An Analysis of On-time Performance and Cancellation Rates across Flights in Australia from 2020 to 2023

by

Kopty, Zian
Lanzona, Banjan
Luo, Wendi
Tran, Ricky

A business analytics report submitted in partial fulfilment
of the requirements of the unit
Programming in Business
University of Western Australia Business School
University of Western Australia

Submitted to:
Tristan Reed

Date of Submission:
May 28, 2024

Table of Contents

I. Introduction

The Australian air transport industry plays a vital role in connecting metropolitan cities and towns across the country. The industry relies heavily on operational efficiency and streamlined functions to provide dependable services to its customers. Our report analyses Bureau of Infrastructure and Transport Research Economics (BITRE) data containing extensive flight records to uncover actionable insights that airlines in Australia can leverage in streamlining their operations and boosting profitability.

The BITRE dataset contains information regarding departure and arrival times, deferments, and cancellations across various routes and airlines operating in Australia. Leveraging this data, airlines can gain insight into key patterns and trends that they could use to identify operational bottlenecks and recognise underperforming routes that should be addressed.

In this report, we conduct an extensive analysis of the BITRE dataset, mainly covering the period from 2020 to 2023. Our objective is two-fold: first to uncover underlying trends and patterns concerning punctuality and cancellations of flights across key routes and airports in Australia. Second, to review our findings and derive actionable insights.

We undertake detailed data cleaning and manipulation to aid our analysis. We also leverage Python code to derive important conclusions from the data. Besides quantitative analysis, it also utilises various visuals to bridge the gap between raw data and actionable insights. The visuals allow us to highlight trends and patterns that may not be apparent with numerical data and portray complex data into easily interpretable formations to aid comprehension and decision-making.

II. Data cleaning

The section outlines the statistical techniques and data-cleaning methods deployed to prepare the dataset for deeper analysis. Having clean and well-formatted data is fundamental for laying a solid foundation for our analysis and ensuring that our findings are accurate and reliable. The pre-analysis phase comprises three stages.

- Package Installation
- Importing Data
- Data cleaning
- Data transformation

Package Installation

Before any analysis, importing and storing all the necessary Python packages is vital to facilitate data manipulation, computation, and visualisation. For the scope of this report, we relied primarily on the following Python packages *(See Appendix H.1.),*

- **NumPy, SciPy, and Math** packages were deployed due to their robust ability to manipulate data and perform complex mathematical computations.
- **Pandas** helped prepare and transform the data into a structure better aligned with the scope of our analysis.
- **Seaborn, Matplotlib, and Plotly** are data visualisation packages used to create graphs that helped us communicate complex data in a manner that was easy to comprehend.
- **Geopy** and **Shapely** are additional packages we have utilised to extract and transform geospatial data of the ports, particularly using the "*nominatim " geolocator* to access the Open Street Map database.

Importing data

Once the necessary packages were installed, we proceeded to import the dataset. We leveraged the Pandas package to extract data from the Excel file for our analysis. Additionally, we compiled data from the different sheets combining flight records from 2010 to 2023 into one spreadsheet to ensure all the data was readily available for analysis. Such a strategy allows us to implement more efficient data manipulation and cleaning processes. We also maintained copies of each of the individual sheets in case they are required *(See Appendix H.1. for the code)*.

Data Cleaning

We removed the newline characters (\n) and parenthesis from the column names to achieve a cleaner code. Having more convenient column names allows us to simplify subsequent operations and minimises the risk of errors concerning naming conventions during data processing. In addition, we identified some records having "Virgin Australia" as their Airline value in the 2023 portion of the latest dataset. We have reconciled these with the proper label "Virgin Australia" to be able to conduct accurate aggregation with records of the same airline.

We then shifted our focus towards eliminating the null values. Removing null values was particularly important to maintain overall data integrity. We removed every row labelled "All Ports" to avoid our data being skewed as their high values could disproportionately influence analysis and aggregation resulting in misleading conclusions. Similarly, we removed a set of rows from the 2012 sheet that were comments from the original file that were irrelevant to our analysis.

The original dataset included string 'na' values to indicate non-computable entries. As string values hinder mathematical operations leading to potential errors, we replaced the 'na' values with blank cells to facilitate smoother computations ensuring the dataset comprised solely numeric data. Once the initial data cleaning processes were completed, we revalidated each column to ensure consistency with the appropriate data type. Such a step was vital in ensuring that succeeding data analysis would avoid errors concerning data types. For example, we treated the percentage column as numeric to enable arithmetic calculations *(See Appendix H.2. for the code)*.

Data Transformation

To further enhance our analysis, we created individual columns for years, months, and days based on the original dates column. The transformation enabled us to perform a more detailed time-series analysis by helping us identify trends across different units of time. Furthermore, it helps us examine seasonal variations in performance across multiple years. Additionally, we created a new column Route ID as the index to aggregate the one-way routes. The addition allowed us to analyse two-way routes and treat route Airport A to Airport B as identical to route Airport B to Airport A.

We assigned each year its data frame to facilitate an independent, year-by-year analysis and data comparison. The split also eliminated any irrelevant rows maintaining a tidy and structured dataset. Our detailed and comprehensive data cleaning and processing strategy ensures that our analysis has been built on a solid foundation of reliable and accurate data *(See Appendix H.3. for the code)*.

In addition, we extracted latitude and longitude data for all the departure and arrival ports. The information aided us in visualising the trends in the metrics on a map-based graph *(See Appendix H.4. for the code)*.

Next, we will move to the results section where we will deep-dive into the results of our analysis.

III. Result and Analysis

On-time Performance of One-way Routes by Year

The section below discusses the yearly on-time performance of local Australian flights from 2020 to 2023. Key metrics include the yearly proportion of on-time arrivals and departures relative to flights flown for each route. Such routes are considered independent and one-way, treating the route from Airport A to Airport B differently from Airport B to Airport A *(See Appendix H.5. for the code)*.

Table 3.1.1. The Best and Worst On-time Arrival and Departure Performance of One-way Routes from 2020 to 2023

2020

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Alice Springs-Darwin	95.5%	**Proserpine-Sydney**	56.5%	**Alice Springs-Darwin**	95.5%	**Proserpine-Sydney**	54.3%
Cairns-Townsville	93.3%	**Sunshine Coast-Sydney**	66.1%	**Cairns-Townsville**	93.1%	**Sydney-Proserpine**	54.3%
Brisbane-Emerald	92.6%	**Sunshine Coast-Melbourne**	67.5%	**Brisbane-Emerald**	92.3%	**Gold Coast-Melbourne**	60.0%
Emerald-Brisbane	91.5%	**Hobart-Melbourne**	67.9%	**Perth-Darwin**	92.3%	**Sunshine Coast-Melbourne**	64.8%
Adelaide-Canberra	91.4%	**Darwin-Brisbane**	68.3%	**Perth-Port Hedland**	92.1%	**Gold Coast-Sydney**	65.8%

2021

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Emerald-Brisbane	93.2%	**Perth-Melbourne**	71.3%	**Brisbane-Mount Isa**	94.9%	**Darwin-Brisbane**	68.1%
Brisbane-Emerald	92.9%	**Darwin-Brisbane**	72.0%	**Brisbane-Gladstone**	94.0%	**Perth-Melbourne**	70.8%
Adelaide-Gold Coast	92.2%	**Perth-Brisbane**	74.8%	**Brisbane-Hamilton Island**	93.2%	**Alice Springs-Adelaide**	72.7%
Brisbane-Gladstone	92.0%	**Port Hedland-Perth**	75.1%	**Brisbane-Emerald**	92.9%	**Darwin-Melbourne**	74.9%
Townsville-Cairns	91.8%	**Darwin-Perth**	75.8%	**Emerald-Brisbane**	92.6%	**Perth-Sydney**	75.6%

2022

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Cairns-Townsville	87.4%	**Darwin-Melbourne**	44.6%	**Cairns-Townsville**	86.6%	**Darwin-Sydney**	38.7%
Brisbane-Hamilton Island	84.3%	**Darwin-Sydney**	47.9%	**Brisbane-Hamilton Island**	85.1%	**Darwin-Melbourne**	41.6%
Brisbane-Emerald	83.5%	**Sydney-Perth**	48.9%	**Adelaide-Alice Springs**	83.8%	**Sydney-Perth**	47.6%
Townsville-Cairns	83.4%	**Darwin-Perth**	52.9%	**Brisbane-Emerald**	83.5%	**Darwin-Perth**	47.9%
Perth-Newman	81.0%	**Sunshine Coast-Melbourne**	54.3%	**Adelaide-Gold Coast**	82.2%	**Melbourne-Perth**	50.7%

2023

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Townsville-Cairns	88.3%	**Hamilton Island-Sydney**	52.4%	**Cairns-Townsville**	88.1%	**Launceston-Brisbane**	49.9%
Cairns-Townsville	87.9%	**Melbourne-Darwin**	55.8%	**Townsville-Cairns**	87.2%	**Hamilton Island-Sydney**	53.6%
Brisbane-Mount Isa	85.9%	**Launceston-Brisbane**	57.2%	**Adelaide-Canberra**	85.4%	**Darwin-Melbourne**	56.4%
Adelaide-Canberra	84.1%	**Sunshine Coast-Melbourne**	57.2%	**Brisbane-Mount Isa**	84.9%	**Sydney-Darwin**	56.5%
Adelaide-Gold Coast	83.7%	**Sunshine Coast-Sydney**	58.8%	**Adelaide-Port Lincoln**	84.6%	**Melbourne-Darwin**	57.4%

Table 3.1.1 demonstrates the five best and worst-performing routes regarding on-time arrival and departure performance between 2020 and 2023. During the four years, 2020 recorded the highest on-time performance for both arrival and departure. The 2020 flights from Alice Springs to Darwin, in particular, delivered customers the most punctual service with 95.5% of the flights departing and arriving on time. In contrast, 2022 reported the worst performance, mainly due to the rapid COVID-19 outbreak during the year that infected more than 10 million Australians (Briggs, 2022). Notably, the gap between the best and worst was the smallest in 2021 with a 21.9% difference.

In terms of performance consistency, flights from Townsville to Cairns and vice-versa have continually performed best across the years. Brisbane airport, in addition, is one of the most reliable airports having the most flights departing from this airport included in the top lists. On the other hand, passengers observed a consistently high number of delays in Sunshine Coast-Sydney and Darwin-Melbourne flights.

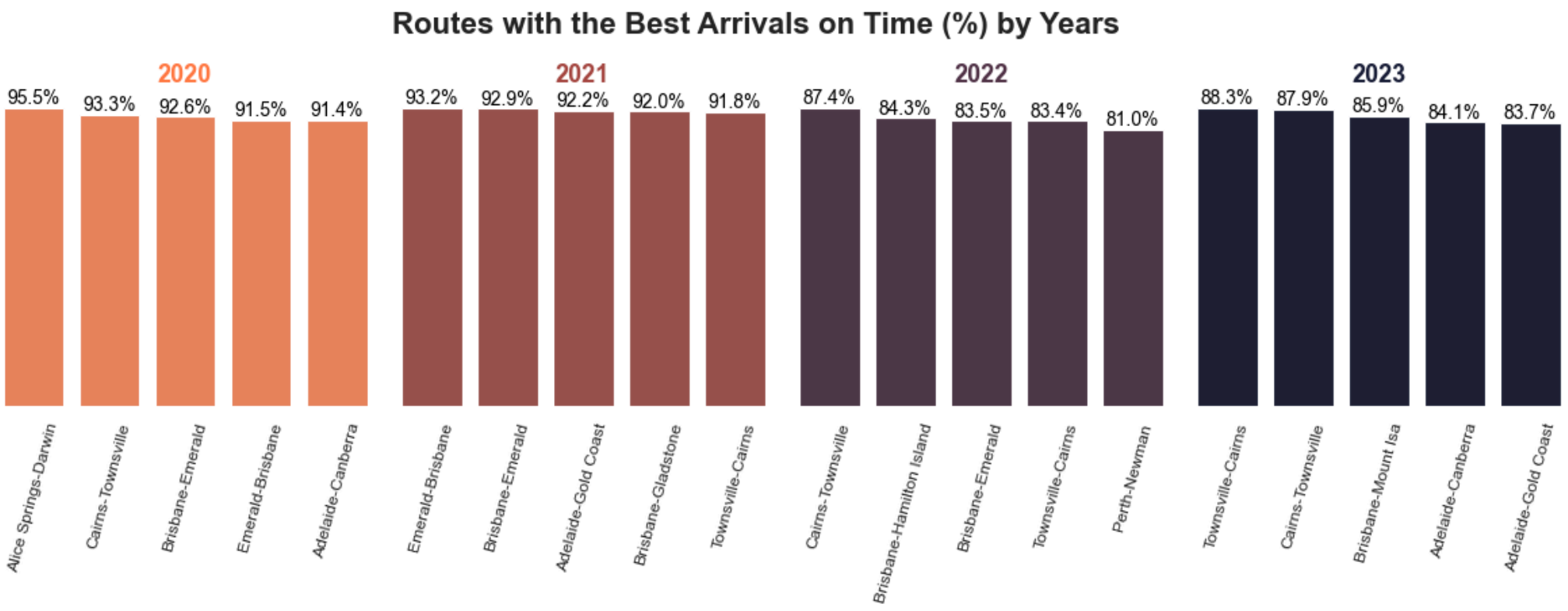

Figure 3.1.2. The Best On-time Departure Performances of One-way Routes 2020-2023

Figure 3.1.2 illustrates the ranking within a year and facilitates comparison between different years where possible. Although the difference is arguably marginal, a comparison analysis using the bar chart can answer the question of which route performed superior or inferior. The choice of different colours aids readers in easily identifying what group of routes belong to the same year *(See Appendix A for the other years and H.9. for visualisation codes)*.

On-time Performance of Aggregated Routes by Year

The section below discusses the yearly on-time performance of local Australian flights from 2020 to 2023. Key metrics include the yearly proportion of on-time arrivals and departures relative to flights flown for each route. Such routes are considered aggregated, treating routes from port A to port B as identical to those from port B to port A *(See Appendix H.6. for the code)*.

Table 3.2.1. The Best and Worst On-time Arrival and Departure Performance of Two-way Routes 2020-2023

2020

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Cairns-Townsville	92.2%	**Proserpine-Sydney**	63.0%	**Alice Springs-Darwin**	93.2%	**Proserpine-Sydney**	54.3%
Brisbane-Emerald	92.0%	**Gold Coast-Melbourne**	69.4%	**Brisbane-Emerald**	92.1%	**Gold Coast-Melbourne**	68.3%
Kalgoorlie-Perth	89.4%	**Sydney-Tamworth**	69.7%	**Cairns-Townsville**	91.4%	**Sunshine Coast-Sydney**	69.4%
Alice Springs-Darwin	88.6%	**Sunshine Coast-Sydney**	70.0%	**Kalgoorlie-Perth**	89.5%	**Gold Coast-Sydney**	70.7%
Brisbane-Hamilton Island	88.6%	**Hobart-Melbourne**	72.2%	**Adelaide-Canberra**	88.9%	**Sydney-Tamworth**	70.8%

2021

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Brisbane-Emerald	93.0%	**Melbourne-Perth**	75.0%	**Brisbane-Emerald**	92.8%	**Melbourne-Perth**	75.6%
Adelaide-Gold Coast	91.7%	**Brisbane-Perth**	76.0%	**Brisbane-Gladstone**	92.3%	**Perth-Sydney**	76.1%
Albury-Sydney	91.3%	**Brisbane-Darwin**	76.2%	**Brisbane-Mount Isa**	92.3%	**Brisbane-Darwin**	78.8%
Cairns-Townsville	91.3%	**Perth-Sydney**	78.0%	**Adelaide-Gold Coast**	90.3%	**Brisbane-Perth**	78.9%
Brisbane-Mount Isa	90.8%	**Sunshine Coast-Sydney**	79.6%	**Cairns-Townsville**	89.7%	**Adelaide-Alice Springs**	79.7%

2022

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Cairns-Townsville	85.4%	**Darwin-Melbourne**	50.9%	**Cairns-Townsville**	84.0%	**Darwin-Melbourne**	46.6%
Brisbane-Emerald	82.1%	**Darwin-Sydney**	53.8%	**Brisbane-Emerald**	81.4%	**Darwin-Sydney**	48.3%
Brisbane-Hamilton Island	79.0%	**Perth-Sydney**	57.3%	**Adelaide-Port Lincoln**	80.6%	**Perth-Sydney**	50.3%
Adelaide-Port Lincoln	78.8%	**Launceston-Melbourne**	58.0%	**Adelaide-Canberra**	77.1%	**Melbourne-Perth**	54.0%
Brisbane-Mount Isa	76.8%	**Canberra-Gold Coast**	58.1%	**Brisbane-Mount Isa**	76.8%	**Darwin-Perth**	55.0%

2023

Best Arrival		Worst Arrival		Best Departure		Worst Departure	
Route	**On-time Arrival %**	**Route**	**On-time Arrival %**	**Route**	**On-time Departure %**	**Route**	**On-time Departure %**
Cairns-Townsville	88.1%	**Hamilton Island-Sydney**	60.0%	**Cairns-Townsville**	87.7%	**Brisbane-Launceston**	56.5%
Brisbane-Mount Isa	83.2%	**Darwin-Melbourne**	60.0%	**Adelaide-Port Lincoln**	84.3%	**Darwin-Melbourne**	56.9%
Adelaide-Port Lincoln	82.8%	**Hobart-Melbourne**	60.7%	**Adelaide-Canberra**	83.4%	**Perth-Sydney**	59.7%
Brisbane-Emerald	81.1%	**Brisbane-Launceston**	61.1%	**Brisbane-Mount Isa**	82.6%	**Hamilton Island-Sydney**	60.5%
Adelaide-Canberra	80.7%	**Sunshine Coast-Sydney**	64.0%	**Brisbane-Emerald**	80.6%	**Darwin-Sydney**	61.3%

In general, the list of the best-performing routes experienced a slight decrease in performance across the four-year period, in contrast to the slight improvement of the least-performing routes. The contrast results in narrower differences between the best and the worst. The observable trend is due to the aggregation of the performance of both directions. Compared to the one-way routes, the lists observed notable changes. For instance, the merger of the routes that have both directions performing well maintained their ranks for the best on-time performance in both arrival and departure, such as flights between Brisbane and Emerald from 2020 to 2021 and flights between Cairns and Townsville in 2022 and 2023. However, the underperforming routes in terms of departure list in 2022 observed no changes except in the ranking orders. Similar to the previous section, the Cairns-Townsville and Brisbane-Emerald routes were consistently listed as top performers, delivering high-quality service throughout the years. In contrast, from 2022 to 2023, the Darwin-Melbourne, Darwin-Sydney, and Sydney-Perth routes recorded record-breaking proportions of delays with fewer than 50% of flights on time. Such results further support the notion that airports of major capital cities, such as Darwin, Melbourne and Sydney, experienced the most significant impact from the 2022 COVID-19 outbreak.

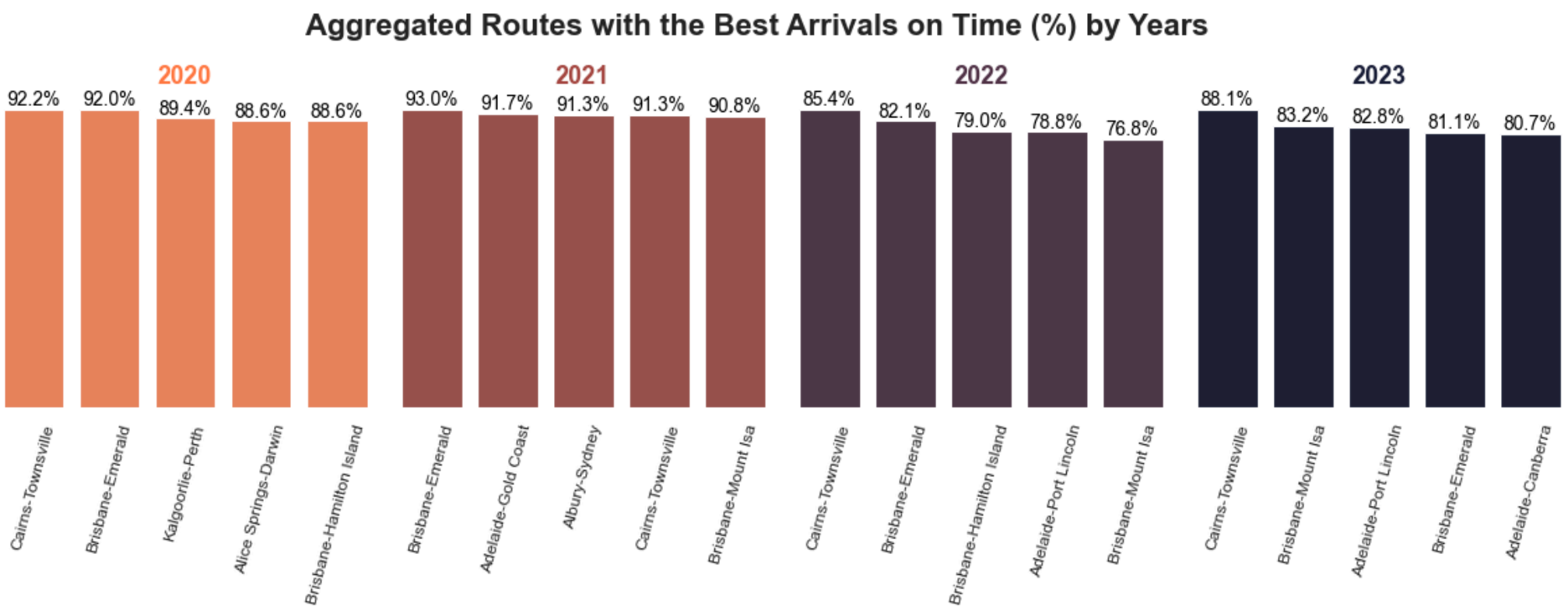

Figure 3.2.2. The Best On-time Arrival Performance of Two-way Routes 2020-2023

Likewise, the choice of visualisation above is similar to the previous section. Applying the same colour palette and chart type, the report maintains consistency and familiarity and will help readers easily grasp the meaning of the x and y values and the associated year *(See Appendix B for the other years and H.9. for visualisation codes)*.

