Islas Canarias
Del 15 al 19 de noviembre de 2021

Junta de Andalucía
Consejería de Transformación Económica, Industria, Conocimiento y Universidades
Instituto de Estadística y Cartografía de Andalucía

INFORMES AUTOMATIZADOS CON R. EJEMPLOS DE USO EN EL SEGUIMIENTO DE LA PANDEMIA

María Escudero Tena
Instituto de Estadística y Cartografía de Andalucía
maria.escudero.tena@juntadeandalucia.es

Cristina Fernández Álvaro
Instituto de Estadística y Cartografía de Andalucía
cristina.fernandez@juntadeandalucia.es

Introducción

La situación excepcional que generó la pandemia causada por el COVID-19 provocó la necesidad de monitorizar distintas fuentes de información con datos de alta frecuencia y de gran volumen, con el objetivo de analizar diariamente la evolución de la pandemia.

Esta monitorización se estructuró a través de una serie de informes de seguimiento automatizados, elaborados con Rmarkdown, los cuales fueron generados con frecuencia diaria o semanal, según su tipología.

La razón principal para la elección de Rmarkdown como vía de difusión de la información fue su versatilidad a la hora de crear documentos que incorporan por un lado texto fijo con explicaciones genéricas, y por otro lado texto o resultados dinámicos que se actualizan con cada generación del informe.

Las principales fuentes utilizadas en la elaboración de estos informes están relacionadas con indicadores sanitarios e indicadores de movilidad. Por un lado se desarrolló, en colaboración con la Consejería de Salud y Familias, unos informes sanitarios diarios que permitieran el seguimiento de los contagios, casos sospechosos y la incidencia en Andalucía, a partir de los datos recabados en la base de datos RedAlerta que recoge información diaria e individualizada sobre todos los casos sospechosos y confirmados de COVID-19 que se identifican en Andalucía. Por otro lado se elaboraron informes diarios de seguimiento geoestadístico de la COVID-19 a partir de datos procedentes de la huella digital de los teléfonos móviles. En este caso se trabajó con dos fuentes de datos: los

procedentes de la operación experimental sobre movilidad del INE y los datos abiertos ofrecidos por el Ministerio de Transportes, Movilidad y Agenda Urbana.

Esta ponencia pretende mostrar, a modo de ejemplo, algunos de los informes generados con resultados dinámicos a partir de las fuentes anteriores, tanto en formato HTML como Word, con indicación de las librerías y funciones empleadas en cada caso.

Objetivos

El objetivo de estos informes es monitorizar distintas fuentes de información con datos de alta frecuencia y de gran volumen para analizar diariamente la evolución de la pandemia en Andalucía. Las características deseables de estos informes son las siguientes:

- Que estén **automatizados**, de cara a que se puedan ejecutar diariamente para actualizar los resultados con la incorporación de nuevos datos
- Que incorporen texto fijo con explicaciones genéricas y **resultados dinámicos** que se actualicen con cada generación del informe
- Que sean **flexibles** en la generación de diversos tipos de tablas, representaciones gráficas y mapas
- Que permitan trabajar con un gran **volumen de datos**, los cuales se encuentran en diferentes formatos
- Que la implantación sea rápida y **sencilla**
- Que permitan la **transparencia** en los análisis realizados.
- Que los análisis realizados puedan ser **replicados**.

Metodología y resultados

A) Informes sanitarios

Los primeros informes que se llevaron a cabo se iniciaron en marzo de 2020. Estos informes se desarrollaron con Rmarkdown para el seguimiento de los contagios, casos sospechosos y la incidencia en Andalucía a partir de la base de datos RedAlerta, con información diaria e individualizada sobre todos los casos sospechosos y confirmados de COVID-19 que se identifican en Andalucía.

Los análisis realizados se plasmaron en diversos informes independientes que variaban en función de la temática, del destinatario del informe o de la frecuencia de generación de la información:

- Según temática. Se definieron, por ejemplo, informes enfocados en el impacto de la pandemia sobre el personal sanitario y sociosanitario, otro sobre características del conjunto de la población contagiada así como otros específicos sobre la evolución más reciente de la pandemia.

- Según el destinatario del informe. Se distribuyeron distintos modelos de informes, a los profesionales de la Red de Vigilancia Epidemiológica de Andalucía se les presentó por ejemplo el más completo mientras que otro más resumido se dirigía al Consejo de Gobierno

- Según la frecuencia de generación. Las necesidades de monitorización de las variables derivaron en la generación de informes de frecuencia diaria o semanal.

Procedencia de los datos

Desde la aplicación de RedAlerta, los datos se exportaban diariamente a Excel y posteriormente desde R se importaban con la *librería readxl* de *tidyverse*.

La limitación al uso de Excel venía impuesta por la propia aplicación de RedAlerta que sólo permitía la exportación en formato "xls". De hecho, esto generó problemas cuando el número de casos creció, debido a las restricciones de Excel respecto al número de registros, obligando a partir el fichero de entrada conforme aumentaba el número de personas contagiadas.

Quedó planteado, como aspecto de mejora, realizar una conexión directamente a la base de datos de RedAlerta o bien la posibilidad de exportación de los datos a otro tipo de fichero como ".csv" o ".dbf", pero el hecho de ser RedAlerta un modelo bastante cerrado no permitía, al menos desde la posición en la que nos encontrábamos nosotros, encontrar una solución fácil a este inconveniente.

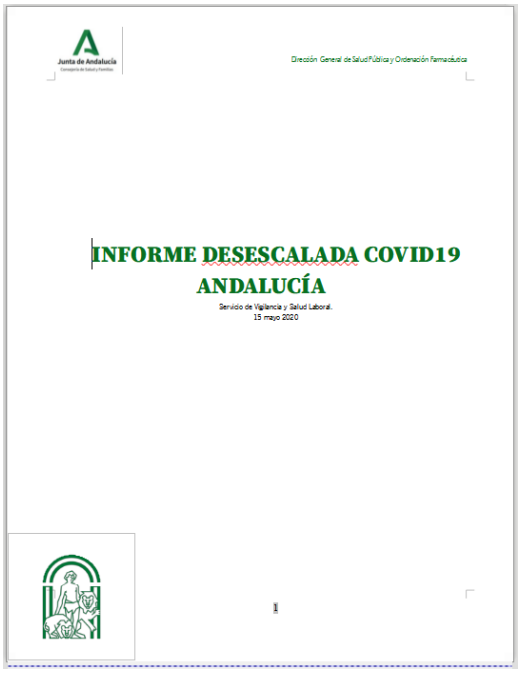
Visualización final

De cara a la visualización final de la información se valoró la opción de utilizar *Rshiny* de forma que permitiese a los usuarios interactuar con los datos, sin embargo la necesidad de que estos informes se implantasen con mucha celeridad y la no disponibilidad de un servidor R hizo descartar esta posibilidad.

