Replication code for The institutional impacts of algorithmic distribution: Facebook and the Australian news media

Francesco Bailo, James Meese, Eddy Hurcombe

19 April, 2021

Libraries

```
required_packages <-
  c("tidyverse", "readxl", "zoo", "parallel", "kableExtra")

new_packages <-
  required_packages[!(required_packages %in%
                        installed.packages()[,"Package"])]

if(length(new_packages)) install.packages(new_packages)

lapply(required_packages, require, character.only = TRUE)
```

```
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
```

```
sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_AU.UTF-8/en_AU.UTF-8/en_AU.UTF-8/C/en_AU.UTF-8/en_AU.UTF-8
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
```

```
## [8] base
##
## other attached packages:
##  [1] kableExtra_1.1.0 zoo_1.8-6        readxl_1.3.1     forcats_0.5.1
##  [5] stringr_1.4.0    dplyr_1.0.5      purrr_0.3.4      readr_1.4.0
##  [9] tidyr_1.1.3      tibble_3.0.4     ggplot2_3.3.3    tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.1.0  xfun_0.21         lattice_0.20-38   haven_2.4.0
##  [5] colorspace_1.4-1  vctrs_0.3.7       generics_0.1.0    viridisLite_0.3.0
##  [9] htmltools_0.4.0   yaml_2.2.0        utf8_1.1.4        rlang_0.4.10
## [13] pillar_1.6.0      glue_1.4.2        withr_2.4.1       DBI_1.0.0
## [17] dbplyr_2.1.1      modelr_0.1.8      lifecycle_1.0.0   munsell_0.5.0
## [21] gtable_0.3.0      cellranger_1.1.0  rvest_1.0.0       evaluate_0.14
## [25] knitr_1.26        fansi_0.4.1       broom_0.7.6       Rcpp_1.0.3
## [29] scales_1.1.0      backports_1.1.5   webshot_0.5.2     jsonlite_1.7.2
## [33] fs_1.3.1          hms_1.0.0         digest_0.6.27     stringi_1.5.3
## [37] grid_3.6.1        cli_2.4.0         tools_3.6.1       magrittr_2.0.1
## [41] crayon_1.4.1      pkgconfig_2.0.3   ellipsis_0.3.1    xml2_1.3.2
## [45] reprex_2.0.0      lubridate_1.7.10  assertthat_0.2.1  rmarkdown_1.17
## [49] httr_1.4.2        rstudioapi_0.13   R6_2.5.0          compiler_3.6.1
```

Setup

```
ggplot2::theme_set(theme_bw())

knitr::opts_chunk$set(
  dev = "pdf", fig.width = 8, fig.height = 7, dpi=300, message=FALSE, warning=FALSE
)
```

Traffic data

```
dat <-
  read.csv("data-semrush.csv")

dat$date <-
  as.Date(dat$date)

dat <-
  dat[!is.na(dat$date),]

dat$domain[dat$domain == 't.co'] <-
  "twitter.com"

labelling_list <- c(abc = "abcnews.au",
                    adelaideadvertiser = "theadvertiser",
                    afr = "financialreview",
                    brisbanetimes = "brisbanetimes",
                    buzzfeed = "BuzzFeedOz",
                    canberratimes = "CanberraTimes",
                    couriermail = "couriermail",
                    crikey = "crikey.com.au",
```

```
                    dailytele = "dailytelegraph",
                    diewelt = "NA",
                    heraldsun = "heraldsun",
                    huffingtonpostau = "HuffPostAustralia",
                    junkee = "junkeedotcom",
                    mamamia = "mamamia",
                    newmatilda = "newmatildadotcom",
                    newscomau = "news.com.au",
                    nine = "Ninecomau", #?
                    ntnews = "TheNTNews",
                    ped = "pedestriandottv",
                    sbs = "sbsnews",
                    smh = "sydneymorningherald",
                    svd = "NA",
                    telegraphuk = "NA",
                    theaustralian = "theaustralian",
                    theguardian = "theguardianaustralia",
                    wapo = "NA",
                    watoday = "WAtoday",
                    `the new daily` = "TheNewDaily", # Digital
                    `the west` = "thewestaustralian", #  Print
                    theage = "theageAustralia"
)

dat$publication_join <-
  labelling_list[match(dat$publication,
                       names(labelling_list))]

Categoryforanalysis <-
  read_excel("Categoryforanalysis.xlsx")

dat$category <-
  Categoryforanalysis$Category[match(
    dat$publication_join,
    gsub("https://www.facebook.com/|/$", "", Categoryforanalysis$`Site/Publication`))]

dat$source_type[dat$search_engine == "Google" & dat$source_type == "search"] <-
  "Google search"
dat$source_type[dat$search_engine != "Google" & dat$source_type == "search"] <-
  "other search"

fig1_dat <-
  dat %>%
  dplyr::filter(date < as.Date("2020-01-01") & !is.na(category)) %>%
  dplyr::mutate(date = factor(format(date, "%b %Y"), levels = c("Oct 2017",
                                                                "Nov 2017",
                                                                "Dec 2017",
                                                                "Jan 2018",
                                                                "Feb 2018",
                                                                "Mar 2018"))) %>%
  dplyr::group_by(publication, category, date, source_type) %>%
  dplyr::summarise(traffic_share = sum(traffic_share, na.rm = T)) %>%
  dplyr::ungroup() %>%
```