On-time Performance of High-traffic Routes by Year

Table 3.3.1. The Best and Worst On-time Departure Performance of High-traffic One-way Routes 2020-2023

2020

Best departure

Route	On-time Departure %
Alice Springs-Darwin	95.5%
Adelaide-Port Lincoln	92.8%
Brisbane-Emerald	92.1%
Cairns-Townsville	91.9%
Perth-Darwin	91.6%

Worst departure

Route	On-time Departure %
Sydney-Proserpine	51.0%
Proserpine-Sydney	56.9%
Gold Coast-Melbourne	58.7%
Townsville-Sydney	64.0%
Gold Coast-Sydney	64.2%

2021

Best departure

Route	On-time Departure %
Brisbane-Mount Isa	94.9%
Brisbane-Gladstone	92.9%
Brisbane-Hamilton Island	92.8%
Adelaide-Gold Coast	92.1%
Port Macquarie-Sydney	92.1%

Worst departure

Route	On-time Departure %
Darwin-Brisbane	68.1%
Alice Springs-Adelaide	68.6%
Melbourne-Mildura	73.0%
Perth-Melbourne	73.1%
Sydney-Perth	73.7%

2022

Best departure

Route	On-time Departure %
Cairns-Townsville	85.9%
Brisbane-Hamilton Island	85.3%
Brisbane-Emerald	83.1%
Adelaide-Canberra	82.8%
Adelaide-Alice Springs	82.4%

Worst departure

Route	On-time Departure %
Darwin-Sydney	38.7%
Darwin-Melbourne	41.6%
Darwin-Perth	42.4%
Sydney-Perth	48.3%
Melbourne-Perth	51.1%

2023

Best departure

Route	On-time Departure %
Cairns-Townsville	88.7%
Townsville-Cairns	86.8%
Adelaide-Canberra	85.8%
Port Lincoln-Adelaide	85.8%
Adelaide-Port Lincoln	85.2%

Worst departure

Route	On-time Departure %
Launceston-Brisbane	50.9%
Hamilton Island-Sydney	53.6%
Darwin-Melbourne	55.6%
Sydney-Darwin	56.5%
Melbourne-Darwin	57.4%

The table above comprises the best and worst on-time departure percentages above the 75th percentile between 2020 and 2023 *(See Appendix H.7. for the code)*. The analysis of this data reveals a few interesting trends in the overall performance of different Australian airports. Routes such as Cairns-Townsville and those involving Adelaide have consistently performed well across the four years, indicating consistent operational efficiency and air traffic management. On the other hand, long-haul flights, especially those to and from larger cities like Melbourne and Sydney (Darwin-Sydney, Darwin-Melbourne, and Sydney-Perth) have consistently been underperforming across the four years, recording the lowest on-time departure to flights flown ratios.

Since the data is recorded in a monthly performance of individual airlines in each route, observations such as months with few flights or airlines with few operations on the routes are also filtered out. Comparing the unfiltered list, no significant changes are observed regarding the ranking and the route on-time performance. Only eight routes were different for both the best-performing and least-performing routes across the years. The finding implies that the performances of the unfiltered routes are consistent even when considered in high-traffic operations. Furthermore, though some changes are observed regarding the on-time performance values, the differences are marginal.

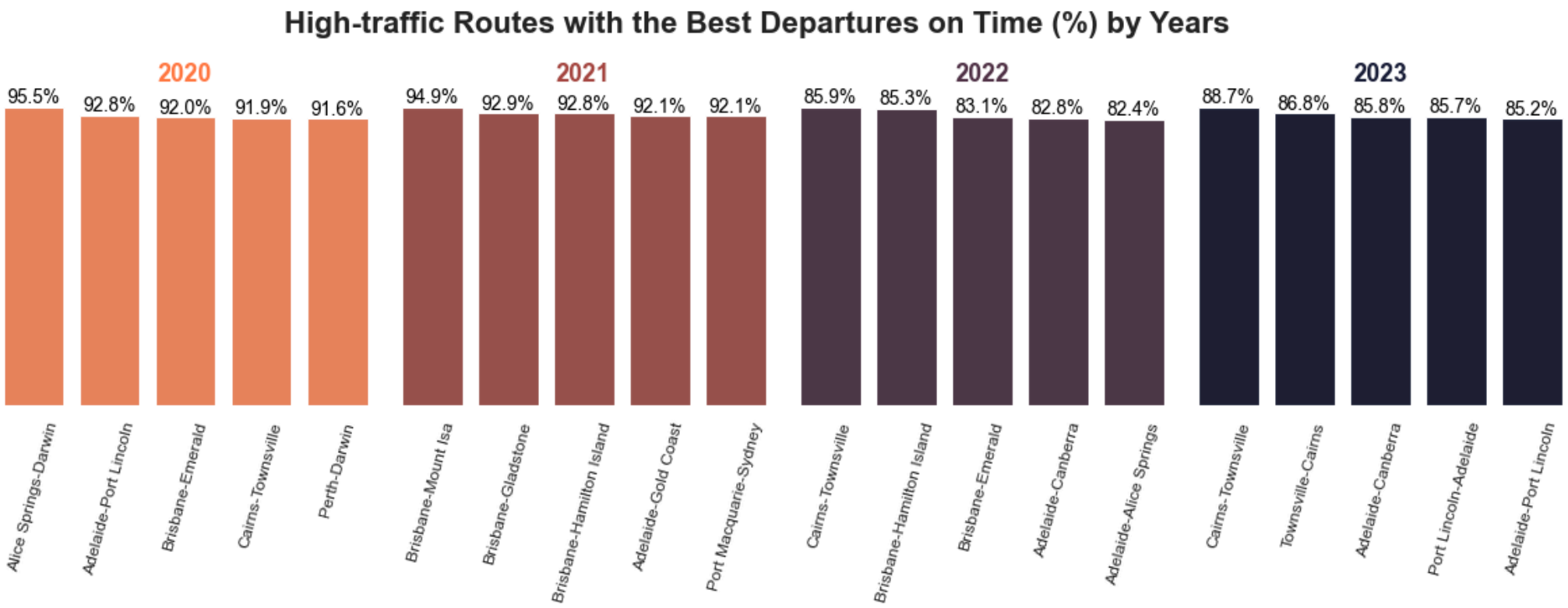

Figure 3.3.2. The Best Departure Performance of High-traffic One-way Routes 2020-2023

Since this section discusses similar metrics to the previous graphs, the choice of visualisation and presentation style remains unchanged, conveying the same intention consistently *(See Appendix C for the other years and H.9. for visualisation codes)*.

Cancellation Proportion of Ports by Year

Table 3.4.1. Cancellation Rates of Arriving and Departing Port 2020-2023

2020

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Sydney	10.9%	70,748
Melbourne	10.8%	53,238
Canberra	9.3%	10,970
Newcastle	9.0%	3,296
Ayers Rock	8.8%	340

Arriving port

Arriving port	Cancellation rate	Sectors Scheduled
Sydney	11.2%	70,808
Melbourne	10.9%	53,306
Canberra	9.3%	10,982
Gold Coast	8.6%	9,148
Newcastle	8.5%	3,302

2021

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Melbourne	21.3%	77,174
Gold Coast	18.2%	15,806
Sunshine Coast	15.2%	5,274
Sydney	14.8%	83,234
Launceston	14.0%	5,472

Arriving port

Arriving Port	Cancellation rate	Sectors Scheduled
Melbourne	21.3%	76,836
Gold Coast	18.4%	15,868
Sunshine Coast	14.8%	5,284
Sydney	14.8%	83,264
Launceston	14.2%	5,506

2022

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Ballina	7.6%	5,616
Sydney	6.8%	178,770
Melbourne	6.7%	148,876
Gold Coast	6.1%	34,056
Canberra	6.1%	32,716

Arriving port

Arriving Port	Cancellation rate	Sectors Scheduled
Ballina	7.3%	5,616
Sydney	7.0%	178,894
Melbourne	6.9%	148,690
Gold Coast	6.0%	34,058
Canberra	6.0%	32,740

2023

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Armidale	6.4%	3,208
Gladstone	5.6%	4,202
Sydney	5.4%	178,586
Canberra	5.1%	35,074
Melbourne	4.9%	152,812

Arriving port

Arriving Port	Cancellation rate	Sectors Scheduled
Armidale	6.0%	3,180
Sydney	5.7%	178,522
Gladstone	5.3%	4,206
Melbourne	5.1%	152,800
Canberra	4.8%	35,062

Table 3.4.1 presents data on the five arriving and departing airports with the highest cancellation rates from 2020 to 2023. In 2020, major cities like Sydney, Melbourne, and Canberra had the highest rates with Sydney leading both arrivals and departures. Melbourne, Gold Coast, and Sunshine Coast, on the other hand, witnessed a surge in cancellations which could be attributed to the COVID-19 pandemic. NSW and Victoria, in particular, implemented several lockdowns during the first half of the year (Knowlton, 2023).

In 2022, however, an overall reduction in cancellations was observed nationwide as state borders continually opened. The trend continues into 2023 with ports like Armidale and Gladstone maintaining performance in terms of low cancellations. Overall, the trend suggests that smaller ports exhibit more stability and lower cancellations as opposed to busier airports in Metropolitan cities like Melbourne and Sydney.

Cancellation Count vs Cancellation Rates

While the number of cancellations provides a good metric to identify how many flights were affected, the cancellation rate allows us to have a more standardised metric for measuring the relative performance across the ports *(See Appendix H.8. for the code)*. Rates consider their size, flight capacity, and flight frequencies. Furthermore, count-based rankings are usually skewed towards naturally high-traffic areas. Most of the ports with the highest count, in particular, are major cities, undermining the performance of smaller ports *(See Appendix D and E for cancellation counts).*

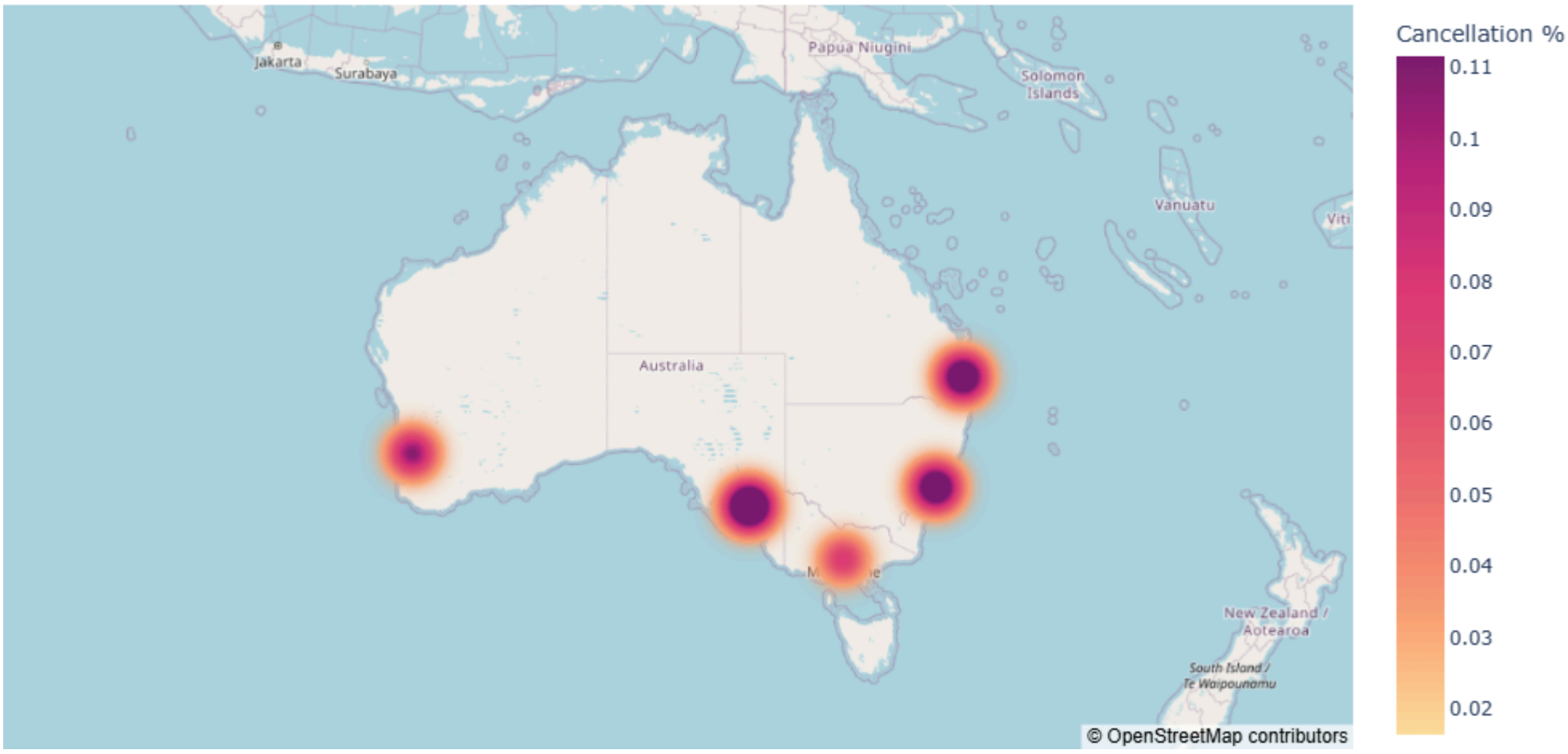

Figure 3.4.2. Cancellation Rates by Arriving Port 2020

Realising the trend predominantly in the metropolitans, we utilised a different approach to visualising the findings for cancellations. The choice of density map not only provides additional geographical information but also emphasises the differences between the areas, conveying the urgency of countermeasures *(See Appendix F for the other years and H.9. for visualisation codes)*.

IV. Analysis of Historical Data

To further expand our analysis, we investigated the flight records from 2010 to 2019. We initially grouped them into three time periods as follows:

- **Post-recovery from Global Financial Crisis**. The period from 2010 to 2012 started a few years after the Global Financial Crisis of 2008.
- **Period of Economic Stability**. The years 2013 to 2018 saw a stable global economy compared to other periods (Maggi & Andaloussi, 2023). We note, however, the disappearance of Malaysia Airlines Flight 370 that heightened aviation security globally.
- **Period of Increasing Geopolitical Friction**. The pre-pandemic period from 2017 to 2019 initiated a geopolitical recession between countries, particularly between the United States and China.

On-time performance

Compared to recent years, routes recorded consistent on-time departure and arrival performance between 2010 and 2019 *(see Appendix G for more detailed information and H.10 for the code)*. On-time performance values range from 60% to 94%, across different considerations such as one-way flights, and high-traffic routes. Several of the best-performing routes across the period feature interstate routes and flights connected to regional areas across Australia. Notably, ports of Alice Springs and Adelaide were mentioned consistently within the top-performing lists, particularly in the first half of the 2010s. Such a trend is also seen in more recent years, from 2020 to 2023, with smaller airports outperforming metropolitan cities in terms of on-time arrivals and departures.

On the other hand, major cities like Melbourne and Sydney were consistently listed as the worst performers between 2010-2019, also mirroring performance trends highlighted by data in the early 2020s.

Cancellations

The analysis of flight cancellations reveals that a large number of cancellations are concentrated in routes involving major cities *(see Appendix G for more detailed information and H.10 for the code)*. Sydney, in particular, recorded 50,000 cancellations, the highest record between 2010-2019. However, it is important to note that a proportion of these cancellations were caused by bad weather and not necessarily due to operational inefficiencies (Coulton, 2019; Weir, 2018; Reddie, 2017). Overall, the average cancellation rate ranged between 2% and 5% in the last decade. The best performers once again were smaller regional airports like Karratha, Gladstone, Moranbah, and Tamworth.

While peak pandemic numbers saw the flight cancellation rate surge to 20%, in recent years cancellations seem to have fallen and stabilised back to pre-pandemic levels. Nevertheless, the abrupt emergence of COVID-19 revealed a lack of readiness and emergency measures in airports, leading to high cancellation rates.

V. Conclusion and Recommendations

Between 2020 and 2023, cancellations and delays declined, indicating that the industry stabilised after the pandemic. Major airports such as Sydney, Melbourne, and Canberra had the highest percentages of delays and cancellations with Sydney leading both categories. Overall, the data suggests that smaller airports such as Cairns, Townsville, Armidale, and Gladstone consistently performed well, maintaining a low percentage of cancellations and delays. Major ports like Sydney and Melbourne, on the other hand, have displayed a significant variability in performance which could be attributed to increasing air traffic.

Based on the analysis, we recommend the Australian aviation industry to implement a focused approach toward driving operational efficiency in major ports. Firstly, they must consider investing in upgrading and expanding infrastructure in major airports. Delays could be occurring due to a lack of resources in handling the ever-increasing traffic. This would include building new runways, aircraft bays, and more terminals. Considering that flight delays can also occur because flights occasionally wait for all passengers to board, building more check-in counters and security terminals could prove beneficial in speeding up the passenger's time from entry to the terminal, thereby reducing overall boarding times and minimising delays.

The next strategy is to enhance air traffic control. Actions could involve collaborating with air traffic authorities to better optimise flight paths and allow a more streamlined influx of traffic in major ports. Such strategies could ultimately reduce delays due to congestion in the airspace. Airport authorities could also consider leveraging the stability of smaller ports to re-reroute traffic to reduce congestion and thus improve overall efficiency. Furthermore, providing regular training and upskilling to air traffic control personnel could help them face new issues, such as the advancement of technology used by airlines and the increasing traffic due to rapid urban development in Australia.

The air transport industry could also consider implementing real-time monitoring systems to better predict potential delays. After collecting sufficient data, they can begin identifying trends, patterns, and factors that cause cancellations and delays. For example, if bad weather during winter causes 15-minute delays for flights between Sydney- and Perth, airport authorities could consider factoring this information into their analysis before estimating the arrival time. Another way in which analytics could be used is to forecast passenger demand. This will help airlines better schedule their flights, minimising the risk of overbooking and on the other hand reducing the probability of a flight being cancelled due to insufficient passengers.

Similarly, emerging technologies such as blockchain and the metaverse could accelerate data-driven decisions across aviation systems. Digital twins, for instance, are digital representations of real-life objects that enable deeper analysis and simulate real scenarios (McKinsey, 2023). Such technology could utilise real-time data from ongoing and scheduled flights to simulate and smooth out traffic inside ports while considering external factors such as weather and flight demands. Digital twins can be further enhanced by integrating blockchain technology to provide a more secure and faster exchange of information from one port to another.