Se optó por un formato de salida tipo docx. (Word), con el objeto de que el documento fuera completamente editable a posteriori por parte del Servicio de Vigilancia Epidemiológica. Para controlar el formato del archivo resultante se partía de una plantilla ".dotx" que definía entre otros aspectos los tipos de letras o los logos corporativos.

Figura 1: Fragmento de script con la definición del fichero salida (izquierda) junto con captura de pantalla del fichero salida (derecha)

```
---
title: "INFORME DESESCALADA COVID19 ANDALUCÍA"
author: "Servicio de Vigilancia y Salud Laboral."
date: "`r format(Sys.time(), '%d %B %Y')`"
output:
  word_document:
    reference_docx: INFORME_MODELO3.dotx
always_allow_html: true
---
```



Principales scripts de los informes sanitarios

A continuación se describen, a modo de ejemplo, algunos fragmentos de los scripts utilizados en los informes sanitarios, con indicación de las librerías y funciones empleadas, así como algunas capturas de pantalla de los resultados obtenidos.

Generación de ficheros

Para centralizar la generación de todos los informes sanitarios, se generó un script desde el cuál se lanzaban todos los ficheros de tipo ".Rmd" mediante la función *render* de la librería *rmarkdown*, con indicación de la ruta en la que se guardarían los informes así como de la ruta en la que se encontraba el fichero de tipo ".Rmd". Los nombres de los ficheros resultantes concatenan la fecha en las que se generan, de forma que no se sobreescribieran con los lanzados anteriormente y se mantuviera registro de todos los ficheros generados.

Figura 2: Fragmento de script con la generación de los informes Rmarkdown

```
library(rmarkdown)

################################################################################################################
#Informe semanal
################################################################################################################

# Ruta donde se guardará el informe
salida<<-"C:/Users/Nestor/Desktop/Salud/covid/Propuesta Informe R/Informes/"
# Ruta en la que se encuentra guardado el script
entrada<<-"C:/Users/Nestor/Desktop/Salud/covid/Propuesta Informe R/"

output_dir=paste(salida,gsub(":","",gsub(" ", "_", format(Sys.time(),'%Y %m %d'))),"INFORME_SEMANAL.doc",sep='')
rmarkdown::render(paste0(entrada,"INFORME_CORONAVIRUS_v12.Rmd"), "word_document",
                  encoding="UTF-8",runtime = "static",output_dir)

################################################################################################################
#Informe petición, incidencias acumuladas 14 días
################################################################################################################

# Ruta donde se guardará el informe
salida<<-"C:/Users/Nestor/Desktop/Salud/covid/Propuesta Informe R/Informes/"
# Ruta en la que se encuentra guardado el script
entrada<<-"C:/Users/Nestor/Desktop/Salud/covid/Propuesta Informe R/"

output_dir=paste(salida,gsub(":","",gsub(" ", "_", format(Sys.time(),'%Y %m %d'))),"INFORME_PETICION_TASAS.doc",sep='')
rmarkdown::render(paste0(entrada,"Peticion_v3.Rmd"), "word_document",
                  encoding="UTF-8",runtime = "static",output_dir)

################################################################################################################
#Informe petición, profesional sanitario PCR
################################################################################################################

# Ruta donde se guardará el informe
salida<<-"C:/Users/Nestor/Desktop/Salud/covid/Propuesta Informe R/Informes/"
```

Texto integrado

Con el objetivo de mostrar resultados dinámicos, en los textos de los informes se incorporaron sentencias en R con la *librería dplyr*, que actualizaban la parte numérica del texto en cada generación, a partir de los datos de entrada.

Para ello se hizo uso de sentencias tipo *`r orden_de_R`* entre los literales estáticos del texto

