Table 5: Summary of ToothGrowth data

	length	supplement	dose
	Min. : 4.20	OJ:30	0.5:20
	1st Qu.:13.07	VC:30	1 :20
	Median :19.25	NA	2 :20
	Mean :18.81	NA	NA
	3rd Qu.:25.27	NA	NA
	Max. :33.90	NA	NA

Table 5 above contains summary statistics of the *Tooth Growth* data.

While the code is not displayed to create the graph below (`echo=FALSE`), it is displayed in the Appendix by referencing the `boxplot` chunk name..

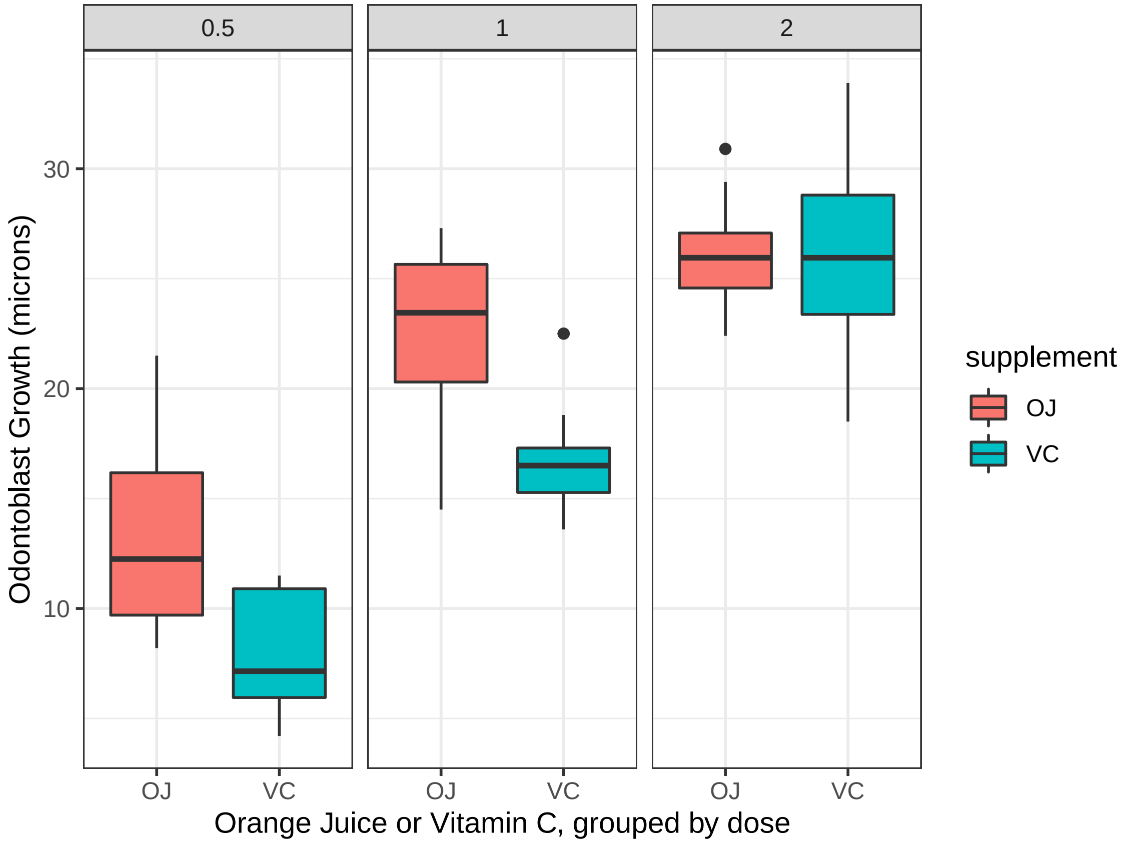

Figure 5: Avg. length by supplement and dose

Figure 5 was created with the `ggplot2` package. We can visually compare the average tooth growth by `supplement` and `dose`.

Modeling results

First, use a `t.test()` to test *if* dosage leads to growth of incisor length. From the results below, it appears every test rejects the null hypothesis.

```
test1 <- t.test(length ~ dose, ToothGrowth, dose %in% c(0.5,1))
test2 <- t.test(length ~ dose, ToothGrowth, dose %in% c(0.5,2))
test3 <- t.test(length ~ dose, ToothGrowth, dose %in% c(1,2))

testAgg <- data.frame(Name = c("Test 0.5-1", "Test 0.5-2", "Test 1-2"),
                  Method = c(test1$method, test2$method, test3$method),
               Pvalue = c(test1$p.value, test2$p.value, test3$p.value),
          Tstat = c(test1$statistic, test2$statistic, test3$statistic))

kable(testAgg, digit = 7, align = "r", caption = "t-test results",
       format = "latex", longtable = TRUE)
```

Table 6: t-test results

Name	Method	Pvalue	Tstat
Test 0.5-1	Welch Two Sample t-test	1.00e-07	-6.476648
Test 0.5-2	Welch Two Sample t-test	0.00e+00	-11.799046
Test 1-2	Welch Two Sample t-test	1.91e-05	-4.900484

Table 6

Results of model performance and validation

Next, subset the `ToothGrowth` data into seperate data sets defined by supplement dose of 0.5, 1, and 2 mg. This allow us to controlling for dose increases of *economic* significance.

