A Study in Replication

Creative Coding in R

Thomas Wise; 6664202

Introduction

Aim:

In this document, we will explore the creative implications of using replication, in order to generate reproducible aRt-work. This will intergrate components which performs and conducts statistical analysis in order to contribute to the artist design. In this study of replication, it will directly build upon both the creative and core simulating factors discussed in the previous document *Art in Many Forms*.

Package Loading:

The same packages from the previous document will be loaded, these include both *core* packages which I typically use for any project, and additional packages which are useful for this document.

```
## Core Packages
  library(tidyverse)      ## In particular ggplot2
  library(RColorBrewer)   ## For colour experimentation
  library(Rcpp)           ## For CPP intergration


## Additional Packages
  # Those used within publication of this pdf
  library(kableExtra)
  library(rmarkdown)
  library(psych)
  library(DescTools)
```

Fixing the Random Seed

```
set.seed(300920)
```

Setup

For this study in Replication, we will be conducting statistical tests upon several different types of simulated distributions, in order to be able to compare their visual and statistical differences. And therefore, how this statistical knowledge can be applied to create interesting artist designs.

Distribution Definitions

For this investigation we will use several classical distributions, which can be compared in both polar and cartesian space.

Normal Distribution

A normal distribution is achieved using the noted, equation 1, this indicates that to construct a normal distribution both the *mean* (μ) and *standard deviation* (σ^2) are required. For this example, let us deviate from the traditional default and use the parameters defined in equation 2.

$$X \sim \mathcal{N}(\mu, \sigma^2) \tag{1}$$

$$X \sim \mathcal{N}(10, 2) \tag{2}$$

From this, we can generate a data set of 1000 data points, using the function *rnorm* (stats).

```r
## Set General Population Count (this will remain Constant)
  pop.n <- 1000

## Normal Distribution Parameters
  mu.nd <- 10
  sd.nd <- 2

## Simulate the Distribution (this is saved as an array)
  nd.sim <- rnorm(n = pop.n, mean = mu.nd, sd = sd.nd)
```

Gamma Distribution

Next, we will discuss the Gamma Distribution (Equation 3). This although more common in fields such as econometric and life sciences than social sciences (as the case for normal distributions). This distribution is defined through its parameters *shape* ($\alpha = k$), and *rate* ($\beta = 1/\theta$). For this, let us use the values defined in equation 4.

$$X \sim \Gamma(\alpha,\, \beta) \tag{3}$$

$$X \sim \Gamma(2,\, 2) \tag{4}$$

As a result the following code can be generated.

```r
## Gamma Distribution Parameters
  alpha.gd <- 2
  beta.gd <- 2

## Simulate the Distribution (this is saved as an array)
  gd.sim <- rgamma(n = pop.n, shape = alpha.gd, rate = beta.gd)
```

Poisson Distribution

Next, we will explore the Poisson distribution, this classical can form a distribution similar to that of a normal distribution, however it is precisely for this reason it is important to explore how the values randomly generated through this function differs from that of the normal distribution and therefore how differs artistically. This distribution is displayed using the notation in Equation 5, wherein the parameter λ is equal to both the expected value of X and its Variance. For our example, we will use the value wherein $\lambda = 5$, as seen in equation 6.

$$X \sim Pois(\lambda) \tag{5}$$

$$X \sim Pois(5) \tag{6}$$

```r
## Poisson Distribution Parameters
  lamd.pd <- 5

## Simulate the Distribution (this is saved as an array)
  pd.sim <- rpois(n = pop.n, lambda = lamd.pd)
```

Binomial Distribution

The final distribution to examine is the binomial distribution.

$$X \sim \mathcal{B}(n, p) \tag{7}$$

$$X \sim \mathcal{B}(n, p) \tag{8}$$

```r
## Binomial Distribution Parameters
  size.bd <- 10
  prob.bd <- 0.5

## Simulate the Distribution (this is saved as an array)
  bd.sim <- rbinom(n = pop.n, size = size.bd, prob.bd)
```