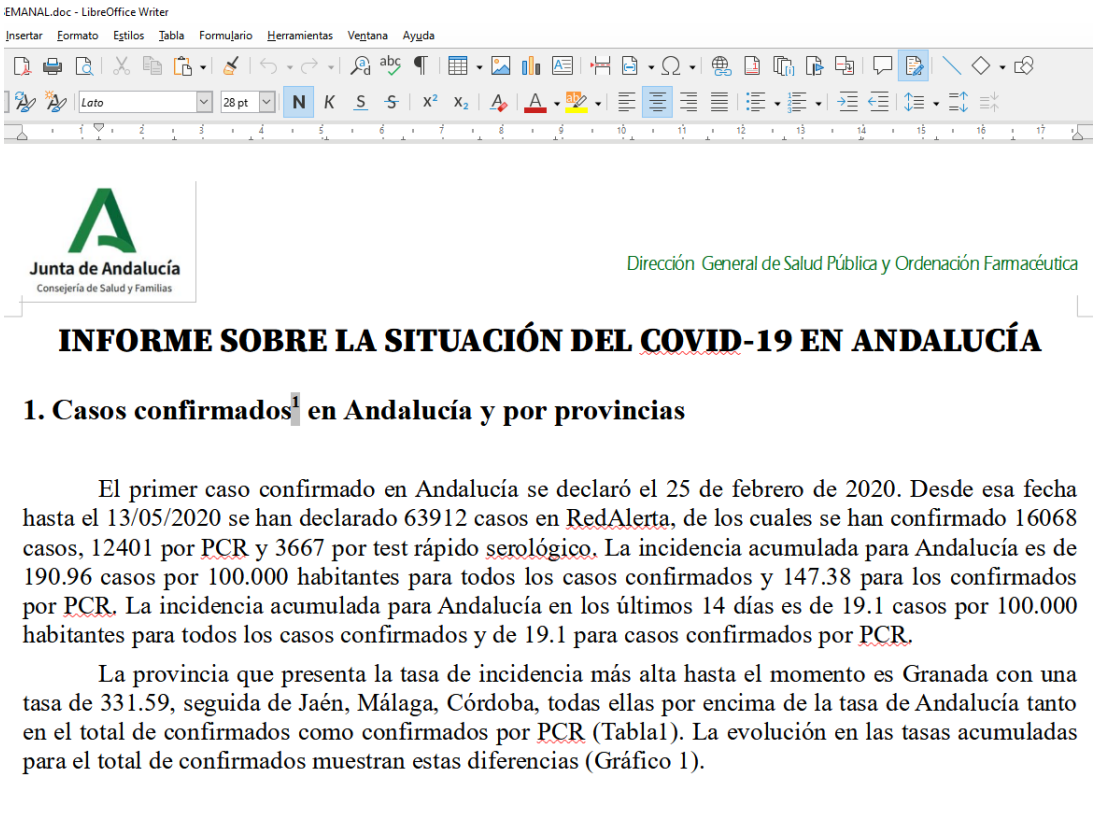
Figura 3: Fragmento de código con ejemplo de texto que incluye información dinámica (arriba) junto con captura de pantalla del fichero salida (abajo)

```
### 1. Casos confirmados[^1] en Andalucía y por provincias

<br/>

El primer caso confirmado en Andalucía se declaró el 25 de febrero de 2020. Desde esa fecha hasta el `r format(fecha_datos,format="%d/%m/%Y")` se han declarado `r datosbrutos %>% nrow()`  casos en RedAlerta, de los cuales se han confirmado `r coronavirus_confirmados %>% nrow()` casos, `r coronavirus_confirmados %>% filter(Clasificación_de_caso_COVID_19 == "Confirmado PCR") %>% nrow()` por PCR y `r coronavirus_confirmados %>% filter(Clasificación_de_caso_COVID_19 == "Confirmado serología (ELISA, QLIA, EQLIA)"| Clasificación_de_caso_COVID_19 == "Confirmado test rápido") %>% nrow()` por test rápido serológico. La incidencia acumulada para Andalucía es de `r round((coronavirus_confirmados %>% nrow())/(poblacion_provincias %>% filter(Numero == "0") %>% pull())*100000,digits = 2)` casos por 100.000 habitantes para todos los casos confirmados y `r round((coronavirus_confirmados %>% filter(Clasificación_de_caso_COVID_19 == "Confirmado PCR")%>% nrow())/(poblacion_provincias %>% filter(Numero == "0") %>% pull())*100000,digits = 2)` para los confirmados por PCR. La incidencia acumulada para Andalucía en los últimos 14 días es de `r round((coronavirus_confirmados %>%filter(fecha_datos - Fecha_Declaración_ddmmaaaa < 14)%>%nrow())/(poblacion_provincias %>% filter(Numero == "0") %>% pull())*100000,digits = 2)` casos por 100.000 habitantes para todos los casos confirmados y de `r round((coronavirus_confirmados %>%filter(fecha_datos - Fecha_Declaración_ddmmaaaa < 14)%>%nrow())/(poblacion_provincias %>% filter(Numero == "0") %>% pull())*100000,digits = 2)` para casos confirmados por PCR.

La provincia que presenta la tasa de incidencia más alta hasta el momento es `r incidencia_provincias$Provincia[1]` con una tasa de `r round(incidencia_provincias$Incidencia_acumulada[1],digits=2)`, seguida de `r incidencia_provincias$Provincia[2]`, `r incidencia_provincias$Provincia[3]`, `r incidencia_provincias$Provincia[4]`, todas ellas por encima de la tasa de Andalucía tanto en el total de confirmados como confirmados por PCR (Tabla1). La evolución en las tasas acumuladas para el total de confirmados muestran estas diferencias (Gráfico 1).
```



Generación de tablas

La librería más extendida para dar formato a las tablas generadas con Rmarkdown es *kableExtra*, sin embargo está enfocada exclusivamente a los documentos de tipo PDF y HTML. Para los documentos de tipo Word, una buena alternativa es la *librería flextable*, que permite generar tablas con buena presentación, pensadas para que se inserten directamente en documentos HTML o Word. Entre otros aspectos, *flextable* permite definir el formato de las columnas del dataframe de entrada, el tamaño de fuente, los nombres de las variables a mostrar, la alineación, las celdas que van en negrita o el título de la tabla.

Figura 4: Fragmento de script con ejemplo de tabla generada con la librería flextable (arriba) junto con captura de pantalla del fichero salida (abajo)

```
717 ```{r ft.align="left"}
718 #Pintamos la tabla
719 total %>%
720   select(Provincia,Confirmados,hosp2,Tasa_hosp,UCI2,Defunciones2) %>%
721 flextable() %>%
722 colformat_num(j=3:6,big.mark="",digits = 1,na_str = 0) %>%
723 autofit() %>%
724 fontsize(size=9,part="all") %>%
725   align(align="right",part="all") %>%
726 bold(bold = TRUE, part = "header") %>%
727 bold(i=9,j=1:6,bold = TRUE, part = "body") %>%
728 set_header_labels("hosp2"="Hospitalizados (%)","Tasa_hosp"="Tasa de hospitalización","UCI2"="UCI
    (%)","Defunciones2"="Defunciones (%)") %>%
729 set_caption("Tabla 2. Casos confirmados por situación clínica", html_escape = TRUE)
730
731 ```
```

Tabla 2. Casos confirmados por situación clínica

Provincia	Confirmados	Hospitalizados (%)	Tasa de hospi-talización	UCI (%)	Defunciones (%)
Almería	695	218 (31.4%)	30.4	39 (5.6%)	50 (7.2%)
Cádiz	1480	566 (38.2%)	45.6	79 (5.3%)	142 (9.6%)
Córdoba	1682	551 (32.8%)	70.0	74 (4.4%)	105 (6.2%)
Granada	3033	1184 (39%)	129.4	132 (4.4%)	278 (9.2%)
Huelva	520	219 (42.1%)	42.9	30 (5.8%)	48 (9.2%)
Jaén	1751	751 (42.9%)	118.5	85 (4.9%)	173 (9.9%)
Málaga	3983	1463 (36.7%)	88.0	167 (4.2%)	274 (6.9%)
Sevilla	2924	1194 (40.8%)	61.2	148 (5.1%)	274 (9.4%)
Total	**16068**	**6146 (38.2%)**	**73.0**	**754 (4.7%)**	**1344 (8.4%)**

Representaciones gráficas

Las representaciones gráficas más relevantes de los informes sanitarios fueron las siguientes:

Pirámides de población

Los casos totales, el número de hospitalizados, de ingresados en UCI y de fallecidos, se representaron por grupo de edad y sexo en forma de pirámide de población.

