PSY6422 Final Project

240189002

Research Question

How Does Housing Supply Compare Across Regions of England When Adjusting for Population?

In light of the ongoing housing crisis and shortage in the UK, I became interested in how different regions of England are performing with their supply of new homes. Using data from gov.uk, I analyzed year-on-year changes in "net additional dwellings". This metric captures the absolute change in housing stock, factoring in gains and losses from new builds, conversions, changes of use (e.g., residential to office), and demolitions.

I also found mid-year population estimates for regions from the Office for National Statistics (ONS) to create a "per 1000 people figure". This would allow easier comparisons across regions rather than having the absolute figures of housing, given that highly populated regions would have more housing completed.

As the final visualisation, I decided on a line plot with "Year" on the X axis, and "New Homes completed per 1000 people" on the Y axis, with 9 lines to represent each of the 9 English regions to display the trends over time. A key aspect of the final visualisation is the interactive element: I wanted users to compare, isolate, and zoom into areas and years of interest.

Please see the References section at the end of the document/webpage for links to data sources.

Data Preparation

Setting up & loading packages

The code below is the setup and the required libraries. Renv was used for package management, with package versions used listed in /renv.lock in the repository.

```
# Package version management:
# renv package to ensure the packages used are preserved for cloning
if(!require('renv')) install.packages('renv')
library(renv)
# Restores the environment from renv.lock file
renv::restore()
# here determines the project root directory automatically
# constructs file paths relative to that root, not current working directory
library(here)
# readODS package needed to read the Net Additional Dwellings .ods dataset
library(readODS)
# readxl package needed to read the Regional Population .xlsx dataset
library(readxl)
# Library needed for data wrangling: mainly using tidyr and dplyr from tidyverse
library(tidyverse)
```

```r
# ggplot2 allows for plotting with ggplot
library(ggplot2)
# plotly allows interactivity from ggplot plots
# Chosen specifically for use of plot zooming and custom hover text
library(plotly)
# scales allows better axes formatting: only using for the population plot
library(scales)
# allows for better data handling
library(tools)
# webshot is needed for saving my final interactive plot as a static pdf
library(webshot2)
```

Importing data

I will be merging these data sets for a figure of the number of new housing units relative to that region's population for a given year, by 1000 people. This means columns that need to be merged and therefore need to match are "Region" and "Year".

```r
# Ensure "data" folder is made and file names match below if downloading sources
file_path_homes <- here("data", "raw_netadditionaldwellings.ods")
file_path_pop <- here("data", "raw_population.xlsx")
# Read data from the files:
data_homes <- read_ods(file_path_homes, sheet = 5)
# Unrounded sheet used for more accurate figures
data_pop <- read_xlsx(file_path_pop, sheet = 11)
# Mid-year population estimates 2011-2023

# See new imported data sets are correct and review what edits need to be made
# Best to View() in R Studio initially; using head() in R Markdown for publishing
```

