# Pedalling Forward: The Evolution of Dedicated Cycling Infrastructure in Canadian Cities from 2010 to 2022

R Code for Figures and Tables

Richard Wen richard.wen@utoronto.ca

March 09, 2024

## Contents

## Libraries

Install R libraries if needed.

```
install.packages("rmarkdown")
install.packages("bookdown")
install.packages("knitr")
install.packages("tidyverse")
install.packages("glue")
install.packages("readxl")
install.packages("ggtext")
install.packages("scales")
install.packages("patchwork")
```

Load R libraries.

```
library(tidyverse)
library(ggtext)
library(glue)
library(patchwork)
library(readxl)
```

## Settings

```
settings <- list()

# Infrastructure types in order
settings$type_recode_infra <- c(
    PBL = "Cycle Track",
    BUF = "Buffered Lane",
    PL = "Painted Lane",
    LSB = "Local Street\nBikeway"
)

# Infrastructure types to remove
settings$type_filter_infra <- c("N", "None", "SR")

# Road types in order
settings$type_recode_road <- c(
    Arterial = "Arterial",
    Collector = "Collector",
    Local = "Local"
)

# Column references
settings$year_col_road <- "install_year"
settings$type_col_road <- "road_type"
settings$type_col_infra <- "infra_type"

# Set years of interest
settings$year_min <- 2009
settings$year_max <- 2022

# Plot settings
```

```r
settings$line_year <- 2019
```

# Functions

## Calculate Yearly Road Length

The following function calculates yearly road lengths by infrastructure type using cumulative sums and filling in missing years and types.

For a given infrastructure type, the total road length for a given year is expressed below:

$$length_{year,type} = f(year, type) = \sum_{i=year_{min}}^{year} l_{i,type}$$

Where:

- $year$ is the given year
- $type$ is the infrastructure type
- $year_{min}$ is the earliest year available in the data
- $l_{i,type}$ is the road length $l$ for previous years $i$ and infrastructure $j$
- $l_{i,type}$ is set to 0 if there is no data

```r
#' Calculate Yearly Road Lengths By Infrastructure Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param year_col The name (char) or index (int) of the column containing the years.
#' @param type_col The name (char) or index (int) of the column containing the infrastructure type
#' @param len_col The name (char) or index (int) of the column containing the road lengths.
#' @param out_col The name (char) of the column containing the calculated yearly road lengths by type.
#'
#' @return A data.frame with three columns containing the year, type, and calculated yearly road lengths
#' @export
#'
calc_yearly_len <- function(
        df,
        year_col = "install_year",
        type_col = "install_type",
        len_col = "segment_len",
        out_col = "len",
        year_min = settings$year_min,
        year_max = settings$year_max
    ) {

    # Convert data types
    df[[year_col]] <- as.integer(df[[year_col]])
    df[[type_col]] <- as.character(df[[type_col]])
    df[[len_col]] <- as.numeric(df[[len_col]])

    # Remove rows with empty type
    out <- df %>% filter(
        !is.na(.data[[type_col]])
    )

    # Filter to min and max years
```

```
    if (year_min > 0) {
        df <- df %>% filter(
            .data[[year_col]] >= year_min
        )
    } else {
        year_min <- min(out[[year_col]], na.rm = TRUE)
    }
    if (year_max > 0) {
        df <- df %>% filter(
            .data[[year_col]] <= year_max
        )
    } else {
        year_max <- max(out[[year_col]], na.rm = TRUE)
    }

    # Add dummy len for each type and year combo
    # Covers cases where type and year combo does not exist
    # E.g. No new PL installs in 2021, hence a record PL in 2021 does not exist
    type_uniq <- unique(out[[type_col]])
    type_n <- length(type_uniq)
    year_uniq <- year_min:year_max
    year_n <- length(year_uniq)
    out <- out %>% add_row(
        !!year_col := rep(year_uniq, each = type_n),
        !!type_col := rep(type_uniq, year_n),
        !!len_col := rep(0, type_n * year_n)
    )

    # Calc cumsum for each non-empty type ordered by year
    out <- out %>%
        arrange(.data[[year_col]]) %>%
        group_by(.data[[type_col]]) %>%
        mutate(
            !!out_col := cumsum(.data[[len_col]])
        )

    # Get the last cumsum for each year and type
    out <- out %>%
        group_by(.data[[year_col]], .data[[type_col]]) %>%
        arrange(desc(row_number())) %>%
        slice(1)

    # Return only the columns spec
    out <- out %>% select(c(
            year_col,
            type_col,
            out_col
        ))
    return(out)
}
```

## Calculate Yearly Adjusted Road Length

The following function calculates yearly adjusted road lengths by infrastructure type using cumulative sums and filling in missing years and types.

For a given infrastructure type, the total adjusted road length for a given year is expressed below:

$$length^{install}_{year,type} + length^{change_i}_{year,type} - length^{replacement_i}_{year,type}$$

Where:

- $length^{install}_{year,type}$ are the yearly cumulative road lengths for an infrastructure *type* installation
- $length^{change_i}_{year,type}$ are the yearly cumulative road lengths for an infrastructure *type* change in order $i$
- $length^{replacement_i}_{year,type}$ are the yearly cumulative road lengths for an infrastructure *type* replaced by change in order $i$

```r
#' Calculate Yearly Adjusted Road Lengths By Infrastructure Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param year_cols A vector of the names (char) or indices (int) of the columns containing the years o
#' @param type_cols A vector of the names (char) or indices (int) of the columns containing the infrast
#' @param type_col The name (char) of the column containing the type.
#' @param len_cols A vector of the names (char) or indices (int) of the columns containing the road leng
#' @param out_cols The name (char) of the column containing the calculated yearly road lengths by type.
#' @param out_col The name (char) of the column containing the calculated yearly adjusted road lengths
#' @param repl_suffix A suffix (char) to append to the columns representing the road lengths of replace
#' @param ... Additional arguments passed to calc_yearly_len.
#'
#' @return A data.frame with columns containing the year, type, cumulative road lengths of installation
#' @export
#'
calc_yearly_adj_len <- function(
        df,
        year_cols = c("install_year", "upgrade1_year", "upgrade2_year"),
        type_cols = c("install_type", "upgrade1_type", "upgrade2_type"),
        type_col = "type",
        len_cols = "segment_len",
        out_cols = c("install_len", "upgrade1_len", "upgrade2_len"),
        out_col = "adj_len",
        repl_suffix = "_replaced",
        ...
    ) {

    # Convert len_col if char
    len_cols <- rep(len_cols, length(year_cols))

    # Check cols same size
    year_cols_n <- length(year_cols)
    type_cols_n <- length(type_cols)
    len_cols_n <- length(len_cols)
    out_cols_n <- length(out_cols)
    if (length(unique(c(year_cols_n, type_cols_n, len_cols_n, out_cols_n))) != 1) {
        stop(glue(
            "The arguments 'year_cols' ({year_cols_n}), 'type_cols' ({type_cols_n}), 'len_cols' ({len_c
        ))
    }
```

```
# Calc yearly lens by infra type per install or change
out <- list()
for (i in 1:length(year_cols)) {

    # Get year, type, and len cols
    ycol <- year_cols[[i]]
    tcol <- type_cols[[i]]
    lcol <- len_cols[[i]]
    ocol <- out_cols[[i]]

    # Calc yearly len for install or change
    out <- append(
        out,
        calc_yearly_len(
            df,
            year_col = ycol,
            type_col = tcol,
            len_col = lcol,
            out_col = ocol,
            ...
        ) %>%
            rename(
                "year" := !!ycol,
                "type" := !!tcol
            ) %>% list
    )

    # Calc yearly len for replacement
    if (i > 1) {

        # Get repl cols
        tcol_repl <- type_cols[[i - 1]]
        lcol_repl <- len_cols[[i - 1]]

        # Filter for repl records only where type is not eq to change type
        df_repl <- df %>% filter(.data[[tcol]] != .data[[tcol_repl]])

        # Calc repl len if there are any changes
        has_change <- !is.na(df_repl[[tcol]]) %>% all
        if (has_change) {
            out <- append(
                out,
                calc_yearly_len(
                    df_repl,
                    year_col = ycol,
                    type_col = tcol_repl,
                    len_col = lcol_repl,
                    out_col = glue("{ocol}{repl_suffix}"),
                    ...
                ) %>%
                rename(
                    "year" := !!ycol,
                    "type" := !!tcol_repl
```

```
                    ) %>% list
                )
            }
        }
    }

    # Combine all lens in list to single df
    out <- out %>%
        reduce(
            left_join, by = c("year", "type")
        ) %>%
        ungroup()

    # Create template for change and repl cols
    change_cols <- paste0(out_cols[2:out_cols_n])# change cols
    change_cols <- c(change_cols, paste0(out_cols[2:out_cols_n], repl_suffix)) # repl cols
    change_cols_add <- rep(0, length(change_cols)) # set default vals
    names(change_cols_add) <- change_cols

    # Add change and repl cols set to 0 if not present
    out <- out %>% add_column(
        !!!change_cols_add[setdiff(names(change_cols_add), names(.))]
    )

    # Set NA to 0
    out <- out %>% mutate(
        across(everything(), ~replace_na(., 0))
    )

    # Calc yearly adj lens by infra type
    out <- out %>%
        mutate( # added len by infra types due to install or changes
            !!out_col := reduce(across(all_of(out_cols)), `+`)
        ) %>%
        mutate( # removed len by infra types due to replacements
            !!out_col := .data[[out_col]] - reduce(
                across(all_of(
                    paste0(out_cols[2:out_cols_n], repl_suffix)
                )),
                `-`
            )
        )

    # Rename type col
    out <- out %>% rename(!!type_col := type)
    return(out)
}
```