Para ello se utilizó la *librería ggplot2* para definir cada uno de los 4 gráficos, y la función *grid.arrange* de la librería gridExtra para representar los 4 gráficos en la misma imagen.

Figura 5: Fragmento de script con ejemplo de distintas pirámides de población representadas en conjunto (arriba) y captura de pantalla del fichero de salida (abajo)

```
```{r,fig.height=7, echo=FALSE}

#Tablas auxiliares para ver los límites de los ejes
aux_ejes1<-coronavirus_confirmados %>%
  group_by(grupoedad,Sexo) %>%
  summarise (C = n())

max_ejes1=max(aux_ejes1$C)+50

#Tablas auxiliares para ver los límites de los ejes
aux_ejes2<-coronavirus_confirmados %>%
  filter(Estancia_en_UCI == "Si") %>%
  group_by(grupoedad,Sexo) %>%
  summarise (C = n())

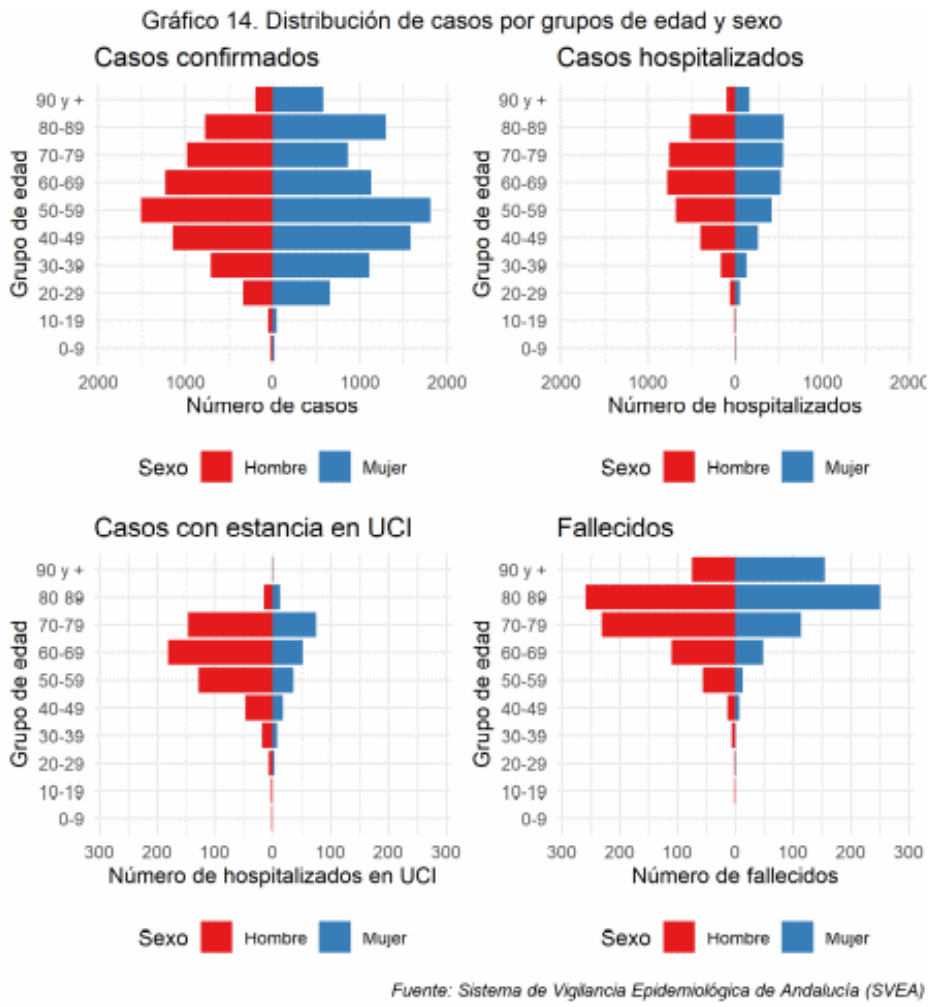
max_ejes2=max(aux_ejes2$C)+100

coronavirus_confirmados$Casos=1
## barplots for male populations goes to the left (thus negative sign)
coronavirus_confirmados$Casos <- ifelse(coronavirus_confirmados$Sexo == "Hombre",
-1*coronavirus_confirmados$Casos, coronavirus_confirmados$Casos)

A=ggplot(coronavirus_confirmados, aes(x = grupoedad, y = Casos, fill = Sexo)) +
    theme (text = element_text(size=5)) +
  geom_bar(data = subset(coronavirus_confirmados, Sexo == "Mujer"), stat = "identity") +
  geom_bar(data = subset(coronavirus_confirmados, Sexo == "Hombre"), stat = "identity") +
  coord_flip()+
  xlab("Número de casos") +
  ylab("Grupos de edad") +
  scale_y_continuous(labels = abs, limits = max_ejes1 * c(-1,1)) +
  labs(title = "Casos confirmados", x = "Grupo de edad", y = "Número de casos") +
  scale_fill_brewer(palette = "Set1") +
    theme_minimal() +
  theme(legend.position="bottom")
```

...