References

Andaloussi, M & Maggi, C. (2023, April 19). World economic outlook shows economies facing high uncertainty. International Monetary Fund. https://www.imf.org/en/Blogs/Articles/2024/01/12/charts-spotlight-inflation-economic-growth-globalization-and-climate-change

Briggs, C. (2022). Australia had more than 10 million COVID-19 cases this year. But how many more went unrecorded? ABC News. https://www.abc.net.au/news/2022-12-29/how-many-covid-19-cases-are-out-there-as-2022-ends/101812426

Coulton, A. (2019). Sydney Airport hit with cancellations and delays. TravelWeekly. https://www.travelweekly.com.au/article/sydney-airport-hit-cancellations-delays/

Knowlton, C. (2023, February 23). A timeline of Covid-19 in Australia, two years on. Timeout. https://www.timeout.com/melbourne/things-to-do/a-timeline-of-covid-19-in-australia-two-years-on

McKinsey. (2023, July 12). What is digital-twin technology? https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-digital-twin-technology

Reddie, M. (2017). Sydney Airport flight chaos continues as strong winds force more cancellations. ABC News. https://www.abc.net.au/news/2017-09-15/sydney-airport-delays-continue-wind/8947914

Appendices

Appendix A. Best and Worst Performance of One-way Routes

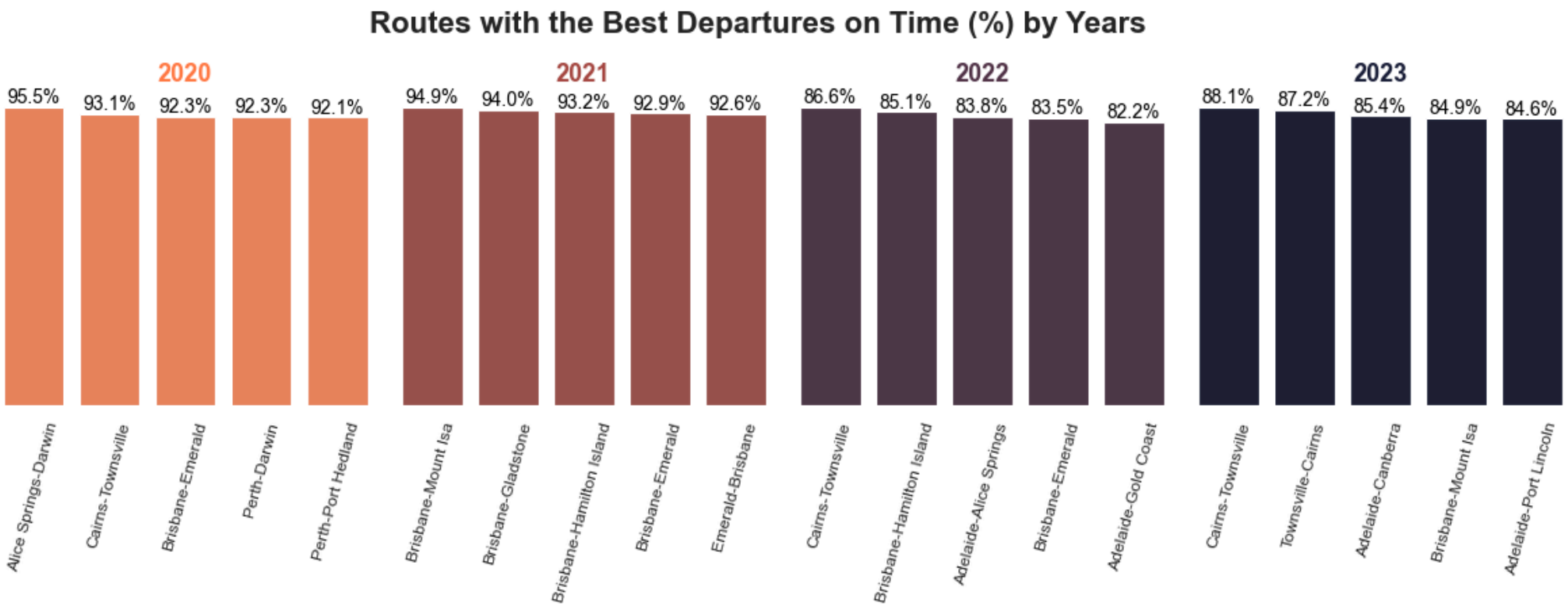

Figure A1. The Best On-time Departure of One-way Routes 2020-2023

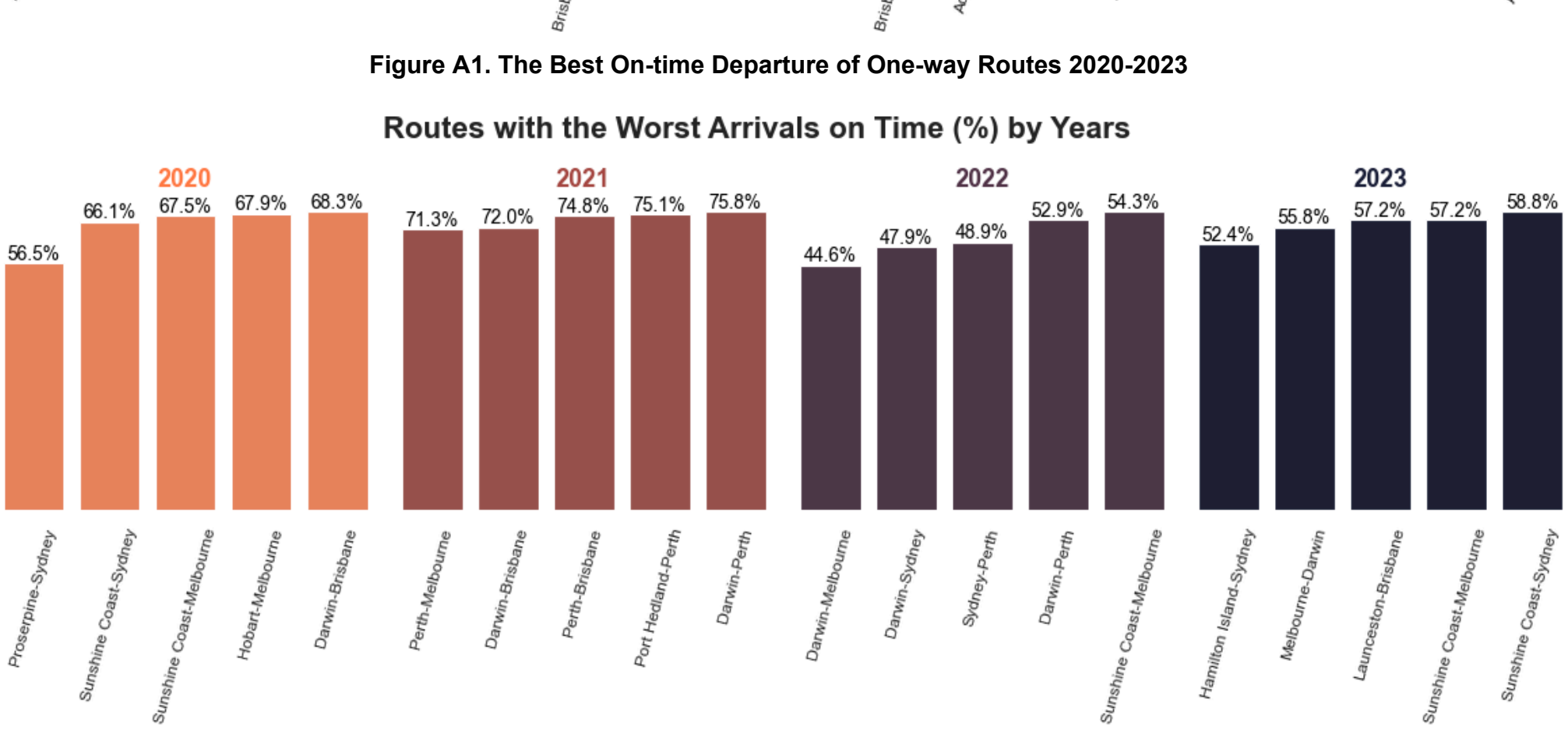

Figure A2. The Worst On-time Arrival of One-way Routes 2020-2023

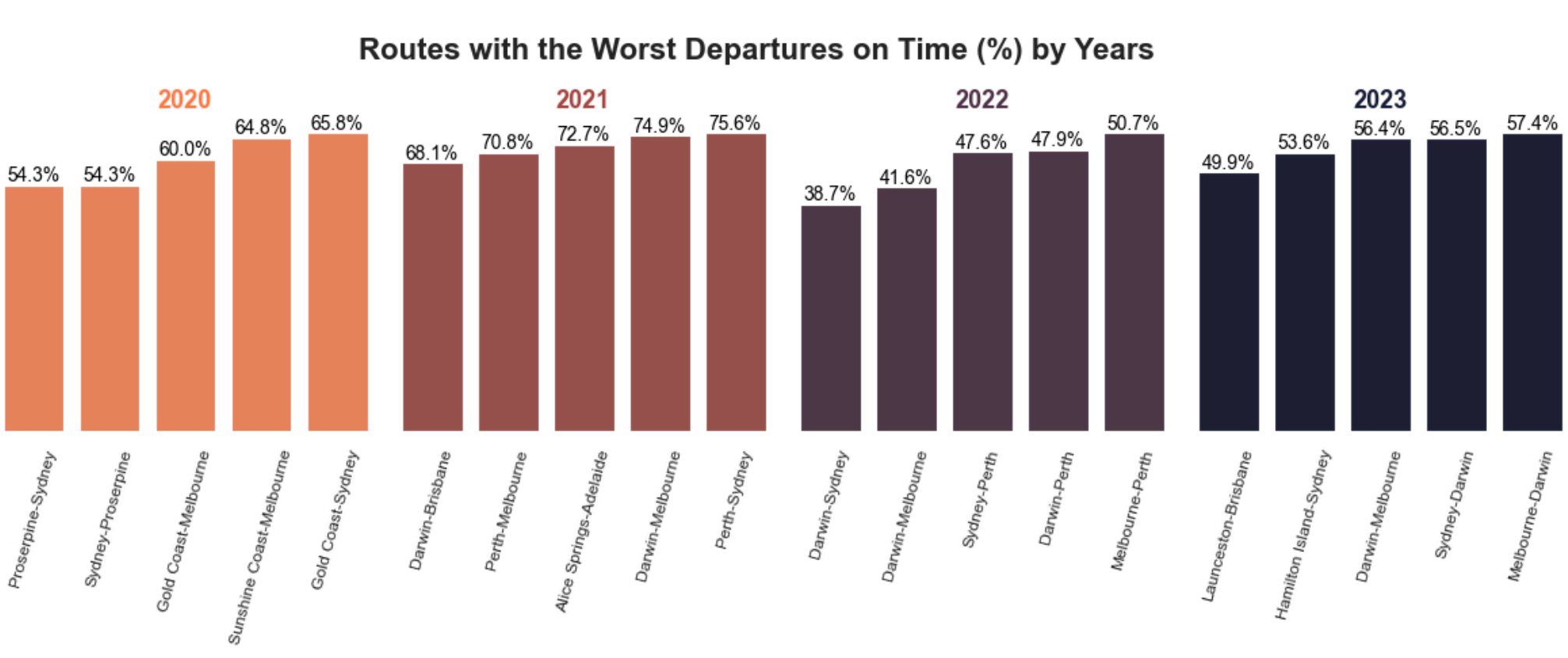

Figure A3. The worst On-time Departure of One-way Routes 2020-2023

Appendix B. Best and Worst Performance of Two-way Routes

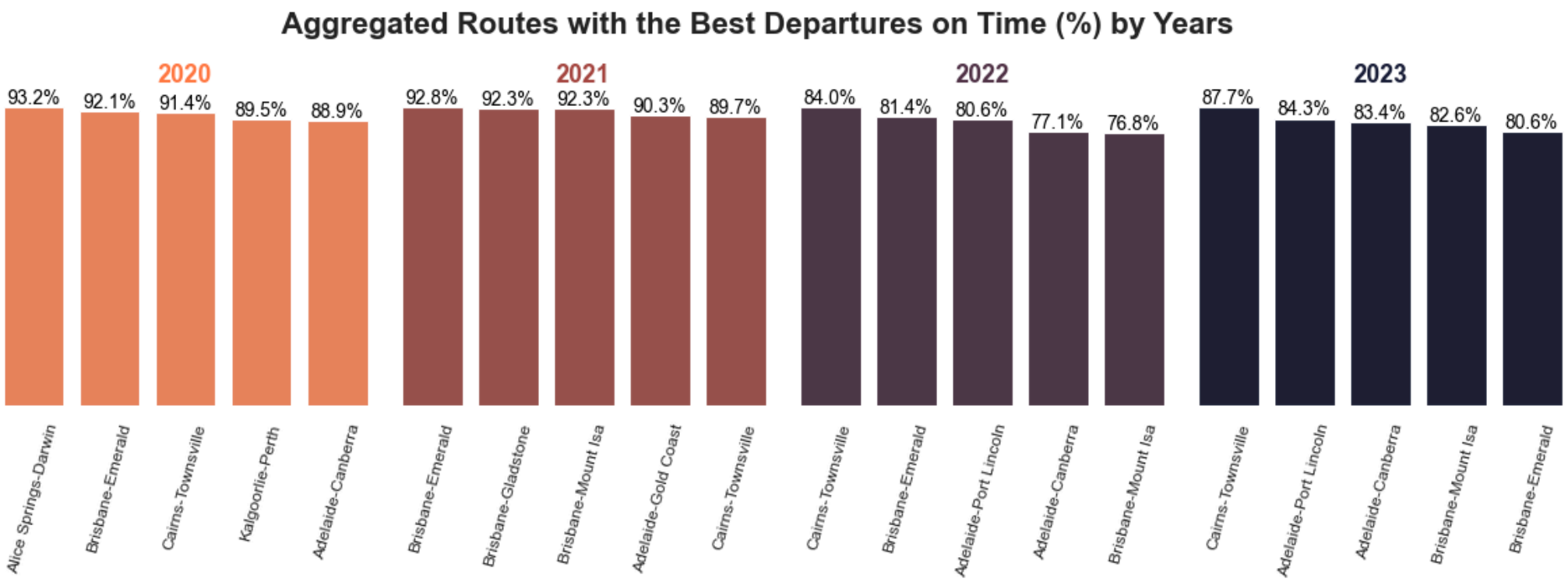

Figure B1. The Best On-time Departure of Two-way Routes 2020-2023

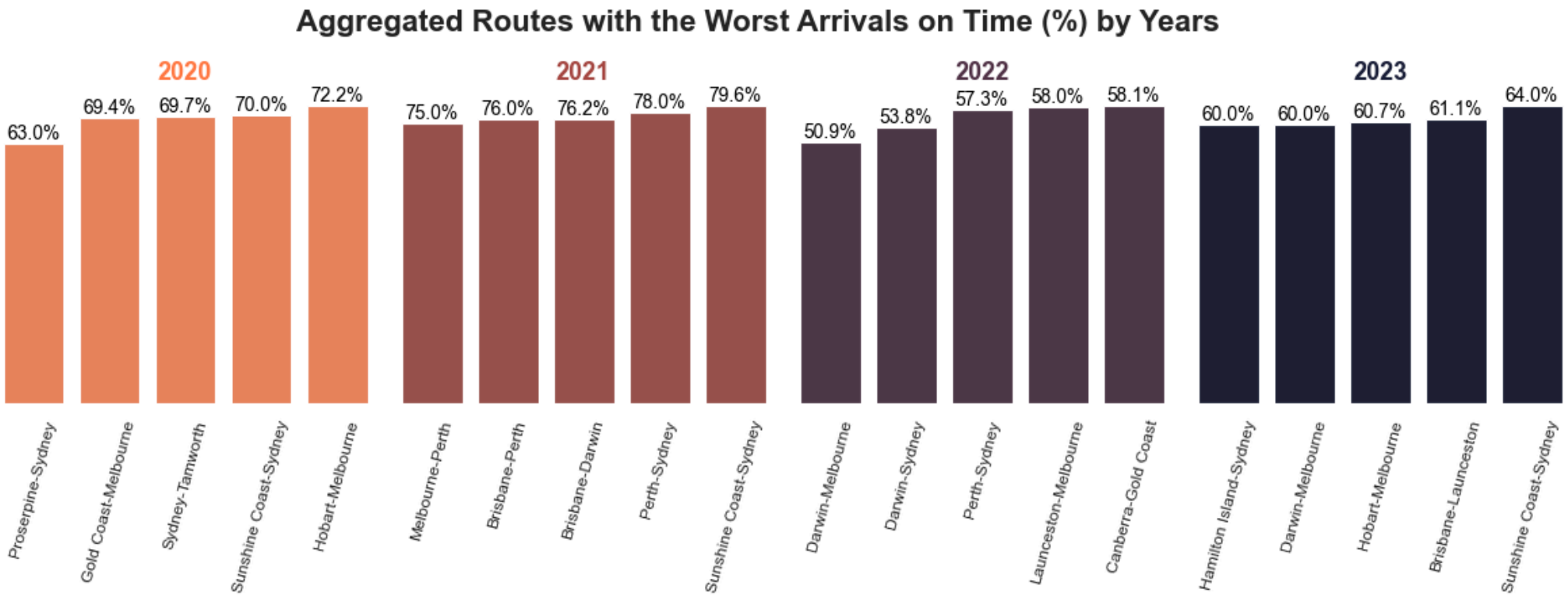

Figure B2. The Worst On-time Arrival of One-way Routes 2020-2023

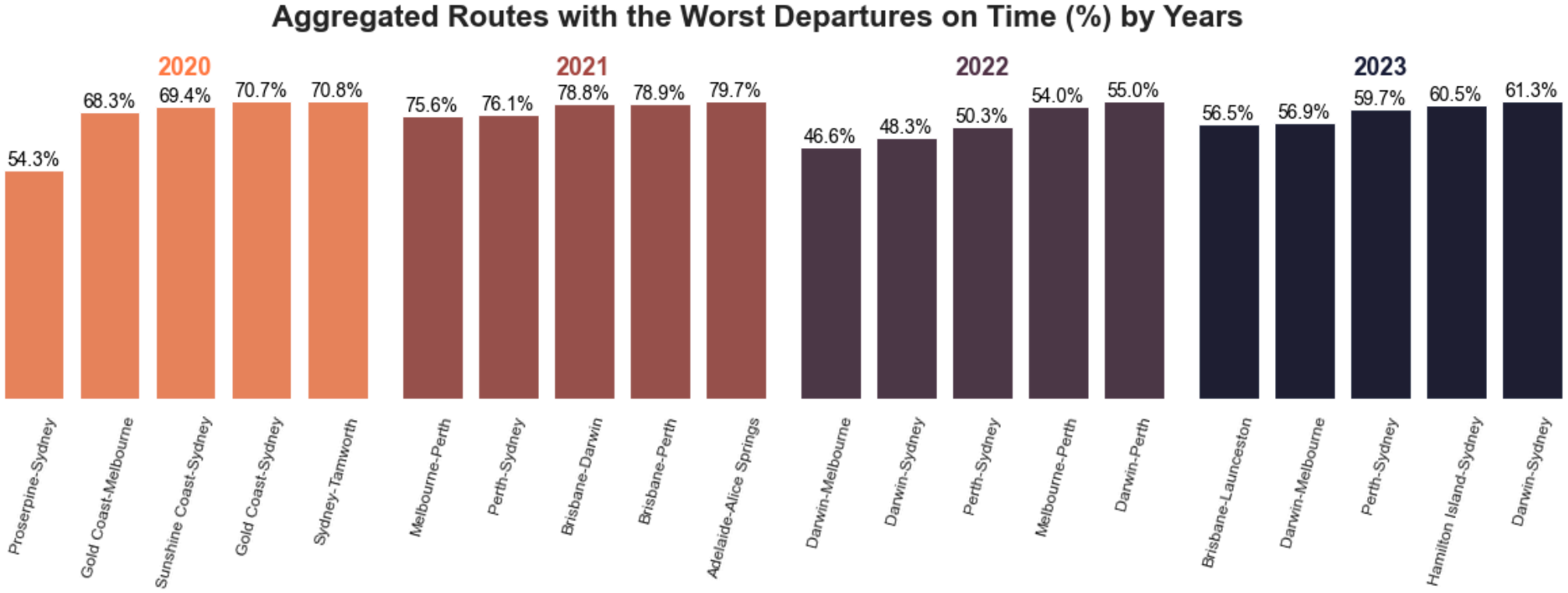

Figure B3. The Worst On-time Departure of Two-way Routes 2020-2023

Appendix C. Best and Worst Performance of One-way High-traffic Routes

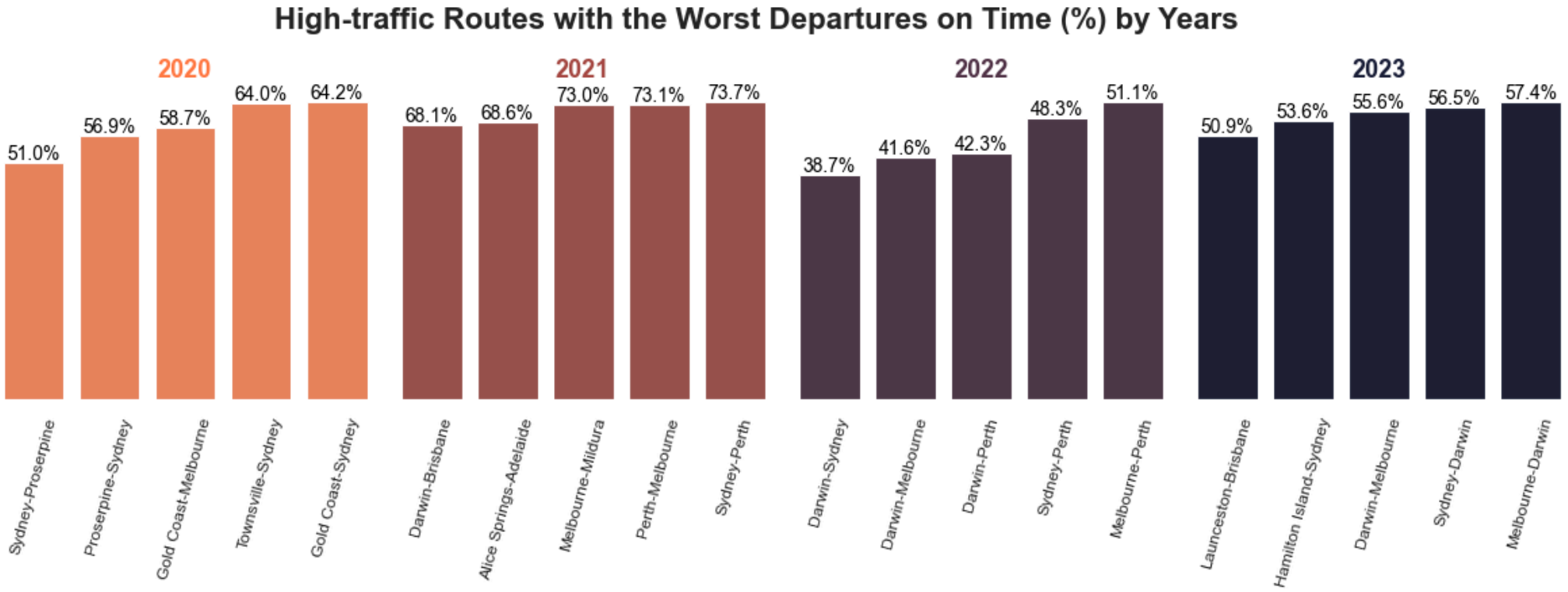

Figure C1. The Worst On-time Departure of One-way High-traffic Routes 2020-2023

Appendix D. Cancellation Counts of Airport

Table D.1. Cancellation Count of Arriving and Departing Port 2010-2019

2020

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	7734	70,748
Melbourne	5760	53,238
Brisbane	3732	59,698
Adelaide	1644	20,836
Perth	1504	25,590