## Plot Lengths by Year for Generic Types

Plots an area chart showing the cumulative road lengths by a user-defined type for each year.

This is a generic function for user-defined types such as infrastructure or road types.

```
#' Plot Yearly Road Lengths By Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param title The title (char) of the plot.
#' @param title_underline Set to TRUE to underline the title.
#' @param x_title The title (char) of the x-axis.
#' @param y_title The title (char) of the y-axis.
#' @param legend_title The title (char) of the legend.
#' @param legend Set to TRUE to include a legend.
#' @param year_col The name (char) or index (int) of the column containing the years.
#' @param year_min The minimum year (int) to display.
#' @param year_max The maximum year (int) to display.
#' @param year_int The year intervals (int) to display. For example, 1 displays every year, and 2 displa
#' @param len_col The name (char) or index (int) of the column containing the road lengths.
#' @param type_col The name (char) or index (int) of the column containing the type.
#' @param type_filter A vector (char) of types to remove fomr the plot.
#' @param type_recode A named vector (char) of names representing types and values representing the valu
#' @param line_50km Set to TRUE to draw the 50 km red reference line.
#' @param line_year Set to a year (int) to draw a reference line for a year. If FALSE, a line will not l
#' @param color_low The bottom color (char) of the type.
#' @param color_high The top color (char) of the type.
#' @return An area ggplot of the cumulative yearly road lengths by type.
#' @export
#'
plot_yearly_len <- function(
        df,
        title = "",
        title_underline = TRUE,
        x_title = "",
        y_title = "",
        legend_title = "Type",
        legend = TRUE,
        year_col = "year",
        year_min = FALSE,
        year_max = FALSE,
        year_int = 1,
        len_col = "adj_len",
        type_col = "type",
        type_filter = c(),
        type_recode = c(),
        line_50km = FALSE,
        line_year = FALSE,
        color_low = "#DFEBF7",
        color_high = "#3683BB"
) {

    # Filter to start and end years
    if (year_min > 0) {
        df <- df %>% filter(
            .data[[year_col]] >= year_min
        )
    }
    if (year_max > 0) {
```

```
        df <- df %>% filter(
            .data[[year_col]] <= year_max
        )
    }

    # Filter out particular infrastructure types
    if (length(type_filter) > 0) {
        df <- df %>% filter(
            !.data[[type_col]] %in% type_filter
        )
    }

    # Recode and reorder category types
    if (length(type_recode) > 0) {

        # Reorder category types
        type_uniq <- unique(df[[type_col]])
        type_reorder <- names(type_recode)
        type_reorder <- c(type_reorder, type_uniq[!type_uniq %in% type_reorder])
        df[[type_col]] <- factor(df[[type_col]], levels = type_reorder)

        # Recode category types
        df[[type_col]] <- recode(df[[type_col]], !!!type_recode)
    }

    # Create fill colors
    type_n <- length(type_uniq)
    type_colors <- scales::seq_gradient_pal(
        color_low,
        color_high
    )(seq(0, 1, length.out = type_n))

    # Create base area plot with legend and labels
    len_max <- max(df[[len_col]], na.rm = TRUE)
    year_max <- max(df[[year_col]], na.rm = TRUE)
    out <- ggplot(
        df,
        aes(
            x = .data[[year_col]],
            y = .data[[len_col]],
            fill = .data[[type_col]],
            order = desc(.data[[type_col]])
        )
    ) +
    geom_area(colour = NA, alpha = 0.7) +
    scale_fill_manual(values = type_colors) +
    geom_line(
        position = "stack",
        size = 0.2
    ) +
    labs(
        x = x_title,
        y = y_title,
```

```
        fill = legend_title
    ) +
    guides(
        fill = FALSE,
        color = FALSE
    ) +
    scale_x_continuous(
        breaks = seq(year_min, year_max, by = year_int),
        labels = seq(year_min, year_max, by = year_int),
        limits = c(year_min, year_max)
    ) +
    scale_y_continuous(
        label = scales::label_number(suffix = " km")
    ) +
    theme_minimal() +
    theme(
        plot.margin = unit(c(5,5,5,5), "points")
    )

    # Add title
    if (title_underline) {
        out <- out + ggtitle(
            bquote(underline(.(title)))
        )
    } else {
        out <- out + ggtitle(title)
    }

    # Add legend
    if (legend) {
        out <- out + guides(fill = guide_legend(
            reverse = FALSE,
            override.aes = list(
                alpha = 0.7,
                color = NA,
                shape = NA
            )
        ))
    }

    # Add dotted year ref line
    if (line_year) {
        out <- out + geom_vline(
            xintercept = line_year,
            color = "black",
            linetype = "dashed"
        )
    }

    # Add red 50km ref line
    if (line_50km) {
        out <- out + geom_segment( # 50km red line
            aes(
```

```
                x = 2009,
                y = 0,
                xend = 2009,
                yend = 50,
                color = "#bb0000",
                hjust = 0.15
            )
        ) +
        geom_segment( # 50km red triangle point down
            aes(
                x = 2009,
                y = 50.01 - (len_max * 0.05),
                xend = 2009,
                yend = 50 - (len_max * 0.05),
                color = "#bb0000",
                hjust = 0.15
            ),
            arrow = arrow(
                length = unit(0.03, "npc"),
                ends = "last",
                type = "closed"
            )
        ) +
        geom_segment( # 50km red triangle point up
            aes(
                x = 2009,
                y = (len_max * 0.05) - 0.01,
                xend = 2009,
                yend = (len_max * 0.05),
                color = "#bb0000",
                hjust = 0.15
            ),
            arrow = arrow(
                length = unit(0.03, "npc"),
                ends = "last",
                type = "closed"
            )
        ) +
        annotate(
            "text",
            x = 2009,
            y = 50,
            label = "50km",
            color = "#bb0000",
            hjust = -0.225
        )
    }
    return(out)
}
```

## Plot Lengths by Year for Infrastructure Types

Plots area charts of yearly road lengths by infrastructure types for a list of data.

