Obtaining Scottish Public Transport Data

Lisa Hopcroft

16 December 2020

Introduction

- Principal Information Analyst at Public Health Scotland
- Clinical trials statistician
- Effect of public transport accessibility on attendance

The problem

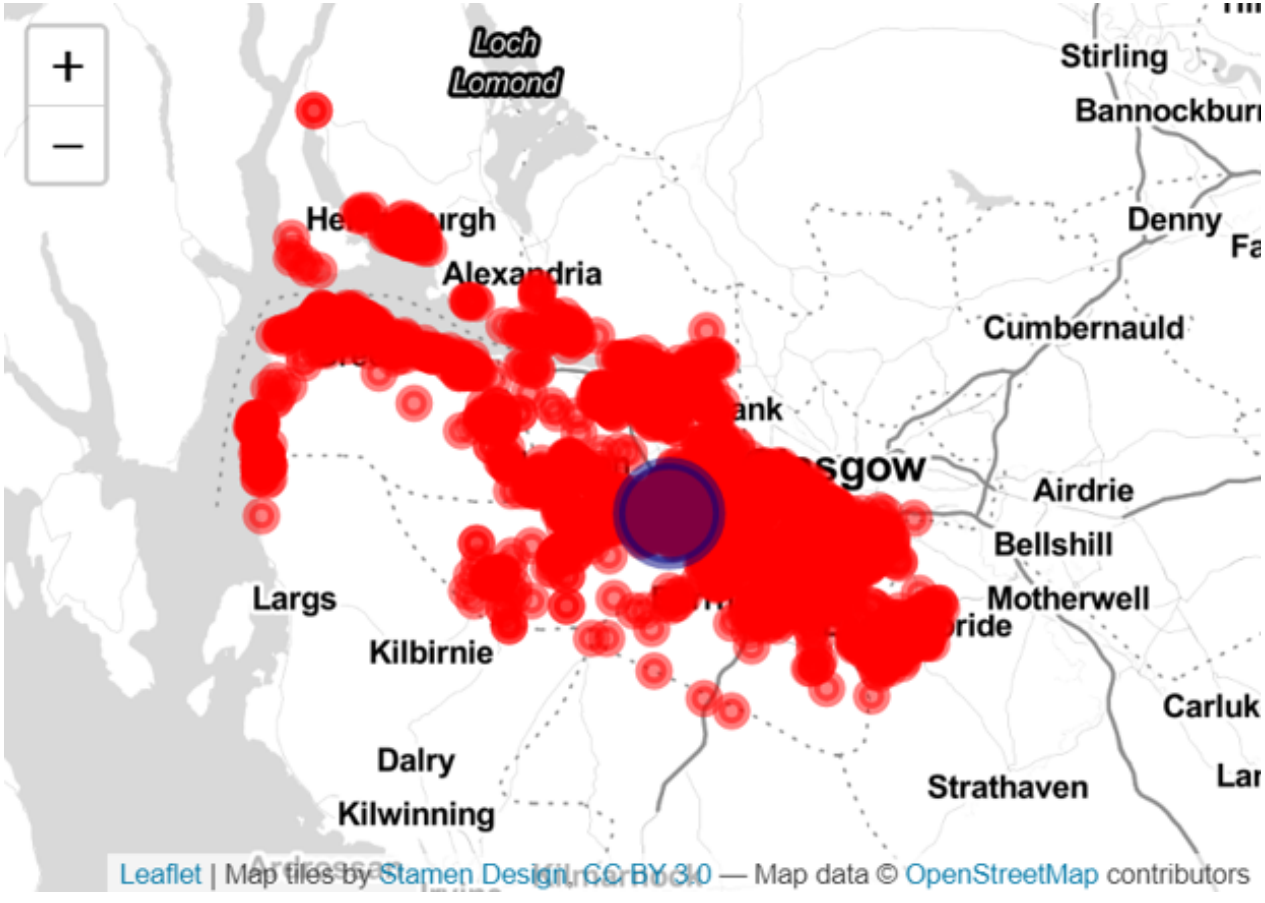


Figure 1: A map of clinical trial participants

Postcode information

```
POSTCODE.information %>% glimpse
```

```
## Rows: 6
## Columns: 9
## $ idnum              <int> 1, 2, 3, 4, 5, NA
## $ outcode            <chr> "PA9", "PA14", "PA19", "PA16", "G76", "PA2"
## $ postcode           <chr> "PA91BJ", "PA146PE", "PA191QP", "PA167PS", "G768D.
## $ longitude          <dbl> -4.555789, -4.584375, -4.814558, -4.785852, -4.26.
## $ latitude           <dbl> 55.81193, 55.92304, 55.95944, 55.96135, 55.78054,.
## $ zone_intermediate <chr> "Renfrewshire Rural South and Howwood", "Renfrews.
## $ zone_lower         <chr> "Renfrewshire Rural South and Howwood - 03", "Ren.
## $ admin_authority    <chr> "Renfrewshire", "Renfrewshire", "Inverclyde", "In.
## $ group              <chr> "Participant", "Participant", "Participant", "Par.
```

