The Impact of the EU Carbon Border Adjustment Mechanism: Early evidence from the EU-India Steel Supply Chain–Code (1)

Gian Luca Vriz, Theodor Cojoianu, Carolyn Fischer, Luca Taschini

2025-08-08

Introduction

This analysis will examine the relationship between India and the European Union, specifically focusing on their interaction within the "*iron & steel*" sector. The list of items are available here, which are all covered by the Carbon Border Adjustment Mechanism (CBAM). The project incorporates the utilization of the following libraries.

```
library(viridis)
library(Metrics)
library(readxl)
library(ggResidpanel)
library(xtable)
library(ggdark)
library(lme4)
library(countrycode)
library(gtools)
library(hrbrthemes)
library(paletteer)
library(ggthemes)
library(forcats)
library(reshape2)
library(circlize)
library(dplyr)
library(ggforce)
library(stringr)
library(tidyr)
library(lubridate)
library(zoo)
library(data.table)
library(ggplot2)
library(randomcoloR)
library(grid)
library(gridBase)
```

Firstly the environment for the analysis is uploaded.

```
load("Data_Code1.RData")
```

Macroeconomic Data Snapshot

The upcoming chunks visualize the percentage share of item values for each HS code.

```
df_group0 <- df_i  %>%
  group_by(Monthly.Date, HS.Code) %>%
  dplyr::summarise(n = sum(Reported.Item.Value..USD.)) %>%
  mutate(percentage = n / sum(n))
```

```
## `summarise()` has grouped output by 'Monthly.Date'. You can override using the
## `.groups` argument.
```

```
#Overall share
df_group1 <- df_i  %>%
  dplyr::group_by(HS.Code) %>%
  dplyr::summarise(N = sum(Reported.Item.Value..USD.)) %>%
  mutate(Overall = N / sum(N))
df_group <- merge(df_group0, df_group1, by = 'HS.Code')
#Check (TRUE)
round(sum(df_group1$Overall)) == 1
```

```
## [1] TRUE
```

```
#Top HS codes
Best_item <-
  df_group1 %>% arrange(desc(Overall))
Best_item <- Best_item$HS.Code
df_i$HS.Code2 <- df_i$HS.Code
df_i[!df_i$HS.Code2 %in% Best_item[1:30], 'HS.Code2'] <- 'Other codes'
#Removing
remove(df_group)
```

```
#Colors
cols <- c("gray30", paletteer_c("grDevices::rainbow", 30))
names(cols) <- c("Other codes", Best_item[1:30])
df_group0 <- df_i  %>%
  group_by(Monthly.Date, HS.Code2) %>%
  dplyr::summarise(n = sum(Reported.Item.Value..USD.)) %>%
  mutate(percentage = n / sum(n))
```

```
## `summarise()` has grouped output by 'Monthly.Date'. You can override using the
## `.groups` argument.
```

```
#Overall share
df_group1 <- df_i  %>%
  dplyr::group_by(HS.Code2) %>%
  dplyr::summarise(N = sum(Reported.Item.Value..USD.)) %>%
  mutate(Overall = N / sum(N))
df_group <- merge(df_group0, df_group1, by = 'HS.Code2')
#Check (TRUE)
round(sum(df_group1$Overall)) == 1
```

```
## [1] TRUE
```

```
#Plot by top 30 HS codes
df_group %>% mutate(HS.Code2 = fct_reorder(HS.Code2, desc(Overall))) %>%
  ggplot(aes(
    x = Monthly.Date,
    y = n,
    fill = HS.Code2,
    colour = HS.Code2
  )) + geom_area(alpha = 0.8, position = "fill") +
  scale_y_continuous(labels = scales::percent) + ylab('Share') +
  xlab('Time') + ggtitle('USD Value of Iron and Steel Items Traded with EU by HS Code (%)') + scale_colo
    values = cols,
    aesthetics = c("colour", "fill"),
    name = "Top 30 HS codes"
  ) + facet_zoom(xlim = c(as.Date('2022-12-01'), as.Date('2024-04-01'))) + theme(
    axis.text = element_text(size = 7),
    legend.text = element_text(size = 7),
    legend.key.size = unit(0.7, 'cm')
  )
```