Subset tooth data into a separate `data.frame` for each dosage level. Then Execute the `t.test()` function for the dosage of 0.5 mg and display the results.

```
dose05 <- ToothGrowth[ToothGrowth$dose == 0.5, ]
 dose1 <- ToothGrowth[ToothGrowth$dose == 1, ]
 dose2 <- ToothGrowth[ToothGrowth$dose == 2, ]

 test4 <- t.test(length ~ supplement, data = dose05)
 test5 <- t.test(length ~ supplement, data = dose1)
 test6 <- t.test(length ~ supplement, data = dose2)
```

Place the results of the analysis directly into your content with ***inline code*** functions:

With a very low p-value of 0.0064 and a corresponding t-statistic of 3.1697, it appears that at low doses, *Orange Juice* is the preferable delivery mechanism to *Vitamin C* for Ascorbic Acid delivery.

The p-value and t-statistic above have been directly extracted from the model object and printed inline. using the ‘r foo’ syntax with quotes(’) replaced by back-ticks (‘).

Conclusion

This section includes a concise summary of the findings. Your summary might be organized by the research objectives or hypotheses. Make sure you address the extent to which research objectives are achieved, and if they are not achieved, explain why. Make sure to interpret your findings in a way that acknowledges the limitations of the research. That is, do not extrapolate the insights derived from your research to situations you have not examined.

While increasing dosage leads to larger incisor length, the choice of delivery mechanism between Orange Juice and Vitamin C does not seem to make a difference. However, at very low levels, Orange Juice appears more effective, displaying higher average growth.

Recommendations

Includes guidelines as to ways in which your results should or could be used in practice. You may discuss other uses of your results, if there are any. The ways to extend your analysis and the benefits of doing so might be included in this section as well.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In section :

```r
plot(pressure, type = "l", col="cornflowerblue", lwd = 2,
               xlab = "Temperature (deg C)",
               ylab = "Pressure (mm of Hg)",
               main = "pressure data: Vapor Pressure of Mercury",
               frame = FALSE)
```

In section :

```r
data(ToothGrowth)
colnames(ToothGrowth) <- c("length", "supplement", "dose")
ToothGrowth$dose <- as.factor(ToothGrowth$dose)

groupedTooth <- aggregate(ToothGrowth, by=ToothGrowth[,2:3], FUN=mean)[,1:3]

library(ggplot2)
 ggplot(ToothGrowth, aes(x = supplement, y = length)) +
                     geom_boxplot(aes(fill=supplement)) +
                     facet_wrap(~dose) +
                     guides(colour = guide_legend("Color = Supplement")) +
                     labs(x="Orange Juice or Vitamin C, grouped by dose",
                          y="Odontoblast Growth (microns)")  +
                     theme_bw()
```

Appendix B

A Second Appendix, for example

References

There are a variety of tools available for creating a bibliography database (stored with the .bib extension). In addition to BibTeX suggested below, you may want to consider using the free and easy-to-use tool called Zotero.

R Markdown uses *pandoc* (`http://pandoc.org/`) to build its bibliographies. To cite references in your thesis (after creating your bibliography database), place the reference name inside square brackets and precede it by the "at" symbol. For example, here's a reference to a book about worrying: (Molina & Borkovec, 1994). This `Molina1994` entry appears in a file called `thesis.bib` in the `bib` folder. This bibliography database file was created by a program called BibTeX. You can call this file something else if you like (look at the YAML header in the main .Rmd file) and, by default, is to placed in the `bib` folder.

Additional Tips

- The sooner you start compiling your bibliography for something as large as a capstone, the better. Typing in source after source is mind-numbing enough; do you really want to do it for hours on end at the last minute?
- The cite key (a citation's label) needs to be unique from the other entries.
- When you have more than one author or editor, you need to separate each author's name by the word "and" e.g. `Author = {Noble, Sam and Youngberg, Jessica},`

Example output generated from bib file

Angel, E. (2000). *Interactive computer graphics : A top-down approach with opengl*. Boston, MA: Addison Wesley Longman.

Angel, E. (2001a). *Batch-file computer graphics : A bottom-up approach with quicktime*. Boston, MA: Wesley Addison Longman.

Angel, E. (2001b). *Test second book by angel*. Boston, MA: Wesley Addison Longman.

Molina, S. T., & Borkovec, T. D. (1994). The Penn State worry questionnaire: Psychometric properties and associated characteristics. In G. C. L. Davey & F. Tallis (Eds.), *Worrying: Perspectives on theory, assessment and treatment* (pp. 265–283). New York: Wiley.