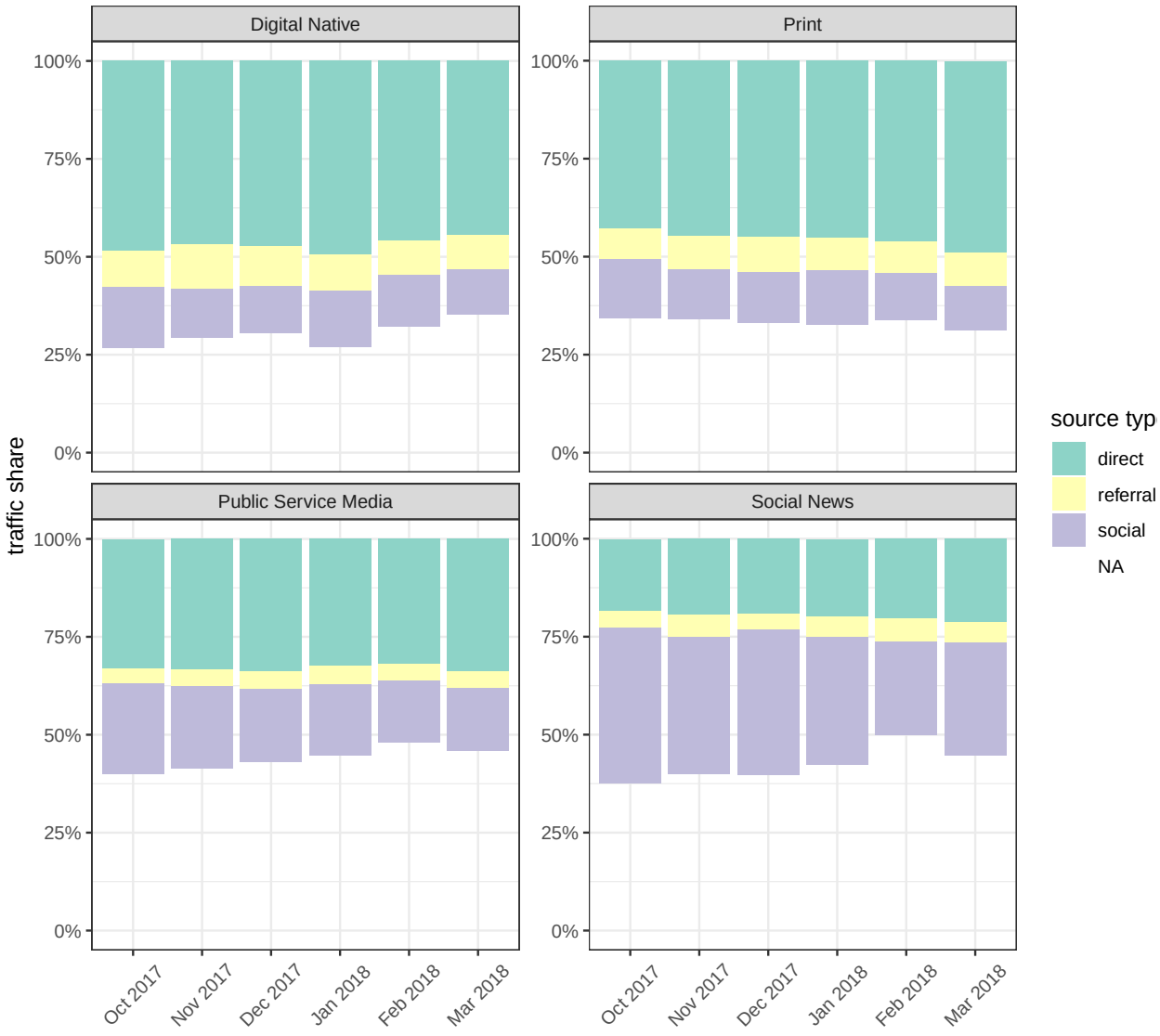


Figure 1: Traffic share (by source type, mean)

```
  dplyr::group_by(category, date, source_type) %>%
  dplyr::summarise(traffic_share = mean(traffic_share, na.rm = T))

fig1_dat %>%
  ggplot(aes(x=date, y=traffic_share, fill = source_type)) +
  scale_fill_brewer(palette = "Set3") +
  scale_y_continuous(label = function(x) paste0(x*100,"%")) +
  geom_bar(stat = 'identity') +
  facet_wrap(category~., scales = "free_y") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5)) +
  labs(x = NULL, y = "traffic share", fill = "source type")
```

```
write.csv(fig1_dat, file = 'out/fig1_dat.csv')
```