Postcode information

```
orig = POSTCODE.information %>%
  filter( group=="Participant" ) %>%
  head( 1 ) %>%
  pull( postcode )

orig
```

```
## [1] "PA91BJ"
```

```
dest = POSTCODE.information %>%
  filter( group=="Hospital" ) %>%
  pull( postcode )


dest
```

```
## [1] "PA29PN"
```

Traveline Scotland's Bulk Journey Planner

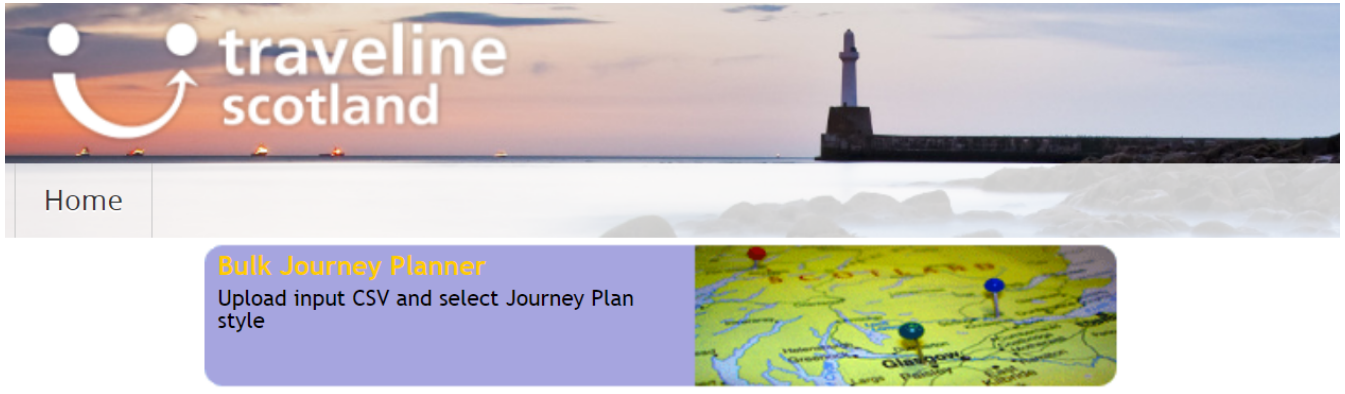


Figure 2: Traveline Scotland's Bulk Journey Planner

API access

- A way to interact with services online
  1. Construct your QUERY
  2. Submit the QUERY to the API
  3. Parse the RESPONSE from the API
- In order to access the API, you may need to be issued a KEY

Bulk Journey Planner API

- Locations need to be described by their ‘Eastings’ and ‘Northings’.

```
orig_info = PostcodesioR::postcode_lookup( orig )
orig_info.e = orig_info$eastings %>% as.character()
orig_info.e
```

```
## [1] "239941"
```

```
orig_info.n = orig_info$northings %>% as.character()
orig_info.n
```

```
## [1] "660585"
```

