Take-Home Exercise 2

Belinda Lim

2025-05-19

# **1  Mini-Challenge 1**

The case was extracted from [VAST Challenge 2025](https://vast-challenge.github.io/2025/MC1.html).

## 1.1 Background

One of music’s biggest superstars is Oceanus native Sailor Shift. From humble beginnings, Sailor has grown in popularity and now enjoys fans around the world. Sailor started her career on the island nation of Oceanus which can be clearly seen in her early work, she started in the genre of “Oceanus Folk”. While Sailor has moved away from the traditional Oceanus style, the Oceanus Folk has made a name for itself in the musical world. The popularity of this music is one of the factors driving an increase in tourism to a quiet island nation that used to be known for fishing.

In 2023, Sailor Shift joined the Ivy Echoes – an all-female Oceanus Folk band consisting of Sailor (vocalist), Maya Jensen (vocalist), Lila “Lilly” Hartman (guitarist), Jade Thompson (drummer), and Sophie Ramirez (bassist). They played together at venues throughout Oceanus but had broken up to pursue their individual careers by 2026. Sailor’s breakthrough came in 2028 when one of her singles went viral, launched to the top of the global charts (something no other Oceanus Folk song had ever done). Since then, she has only continued to grow in popularity worldwide.

Sailor has released a new album almost every year since her big break, and each has done better than the last. Although she has remained primarily a solo artist, she has also frequently collaborated with other established artists, especially in the Indie Pop and Indie Folk genres. She herself has branched out musically over the years but regularly returns to the Oceanus Folk genre — even as the genre’s influence on the rest of the music world has spread even more.

Sailor has always been passionate about two things: (1) spreading Oceanus Folk, and (2) helping lesser-known artists break into music. Because of those goals, she’s particularly famous for her frequent collaborations.

Additionally, because of Sailor’s success, more attention began to be paid over the years to her previous bandmates. All 4 have continued in the music industry—Maya as an independent vocalist, Lilly and Jade as instrumentalists in other bands, and Sophie as a music producer for a major record label. In various ways, all of them have contributed to the increased influence of Oceanus folk, resulting in a new generation of up-and-coming Oceanus Folk artists seeking to make a name for themselves in the music industry.

Now, as Sailor returns to Oceanus in 2040, a local journalist – Silas Reed – is writing a piece titled Oceanus Folk: Then-and-Now that aims to trace the rise of Sailor and the influence of Oceanus Folk on the rest of the music world. He has collected a large dataset of musical artists, producers, albums, songs, and influences and organized it into a knowledge graph. Your task is to help Silas create beautiful and informative visualizations of this data and uncover new and interesting information about Sailor’s past, her rise to stardom, and her influence.

## 1.2 Tasks and Questions

1. **Design and develop visualizations and visual analytic tools that will allow Silas to explore and understand the profile of Sailor Shift’s career**
   1. Who has she been most influenced by over time?
   2. Who has she collaborated with and directly or indirectly influenced?
   3. How has she influenced collaborators of the broader Oceanus Folk community?
2. **Develop visualizations that illustrate how the influence of Oceanus Folk has spread through the musical world.**
   1. Was this influence intermittent or did it have a gradual rise?
   2. What genres and top artists have been most influenced by Oceanus Folk?
   3. On the converse, how has Oceanus Folk changed with the rise of Sailor Shift? From which genres does it draw most of its contemporary inspiration?
3. **Use your visualizations to develop a profile of what it means to be a rising star in the music industry.**
   1. Visualize the careers of three artists. Compare and contrast their rise in popularity and influence.
   2. Using this characterization, give three predictions of who the next Oceanus Folk stars with be over the next five years.

# **2  Data Pre-processing**

## **2.1 Load packages**

In the code chunk below, p\_load() of pacman packages is used to load the R packages into R environment:

Utility Tools

* jsonlite: To parse JSON
* tidyverse - Data science tools
* ggtext - Tools for text formatting
* knitr - For better table displays
* lubridate - For processing date and time

Graphing Tools

* patchwork - For combining ggplot plots
* ggraph - For plotting network data
* tidygraph - For graph manipulations
* igraph - Contains functions for network analysis
* ggiraph - Interactive plots

|  |
| --- |
| **Code**  pacman::p\_load(jsonlite, tidyverse, ggtext,  knitr, lubridate, patchwork,  ggraph, tidygraph, igraph,  ggiraph) |

## **2.2 Import Data and Inspect Structure**

In the code chunk below, fromJSON() of **jsonlite** package is used to import *MC1\_graph.json* file into R and save the output object and the data structure is inspected

|  |
| --- |
| **Code**  mc1\_data <- fromJSON("MC1data/MC1\_graph.json") glimpse(mc1\_data)  List of 5  $ directed : logi TRUE  $ multigraph: logi TRUE  $ graph :List of 2  ..$ node\_default: Named list()  ..$ edge\_default: Named list()  $ nodes :'data.frame': 17412 obs. of 10 variables:  ..$ Node Type : chr [1:17412] "Song" "Person" "Person" "Person" ...  ..$ name : chr [1:17412] "Breaking These Chains" "Carlos Duffy" "Min Qin" "Xiuying Xie" ...  ..$ single : logi [1:17412] TRUE NA NA NA NA FALSE ...  ..$ release\_date : chr [1:17412] "2017" NA NA NA ...  ..$ genre : chr [1:17412] "Oceanus Folk" NA NA NA ...  ..$ notable : logi [1:17412] TRUE NA NA NA NA TRUE ...  ..$ id : int [1:17412] 0 1 2 3 4 5 6 7 8 9 ...  ..$ written\_date : chr [1:17412] NA NA NA NA ...  ..$ stage\_name : chr [1:17412] NA NA NA NA ...  ..$ notoriety\_date: chr [1:17412] NA NA NA NA ...  $ links :'data.frame': 37857 obs. of 4 variables:  ..$ Edge Type: chr [1:37857] "InterpolatesFrom" "RecordedBy" "PerformerOf" "ComposerOf" ...  ..$ source : int [1:37857] 0 0 1 1 2 2 3 5 5 5 ...  ..$ target : int [1:37857] 1841 4 0 16180 0 16180 0 5088 14332 11677 ...  ..$ key : int [1:37857] 0 0 0 0 0 0 0 0 0 0 ... |

## **2.3 Inspect Data Structure of Nodes and Edges**

### Preview and Separate node and edge data

For Nodes

mc1\_nodes\_raw <- as\_tibble(mc1\_data$nodes)  
  
glimpse(mc1\_nodes\_raw)

Rows: 17,412  
Columns: 10  
$ `Node Type` <chr> "Song", "Person", "Person", "Person", "RecordLabel", "S…  
$ name <chr> "Breaking These Chains", "Carlos Duffy", "Min Qin", "Xi…  
$ single <lgl> TRUE, NA, NA, NA, NA, FALSE, NA, NA, NA, NA, TRUE, NA, …  
$ release\_date <chr> "2017", NA, NA, NA, NA, "2026", NA, NA, NA, NA, "2020",…  
$ genre <chr> "Oceanus Folk", NA, NA, NA, NA, "Lo-Fi Electronica", NA…  
$ notable <lgl> TRUE, NA, NA, NA, NA, TRUE, NA, NA, NA, NA, TRUE, NA, N…  
$ id <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1…  
$ written\_date <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "2020", NA, NA,…  
$ stage\_name <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…  
$ notoriety\_date <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…

kable(head(mc1\_nodes\_raw, 5))

| Node Type | name | single | release\_date | genre | notable | id | written\_date | stage\_name | notoriety\_date |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Song | Breaking These Chains | TRUE | 2017 | Oceanus Folk | TRUE | 0 | NA | NA | NA |
| Person | Carlos Duffy | NA | NA | NA | NA | 1 | NA | NA | NA |
| Person | Min Qin | NA | NA | NA | NA | 2 | NA | NA | NA |
| Person | Xiuying Xie | NA | NA | NA | NA | 3 | NA | NA | NA |
| RecordLabel | Nautical Mile Records | NA | NA | NA | NA | 4 | NA | NA | NA |