```
grid.arrange(
  A,
  B,
  C,
  D,
  ncol = 2,
  top="Gráfico 14. Distribución de casos por grupos de edad y sexo",
  bottom = textGrob(
    "Fuente: Sistema de Vigilancia Epidemiológica de Andalucía (SVEA)",
    gp = gpar(fontface = 3, fontsize = 9),
    hjust = 1,
    x = 1
```
```



Eje secundario

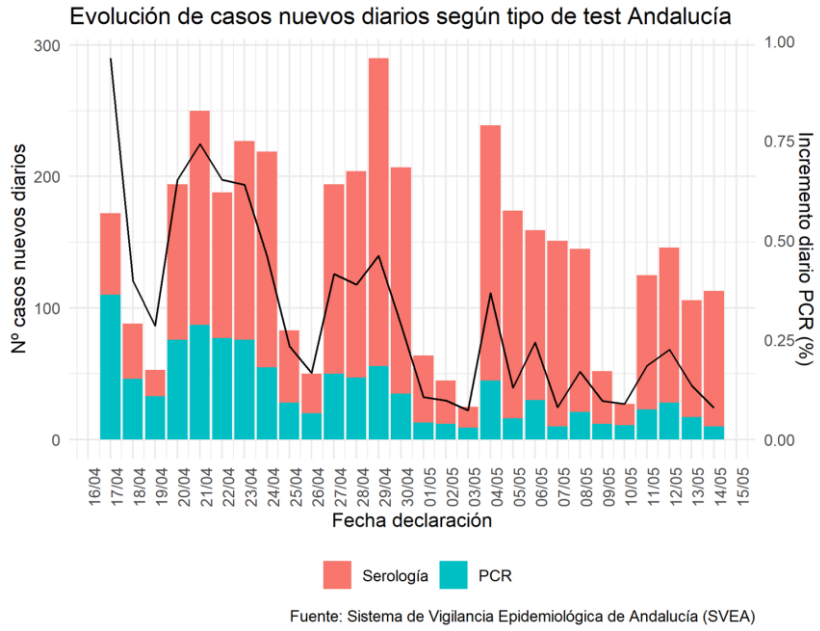
La representación de variables en dos ejes distintos suele ser complicada. Ponemos un ejemplo de barras apiladas con eje secundario generado con la *librería ggplot2*.

Figura 6: Fragmento de script de gráfico con barras apiladas y eje secundario generado con la ggplot2 (arriba) y fichero de salida (abajo)

```
#Gráfica
prov1 %>%
  ggplot() + aes(x= Fecha_Declaración_ddmmaaaa, y = value, fill = variable) %>%
  geom_bar(stat="identity") +

 geom_line(data = prov2, aes(x = Fecha_Declaración_ddmmaaaa, y = (porcentaje)*f, group = 1),
inherit.aes = FALSE) +
 scale_y_continuous(sec.axis = sec_axis(~./f, name = "Incremento diario PCR (%)"))+

  theme_minimal() +
  ggtitle("Evolución de casos nuevos diarios según tipo de test Andalucía") +
  xlab("Fecha declaración") + ylab("Nº casos nuevos diarios") +
  theme(legend.position = "bottom") +
  scale_fill_discrete(name="",labels = c("Serología", "PCR")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    scale_x_date(labels = date_format("%d/%m"),
                 breaks = date_breaks("1 days")) +
 labs(caption = "Fuente: Sistema de Vigilancia Epidemiológica de Andalucía (SVEA)")
```



Mapa de calor

Un tipo de gráfico muy utilizado en el seguimiento de la pandemia para detectar nuevos brotes fueron los mapas de calor. Mediante estos gráficos se representaba la evolución de nuevos casos con frecuencia semanal, bisemanal o diaria. Los gráficos se realizaron para diferentes desagregaciones geográficas: provincias, distritos sanitarios, zonas básicas de salud y Unidades de Gestión Clínica.

Uno de los mapas de calor que se realizó a partir de la *librería ggplot2* fue el siguiente.

Figura 7: Fragmento de script con ejemplo de mapa de calor (arriba) y fichero de salida (abajo)

```
total%>%
  mutate(Distrito=paste0(Provincia,"-",Distrito))%>%
  mutate(Distrito=fct_reorder(Distrito, desc(Distrito)))%>%
  ggplot(aes(Fecha,y=Distrito)) +
  theme (text = element_text(size=10)) +
  geom_tile(aes(fill = Casos)) +
  geom_text(aes(label =Casos),size=3) +
  scale_x_date(labels = date_format("%d/%m"),
               breaks = date_breaks("1 day"))+
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  xlab("Fecha")+ylab("Distrito de Salud")+
  scale_fill_gradient(low = "white", high = "red",na.value = NA) +
  ggtitle("Evolución de nuevos casos PCR por distritos de salud") +
  theme(legend.position = "bottom") +
  facet_wrap( ~ variable, ncol = 2, scales = 'fixed') +
  labs(caption = "Fuente: Sistema de Vigilancia Epidemiológica de Andalucía (SVEA)")
```

Estimación de la transmisión

A partir de R, también fue posible calcular el número de transmisiones o reproducciones de casos en función del tiempo de los casos confirmados por PCR mediante la función *estimate_R* de la *librería EpiEstim*, librería específica de estudios epidemiológicos. Los resultados de esas estimaciones de transmisión se mostraron en gráficos utilizando la librería ggplot2.

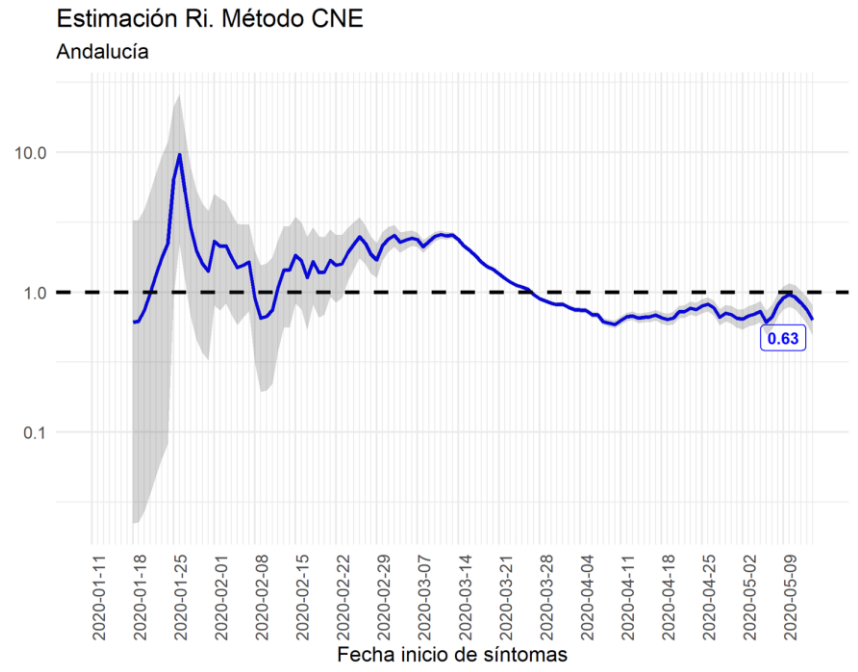
Figura 8: Fragmento de script con cálculo del número de reproducción en función del tiempo (arriba) y representación gráfica del fichero de salida (abajo)

```
###calculo de Numero reproductivo instantaneo Ri
mu <- 4 #mean in days
sigma <- 2 #standard deviation in days

i<-as.incidence(datos$casos, dates = datos$fecha)
res <-EpiEstim::estimate_R(i, method = "parametric_si", config =make_config(list(mean_si = mu,
std_si = sigma)))

temp<-data.table(res$R)
temp[,fecha:=res$dates[t_end]]

Res=merge(data.table(fecha=datos$fecha),temp,by="fecha",all.x=TRUE)
setnames(Res,names(Res),sub("(R)","",names(Res),fixed=TRUE))
Res[,c("R0","lower","upper"):=.(Median,Quantile.0.025,Quantile.0.975)]
tablas_R0<-Res[,.(fecha,R0,lower,upper)]
tablas_R0$Provincia=prov
```