1. Construct the QUERY

```
## <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
##   <soapenv:Header user_key="a1fe53720f5c5261d2043dbab2775c45"/>
##   <soapenv:Body>
##     <v4:ItineraryRequestStructureElement RequestId="VALUE_REQUEST_ID">
##       <sch:Origins>
##         <sch:Origin>
##           <sch:OriginPlace>
##             <sch:Geocode>
##               <sch:Easting>VALUE_ORIGIN_EASTING</sch:Easting>
##               <sch:Northing>VALUE_ORIGIN_NORTHING</sch:Northing>
##             </sch:Geocode>
##           </sch:OriginPlace>
##         </sch:Origin>
##       </sch:Origins>
##
```

...

1. Construct the QUERY

```
my_query = BASE_QUERY_STRING %>%
    str_replace( "VALUE_REQUEST_ID"           , sprintf( "%s>%s",
                                                         orig,
                                                         dest)) %>%
    str_replace( "VALUE_ORIGIN_EASTING"       , orig_info.e ) %>%
    str_replace( "VALUE_ORIGIN_NORTHING"      , orig_info.n ) %>%
    str_replace( "VALUE_DESTINATION_EASTING"  , dest_info.e ) %>%
    str_replace( "VALUE_DESTINATION_NORTHING" , dest_info.n ) %>%
    str_replace( "VALUE_ARRIVAL_TIME"         , "2020-12-10T13:00:00" )
```

1. Construct the QUERY

```
## <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
##   <soapenv:Header user_key="a1fe53720f5c5261d2043dbab2775c45"/>
##   <soapenv:Body>
##     <v4:ItineraryRequestStructureElement RequestId="PA91BJ>PA29PN">
##       <sch:Origins>
##         <sch:Origin>
##           <sch:OriginPlace>
##             <sch:Geocode>
##               <sch:Easting>239941</sch:Easting>
##               <sch:Northing>660585</sch:Northing>
##             </sch:Geocode>
##           </sch:OriginPlace>
##         </sch:Origin>
##       </sch:Origins>
##
```

2. Submit the QUERY to the API

```
post_response = httr::POST( API_URL,
                            httr::add_headers(user_key = API_KEY),
                            body=my_query )

post_response.xml = httr::content( post_response, as = "parsed" )
```

3. Parse the RESPONSE from the API

```
# xml2::write_xml( x=post_response.xml,
#                  file="journey_output.xml",
#                  as="xml" )

xml2::xml_ns_strip( post_response.xml )
xml_ilist = xml2::xml_find_all( post_response.xml,
                                ".//Itinerary")
n_i = xml_ilist %>% length
n_i
```

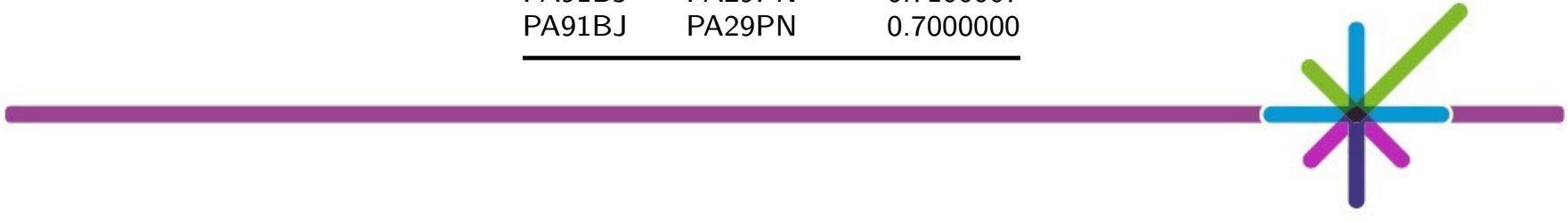
```
## [1] 4
```

3. Parse the RESPONSE from the API

origin	destination	start	end
PA91BJ	PA29PN	2020-12-10T11:07:00.000Z	2020-12-10T11:49:00.000Z
PA91BJ	PA29PN	2020-12-10T11:37:00.000Z	2020-12-10T12:19:00.000Z
PA91BJ	PA29PN	2020-12-10T12:06:00.000Z	2020-12-10T12:49:00.000Z
PA91BJ	PA29PN	2020-12-10T12:37:00.000Z	2020-12-10T13:19:00.000Z

```
journeys = journeys %>% mutate(
  duration = as.numeric(difftime(ymd_hms(end),
                                 ymd_hms(start)),
                        units="hours") ) %>%
  select( -start, -end )
```