For Edges:

mc1\_edges\_raw <- as\_tibble(mc1\_data$links)  
  
glimpse(mc1\_edges\_raw)

Rows: 37,857  
Columns: 4  
$ `Edge Type` <chr> "InterpolatesFrom", "RecordedBy", "PerformerOf", "Composer…  
$ source <int> 0, 0, 1, 1, 2, 2, 3, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5…  
$ target <int> 1841, 4, 0, 16180, 0, 16180, 0, 5088, 14332, 11677, 2479, …  
$ key <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0…

kable(head(mc1\_edges\_raw, 10))

| Edge Type | source | target | key |
| --- | --- | --- | --- |
| InterpolatesFrom | 0 | 1841 | 0 |
| RecordedBy | 0 | 4 | 0 |
| PerformerOf | 1 | 0 | 0 |
| ComposerOf | 1 | 16180 | 0 |
| PerformerOf | 2 | 0 | 0 |
| ProducerOf | 2 | 16180 | 0 |
| PerformerOf | 3 | 0 | 0 |
| InterpolatesFrom | 5 | 5088 | 0 |
| InStyleOf | 5 | 14332 | 0 |
| InterpolatesFrom | 5 | 11677 | 0 |

## **2.3 Data Pre-processing**

### **2.3.1 New columns created**

New columns created to identify “Sailor” and “Oceanus\_folk” related works for subsequent analysis

mc1\_nodes\_raw <- mc1\_nodes\_raw %>%  
 mutate(  
 is\_sailor = (  
 str\_detect(name, regex("sailor shift", ignore\_case = TRUE))  
 ) %>% replace\_na(FALSE),  
   
 is\_oceanus\_folk = str\_detect(genre, regex("oceanus folk", ignore\_case = TRUE)) %>%   
 replace\_na(FALSE)  
 )

### **2.3.2 Convert date fields**

Convert date fields from “chr” to “int” for subsequent analysis across years.

Note: Date fields are only relevant for Songs and Albums.

mc1\_nodes\_raw <- mc1\_nodes\_raw %>%  
 mutate(across(c(release\_date, notoriety\_date, written\_date),  
 ~as.integer(if\_else(`Node Type` %in% c("Song", "Album"), ., NA\_character\_))))  
  
mc1\_nodes\_raw$release\_date %>% unique()

[1] 2017 NA 2026 2020 2027 2022 2007 2010 2003 2023 1997 2013 2000 2025 2029  
[16] 2015 2018 2016 2014 2028 2021 2030 2011 1994 2004 1998 1991 1999 2024 2012  
[31] 2002 2006 2008 2019 1995 1989 2032 2009 2001 1996 1990 1984 2005 1993 1986  
[46] 1985 1981 1992 1987 1988 1983 2031 1975 2035 2033 2037 2036 2039 2038 2034  
[61] 1977 1979 1980 1982 2040

### **2.3.3 Check and remove duplicates**

For Nodes, there are no duplicates after checking.

mc1\_nodes\_raw %>%  
 count(id) %>%  
 filter(n > 1)

# A tibble: 0 × 2  
# ℹ 2 variables: id <int>, n <int>

For Edges, there are duplicates shown below. The only difference are their Keys. However, Keys are not used for subsequent data analysis.

duplicate\_summary <- mc1\_edges\_raw %>%  
 count(source, target, `Edge Type`) %>%  
 filter(n > 1)  
  
mc1\_edges\_raw %>%  
 inner\_join(duplicate\_summary, by = c("source", "target", "Edge Type"))

# A tibble: 6 × 5  
 `Edge Type` source target key n  
 <chr> <int> <int> <int> <int>  
1 PerformerOf 17057 17058 0 2  
2 PerformerOf 17057 17058 1 2  
3 PerformerOf 17349 17350 0 2  
4 PerformerOf 17349 17350 2 2  
5 PerformerOf 17355 17356 0 2  
6 PerformerOf 17355 17356 2 2

Hence, we removed all duplicates using the code below:

mc1\_edges\_raw <- mc1\_edges\_raw %>%  
 distinct(source, target, `Edge Type`, .keep\_all = TRUE)  
  
mc1\_edges\_raw %>%  
 count(source, target, `Edge Type`) %>%  
 filter(n > 1)

# A tibble: 0 × 4  
# ℹ 4 variables: source <int>, target <int>, Edge Type <chr>, n <int>

### **2.3.4 Remove symbol in text**

We remove ” **’** ” from name to prevent coding errors subsequently.

mc1\_nodes\_clean <- mc1\_nodes\_raw %>%  
 mutate(  
 name = gsub("'", "", name)  
 )

### **2.3.5 Explore and Inspect Edges**

Confirm that the source and target IDs used in edges match the range of IDs in nodes

all(mc1\_edges\_raw$source %in% mc1\_nodes\_clean$id)

[1] TRUE

all(mc1\_edges\_raw$target %in% mc1\_nodes\_clean$id)

[1] TRUE

### **2.3.6** Mapping from node name to edges indexes

Map each node’s ID to its name. This is useful for labeling, joining and plotting later.

id\_map <- mc1\_nodes\_clean %>%  
 select(id, name)  
  
id\_map

# A tibble: 17,412 × 2  
 id name   
 <int> <chr>   
 1 0 Breaking These Chains   
 2 1 Carlos Duffy   
 3 2 Min Qin   
 4 3 Xiuying Xie   
 5 4 Nautical Mile Records   
 6 5 Unshackled Heart   
 7 6 Luke Payne   
 8 7 Xiulan Zeng   
 9 8 David Franklin   
10 9 Colline-Cascade Studios  
# ℹ 17,402 more rows

### **2.3.7** Map source and target to node name while keeping the ID

Join (1) from\_name based on source ID and (2) to\_name based on target ID.

The source and target are mapped to “from\_id” and “to\_id” respectively.

Create cleaned edge and norm columns while tagged to the new “from\_id” and “to\_id”:

mc1\_edges\_clean <- mc1\_edges\_raw %>%  
 left\_join(mc1\_nodes\_clean %>% select(id, from\_name = name), by = c("source" = "id")) %>%  
 rename(from\_id = source) %>%  
 left\_join(mc1\_nodes\_clean %>% select(id, to\_name = name), by = c("target" = "id")) %>%  
 rename(to\_id = target)

mc1\_edges\_clean <- mc1\_edges\_clean %>%  
 left\_join(mc1\_nodes\_clean %>% select(id, genre, `Node Type`, release\_date),   
 by = c("from\_id" = "id"))

### **2.3.8** Filter out any unmatched (invalid) edges

mc1\_edges\_clean <- mc1\_edges\_clean %>%  
 filter(!is.na(from\_id), !is.na(to\_id))

### **2.3.9** Checking for any missing nodes

missing\_nodes <- setdiff(  
 unique(c(mc1\_edges\_clean$from\_name, mc1\_edges\_clean$to\_name)),  
 mc1\_nodes\_clean$name  
)  
  
missing\_nodes

character(0)

### **2.3.10** Check Edges Schema

Define valid source and destination types for each edge type as per the data description file:

edge\_rules <- list(  
 PerformerOf = list(source = c("Person", "MusicalGroup"), target = c("Song", "Album")),  
   
