Dublin Road Speeds

Ruaridh Williamson

Supervised by Dessie Petrova

SATALIA

Context

Explore the historical travel speeds of Dublin's roads

- **Shapefile** containing the geometry of road links located in Dublin. The speed for each road link is recorded at 15 minute intervals over one Thursday.
- **CSV** containing the road link attribute data
  - Road link length (metres)
  - Function class (road type)
  - Urban/Rural flag
  - Travel direction (True or False)
  - Road speed (km/h every 15min from 0:00 to 23:45)

Agenda

- **Initial exploration**
  - Variable summaries and distributions
  - Confirmatory analysis
- **Understanding the dataset**
  - What caveats apply?
  - How does it look like the data has been measured and treated?
  - Variable creation
- **Use cases**
  - Visualising speeds over time
  - Traffic levels
  - Network algorithms
  - Commutability rating

Commutability rating = index of time from all roads to a given point in the CBD

Initial Exploration

Primary key

Start with CSV

What is the dataset *Point of View*?

- Road Link
- Travel Direction

...almost

	Trav Dir		
Link Id	False	True	G..
549454034	1	1	2
549454061	1	1	2
549454069	1	1	2
549454092	1	1	2
549454099	1	1	2
549454100	1	1	2
549454103	1	1	2
549454128	1	1	2
549454133	1	1	2
1143298779	1		1
1143298780	1		1
1143298781	1		1
1143298782		1	1
1143298783		1	1
1143298784		1	1
1143298785		1	1
1143298786		1	1

Point of View = Primary Key

For nearly all road links there are two travel directions however this Primary Key is not strict as some roads are one-way only.

Initial Exploration

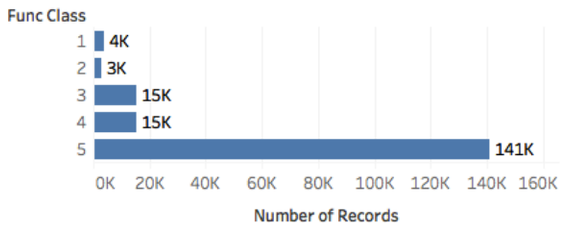
Primary key

Start with CSV

What is the dataset *Point of View*?

- Road Link
- Travel Direction

…almost



	Trav Dir		
Link Id	False	True	G..
549454034	1	1	2
549454061	1	1	2
549454069	1	1	2
549454092	1	1	2
549454099	1	1	2
549454100	1	1	2
549454103	1	1	2
549454128	1	1	2
549454133	1	1	2
1143298779	1		1
1143298780	1		1
1143298781	1		1
1143298782		1	1
1143298783		1	1
1143298784		1	1
1143298785		1	1
1143298786		1	1

The vast majority of roads are Class 5, with just a small proportion making up Class 1 and 2

Initial Exploration

Tidy data

Convert from *wide* to *long* format

```
# Melt data columns labelled as u00_00 ... u23_45 into one variable "Time" with value "Speed"
melted_dt <- data.table::melt(dublin_csv, measure.vars = grep("u\\d", names(dublin_csv), value = TRUE),
                              variable.name = "Time", value.name = "Speed")

# Convert Time variable into datetime representation
melted_dt <- melted_dt[, Time := as.POSIXct(Time, format = "u%H_%M")]
```

PoV is now Link (100k) by Time (96) by Direction (2)

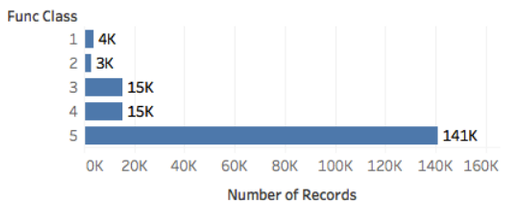
Nearly 20m rows, ~1.5GB on disk

Use data.table to pivot the 96 time columns into two columns

Initial Exploration

What does Function Class represent?