Wrangle (1/3): Net Additional Dwellings

I started with cleaning up the Net Additional Dwellings data. This will be referred to as "New Homes" for clarity. Note that this data is on financial year additions to housing stock, so data from 2011-2012 will be representative of 2011 data and so on, for the purposes of the project. Population estimates data only held information from 2011 to 2023, so any New Homes data pre-2011 was not used in the project.

```r
# Quick view of our imported data
head(data_homes)
```

```
## # A tibble: 6 x 12
##   Table 118  Annual net ~1 ...2  ...3  ...4  ...5  ...6  ...7  ...8  ...9  ...10
##   <chr>                  <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 This worksheet contains~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 2 Back to contents         <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 3 Components of net housi~ Year  Nort~ Nort~ York~ East~ West~ East~ Lond~ Sout~
## 4 Net additions [note 1]   2000~ 2890  10720 10800 14830 13790 17780 15710 25680
## 5 Net additions [note 1]   2001~ 4489  13543 12752 16188 13052 19154 19688 26219
## 6 Net additions [note 1]   2002~ 5343  18120 13452 16938 13742 21554 21648 27269
## # i abbreviated name:
## #   1: 'Table 118  Annual net additional dwellings and components, England and the regions, 2000-01 t
## # i 2 more variables: ...11 <chr>, ...12 <chr>
```

```r
# See the structure of the data: what data types need to be converted
str(data_homes)
```

```
## tibble [296 x 12] (S3: tbl_df/tbl/data.frame)
##  $ Table 118  Annual net additional dwellings and components, England and the regions, 2000-01 to 20
##  $ ...2
##  $ ...3
##  $ ...4
##  $ ...5
##  $ ...6
##  $ ...7
##  $ ...8
##  $ ...9
##  $ ...10
##  $ ...11
##  $ ...12
```

```r
# Remove 1st 2 rows as they are irrelevant
data_homes <- data_homes[-c(1:2), ]
# Make first row the headings for columns, then delete the first row
colnames(data_homes) <- data_homes[1, ]
data_homes <- data_homes[-1, ]
# Delete "England" column as we're just seeing regions
data_homes <- data_homes %>%
  select(-"England")
# Filtering for only Net data
data_homes <- data_homes %>% filter(
  str_starts(`Components of net housing supply`, "Net additions")
  )
# Can now delete this "Components of net housing supply" column as have used filters
data_homes <- data_homes %>%
  select(-`Components of net housing supply`)
# Rename Year as first 4 digits: cannot convert to numeric otherwise
data_homes$Year <- str_sub(data_homes$Year, 1, 4)
# Filter to remove pre-2011 data to match with the population data we have
data_homes <- data_homes %>%
  filter(Year >= 2011)
# Make long
data_homes <- pivot_longer(data_homes,
             cols = -Year, #all columns except Year as it's already long
             names_to = "Region",
             values_to = "New_Homes")
# Check whether we have missing data in the columns we need;
# Function created to give us a statement rather than a list of output
check_missing_vals <- function(data) {
  if (any(is.na(data))) {
    message("There are missing values (NA) in the data")
  } else {
    message("No missing values in the data")
  }
}
check_missing_vals(data_homes)
```

```
## No missing values in the data
```

```
# Looks fine: can convert to numeric to allow plotting
data_homes <- data_homes %>%
  mutate(across(c(Year, New_Homes), as.numeric))
```

```
# How does it look?
head(data_homes)
```

```
## # A tibble: 6 x 3
##    Year Region                    New_Homes
##   <dbl> <chr>                         <dbl>
## 1  2011 North East                     3939
## 2  2011 North West                    10612
## 3  2011 Yorkshire and The Humber      12066
## 4  2011 East Midlands                 12426
## 5  2011 West Midlands                 10206
## 6  2011 East of England               18460
```

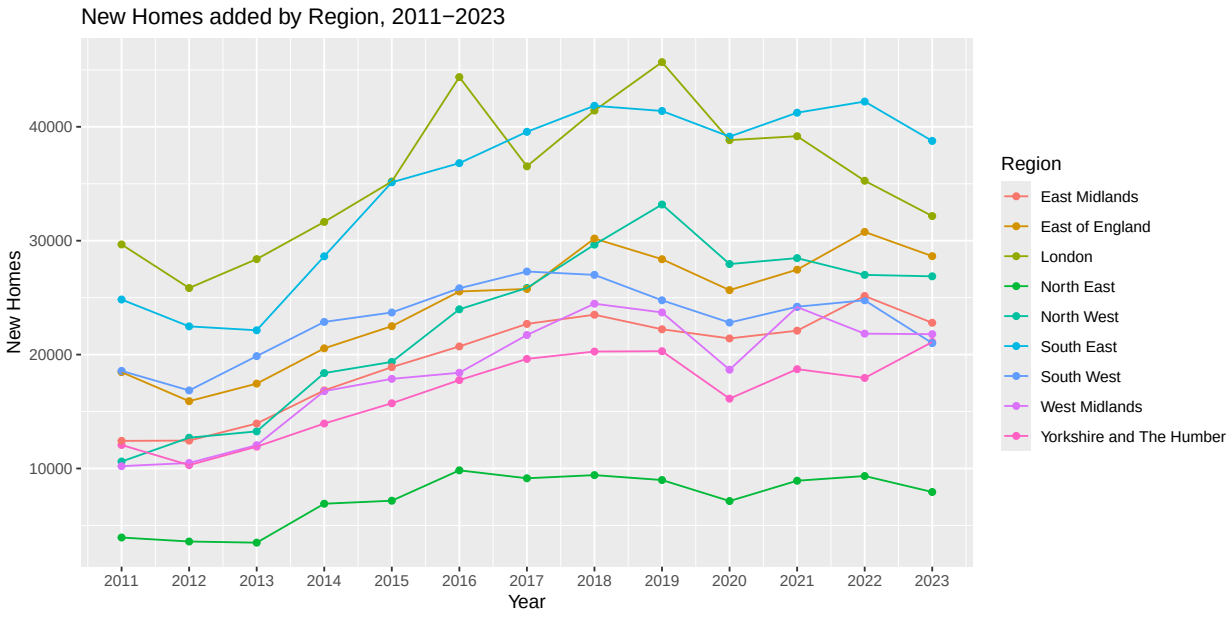
```
str(data_homes)
```

```
## tibble [117 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Year     : num [1:117] 2011 2011 2011 2011 2011 ...
##  $ Region   : chr [1:117] "North East" "North West" "Yorkshire and The Humber" "East Midlands" ...
##  $ New_Homes: num [1:117] 3939 10612 12066 12426 10206 ...
```

```
# Looks good! Tidy, relevant, and needed columns are numeric
# 117 x 3 also makes sense: 9 regions*13 years of data
# Let's plot
```