B) Informes para la desescalada

La información sobre la movilidad de las personas ha sido fundamental durante la pandemia para conocer por un lado si se estaban cumpliendo las restricciones a la movilidad impuestas por el gobierno y por otro lado para determinar si las restricciones eran efectivas al compararlas con información de tipo sanitario.

La información sobre movilidad de la población a partir de la telefonía móvil comenzó a llegar al IECA 5 o 6 semanas después de la declaración del estado de alarma. En esas fechas lo que se consideró el uso más relevante para los datos fue ofrecer información que ayudara en la toma de decisiones respecto a la desescalada.

Esto hizo que los informes que se realizaron incluyeran por un lado información sanitaria, aunque a un nivel más informativo que el evaluativo comentado de los informes sanitarios y por otro lado información sobre movilidad. La frecuencia de estos informes fue diaria.

Procedencia de los datos

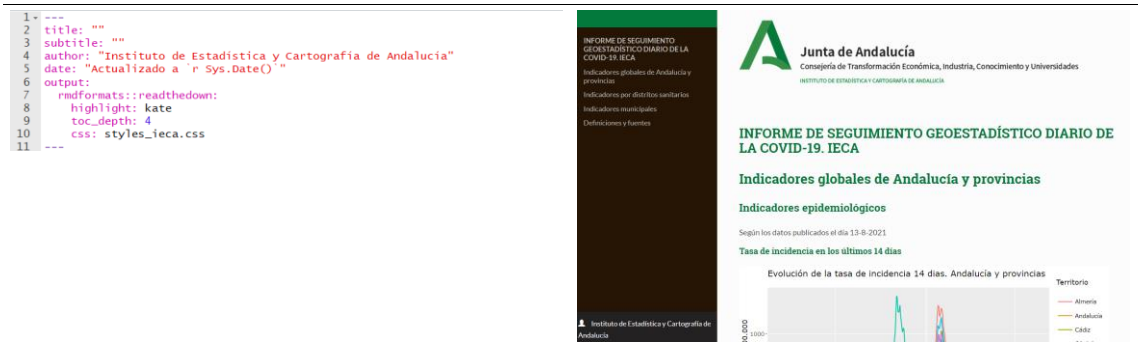
En el caso de estos informes para la desescalada, los datos sanitarios procedían de las tablas diseñadas para la difusión de la información a la ciudadanía. Se obtenían de una descarga en hoja de cálculo de los mismos. La lectura en JSON no era posible porque fueron tablas a medida realizadas para el informe, y no estaban publicadas de cara a terceros, lo que impedía esta forma de acceder a los datos.

En referencia a los datos de movilidad los datos del INE utilizados en los primeros informes tenían formato ".xls" y se incorporaban directamente al fichero Rmd. Los datos de ministerio, al tener un carácter más detallado, tenían un tamaño mucho mayor y su formato era "txt". Estos ficheros se incorporaban directamente a una base de datos postgres, a la que se llamaba para la elaboración del informe.

Visualización final

Para la visualización de estos informes se optó por HTML, que permitía una cierta interacción con los datos y asegurar que los mismos no fueran manipulables, ya que eran informes que iban a ser distribuidos a un gran número de gestores y también a la ciudadanía.

Figura 9: Script de inicio del informe (izquierda) y captura de salida (derecha)



Principales scripts del informe de desescalada

Algunos de los scripts más interesantes utilizados en los informes de desescalada fueron los siguientes:

Gráficos evolutivos interactivos

Al optarse por una salida en HTML se optó también por hacer los gráficos lo más atractivos posibles. Por ello se decidió utilizar en ellos la *librería plotly*, que volvía interactivos los gráficos de la *librería ggplot2*.

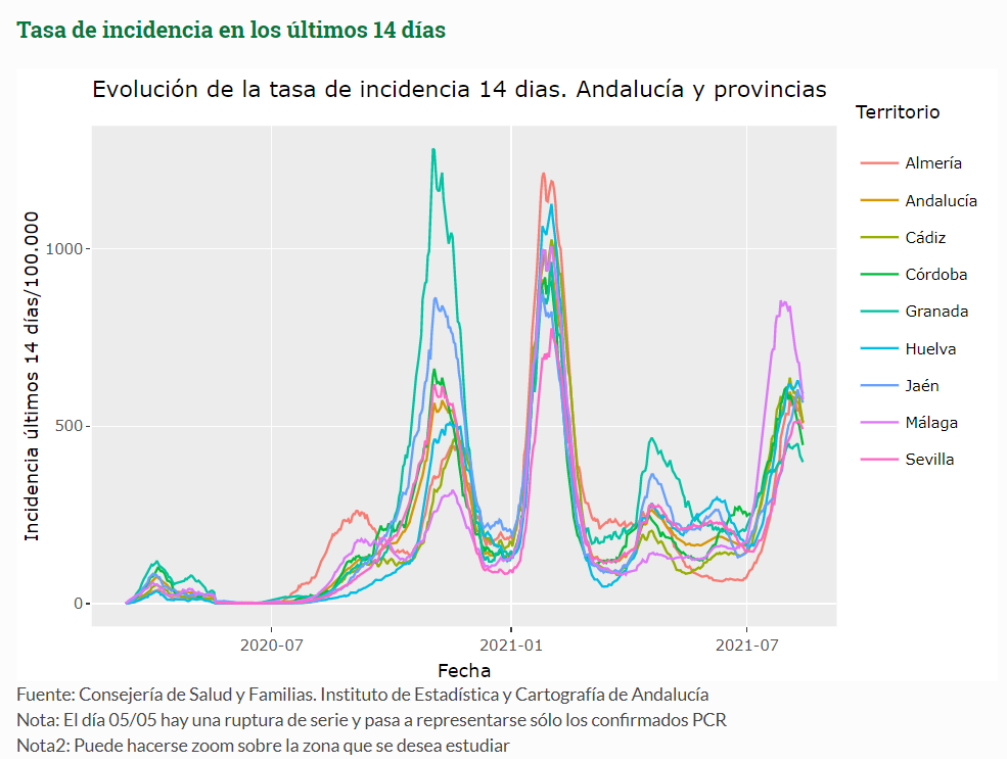
La interactividad se traducía en poder activar y desactivar las distintas series representadas (provincias en el ejemplo mostrado) o mostrar el valor de cada punto que marcara el ratón.