Distribution of Function Class (indicates the road category)



	Func Class				
	1	2	3	4	5
Avg. U00 00	64	46	45	42	24
Avg. U00 15	63	46	45	42	24
Avg. U00 30	63	46	45	42	24
Avg. U00 45	63	46	45	42	24
Avg. U01 00	63	46	45	42	24
Avg. U01 15	63	46	45	42	24
Avg. U01 30	63	46	45	42	24

Average Speed (km/h) by time and Func Class

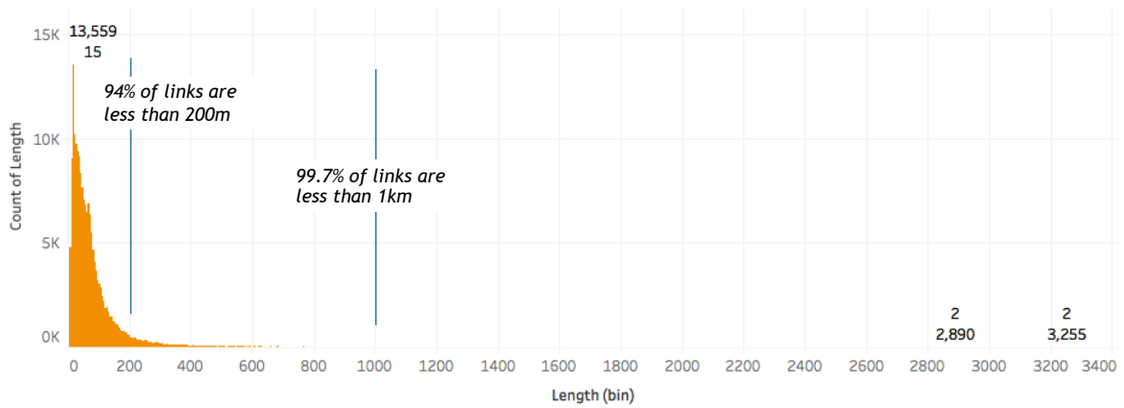
… so 1 is a highway and 5 is suburban

Comparing the frequency chart with the avg speeds we can conclude that Class 1 represents major highways and Class 5 represents small suburban roads and inner-city laneways.

Initial Exploration

How are the road lengths distributed?

Distribution of Road Lengths



Road Lengths (metres) grouped by 5m bins

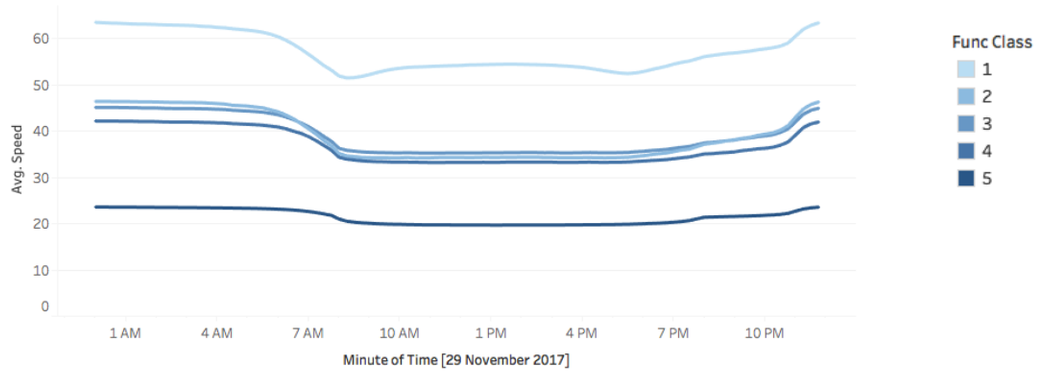
The vast majority of road links are less than 100-200m.

This indicates that all roads are broken up into very small fragments, likely at turns and intersections.

Initial Exploration

How are the road speeds distributed?

Distribution of Speed over time



Shows effect of morning rush hour, traffic during the day and afternoon peak

Understanding the Dataset

Grouping 15min intervals throughout the day

3rd party pre-processing

Dataset Treatment

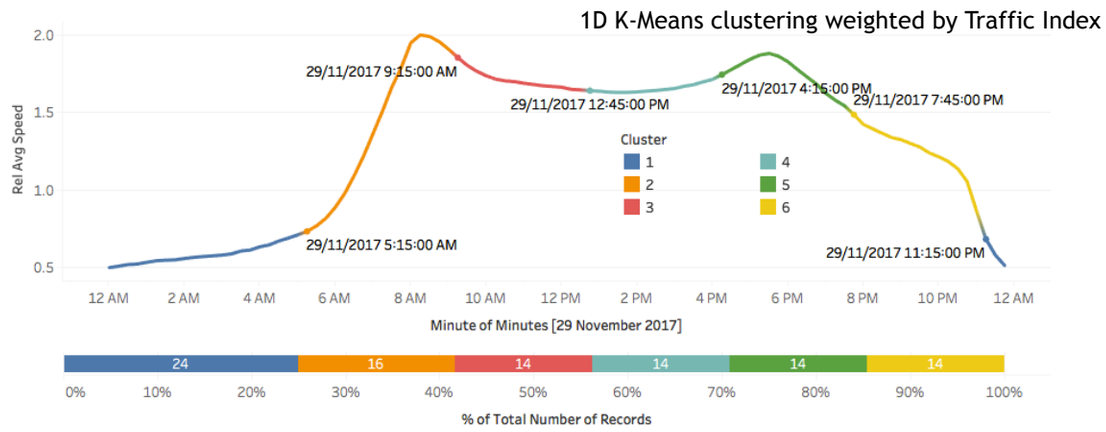
Clustering times of the day for collective analysis