Arriving port

Arriving port	Cancellations	Sectors Scheduled
Sydney	7902	70,808
Melbourne	5786	53,306
Brisbane	3726	59,568
Adelaide	1628	20,838
Perth	1564	25,542

2021

Departing port

Departing Port	Cancellations	Sectors Scheduled
Melbourne	16448	77,174
Sydney	12288	83,234
Brisbane	8446	79,980
Adelaide	3666	30,300
Gold Coast	2874	15,806

Arriving port

Arriving Port	Cancellations	Sectors Scheduled
Melbourne	16346	76,836
Sydney	12280	83,264
Brisbane	8506	79,962
Adelaide	3692	30,340
Gold Coast	2914	15,868

2022

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	12222	178,770
Melbourne	9976	148,876
Brisbane	5106	115,266
Gold Coast	2092	34,056
Canberra	2004	32,716

Arriving port

Arriving Port	Cancellations	Sectors Scheduled
Sydney	12580	178,894
Melbourne	10186	148,690
Brisbane	5246	115,368
Gold Coast	2054	34,058
Canberra	1958	32,740

2023

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	9604	178,586
Melbourne	7482	152,812
Brisbane	4124	122,992
Canberra	1804	35,074
Adelaide	1620	51,390

Arriving port

Arriving Port	Cancellations	Sectors Scheduled
Sydney	10116	178,522
Melbourne	7754	152,800
Brisbane	4302	123,174
Canberra	1664	35,062
Adelaide	1564	51,426