 ComposerOf = list(source = c("Person"), target = c("Song", "Album")),  
   
 ProducerOf = list(source = c("Person", "RecordLabel"), target = c("Song", "Album", "Person", "MusicalGroup")),  
   
 LyricistOf = list(source = c("Person"), target = c("Song", "Album")),  
   
 RecordedBy = list(source = c("Song", "Album"), target = c("RecordLabel")),  
   
 DistributedBy = list(source = c("Song", "Album"), target = c("RecordLabel")),  
   
 InStyleOf = list(source = c("Song", "Album"), target = c("Song", "Album", "Person", "MusicalGroup")),  
   
 InterpolatesFrom = list(source = c("Song", "Album"), target = c("Song", "Album")),  
   
 CoverOf = list(source = c("Song", "Album"), target = c("Song", "Album")),  
   
 LyricalReferenceTo = list(source = c("Song", "Album"), target = c("Song", "Album")),  
   
 DirectlySamples = list(source = c("Song", "Album"), target = c("Song", "Album")),  
   
 MemberOf = list(source = c("Person"), target = c("MusicalGroup"))  
)

Check if there were erroneous edge and node relationships:

node\_type\_lookup <- mc1\_nodes\_clean %>%  
 select(name, `Node Type`) %>%  
 deframe()  
  
mc1\_edges\_checked <- mc1\_edges\_clean %>%  
 mutate(  
 source\_type = node\_type\_lookup[from\_name],  
 target\_type = node\_type\_lookup[to\_name]  
)  
  
mc1\_edges\_tagged <- mc1\_edges\_checked %>%  
 rowwise() %>%  
 mutate(  
 valid = {  
 rule <- edge\_rules[[`Edge Type`]]  
 if (is.null(rule)) TRUE  
 else {  
 source\_type %in% rule$source && target\_type %in% rule$target  
 }  
 }  
 ) %>%  
 ungroup()  
  
invalid\_edge\_summary <- mc1\_edges\_tagged %>%  
 filter(!valid) %>%  
 count(`Edge Type`, source\_type, target\_type, sort = TRUE)  
  
print(invalid\_edge\_summary)

# A tibble: 33 × 4  
 `Edge Type` source\_type target\_type n  
 <chr> <chr> <chr> <int>  
 1 LyricistOf MusicalGroup Song 106  
 2 RecordedBy RecordLabel Album 102  
 3 ProducerOf MusicalGroup Song 100  
 4 ComposerOf MusicalGroup Song 97  
 5 ProducerOf MusicalGroup Album 31  
 6 LyricistOf MusicalGroup Album 28  
 7 ComposerOf MusicalGroup Album 17  
 8 InStyleOf MusicalGroup MusicalGroup 12  
 9 InStyleOf Person MusicalGroup 11  
10 InterpolatesFrom MusicalGroup MusicalGroup 10  
# ℹ 23 more rows

# Check total invalid edge count  
cat("Total invalid edges:", sum(!mc1\_edges\_tagged$valid), "\n")

Total invalid edges: 570

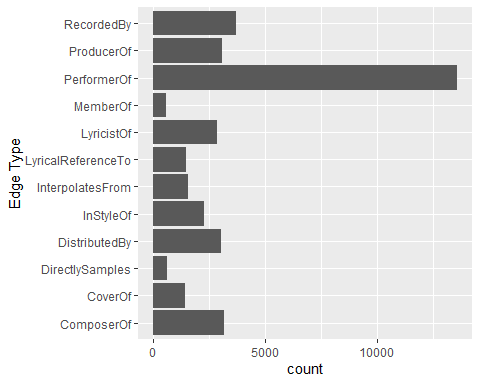
There were 570 invalid edges that are not in accordance to the data description file which are suggested to be removed as follows:

mc1\_edges\_clean <- mc1\_edges\_tagged %>%  
 filter(valid)

# **3  Exploratory Data Analysis**

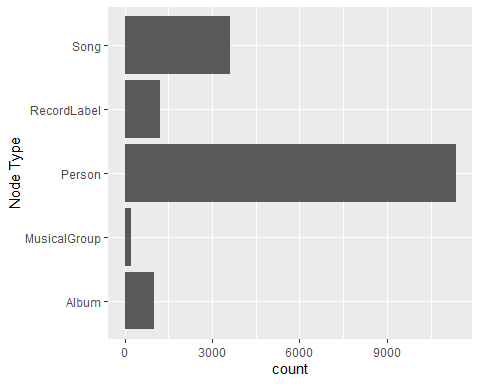
## **3.1 Visualising Edge Types**

ggplot(data = mc1\_edges\_clean,  
 aes(y = `Edge Type`)) +  
 geom\_bar()



## **3.2 Visualising Node Types**

ggplot(data = mc1\_nodes\_clean,  
 aes(y = `Node Type`)) +  
 geom\_bar()



## **3.3 Creating the global knowledge graph**

tbl\_graph() is used to create tidygraph’s graph object by using the code chunks below:

mc1\_nodes\_clean <- mc1\_nodes\_clean %>%  
 mutate(id = as.character(id))  
  
mc1\_edges\_clean <- mc1\_edges\_clean %>%  
 mutate(from\_id = as.character(from\_id),  
 to\_id = as.character(to\_id))

music <- tbl\_graph(  
 nodes = mc1\_nodes\_clean %>%  
 rename(label = name, # preserve the original artist/song name  
 name = id), # rename ID column to 'name' so tidygraph can use it  
 edges = mc1\_edges\_clean %>%  
 rename(from = from\_id, to = to\_id),  
 directed = TRUE  
)  
  
music %>%   
 activate(nodes) %>%  
 as\_tibble() %>%  
 select(name, label, `Node Type`) %>%  
 head()

# A tibble: 6 × 3  
 name label `Node Type`  
 <chr> <chr> <chr>   
1 0 Breaking These Chains Song   
2 1 Carlos Duffy Person   
3 2 Min Qin Person   
4 3 Xiuying Xie Person   
5 4 Nautical Mile Records RecordLabel  
6 5 Unshackled Heart Song

# **4  Exploring Sailor’s Network**

Who has Sailor been most influenced by over time?

## **4.1 Find Sailor’s ID and Works**

The code chunks filter to Sailor’s ID and her works. Sailor was was listed as a performer and/or lyricist 47 times and was a contributor to 38 unique works.

sailor\_id <- mc1\_nodes\_clean %>%  
 filter(is\_sailor) %>%  
 pull(id)  
  
sailor\_works <- mc1\_edges\_clean %>%  
 filter(  
 from\_id %in% sailor\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf")  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Count by role (edge type)  
mc1\_edges\_clean %>%  
 filter(  
 from\_id %in% sailor\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf")  
 ) %>%  
 count(`Edge Type`, sort = TRUE)

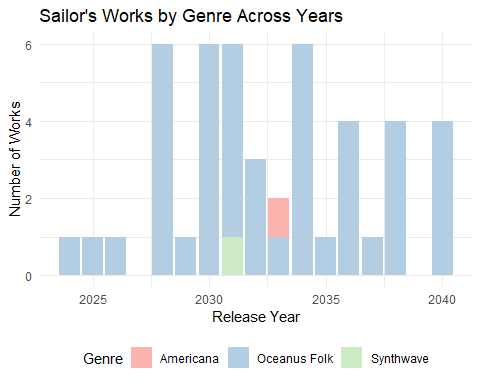
# A tibble: 2 × 2  
 `Edge Type` n  
 <chr> <int>  
1 PerformerOf 26  
2 LyricistOf 21