This uses the `plot_yearly_len` function.

```
#' Plot Yearly Road Lengths By Infrastructure Type
#'
#' @param df_list A list of data.frame containing the install and change years, type, and road segment l
#' @return An area ggplot of the cumulative yearly road lengths by infrastructure type.
#' @export
#'
plot_yearly_len_infra <- function(df_list) {

    # Create infra plots from data
    p <- list()
    for (i in 1:length(df_list)) {

        # Get data and plot title
        df <- df_list[[i]]
        ptitle <- names(df_list)[[i]]

        # Create and add infra plot to list
        p[[i]] <- calc_yearly_adj_len(df, type_col = settings$type_col_infra) %>%
            plot_yearly_len(
                title = ptitle,
                year_min = settings$year_min,
                year_max = settings$year_max,
                type_col = settings$type_col_infra,
                type_filter = settings$type_filter_infra,
                type_recode = settings$type_recode_infra,
                legend_title = "Infrastructure Type",
                line_50km = TRUE,
                line_year = settings$line_year
            )
    }

    # Y-axis title
    y_title <- ggplot() +
        annotate(
            geom = "text",
            x = 1,
            y = 1,
            label = "Total Length (Centreline km)",
            angle = 90,
            size = 5
        ) +
        coord_cartesian(clip = "off")+
        theme_void()

    # Combine all infra plots together
    out <- (y_title | wrap_plots(p, nrow = length(p))) +
        plot_annotation(
            title = "Roadways with Dedicated Cycling Infrastructure",
            caption = sprintf("Years (%s-%s)", settings$year_min, settings$year_max),
            theme = theme(
                plot.title = element_text(hjust = 0.5, size = 16),
                plot.caption = element_text(hjust = 0.5, size = 14)
```

```
            )
        ) +
        plot_layout(widths = c(0.05, 1))
    return(out)
}
```

## Plot Lengths by Year for Road Types

Plots area charts of yearly road lengths by overall road type and by infrastructure separated by each road type.

This uses the `plot_yearly_len` function.

```
#' Plot Yearly Road Lengths By Road Type
#'
#' @param df The data.frame containing the install and change years, type, and road segment types and le
#' @return An area ggplot of the cumulative yearly road lengths by road type.
#' @export
#'
plot_yearly_len_road <- function(df, title = "Roadways with Dedicated Cycling Infrastructure") {

    # Create list to store plots
    p <- list()

    # Plot overall road types
    p[[1]] <- calc_yearly_len(
        df,
        year_col = settings$year_col_road,
        type_col = settings$type_col_road
    ) %>%
        plot_yearly_len(
            year_col = settings$year_col_road,
            year_min = settings$year_min,
            year_max = settings$year_max,
            y_title = "Total Length (Centreline km)",
            legend_title = "Roadway Type",
            type_col = settings$type_col_road,
            type_recode = settings$type_recode_road,
            len_col = "len",
            line_50km = FALSE,
            line_year = settings$line_year,
            color_low = "#C1DDB3",
            color_high = "#297A22"
        )

    # Plot arterial, collector, and local road by infra
    rtypes <- c("Arterial", "Collector", "Local")
    for (i in 1:length(rtypes)) {

        # Get road type
        r <- rtypes[i]

        # Create infra plot for road type
        p[[i + 1]] <- calc_yearly_adj_len(
```

```
            df %>% filter(road_type == r),
            type_col = settings$type_col_infra
        ) %>%
            plot_yearly_len(
                title = sprintf("%s Roadways", r),
                title_underline = FALSE,
                line_50km = FALSE,
                line_year = settings$line_year,
                year_int = 2,
                y_title = "Total Length (Centreline km)",
                year_min = settings$year_min,
                year_max = settings$year_max,
                type_col = settings$type_col_infra,
                type_filter = settings$type_filter_infra,
                type_recode = settings$type_recode_infra,
                legend_title = "Infrastructure Type"
            )
    }

    # Plot overall and road type plots together
    out <- (
        plot_spacer() +
        p[[1]] +
        plot_spacer() +
        plot_layout(
            widths = c(0.25, 0.3, 0.2)
        ) +
        theme(plot.margin = margin(0.15, 0, 0.15, 0, "in"))
    ) / (
        p[[2]] +
        p[[3]] +
        p[[4]] +
        theme(plot.margin = margin(0, 0, 0.15, 0, "in"))
    ) + plot_annotation(
        title = title,
        caption = sprintf("Years (%s-%s)", settings$year_min, settings$year_max),
        tag_levels = list(c("A", "B", "", "")),
        theme = theme(
            plot.title = element_text(hjust = 0.5, size = 26),
            plot.caption = element_text(hjust = 0.5, size = 22),
        )
    ) & theme(
        plot.tag = element_text(face = "bold", size = 22)
    )
    return(out)
}
```

# Data

## Vancouver Data

```r
# Load raw data
vanc_bikeways <- read_csv("../data/vancouver_bikeways_2009_2022_v1.csv")
```

```
## Rows: 745 Columns: 89
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (73): bike_rout0, street_na0, bikeway_t0, subtype, status, street_se0, o...
## dbl (16): OID_, object_id, speed_lim0, year_of_c0, upgrade_y0, ID_DATAENTRY,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
vanc_roads <- read_csv("../data/vancouver_roads_2009_2022_v1.csv")
```

```
## Rows: 780 Columns: 72
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (61): ID_ROUTE, DPR_CHECK_FLAG, DPR_ENTRY, DPR_EXCL_FLAG, DPR_EXCL1318_R...
## dbl (11): ID_CITY, ID_DATAENTRY, DPR_ORDER, ATR_SEGMENT_LENGTH, ATR_SPEEDLIM...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Combine raw data
vanc <- vanc_bikeways %>%
    select(
        ID_DATAENTRY,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
        UPGR2_MIN_TYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
    left_join(
        vanc_roads %>% select(
            ID_DATAENTRY,
            ATR_SEGMENT_TYPE
        ),
        by = "ID_DATAENTRY"
    ) %>%
    rename(
        id = ID_DATAENTRY,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_TYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = ATR_SEGMENT_TYPE
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
```

```
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Arterial"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Collector",
                "Secondary Arterial",
                "Sec Arterial"
            ) ~ "Collector",
            segment_type %in% c( # local equiv
                "Lane",
                "Residential",
                "Leased",
                "Recreational"
            ) ~ "Local",
            .default = segment_type
        )
    )
vanc
```

```
## # A tibble: 745 x 10
##       id install_year install_type upgrade1_year upgrade1_type upgrade2_year
##    <dbl>        <dbl> <chr>                <dbl> <chr>                 <dbl>
##  1   775         2014 PBL                     NA <NA>                     NA
##  2   774         2014 PBL                     NA <NA>                     NA
##  3   773         1999 None                  2021 PBL                      NA
##  4   770         2015 PL                      NA <NA>                     NA
##  5   769         2015 PL                      NA <NA>                     NA
##  6   768         2015 PL                      NA <NA>                     NA
##  7   767         2015 PL                      NA <NA>                     NA
##  8   766         2015 PL                      NA <NA>                     NA
##  9   765         2015 PL                      NA <NA>                     NA
## 10   764         2015 PL                      NA <NA>                     NA
## # i 735 more rows
## # i 4 more variables: upgrade2_type <chr>, segment_len <dbl>,
## #   segment_type <chr>, road_type <chr>
```

## Calgary Data

```
# Load raw data
calg_bikeways <- read_csv("../data/calgary_bikeways_2009_2022_v1.csv")
```

```
## Rows: 750 Columns: 54
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (41): STATUS, TYPE, BICYCLE_CLASS, COMFORT_LEVEL, CURRENT_TYPE_VERIFIED...
## dbl  (11): ORIG_ID, ATR_SEGMENT_LENGTH, INST_YR, UPGR2_YR, SHAPE_ID, STARTIN...
## date  (2): CREATED_DT, MODIFIED_DT
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
calg_roads <- read_csv("../data/calgary_roads_2009_2022_v1.csv")
```

```
## Rows: 4170 Columns: 39
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (18): status, type, bicycle_cl, comfort_le, date_creat, date_modif, ful...
## dbl  (15): OID_, len_m, lenm, startx, starty, endx, endy, shape_id, OBJECTID...
## lgl   (1): length
## time  (5): time_creat, time_modif, time_creat_1, time_modif_1, time_mod_2
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Combine raw data
calg <- calg_bikeways %>%
    select(
        SHAPE_ID,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
        UPGR2_MIN_HTYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
    left_join(
        calg_roads %>% select(
            shape_id,
            ctp_class
        ),
        by = join_by(SHAPE_ID == shape_id)
    ) %>%
    rename(
        id = SHAPE_ID,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_HTYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = ctp_class
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Arterial Street",
                "Industrial Arterial",
                "Local Arterial",
                "Parkway",
                "Urban Boulevard"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Neighbourhood Boulevard",
                "Collector",
                "Primary Collector",
```

```
                "Skeletal Road"
            ) ~ "Collector",
            segment_type %in% c( # local equiv
                "Access Route",
                "Residential Street",
                "Activity Center Street",
                "Historic Road Allowance",
                "Lanes (Alleys)",
                "Industrial Street"
            ) ~ "Local",
            .default = segment_type
        )
    )
calg
```

```
## # A tibble: 750 x 10
##       id install_year install_type upgrade1_year upgrade1_type upgrade2_year
##    <dbl>        <dbl> <chr>        <chr>         <chr>                 <dbl>
##  1   498         2011 PL           <NA>          <NA>                     NA
##  2   497         2011 PL           <NA>          <NA>                     NA
##  3   499         2011 PL           <NA>          <NA>                     NA
##  4   493         2012 None         2015          PL                       NA
##  5  1574         2014 PL           <NA>          <NA>                     NA
##  6  1572         2014 PL           <NA>          <NA>                     NA
##  7   671         2009 PL           <NA>          <NA>                     NA
##  8  2549         2021 PBL          <NA>          <NA>                     NA
##  9  2558         2021 PBL          <NA>          <NA>                     NA
## 10  2560         2021 PBL          <NA>          <NA>                     NA
## # i 740 more rows
## # i 4 more variables: upgrade2_type <chr>, segment_len <dbl>,
## #   segment_type <chr>, road_type <chr>
```