Plot (1/3): Net Additional Dwellings

```
# Quick visual of our data: does this look correct?
plot_homes <- ggplot(data = data_homes, mapping = aes(
  x = Year, y = New_Homes,
  group = Region, colour = Region)) +
  geom_point() +
  geom_line() +
  labs(title = "New Homes added by Region, 2011-2023",
         x = "Year",
         y = "New Homes") +
    scale_x_continuous(breaks = seq(2011, 2023)) #Specify X scale with all years
# View the plot
print(plot_homes)
```




```
# Noticeable dip in 2019-2020 is interesting to see
```

Wrangle (2/3): Population

Here I'm wrangling the population data. Note that these are mid-year estimates. For the purposes of the project, the mid-figure represents the population for that year ("Mid-2011" will represent "2011", etc).

```
# View imported messy data
head(data_pop)
```

```
## # A tibble: 6 x 16
##   MYE4: Population estim~1 ...2  ...3  ...4  ...5  ...6  ...7  ...8  ...9  ...10
##   <chr>                    <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 This worksheet contains~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 2 To turn off freeze pane~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 3 Please choose from the ~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 4 This met my needs, plea~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 5 I need something slight~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 6 This is not what I need~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## # i abbreviated name:
## #   1: 'MYE4: Population estimates: Summary for England and Wales, mid-2011 to mid-2023'
## # i 6 more variables: ...11 <chr>, ...12 <chr>, ...13 <chr>, ...14 <chr>,
## #   ...15 <chr>, ...16 <chr>
```

```
str(data_pop)
```

```
## tibble [364 x 16] (S3: tbl_df/tbl/data.frame)
##  $ MYE4: Population estimates: Summary for England and Wales, mid-2011 to mid-2023: chr [1:364] "This
##  $ ...2                                                                           : chr [1:364] NA NA
##  $ ...3                                                                           : chr [1:364] NA NA
##  $ ...4                                                                           : chr [1:364] NA NA
```

```
##  $ ...5                                                                  : chr [1:364] NA NA
##  $ ...6                                                                  : chr [1:364] NA NA
##  $ ...7                                                                  : chr [1:364] NA NA
##  $ ...8                                                                  : chr [1:364] NA NA
##  $ ...9                                                                  : chr [1:364] NA NA
##  $ ...10                                                                 : chr [1:364] NA NA
##  $ ...11                                                                 : chr [1:364] NA NA
##  $ ...12                                                                 : chr [1:364] NA NA
##  $ ...13                                                                 : chr [1:364] NA NA
##  $ ...14                                                                 : chr [1:364] NA NA
##  $ ...15                                                                 : chr [1:364] NA NA
##  $ ...16                                                                 : chr [1:364] NA NA
```

```r
# Remove first 6 rows, no data
data_pop <- data_pop %>% slice(-1:-6)
# Remove first column, no data
data_pop <- data_pop[ ,-1]
# Rename headers as the first row, then remove first row
colnames(data_pop) <- data_pop[1, ]
data_pop <- data_pop[-1, ]
# Rename just 1st col to Region to match New Homes data
colnames(data_pop)[colnames(data_pop) == "Name"] <- "Region"
# Filter to view only regional data
data_pop <- data_pop %>%
  filter(grepl("Region", Geography))
# Remove Geography column as now not needed
data_pop <- data_pop[ ,-2]
# Convert to long
data_pop <- data_pop %>%
  pivot_longer(
    cols = starts_with("Mid"),  # Select the year columns (2011, 2012, 2013)
    names_to = "Year",          # The new column for year
    values_to = "Population"    # The new column for population values
  )
# Remove 'Mid-' from the Year column
data_pop$Year <- gsub("Mid-", "", data_pop$Year)
# Change from UPPERCASE to LikeThis for consistency
data_pop$Region <- toTitleCase(tolower(data_pop$Region))
# Missing data check
check_missing_vals(data_pop)
```

```
## No missing values in the data
```

```r
# None
# Make relevant columns numeric for plot
data_pop <- data_pop %>%
  mutate(across(c(Year, Population), as.numeric))
```

```r
# View cleaned data
head(data_pop)
```

```
## # A tibble: 6 x 3
```

```
##   Region      Year Population
##   <chr>      <dbl>      <dbl>
## 1 North East  2023    2711380
## 2 North East  2022    2682069
## 3 North East  2021    2647493
## 4 North East  2020    2637426
## 5 North East  2019    2636676
## 6 North East  2018    2629393
```

```
str(data_pop)
```