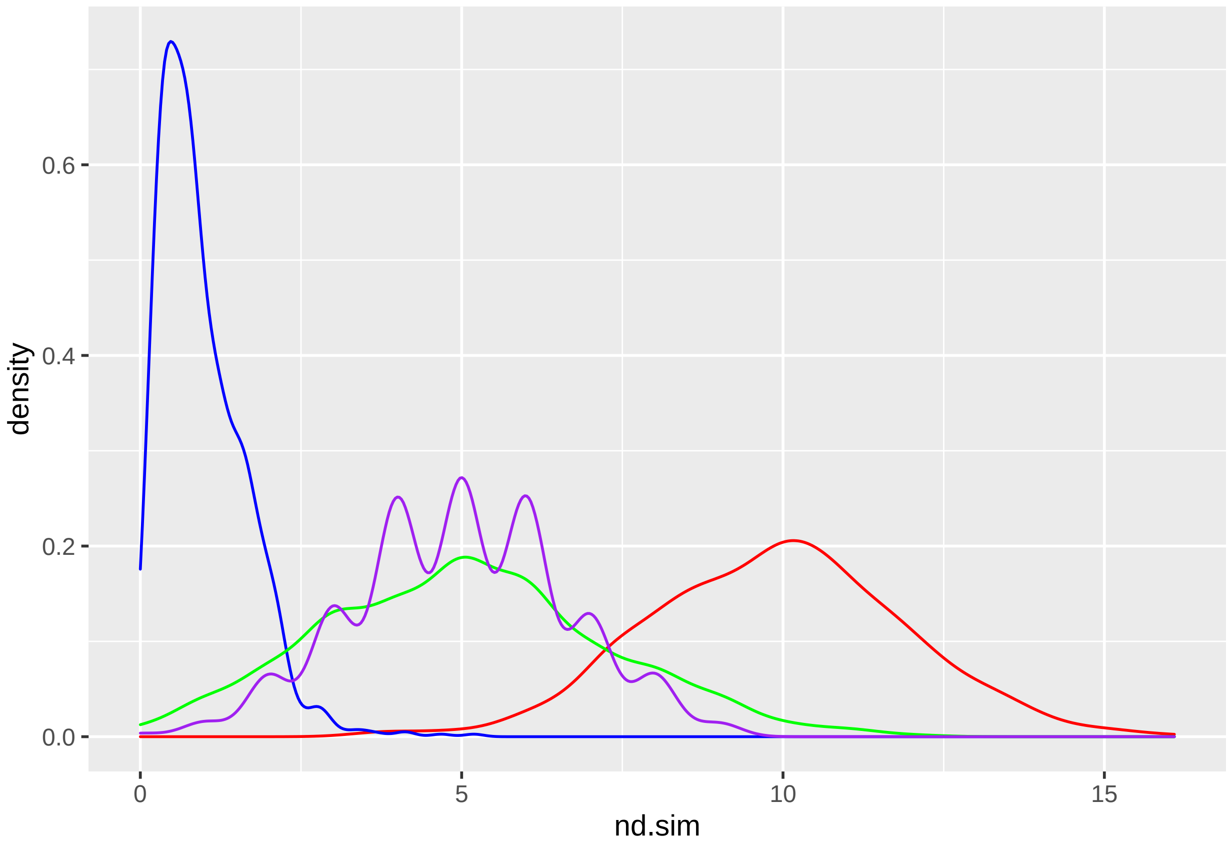
Distribution Visualisation

Before we conduct any statistical tests upon these simulated distributions, we should visualize the distribution using the *geom_density()* function (ggplot2). To understand the differences between the distributions.

```r
## First Convert all the arrays to Data Frames and rename column names.
  nd.sim <- as.data.frame(nd.sim)
  colnames(nd.sim) <- "nd.sim"
  gd.sim <- as.data.frame(gd.sim)
  colnames(gd.sim) <- "gd.sim"
  pd.sim <- as.data.frame(pd.sim)
  colnames(pd.sim) <- "pd.sim"
  bd.sim <- as.data.frame(bd.sim)
  colnames(bd.sim) <- "bd.sim"

## Plot the Density of these distributions

  ggplot() +
  geom_density(data = nd.sim, mapping = aes(nd.sim), colour = "red") +      # Normal
  geom_density(data = gd.sim, mapping = aes(gd.sim), colour = "blue") +     # Gamma
  geom_density(data = pd.sim, mapping = aes(pd.sim), colour = "green") +    # Poisson
  geom_density(data = bd.sim, mapping = aes(bd.sim), colour = "purple")     # Binomial
```