### Toronto Data

```
# Load raw data
toron_bikeways <- read_csv("../data/toronto_bikeways_2009_2022_v1.csv")
```

```
## Rows: 326 Columns: 53
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (35): CITY_INFRA_HIGHORDER, CITY_INFRA_LOWORDER, STREET_NAME, FROM_STRE...
## dbl  (16): ID_OID, ID_DATAENTRY, ID_1_OBJ2, OBJECTID, CITY_INST_YR, CITY_UPG...
## lgl   (1): DPR_EXCL_FLAG
## dttm  (1): CITY_LAST_REVIEWED
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
toron_roads <- read_csv("../data/toronto_roads_2009_2022_v1.csv")
```

```
## Rows: 331 Columns: 59
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (14): STREET_7, FROM_ST8, TO_STRE9, INFRA_L15, INFRA_H20, LINEAR_26, LI...
```

```
## dbl  (28): OID_, _id1, OBJECTI2, SEGMENT3, INSTALL4, UPGRADE5, CONVERT28, st...
## lgl  (16): PRE_AMA6, ROADCLA10, CNPCLAS11, SURFACE12, OWNER13, DIR_LOW14, SE...
## dttm  (1): LAST_ED26
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Combine raw data
toron <- toron_bikeways %>%
    select(
        ID_OID,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
        UPGR2_MIN_HTYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
    left_join(
        toron_roads %>% select(
            OID_,
            FEATURE36
        ),
        by = join_by(ID_OID == OID_)
    ) %>%
    rename(
        id = ID_OID,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_HTYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = FEATURE36
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Major Arterial",
                "Major Arterial Ramp",
                "Minor Arterial"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Collector"
            ) ~ "Collector",
            segment_type %in% c(  # local equiv
                "Local",
                "Other"
            ) ~ "Local",
            .default = segment_type
        )
```

```
    )
toron
```

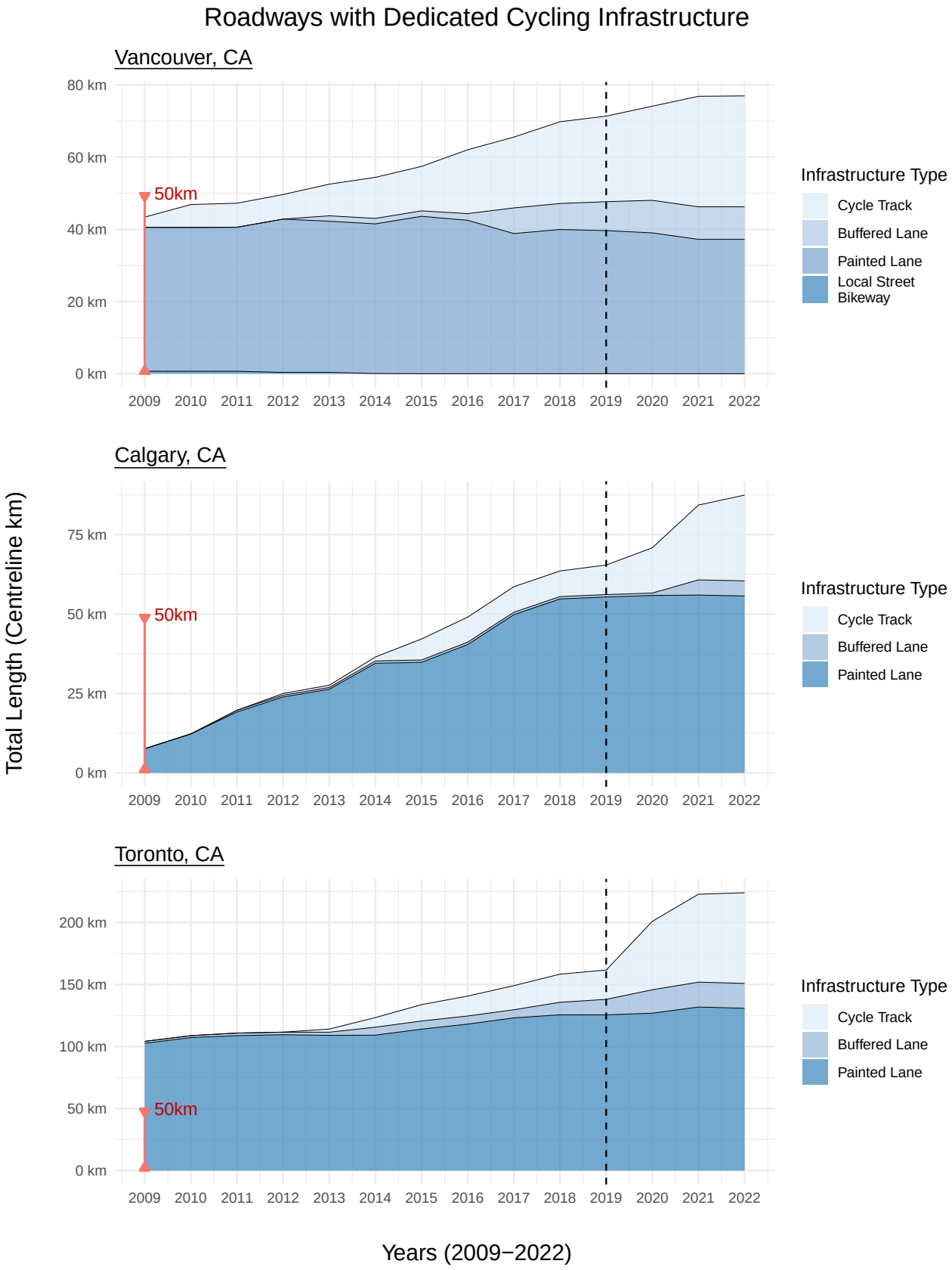
```
## # A tibble: 326 x 10
##       id install_year install_type upgrade1_year upgrade1_type upgrade2_year
##    <dbl>        <dbl> <chr>                <dbl> <chr>                  <dbl>
##  1  1133         2015 PBL                   2020 PBL                       NA
##  2  1136         2015 PL                    2020 PL                        NA
##  3  1135         2015 BUF                   2020 BUF                       NA
##  4  1134         2014 PBL                   2020 PBL                       NA
##  5  1004         2009 PL                      NA <NA>                      NA
##  6  1009         2009 PL                      NA <NA>                      NA
##  7  1220         2015 None                  2020 PL                        NA
##  8  1229         2015 None                  2020 PL                        NA
##  9  1230         2015 PL                      NA <NA>                      NA
## 10  1145         2015 PL                      NA <NA>                      NA
## # i 316 more rows
## # i 4 more variables: upgrade2_type <chr>, segment_len <dbl>,
## #   segment_type <chr>, road_type <chr>
```

# Figures

### Figure 2: Changes in dedicated cycling infrastructure between 2009 and 2022 for Vancouver, Calgary, and Toronto by infrastructure category.