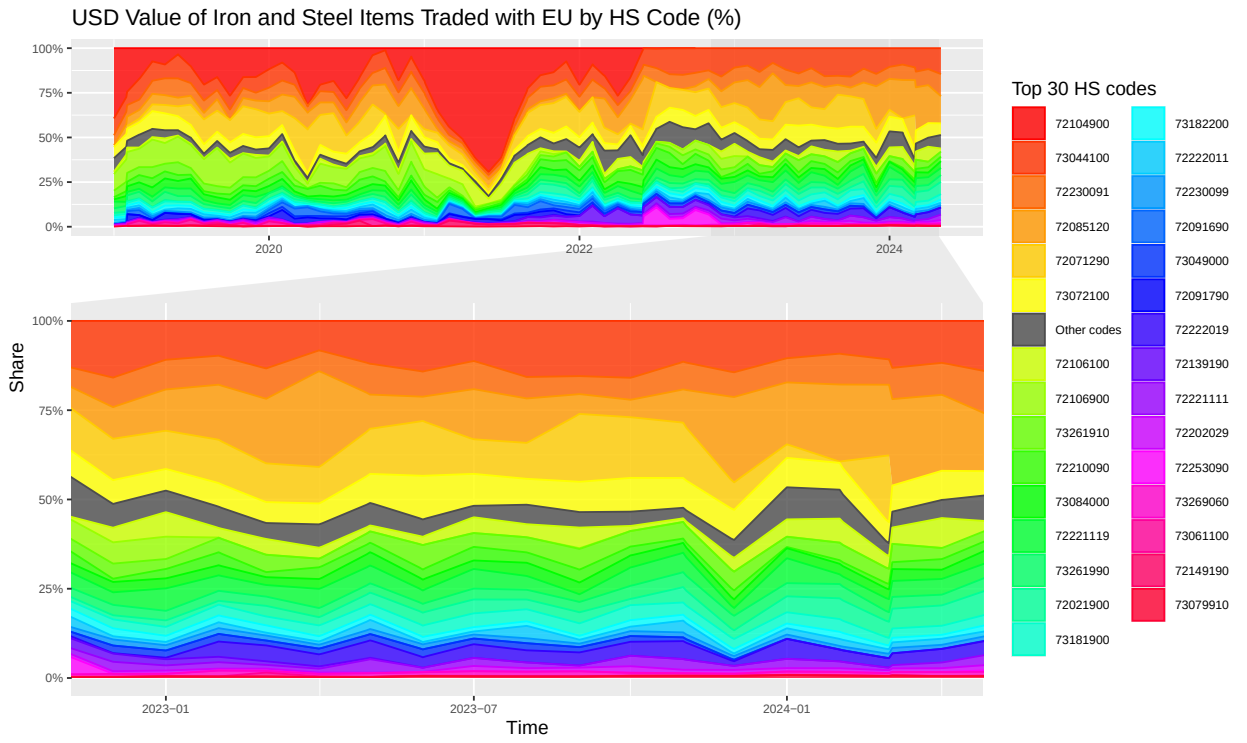

Figure 1: Monthly aggregation of the value (USD) of CBAM-covered iron and steel items traded from India to the EU, categorized by HS code and ordered in descending share (%).

```
#Removing
remove(df_group)
```