Figura 10: Script con ejemplo gráfico evolutivo con ggplot2 e interactivo con plotly (arriba) y salida (abajo)

```
#Grafico

gepi2<-ggplot(Evolucion_casos %>% filter(Fecha > "2020-03-10"))+geom_line(aes(x = Fecha, y = Incidencia_14dias, colour = Territorio))+labs(title="Evolución de la tasa de incidencia 14 dias. Andalucía y provincias",x="Fecha", y="Incidencia últimos 14 dias/100.000")+ labs(caption = "Consejeria de Salud y Familias. Instituto de Estadistica y Cartografia de Andalucia")

ggplotly(gepi2)
```



Generación de tablas

Para las tablas en este informe se optó por la *librería kableExtra*, que daba mayor vistosidad a las tablas, pudiendo marcar en ellas puntos de debilidad cuando los datos tomaban determinados valores.

Figura 11: Script con ejemplo de tabla generada con kableExtra (arriba) y salida (abajo)

```
T_var[,c(1,3:6)] %>%
   select(everything())%>%
  `colnames<-`(c("Territorio","Mayores 65 años","Var 65","Jóvenes: 15-29 años","Var 15-29")) %>%
      kable(escape = F) %>%
  kable_styling("striped","hover", full_width = T, position = "center")%>%
   column_spec(2,    background = "aliceblue")%>%
     column_spec(3, color = ifelse(T_var$Var_65 > 0, "red","black"),  background = "aliceblue")%>%
   column_spec(5, color = ifelse(T_var$Var_15 > 0, "red","black"),  background = "antiquewhite")%>%
   column_spec(4,  background = "antiquewhite")%>%
  footnote(general="La variaciones son semanales",general_title = "Nota 1: ")%>%
 footnote(general="IECA a partir de los datos de la Consejería de Salud y Familias",general_title = "Fuente: ")%>%
add_header_above(c( "Tasas de incidencia en los últimos 14 dias" = 5))
```

Tabla de incidencia por grupos de edad y provincias

	Tasas de incidencia en los últimos 14 dias			
Territorio	Mayores 65 años	Var 65	Jóvenes: 15-29 años	Var 15-29
Andalucía	268.7	7.7	1210.3	-21.8
Almería	296.3	10.3	1182.2	-24.2
Cádiz	234.5	-5.2	1240.0	-22.4
Córdoba	266.4	-10.3	1064.0	-28.9
Granada	251.0	38.4	916.0	-18.5
Huelva	293.9	12.9	1458.9	-13.5
Jaén	268.8	38.7	1324.3	-14.0
Málaga	318.7	-4.5	1211.2	-32.5
Sevilla	239.9	20.2	1298.9	-13.1
Fuente:				
IECA a partir de los datos de la Consejería de Salud y Familias				
Nota 1:				
La variaciones son semanales				

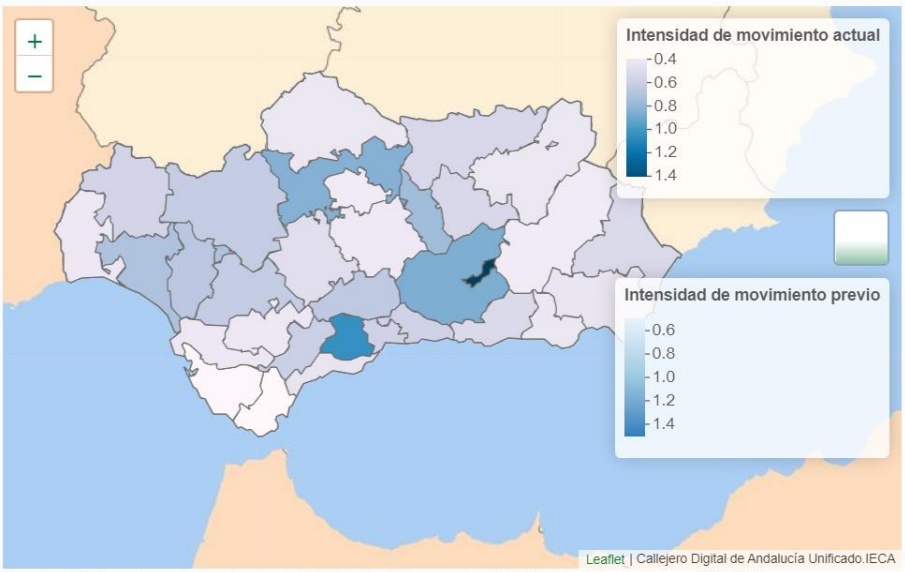
Generación de mapas

Se consideró relevante también, dado que el informe procedía del IECA, incorporar mapas que ayudaran a la gestión. El informe contiene mapas a niveles provincial, de distrito sanitario y municipal.

La librería que se utilizó para los mapas fue *tmap*, que además permitía interactuar sobre el mapa para conocer el valor exacto en cada zona tocada por el ratón.