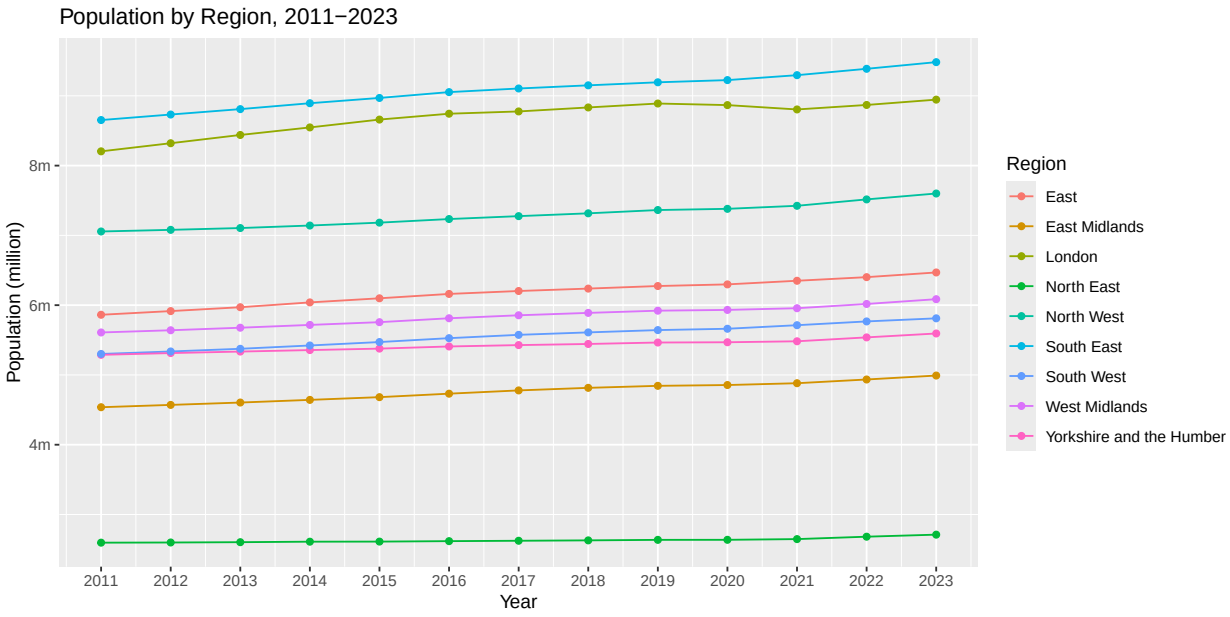
```
## tibble [117 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Region    : chr [1:117] "North East" "North East" "North East" "North East" ...
##  $ Year      : num [1:117] 2023 2022 2021 2020 2019 ...
##  $ Population: num [1:117] 2711380 2682069 2647493 2637426 2636676 ...
```

```
# Nice! Let's plot this
```