# Count of unique works Sailor contributed to  
mc1\_edges\_clean %>%  
 filter(  
 from\_id %in% sailor\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf")  
 ) %>%  
 summarise(unique\_works = n\_distinct(to\_id))

# A tibble: 1 × 1  
 unique\_works  
 <int>  
1 38

## 4.2 Summary Plot of Sailor’s work over years

### The plot



### The code

sailor\_works\_genre <- mc1\_edges\_clean %>%  
 filter(from\_id == sailor\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")) %>%  
 left\_join(mc1\_nodes\_clean, by = c("to\_id" = "id")) %>%  
 filter(`Node Type.y` %in% c("Song", "Album"),  
 !is.na(release\_date.y),  
 !is.na(genre.y)) %>%  
 group\_by(release\_date.y, genre.y) %>%  
 summarise(n = n(), .groups = "drop")  
  
ggplot(sailor\_works\_genre, aes(x = release\_date.y, y = n, fill = genre.y)) +  
 geom\_col() +  
 scale\_fill\_brewer(palette = "Pastel1") +  
 labs(  
 title = "Sailor's Works by Genre Across Years",  
 x = "Release Year",  
 y = "Number of Works",  
 fill = "Genre"  
 ) +  
 scale\_x\_continuous(breaks = pretty(sailor\_works\_genre$release\_date.y)) +  
 theme\_minimal() +  
 theme(legend.position = "bottom")

## 4.3 Find Works that influenced Sailor’s works

Works that influenced Sailor’s works is based on what was cited by in her works (InStyleOf”, “InterpolatesFrom”, “CoverOf”, “DirectlySamples”, “LyricalReferenceTo”). There were 26 songs/ album cited by Sailor’s works.

sailor\_referenced\_works <- mc1\_edges\_clean %>%  
 filter(  
 from\_id %in% sailor\_works,   
 `Edge Type` %in% c("InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples")  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Identify those referenced nodes (songs/albums Sailor referenced)  
influences <- mc1\_nodes\_clean %>%  
 filter(id %in% sailor\_referenced\_works)  
  
influences %>%  
 count(`Node Type`, sort = TRUE)

# A tibble: 2 × 2  
 `Node Type` n  
 <chr> <int>  
1 Song 23  
2 Album 3

## 4.4 Find Creators of Works that influenced Sailor’s works

Based on the code chunk below, we linked the work to the influencers to find out who were the influencers influencing Sailor.

There were 91 persons and 2 musical groups influencing Sailor.

# Extract their IDs to find who made those works  
influencer\_work\_ids <- influences$id  
  
# Who created those works?  
influencer\_people <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% influencer\_work\_ids,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, name, Node\_Type = `Node Type`, genre, release\_date),  
 by = c("from\_id" = "id") # use from\_id for join  
 ) %>%  
 filter(Node\_Type %in% c("Person", "MusicalGroup"))  
  
# Count top contributors who influenced Sailor  
influencer\_people %>%  
 count(name, sort = TRUE)

# A tibble: 93 × 2  
 name n  
 <chr> <int>  
 1 Joshua Taylor 3  
 2 Shannon Harvey 3  
 3 Urszula Stochmal 3  
 4 Debra Graham 2  
 5 Gang Shao 2  
 6 Igor Dyś 2  
 7 Jeremiah Love 2  
 8 Jing Kang 2  
 9 Lei Fu 2  
10 Lori Massey 2  
# ℹ 83 more rows

## 4.5 Check Sailor’s group membership and ID

A check was done to confirm if Sailor was in any musical group. It is confirmed that she only belonged to Ivy Echos.

# Get Sailor's ID  
sailor\_id <- mc1\_nodes\_clean %>%  
 filter(str\_to\_lower(name) == "sailor shift") %>%  
 pull(id)  
  
# Filter edges where Sailor is a member of a group  
sailor\_groups <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` == "MemberOf",  
 from\_id == sailor\_id  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, group\_name = name, node\_type = `Node Type`),  
 by = c("to\_id" = "id")  
 )  
  
sailor\_groups %>% select(group\_name, node\_type)

# A tibble: 1 × 2  
 group\_name node\_type   
 <chr> <chr>   
1 Ivy Echos MusicalGroup

sailor\_groups %>% select(group\_name, `Node Type`)

# A tibble: 1 × 2  
 group\_name `Node Type`  
 <chr> <chr>   
1 Ivy Echos Person

As Sailor could be influenced under Ivy Echos as she is a member. We obtained the musical group ID for Ivy Echos as follows:

sailor\_group\_ids <- mc1\_edges\_clean %>%  
 filter(`Edge Type` == "MemberOf", from\_id == sailor\_id) %>%  
 pull(to\_id)  
  
mc1\_edges\_clean %>%  
 filter(`Edge Type` == "MemberOf", from\_id == sailor\_id) %>%  
 left\_join(mc1\_nodes\_clean, by = c("to\_id" = "id")) %>%  
 select(to\_id, group\_name = name, node\_type = `Node Type.y`)

# A tibble: 1 × 3  
 to\_id group\_name node\_type   
 <chr> <chr> <chr>   
1 17260 Ivy Echos MusicalGroup

# **5  Building Subgraphs of Sailor’s influencers**

## 5.1 Define IDs to include in the subgraph

influencer\_people\_ids <- influencer\_people %>% pull(from\_id)  
referenced\_work\_ids <- sailor\_referenced\_works   
  
all\_ids <- unique(c(  
 sailor\_id,  
 sailor\_works,  
 referenced\_work\_ids,  
 influencer\_people\_ids,  
 sailor\_group\_ids  
))

## 5.2 Filter details to form the subgraph

However, we do not want to include collaborators except for the musical group that she belongs to. This is executed by the codes below.

We filter nodes to identify collaborators linked to Sailor.

# Filter nodes for Sailor + her works + influencers + cited works  
mini\_nodes <- mc1\_nodes\_clean %>%  
 filter(id %in% all\_ids) %>%  
 rename(label = name)  
  
# Identify collaborators (people who worked on Sailor's works)  
collaborator\_ids <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 from\_id != sailor\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
 ) %>%  
 pull(from\_id) %>%  
 unique()

We only keep collaborators and Ivy Echos if they influenced Sailor directly.

# Find Ivy Echos ID  
ivy\_id <- mc1\_nodes\_clean %>%  
 filter(str\_trim(name) == "Ivy Echos") %>%  
 pull(id)  
  
# Filter out collaborators unless they also influenced Sailor, but keep Ivy Echos  
mini\_nodes <- mini\_nodes %>%  
 filter(!(id %in% collaborator\_ids & !(id %in% influencer\_people\_ids | id == ivy\_id)))  
  
# Rebuild edges based on cleaned mini\_nodes  
valid\_ids <- mini\_nodes$id  
  
# Drop edges where the source is a removed collaborator and the target is a Sailor work  
mini\_edges <- mc1\_edges\_clean %>%  
 filter(from\_id %in% valid\_ids & to\_id %in% valid\_ids) %>%  
 filter(!(to\_id %in% sailor\_works & from\_id %in% collaborator\_ids & !(from\_id %in% influencer\_people\_ids | from\_id == ivy\_id)))  
  
# Remap node IDs to row indices for plotting  
id\_map <- tibble(id = valid\_ids, index = seq\_len(length(valid\_ids)))  
  
mini\_edges <- mini\_edges %>%  
 select(from\_id, to\_id, `Edge Type`, key) %>%  
 left\_join(id\_map %>% rename(from = index), by = c("from\_id" = "id")) %>%  
 left\_join(id\_map %>% rename(to = index), by = c("to\_id" = "id"))  
  
# Drop invalid edges  
mini\_edges\_clean <- mini\_edges %>%  
 filter(!is.na(from) & !is.na(to))

# **6  Building Visualisation**

## 6.1 Prepare Visualisation

We prepare the visualisation using the codes below.