By the analysis, the top 10 HS codes have a cumulative market share of 72%.

```r
df_group_avg <- df_i  %>%
  group_by(HS.Code2) %>%
  dplyr::summarise(n = sum(Reported.Item.Value..USD.)) %>%
  mutate(percentage = n / sum(n))
#Market share top 10 HS codes
df_group_avg <-
  df_group_avg[df_group_avg$HS.Code2 %in% Best_item[c(1:10)], ]  #30 for more than 96%
sum(df_group_avg$percentage)
```

```
## [1] 0.7157316
```

```r
#Removing
remove(df_group_avg)
```

As a final macro-level descriptive statistics, embedded CO2e emissions can be calculated using the default values provided by the European Commission. To do this, the Panjiva data should be restricted to items with quantities listed in KGS or MTS.

```r
#Data on KGS and MTS only
df_im <- merge(df_i, HS_Codes_tot, by = 'HS.Code')
df_i_kg <- df_im[df_im$Item.Quantity.Unit == 'KGS', ]
df_i_mts <- df_im[df_im$Item.Quantity.Unit == 'MTS', ]
#Total Co2e emission
df_i_kg$Co2 <- df_i_kg$Co2_tot * (df_i_kg$Item.Quantity / 1000)
df_i_mts$Co2 <- df_i_mts$Co2_tot * df_i_mts$Item.Quantity
df_ikm <- rbind(df_i_kg, df_i_mts)
```

```r
#2019-2024
#Data pre-processing
df_group <- df_ikm %>%
  group_by(Shipment.Destination, HS.Code2) %>%
  dplyr::summarise(Co2 = sum(Co2))
```

```
## `summarise()` has grouped output by 'Shipment.Destination'. You can override
## using the `.groups` argument.
```

```r
df_group <- acast(df_group, HS.Code2 ~ Shipment.Destination)
```

```
## Using Co2 as value column: use value.var to override.
```

```r
df_group[is.na(df_group)] <- 0
#Parameter-setting
circos.clear()
circos.par(gap.after = c(rep(4, length(rownames(
  df_group
)) - 1), 10, rep(4, length(colnames(
  df_group
)) - 1), 10),
start.degree = 0)
grid.col <-
```

```
  setNames(c(c(
    paletteer_c("grDevices::rainbow", length(Best_item[1:30])), rep("grey30", 1)
  ), rep('grey', length(
    c('Unknown', levels(df_i$Shipment.Destination))
  ))), c(c(Best_item[1:30], "Other codes"), c(
    'Unknown', levels(df_i$Shipment.Destination)
  )))
par(cex = 0.8, mar = c(0, 0, 0, 0))
#Chord diagram
chordDiagram(
  df_group,
  annotationTrack = "grid",
  preAllocateTracks = 1,
  grid.col = grid.col,
  reduce = 0
)
circos.trackPlotRegion(
  track.index = 1,
  panel.fun = function(x, y) {
    xlim = get.cell.meta.data("xlim")
    ylim = get.cell.meta.data("ylim")
    sector.name = get.cell.meta.data("sector.index")
    circos.text(
      mean(xlim),
      ylim[1] + .1,
      sector.name,
      facing = "clockwise",
      niceFacing = TRUE,
      adj = c(0, 0)
    )
  },
  bg.border = NA
) + title('Total Co2e emission by HS code and state, 2019-2024', line = -1)
```

```
## integer(0)
```

The tonnes of CO2e emission for each country during the period between 2019 and 2024 by the India iron and steel sector are listed below.

```
sort(colSums(df_group))
```

```
##          Malta     Luxembourg        Estonia        Hungary         Cyprus
##       207.7137      1034.1806      5620.9354      6138.0991     10543.6880
##        Croatia         Latvia      Lithuania        Austria       Bulgaria
##     13312.3774     13796.5341     16990.6786     17641.1475     30379.9007
##       Slovakia Czech Republic        Ireland        Finland       Slovenia
##     32917.2759     35313.8161     50637.9173     52008.8044     57037.8876
##         Sweden         Greece        Romania         France       Portugal
##     74662.2795    205064.3720    279859.7808    309058.3232    506846.9652
##        Denmark    Netherlands        Germany         Poland          Spain
##    545785.0024    789380.6069    800220.6730   1029541.3510   1894505.3775
##          Italy        Belgium
##   5498928.0006   5954387.3869
```