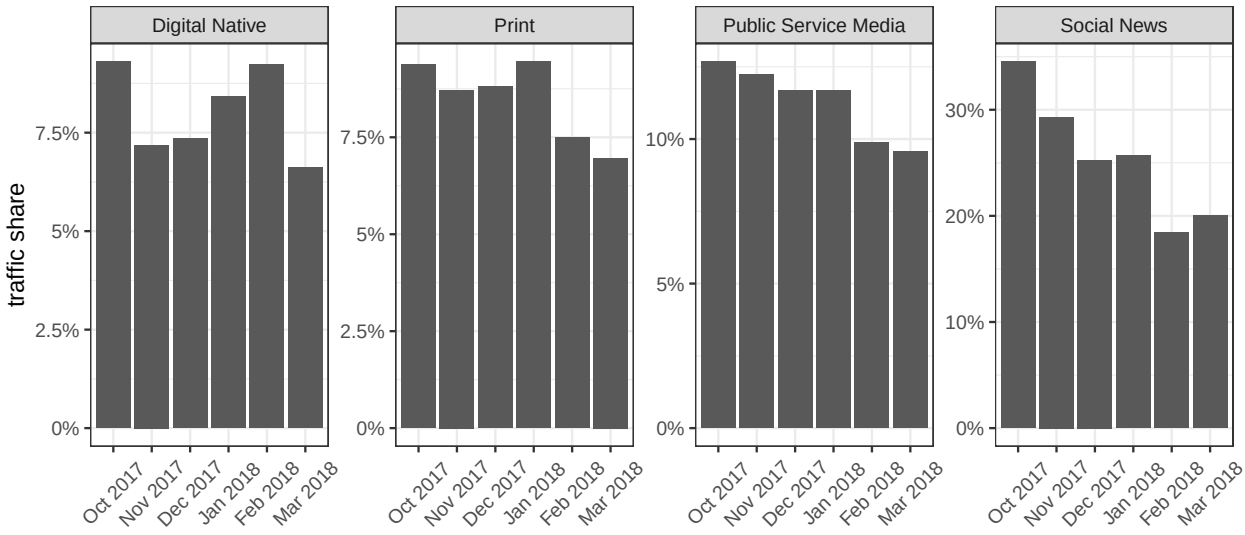


Figure 7: Traffic share (social and Facebook.com only, mean)

```
fig7_dat <-
  dat %>%
  dplyr::filter(date < as.Date("2020-01-01") & !is.na(category)) %>%
  dplyr::mutate(date = factor(format(date, "%b %Y"), levels = c("Oct 2017",
                                                                "Nov 2017",
                                                                "Dec 2017",
                                                                "Jan 2018",
                                                                "Feb 2018",
                                                                "Mar 2018"))) %>%
  dplyr::filter(domain == "facebook.com") %>%
  dplyr::group_by(category, publication, date) %>%
  dplyr::summarise(traffic_share = sum(traffic_share, na.rm = T)) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(category, date) %>%
  dplyr::summarise(traffic_share = mean(traffic_share, na.rm = T))

 fig7_dat %>%
  ggplot(aes(x=date, y=traffic_share)) +
  geom_bar(stat = 'identity') +
  scale_y_continuous(label = function(x) paste0(x*100,"%")) +
  facet_wrap(category~., scales = "free_y", ncol = 4) +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5)) +
  labs(x = NULL, y = "traffic share")
```

```
write.csv(fig7_dat, file = 'out/fig7_dat.csv')
```

```
fig8_dat <-
  dat %>%
  dplyr::select(-X) %>%
  dplyr::distinct() %>%
  dplyr::filter(domain == "facebook.com" &
                  publication %in% c("abc", "ped", "buzzfeed", "heraldsun", "theguardian")) %>%
  dplyr::filter(date < as.Date("2020-01-01") & !is.na(category)) %>%
  dplyr::mutate(date = factor(format(date, "%b %Y"), levels = c("Oct 2017",
```

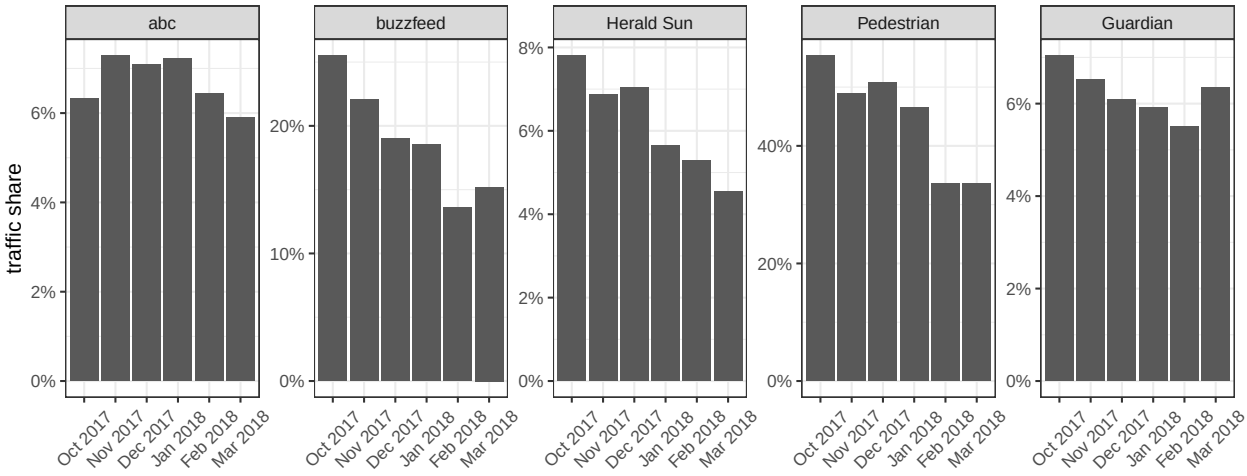


Figure 8: Traffic share (social and Facebook.com only, sum)

```
                                                        "Nov 2017",
                                                        "Dec 2017",
                                                        "Jan 2018",
                                                        "Feb 2018",
                                                        "Mar 2018")),
                publication = recode(publication, sbs = "SBS News", ped = "Pedestrian",
                                     heraldsun = "Herald Sun", theguardian = "Guardian",
                                     )) %>%
  dplyr::group_by(publication, date) %>%
  dplyr::summarise(traffic_share = sum(traffic_share, na.rm = T)) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(publication) %>%
  dplyr::arrange(date) %>%
  # dplyr::mutate(traffic_share = traffic_share / traffic_share[1]) %>%
  dplyr::arrange(publication)

 fig8_dat %>%
  ggplot(aes(x=date, y=traffic_share)) +
  # geom_hline(yintercept = 1) +
  geom_bar(stat='identity') +
  scale_y_continuous(label = function(x) paste0(x*100,"%")) +
  facet_wrap(publication~., scales = "free_y", nrow = 1) +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5)) +
  labs(x = NULL, y = "traffic share")
```

```
write.csv(fig8_dat, file = 'out/fig8_dat.csv')
```

CrowdTangle data

```
load("australian_media_list.df.RData")

load("ct_au_media_dat.RData")
```

Data timeframe: 2014-01-01 - 2020-12-15

Functions

```
benchmarking <- function(x) {
  x <- x[x > quantile(x, probs = .25) &
           x < quantile(x, probs = .75)]
  return(mean(x, na.rm =T))
}

computePerfScoreInteractions <- function(i) {

  this_dat <-
    dat_wt_benchmark %>%
    dplyr::filter(day >= this_seq[i] - 14 &
                    day < this_seq[i] + 15) %>%
    dplyr::group_by(Page.Name, page_type) %>%
    dplyr::summarize(Performance.Score =
                       benchmarking(Interactions) /
                       benchmark.score[1])
  this_dat$day <-
    this_seq[i]

  return(this_dat)
}

computePerfScoreComments <- function(i) {

  this_dat <-
    dat_wt_benchmark %>%
    dplyr::filter(day >= this_seq[i] - 14 &
                    day < this_seq[i] + 15) %>%
    dplyr::group_by(Page.Name, page_type) %>%
    dplyr::summarize(Performance.Score =
                       benchmarking(Comments) /
                       benchmark.score[1])
  this_dat$day <-
    this_seq[i]

  return(this_dat)
}

computePerfScoreShares <- function(i) {

  this_dat <-
    dat_wt_benchmark %>%
    dplyr::filter(day >= this_seq[i] - 14 &
                    day < this_seq[i] + 15) %>%
    dplyr::group_by(Page.Name, page_type) %>%
    dplyr::summarize(Performance.Score =
                       benchmarking(Shares) /
                       benchmark.score[1])
  this_dat$day <-
    this_seq[i]

  return(this_dat)
```