Traffic Index over time by Cluster

Dataset Treatment

Clustering times of the day for collective analysis

Traffic Index over time by Cluster



Clustering algorithm using the R package “Ckmeans.1d.dp” (see https://cran.r-project.org/web/packages/Ckmeans.1d.dp/)

> Weighted univariate k-means algorithm can optimally segment time series and perform peak calling

Clusters must be contiguous on time dimension with weights according to the Traffic Index

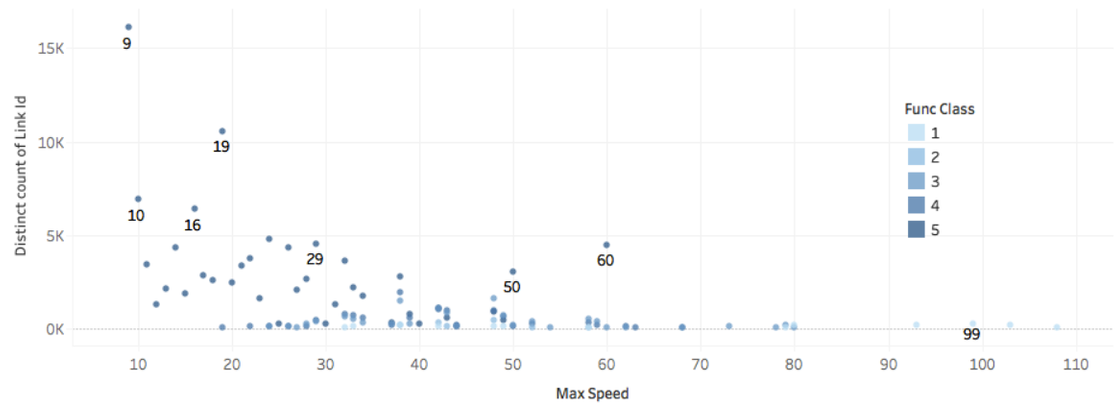
Time intervals come out reasonably evenly distributed

The tail at 23:15 is manually added to Cluster 1 to complete the cycle

Dataset Treatment

Would we expect huge counts of rounded numbers?

Distribution of Max Speed (a continuous variable with no binning…)



Speeds where the count < 100 have been hidden

Should not expect to see such high peaks on a continuous variable

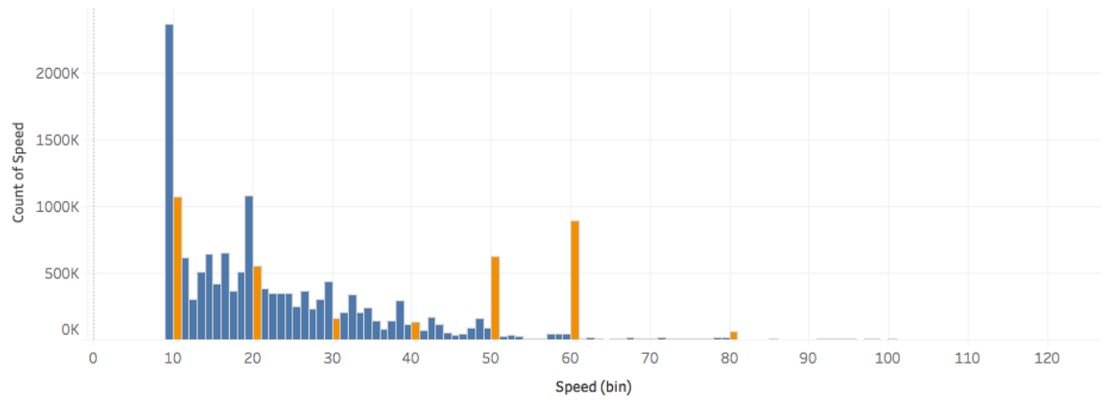
Data points are integers rather than decimals.