Statistical Tests

To explore the impact of these distributions we can run several statistical tests on these different distributions and explore the differences using the artist techniques reviewed before. For this, we will calculate the standard descriptive statistics (Mean (μ), Standard Deviation σ^2, Min, Max, Skew and Quartiles). These can then be displayed in Table 1.

```
## Prepare the data through combining all the different distributions into one data frame
  dis.df <- cbind(nd.sim, gd.sim, pd.sim, bd.sim)

## Generate an Empty Data frame for population
  dsp.df <- as.data.frame(matrix(data = NA, nrow = 4, ncol = 7))
  colnames(dsp.df) <- c("Mean", "SD", "Min", "Max", "Skew", "Q0.025", "Q0.975")
  rownames(dsp.df) <- c("Normal", "Gamma", "Poisson", "Binomial")

## Calculate the Descriptive Statistics for each
  ## Means
    dsp.df[,1] <- colMeans(dis.df)
  ## SD
    dsp.df[,2] <- apply(dis.df, 2, sd)

  ## Min
    dsp.df[,3] <- apply(dis.df, 2, min)

  ## Max
    dsp.df[,4] <- apply(dis.df, 2, max)

  ## Skew
    dsp.df[,5] <- apply(dis.df, 2, Skew)

  ## Quant (lower)
    dsp.df[,6] <- describe(dis.df, quant = c(0.05))[14]

  ## Quant (Upper)
    dsp.df[,7] <- describe(dis.df, quant = c(0.975))[14]

## Print Table
    kable(dsp.df, format = "latex") %>%
      kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

	Mean	SD	Min	Max	Skew	Q0.025	Q0.975
Normal	9.8978026	1.9975693	3.4097582	16.089333	-0.0322573	6.708759	13.769776
Gamma	0.9596199	0.6684158	0.0189848	5.189502	1.3822265	0.178229	2.453973
Poisson	5.0380000	2.2247484	0.0000000	12.000000	0.2148923	1.000000	9.025000
Binomial	4.9710000	1.6591901	0.0000000	9.000000	-0.0498537	2.000000	8.000000

This as a whole will generate a reproducible output, as an .rmd & pdf file, additionally the session information can also be logged.

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] DescTools_0.99.38  psych_2.0.9        rmarkdown_2.5      kableExtra_1.3.1
##  [5] Rcpp_1.0.5         RColorBrewer_1.1-2 forcats_0.5.0      stringr_1.4.0
##  [9] dplyr_1.0.2        purrr_0.3.4        readr_1.4.0        tidyr_1.1.2
## [13] tibble_3.0.4       ggplot2_3.3.2      tidyverse_1.3.0
```

```
##
## loaded via a namespace (and not attached):
##  [1] httr_1.4.2        jsonlite_1.7.1    viridisLite_0.3.0 tmvnsim_1.0-2
##  [5] modelr_0.1.8      assertthat_0.2.1  expm_0.999-5      gld_2.6.2
##  [9] lmom_2.8          blob_1.2.1        cellranger_1.1.0  yaml_2.2.1
## [13] pillar_1.4.6      backports_1.1.10  lattice_0.20-41   glue_1.4.2
## [17] digest_0.6.27     rvest_0.3.6       colorspace_1.4-1  htmltools_0.5.0
## [21] Matrix_1.2-18     pkgconfig_2.0.3   broom_0.7.2       haven_2.3.1
## [25] mvtnorm_1.1-1     scales_1.1.1      webshot_0.5.2     rootSolve_1.8.2.1
## [29] farver_2.0.3      generics_0.0.2    ellipsis_0.3.1    withr_2.3.0
## [33] cli_2.1.0         mnormt_2.0.2      magrittr_1.5      crayon_1.3.4
## [37] readxl_1.3.1      evaluate_0.14     fs_1.5.0          fansi_0.4.1
## [41] nlme_3.1-149      MASS_7.3-53       xml2_1.3.2        class_7.3-17
## [45] tools_4.0.3       hms_0.5.3         lifecycle_0.2.0   Exact_2.1
## [49] munsell_0.5.0     reprex_0.3.0      compiler_4.0.3    e1071_1.7-4
## [53] rlang_0.4.8       grid_4.0.3        rstudioapi_0.11   labeling_0.4.2
## [57] boot_1.3-25       gtable_0.3.0      DBI_1.1.0         R6_2.4.1
## [61] lubridate_1.7.9   knitr_1.30        stringi_1.5.3     parallel_4.0.3
## [65] vctrs_0.3.4       dbplyr_1.4.4      tidyselect_1.1.0  xfun_0.18
```