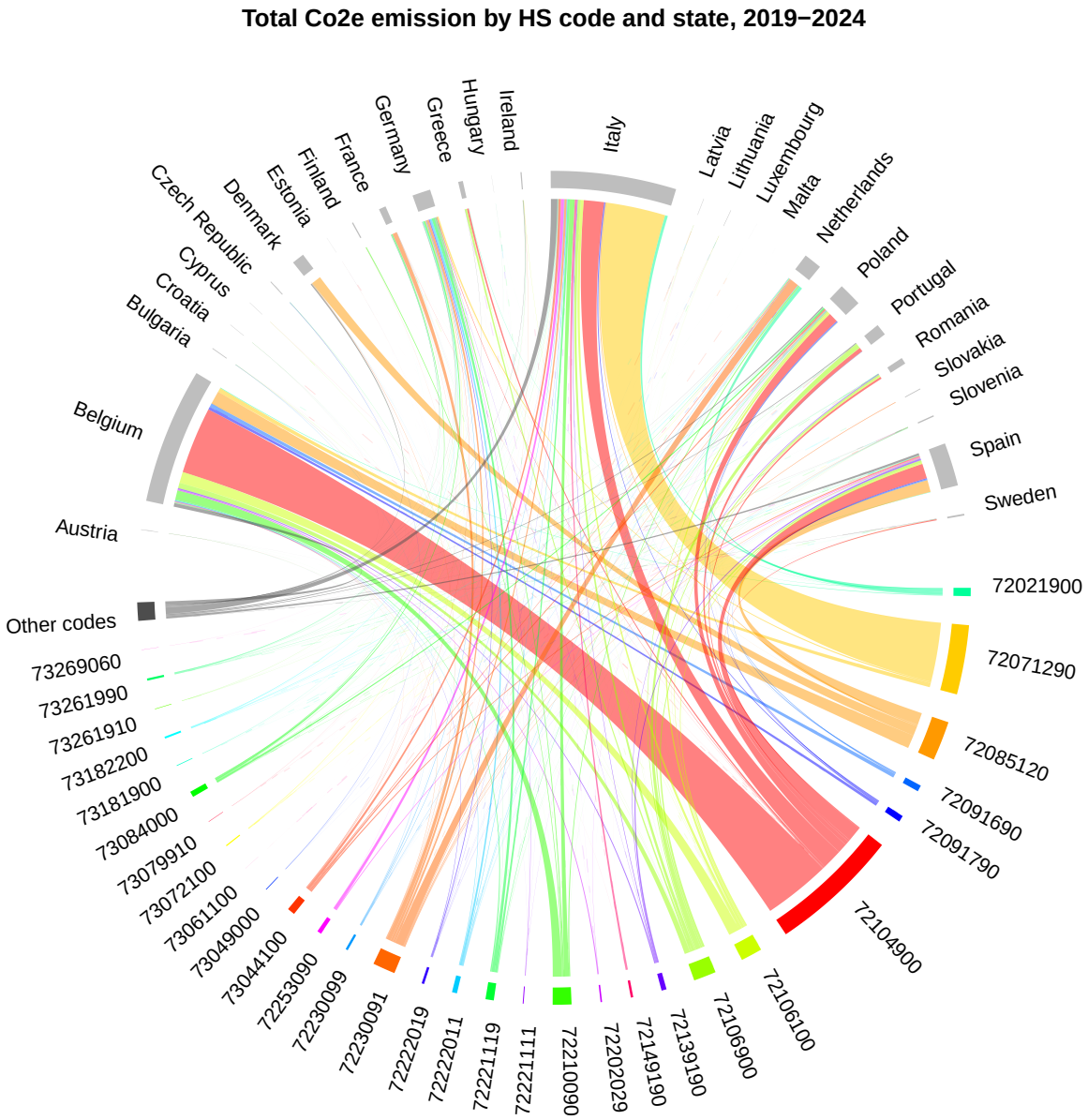

Figure 2: Traded CO2e emissions from the Indian iron and steel sector to the EU.