Abnormal peaks at 50 and 60 km/h suggest default speed values

Dataset Treatment

Speed intervals of 10 highlighted

Distribution of Speed (a continuous variable with no binning…)



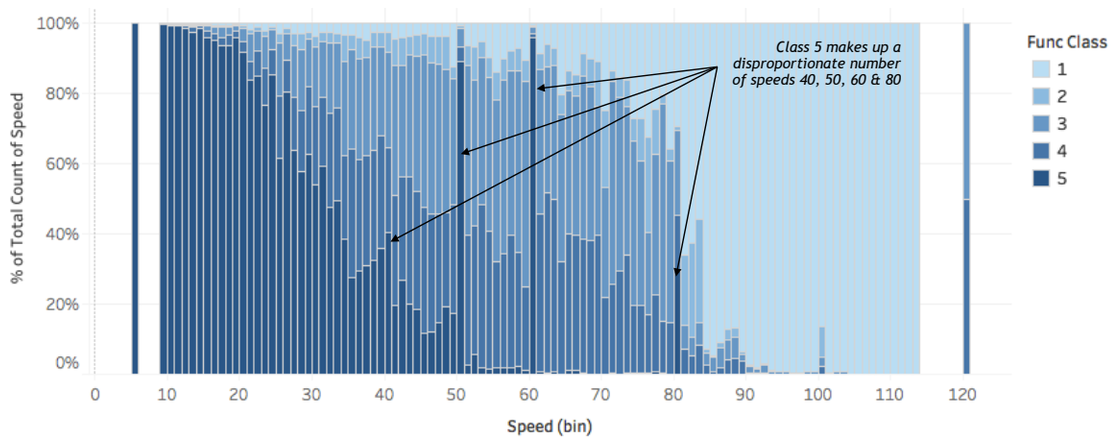
Bin intervals are 1km/h

Peaks at intervals of 10km/h and one less (ie. 9, 19, 39 …)

Dataset Treatment

Proportion of Func Class making up each Speed value

Distribution of Speed (a continuous variable with no binning…)



Bin intervals are 1km/h

Abnormally high proportions of Class 5 at 50, 60 and 80

Use Cases

Grass GIS

Python NetworkX

Spatio-temporal visualisation

Use Cases

GRASS GIS

Build a connected graph object from the Shapefiles

```
v.net --verbose input=dub30_exp_thu points=nodes out=streets_net operation=connect threshold=10
```

Compute Shortest Path between two points

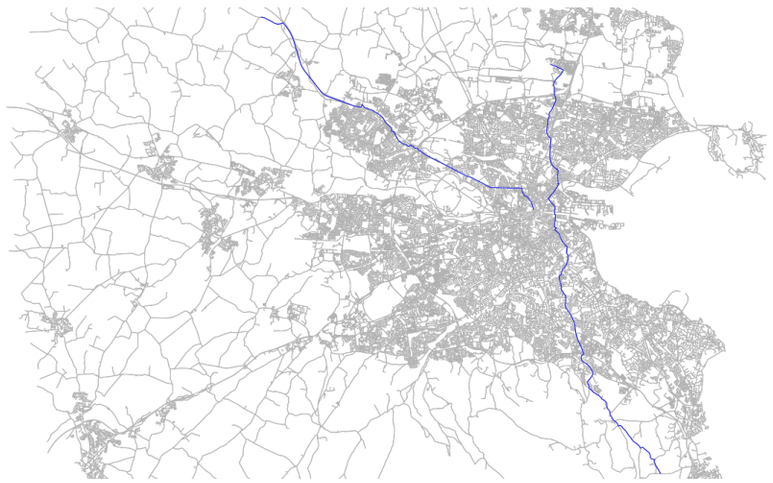
```
v.net.path input=streets_net output=path arc_column=length
```