# Tag Sailor and Ivy Echos  
mini\_nodes <- mini\_nodes %>%  
 mutate(  
 is\_sailor\_node = label == "Sailor Shift",  
 is\_ivy\_echos = str\_trim(label) == "Ivy Echos",  
 is\_sailor\_work = id %in% sailor\_works,  
   
# Define node category  
 node\_category = case\_when(  
 is\_sailor\_node ~ "Sailor Shift",  
 is\_ivy\_echos ~ "Ivy Echos",  
 is\_sailor\_work ~ "Sailor's Work",  
 TRUE ~ "Influencer"  
 ),  
   
# Final node color grouping  
 node\_color\_group = node\_category  
 )  
  
# Filter edges for plotting  
mini\_edges\_scored <- mini\_edges\_clean %>%  
 filter(`Edge Type` %in% c(  
 "InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples",  
 "PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf",  
 "MemberOf"  
 ))

## 6.2 Remove Sailor’s work if not linked to other works

# Add row index to mini\_nodes for alignment  
mini\_nodes <- mini\_nodes %>%  
 mutate(row\_index = row\_number())  
  
# Create the graph  
graph\_tbl <- tbl\_graph(nodes = mini\_nodes, edges = mini\_edges\_scored, directed = TRUE)  
  
# Identify Sailor's works (by ID)  
sailor\_work\_ids <- mini\_nodes %>%  
 filter(id %in% sailor\_works) %>%  
 pull(id)  
  
# Get corresponding row indices of Sailor's works  
sailor\_work\_rows <- mini\_nodes %>%  
 filter(id %in% sailor\_work\_ids) %>%  
 pull(row\_index)  
  
# Get Sailor's works which cite other works   
influence\_edge\_types <- c("InStyleOf", "CoverOf", "InterpolatesFrom", "DirectlySamples", "LyricalReferenceTo")  
  
referencing\_sailor\_works <- mini\_edges\_scored %>%  
 filter(  
 `Edge Type` %in% influence\_edge\_types,  
 from %in% sailor\_work\_rows   
 ) %>%  
 pull(from) %>%  
 unique()  
  
# Compute node degrees  
node\_degrees <- graph\_tbl %>%  
 activate(nodes) %>%  
 mutate(degree = centrality\_degree(mode = "all"),  
 node\_index = row\_number()) %>%  
 as\_tibble()  
  
# Identify works only connected to Sailor (no other collaborators)  
only\_sailor\_edges <- mini\_edges\_scored %>%  
 filter(  
 to %in% sailor\_work\_rows,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
 ) %>%  
 group\_by(to) %>%  
 summarise(all\_from\_sailor = all(from\_id == sailor\_id), .groups = "drop") %>%  
 filter(all\_from\_sailor) %>%  
 pull(to)   
  
# Final filtering  
sailor\_works\_to\_drop <- node\_degrees %>%  
 filter(  
 node\_index %in% only\_sailor\_edges,  
 !(node\_index %in% referencing\_sailor\_works)  
 ) %>%  
 pull(node\_index)

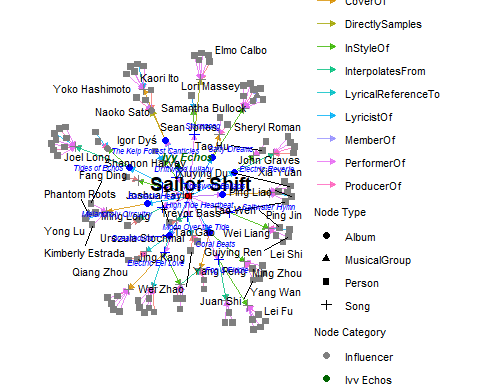
## 6.3 Remove Ivy Echo’s work if not linked to other works

ivy\_work\_ids <- mini\_edges\_scored %>%  
 filter(from\_id == ivy\_id, `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")) %>%  
 pull(to\_id) %>%  
 unique()  
  
ivy\_work\_rows <- mini\_nodes %>%  
 filter(id %in% ivy\_work\_ids) %>%  
 pull(row\_index)  
  
ivy\_influential\_edges <- mini\_edges\_scored %>%  
 filter(  
 (`Edge Type` %in% influence\_edge\_types) &  
 (from %in% ivy\_work\_rows | to %in% ivy\_work\_rows)  
 )  
  
ivy\_works\_to\_drop <- setdiff(ivy\_work\_rows, c(ivy\_influential\_edges$from, ivy\_influential\_edges$to))  
  
# Add to removal list  
nodes\_to\_drop <- unique(c(sailor\_works\_to\_drop, ivy\_works\_to\_drop))  
  
# Drop from graph  
graph\_tbl <- graph\_tbl %>%  
 activate(nodes) %>%  
 mutate(row\_index = row\_number()) %>%  
 filter(!row\_index %in% nodes\_to\_drop)

## 6.4 Visualising Sailor’s Influencers (non-interactive)

The visualisation is plotted as follows:

### The plot



### The code

set.seed(12)  
  
# Assign unique edge ID  
mini\_edges\_scored <- mini\_edges\_scored %>%  
 mutate(edge\_id = row\_number())  
  
# Plot with ggraph  
ggraph(graph\_tbl, layout = "fr") +  
   
 # Edge rendering  
 geom\_edge\_link0(  
 aes(color = `Edge Type`),  
 alpha = 0.8,  
 arrow = arrow(length = unit(1.5, 'mm'), type = "closed"),  
 end\_cap = circle(1.5, 'mm'),  
 lineend = "round"  
 ) +  
  
 # Node visuals  
 geom\_node\_point(aes(  
 color = node\_category,  
 shape = `Node Type`  
 ), size = 2) +  
  
 # Influencer names  
 geom\_node\_text(  
 aes(label = ifelse(id %in% influencer\_people\_ids, label, NA)),  
 size = 3,  
 repel = TRUE,  
 color = "black"  
 ) +  
  
 # Sailor  
 geom\_node\_text(  
 aes(label = ifelse(is\_sailor\_node, label, NA)),  
 size = 5,  
 fontface = "bold",  
 color = "black",  
 repel = TRUE  
 ) +  
  
 # Sailor's Works  
 geom\_node\_text(  
 aes(label = ifelse(node\_category == "Sailor's Work", label, NA)),  
 size = 2,  
 fontface = "italic",  
 color = "blue",  
 repel = TRUE  
 ) +  
   
 # Ivy Echos  
 geom\_node\_text(  
 aes(label = ifelse(str\_trim(label) == "Ivy Echos", label, NA)),  
 size = 3,  
 fontface = "bold.italic",  
 color = "darkgreen",  
 repel = TRUE  
 ) +  
   
 # Color and shape scales  
 scale\_color\_manual(  
 values = c(  
 "Influencer" = "grey50",  
 "Sailor Shift" = "red",  
 "Sailor's Work" = "blue",  
 "Ivy Echos" = "darkgreen"  
 ),  
 name = "Node Category"  
 ) +  
   
 scale\_shape\_manual(  
 values = c("Person" = 15, "MusicalGroup" = 17, "Album" = 16, "Song" = 3),  
 name = "Node Type"  
 ) +  
  
 theme\_void() +  
 theme(  
 plot.margin = margin(30, 30, 30, 30),   
 legend.title = element\_text(size = 8),  
 legend.text = element\_text(size = 8)  
 )

As there are many influencers and overlaps, it is suggested to do an interactive map instead.