Plot (2/3): Population

```
plot_pop <- ggplot(data_pop, aes(
  x = Year, y = Population,
  group = Region, colour = Region)) +
  geom_line() +
  geom_point() +
  labs(title = "Population by Region, 2011-2023",
       x = "Year",
       y = "Population (million)") +
  scale_x_continuous(breaks = seq(2011, 2023)) +
  #Default logged values showed originally: scaled by million with below code
  scale_y_continuous(labels = label_number(scale = 1e-6, suffix = "m"))

# View plot
print(plot_pop)
```



Wrangle (3/3): Merging datasets

Combining the now cleaned data to form a new data frame. "Region" and "Year" will remain as columns, and "New Homes" and "Population" are to be added as new columns. Names of Regions and Years must be the same for all data to merge correctly, let's make sure:

```
unique(data_homes$Year)
```

```
##  [1] 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023
```

```
unique(data_pop$Year)
```

```
##  [1] 2023 2022 2021 2020 2019 2018 2017 2016 2015 2014 2013 2012 2011
```

```
# Great, years matching based off previous filtering
```

```
unique(data_homes$Region)
```

```
## [1] "North East"               "North West"
## [3] "Yorkshire and The Humber" "East Midlands"
## [5] "West Midlands"            "East of England"
## [7] "London"                   "South East"
## [9] "South West"
```

```
unique(data_pop$Region)
```

```
## [1] "North East"               "North West"
## [3] "Yorkshire and the Humber" "East Midlands"
## [5] "West Midlands"            "East"
## [7] "London"                   "South East"
## [9] "South West"
```

```r
# Looks like there are some differences in Region names with East and Yorkshire
# I prefer "East of England" and "Yorkshire and The Humber" from the New Homes data
```

```r
# Changing data_pop names:
# Rename "East" to just "East of England"
# Rename "Yorkshire and the Humber" to "... The Humber"
data_pop <- data_pop %>%
  mutate(
    `Region` = recode(
      `Region`, "East" = "East of England"),
    `Region` = recode(
      `Region`, "Yorkshire and the Humber" = "Yorkshire and The Humber")
  )
# Can now create a new df where regional new homes are adjusted by population
```

```r
# Merging with Region and Year labels remaining
data_final <- merge(data_homes, data_pop,
                    by.x = c("Region", "Year"),
                    by.y = c("Region", "Year"))
#Check all 9 regions are here
unique(data_final$Region)
```

```
## [1] "East Midlands"            "East of England"
## [3] "London"                   "North East"
## [5] "North West"               "South East"
## [7] "South West"               "West Midlands"
## [9] "Yorkshire and The Humber"
```

```r
# Yep
# Check all years we're interested in are accounted for
unique(data_final$Year)
```

```
##  [1] 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023
```

```r
# Yep
# Any data lost in the merge, resulting in NA values?
check_missing_vals(data_final)
```

```
## No missing values in the data
```

```r
# No missing!
# Make new columns numeric for plotting
data_final <- data_final %>%
  mutate(across(c(Year, New_Homes, Population), as.numeric))
# Calculate the Homes/Population*1000 ratio, create new "Homes_By_Pop" column for figures
data_final$Homes_By_Pop <- ((
  data_final$`New_Homes` / data_final$`Population`)*1000)
```

```r
# Let's see the final dataset used for plotting and the new column
head(data_final)
```

```
##          Region Year New_Homes Population Homes_By_Pop
## 1 East Midlands 2011     12426    4537448     2.738544
## 2 East Midlands 2012     12449    4570702     2.723652
## 3 East Midlands 2013     13949    4604568     3.029383
## 4 East Midlands 2014     16857    4642629     3.630917
## 5 East Midlands 2015     18896    4681640     4.036192
## 6 East Midlands 2016     20717    4731615     4.378420
```

```
tail(data_final)
```

```
##                       Region Year New_Homes Population Homes_By_Pop
## 112 Yorkshire and The Humber 2018     20269    5443883     3.723262
## 113 Yorkshire and The Humber 2019     20297    5464651     3.714235
## 114 Yorkshire and The Humber 2020     16132    5468487     2.949993
## 115 Yorkshire and The Humber 2021     18721    5482455     3.414711
## 116 Yorkshire and The Humber 2022     17952    5538213     3.241479
## 117 Yorkshire and The Humber 2023     21102    5594125     3.772172
```

```
str(data_final)
```

```
## 'data.frame':    117 obs. of  5 variables:
##  $ Region      : chr  "East Midlands" "East Midlands" "East Midlands" "East Midlands" ...
##  $ Year        : num  2011 2012 2013 2014 2015 ...
##  $ New_Homes   : num  12426 12449 13949 16857 18896 ...
##  $ Population  : num  4537448 4570702 4604568 4642629 4681640 ...
##  $ Homes_By_Pop: num  2.74 2.72 3.03 3.63 4.04 ...
```