GRASS can impute nodes from a shape file of edges

Use Cases

GRASS GIS Shortest Path

Compute Shortest Path between two points *by Length*

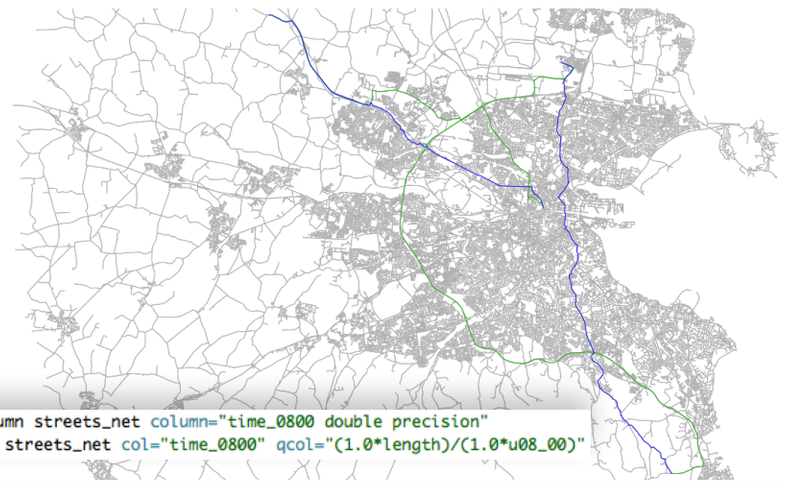


Links are from two points in the outer suburbs to The Spire and Dublin Airport

Use Cases

GRASS GIS Shortest Path

Compute Shortest Path between two points *by Time*



```
v.db.addcolumn streets_net column="time_0800 double precision"
v.db.update streets_net col="time_0800" qcol="(1.0*length)/(1.0*u08_00)"
```

Shortest Path at 8:00

Add a calculated column to the GRASS database

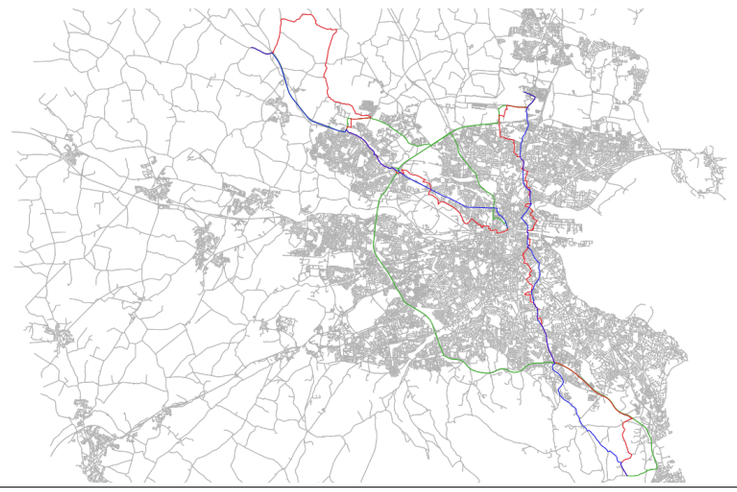
Units of “time_0800” are meaningless as “length” is in metres and “u08_00” is in km/h

Off by a constant of 1000 so shortest path direction is still correct

Use Cases

GRASS GIS Shortest Path

Compute Shortest Path between two points *by Speed*

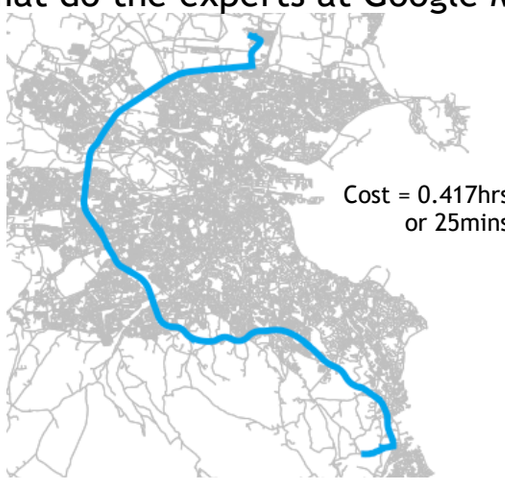


Obviously optimising for the smallest Speed is particularly desired however it provides an interesting proxy for “slowest direct path”

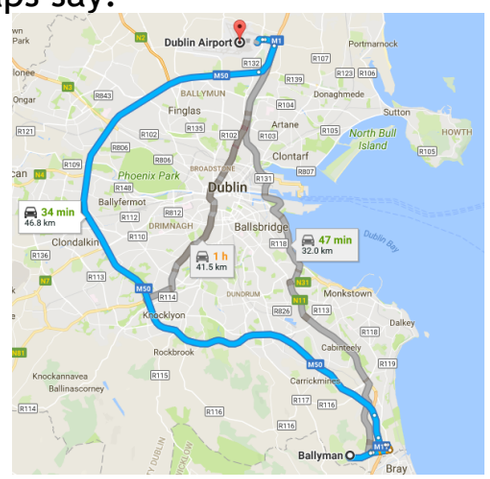
Use Cases

GRASS GIS Shortest Path

What do the experts at Google Maps say?



Cost = 0.417hrs
or 25mins



Shortest Paths at 20:00

Path is extremely close, cost is not as accurate (even after accounting for mis-calculation)

Use Cases

Python's NetworkX

Use a dedicated Graph analysis package…