## 6.5 Prepare Interactive Plot

# Identify Sailor node index  
sailor\_index <- graph\_tbl %>%  
 activate(nodes) %>%  
 mutate(node\_index = row\_number()) %>%  
 filter(is\_sailor\_node) %>%  
 pull(node\_index)  
  
# Pull all edges and check which nodes are only connected to Sailor  
edges\_tbl <- graph\_tbl %>%  
 activate(edges) %>%  
 as\_tibble()  
  
# Count how many unique nodes each target is connected to. We are only keen on degree = 1 and only connected to Sailor.  
solo\_sailor\_targets <- edges\_tbl %>%  
 filter(from == sailor\_index) %>%  
 pull(to)  
  
# Calculate degrees  
node\_degrees <- graph\_tbl %>%  
 activate(nodes) %>%  
 mutate(node\_index = row\_number(),  
 deg = centrality\_degree(mode = "all")) %>%  
 as\_tibble()  
  
# Find target nodes with degree = 1 and only connected to Sailor  
sailor\_solo\_works <- node\_degrees %>%  
 filter(node\_index %in% solo\_sailor\_targets, deg == 1) %>%  
 pull(node\_index)  
  
# Drop them from the graph  
graph\_tbl <- graph\_tbl %>%  
 activate(nodes) %>%  
 mutate(node\_index = row\_number()) %>% # safe if not changed above  
 filter(!row\_number() %in% sailor\_solo\_works)

## **6.6 Visualising Sailor’s Influencers (Interactive)**

### The plot

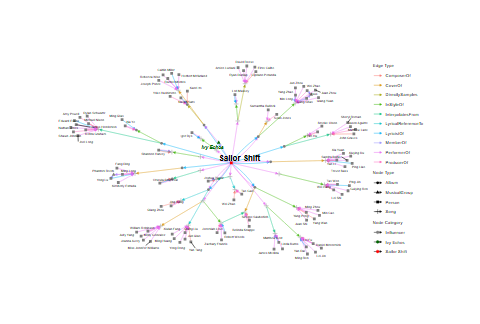
Warning: Removed 44 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 137 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 138 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 137 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

file:///C:/Users/Belinda/AppData/Local/Temp/Rtmps9glng/file53e836ee2921/widget53e859b45d61.html screenshot completed



### The code

set.seed(12)  
  
# Clean characters and create tooltip   
graph\_tbl <- graph\_tbl %>%  
 mutate(across(where(is.character), ~ gsub("'", "", .x))) %>%  
 mutate(  
 tooltip = paste0(  
 label,   
 " (", `Node Type`, ")",   
 ifelse(!is.na(release\_date), paste0("\nRelease Year: ", release\_date), "")  
 )  
 )  
  
# Create layout   
layout\_tbl <- create\_layout(graph\_tbl, layout = "fr")  
  
# Plot Interactive Graph  
g <- ggraph(layout\_tbl) +  
 geom\_edge\_fan(  
 aes(color = `Edge Type`),  
 alpha = 0.8,  
 arrow = arrow(length = unit(1.2, 'mm'), type = "closed"),  
 start\_cap = circle(1, 'mm'),  
 end\_cap = circle(1, 'mm'),  
 lineend = "round",  
 show.legend = TRUE  
 ) +  
 geom\_point\_interactive(  
 aes(  
 x = x,  
 y = y,  
 color = node\_category,  
 shape = `Node Type`,  
 tooltip = tooltip,  
 data\_id = label  
 ),  
 size = 2  
 ) +  
 geom\_node\_text(  
 aes(label = ifelse(id %in% influencer\_people\_ids, label, NA)),  
 size = 2,  
 repel = TRUE,  
 color = "black"  
 ) +  
 geom\_node\_text(  
 aes(label = ifelse(is\_sailor\_node, label, NA)),  
 size = 5,  
 fontface = "bold",  
 color = "black",  
 repel = TRUE  
 ) +  
 geom\_node\_text(  
 aes(label = ifelse(node\_category == "Sailor's Work", label, NA)),  
 size = 2,  
 fontface = "italic",  
 color = "blue",  
 repel = TRUE  
 ) +  
 geom\_node\_text(  
 aes(label = ifelse(label == "Ivy Echos", label, NA)),  
 size = 3,  
 fontface = "bold.italic",  
 color = "darkgreen",  
 repel = TRUE  
 ) +  
 scale\_color\_manual(  
 values = c(  
 "Influencer" = "grey50",  
 "Sailor Shift" = "red",  
 "Sailor's Work" = "blue",  
 "Ivy Echos" = "darkgreen"  
 ),  
 name = "Node Category"  
 ) +  
 scale\_shape\_manual(  
 values = c("Person" = 15, "MusicalGroup" = 17, "Album" = 16, "Song" = 3),  
 name = "Node Type"  
 ) +  
 theme\_void() +  
 theme(  
 plot.margin = margin(30, 30, 30, 30),  
 legend.title = element\_text(size = 8),  
 legend.text = element\_text(size = 8)  
 )  
  
girafe(ggobj = g, width\_svg = 10, height\_svg = 6)

# **7  Sailor’s Collaboration and Influence**

Who has Sailor collaborated with and directly or indirectly influenced how has she influenced collaborators of the broader Oceanus Folk community?

## **7.1 Find Sailor’s collaborators**

Find Sailor’s collaborators where she is a “PerformerOf”, “ComposerOf”, “ProducerOf”, “LyricistOf” and “Memberof”.

sailor\_collaborators <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf", "MemberOf"),  
 from\_id != sailor\_id  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>%   
 select(id, name, node\_type = `Node Type`),   
 by = c("from\_id" = "id")  
 ) %>%  
 distinct(from\_id, name, node\_type)

sailor\_collaborators %>%  
 count(node\_type, sort = TRUE)

# A tibble: 2 × 2  
 node\_type n  
 <chr> <int>  
1 Person 40  
2 MusicalGroup 8

sailor\_collaborators\_roles <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf"),  
 from\_id != sailor\_id  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, name, node\_type = `Node Type`),  
 by = c("from\_id" = "id")  
 ) %>%  
 distinct(from\_id, name, node\_type, `Edge Type`) %>%  
 arrange(name, `Edge Type`)  
   
 sailor\_collaborators\_roles

# A tibble: 51 × 4  
 from\_id name node\_type `Edge Type`  
 <chr> <chr> <chr> <chr>   
 1 17227 Aiden Harper Person LyricistOf   
 2 17041 Arlo Sterling Person LyricistOf   
 3 17190 Astrid Nørgaard Person ComposerOf   
 4 17349 Beatrice Albright Person LyricistOf   
 5 17349 Beatrice Albright Person PerformerOf  
 6 17045 Cassian Rae Person ComposerOf   
 7 17296 Coralia Bellweather Person ComposerOf   
 8 16996 Crimson Carriage MusicalGroup PerformerOf  
 9 17355 Daniel OConnell Person ComposerOf   
10 17355 Daniel OConnell Person PerformerOf  
# ℹ 41 more rows

There were 48 persons and musical groups whom collaborated with her.