origin	destination	duration
PA91BJ	PA29PN	0.7000000
PA91BJ	PA29PN	0.7000000
PA91BJ	PA29PN	0.7166667
PA91BJ	PA29PN	0.7000000



Comparing to other data sources?

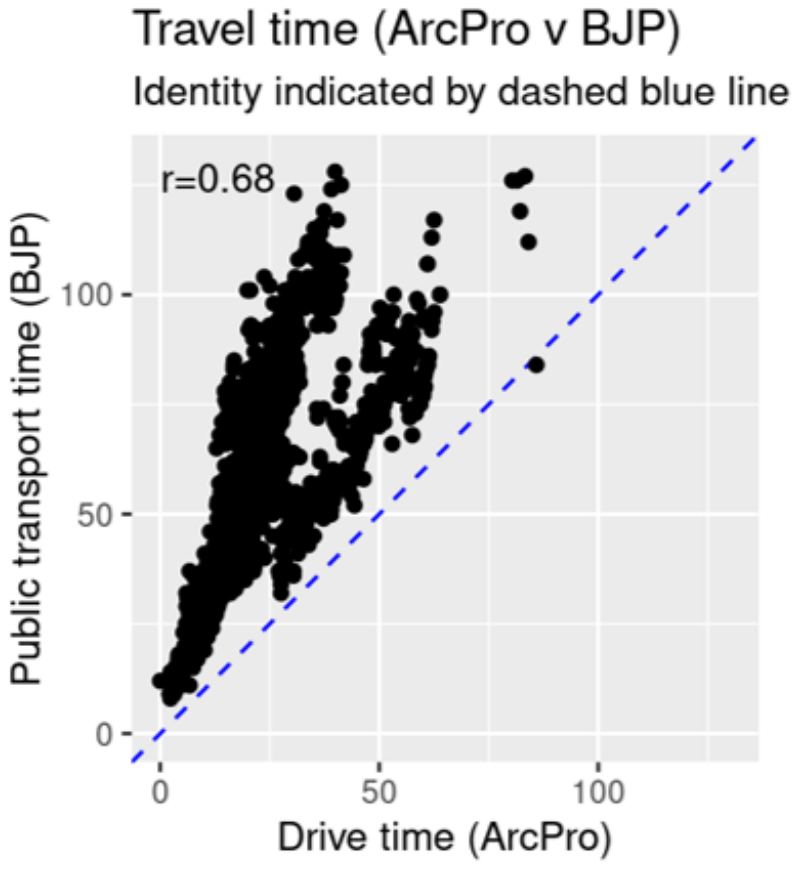


Figure 3: Travel time by road (ArcPro) and by public transport (BJP)

Bulk Journey Planner Options

Users can specify:

- modes of transport (bus/rail/walk/metro/ferry)
- how many itineraries
- depart after (rather than arrive by)

Traveline Scotland provide query templates for different queries:

- Fare calculation (where possible)
- Service timetables
- Information about service providers

Bulk Journey Planner advice

- Don't expect helpful error messages
  - If your arrive by date/time is the past, it will silently fail
  - If you can't arrive by your date/time within a day, it will silently fail
- Don't send big batches
  - If >3 jobs fail, the whole batch fails
- Add a 'Sys.sleep()' between queries

Thank you

✉ Lisa.Hopcroft@phs.scot

@LisaHopcroft

LisaHopcroft // TALK-RLadies-Edinburgh-16-Dec-2020