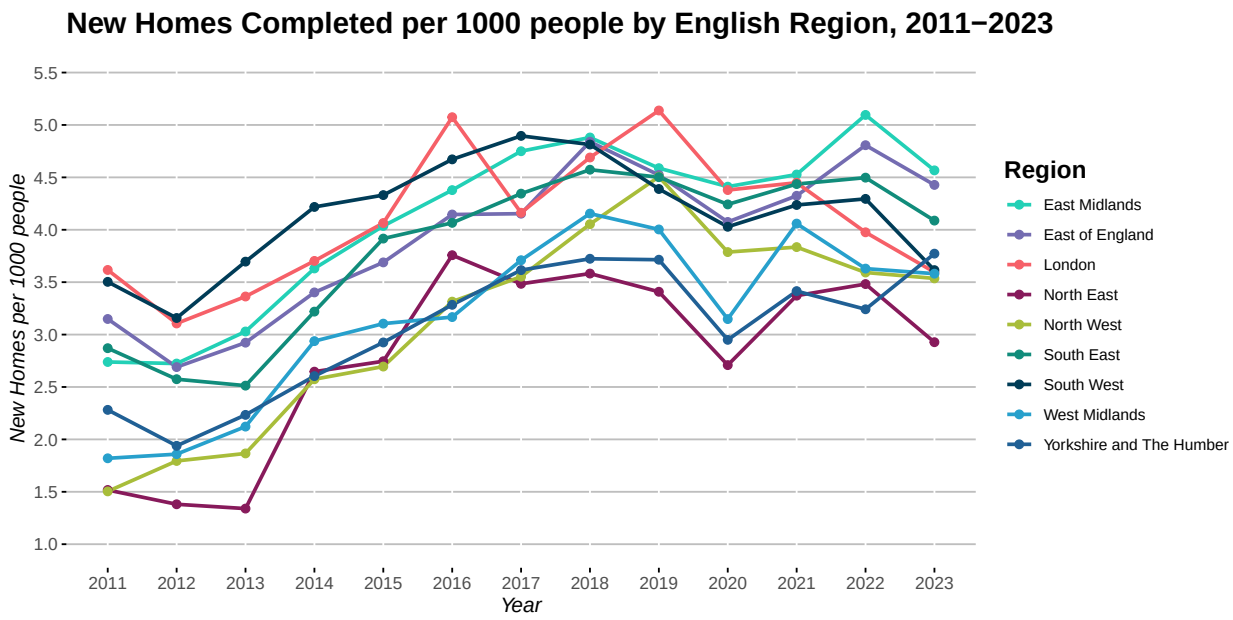
Plot (3/3): Merged data

Plotting it all: ggplot and plotly will be used for creating the visualisation. I took inspiration from ONS colour palette guidance: it was hard to find 9 distinct colours as pre-made packages that were also visually pleasing, so ONS-recommended hex codes were manually added in.

```
# Palette made using ONS guidelines
# Reverse order used: allows better differentiation of overlapping lines
palette <- c(
"East Midlands" = "#22d0b6", #"Mint green"
"East of England" = "#746cb1", #Lavender purple"
"London" = "#f66068", #"Coral pink"
"North East" = "#871a5b", #"Beetroot purple"
"North West" = "#a8bd3a", #"Spring green"
"South East" = "#118c7b", #"Emerald green"
"South West" = "#003c57", #"Night blue"
"West Midlands" = "#27a0cc", #"Sky blue"
"Yorkshire and The Humber" = "#206095") #"Ocean blue"
```

```
# Plot it all: initially plotting as a static ggplot
plot_final_static <- ggplot(data_final, aes(
  x = Year, y = Homes_By_Pop, group = `Region`, colour = `Region`,
  text = paste( #Custom hover text for interactivity and html formatting added below
    "<b><i>", Region, "</i></b>", "-", "<b><i>", Year, "</i></b>",
```

```
    "<br>Population:", format(Population, big.mark = ","),#bigmark for chunking large numbers and separ
    "<br>Completed Builds:", format(New_Homes, big.mark = ","),
    "<br><b>Homes completed per 1000 people:</b>", round(Homes_By_Pop, 2)#2dp
  ))) +
  geom_line(size = 1) +
  geom_point(size = 2) +
  labs(title = "New Homes Completed per 1000 people by English Region, 2011-2023",
       x = "Year",
       y = "New Homes per 1000 people") +
  scale_x_continuous(
    breaks = seq(2011, 2023)) +
  scale_y_continuous(
    breaks = seq(1, 5.5, by = 0.5), #Every .5 is shown on the Y axis
    #Starts at 1 and ends at 5.5: before the smallest value and beyond largest