## **7.2 Frequency of Collaborations**

sailor\_collaborator\_role\_frequency <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf"),  
 from\_id != sailor\_id  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, name),  
 by = c("from\_id" = "id")  
 ) %>%  
 count(name, `Edge Type`, sort = TRUE)   
  
sailor\_collaborator\_role\_frequency

# A tibble: 51 × 3  
 name `Edge Type` n  
 <chr> <chr> <int>  
 1 "Maya Jensen" ComposerOf 4  
 2 "Arlo Sterling" LyricistOf 3  
 3 "Cassian Rae" ComposerOf 3  
 4 "Elara May" ComposerOf 3  
 5 "Ivy Echos" PerformerOf 3  
 6 "Jade Thompson" ComposerOf 3  
 7 "Lila \"Lilly\" Hartman" ComposerOf 3  
 8 "Lyra Blaze" ComposerOf 3  
 9 "Orion Cruz" ComposerOf 3  
10 "Sophie Ramirez" LyricistOf 3  
# ℹ 41 more rows

Using the code chunk below, it is confirmed that there were 77 collaborations.

total\_collaborations <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf"),  
 from\_id != sailor\_id # exclude Sailor herself  
 ) %>%  
 nrow()  
  
total\_collaborations

[1] 77

## **7.2 Find works and musicians which were directly influenced by Sailor and Ivy Echos**

Filter by works that cited Sailor based on “InStyleOf”, “InterpolatesFrom”, “CoverOf”, “LyricalReferenceTo”, “DirectlySamples”

referencing\_works <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples")  
 ) %>%  
 pull(from\_id) %>%  
 unique()  
  
# Find people or groups who created those referencing works  
  
direct\_influencees <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% referencing\_works,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf")  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, name, node\_type = `Node Type`),  
 by = c("from\_id" = "id")  
 ) %>%  
 filter(node\_type %in% c("Person", "MusicalGroup")) %>%  
 distinct(from\_id, name, node\_type)

## **7.3 Sailor’s influence**

Find how often Sailor’s works were cited.

direct\_influence\_counts <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples")  
 ) %>%  
 left\_join(mc1\_nodes\_clean %>% select(id, name), by = c("from\_id" = "id")) %>%  
 count(name, sort = TRUE)  
  
direct\_influence\_counts

# A tibble: 0 × 2  
# ℹ 2 variables: name <chr>, n <int>

We can conclude that there was no Sailor’s work cited by others.

## **7.4 Sailor’s influence through Ivy Echos**

Let’s check if there was any Ivy Echo’s work cited by others using the code below.

# Get Ivy Echos' ID and works  
ivy\_id <- mc1\_nodes\_clean %>%  
 filter(str\_trim(name) == "Ivy Echos") %>%  
 pull(id)  
  
ivy\_works <- mc1\_edges\_clean %>%  
 filter(from\_id == ivy\_id,   
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")) %>%  
 pull(to\_id) %>%  
 unique()  
  
referencing\_works\_sailor <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% sailor\_works,  
 `Edge Type` %in% c("InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples")  
 ) %>%  
 pull(from\_id)  
  
referencing\_works\_ivy <- mc1\_edges\_clean %>%  
 filter(  
 to\_id %in% ivy\_works,  
 `Edge Type` %in% c("InStyleOf", "InterpolatesFrom", "CoverOf",   
 "LyricalReferenceTo", "DirectlySamples")  
 ) %>%  
 pull(from\_id)  
  
referencing\_works <- unique(c(referencing\_works\_sailor, referencing\_works\_ivy))  
  
mc1\_nodes\_clean %>%  
 filter(id %in% referencing\_works) %>%  
 select(id, name, `Node Type`, release\_date)

# A tibble: 1 × 4  
 id name `Node Type` release\_date  
 <chr> <chr> <chr> <int>  
1 17112 Embers & Echoes Album 2031

There was one album “Embers & Echoes” which was influenced by Ivy Echos’ work.

## **7.5 Musicians whom were directly influenced by Ivy Echos**

mc1\_edges\_clean %>%  
 filter(  
 to\_id == "17112", # ID of Embers & Echoes  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>% select(id, name, node\_type = `Node Type`),  
 by = c("from\_id" = "id")  
 ) %>%  
 filter(node\_type %in% c("Person", "MusicalGroup")) %>%  
 select(from\_id, name, node\_type) %>%  
 distinct()

# A tibble: 5 × 3  
 from\_id name node\_type   
 <chr> <chr> <chr>   
1 17110 The Hollow Monarchs MusicalGroup  
2 17402 N.V. Blake Person   
3 17406 Ivy Steele Person   
4 17407 Nate Wild Person   
5 17408 Scarlett Moon Person

Membership of The Hollow Monarchs musical group was checked to include in the plot.

# Get the ID of The Hollow Monarchs  
hollow\_monarchs\_id <- mc1\_nodes\_clean %>%  
 filter(str\_trim(name) == "The Hollow Monarchs") %>%  
 pull(id)  
  
# Find people who have a MemberOf edge pointing to the group  
hollow\_monarchs\_members <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` == "MemberOf",  
 to\_id == hollow\_monarchs\_id  
 ) %>%  
 left\_join(mc1\_nodes\_clean %>% select(id, member\_name = name, member\_type = `Node Type`),  
 by = c("from\_id" = "id")) %>%  
 filter(member\_type == "Person") %>%  
 select(from\_id, member\_name)  
  
hollow\_monarchs\_members

# A tibble: 3 × 2  
 from\_id member\_name   
 <chr> <chr>   
1 17107 Chloe Montgomery  
2 17108 Nathaniel Brooks  
3 17109 Clara Davis

## **7.6 Musicians whom were indirectly influenced by Sailor/ Ivy Echos**

First, we found the works that were influenced by Embers & Echoes album to trace if anyone had cited this work.

# Get the ID of "Embers & Echoes"  
embers\_id <- mc1\_nodes\_clean %>%  
 filter(str\_trim(name) == "Embers & Echoes") %>%  
 pull(id) %>%  
 as.character()  
  
# Check if any works cited Embers & Echoes using influence-type edges  
works\_citing\_embers <- mc1\_edges\_clean %>%  
 filter(  
 to\_id == embers\_id,  
 `Edge Type` %in% c("InStyleOf", "CoverOf", "InterpolatesFrom",  
 "DirectlySamples", "LyricalReferenceTo")  
 ) %>%  
 pull(from\_id) %>%  
 unique()  
  
# Show the works that cited it  
mc1\_nodes\_clean %>%  
 filter(id %in% works\_citing\_embers) %>%  
 select(id, name, `Node Type`, release\_date)

# A tibble: 0 × 4  
# ℹ 4 variables: id <chr>, name <chr>, Node Type <chr>, release\_date <int>

We confirmed that no other works cited Ember & Echoes.Hence, no one had been indirectly influenced by Sailor / Ivy Echos.

## **7.7 Plot Preparation**

We begin by filtering Sailor’s work that were cited by others and had collaboration:

# Identify works of Sailor that were cited  
sailor\_works\_cited <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` %in% c("InStyleOf", "CoverOf", "InterpolatesFrom", "DirectlySamples", "LyricalReferenceTo"),  
 to\_id %in% sailor\_works  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Identify works Sailor collaborated on (other people involved)  
sailor\_works\_collab <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf"),  
 to\_id %in% sailor\_works,  
 from\_id != sailor\_id  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Filter only the Sailor works that are either cited or collaborations  
sailor\_works\_filtered <- intersect(sailor\_works, union(sailor\_works\_cited, sailor\_works\_collab))

Filter away Ivy Echos’ work if it is not cited by others

# Ivy Echos works cited by others  
ivy\_works\_cited <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` %in% c("InStyleOf", "CoverOf", "InterpolatesFrom", "DirectlySamples", "LyricalReferenceTo"),  
 to\_id %in% ivy\_works  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Ivy Echos works with collaboration  
ivy\_works\_collab <- mc1\_edges\_clean %>%  
 filter(  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "ProducerOf", "LyricistOf"),  
 to\_id %in% ivy\_works,  
 from\_id != ivy\_id  
 ) %>%  
 pull(to\_id) %>%  
 unique()  
  