The dataset contains the following HS codes, which were used in the analysis along with their respective total, direct and indirect default values, as provided by the European Commission.

```
HS_Codes_tot <- HS_Codes_tot[order(HS_Codes_tot$HS.Code), ]
rownames(HS_Codes_tot) <- NULL
xtable(HS_Codes_tot)
```

```
## % latex table generated in R 4.4.1 by xtable 1.8-4 package
## % Wed Aug 13 21:45:26 2025
## \begin{table}[ht]
## \centering
## \begin{tabular}{rlrrr}
##   \hline
##  & HS.Code & Co2\_dir & Co2\_indir & Co2\_tot \\
##   \hline
## 1 & 72012000 & 1.90 & 0.17 & 2.07 \\
##   2 & 72015010 & 1.90 & 0.17 & 2.07 \\
##   3 & 72015090 & 1.90 & 0.17 & 2.07 \\
##   4 & 72021900 & 1.44 & 2.08 & 3.51 \\
##   5 & 72026000 & 3.48 & 2.81 & 6.26 \\
##   6 & 72031000 & 4.81 & 0.00 & 4.81 \\
##   7 & 72039000 & 4.81 & 0.00 & 4.81 \\
##   8 & 72052100 & 1.90 & 0.17 & 2.07 \\
##   9 & 72071190 & 2.65 & 0.62 & 3.27 \\
##   10 & 72071290 & 2.65 & 0.62 & 3.27 \\
##   11 & 72081000 & 2.01 & 0.27 & 2.28 \\
##   12 & 72085120 & 2.01 & 0.27 & 2.28 \\
##   13 & 72085210 & 2.01 & 0.27 & 2.28 \\
##   14 & 72085310 & 2.01 & 0.27 & 2.28 \\
##   15 & 72085390 & 2.01 & 0.27 & 2.28 \\
##   16 & 72091690 & 2.03 & 0.36 & 2.39 \\
##   17 & 72091710 & 2.03 & 0.36 & 2.39 \\
##   18 & 72091790 & 2.03 & 0.36 & 2.39 \\
##   19 & 72092690 & 2.03 & 0.36 & 2.39 \\
##   20 & 72092790 & 2.03 & 0.36 & 2.39 \\
##   21 & 72092890 & 2.03 & 0.36 & 2.39 \\
##   22 & 72102000 & 1.97 & 0.39 & 2.35 \\
##   23 & 72104100 & 1.97 & 0.39 & 2.35 \\
##   24 & 72104900 & 1.97 & 0.39 & 2.35 \\
##   25 & 72105000 & 1.97 & 0.39 & 2.35 \\
##   26 & 72106100 & 1.97 & 0.39 & 2.35 \\
##   27 & 72106900 & 1.97 & 0.39 & 2.35 \\
##   28 & 72111300 & 2.01 & 0.27 & 2.28 \\
##   29 & 72112320 & 2.03 & 0.36 & 2.39 \\
##   30 & 72112330 & 2.03 & 0.36 & 2.39 \\
##   31 & 72121010 & 1.97 & 0.39 & 2.35 \\
##   32 & 72121090 & 1.97 & 0.39 & 2.35 \\
##   33 & 72125030 & 1.97 & 0.39 & 2.35 \\
##   34 & 72125040 & 1.97 & 0.39 & 2.35 \\
##   35 & 72125090 & 1.97 & 0.39 & 2.35 \\
##   36 & 72126000 & 1.97 & 0.39 & 2.35 \\
##   37 & 72139110 & 1.89 & 0.32 & 2.21 \\
##   38 & 72139190 & 1.89 & 0.32 & 2.21 \\
##   39 & 72139910 & 1.89 & 0.32 & 2.21 \\
```