    limits = c(1, 5.5)) +
  scale_color_manual(values = palette) +
 theme(
   panel.grid.major.y = element_line(color = "gray", size = 0.5),
    title = element_text(face = "bold", size = 14),
    axis.title.x = element_text(face = "italic", size = 12),
    axis.title.y = element_text(face = "italic", size = 12),
    axis.text = element_text(size = 10),
    #White panel and plot background coded below
    panel.background = element_rect(fill = "white"),
    plot.background = element_rect(fill = "white"),
    axis.ticks = element_line(color = "black")#Black ticks
        )
# Viewing the static, non interactive plot...
plot_final_static
```




```
# Making static plot interactive for the assessed plot
#Tooltip settings ensure my custom text is added
plot_final <- ggplotly(plot_final_static, tooltip = "text",
```

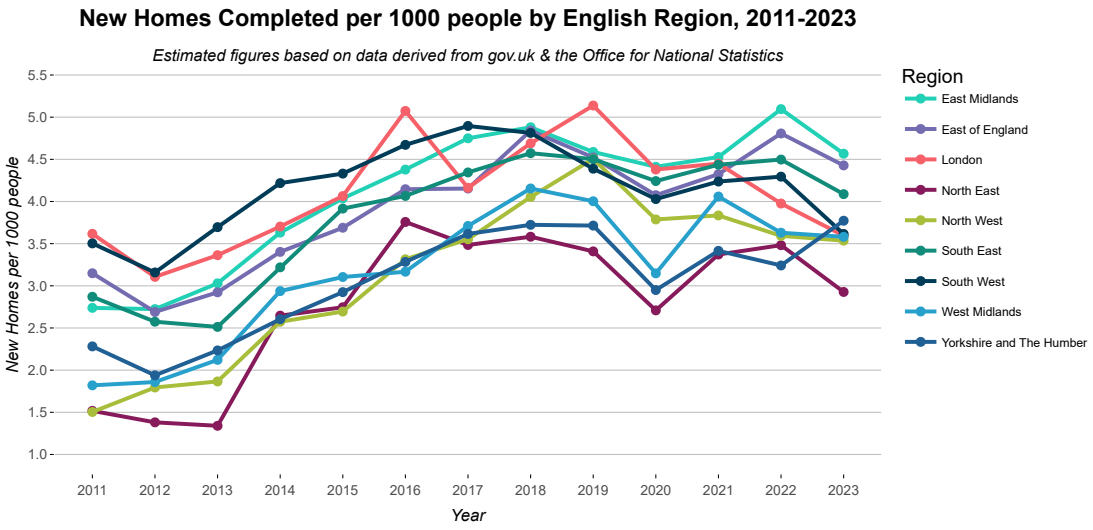
```
                        width = 1050, height = 500)

#Add title and subtitle with HTML formatting to plotly layout
plot_final <- plot_final %>% layout(
  #Suggested font by gov.uk fonts, also in ONS font family
  font = list(family = "Helvetica"),
  title = list(
    text = "<b>New Homes Completed per 1000 people by English Region, 2011-2023</b><br><sub><i>Estimate
    x = 0.5,  #Centre-align title and subtitle
    xanchor = 'center',
    yanchor = 'top',
    y = 0.95), #to avoid title being too close to the top chunk
  #Adding crosshairs for x and y axes: easier to pinpoint data of interest
  #Will present when interacting
  xaxis = list(
    showspikes = TRUE,
    spikesnap = "cursor", #based on cursor location: follows your movement
    spikemode = "across",
    spikedash = "dash",
    spikethickness = 1.5,
    spikecolor = "gray"
  ),
  yaxis = list(
    showspikes = TRUE,
    spikesnap = "cursor",
    spikemode = "across",
    spikedash = "dash",
    spikethickness = 1.5,
    spikecolor = "gray"
  )
)
```