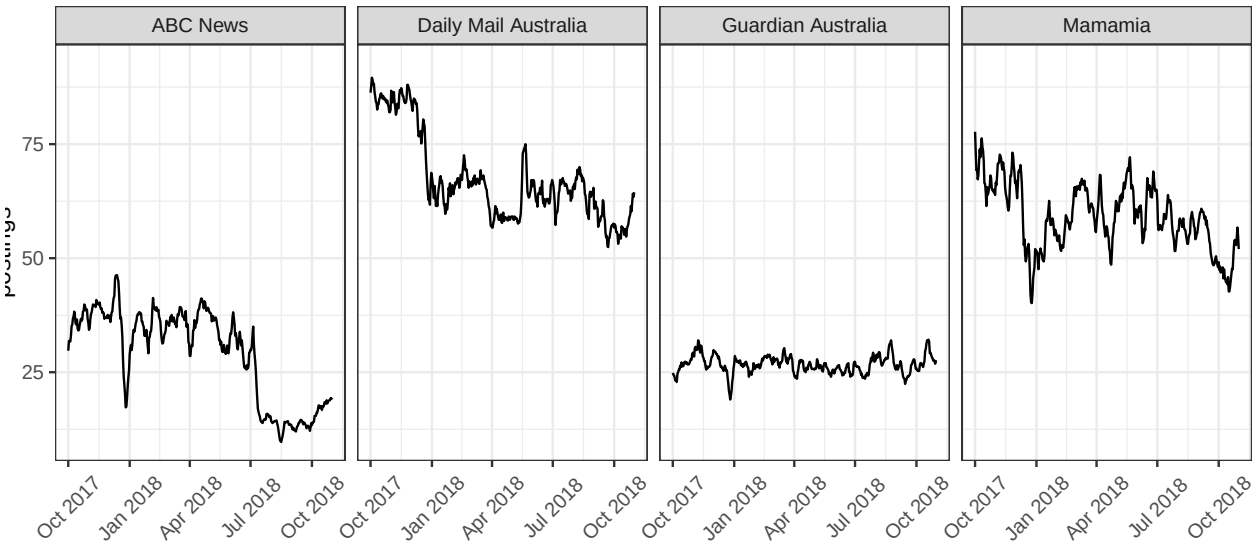


Figure 9: Number of postings (7-day moving average)

```
}
```

Time frame of analysis

```
benchmarking_timeframe <-
  c(as.Date("2017-01-01"), as.Date("2017-04-01"))

date_limits <-
  c(as.Date("2014-01-01"), as.Date("2020-12-15"))
```

Posting frequency

```
dat %>%
  dplyr::filter(Page.Name %in% c("Daily Mail Australia",
                                 "ABC News", "Mamamia",
                                 "Guardian Australia"),
                posix >= as.Date("2017-09-01") &
                  posix < as.Date("2018-12-01")) %>%
  dplyr::group_by(Page.Name, date = as.Date(posix)) %>%
  dplyr::count() %>%
  dplyr::ungroup() %>%
  dplyr::group_by(Page.Name) %>%
  dplyr::arrange(date) %>%
  dplyr::mutate(n_ma7 = rollmean(n, 7, fill = NA)) %>%
  ggplot(aes(x = date, y = n_ma7)) +
  geom_line() +
  facet_wrap(Page.Name~., ncol = 4) +
  labs(x = NULL, y = "postings") +
  scale_x_date(date_labels = "%b %Y", limits = as.Date(c("2017-10-01", "2018-10-31"))) +
    theme(axis.text.x = element_text(angle = 45, vjust = 0.5))
```

Weighting

```
weighting_by_type_on_links.df <-
  dat %>%
  dplyr::filter(Type == "Link") %>%
  dplyr::group_by(Page.Name, page_type, day, .drop = FALSE) %>%
  dplyr::count() %>%
  dplyr::ungroup() %>%
  dplyr::group_by(Page.Name) %>%
  dplyr::arrange(day) %>%
  dplyr::mutate(n_ma30 = rollmean(n, k = 30, fill = NA, align = 'center')) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(day) %>%
  dplyr::mutate(global_weight = n_ma30 / sum(n_ma30)) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(page_type, day) %>%
  dplyr::mutate(page_type_weight = n_ma30 / sum(n_ma30)) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(Page.Name) %>%
  dplyr::arrange(day) %>%
  dplyr::mutate()
```

```
weighting_by_type_on_links.df %>%
  dplyr::filter(day == "2020-01-01") %>%
  dplyr::arrange(page_type, desc(global_weight)) %>%
  dplyr::mutate(global_weight = round(global_weight*100,2),
                page_type_weight = round(page_type_weight*100,2)) %>%
  # write.csv(file = "table8.csv", row.names = FALSE) %>%
  kable(booktabs = T, caption = "") %>%
  kable_styling(latex_options = c("striped", "scale_down"))
```