Assessed using roadway centreline-km, with infrastructure classifications determined by the most protective element present along each road segment.

```
plot_yearly_len_infra(list(
    "Vancouver, CA" = vanc,
    "Calgary, CA" = calg,
    "Toronto, CA" = toron
))
```

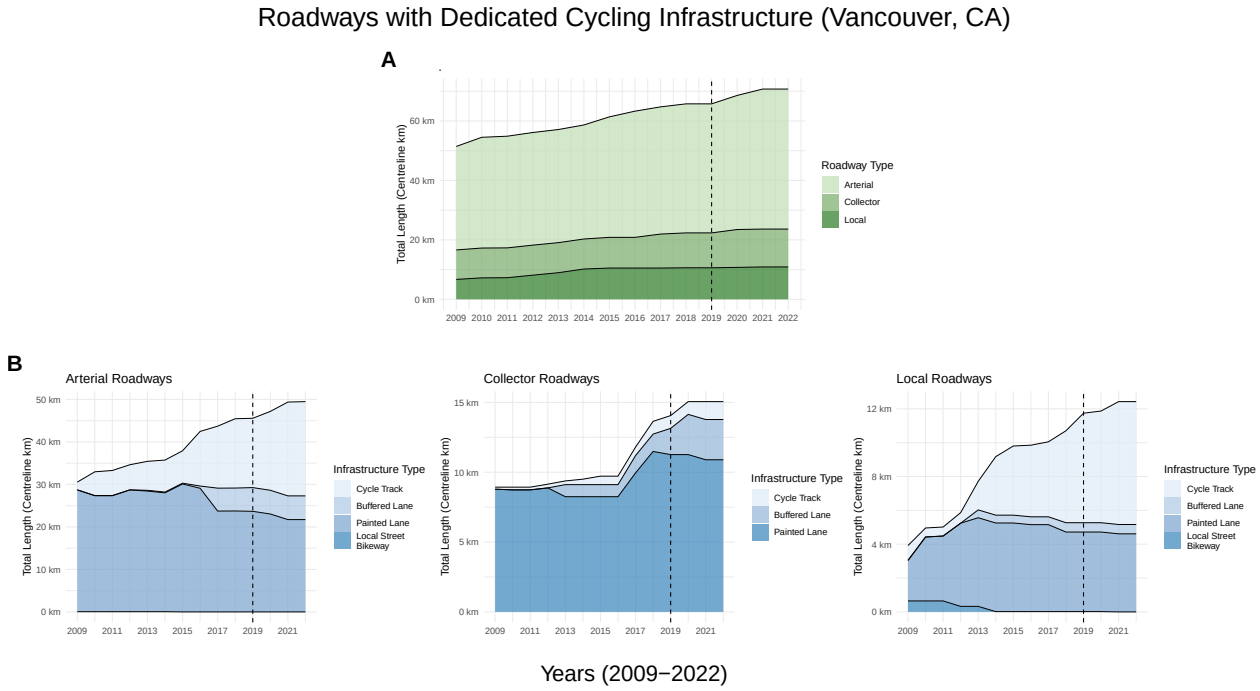

Roadways with Dedicated Cycling Infrastructure
Vancouver, CA
Calgary, CA
Toronto, CA
Total Length (Centreline km)
Years (2009−2022)
50km
Infrastructure Type
Cycle Track
Buffered Lane
Painted Lane
Local Street Bikeway
80 km
60 km
40 km
20 km
0 km
75 km
50 km
25 km
200 km
150 km
100 km
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022

# Supplementary Figures

## Supplementary Figure 4: Changes in dedicated cycling infrastructure between 2009 and 2021 for the Municipality of Vancouver, CA

By (A) roadway classification, and (B) infrastructure distribution within each road class. Assessed using roadway centreline-km, with infrastructure classification determined by the most protective element present along each road segment.

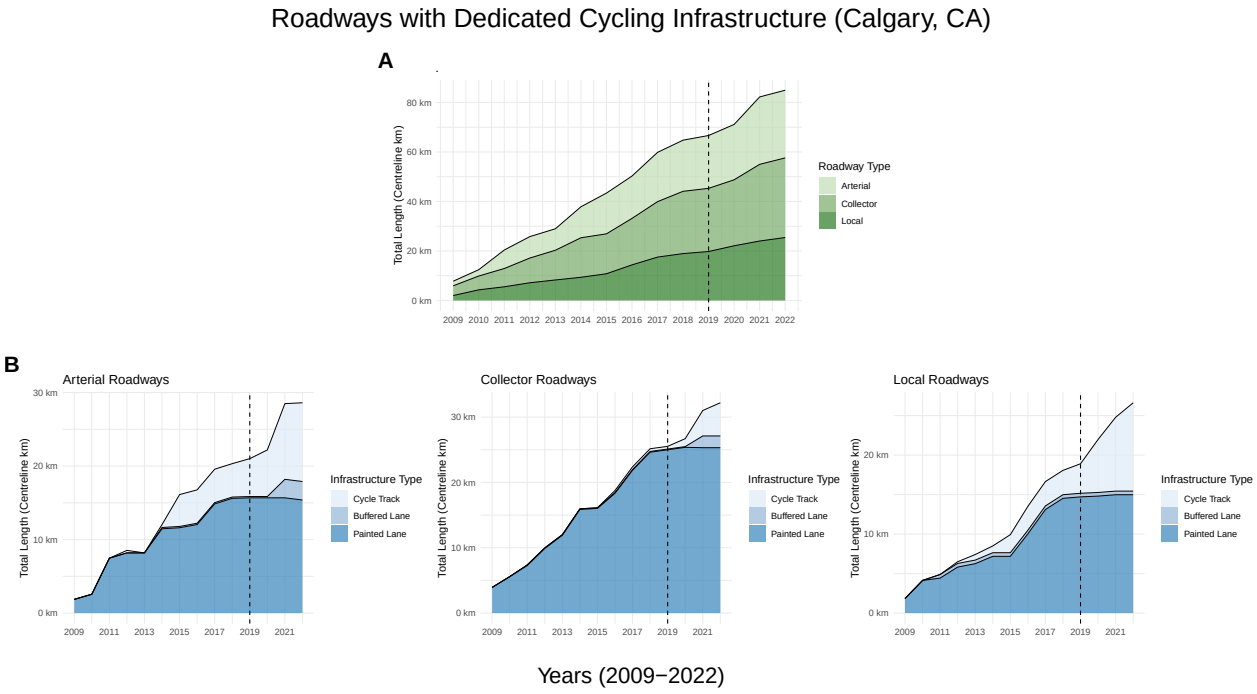
```
plot_yearly_len_road(
    vanc,
    title = "Roadways with Dedicated Cycling Infrastructure (Vancouver, CA)"
)
```




## Supplementary Figure 5: Changes in dedicated cycling infrastructure between 2009 and 2022 for the Municipality of Calgary, CA

By (A) roadway classification, and (B) infrastructure distribution within each road class. Assessed using roadway centreline-km, with infrastructure classification determined by the most protective element present along each road segment.

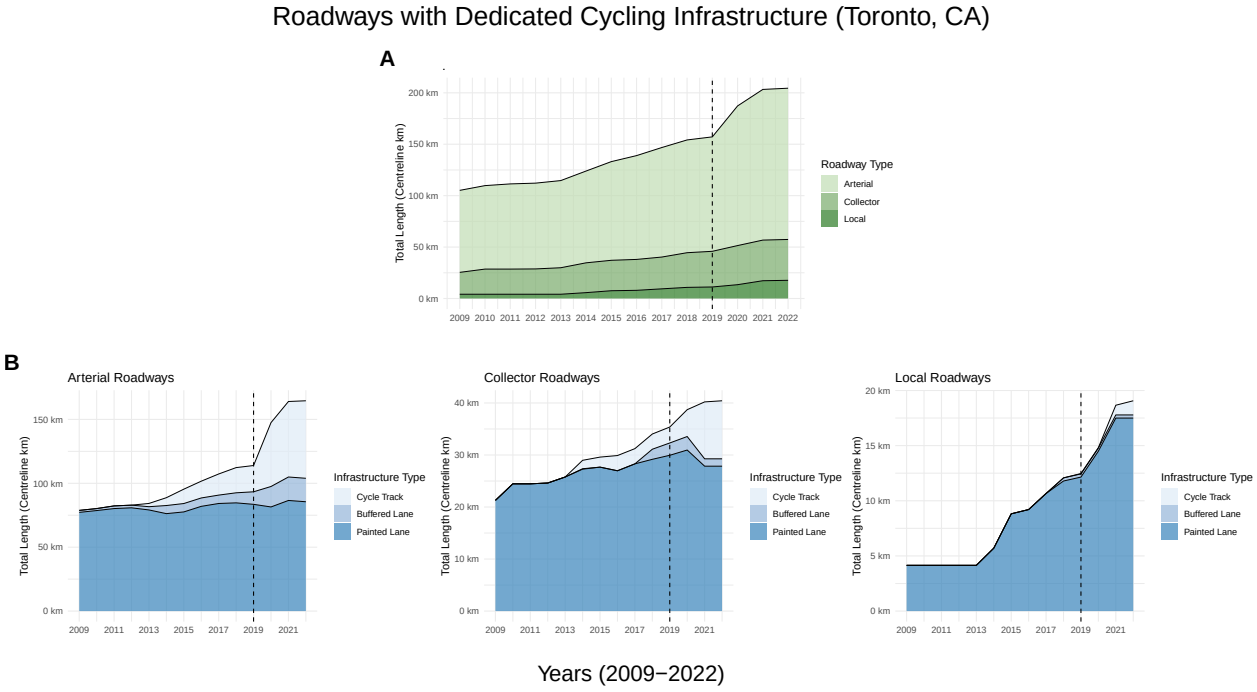
```
plot_yearly_len_road(
    calg,
    title = "Roadways with Dedicated Cycling Infrastructure (Calgary, CA)"
)
```