Appendix E. Visualising Flight Cancellations of Airports by Count

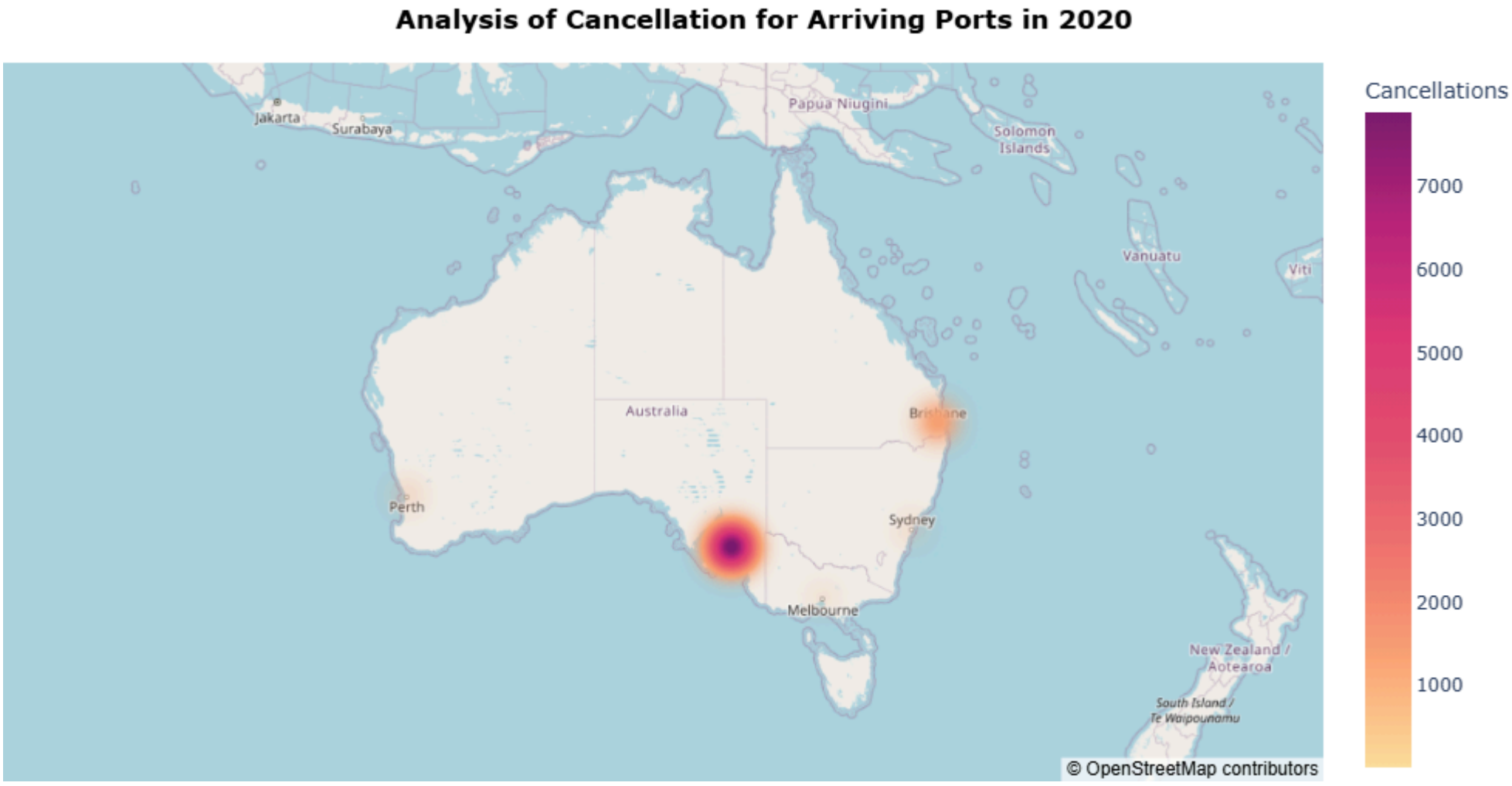

Figure E1. Cancellations by Arriving Port 2020

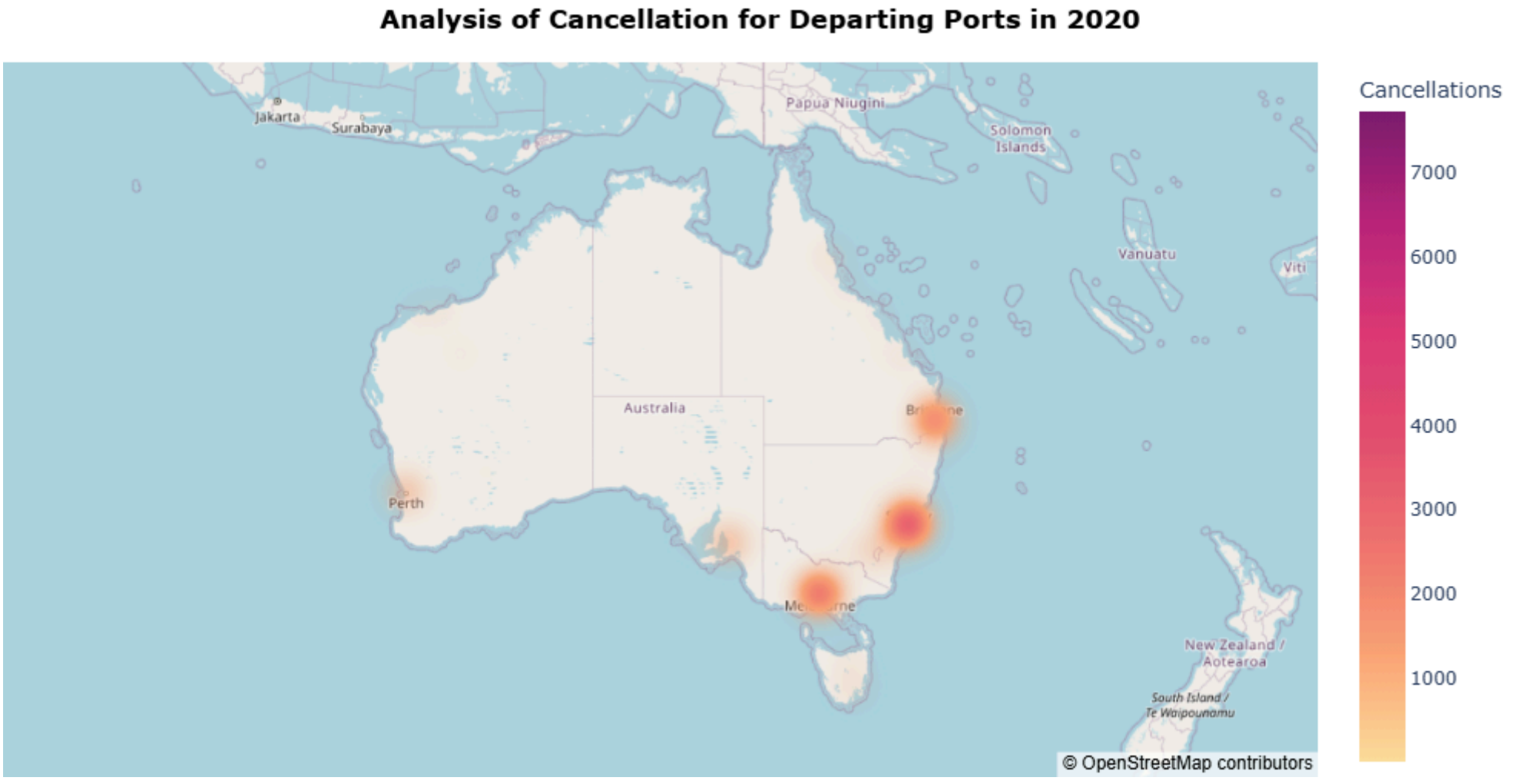

Figure E2. Cancellations by Departing Port 2020

Analysis of Cancellation for Arriving Ports in 2021

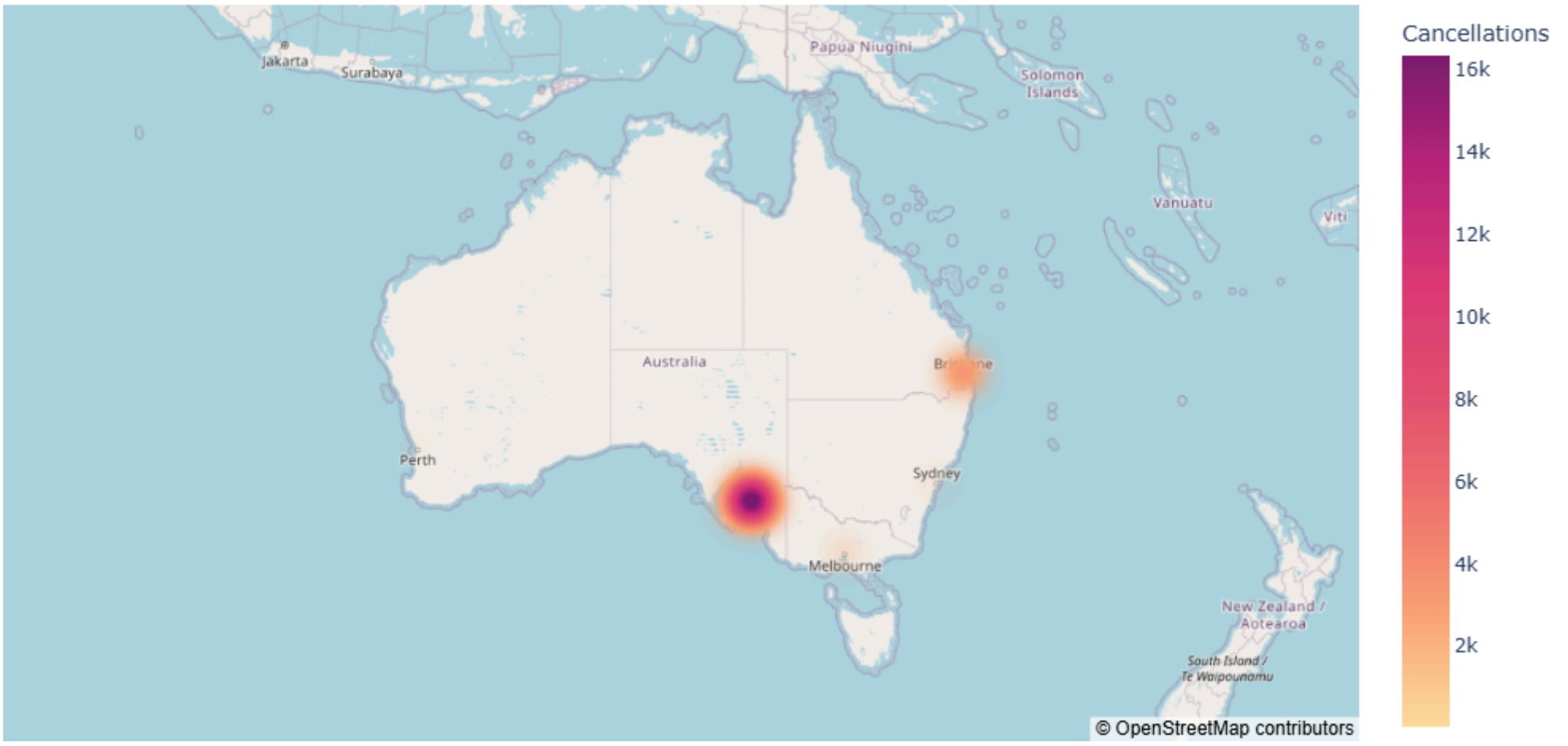

Figure E3. Cancellations by Arriving Port 2021

Analysis of Cancellation for Departing Ports in 2021

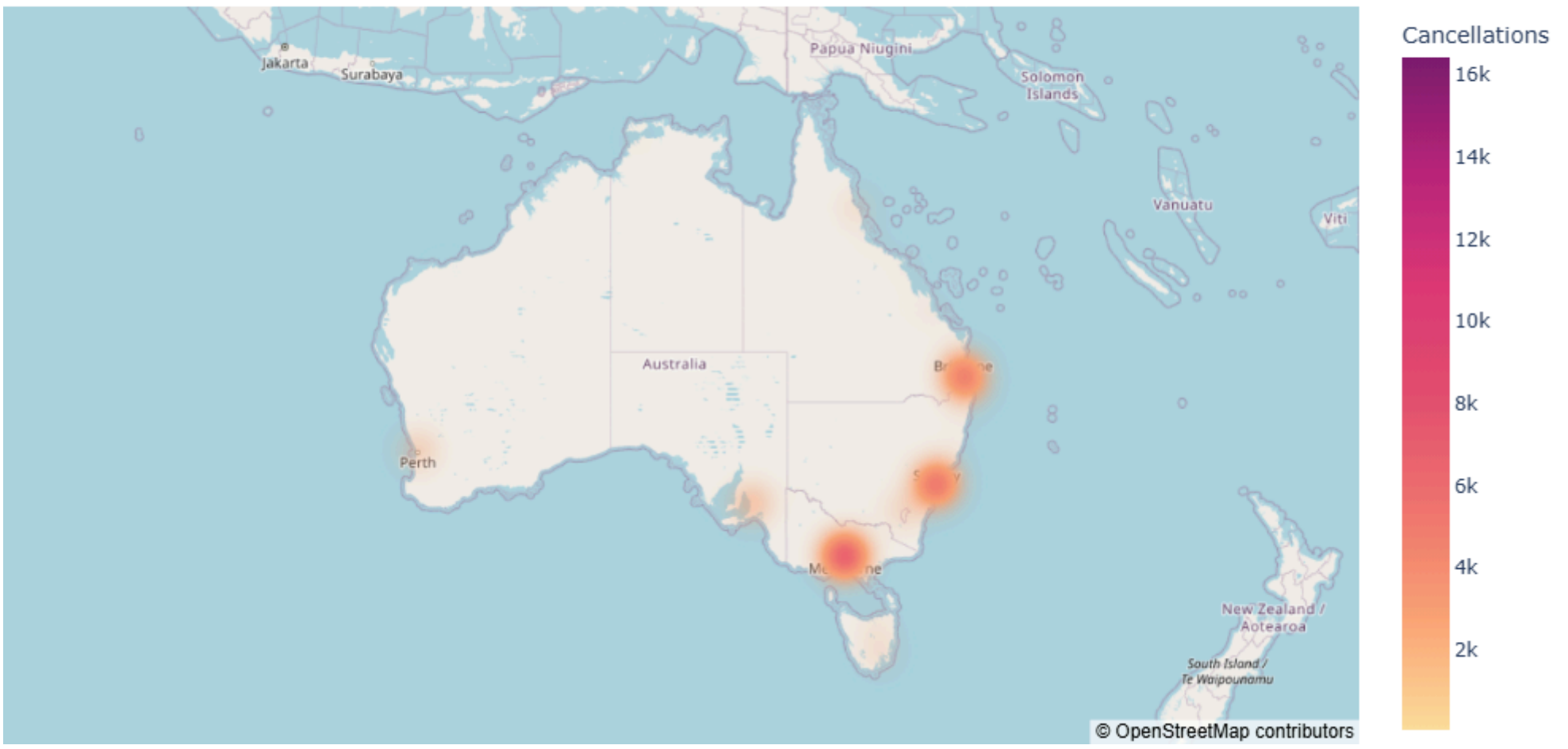

Figure E4. Cancellations by Departing Port 2021

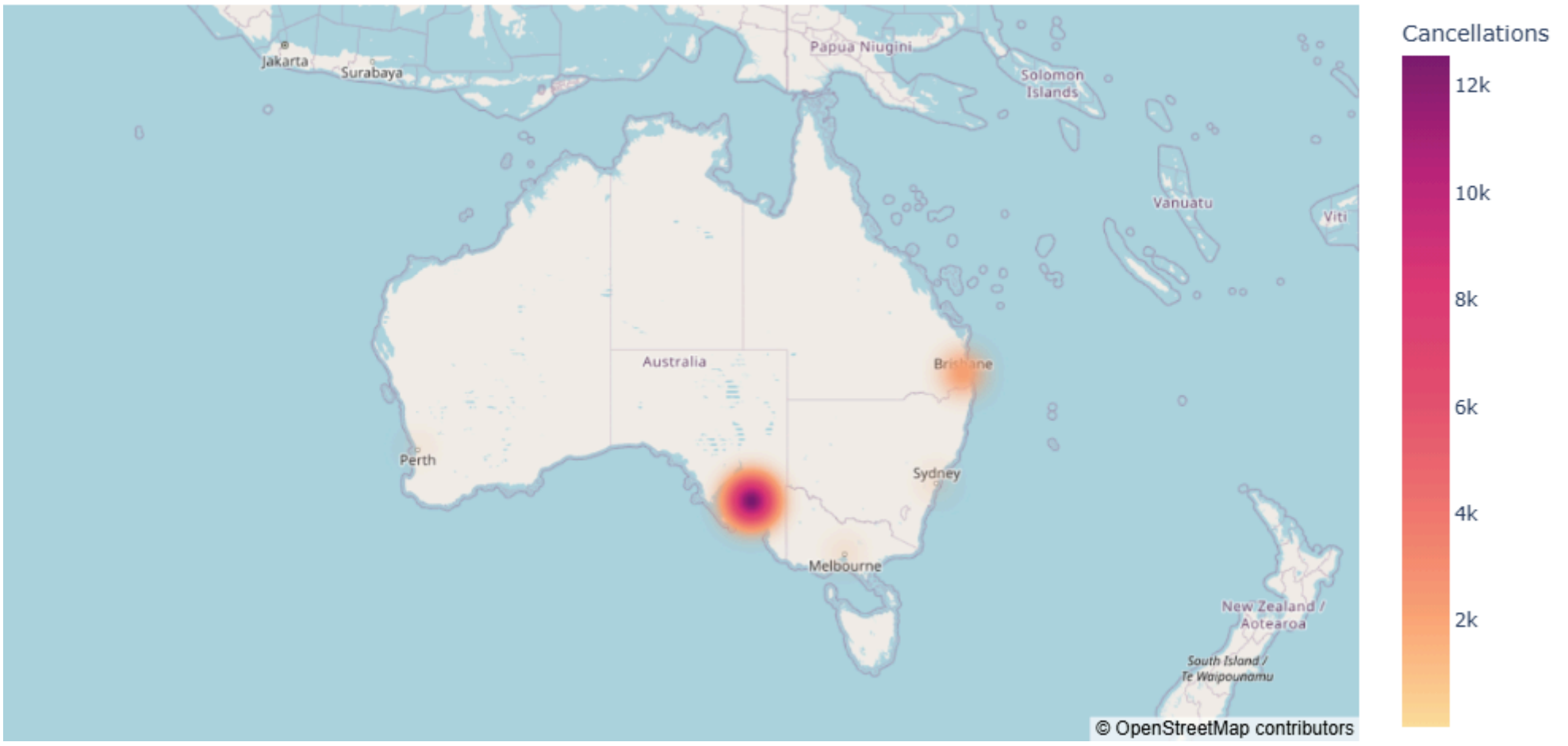

Figure E5. Cancellations by Arriving Port 2022

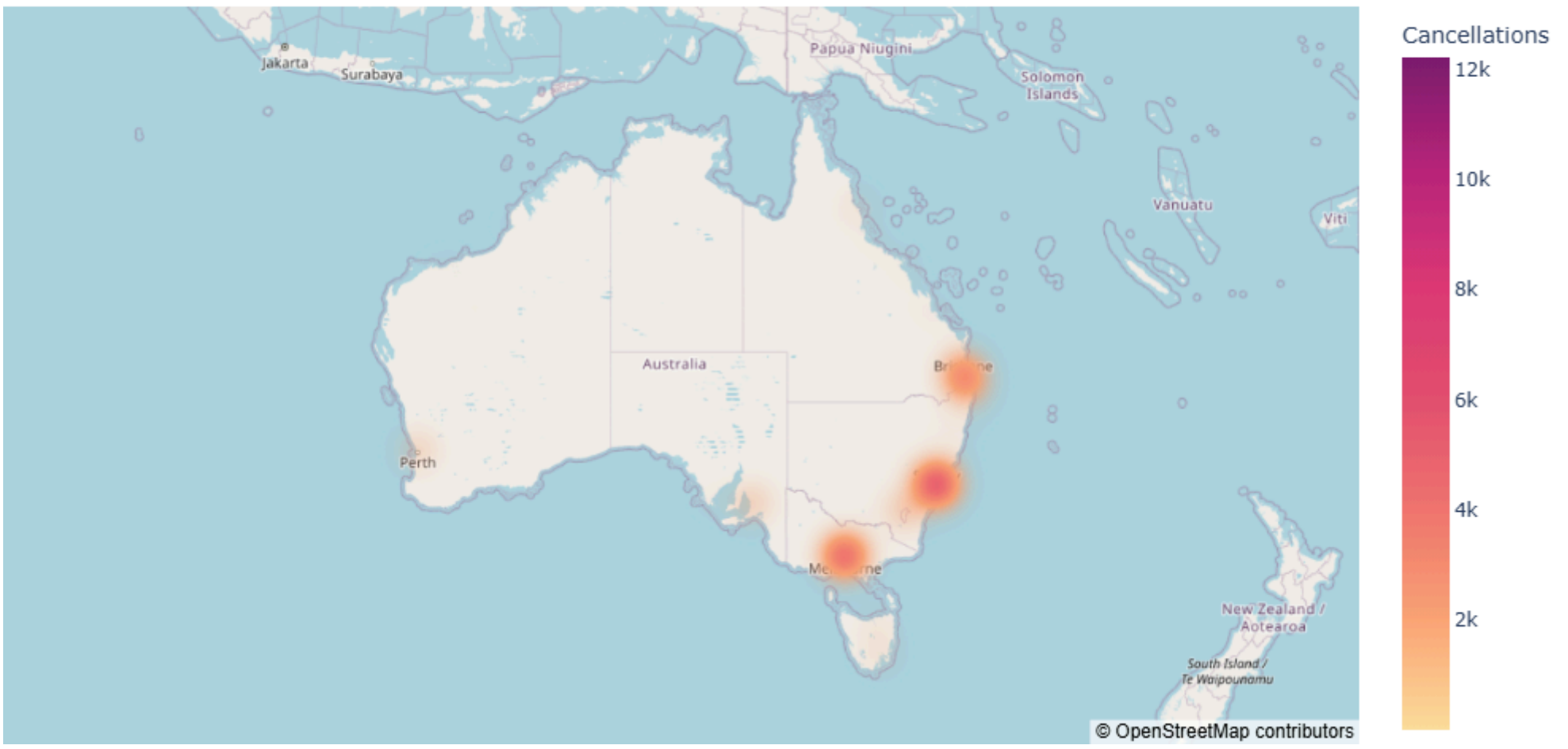

Figure E6. Cancellations by Departing Port 2022

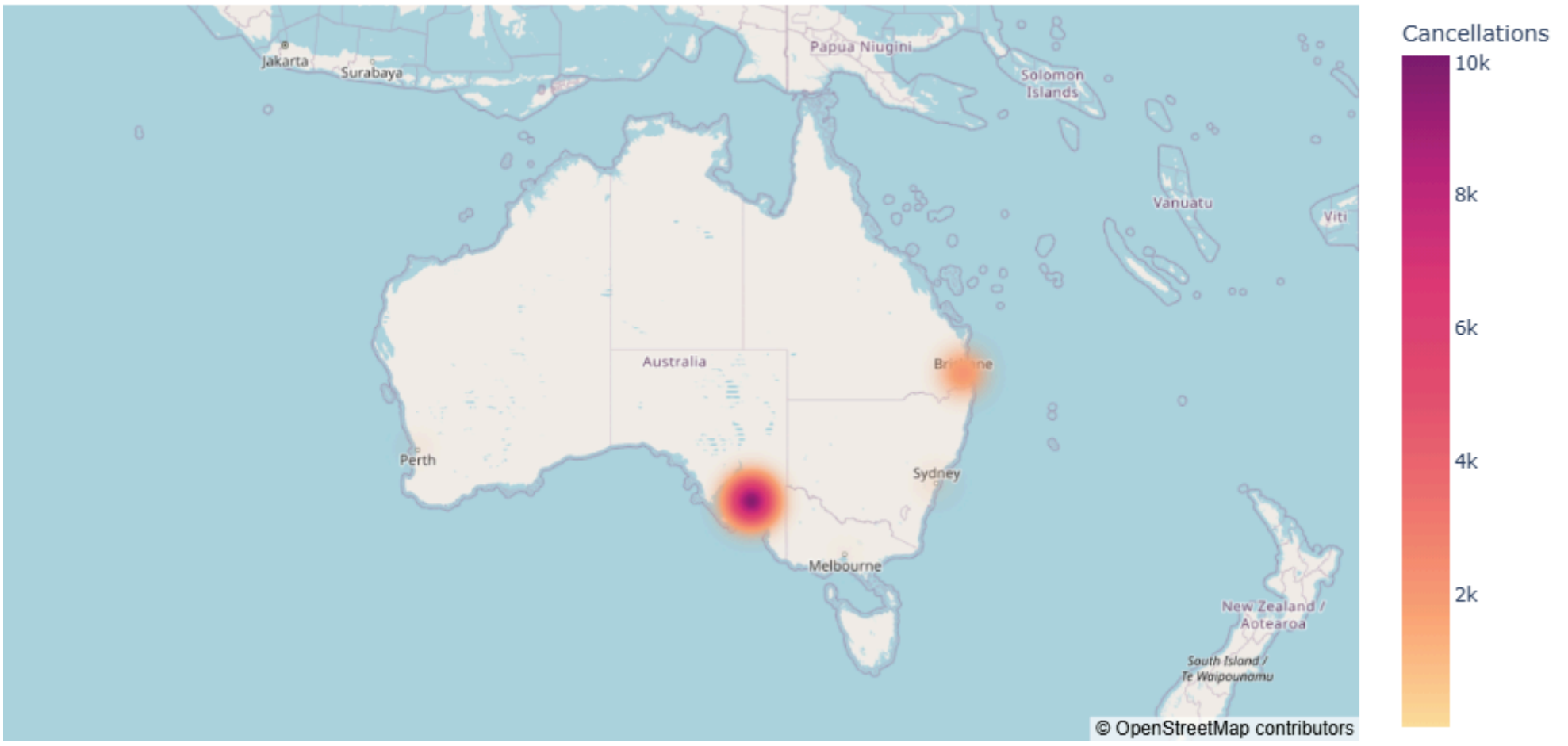

Figure E7. Cancellations by Arriving Port 2023

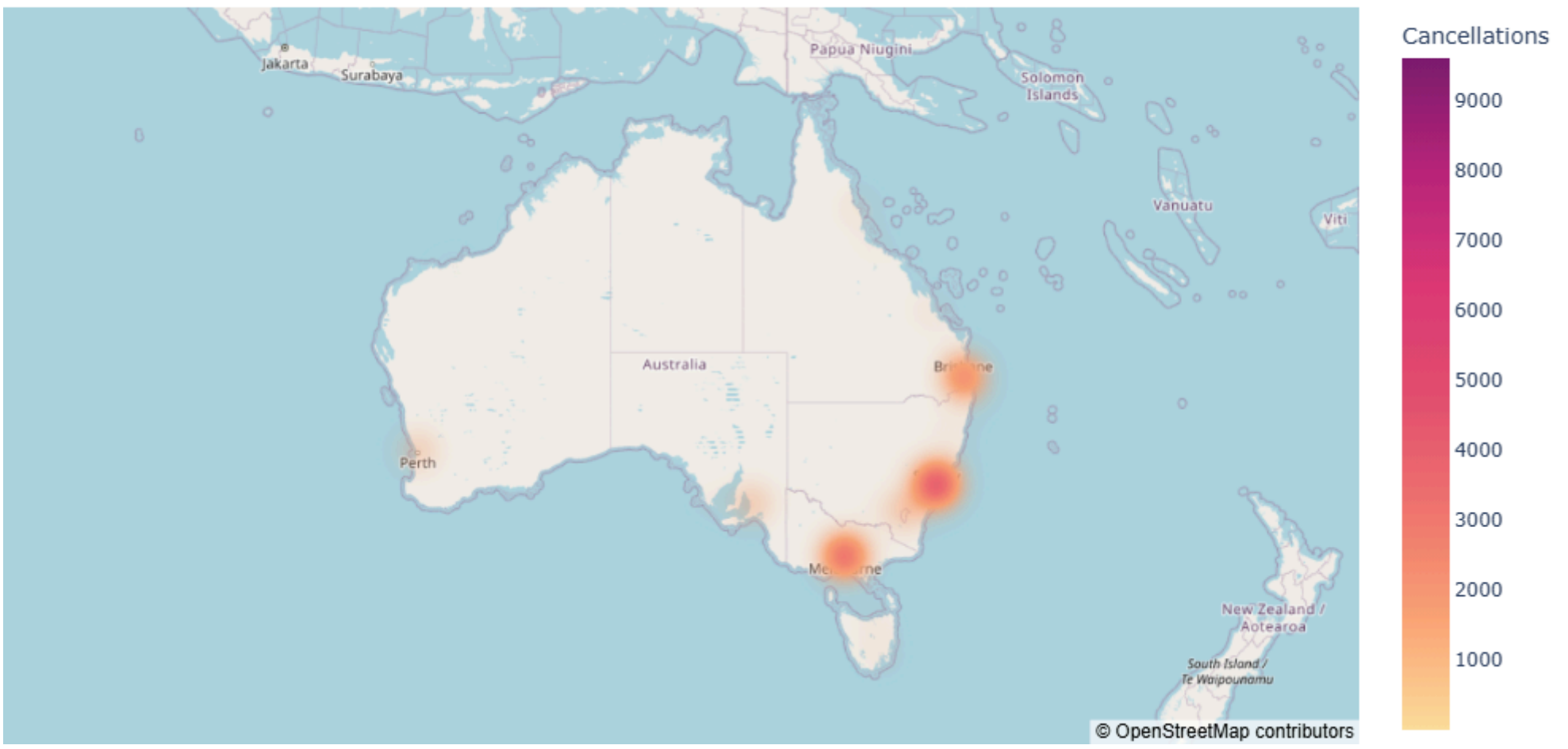

Figure E8. Cancellations by Departing Port 2023

Appendix F. Visualising Flight Cancellation Rates of Airports

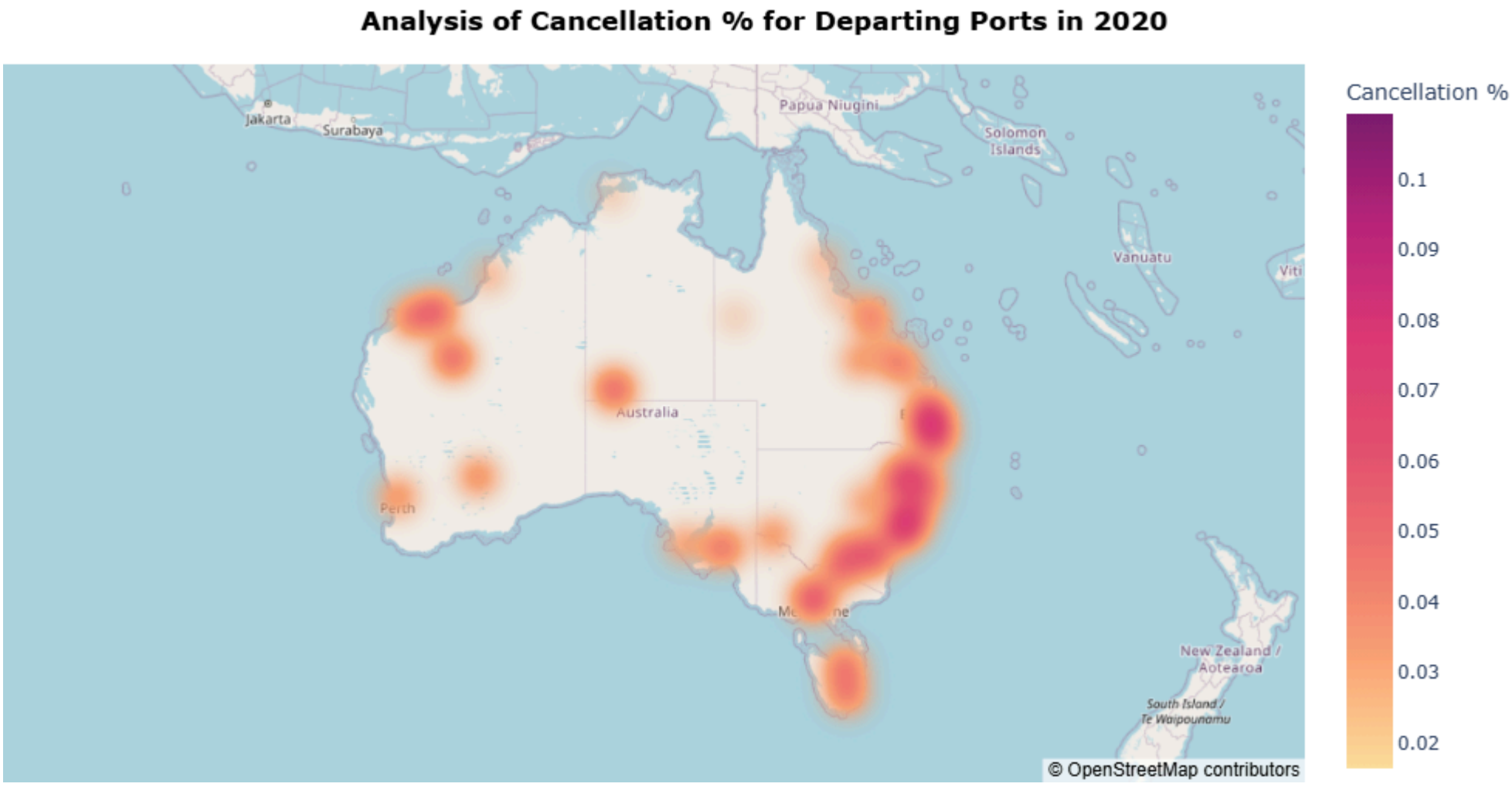

Figure F1. Cancellation Rates by Departing Port 2020

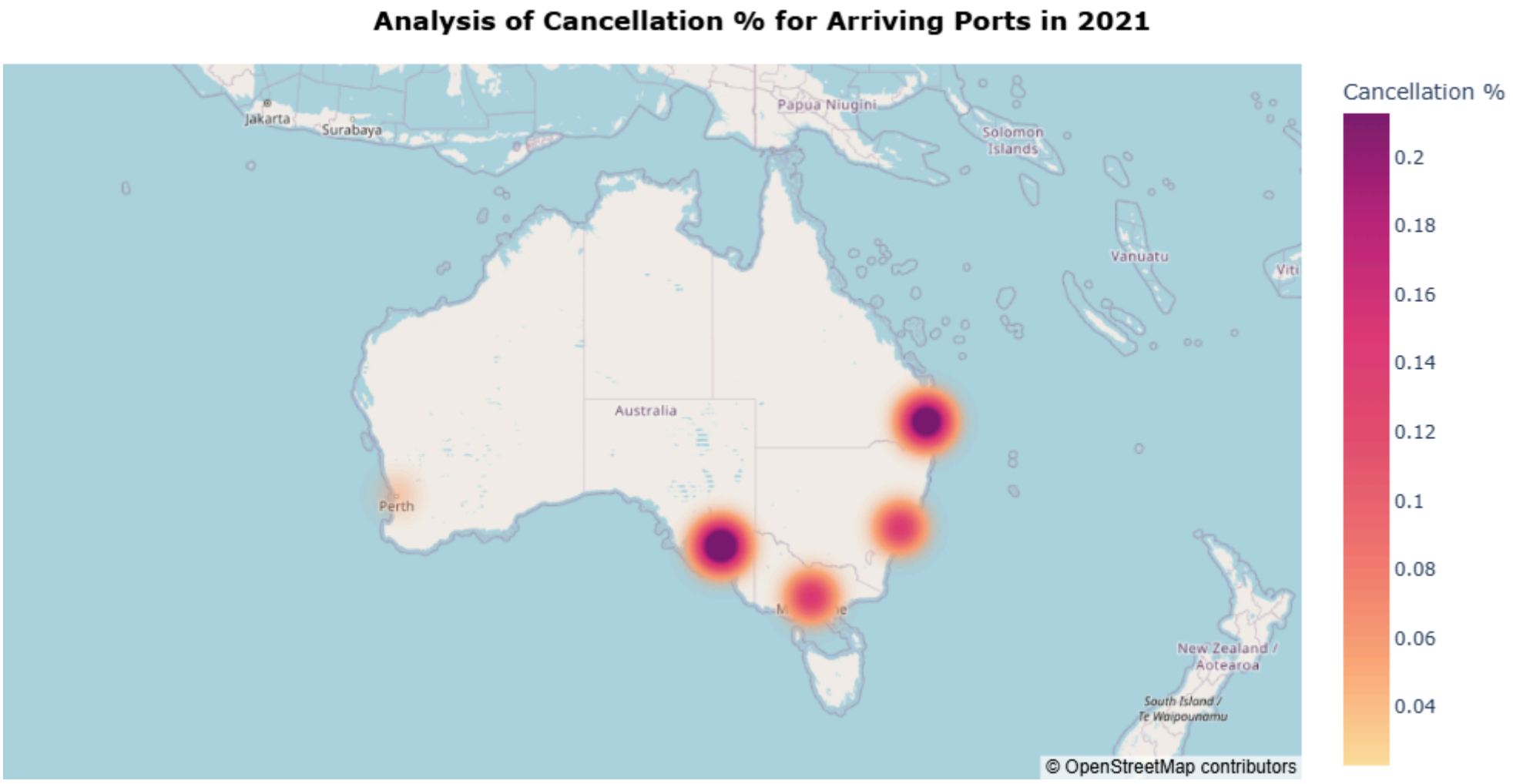

Figure F2. Cancellation Rates by Arriving Port 2021

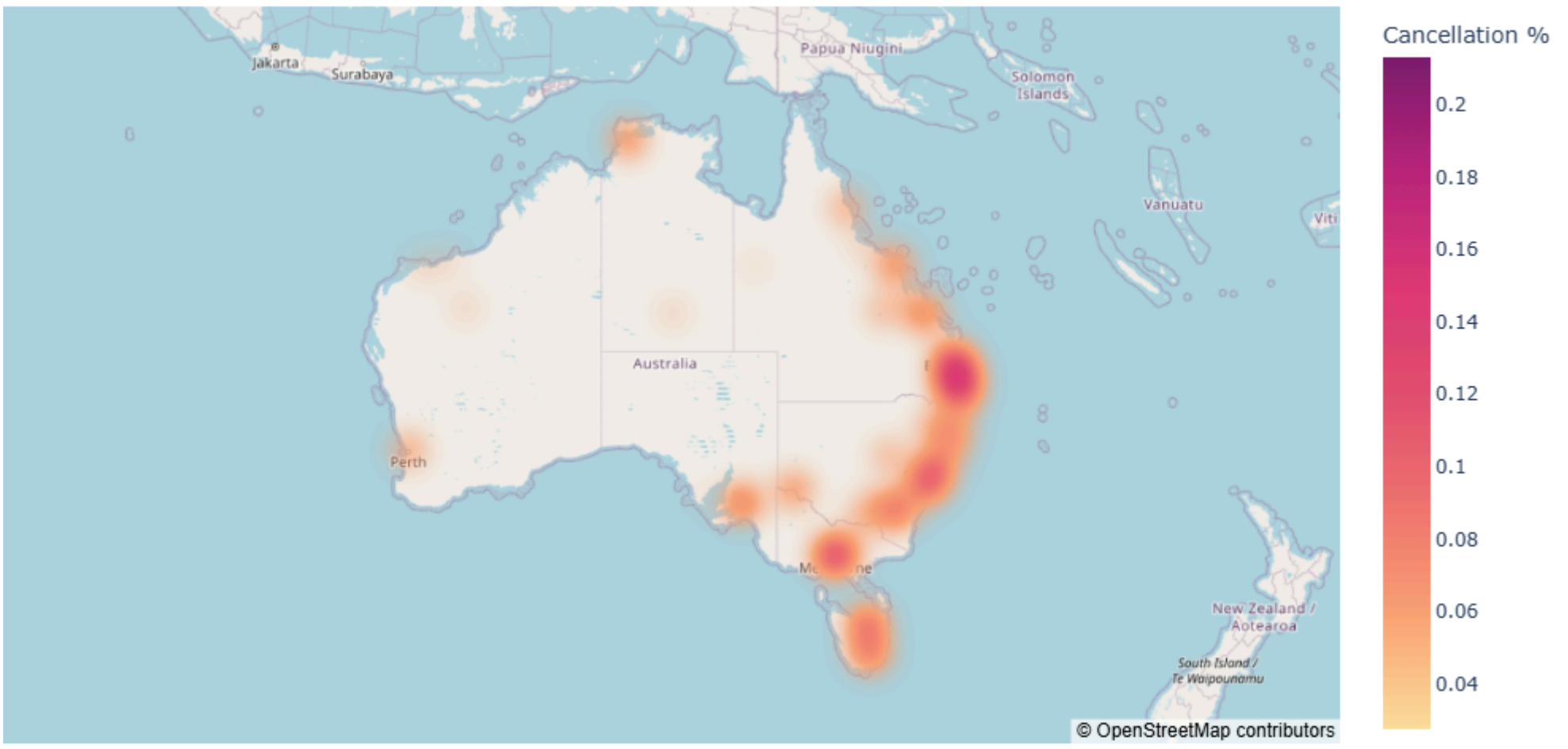

Figure F3. Cancellation Rates by Departing Port 2021

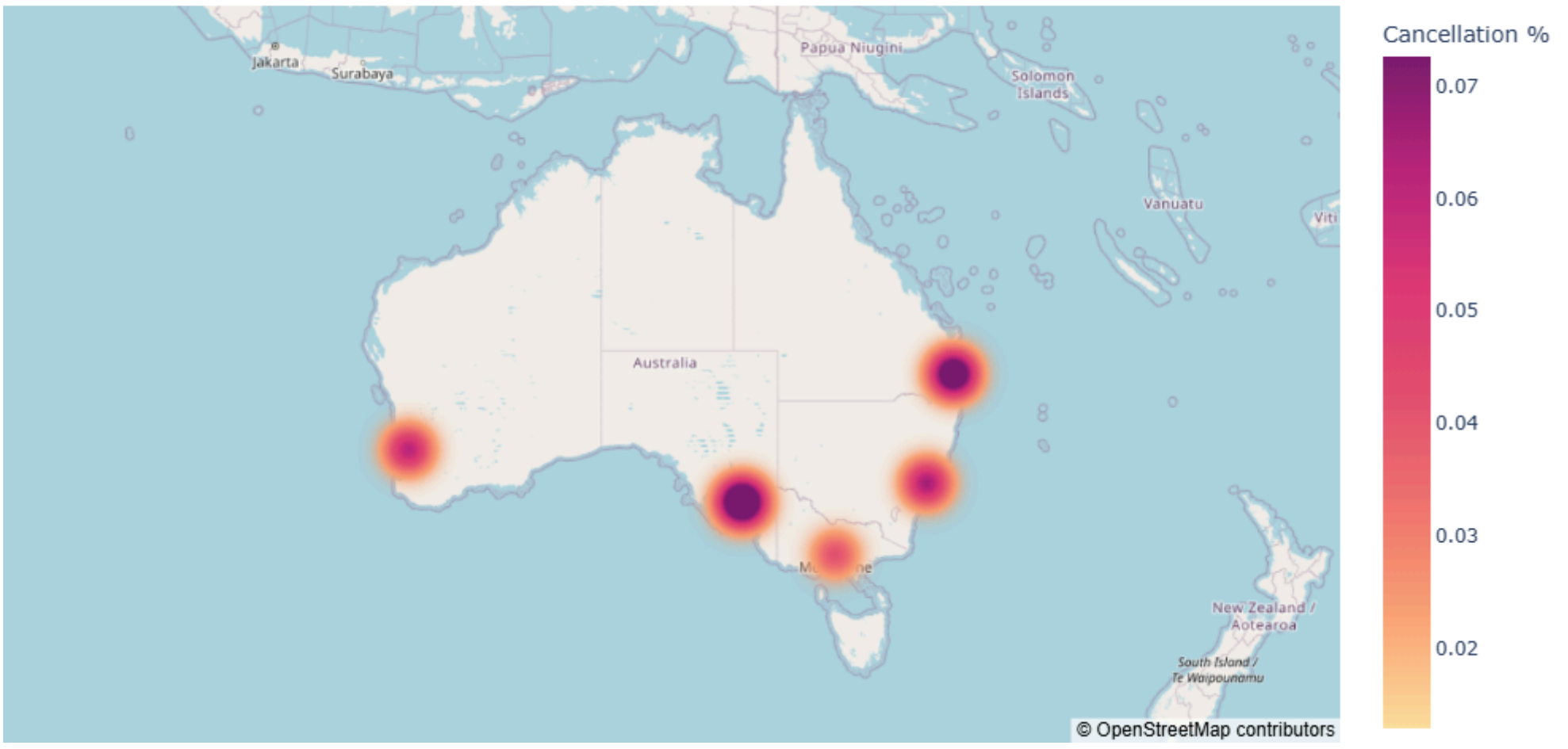

Figure F4. Cancellation Rates by Arriving Port 2022

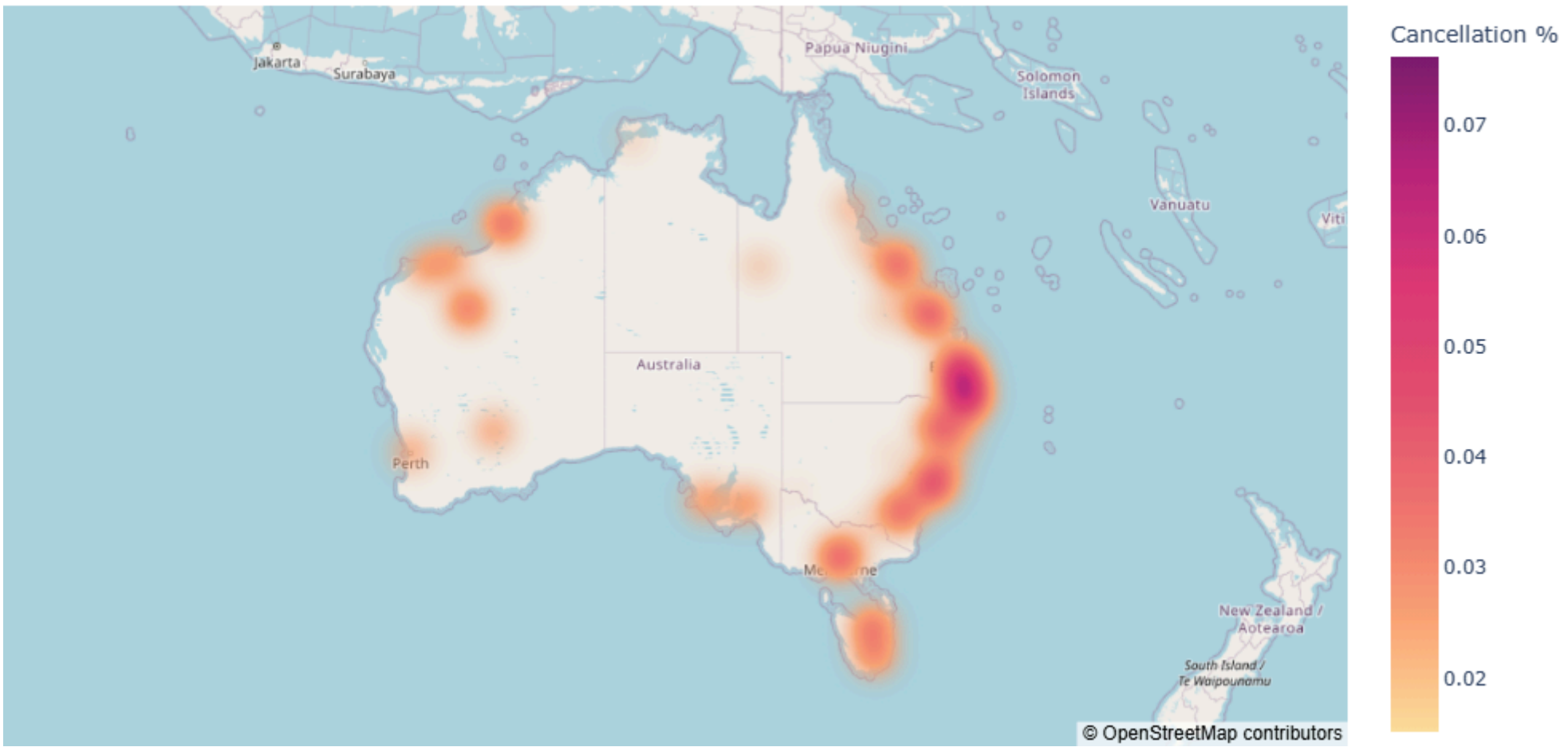

Figure F5. Cancellation Rates by Departing Port 2022

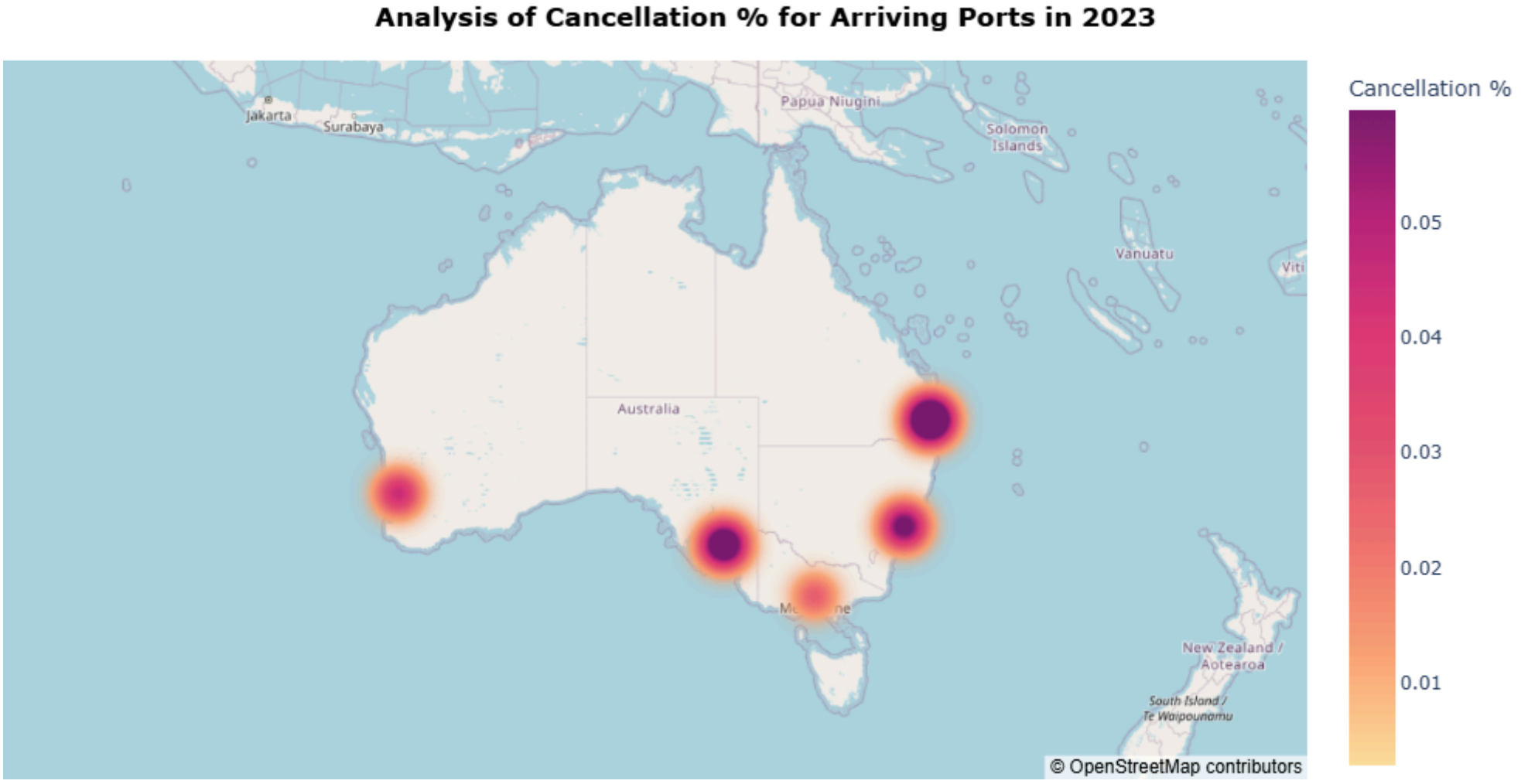

Figure F6. Cancellation Rates by Arriving Port 2023

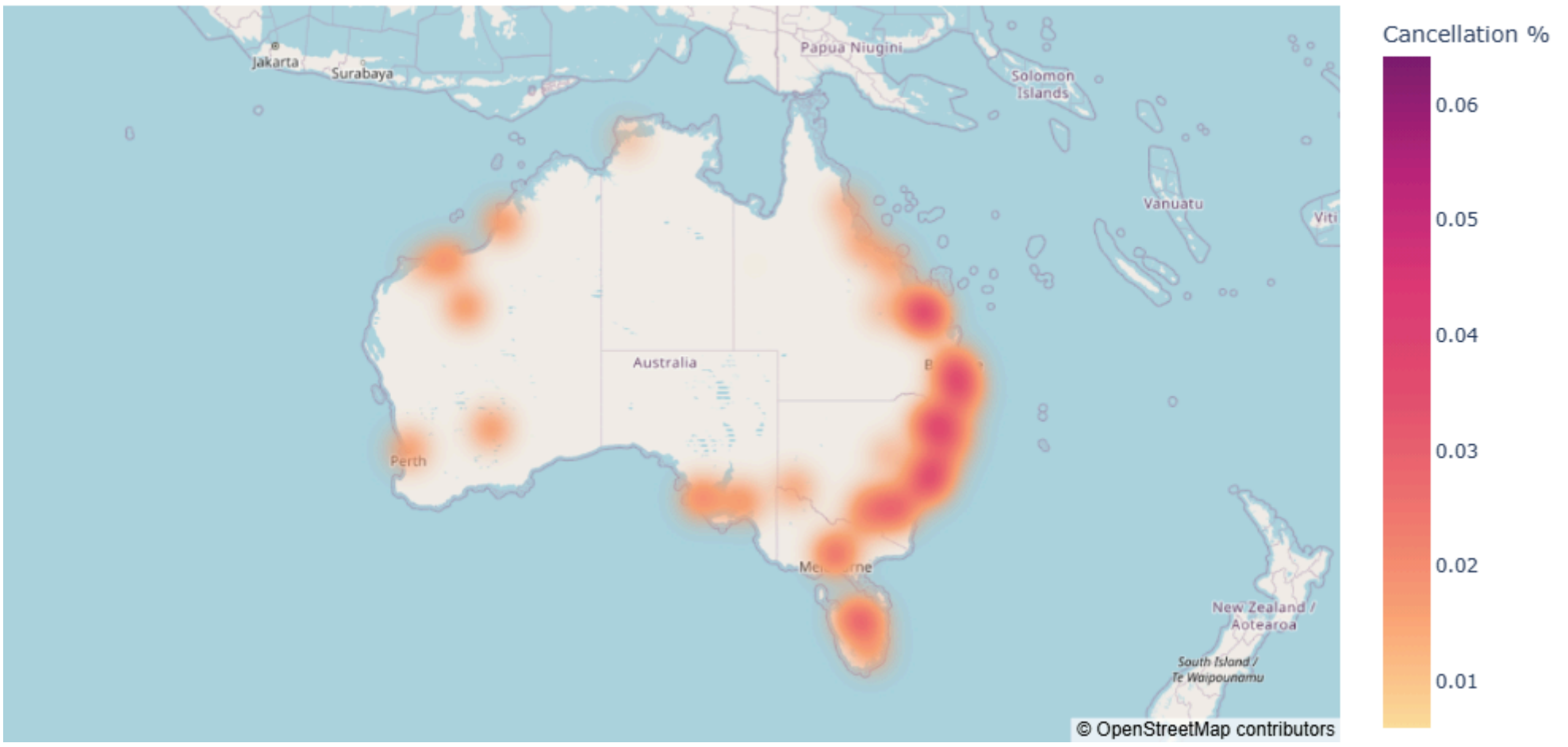

Figure F7. Cancellation Rates by Departing Port 2023

Appendix G. Historical Analysis from 2010 to 2019

Table G.1. The Best and Worst On-time Arrival and Departure Performance of One-way Routes from 2010 to 2019

2010 t0 2012

Best Departure		Worst Departure		Best Arrival		Worst Arrival	
Route	**On-time Departure %**	**Route**	**On-time Departure %**	**Route**	**On-time Arrival %**	**Route**	**On-time Arrival %**
Adelaide-Port Lincoln	92.4%	**Sunshine Coast-Melbourne**	63.4%	**Adelaide-Alice Springs**	91.8%	**Geraldton-Perth**	60.3%
Brisbane-Mount Isa	91.6%	**Geraldton-Perth**	68.7%	**Alice Springs-Adelaide**	90.6%	**Sunshine Coast-Melbourne**	68.1%
Port Lincoln-Adelaide	91.2%	**Perth-Geraldton**	71.0%	**Adelaide-Gold Coast**	90.4%	**Ballina-Sydney**	68.5%
Canberra-Brisbane	90.4%	**Darwin-Brisbane**	71.8%	**Brisbane-Mount Isa**	90.3%	**Perth-Geraldton**	69.9%
Adelaide-Gold Coast	89.8%	**Townsville-Sydney**	72.4%	**Port Lincoln-Adelaide**	89.8%	**Townsville-Cairns**	70.8%

2013 to 2016

Best Departure		Worst Departure		Best Arrival		Worst Arrival	
Route	**On-time Departure %**	**Route**	**On-time Departure %**	**Route**	**On-time Arrival %**	**Route**	**On-time Arrival %**