Final Assessed Plot

This is the final interactive plot.

```
plot_final
```


New Homes Completed per 1000 people by English Region, 2011-2023
Estimated figures based on data derived from gov.uk & the Office for National Statistics
New Homes per 1000 people
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
Year
Region
East Midlands
East of England
London
North East
North West
South East
South West
West Midlands
Yorkshire and The Humber

Conclusions

I was quite surprised to see how few homes were being built relative to regional populations; I'd expected more of a noticeable increase as years and populations increased. There is the obvious dip due to the pandemic in 2019-2020 due to halting construction activity, and it looks like we're still trying to gain that momentum back in providing new homes.

I found Yorkshire's 2023 increase relative to 2022 interesting, when all other regions drop. This I'm supposing is due to the drive in new housing alongside a moderate increase to population.

Limitations & Future ideas

Although my project offers an interesting initial insight into new housing supply differences across English regions, it of course overlooks the contextual and political factors fundamental to housing supply. Economic conditions (e.g., unemployment, income levels), government policies (e.g., housing subsidies, planning permissions/ regulations) and details regarding types of housing or population density, are not considered. Future projects may wish to investigate adjusting to these factors for a more nuanced insight.

Another limitation is that the data sets don't match by point in the year, with Net Additional Dwellings (new homes) data representing the financial year (i.e. 2011-2012 being April 2011 to March 2012) and population data being on mid-year estimates. This means the data won't represent the years noted as accurately as I'd wanted, however end-of-calendar year data wasn't available for either data set.

My initial idea for this project was to use data on affordable units of housing available regionally and find proportions of net housing that are affordable, to see where in England there are the most affordable options (both ownership and rent) for housing. This fell short when I realised the affordable housing data is only available as a gross figure (i.e. not taking into account any losses of affordable housing), therefore any data presented would be misleading. This would be an interesting future idea given the net data is published. There is some "Official Statistics in Development" data only on affordable housing for rent, if interested (see references).

Reflection

I'm quite proud of my plot! It is a relatively simple plot but I learned a lot, and felt a sense of accomplishment when my ideas worked out. I really enjoyed adding in the small details with html too, then seeing it all come together. I think the packages and approach chosen make sense for the question I wanted to look into.

As a personal project and to build on my interests in interactive plots, I'd like to develop a Shiny dashboard relating to housing, specifically into affordability across England. It's all good research into the question: will I ever own my own house?

References

Net Additional Dwellings data: gov.uk | "Table 118: annual net additional dwellings and components, England and the regions" (ODS,54KB) https://www.gov.uk/government/statistical-data-sets/live-tables-on-net-supply-of-housing#live-tables

gov.uk collection on net dwelling supply & confirmation of this being financial year data: https://www.gov.uk/government/collections/net-supply-of-housing

Population estimates data: ONS | "Mid-2023: 2023 local authority boundaries edition of this dataset edition of this dataset" (xlsx, 813.1KB) https://www.ons.gov.uk/peoplepopulationandcommunity/populationandmigration/populationestimates/datasets/estimatesofthepopulationforenglandandwales

Colour palette hex codes & guidance from ONS: https://service-manual.ons.gov.uk/data-visualisation/colours/using-colours-in-charts#standard-category-colour-palette

Unused data - Links for anyone interested in affordable data sets mentioned: https://www.gov.uk/government/statistical-data-sets/live-tables-on-affordable-housing-supply / https://assets.publishing.service.gov.uk/media/65c0b5dec43191000d1a451f/Net_Affordable_Housing_for_Rent.ods/preview