## Supplementary Figure 6: Changes in dedicated cycling infrastructure between 2009 and 2022 for the Municipality of Toronto, CA

By (A) roadway classification, and (B) infrastructure distribution within each road class. Assessed using roadway centreline-km, with infrastructure classification determined by the most protective element present along each road segment.

```
plot_yearly_len_road(
    toron,
    title = "Roadways with Dedicated Cycling Infrastructure (Toronto, CA)"
)
```




# Appendix

## R Version

R and RMarkdown in RStudio was used to generate this document.

```
##                _
## platform       x86_64-apple-darwin20
## arch           x86_64
## os             darwin20
## system         x86_64, darwin20
## status
## major          4
## minor          3.1
## year           2023
## month          06
## day            16
## svn rev        84548
## language       R
## version.string R version 4.3.1 (2023-06-16)
## nickname       Beagle Scouts
```

## R Code

The R script below runs all the code in this document.

```
knitr::opts_chunk$set(warning = FALSE)
install.packages("rmarkdown")
install.packages("bookdown")
install.packages("knitr")
install.packages("tidyverse")
install.packages("glue")
```

```r
install.packages("readxl")
install.packages("ggtext")
install.packages("scales")
install.packages("patchwork")
library(tidyverse)
library(ggtext)
library(glue)
library(patchwork)
library(readxl)
settings <- list()

# Infrastructure types in order
settings$type_recode_infra <- c(
    PBL = "Cycle Track",
    BUF = "Buffered Lane",
    PL = "Painted Lane",
    LSB = "Local Street\nBikeway"
)

# Infrastructure types to remove
settings$type_filter_infra <- c("N", "None", "SR")

# Road types in order
settings$type_recode_road <- c(
    Arterial = "Arterial",
    Collector = "Collector",
    Local = "Local"
)

# Column references
settings$year_col_road <- "install_year"
settings$type_col_road <- "road_type"
settings$type_col_infra <- "infra_type"

# Set years of interest
settings$year_min <- 2009
settings$year_max <- 2022

# Plot settings
settings$line_year <- 2019

#' Calculate Yearly Road Lengths By Infrastructure Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param year_col The name (char) or index (int) of the column containing the years.
#' @param type_col The name (char) or index (int) of the column containing the infrastructure type
#' @param len_col The name (char) or index (int) of the column containing the road lengths.
#' @param out_col The name (char) of the column containing the calculated yearly road lengths by type.
#'
#' @return A data.frame with three columns containing the year, type, and calculated yearly road lengths
#' @export
#'
calc_yearly_len <- function(
```

```
    df,
    year_col = "install_year",
    type_col = "install_type",
    len_col = "segment_len",
    out_col = "len",
    year_min = settings$year_min,
    year_max = settings$year_max
) {

# Convert data types
df[[year_col]] <- as.integer(df[[year_col]])
df[[type_col]] <- as.character(df[[type_col]])
df[[len_col]] <- as.numeric(df[[len_col]])

# Remove rows with empty type
out <- df %>% filter(
    !is.na(.data[[type_col]])
)

# Filter to min and max years
if (year_min > 0) {
    df <- df %>% filter(
        .data[[year_col]] >= year_min
    )
} else {
    year_min <- min(out[[year_col]], na.rm = TRUE)
}
if (year_max > 0) {
    df <- df %>% filter(
        .data[[year_col]] <= year_max
    )
} else {
    year_max <- max(out[[year_col]], na.rm = TRUE)
}

# Add dummy len for each type and year combo
# Covers cases where type and year combo does not exist
# E.g. No new PL installs in 2021, hence a record PL in 2021 does not exist
type_uniq <- unique(out[[type_col]])
type_n <- length(type_uniq)
year_uniq <- year_min:year_max
year_n <- length(year_uniq)
out <- out %>% add_row(
    !!year_col := rep(year_uniq, each = type_n),
    !!type_col := rep(type_uniq, year_n),
    !!len_col := rep(0, type_n * year_n)
)

# Calc cumsum for each non-empty type ordered by year
out <- out %>%
    arrange(.data[[year_col]]) %>%
    group_by(.data[[type_col]]) %>%
    mutate(
```

```
            !!out_col := cumsum(.data[[len_col]])
        )

    # Get the last cumsum for each year and type
    out <- out %>%
        group_by(.data[[year_col]], .data[[type_col]]) %>%
        arrange(desc(row_number())) %>%
        slice(1)

    # Return only the columns spec
    out <- out %>% select(c(
            year_col,
            type_col,
            out_col
        ))
    return(out)
}

#' Calculate Yearly Adjusted Road Lengths By Infrastructure Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param year_cols A vector of the names (char) or indices (int) of the columns containing the years o
#' @param type_cols A vector of the names (char) or indices (int) of the columns containing the infrast
#' @param type_col The name (char) of the column containing the type.
#' @param len_cols A vector of the names (char) or indices (int) of the columns containing the road leng
#' @param out_cols The name (char) of the column containing the calculated yearly road lengths by type.
#' @param out_col The name (char) of the column containing the calculated yearly adjusted road lengths
#' @param repl_suffix A suffix (char) to append to the columns representing the road lengths of replace
#' @param ... Additional arguments passed to calc_yearly_len.
#'
#' @return A data.frame with columns containing the year, type, cumulative road lengths of installation
#' @export
#'
calc_yearly_adj_len <- function(
        df,
        year_cols = c("install_year", "upgrade1_year", "upgrade2_year"),
        type_cols = c("install_type", "upgrade1_type", "upgrade2_type"),
        type_col = "type",
        len_cols = "segment_len",
        out_cols = c("install_len", "upgrade1_len", "upgrade2_len"),
        out_col = "adj_len",
        repl_suffix = "_replaced",
        ...
    ) {

    # Convert len_col if char
    len_cols <- rep(len_cols, length(year_cols))

    # Check cols same size
    year_cols_n <- length(year_cols)
    type_cols_n <- length(type_cols)
    len_cols_n <- length(len_cols)
    out_cols_n <- length(out_cols)
```

```
    if (length(unique(c(year_cols_n, type_cols_n, len_cols_n, out_cols_n))) != 1) {
        stop(glue(
            "The arguments 'year_cols' ({year_cols_n}), 'type_cols' ({type_cols_n}), 'len_cols' ({len_c
        ))
    }

    # Calc yearly lens by infra type per install or change
    out <- list()
    for (i in 1:length(year_cols)) {

        # Get year, type, and len cols
        ycol <- year_cols[[i]]
        tcol <- type_cols[[i]]
        lcol <- len_cols[[i]]
        ocol <- out_cols[[i]]

        # Calc yearly len for install or change
        out <- append(
            out,
            calc_yearly_len(
                df,
                year_col = ycol,
                type_col = tcol,
                len_col = lcol,
                out_col = ocol,
                ...
            ) %>%
                rename(
                    "year" := !!ycol,
                    "type" := !!tcol
                ) %>% list
        )

        # Calc yearly len for replacement
        if (i > 1) {

            # Get repl cols
            tcol_repl <- type_cols[[i - 1]]
            lcol_repl <- len_cols[[i - 1]]

            # Filter for repl records only where type is not eq to change type
            df_repl <- df %>% filter(.data[[tcol]] != .data[[tcol_repl]])

            # Calc repl len if there are any changes
            has_change <- !is.na(df_repl[[tcol]]) %>% all
            if (has_change) {
                out <- append(
                    out,
                    calc_yearly_len(
                        df_repl,
                        year_col = ycol,
                        type_col = tcol_repl,
                        len_col = lcol_repl,
```

```
                        out_col = glue("{ocol}{repl_suffix}"),
                        ...
                    ) %>%
                    rename(
                        "year" := !!ycol,
                        "type" := !!tcol_repl
                    ) %>% list
                )
            }
        }
    }

    # Combine all lens in list to single df
    out <- out %>%
        reduce(
            left_join, by = c("year", "type")
        ) %>%
        ungroup()