Darwin-Alice Springs	94.9%	**Alice Springs-Melbourne**	65.0%	**Darwin-Alice Springs**	93.5%	**Port Macquarie-Sydney**	66.8%
Adelaide-Alice Springs	93.4%	**Alice Springs-Sydney**	69.4%	**Perth-Newman**	92.9%	**Sydney-Port Macquarie**	72.2%
Perth-Port Hedland	93.2%	**Sunshine Coast-Melbourne**	70.2%	**Brisbane-Mount Isa**	92.6%	**Sunshine Coast-Melbourne**	72.3%
Perth-Newman	93.2%	**Port Macquarie-Sydney**	72.7%	**Alice Springs-Darwin**	92.5%	**Alice Springs-Melbourne**	72.9%
Brisbane-Mount Isa	93.1%	**Townsville-Sydney**	74.0%	**Adelaide-Alice Springs**	92.3%	**Hobart-Melbourne**	75.2%

2017 to 2019

Best Departure		Worst Departure		Best Arrival		Worst Arrival	
Route	**On-time Departure %**	**Route**	**On-time Departure %**	**Route**	**On-time Arrival %**	**Route**	**On-time Arrival %**
Adelaide-Port Lincoln	92.8%	**Ayers Rock-Sydney**	63.3%	**Cairns-Townsville**	91.6%	**Sunshine Coast-Melbourne**	67.0%
Darwin-Alice Springs	91.9%	**Sunshine Coast-Melbourne**	64.5%	**Adelaide-Gold Coast**	91.5%	**Hobart-Melbourne**	67.4%
Melbourne-Hamilton Island	91.7%	**Townsville-Melbourne**	65.4%	**Townsville-Cairns**	90.9%	**Townsville-Melbourne**	69.2%
Perth-Newman	91.7%	**Townsville-Sydney**	67.8%	**Adelaide-Port Lincoln**	90.6%	**Armidale-Sydney**	69.9%
Cairns-Townsville	91.6%	**Sunshine Coast-Sydney**	70.0%	**Perth-Newman**	90.4%	**Sunshine Coast-Sydney**	70.6%

Table G.2. The Best and Worst On-time Arrival and Departure Performance of Aggregated Routes from 2010 to 2019

2010 to 2012

Best Departure

Route	On-time Departure %
Adelaide-Port Lincoln	91.8%
Brisbane-Mount Isa	90.3%
Adelaide-Gold Coast	89.1%
Brisbane-Canberra	88.9%
Adelaide-Alice Springs	88.6%

Worst Departure

Route	On-time Departure %
Sunshine Coast-Melbourne	68.5%
Geraldton-Perth	69.8%
Sydney-Townsville	74.9%
Brisbane-Darwin	75.4%
Hamilton Island-Sydney	75.4%

Best Arrival

Route	On-time Arrival %
Adelaide-Alice Springs	91.2%
Adelaide-Gold Coast	90.1%
Adelaide-Port Lincoln	89.8%
Adelaide-Canberra	88.9%
Darwin-Perth	87.7%

Worst Arrival

Route	On-time Arrival %
Geraldton-Perth	65.1%
Sunshine Coast-Melbourne	70.7%
Port Macquarie-Sydney	73.4%
Albury-Sydney	73.9%
Ballina-Sydney	74.5%

2013 to 2016

Best Departure

Route	On-time Departure %
Alice Springs-Darwin	93.3%
Newman-Perth	92.7%
Adelaide-Port Lincoln	92.2%
Karratha-Perth	92.1%
Perth-Port Hedland	92.0%

Worst Departure

Route	On-time Departure %
Alice Springs-Melbourne	74.5%
Melbourne-Sunshine Coast	75.0%
Port Macquarie-Sydney	75.8%
Launceston-Sydney	78.1%
Hobart-Melbourne	78.3%

Best Arrival

Route	On-time Arrival %
Alice Springs-Darwin	93.0%
Newman-Perth	92.1%
Darwin-Perth	91.3%
Perth-Port Hedland	91.2%
Karratha-Perth	91.0%

Worst Arrival

Route	On-time Arrival %
Port Macquarie-Sydney	69.5%
Alice Springs-Melbourne	74.9%
Melbourne-Sunshine Coast	76.4%
Armidale-Sydney	76.9%
Hobart-Melbourne	77.2%