Figura 12: Script con ejemplo gráfico de mapa interactivo con tmap (arriba) y salida (abajo)

```
47
48 a02DS<-tm_basemap(server = NA) + tmap_options(basemaps=NULL) +
49   tm_shape(DS1, simplify = 0.05) +
50   tm_fill(col="IND_MOV_DS",style="cont",palette="Blues",id="literalds",
51   title = "Intensidad de movimiento previo",popup.vars=c("Intensidad de movimiento previo"="IND_MOV_DS"),
52   popup.format = list(fun =
53               function(x) formatC(x, digits = 1,big.mark = ".", decimal.mark=",",format = "f")),
54   legend.format = list(fun =
55               function(x) formatC(x, digits = 1,big.mark = ".", decimal.mark=",",format = "f")),
56 group="Intensidad de movimiento previo")+
57   tm_borders()+tm_layout(legend.position = c("left", "bottom"),frame=FALSE) +
58 tm_shape(DS2, simplify = 0.05) +
59   tm_fill(col="IND_MOV2",style="cont",palette="PuBu",id="literalds",title = "Intensidad de movimiento
   actual",popup.vars=c("Intensidad de movimiento actual"="IND_MOV2"),
60   popup.format = list(fun =
61               function(x) formatC(x, digits = 1,big.mark = ".", decimal.mark=",",format = "f")),
62   legend.format = list(fun =
63               function(x) formatC(x, digits = 1,big.mark = ".", decimal.mark=",",format = "f")),
64 group="Intensidad de movimiento actual")+
65   tm_borders()+tm_layout(legend.position = c("left", "bottom"),frame=FALSE)
66
67
68 tmap_leaflet(a02DS,options = leafletOptions(
69             attributionControl=FALSE)) %>%
70   addWMSTiles("http://www.callejerodeandalucia.es/servicios/base/wms?",
71               layers = c("contexto_andalucia","nucleos_poblacion","batimetria"),
72               options = WMSTileOptions(format = "image/jpeg", transparent = FALSE),
73               attribution = "Callejero Digital de Andalucía Unificado.IECA",group = "CDAU") %>%
74   addLayersControl(baseGroups = c("CDAU"),
75               overlayGroups = c("Intensidad de movimiento previo","Intensidad de movimiento actual"),
   options = layersControlOptions(collapsed = TRUE)) %>%
76   hideGroup(c("Intensidad de movimiento previo"))
77
```



Fuente: IECA basado en datos abiertos del Ministerio de Transportes, Movilidad y Agenda Urbana (MITMA)

Conclusión

Rmarkdown se muestra como una buena estructura para la realización de informes que deban **distribuirse a terceros**. Las herramientas de análisis estadístico tipo R (ocurre lo mismo para otras como SPSS, SQL, hoja de cálculo…) y el formato en el que devuelven sus resultados está muy lejos de ser comprendido, atractivo y/o divulgativo por aquellos que no estén involucrados en el proceso. Rmarkdown consigue esa fusión entre el cálculo llevado a cabo por los analistas y la visualización de resultados que desean los gestores o decisores, mostrando a los segundos en la salida la información procesada y "ocultándoles" la programación, la cual queda en manos de los analistas. Ambos trabajan con el mismo programa sin los inconvenientes e interferencias que se producen cuando se copian ficheros o imágenes de unos formatos a otros.

Como código de programación, Rmarkdown presenta la ventaja de que los resultados dependen exclusivamente de la nueva llegada de datos, siendo el código que lo define estable para una estructura concreta de datos y mientras esta se mantenga solo habrá que ejecutar el programa. Es perfecto entonces cuando los informes tienen un **carácter repetitivo** con estructura de datos estables, como han sido estos informes.

Los principales inconvenientes que encontramos en la realización de estos informes fueron:

- El **tiempo**. La elaboración de los informes, tanto los sanitarios como los de desescalada, se llevó a cabo en un periodo tiempo corto que supuso improvisación y no optimización de los procesos. Eso no le restó eficiencia al resultado, que sí que lo obtuvo, pero somos conscientes que de haber tenido tiempo de contrastar procesos estos se hubieran realizado de otra forma.

- La información diaria requería procedimientos bastante generales de tratamiento de datos. Aun así, en más de una ocasión, datos puntuales demasiado extraños o falta de datos, **bloqueaban procesos** diseñados para funcionar con una estructura determinada y donde no podía estar previsto esos cambios.

- Las grandes cantidades de datos obligó a la creación de **otras bases de datos** en otros formatos (Postgres) o a particionar ficheros (caso de los informes sanitarios). Bases de datos numerosas siempre ralentizan los procesos y genera ficheros más pesados

- La **interactividad** es una ventaja pero también hace que se "arrastren" todos los datos en todo momento, generándose ficheros en ocasiones demasiados grandes.

- La visualización de Rmarkdown en HTML (informes de desescalada), si bien es atractivo visualmente, encapsula un fichero, generalmente de gran tamaño, que no podía ser visualizado directamente sino **descargado previamente** en un zip y luego abierto.

- Como propuesta de mejora con disponibilidad de tiempo y medios, se podrían construir **Aplicaciones Web** desde R utilizando el paquete **Shiny**. Esto permitiría la interacción de los usuarios con los datos además de facilitar la distribución de los informes entre los usuarios finales, sin necesidad de descarga de archivos pesados.

Principales referencias bibliográficas

- Package ‘rmarkdown’ (Yihui Xie et al, 2021) (cran.r-project.org)
- Package ‘dplyr’ (Hadley Wickham et al, 2021) (cran.r-project.org)
- Package ‘ggplot2’ (Hadley Wickham et al, 2021) (cran.r-project.org)
- Package ‘gridExtra’ (Baptiste Auguie et al, 2017) (cran.r-project.org)
- Package ‘flextable’ (David Gohel et al, 2021) (cran.r-project.org)
- Package ‘epiEstim’ (Anne Cori et al, 2021) (cran.r-project.org)
- Package ‘tmap’ (Richard A. Becker et al, 2021) (cran.r-project.org)
- Package ‘kableExtra’ (Hao Zhu et al, 2021) (cran.r-project.org)
- Package ‘plotly’ (Carson Sievert et al, 2021) (cran.r-project.org)
- Package ‘readxl’ (Jennifer Bryan et al, 2019) (cran.r-project.org)