# Filter Ivy Echos works that are either cited or collaborative  
ivy\_works\_filtered <- intersect(ivy\_works, union(ivy\_works\_cited, ivy\_works\_collab))

Add in Ember & Echoes’s contributors as they were directly influenced by Ivy Echos:

embers\_contributors <- mc1\_edges\_clean %>%  
 filter(  
 to\_id == embers\_id,  
 `Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
 ) %>%  
 left\_join(  
 mc1\_nodes\_clean %>%  
 select(id, name, node\_type = `Node Type`),  
 by = c("from\_id" = "id")  
 ) %>%  
 filter(node\_type %in% c("Person", "MusicalGroup")) %>%  
 distinct(from\_id, name, node\_type)

# Combine node IDs  
all\_ids <- unique(c(  
 sailor\_id,  
 ivy\_id,   
 sailor\_works\_filtered,  
 ivy\_works\_filtered,  
 sailor\_collaborators$from\_id,  
 referencing\_works,  
 direct\_influencees$from\_id,  
 embers\_contributors$from\_id   
))  
  
# Filter node table  
mini\_nodes <- mc1\_nodes\_clean %>%  
 filter(id %in% all\_ids) %>%  
 mutate(  
 label = name,  
 is\_sailor = id == sailor\_id,  
 is\_ivy = id == ivy\_id,  
 is\_work = id %in% sailor\_works\_filtered,  
 is\_ivy\_work = id %in% ivy\_works\_filtered,  
 is\_collaborator = id %in% sailor\_collaborators$from\_id,  
 is\_direct\_influencee = id %in% direct\_influencees$from\_id,  
 is\_embers\_contributor = id %in% embers\_contributors$from\_id,   
 category = case\_when(  
 is\_sailor ~ "Sailor Shift",  
 is\_ivy ~ "Ivy Echos",  
 is\_work ~ "Sailor's Work",  
 is\_ivy\_work ~ "Ivy's Work",  
 is\_collaborator ~ "Collaborator",  
 is\_direct\_influencee ~ "Direct Influencee",  
 is\_embers\_contributor ~ "Embers Contributor",   
 TRUE ~ "Other"  
 )  
 )  
  
# Filter edges between valid nodes  
mini\_edges <- mc1\_edges\_clean %>%  
 filter(from\_id %in% all\_ids, to\_id %in% all\_ids) %>%  
 select(from = from\_id, to = to\_id, `Edge Type`) %>%  
 mutate(from = as.character(from), to = as.character(to))  
  
# Build tidygraph object  
graph\_tbl <- tbl\_graph(  
 nodes = mini\_nodes %>% mutate(name = as.character(id)),   
 edges = mini\_edges %>% mutate(  
 from = as.character(from),  
 to = as.character(to)  
 ),  
 directed = TRUE  
)  
  
# Add tooltip  
graph\_tbl <- graph\_tbl %>%  
 mutate(tooltip = paste0(label, " (", `Node Type`, ")"))

## **7.8 Plot interactive graph**

The visualisation showing who has Sailor collaborated with and directly or indirectly influenced is shown below.

### The plot

Warning: Removed 75 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 56 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

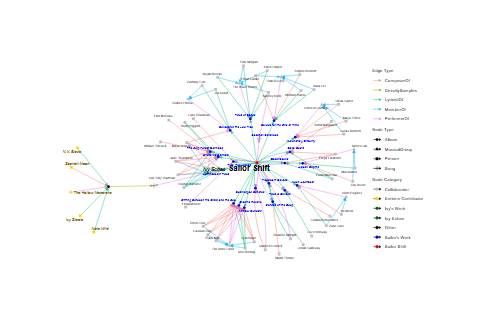
Warning: Removed 29 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 76 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 75 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

Warning: Removed 71 rows containing missing values or values outside the scale range  
(`geom\_text\_repel()`).

file:///C:/Users/Belinda/AppData/Local/Temp/Rtmps9glng/file53e84e56170d/widget53e83a05431.html screenshot completed



### The code

set.seed(1111)  
layout\_tbl <- create\_layout(graph\_tbl, layout = "stress")  
  
graph\_tbl <- graph\_tbl %>%  
 mutate(across(where(is.character), ~ gsub("'", "", .x))) %>%  
 mutate(  
 tooltip = paste0(  
 label,   
 " (", `Node Type`, ")",   
 ifelse(!is.na(release\_date), paste0("\nRelease Year: ", release\_date), ""),  
 ifelse(!is.na(genre), paste0("\nGenre: ", genre), "")  
 )  
 )  
  
# Define graph  
g <- ggraph(layout\_tbl) +  
   
 # Edge rendering  
 geom\_edge\_fan(  
 aes(color = `Edge Type`),  
 alpha = 0.8,  
 arrow = arrow(length = unit(1.2, 'mm'), type = "closed"),  
 start\_cap = circle(1, 'mm'),  
 end\_cap = circle(1, 'mm'),  
 lineend = "round",  
 show.legend = TRUE  
 ) +  
   
 # Nodes with interactivity  
 geom\_point\_interactive(  
 aes(  
 x = x,  
 y = y,  
 color = category,  
 shape = `Node Type`,  
 tooltip = tooltip,  
 data\_id = label  
 ),  
 size = 2  
 ) +  
  
 # Labels for key nodes  
 geom\_node\_text(  
 aes(label = ifelse(category == "Sailor Shift", label, NA)),  
 size = 5,  
 fontface = "bold",  
 color = "black",  
 repel = TRUE  
 ) +  
   
 geom\_node\_text(  
 aes(label = ifelse(category == "Sailor's Work", label, NA)),  
 size = 2,  
 fontface = "italic",  
 color = "blue",  
 repel = TRUE  
 ) +  
  
 geom\_node\_text(  
 aes(label = ifelse(category == "Collaborator", label, NA)),  
 size = 2,  
 color = "black",  
 repel = TRUE  
 ) +  
  
 geom\_node\_text(  
 aes(label = ifelse(category == "Direct Influencee", label, NA)),  
 size = 3,  
 color = "black",  
 repel = TRUE  
 ) +  
  
 geom\_node\_text(  
 aes(label = ifelse(label == "Ivy Echos", label, NA)),  
 size = 3,  
 fontface = "bold.italic",  
 color = "darkgreen",  
 repel = TRUE  
 ) +  
  
 geom\_node\_text(  
 aes(label = ifelse(category == "Embers Contributor", label, NA)),  
 size = 2.5,  
 fontface = "italic",  
 color = "goldenrod4",  
 repel = TRUE  
 ) +  
  
 # Color and shape scales  
scale\_color\_manual(  
 values = c(  
 "Sailor Shift" = "red",  
 "Ivy Echos" = "darkgreen",  
 "Sailor's Work" = "blue",  
 "Ivy's Work" = "darkgreen",  
 "Collaborator" = "grey",  
 "Direct Influencee" = "orange",  
 "Embers Contributor" = "gold",   
 "Other" = "black"  
 ),  
 name = "Node Category"  
) +  
  
 scale\_shape\_manual(  
 values = c("Person" = 15, "MusicalGroup" = 17, "Album" = 16, "Song" = 3),  
 name = "Node Type"  
 ) +  
  
 theme\_void() +  
 theme(  
 plot.margin = margin(30, 30, 30, 30),  
 legend.title = element\_text(size = 8),  
 legend.text = element\_text(size = 8)  
 )  
  
# Wrap in interactive output  
girafe(ggobj = g, width\_svg = 10, height\_svg = 6)