2017 to 2019

Best Departure

Route	On-time Departure %
Adelaide-Port Lincoln	91.9%
Cairns-Townsville	91.0%
Newman-Perth	90.7%
Alice Springs-Darwin	90.6%
Kalgoorlie-Perth	89.8%

Worst Departure

Route	On-time Departure %
Melbourne-Townsville	69.9%
Sydney-Townsville	71.2%
Melbourne-Sunshine Coast	72.3%
Sunshine Coast-Sydney	72.3%
Darwin-Sydney	72.6%

Best Arrival

Route	On-time Arrival %
Cairns-Townsville	91.2%
Adelaide-Port Lincoln	90.1%
Newman-Perth	89.8%
Kalgoorlie-Perth	89.4%
Alice Springs-Darwin	89.3%

Worst Arrival

Route	On-time Arrival %
Hobart-Melbourne	71.4%
Melbourne-Townsville	71.9%
Armidale-Sydney	72.5%
Melbourne-Newcastle	73.0%
Melbourne-Sydney	73.1%

Table G.2. The Best and Worst On-time Arrival and Departure Performance of One-way High-traffic Routes from 2010 to 2019

2010 to 2012

Best Departure

Route	On-time Departure %
Adelaide-Port Lincoln	92.4%
Brisbane-Mount Isa	91.6%
Port Lincoln-Adelaide	91.3%
Canberra-Brisbane	90.4%
Adelaide-Gold Coast	89.5%

Worst Departure

Route	On-time Departure %
Sunshine Coast-Melbourne	63.2%
Geraldton-Perth	68.9%
Perth-Geraldton	69.8%
Darwin-Brisbane	71.0%
Townsville-Sydney	71.7%

Best Arrival

Route	On-time Arrival %
Adelaide-Alice Springs	91.6%
Brisbane-Mount Isa	90.3%
Alice Springs-Adelaide	90.1%
Adelaide-Gold Coast	90.0%
Adelaide-Port Lincoln	89.8%

Worst Arrival

Route	On-time Arrival %
Geraldton-Perth	60.4%
Sunshine Coast-Melbourne	67.8%
Ballina-Sydney	68.7%
Perth-Geraldton	69.7%
Townsville-Cairns	70.2%

2013 to 2016

Best Departure

Route	On-time Departure %
Darwin-Alice Springs	94.8%
Perth-Port Hedland	93.3%
Adelaide-Alice Springs	93.2%
Adelaide-Port Lincoln	93.1%
Perth-Newman	93.0%

Worst Departure

Route	On-time Departure %
Alice Springs-Melbourne	64.7%
Alice Springs-Sydney	69.2%
Sunshine Coast-Melbourne	69.4%
Launceston-Sydney	72.5%
Port Macquarie-Sydney	72.7%

Best Arrival

Route	On-time Arrival %
Darwin-Alice Springs	93.4%
Alice Springs-Darwin	93.0%
Perth-Newman	92.7%
Brisbane-Mount Isa	92.5%
Melbourne-Hamilton Island	92.3%

Worst Arrival

Route	On-time Arrival %
Port Macquarie-Sydney	66.4%
Sunshine Coast-Melbourne	71.3%
Sydney-Port Macquarie	72.3%
Alice Springs-Melbourne	73.5%
Launceston-Sydney	74.1%

2017 to 2019

Best Departure

Route	On-time Departure %
Darwin-Alice Springs	92.6%
Adelaide-Port Lincoln	92.4%
Cairns-Townsville	92.2%
Melbourne-Hamilton Island	91.7%
Perth-Newman	91.3%

Worst Departure

Route	On-time Departure %
Ayers Rock-Sydney	59.3%
Townsville-Melbourne	61.9%
Sunshine Coast-Melbourne	64.4%
Townsville-Sydney	67.5%
Launceston-Sydney	67.9%

Best Arrival

Route	On-time Arrival %
Cairns-Townsville	92.2%
Adelaide-Gold Coast	91.3%
Townsville-Cairns	91.1%
Darwin-Alice Springs	90.7%
Perth-Newman	90.2%

Worst Arrival

Route	On-time Arrival %
Sunshine Coast-Melbourne	66.0%
Townsville-Melbourne	66.1%
Hobart-Melbourne	67.3%
Armidale-Sydney	69.3%
Ayers Rock-Sydney	69.4%

Table G.3. Cancellation Rates of Arriving and Departing Port 2010-2019

2010 to 2012

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Karratha	3.0%	21,312
Gladstone	3.0%	8,208
Port Hedland	2.5%	13,286
Canberra	2.5%	113,894
Rockhampton	2.4%	25,240

Arriving port

Arriving port	Cancellation rate	Sectors Scheduled
Gladstone	2.6%	7,700
Karratha	2.5%	21,266
Port Hedland	2.2%	13,280
Canberra	2.2%	113,902
Sydney	2.1%	561,508

2013 to 2016

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Moranbah	5.2%	16,350
Karratha	3.7%	28,968
Tamworth	3.4%	6,044
Gladstone	3.0%	34,154
Canberra	2.9%	151,418

Arriving port

Arriving Port	Cancellation rate	Sectors Scheduled
Moranbah	5.2%	16,738
Karratha	3.2%	28,994
Tamworth	3.1%	6,042
Gladstone	2.8%	33,208
Canberra	2.8%	151,412

2017 to 2019

Departing port

Departing Port	Cancellation rate	Sectors Scheduled
Hamilton Island	3.1%	10,214
Karratha	3.0%	18,298
Sydney	2.8%	640,778
Moranbah	2.8%	2,308
Port Hedland	2.8%	13,508

Arriving port

Arriving Port	Cancellation rate	Sectors Scheduled
Sydney	3.0%	640,850
Hamilton Island	2.9%	10,194
Melbourne	2.8%	518,298
Canberra	2.4%	108,220
Karratha	2.2%	18,336

Table G.4. Cancellation Count of Arriving and Departing Port 2010-2019

2010 to 2012

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	10950	561,218
Melbourne	7952	457,320
Brisbane	4534	352,122
Canberra	2802	113,894
Perth	2166	161,012

Arriving port

Arriving port	Cancellations	Sectors Scheduled
Sydney	12030	561,508
Melbourne	8492	457,472
Brisbane	4818	350,808
Canberra	2498	113,902
Perth	2168	161,174

2013 to 2016

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	17824	829,128
Melbourne	13206	675,636
Brisbane	9732	591,026
Canberra	4424	151,418
Perth	3038	238,294

Arriving port

Arriving Port	Cancellations	Sectors Scheduled
Sydney	19198	830,800
Melbourne	13686	675,754
Brisbane	10270	588,996
Canberra	4180	151,412
Perth	3192	238,596

2017 to 2019

Departing port

Departing Port	Cancellations	Sectors Scheduled
Sydney	18220	640,778
Melbourne	14196	518,286
Brisbane	7188	416,404
Canberra	2808	108,204
Adelaide	2310	164,038

Arriving port

Arriving Port	Cancellations	Sectors Scheduled
Sydney	19374	640,850
Melbourne	14656	518,298
Brisbane	7428	416,246
Canberra	2592	108,220
Perth	2416	169,216

Appendix H. Python Code

Code H.1. Importing Packages and Data

```
import numpy as np
import pandas as pd
import seaborn as sns
import os
import matplotlib.pyplot as plt
import scipy as spx
import math
from pandas.api.types import is_string_dtype
from pandas.api.types import is_numeric_dtype
import csv
import plotly.express as px


#Setting the packages
sns.set(context="notebook", palette="Spectral", style = 'darkgrid' ,font_scale = 1.5, color_codes=True)
import warnings
warnings.filterwarnings('ignore')

#For geomapping and location
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent='myapplication')
from shapely.geometry import Point
import geopandas as gpd
from geopandas import GeoDataFrame
from shapely.geometry import Polygon
```

```
#Importing the excel file
xls = pd.ExcelFile("DataFile.xlsx")
sheets = xls.sheet_names

data_xls = pd.read_excel("DataFile.xlsx", sheet_name = sheets)
```

```
#Extracting Data from the data_xls, in which each year will have their own data frame
OTP2020_df = data_xls['2020-23 OTP']
Y2020_df = data_xls['2020']
Y2019_df = data_xls['2019']
Y2018_df = data_xls['2018']
Y2017_df = data_xls['2017']
Y2016_df = data_xls['2016']
Y2015_df = data_xls['2015']
Y2014_df = data_xls['2014']
Y2013_df = data_xls['2013']
Y2012_df = data_xls['2012']
Y2011_df = data_xls['2011']
Y2010_df = data_xls['2010']

#Creating a list bank of the dataframes to easily conduct data cleaning and transformation
list_allyears = [OTP2020_df,Y2020_df,Y2019_df,Y2018_df,Y2017_df,Y2016_df,Y2015_df,Y2014_df,Y2013_df,Y2012_df,Y2011_df,Y2010_df]
```

Code H.2. Data Cleaning

```
#Renaming the columns appropriately
#We identify that columns with percentages have bad naming format

for i in range(len(list_allyears)):
    for columns in list_allyears[i]:
        list_allyears[i] = list_allyears[i].rename(columns={"OnTime Departures \n(%)":"OnTime Departures %",
                                "OnTime Arrivals \n(%)":"OnTime Arrivals %",
                                "Cancellations \n\n(%)":"Cancellations %",
                                "Month":"Date",
                               })
```

```
#Identifying how many null values are there

for i in range(len(list_allyears)):
    print(i)
    print(list_allyears[i].isna().sum())

#Looking at the results, we find that Y2018 has a null value column which was not intended, called ["Unnamed: 15"]
#This might possibly from importing
```

```
#Furthermore, we will also remove the total columns for all the dataframe called All Ports
#This is so these summations do not

for i in range(len(list_allyears)):
    list_allyears[i] = list_allyears[i][list_allyears[i]["Route"].str.contains("All Ports-All Ports") == False]

#There were also unintended values that were converted with the dataframe in 2012, particularly the last five: index 5231 to 5235

list_allyears[9] = list_allyears[9].drop(index=[5233,5234,5235])
```

```
#Validating the removal of nulls

for i in range(len(list_allyears)):
    print(i)
    print(list_allyears[i].isna().sum())

#We also found out that there are "na" values wihtin the dataset (with 0 departures and arrivals)
#We assume that these cancellations are not indicative of performance related to arrival and departure

#We will be keeping these values as they contribute to the cancellation count.
#Furthermore, there is a valid reason why they are NA. Nevertheless we will keep them in mind all throughout the code.
```

```
#Replacing na with blank

def remove_na(x):
    return(pd.to_numeric(str(x).replace("na",""), errors = 'coerce'))

for i in range(len(list_allyears)):
    for columns in list_allyears[i]:
        list_allyears[i][["OnTime Departures %","OnTime Arrivals %","Cancellations %"]] = list_allyears[i][["OnTime Departures %","OnTime
Arrivals %","Cancellations %"]].applymap(remove_na)
```

```
#Revalidating the dataformats while considering the errors or nulls
#We will be ignoring them in the meantime as they

for i in range(len(list_allyears)):
    list_allyears[i]["Route"].astype(str, errors='ignore')
    list_allyears[i]["Departing Port"].astype(str,errors='ignore')
    list_allyears[i]["Arriving Port"].astype(str,errors='ignore')
    list_allyears[i]["Airline"].astype(str,errors='ignore')
    list_allyears[i]["Date"].astype('datetime64[ns]',errors='ignore')
    list_allyears[i]["Sectors Scheduled"].astype(int,errors='ignore')
    list_allyears[i]["Sectors Flown"].astype(int,errors='ignore')
    list_allyears[i]["Cancellations"].astype(int,errors='ignore')
    list_allyears[i]["Departures On Time"].astype(int,errors='ignore')
    list_allyears[i]["Arrivals On Time"].astype(int,errors='ignore')
    list_allyears[i]["Departures Delayed"].astype(int,errors='ignore')
    list_allyears[i]["Arrivals Delayed"].astype(int,errors='ignore')
    list_allyears[i]["OnTime Departures %"].astype(float,errors='ignore')
    list_allyears[i]["OnTime Arrivals %"].astype(float,errors='ignore')
    list_allyears[i]["Cancellations %"].astype(float,errors='ignore')
```

```
#Knowing if the data formats are properly validated

for i in range(len(list_allyears)):
    print(i)
    print(list_allyears[i].dtypes)
```

Code H.3. Creating New Variables

```
#Creating the function to be applied across the years, creating three columns derived from the date column

def year_var(sheet):
    """Extracting date details from the date column Date"""
    result_year = []
    result_month = []
    result_day = []
    for element in sheet['Date']:
        result_year.append(str(element.year))
        result_month.append(str(element.month))
        result_day.append(str(element.day))
    sheet['Year'] = result_year
    sheet['Month'] = result_month
    sheet['Day'] = result_day
```

```
#Applying the function to all the dataframes

for i in range(len(list_allyears)):
    year_var(list_allyears[i])
```

```
#To keep data stored well, we will extract the dataframe we need for the problem for each year
#Index were reset here since we removed some values and just to be sure, so that functions and element-wise functions can work

df2020 = list_allyears[0][list_allyears[0]["Year"] == "2020"].reset_index(drop=True)
df2021 = list_allyears[0][list_allyears[0]["Year"] == "2021"].reset_index(drop=True)
df2022 = list_allyears[0][list_allyears[0]["Year"] == "2022"].reset_index(drop=True)
df2023 = list_allyears[0][list_allyears[0]["Year"] == "2023"].reset_index(drop=True)

df2019 = list_allyears[2].reset_index(drop=True)
df2018 = list_allyears[3].reset_index(drop=True)
df2017 = list_allyears[4].reset_index(drop=True)
df2016 = list_allyears[5].reset_index(drop=True)
df2015 = list_allyears[6].reset_index(drop=True)
df2014 = list_allyears[7].reset_index(drop=True)
df2013 = list_allyears[8].reset_index(drop=True)
df2012 = list_allyears[9].reset_index(drop=True)
df2011 = list_allyears[10].reset_index(drop=True)
df2010 = list_allyears[11].reset_index(drop=True)

#There were also problems with the Airlines at year 2023, specifically on "virgin Airlines"

for i in range(len(df2023)):
    if df2023["Airline"][i] == "virgin Australia":
        df2023["Airline"][i] = "Virgin Australia"
    else:
        continue
```

```
#Creating all year dataframe

all_years = pd.DataFrame()

for dataframes in (df2020,df2021,df2022,df2023,df2019,df2018,df2017,df2016,df2015,df2014,df2013,df2012,df2011,df2010):
    all_years =  pd.concat([all_years, dataframes], ignore_index=True)
```

Code H.4. Extracting Spatial Data

```
#Extracting coordinates for geomaps, etc.
aggregate_list = {}

for column in list(all_years.columns):
    if is_numeric_dtype(all_years[column]):
        aggr_method = 'sum'
    else:
        aggr_method = 'first'
    aggregate_list.update({column:aggr_method})

port_locations = all_years.groupby('Departing Port').aggregate(aggregate_list).index.values.tolist()
geo_list = []

#We utilise the nominatim geolocator to retrieve the coordinates and polygons
for locate in port_locations:
    location = geolocator.geocode(locate + ", Australia", geometry='wkt')
    geo_list.append([locate,location.latitude,location.longitude,location.raw["geotext"]])
```

```
#Creating a dataframe for geo data to match each route

#For departure
geo_df = pd.DataFrame(geo_list, columns=['Port_dep','latitude_dep','longitude_dep','Polygon'])
geo_df["Polygon"] = gpd.GeoSeries.from_wkt(geo_df["Polygon"])
geo_df = GeoDataFrame(geo_df, geometry=geo_df["Polygon"])

#For arrival
geo_df2 = pd.DataFrame(geo_list, columns=['Port_arr','latitude_arr','longitude_arr','Polygon'])
geo_df2["Polygon"] = gpd.GeoSeries.from_wkt(geo_df2["Polygon"])
geo_df2 = GeoDataFrame(geo_df2, geometry=geo_df2["Polygon"])
```

```
#Merging with the dataframe for all years
#These dataframes will be used in visualising map graphs

all_yearswithcoors = pd.merge(left=all_years,
```

```
    right=geo_df,
    left_on="Departing Port",
    right_on="Port_dep",
    how="left"
)

all_yearswithcoors_all = pd.merge(left=all_yearswithcoors,
    right=geo_df2,
    left_on="Arriving Port",
    right_on="Port_arr",
    how="left"
)

all_yearswithcoors_all = all_yearswithcoors_all.drop(columns=["Polygon_x","Polygon_y","Port_dep","Port_arr"])
```

Code H.5. Question 1: Analysis of One-way Routes on On-time Performance

```
#Creating a function to apply to easily apply to the years

def year_eval(year):
    """The function returns the dataframes that contain the best and worst routes. Kindly assign four values for this function to store best departure, worst departure, best arrival and worst arrival."""
    #We create an aggregation list to properly aggregate the columns of different types
    aggregate_list = {}

    for column in list(year.columns):
        if column in ["latitude_dep","longitude_dep","latitude_arr","longitude_arr"]:
            aggr_method = 'first'
        elif is_numeric_dtype(year[column]):
            aggr_method = 'sum'
        else:
            aggr_method = 'first'
        aggregate_list.update({column:aggr_method})

    #We calculate the main metric using the grouped by data frame
    base = year.groupby('Route').aggregate(aggregate_list)
    base['Aggregated Departures'] = base["Departures On Time"]/base["Sectors Flown"]
    base['Aggregated Arrivals'] = base["Arrivals On Time"]/base["Sectors Flown"]

    #We base the best and worst performance on aggregated arrivals and departures over their sectors flown.
    prelist_1 = base.sort_values(['Aggregated Departures','Sectors Flown'], ascending = False).head(5)[['Aggregated Departures']]
    prelist_2 = base.sort_values(['Aggregated Departures','Sectors Flown'], ascending = True).head(5)[['Aggregated Departures']]

    prelist_3 = base.sort_values(['Aggregated Arrivals','Sectors Flown'], ascending = False).head(5)[['Aggregated Arrivals']]
    prelist_4 = base.sort_values(['Aggregated Arrivals','Sectors Flown'], ascending = True).head(5)[['Aggregated Arrivals']]

    #the funciton returns the list and the
    return prelist_1,prelist_2,prelist_3,prelist_4
```

```
#Legend: dep - departure, arr - arrival
#For year 2020
bestdep2020,wordep2020,bestarr2020,worarr2020 = year_eval(df2020)

#For year 2021
bestdep2021,wordep2021,bestarr2021,worarr2021 = year_eval(df2021)

#For year 2022
bestdep2022,wordep2022,bestarr2022,worarr2022 = year_eval(df2022)

#For year 2023
bestdep2023,wordep2023,bestarr2023,worarr2023 = year_eval(df2023)
```

```
#The code below outputs the dataframe containing the best (bes) departure (dep) routes in 2020
bestdep2020
```

Code H.6. Question 2: Analysis of Aggregated Routes on On-time Performance

```
def route_aggr(year):
    """The function aggregates the similar routes into one. """
    year['Route ID'] = 0
    count = 1
```

```
    #Initially, the function creates a new column and uses a loop to mark the routes that are similar to each other.
    for i in range(len(year)):
        if year["Route ID"][i] != 0:
            continue
        else:
            year["Route ID"][i] = count
            for j in range(len(year)):
                if year["Departing Port"][i] == year["Arriving Port"][j] and year["Arriving Port"][i] == year["Departing Port"][j]:
                    year["Route ID"][j] = '%s' % (year["Route"][i])
                elif year["Departing Port"][i] == year["Departing Port"][j] and year["Arriving Port"][i] == year["Arriving Port"][j]:
                    year["Route ID"][j] = '%s' % (year["Route"][i])
                else:
                    continue
        count += 1

def year_eval_agg(year):
    """The function evaluates the dataset to determine the best and worst departure and arrival perfoamnce with aggregated routes. Kindly
assign four values for this function to store best departure, worst departure, best arrival and worst arrival."""

    #We apply the same process and methods from question 1, just with different metrics.
    aggregate_list = {}

    for column in list(year.columns):
        if column in ["latitude_dep","longitude_dep","latitude_arr","longitude_arr"]:
            aggr_method = 'first'
        elif is_numeric_dtype(year[column]):
            aggr_method = 'sum'
        else:
            aggr_method = 'first'
        aggregate_list.update({column:aggr_method})

    base = year.groupby('Route ID').aggregate(aggregate_list)
    base['Aggregated Departures'] = base["Departures On Time"]/base["Sectors Flown"]
    base['Aggregated Arrivals'] = base["Arrivals On Time"]/base["Sectors Flown"]

    prelist_1 = base.sort_values(['Aggregated Departures','Sectors Flown'], ascending = False).head(5)[['Aggregated Departures']]
    prelist_2 = base.sort_values(['Aggregated Departures','Sectors Flown'], ascending = True).head(5)[['Aggregated Departures']]

    prelist_3 = base.sort_values(['Aggregated Arrivals','Sectors Flown'], ascending = False).head(5)[['Aggregated Arrivals']]
    prelist_4 = base.sort_values(['Aggregated Arrivals','Sectors Flown'], ascending = True).head(5)[['Aggregated Arrivals']]

    #The function then returns the four list for
    return prelist_1,prelist_2,prelist_3,prelist_4
```

```
#Prepairing aggregation for the years by utilising the function
#It creates a column inside the dataframe
route_aggr(df2020)
route_aggr(df2021)
route_aggr(df2022)
route_aggr(df2023)
```

```
#Legend: dep - departure, arr - arrival, agg - aggregated routes
#For year 2020
bestdep2020_agg,wordep2020_agg,bestarr2020_agg,worarr2020_agg = year_eval_agg(df2020)

#For year 2021
bestdep2021_agg,wordep2021_agg,bestarr2021_agg,worarr2021_agg = year_eval_agg(df2021)

#For year 2022
bestdep2022_agg,wordep2022_agg,bestarr2022_agg,worarr2022_agg = year_eval_agg(df2022)

#For year 2023
bestdep2023_agg,wordep2023_agg,bestarr2023_agg,worarr2023_agg = year_eval_agg(df2023)
```

```
#The code below outputs the dataframe containing the best (bes) departure (dep) aggregated (agg) routes in 2020
bestdep2020_agg
```