Interactions

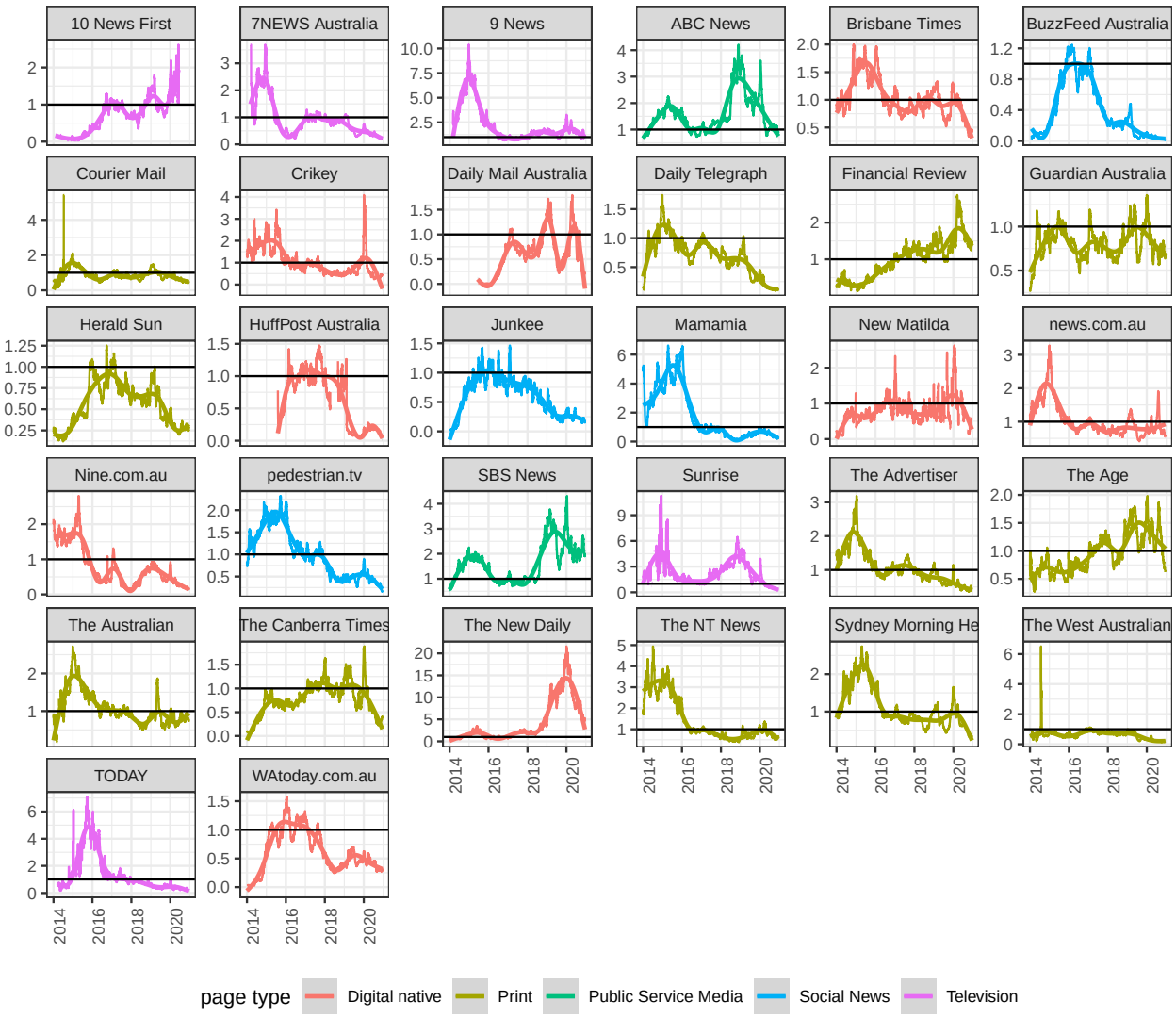
```
{
  sink("/dev/null");
  cl <- makeCluster(10)
  clusterEvalQ(cl, {
    library(dplyr)
    library(magrittr)
  })
}

clusterExport(cl, "benchmarking")

benchmark <-
  dat %>%
  dplyr::filter(posix >= benchmarking_timeframe[1] &
                  posix <= benchmarking_timeframe[2]) %>%
  dplyr::group_by(Page.Name, Type) %>%
  dplyr::summarise(benchmark.score =
                     benchmarking(Likes+Love+Wow+Haha+Sad+Angry+Care))

dat_wt_benchmark <-
```

Table 1

Page.Name	page_type	day	n	n_ma30	global_weight	page_type_weight
news.com.au	Digital native	2020-01-01	38	41.000000	5.98	24.42
Daily Mail Australia	Digital native	2020-01-01	23	36.533333	5.33	21.76
Nine.com.au	Digital native	2020-01-01	12	21.300000	3.11	12.69
HuffPost Australia	Digital native	2020-01-01	11	19.933333	2.91	11.87
WAtoday.com.au	Digital native	2020-01-01	19	19.800000	2.89	11.80
Brisbane Times	Digital native	2020-01-01	17	17.433333	2.54	10.39
The New Daily	Digital native	2020-01-01	11	9.000000	1.31	5.36
Crikey	Digital native	2020-01-01	1	1.533333	0.22	0.91
New Matilda	Digital native	2020-01-01	0	1.333333	0.19	0.79
The Sydney Morning Herald	Print	2020-01-01	33	36.333333	5.30	12.10
The West Australian	Print	2020-01-01	37	36.233333	5.29	12.07
Daily Telegraph	Print	2020-01-01	36	30.533333	4.46	10.17
Courier Mail	Print	2020-01-01	32	30.433333	4.44	10.14
Herald Sun	Print	2020-01-01	26	27.400000	4.00	9.13
The Age	Print	2020-01-01	30	26.166667	3.82	8.72
The Advertiser	Print	2020-01-01	20	24.700000	3.60	8.23
Guardian Australia	Print	2020-01-01	22	24.400000	3.56	8.13
The Australian	Print	2020-01-01	15	21.133333	3.08	7.04
The NT News	Print	2020-01-01	22	17.600000	2.57	5.86
The Canberra Times	Print	2020-01-01	15	13.233333	1.93	4.41
Financial Review	Print	2020-01-01	10	12.066667	1.76	4.02
SBS News	Public Service Media	2020-01-01	17	19.166667	2.80	51.99
ABC News	Public Service Media	2020-01-01	15	17.700000	2.58	48.01
Mamamia	Social News	2020-01-01	25	24.866667	3.63	36.84
pedestrian.tv	Social News	2020-01-01	14	21.900000	3.20	32.44
BuzzFeed Australia	Social News	2020-01-01	11	12.733333	1.86	18.86
Junkee	Social News	2020-01-01	3	8.000000	1.17	11.85
7NEWS Australia	Television	2020-01-01	53	58.900000	8.60	52.26
9 News	Television	2020-01-01	19	21.833333	3.19	19.37
10 News First	Television	2020-01-01	15	16.900000	2.47	15.00
Sunrise	Television	2020-01-01	7	10.966667	1.60	9.73
TODAY	Television	2020-01-01	5	4.100000	0.60	3.64

```
  merge(dat, benchmark,
        by = c("Page.Name","Type"))

dat_wt_benchmark$Interactions <-
  dat_wt_benchmark$Likes +
  dat_wt_benchmark$Love +
  dat_wt_benchmark$Wow +
  dat_wt_benchmark$Haha +
  dat_wt_benchmark$Sad +
  dat_wt_benchmark$Angry +
  dat_wt_benchmark$Care

this_seq <- seq(date_limits[1],
                date_limits[2], by = "day")

clusterExport(cl, "this_seq")

dat_wt_benchmark <-
  dat_wt_benchmark %>%
  dplyr::filter(Type %in% c("Link")) %>%
  dplyr::mutate(day = as.Date(posix)) %>%
  dplyr::select(Page.Name, page_type, Interactions, benchmark.score, day)

clusterExport(cl, "dat_wt_benchmark")

par_res <-
  parLapply(cl, 1:length(this_seq), fun = computePerfScoreInteractions)

interactions_ma30_dat <-
  bind_rows(par_res)

stopCluster(cl)

interactions_dat_wt_benchmark <-
  dat_wt_benchmark

save(interactions_dat_wt_benchmark, file = "interactions_dat_wt_benchmark.RData")
save(interactions_ma30_dat, file = "interactions_ma30_dat.RData")
```

```
load("interactions_ma30_dat.RData")

interactions_ma30_dat %>%
  ggplot(aes(x = day, y = Performance.Score, colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(Page.Name~., scales = "free_y", ncol = 6) +
  geom_hline(yintercept = 1) +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=90, hjust=1),
        legend.position = 'bottom') +
  labs(x = NULL, y = NULL, colour = "page type")
```