    # Create template for change and repl cols
    change_cols <- paste0(out_cols[2:out_cols_n])# change cols
    change_cols <- c(change_cols, paste0(out_cols[2:out_cols_n], repl_suffix)) # repl cols
    change_cols_add <- rep(0, length(change_cols)) # set default vals
    names(change_cols_add) <- change_cols

    # Add change and repl cols set to 0 if not present
    out <- out %>% add_column(
        !!!change_cols_add[setdiff(names(change_cols_add), names(.))]
    )

    # Set NA to 0
    out <- out %>% mutate(
        across(everything(), ~replace_na(., 0))
    )

    # Calc yearly adj lens by infra type
    out <- out %>%
        mutate( # added len by infra types due to install or changes
            !!out_col := reduce(across(all_of(out_cols)), `+`)
        ) %>%
        mutate( # removed len by infra types due to replacements
            !!out_col := .data[[out_col]] - reduce(
                across(all_of(
                    paste0(out_cols[2:out_cols_n], repl_suffix)
                )),
                `_`
            )
        )

    # Rename type col
    out <- out %>% rename(!!type_col := type)
    return(out)
}
```

```r
#' Plot Yearly Road Lengths By Type
#'
#' @param df A data.frame with three columns containing the year, type, and road lengths.
#' @param title The title (char) of the plot.
#' @param title_underline Set to TRUE to underline the title.
#' @param x_title The title (char) of the x-axis.
#' @param y_title The title (char) of the y-axis.
#' @param legend_title The title (char) of the legend.
#' @param legend Set to TRUE to include a legend.
#' @param year_col The name (char) or index (int) of the column containing the years.
#' @param year_min The minimum year (int) to display.
#' @param year_max The maximum year (int) to display.
#' @param year_int The year intervals (int) to display. For example, 1 displays every year, and 2 displa
#' @param len_col The name (char) or index (int) of the column containing the road lengths.
#' @param type_col The name (char) or index (int) of the column containing the type.
#' @param type_filter A vector (char) of types to remove fomr the plot.
#' @param type_recode A named vector (char) of names representing types and values representing the valu
#' @param line_50km Set to TRUE to draw the 50 km red reference line.
#' @param line_year Set to a year (int) to draw a reference line for a year. If FALSE, a line will not b
#' @param color_low The bottom color (char) of the type.
#' @param color_high The top color (char) of the type.
#' @return An area ggplot of the cumulative yearly road lengths by type.
#' @export
#'
plot_yearly_len <- function(
        df,
        title = "",
        title_underline = TRUE,
        x_title = "",
        y_title = "",
        legend_title = "Type",
        legend = TRUE,
        year_col = "year",
        year_min = FALSE,
        year_max = FALSE,
        year_int = 1,
        len_col = "adj_len",
        type_col = "type",
        type_filter = c(),
        type_recode = c(),
        line_50km = FALSE,
        line_year = FALSE,
        color_low = "#DFEBF7",
        color_high = "#3683BB"
) {

    # Filter to start and end years
    if (year_min > 0) {
        df <- df %>% filter(
            .data[[year_col]] >= year_min
        )
    }
    if (year_max > 0) {
```

```
        df <- df %>% filter(
            .data[[year_col]] <= year_max
        )
    }

    # Filter out particular infrastructure types
    if (length(type_filter) > 0) {
        df <- df %>% filter(
            !.data[[type_col]] %in% type_filter
        )
    }

    # Recode and reorder category types
    if (length(type_recode) > 0) {

        # Reorder category types
        type_uniq <- unique(df[[type_col]])
        type_reorder <- names(type_recode)
        type_reorder <- c(type_reorder, type_uniq[!type_uniq %in% type_reorder])
        df[[type_col]] <- factor(df[[type_col]], levels = type_reorder)

        # Recode category types
        df[[type_col]] <- recode(df[[type_col]], !!!type_recode)
    }

    # Create fill colors
    type_n <- length(type_uniq)
    type_colors <- scales::seq_gradient_pal(
        color_low,
        color_high
    )(seq(0, 1, length.out = type_n))

    # Create base area plot with legend and labels
    len_max <- max(df[[len_col]], na.rm = TRUE)
    year_max <- max(df[[year_col]], na.rm = TRUE)
    out <- ggplot(
        df,
        aes(
            x = .data[[year_col]],
            y = .data[[len_col]],
            fill = .data[[type_col]],
            order = desc(.data[[type_col]])
        )
    ) +
    geom_area(colour = NA, alpha = 0.7) +
    scale_fill_manual(values = type_colors) +
    geom_line(
        position = "stack",
        size = 0.2
    ) +
    labs(
        x = x_title,
        y = y_title,
```

```
        fill = legend_title
    ) +
    guides(
        fill = FALSE,
        color = FALSE
    ) +
    scale_x_continuous(
        breaks = seq(year_min, year_max, by = year_int),
        labels = seq(year_min, year_max, by = year_int),
        limits = c(year_min, year_max)
    ) +
    scale_y_continuous(
        label = scales::label_number(suffix = " km")
    ) +
    theme_minimal() +
    theme(
        plot.margin = unit(c(5,5,5,5), "points")
    )

    # Add title
    if (title_underline) {
        out <- out + ggtitle(
            bquote(underline(.(title)))
        )
    } else {
        out <- out + ggtitle(title)
    }

    # Add legend
    if (legend) {
        out <- out + guides(fill = guide_legend(
            reverse = FALSE,
            override.aes = list(
                alpha = 0.7,
                color = NA,
                shape = NA
            )
        ))
    }

    # Add dotted year ref line
    if (line_year) {
        out <- out + geom_vline(
            xintercept = line_year,
            color = "black",
            linetype = "dashed"
        )
    }

    # Add red 50km ref line
    if (line_50km) {
        out <- out + geom_segment( # 50km red line
            aes(
```

```
                x = 2009,
                y = 0,
                xend = 2009,
                yend = 50,
                color = "#bb0000",
                hjust = 0.15
            )
        ) +
        geom_segment( # 50km red triangle point down
            aes(
                x = 2009,
                y = 50.01 - (len_max * 0.05),
                xend = 2009,
                yend = 50 - (len_max * 0.05),
                color = "#bb0000",
                hjust = 0.15
            ),
            arrow = arrow(
                length = unit(0.03, "npc"),
                ends = "last",
                type = "closed"
            )
        ) +
        geom_segment( # 50km red triangle point up
            aes(
                x = 2009,
                y = (len_max * 0.05) - 0.01,
                xend = 2009,
                yend = (len_max * 0.05),
                color = "#bb0000",
                hjust = 0.15
            ),
            arrow = arrow(
                length = unit(0.03, "npc"),
                ends = "last",
                type = "closed"
            )
        ) +
        annotate(
            "text",
            x = 2009,
            y = 50,
            label = "50km",
            color = "#bb0000",
            hjust = -0.225
        )
    }
    return(out)
}

#' Plot Yearly Road Lengths By Infrastructure Type
#'
#' @param df_list A list of data.frame containing the install and change years, type, and road segment l
```

```r
#' @return An area ggplot of the cumulative yearly road lengths by infrastructure type.
#' @export
#'
plot_yearly_len_infra <- function(df_list) {

    # Create infra plots from data
    p <- list()
    for (i in 1:length(df_list)) {

        # Get data and plot title
        df <- df_list[[i]]
        ptitle <- names(df_list)[[i]]

        # Create and add infra plot to list
        p[[i]] <- calc_yearly_adj_len(df, type_col = settings$type_col_infra) %>%
            plot_yearly_len(
                title = ptitle,
                year_min = settings$year_min,
                year_max = settings$year_max,
                type_col = settings$type_col_infra,
                type_filter = settings$type_filter_infra,
                type_recode = settings$type_recode_infra,
                legend_title = "Infrastructure Type",
                line_50km = TRUE,