```
##   40 & 72139990 & 1.89 & 0.32 & 2.21 \\
##   41 & 72143000 & 1.89 & 0.32 & 2.21 \\
##   42 & 72149110 & 1.89 & 0.32 & 2.21 \\
##   43 & 72149190 & 1.89 & 0.32 & 2.21 \\
##   44 & 72151000 & 1.89 & 0.32 & 2.21 \\
##   45 & 72161000 & 1.89 & 0.32 & 2.21 \\
##   46 & 72162100 & 1.89 & 0.32 & 2.21 \\
##   47 & 72162200 & 1.89 & 0.32 & 2.21 \\
##   48 & 72166900 & 1.89 & 0.32 & 2.21 \\
##   49 & 72171010 & 1.88 & 0.49 & 2.37 \\
##   50 & 72172010 & 1.95 & 0.51 & 2.46 \\
##   51 & 72172030 & 1.95 & 0.51 & 2.46 \\
##   52 & 72181000 & 2.51 & 2.10 & 4.61 \\
##   53 & 72192190 & 2.18 & 1.90 & 4.08 \\
##   54 & 72193210 & 2.21 & 1.99 & 4.19 \\
##   55 & 72193290 & 2.21 & 1.99 & 4.19 \\
##   56 & 72193310 & 2.21 & 1.99 & 4.19 \\
##   57 & 72193390 & 2.21 & 1.99 & 4.19 \\
##   58 & 72193410 & 2.21 & 1.99 & 4.19 \\
##   59 & 72193490 & 2.21 & 1.99 & 4.19 \\
##   60 & 72193510 & 2.21 & 1.99 & 4.19 \\
##   61 & 72193590 & 2.21 & 1.99 & 4.19 \\
##   62 & 72202021 & 2.21 & 1.99 & 4.19 \\
##   63 & 72202029 & 2.21 & 1.99 & 4.19 \\
##   64 & 72210090 & 2.14 & 2.17 & 4.30 \\
##   65 & 72221111 & 2.14 & 2.17 & 4.30 \\
##   66 & 72221119 & 2.14 & 2.17 & 4.30 \\
##   67 & 72222011 & 2.14 & 2.17 & 4.30 \\
##   68 & 72222019 & 2.14 & 2.17 & 4.30 \\
##   69 & 72224010 & 2.14 & 2.17 & 4.30 \\
##   70 & 72230091 & 2.13 & 2.36 & 4.49 \\
##   71 & 72230099 & 2.13 & 2.36 & 4.49 \\
##   72 & 72251100 & 1.95 & 0.40 & 2.35 \\
##   73 & 72251910 & 1.95 & 0.40 & 2.35 \\
##   74 & 72251990 & 1.98 & 0.49 & 2.46 \\
##   75 & 72253090 & 1.95 & 0.40 & 2.35 \\
##   76 & 72254012 & 1.95 & 0.40 & 2.35 \\
##   77 & 72259200 & 1.92 & 0.51 & 2.43 \\
##   78 & 72259900 & 1.92 & 0.51 & 2.43 \\
##   79 & 72261100 & 1.95 & 0.40 & 2.35 \\
##   80 & 72269910 & 1.92 & 0.51 & 2.43 \\
##   81 & 72269930 & 1.92 & 0.51 & 2.43 \\
##   82 & 72272000 & 1.86 & 0.57 & 2.43 \\
##   83 & 72279010 & 1.86 & 0.57 & 2.43 \\
##   84 & 72279050 & 1.86 & 0.57 & 2.43 \\
##   85 & 72281090 & 1.86 & 0.57 & 2.43 \\
##   86 & 72292000 & 1.84 & 0.75 & 2.59 \\
##   87 & 72299090 & 1.84 & 0.75 & 2.59 \\
##   88 & 73011000 & 2.03 & 0.36 & 2.39 \\
##   89 & 73021010 & 1.93 & 0.29 & 2.21 \\
##   90 & 73021090 & 1.93 & 0.29 & 2.21 \\
##   91 & 73023000 & 1.93 & 0.29 & 2.21 \\
##   92 & 73024000 & 1.93 & 0.29 & 2.21 \\
##   93 & 73030010 & 2.21 & 0.35 & 2.56 \\
```