Performance score on interactions

Clustered (weighted average)

```
interactions_ma30_dat <-
  merge(interactions_ma30_dat,
        weighting_by_type_on_links.df %>%
          dplyr::select(Page.Name, day, global_weight, page_type_weight),
        by = c("Page.Name", "day"))

interactions_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
                                   "2020-10-01"))) %>%
  dplyr::arrange(page_type, day) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
interactions_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score,
             colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(page_type~., ncol = 1, scales = "free_y") +
  theme(strip.text.x = element_text(size = 8, margin = margin(0.05,0,0.05,0, "cm"))) +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL) +
  guides(colour = FALSE)
```

```
load("interactions_dat_wt_benchmark.RData")

interactions_dat_wt_benchmark %>%
  dplyr::filter(page_type %in% c("Public Service Media")) %>%
  dplyr::group_by(Page.Name, day) %>%
  dplyr::summarise(perc.above.benchmark = sum(Interactions > benchmark.score) / n()) %>%
  dplyr::ungroup() %>%
  dplyr::group_by(Page.Name) %>%
  dplyr::arrange(day) %>%
  dplyr::mutate(perc.above.benchmark.30ma = rollmean(perc.above.benchmark, k = 30, fill = NA)) %>%
  ggplot(aes(x = day, y = perc.above.benchmark.30ma)) +
  scale_y_continuous(label = function(x) paste0(x*100,"%")) +
  geom_line() +
```

Table 2

day	page_type	Performance.Score
2014-12-31	Digital native	1.7696288
2015-12-31	Digital native	1.0557573
2016-12-31	Digital native	0.9929409
2017-12-31	Digital native	0.6558987
2018-12-31	Digital native	0.9672086
2019-12-31	Digital native	1.7841097
2020-10-01	Digital native	1.1086158
2014-12-31	Print	1.5394645
2015-12-31	Print	1.1236932
2016-12-31	Print	0.9548350
2017-12-31	Print	0.8703664
2018-12-31	Print	1.0341134
2019-12-31	Print	1.0651512
2020-10-01	Print	0.5329970
2014-12-31	Public Service Media	1.7760692
2015-12-31	Public Service Media	1.7070015
2016-12-31	Public Service Media	1.0637033
2017-12-31	Public Service Media	1.1713905
2018-12-31	Public Service Media	2.9911517
2019-12-31	Public Service Media	3.5441487
2020-10-01	Public Service Media	1.3282974
2014-12-31	Social News	2.2402395
2015-12-31	Social News	3.8263553
2016-12-31	Social News	1.0327610
2017-12-31	Social News	0.6720737
2018-12-31	Social News	0.2828542
2019-12-31	Social News	0.6795199
2020-10-01	Social News	0.2404028
2014-12-31	Television	4.2098702
2015-12-31	Television	1.3409587
2016-12-31	Television	1.0395455
2017-12-31	Television	1.1441845
2018-12-31	Television	1.4002514
2019-12-31	Television	1.5255005
2020-10-01	Television	0.3955611