                line_year = settings$line_year
            )
    }

    # Y-axis title
    y_title <- ggplot() +
        annotate(
            geom = "text",
            x = 1,
            y = 1,
            label = "Total Length (Centreline km)",
            angle = 90,
            size = 5
        ) +
        coord_cartesian(clip = "off")+
        theme_void()

    # Combine all infra plots together
    out <- (y_title | wrap_plots(p, nrow = length(p))) +
        plot_annotation(
            title = "Roadways with Dedicated Cycling Infrastructure",
            caption = sprintf("Years (%s-%s)", settings$year_min, settings$year_max),
            theme = theme(
                plot.title = element_text(hjust = 0.5, size = 16),
                plot.caption = element_text(hjust = 0.5, size = 14)
            )
        ) +
        plot_layout(widths = c(0.05, 1))
    return(out)
```

```r
}

#' Plot Yearly Road Lengths By Road Type
#'
#' @param df The data.frame containing the install and change years, type, and road segment types and le
#' @return An area ggplot of the cumulative yearly road lengths by road type.
#' @export
#'
plot_yearly_len_road <- function(df, title = "Roadways with Dedicated Cycling Infrastructure") {

    # Create list to store plots
    p <- list()

    # Plot overall road types
    p[[1]] <- calc_yearly_len(
        df,
        year_col = settings$year_col_road,
        type_col = settings$type_col_road
    ) %>%
        plot_yearly_len(
            year_col = settings$year_col_road,
            year_min = settings$year_min,
            year_max = settings$year_max,
            y_title = "Total Length (Centreline km)",
            legend_title = "Roadway Type",
            type_col = settings$type_col_road,
            type_recode = settings$type_recode_road,
            len_col = "len",
            line_50km = FALSE,
            line_year = settings$line_year,
            color_low = "#C1DDB3",
            color_high = "#297A22"
        )

    # Plot arterial, collector, and local road by infra
    rtypes <- c("Arterial", "Collector", "Local")
    for (i in 1:length(rtypes)) {

        # Get road type
        r <- rtypes[i]

        # Create infra plot for road type
        p[[i + 1]] <- calc_yearly_adj_len(
            df %>% filter(road_type == r),
            type_col = settings$type_col_infra
        ) %>%
            plot_yearly_len(
                title = sprintf("%s Roadways", r),
                title_underline = FALSE,
                line_50km = FALSE,
                line_year = settings$line_year,
                year_int = 2,
                y_title = "Total Length (Centreline km)",
```

```
                year_min = settings$year_min,
                year_max = settings$year_max,
                type_col = settings$type_col_infra,
                type_filter = settings$type_filter_infra,
                type_recode = settings$type_recode_infra,
                legend_title = "Infrastructure Type"
            )
    }

    # Plot overall and road type plots together
    out <- (
        plot_spacer() +
        p[[1]] +
        plot_spacer() +
        plot_layout(
            widths = c(0.25, 0.3, 0.2)
        ) +
        theme(plot.margin = margin(0.15, 0, 0.15, 0, "in"))
    ) / (
        p[[2]] +
        p[[3]] +
        p[[4]] +
        theme(plot.margin = margin(0, 0, 0.15, 0, "in"))
    ) + plot_annotation(
        title = title,
        caption = sprintf("Years (%s-%s)", settings$year_min, settings$year_max),
        tag_levels = list(c("A", "B", "", "")),
        theme = theme(
            plot.title = element_text(hjust = 0.5, size = 26),
            plot.caption = element_text(hjust = 0.5, size = 22),
        )
    ) & theme(
        plot.tag = element_text(face = "bold", size = 22)
    )
    return(out)
}

# Load raw data
vanc_bikeways <- read_csv("../data/vancouver_bikeways_2009_2022_v1.csv")
vanc_roads <- read_csv("../data/vancouver_roads_2009_2022_v1.csv")

# Combine raw data
vanc <- vanc_bikeways %>%
    select(
        ID_DATAENTRY,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
        UPGR2_MIN_TYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
```

```
    left_join(
        vanc_roads %>% select(
            ID_DATAENTRY,
            ATR_SEGMENT_TYPE
        ),
        by = "ID_DATAENTRY"
    ) %>%
    rename(
        id = ID_DATAENTRY,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_TYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = ATR_SEGMENT_TYPE
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Arterial"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Collector",
                "Secondary Arterial",
                "Sec Arterial"
            ) ~ "Collector",
            segment_type %in% c( # local equiv
                "Lane",
                "Residential",
                "Leased",
                "Recreational"
            ) ~ "Local",
            .default = segment_type
        )
    )
vanc

# Load raw data
calg_bikeways <- read_csv("../data/calgary_bikeways_2009_2022_v1.csv")
calg_roads <- read_csv("../data/calgary_roads_2009_2022_v1.csv")

# Combine raw data
calg <- calg_bikeways %>%
    select(
        SHAPE_ID,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
```

```
        UPGR2_MIN_HTYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
    left_join(
        calg_roads %>% select(
            shape_id,
            ctp_class
        ),
        by = join_by(SHAPE_ID == shape_id)
    ) %>%
    rename(
        id = SHAPE_ID,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_HTYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = ctp_class
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Arterial Street",
                "Industrial Arterial",
                "Local Arterial",
                "Parkway",
                "Urban Boulevard"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Neighbourhood Boulevard",
                "Collector",
                "Primary Collector",
                "Skeletal Road"
            ) ~ "Collector",
            segment_type %in% c( # local equiv
                "Access Route",
                "Residential Street",
                "Activity Center Street",
                "Historic Road Allowance",
                "Lanes (Alleys)",
                "Industrial Street"
            ) ~ "Local",
            .default = segment_type
        )
    )
calg

# Load raw data
toron_bikeways <- read_csv("../data/toronto_bikeways_2009_2022_v1.csv")
toron_roads <- read_csv("../data/toronto_roads_2009_2022_v1.csv")
```

```r
# Combine raw data
toron <- toron_bikeways %>%
    select(
        ID_OID,
        INST_YR,
        INST_MIN_HTYPE,
        UPGR1_YR,
        UPGR1_MIN_HTYPE,
        UPGR2_YR,
        UPGR2_MIN_HTYPE,
        ATR_SEGMENT_LENGTH
    ) %>%
    left_join(
        toron_roads %>% select(
            OID_,
            FEATURE36
        ),
        by = join_by(ID_OID == OID_)
    ) %>%
    rename(
        id = ID_OID,
        install_year = INST_YR,
        install_type = INST_MIN_HTYPE,
        upgrade1_year = UPGR1_YR,
        upgrade1_type = UPGR1_MIN_HTYPE,
        upgrade2_year = UPGR2_YR,
        upgrade2_type = UPGR2_MIN_HTYPE,
        segment_len = ATR_SEGMENT_LENGTH,
        segment_type = FEATURE36
    ) %>%
    mutate(
        segment_len = segment_len / 1000,
        road_type = case_when( # create road types
            segment_type %in% c( # arterial equiv
                "Major Arterial",
                "Major Arterial Ramp",
                "Minor Arterial"
            ) ~ "Arterial",
            segment_type %in% c( # collector equiv
                "Collector"
            ) ~ "Collector",
            segment_type %in% c(  # local equiv
                "Local",
                "Other"
            ) ~ "Local",
            .default = segment_type
        )
    )
toron
plot_yearly_len_infra(list(
    "Vancouver, CA" = vanc,
    "Calgary, CA" = calg,
    "Toronto, CA" = toron
```

```
))
plot_yearly_len_road(
    vanc,
    title = "Roadways with Dedicated Cycling Infrastructure (Vancouver, CA)"
)
plot_yearly_len_road(
    calg,
    title = "Roadways with Dedicated Cycling Infrastructure (Calgary, CA)"
)
plot_yearly_len_road(
    toron,
    title = "Roadways with Dedicated Cycling Infrastructure (Toronto, CA)"
)
version
```