Code H.7. Question 3: Analysis of High-traffic One-way Routes on On-time Performance

```
def quantile_eval(year,quantile):
    """The function creates a column to know whether the sectors flown of a specific row is above or equal to the first quartile relative to its
route."""

    aggregate_list = {}
```

```
for column in list(year.columns):
    if column in ["latitude_dep","longitude_dep","latitude_arr","longitude_arr"]:
        aggr_method = 'first'
    elif is_numeric_dtype(year[column]):
        aggr_method = 'sum'
    else:
        aggr_method = 'first'
    aggregate_list.update({column:aggr_method})

route_list = year.groupby('Route ID').aggregate(aggregate_list).index.values.tolist()
quantile_dict = {}

#The dictionary allows having a storage for the quantile based on the route (aggregated).
for route in route_list:
    quantile_dict.update({ route: np.quantile(year[(year["Route ID"] == route)]['Sectors Flown'], quantile)})

#The process is basically treating each as False initially but evaluating each element if they are above the quantile.
year['Above Quantile'] = False

#The code below turns the False into True if their sectors flown are above the quantile given.
for routeID,quantile in quantile_dict.items():
    for i in range(len(year)):
        if year["Route ID"][i] == routeID:
            if int(year['Sectors Flown'][i]) >= float(quantile):
                year['Above Quantile'][i] = True
            else:
                continue
        else:
            continue
```

```
#Prepairing the filter for those above or equal to the 75th percentile per route
#The function below creates columns inside the dataframe
quantile_eval(df2020,.75)
quantile_eval(df2021,.75)
quantile_eval(df2022,.75)
quantile_eval(df2023,.75)
```

```
#Legend: dep - departure, arr - arrival, quan - quantile routes
#Through the above quantile column we can filter those that are below the 75th percentile

#For year 2020
bestdep2020_quan,wordep2020_quan,bestarr2020_quan,worarr2020_quan = year_eval(df2020[df2020['Above Quantile']==True])

#For year 2021
bestdep2021_quan,wordep2021_quan,bestarr2021_quan,worarr2021_quan = year_eval(df2021[df2021['Above Quantile']==True])

#For year 2022
bestdep2022_quan,wordep2022_quan,bestarr2022_quan,worarr2022_quan = year_eval(df2022[df2022['Above Quantile']==True])

#For year 2023
bestdep2023_quan,wordep2023_quan,bestarr2023_quan,worarr2023_quan = year_eval(df2023[df2023['Above Quantile']==True])
```

Code H.8. Question 4: Analysis of Cancellation and Cancellation Rates of Ports

```
def cancellation_eval(year):
    """The function evaluates a dataframe to determine the cancecllation performance for arrival and departure ports. Kindly assign four values for this function to store highest cancellation rates in departing port, highest cancellation rates in arriving port, highest cancellation count in departing port, and highest cancellation count in arriving port."""

    #The initial step is calculating the aggregated cancellations based on ports.

    aggregate_list = {}

    for column in list(year.columns):
        if column in ["latitude_dep","longitude_dep","latitude_arr","longitude_arr"]:
            aggr_method = 'first'
        elif is_numeric_dtype(year[column]):
            aggr_method = 'sum'
        else:
            aggr_method = 'first'
        aggregate_list.update({column:aggr_method})

    #The difference in this code is that the metric is based on cancellations over sectors scheduled to fly
```

```
    base_depart = year.groupby('Departing Port').aggregate(aggregate_list)
    base_depart['Aggregated Cancellations Depart'] = base_depart["Cancellations"]/base_depart["Sectors Scheduled"]

    base_arrive = year.groupby('Arriving Port').aggregate(aggregate_list)
    base_arrive['Aggregated Cancellations Arrive'] = base_arrive["Cancellations"]/base_arrive["Sectors Scheduled"]

    prelist_1 = base_depart.sort_values(['Aggregated Cancellations Depart','Sectors Scheduled'], ascending = False).head(5)[['Aggregated
Cancellations Depart','Sectors Scheduled']]
    prelist_2 = base_arrive.sort_values(['Aggregated Cancellations Arrive','Sectors Scheduled'], ascending = False).head(5)[['Aggregated
Cancellations Arrive','Sectors Scheduled']]

    prelist_3 = base_depart.sort_values(['Cancellations','Sectors Scheduled'], ascending = False).head(5)[['Cancellations','Sectors Scheduled']]
    prelist_4 = base_arrive.sort_values(['Cancellations','Sectors Scheduled'], ascending = False).head(5)[['Cancellations','Sectors Scheduled']]

    return prelist_1,prelist_2,prelist_3,prelist_4
```

```
#Legend: dep - departure, arr - arrival
#For here, 'raw' means the raw count of cancecllations while the 'per' is the percentage.

#For year 2020
candep2020per,canarr2020per,candep2020raw,canarr2020raw = cancellation_eval(df2020)

#For year 2021
candep2021per,canarr2021per,candep2021raw,canarr2021raw = cancellation_eval(df2021)

#For year 2022
candep2022per,canarr2022per,candep2022raw,canarr2022raw = cancellation_eval(df2022)

#For year 2023
candep2023per,canarr2023per,candep2023raw,canarr2023raw = cancellation_eval(df2023)
```

```
#The code below outputs the dataframe containing the five departuring ports (dep) with the highest cancellation (can) rates % (per) in 2020
candep2020per
```

Code H.9. Visualisations

Code H.8.1. Routes using Bar Charts

```
#We create a function for the bargraph to be applied across the years

def bargraph(yeardata,x_axis,y_axis,barcolor,i,title):
    """Creates a bargraph for the best and worst for 2020 to 2023"""

    bar = sns.barplot(ax=axes[i], x=x_axis, y=y_axis, data=yeardata, color=barcolor, edgecolor=None)
    sns.despine(bottom = True, left = True)
    bar.set(xlabel = None,ylabel =None)
    bar.set_title(title, fontsize = 13,color = barcolor, weight='bold')
    bar.xaxis.label.set_color('black')
    bar.yaxis.label.set_color('black')
    bar.set_xticklabels(bar.get_xticklabels(), rotation=75)
    bar.set_yticks([])
    for i in bar.containers:
        bar.bar_label(i,fontsize=10,fmt=lambda x: f'{round(x*100,1)}%',color="black")
```

```
#Best Departure Route across the years

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

#We utilise subplots to include all the years
fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Routes with the Best Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

#the function is applied here where we can also customise the color, title and what position in the subplot the plot is
bargraph(bestdep2020,"Route","Aggregated Departures","#FF7844",0,"2020")
bargraph(bestdep2021,"Route","Aggregated Departures","#A64942",1,"2021")
bargraph(bestdep2022,"Route","Aggregated Departures","#53354A",2,"2022")
bargraph(bestdep2023,"Route","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Best Arrival Route

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})
```

```
fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Routes with the Best Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(bestarr2020,"Route","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(bestarr2021,"Route","Aggregated Arrivals","#A64942",1,"2021")
bargraph(bestarr2022,"Route","Aggregated Arrivals","#53354A",2,"2022")
bargraph(bestarr2023,"Route","Aggregated Arrivals","#1B1F3A",3,"2023")
```

```
#Worst Departure Route

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Routes with the Worst Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(wordep2020,"Route","Aggregated Departures","#FF7844",0,"2020")
bargraph(wordep2021,"Route","Aggregated Departures","#A64942",1,"2021")
bargraph(wordep2022,"Route","Aggregated Departures","#53354A",2,"2022")
bargraph(wordep2023,"Route","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Worst Arrival Route across the years

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Routes with the Worst Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(worarr2020,"Route","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(worarr2021,"Route","Aggregated Arrivals","#A64942",1,"2021")
bargraph(worarr2022,"Route","Aggregated Arrivals","#53354A",2,"2022")
bargraph(worarr2023,"Route","Aggregated Arrivals","#1B1F3A",3,"2023")
```

```
#Best Aggregated Dep

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Aggregated Routes with the Best Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(bestdep2020_agg,"Route ID","Aggregated Departures","#FF7844",0,"2020")
bargraph(bestdep2021_agg,"Route ID","Aggregated Departures","#A64942",1,"2021")
bargraph(bestdep2022_agg,"Route ID","Aggregated Departures","#53354A",2,"2022")
bargraph(bestdep2023_agg,"Route ID","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Best Aggregated Arr

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Aggregated Routes with the Best Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(bestarr2020_agg,"Route ID","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(bestarr2021_agg,"Route ID","Aggregated Arrivals","#A64942",1,"2021")
bargraph(bestarr2022_agg,"Route ID","Aggregated Arrivals","#53354A",2,"2022")
bargraph(bestarr2023_agg,"Route ID","Aggregated Arrivals","#1B1F3A",3,"2023")
```

```
#Worst Aggregated Dep

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Aggregated Routes with the Worst Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')
```

```
bargraph(wordep2020_agg,"Route ID","Aggregated Departures","#FF7844",0,"2020")
bargraph(wordep2021_agg,"Route ID","Aggregated Departures","#A64942",1,"2021")
bargraph(wordep2022_agg,"Route ID","Aggregated Departures","#53354A",2,"2022")
bargraph(wordep2023_agg,"Route ID","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Worst Aggregated Arrival

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('Aggregated Routes with the Worst Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(worarr2020_agg,"Route ID","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(worarr2021_agg,"Route ID","Aggregated Arrivals","#A64942",1,"2021")
bargraph(worarr2022_agg,"Route ID","Aggregated Arrivals","#53354A",2,"2022")
bargraph(worarr2023_agg,"Route ID","Aggregated Arrivals","#1B1F3A",3,"2023")
```

```
#Best Hightraffic Departure

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('High-traffic Routes with the Best Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(bestdep2020_quan,"Route","Aggregated Departures","#FF7844",0,"2020")
bargraph(bestdep2021_quan,"Route","Aggregated Departures","#A64942",1,"2021")
bargraph(bestdep2022_quan,"Route","Aggregated Departures","#53354A",2,"2022")
bargraph(bestdep2023_quan,"Route","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Best Hightraffic Arrival

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('High-traffic Routes with the Best Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(bestarr2020_quan,"Route","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(bestarr2021_quan,"Route","Aggregated Arrivals","#A64942",1,"2021")
bargraph(bestarr2022_quan,"Route","Aggregated Arrivals","#53354A",2,"2022")
bargraph(bestarr2023_quan,"Route","Aggregated Arrivals","#1B1F3A",3,"2023")
```

```
#Worst Hightraffic Departure

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('High-traffic Routes with the Worst Departures on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(wordep2020_quan,"Route","Aggregated Departures","#FF7844",0,"2020")
bargraph(wordep2021_quan,"Route","Aggregated Departures","#A64942",1,"2021")
bargraph(wordep2022_quan,"Route","Aggregated Departures","#53354A",2,"2022")
bargraph(wordep2023_quan,"Route","Aggregated Departures","#1B1F3A",3,"2023")
```

```
#Worst Hightraffic Arrival

sns.set_style("white")
sns.set_context("paper", rc={"font.size":15,"axes.titlesize":20,"axes.labelsize":15})

fig, axes = plt.subplots(1, 4, figsize=(15, 3))
plt.subplots_adjust(wspace=.05, hspace=0)
fig.suptitle('High-traffic Routes with the Worst Arrivals on Time (%) by Years', fontsize=16, y=1.08,weight='bold')

bargraph(worarr2020_quan,"Route","Aggregated Arrivals","#FF7844",0,"2020")
bargraph(worarr2021_quan,"Route","Aggregated Arrivals","#A64942",1,"2021")
bargraph(worarr2022_quan,"Route","Aggregated Arrivals","#53354A",2,"2022")
bargraph(worarr2023_quan,"Route","Aggregated Arrivals","#1B1F3A",3,"2023")
```

Code H.9.2. Map Visualisations

```
#In this part, we will be creating visulisations utilising the coordinates from geolocator
#We divide the geodataframe by the years and store them in new dataframes
df2020_geo = all_yearswithcoors_all[all_yearswithcoors_all["Year"] == 2020].reset_index(drop=True)
df2021_geo = all_yearswithcoors_all[all_yearswithcoors_all["Year"] == 2021].reset_index(drop=True)
df2022_geo = all_yearswithcoors_all[all_yearswithcoors_all["Year"] == 2022].reset_index(drop=True)
df2023_geo = all_yearswithcoors_all[all_yearswithcoors_all["Year"] == 2023].reset_index(drop=True)
```

```
#We create a copy of the cancellation code as we will be needing the full dataset instead of just a list

def cancellation_eval_geo(year):
    """The function evaluates a dataframe to determine the cancecllation performance for arrival and departure ports."""

    """The initial step is calculating the aggregated cancellations based on ports."""

    aggregate_list = {}

    for column in list(year.columns):
        if column in ["latitude_dep","longitude_dep","latitude_arr","longitude_arr"]:
            aggr_method = 'first'
        elif is_numeric_dtype(year[column]):
            aggr_method = 'sum'
        else:
            aggr_method = 'first'
        aggregate_list.update({column:aggr_method})

    base_depart = year.groupby('Departing Port').aggregate(aggregate_list)
    base_depart['Aggregated Cancellations Depart'] = base_depart["Cancellations"]/base_depart["Sectors Scheduled"]

    base_arrive = year.groupby('Arriving Port').aggregate(aggregate_list)
    base_arrive['Aggregated Cancellations Arrive'] = base_arrive["Cancellations"]/base_arrive["Sectors Scheduled"]

    #here we only need the base data that includes both the cancellation rates and sum of cancellations including their coordinates

    all_listdep_per = base_depart.sort_values(['Aggregated Cancellations Depart','Sectors Scheduled'], ascending = False)
    all_listarr_per = base_arrive.sort_values(['Aggregated Cancellations Arrive','Sectors Scheduled'], ascending = False)

    return all_listdep_per,all_listarr_per
```

```
#Applying the cancellation eval function with the geo dataframe

#For year 2020
alldep2020per_geo,allarr2020per_geo = cancellation_eval_geo(df2020_geo)

#For year 2021
alldep2021per_geo,allarr2021per_geo = cancellation_eval_geo(df2021_geo)

#For year 2022
alldep2022per_geo,allarr2022per_geo = cancellation_eval_geo(df2022_geo)

#For year 2023
alldep2023per_geo,allarr2023per_geo = cancellation_eval_geo(df2023_geo)

alldep2020per_geo = alldep2020per_geo.rename(columns={"Aggregated Cancellations Depart": 'Cancellation %'})
allarr2020per_geo = allarr2020per_geo.rename(columns={"Aggregated Cancellations Arrive": 'Cancellation %'})
alldep2021per_geo = alldep2021per_geo.rename(columns={"Aggregated Cancellations Depart": 'Cancellation %'})
allarr2021per_geo = allarr2021per_geo.rename(columns={"Aggregated Cancellations Arrive": 'Cancellation %'})
alldep2022per_geo = alldep2022per_geo.rename(columns={"Aggregated Cancellations Depart": 'Cancellation %'})
allarr2022per_geo = allarr2022per_geo.rename(columns={"Aggregated Cancellations Arrive": 'Cancellation %'})
alldep2023per_geo = alldep2023per_geo.rename(columns={"Aggregated Cancellations Depart": 'Cancellation %'})
allarr2023per_geo = allarr2023per_geo.rename(columns={"Aggregated Cancellations Arrive": 'Cancellation %'})
```

```
#For 2020: Cancellations

cn_dep = px.density_mapbox(alldep2020per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                           center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                           mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Departing Ports in 2020",
    title_font_color="black",title_font_weight="bold")

cn_dep.show()
```

```
#For 2020: Cancellations %

cn_dep = px.density_mapbox(alldep2020per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Departing Ports in 2020",
    title_font_color="black",title_font_weight="bold")

cn_dep.show()
```

```
#For 2021: Cancellations

cn_dep = px.density_mapbox(alldep2021per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Departing Ports in 2021",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2021: Cancellations %

cn_dep = px.density_mapbox(alldep2021per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Departing Ports in 2021",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2022: Cancellations

cn_dep = px.density_mapbox(alldep2022per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Departing Ports in 2022",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2022: Cancellations %

cn_dep = px.density_mapbox(alldep2022per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Departing Ports in 2022",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2023: Cancellations

cn_dep = px.density_mapbox(alldep2023per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Departing Ports in 2023",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2023: Cancellations %

cn_dep = px.density_mapbox(alldep2023per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Departing Ports in 2023",
    title_font_color="black",title_font_weight="bold")
```

```
cn_dep.show()
```

```
#For 2020: Cancellations

cn_dep = px.density_mapbox(allarr2020per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Arriving Ports in 2020",
    title_font_color="black",title_font_weight="bold")

cn_dep.show()
```

```
#For 2020: Cancellations %

cn_dep = px.density_mapbox(allarr2020per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Arriving Ports in 2020",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2021: Cancellations

cn_dep = px.density_mapbox(allarr2021per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Arriving Ports in 2021",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2021: Cancellations %

cn_dep = px.density_mapbox(allarr2021per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Arriving Ports in 2021",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2022: Cancellations

cn_dep = px.density_mapbox(allarr2022per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation for Arriving Ports in 2022",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2022: Cancellations %

cn_dep = px.density_mapbox(allarr2022per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Arriving Ports in 2022",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2023: Cancellations

cn_dep = px.density_mapbox(allarr2023per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellations", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
```

```
    title_text="Analysis of Cancellation for Arriving Ports in 2023",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

```
#For 2023: Cancellations %

cn_dep = px.density_mapbox(allarr2023per_geo, lat='latitude_dep', lon='longitude_dep', z="Cancellation %", radius=30,
                  center=dict(lat=-27.5, lon=133.7751), zoom=2.7,
                  mapbox_style="open-street-map", color_continuous_scale="sunsetdark")
cn_dep.update_layout(
    title_x=0.5,title_y=0.95,margin={"l": 0, "r": 0, "b": 0, "t": 60},
    title_text="Analysis of Cancellation % for Arriving Ports in 2023",
    title_font_color="black",title_font_weight="bold")
cn_dep.show()
```

Code H.10. Historical Analysis

```
#Here, we also analysed the other years by using the function from the first four questions
#Creating sheets for year periods. We grouped them by global events

df201012 = pd.concat([df2010, df2011,df2012], ignore_index=True) #post recovery from the global financial crises
df201316 = pd.concat([df2013, df2014,df2015,df2016], ignore_index=True) #years of stability but heightened security (MH incident)
df201719 = pd.concat([df2017, df2018,df2019], ignore_index=True) #geopolitical recession, and years before pandemic
```

```
#Question 1

#For year 2010
bestdep201012,wordep201012,bestarr201012,worarr201012,dep201012,arr201012 = year_eval(df201012)

#For year 2013
bestdep201316,wordep201316,bestarr201316,worarr201316,dep201316,arr201316 = year_eval(df201316)

#For year 2017
bestdep201719,wordep201719,bestarr201719,worarr201719,dep201719,arr201719 = year_eval(df201719)
```

```
#Question 2

route_aggr(df201012)
route_aggr(df201316)
route_aggr(df201719)
```

```
#For year 2010
bestdep201012_agg,wordep201012_agg,bestarr201012_agg,worarr201012_agg,dep201012_agg,arr201012_agg = year_eval_agg(df201012)

#For year 2013
bestdep201316_agg,wordep201316_agg,bestarr201316_agg,worarr201316_agg,dep201316_agg,arr201316_agg = year_eval_agg(df201316)

#For year 2017
bestdep201719_agg,wordep201719_agg,bestarr201719_agg,worarr201719_agg,dep201719_agg,arr201719_agg = year_eval_agg(df201719)
```

```
#Question 3
quantile_eval(df201012,.75)
quantile_eval(df201316,.75)
quantile_eval(df201719,.75)
```

```
#For year 2020
bestdep201012_quan,wordep201012_quan,bestarr201012_quan,worarr201012_quan,dep201012_quan,arr201012_quan =
year_eval(df201012[df201012['Above Quantile']==True])

#For year 2013
bestdep201316_quan,wordep201316_quan,bestarr201316_quan,worarr201316_quan,dep201316_quan,arr201316_quan =
year_eval(df201316[df201316['Above Quantile']==True])

#For year 2017
bestdep201719_quan,wordep201719_quan,bestarr201719_quan,worarr201719_quan,dep201719_quan,arr201719_quan =
year_eval(df201719[df201719['Above Quantile']==True])
```

```
#Question 4

#For year 2010
candep201012per,canarr201012per,candep201012raw,canarr201012raw,alldep201012per,allarr201012per,alldep201012raw,allarr201012raw
= cancellation_eval(df201012)

#For year 2013
candep201316per,canarr201316per,candep201316raw,canarr201316raw,alldep201316per,allarr201316per,alldep201316raw,allarr201316raw
```

```
= cancellation_eval(df201316)

#For year 2017
candep201719per,canarr201719per,candep201719raw,canarr201719raw,alldep201719per,allarr201719per,alldep201719raw,allarr201719raw
= cancellation_eval(df201719)
```

END OF CODE