```
##   94 & 73030090 & 2.21 & 0.35 & 2.56 \\
##   95 & 73041910 & 1.93 & 0.29 & 2.21 \\
##   96 & 73041990 & 1.93 & 0.29 & 2.21 \\
##   97 & 73042200 & 1.86 & 0.35 & 2.20 \\
##   98 & 73042400 & 1.86 & 0.35 & 2.20 \\
##   99 & 73042910 & 1.93 & 0.29 & 2.21 \\
##   100 & 73042990 & 1.93 & 0.29 & 2.21 \\
##   101 & 73044100 & 1.86 & 0.35 & 2.20 \\
##   102 & 73045110 & 1.86 & 0.35 & 2.20 \\
##   103 & 73045930 & 1.86 & 0.35 & 2.20 \\
##   104 & 73049000 & 1.93 & 0.29 & 2.21 \\
##   105 & 73061100 & 1.98 & 0.46 & 2.44 \\
##   106 & 73062100 & 1.98 & 0.46 & 2.44 \\
##   107 & 73071110 & 2.54 & 0.57 & 3.11 \\
##   108 & 73071190 & 2.54 & 0.57 & 3.11 \\
##   109 & 73072100 & 1.87 & 0.43 & 2.30 \\
##   110 & 73079210 & 1.93 & 0.29 & 2.21 \\
##   111 & 73079290 & 1.93 & 0.29 & 2.21 \\
##   112 & 73079910 & 1.93 & 0.29 & 2.21 \\
##   113 & 73081000 & 2.46 & 2.55 & 5.01 \\
##   114 & 73083000 & 2.46 & 2.55 & 5.01 \\
##   115 & 73084000 & 2.46 & 2.55 & 5.01 \\
##   116 & 73090010 & 1.97 & 0.39 & 2.35 \\
##   117 & 73090030 & 1.97 & 0.39 & 2.35 \\
##   118 & 73090090 & 1.97 & 0.39 & 2.35 \\
##   119 & 73102910 & 1.97 & 0.39 & 2.35 \\
##   120 & 73102990 & 1.97 & 0.39 & 2.35 \\
##   121 & 73110030 & 1.89 & 0.32 & 2.21 \\
##   122 & 73181300 & 1.89 & 0.32 & 2.21 \\
##   123 & 73181900 & 1.89 & 0.32 & 2.21 \\
##   124 & 73182100 & 1.89 & 0.32 & 2.21 \\
##   125 & 73182200 & 1.89 & 0.32 & 2.21 \\
##   126 & 73182300 & 1.89 & 0.32 & 2.21 \\
##   127 & 73182400 & 1.89 & 0.32 & 2.21 \\
##   128 & 73261100 & 2.65 & 0.62 & 3.27 \\
##   129 & 73261910 & 2.65 & 0.62 & 3.27 \\
##   130 & 73261990 & 2.65 & 0.62 & 3.27 \\
##   131 & 73269030 & 1.89 & 0.32 & 2.21 \\
##   132 & 73269040 & 1.89 & 0.32 & 2.21 \\
##   133 & 73269050 & 1.89 & 0.32 & 2.21 \\
##   134 & 73269060 & 1.89 & 0.32 & 2.21 \\
##    \hline
## \end{tabular}
## \end{table}
```

```
knitr::knit_exit()
```