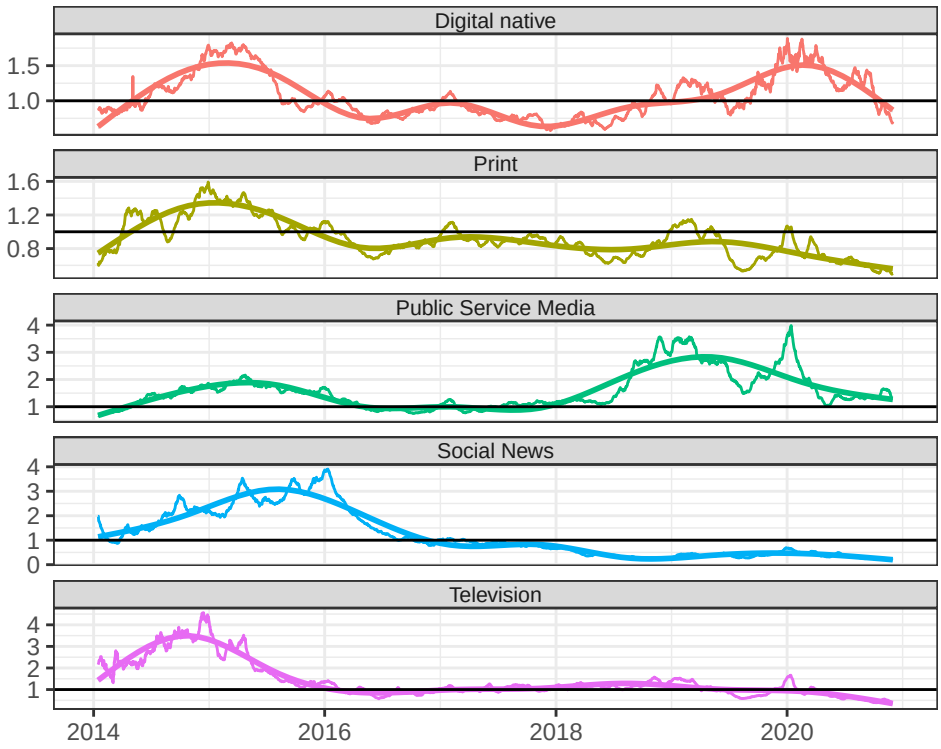
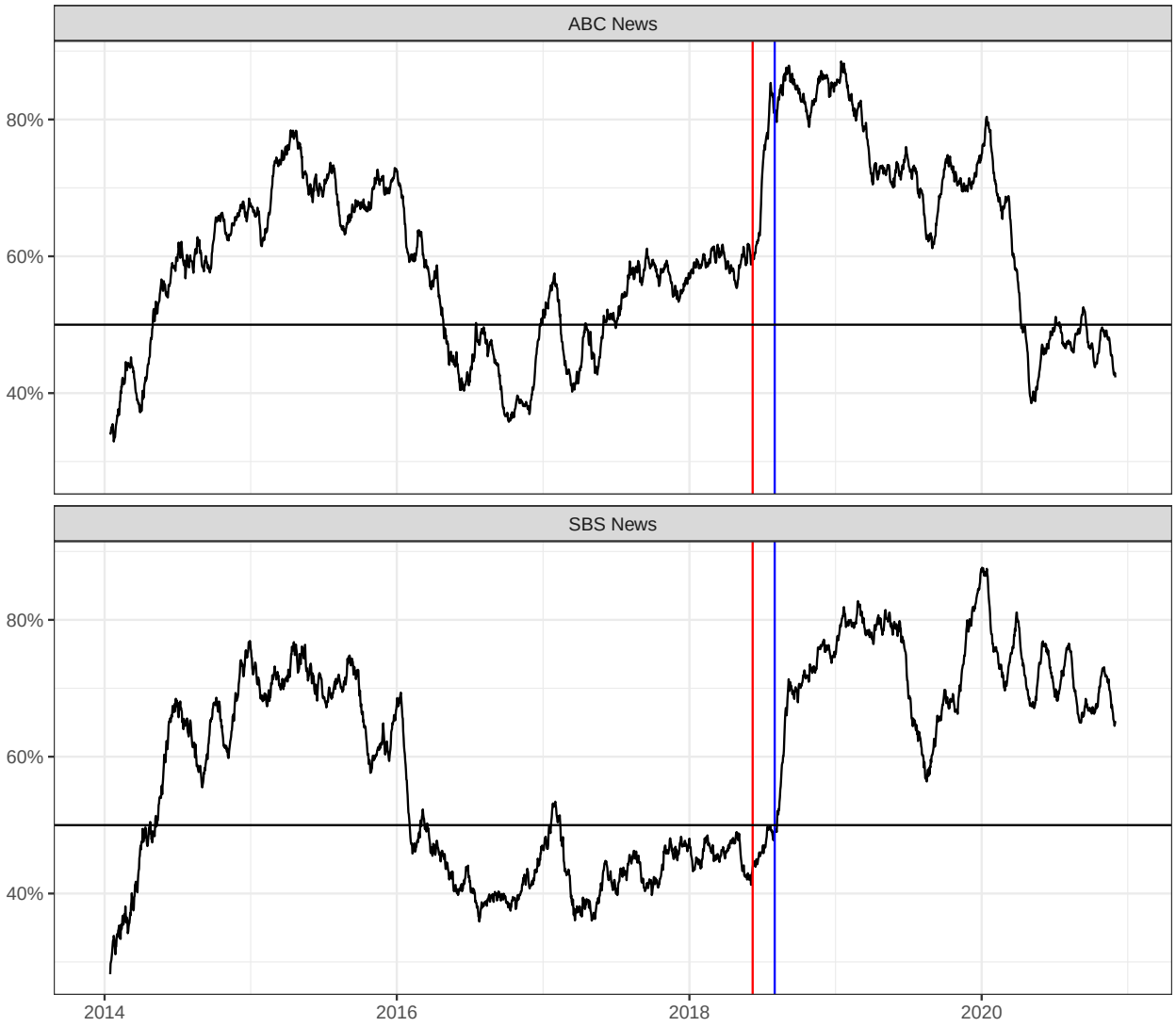


Figure 3: Performance score on interactions

```
  geom_vline(xintercept = as.Date("2018-06-08"), colour = "red") +

  geom_vline(xintercept = as.Date("2018-08-02"), colour = "blue") +

  facet_wrap(Page.Name ~., ncol = 1) +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = .5) +
  labs(x = NULL, y = NULL)
```

```
tmp <-
  dat %>%
  dplyr::filter(Page.Name == "ABC News" &
                  posix >= as.Date("2018-06-08") - 40  &
                  posix <= as.Date("2018-06-08") + 40) %>%
  dplyr::mutate(Overperforming.Score = as.numeric(Overperforming.Score)) %>%
  dplyr::select(Page.Name, URL, Type, Message, Overperforming.Score, Total.Interactions)

WriteXLS::WriteXLS(tmp, ExcelFileName = "abcnews_postings.xls")

tmp <-
  dat %>%
  dplyr::filter(Page.Name == "SBS News" &
                  posix >= as.Date("2018-08-02") - 40  &
                  posix <= as.Date("2018-08-02") + 40) %>%
  dplyr::mutate(Overperforming.Score = as.numeric(Overperforming.Score)) %>%
  dplyr::select(Page.Name, URL, Type, Message, Overperforming.Score, Total.Interactions)
```



Proportion of links above the early-2017 benchmark

Table 3

day	Performance.Score
2014-12-31	1.8025081
2015-12-31	1.4974485
2016-12-31	0.9923793
2017-12-31	0.8503270
2018-12-31	1.0520781
2019-12-31	1.4134448
2020-10-01	0.6949371

```
WriteXLS::WriteXLS(tmp, ExcelFileName = "sbsnews_postings.xls")
```

Industry-wide (weighted average)

```
interactions_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
                                   "2020-10-01"))) %>%
  dplyr::arrange(day) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
interactions_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score)) +
  geom_line() +
  geom_smooth() +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL)
```

Comments

```
{
  sink("/dev/null");
  cl <- makeCluster(10)
  clusterEvalQ(cl, {
    library(dplyr)
```

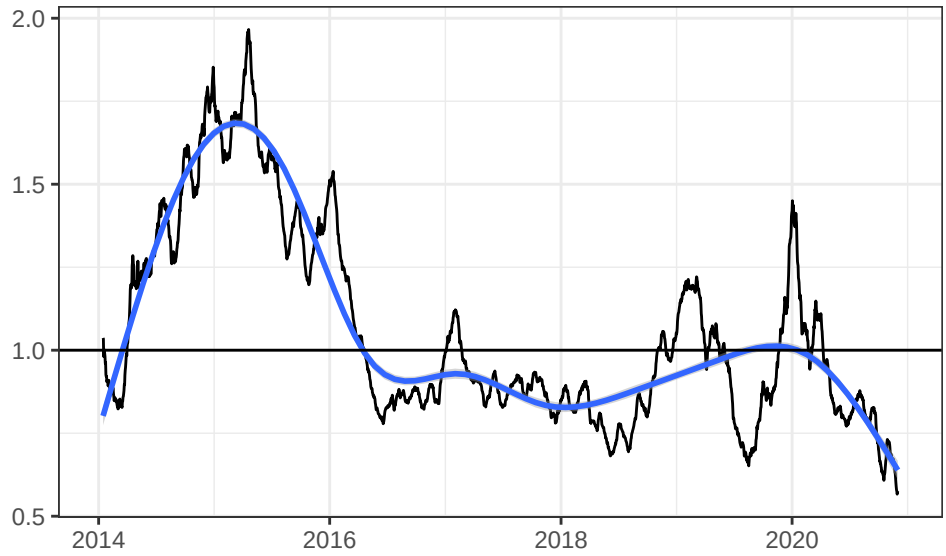


Figure 2: Performance score on interactions

```
    library(magrittr)
  })
}


clusterExport(cl, "benchmarking")

benchmark <-
  dat %>%
  dplyr::filter(posix >= benchmarking_timeframe[1] &
                  posix <= benchmarking_timeframe[2]) %>%
  dplyr::group_by(Page.Name, Type) %>%
  dplyr::summarise(benchmark.score =
                     benchmarking(Comments))

dat_wt_benchmark <-
  merge(dat, benchmark,
        by = c("Page.Name","Type"))

this_seq <- seq(date_limits[1],
                date_limits[2], by = "day")
clusterExport(cl, "this_seq")

dat_wt_benchmark <-
  dat_wt_benchmark %>%
  dplyr::filter(Type %in% c("Link")) %>%
  dplyr::mutate(day = as.Date(posix)) %>%
  dplyr::select(Page.Name, page_type, Comments, benchmark.score, day)

clusterExport(cl, "dat_wt_benchmark")

par_res <-
```

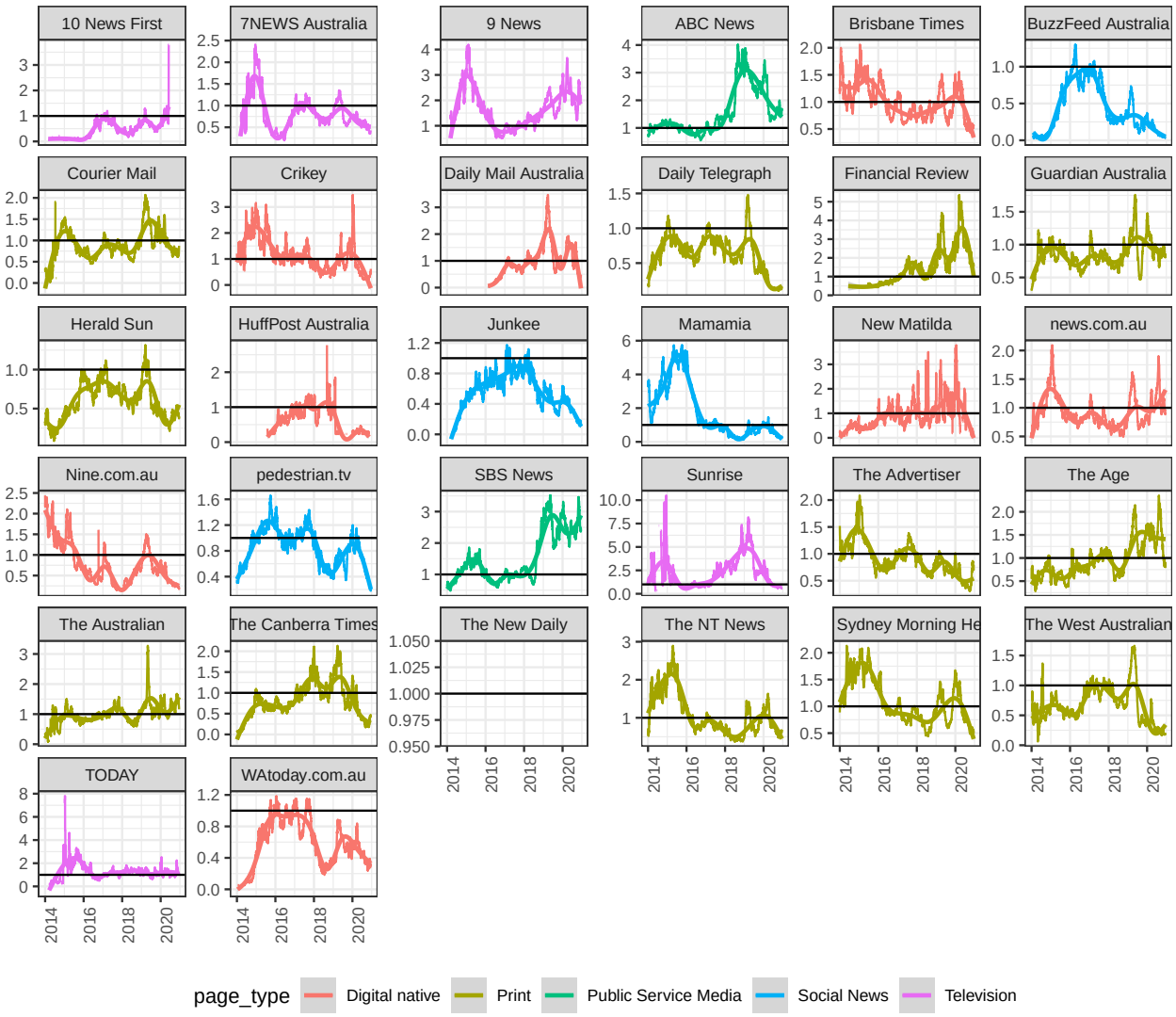
```
  parLapply(cl, 1:length(this_seq), fun = computePerfScoreComments)

comments_ma30_dat <-
  bind_rows(par_res)

stopCluster(cl)

save(comments_ma30_dat, file = "comments_ma30_dat.RData")
```

```
load("comments_ma30_dat.RData")

comments_ma30_dat %>%
  ggplot(aes(x = day, y = Performance.Score, colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(Page.Name~., scales = "free_y", ncol = 6) +
  geom_hline(yintercept = 1) +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=90, hjust=1),
        legend.position = 'bottom') +
  labs(x = NULL, y = NULL)
```

Clustered (weighted average)

```
comments_ma30_dat <-
  merge(comments_ma30_dat,
        weighting_by_type_on_links.df %>%
          dplyr::select(Page.Name, day, global_weight, page_type_weight),
        by = c("Page.Name", "day"))

comments_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
                                   "2020-10-01"))) %>%
  dplyr::arrange(page_type, day) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

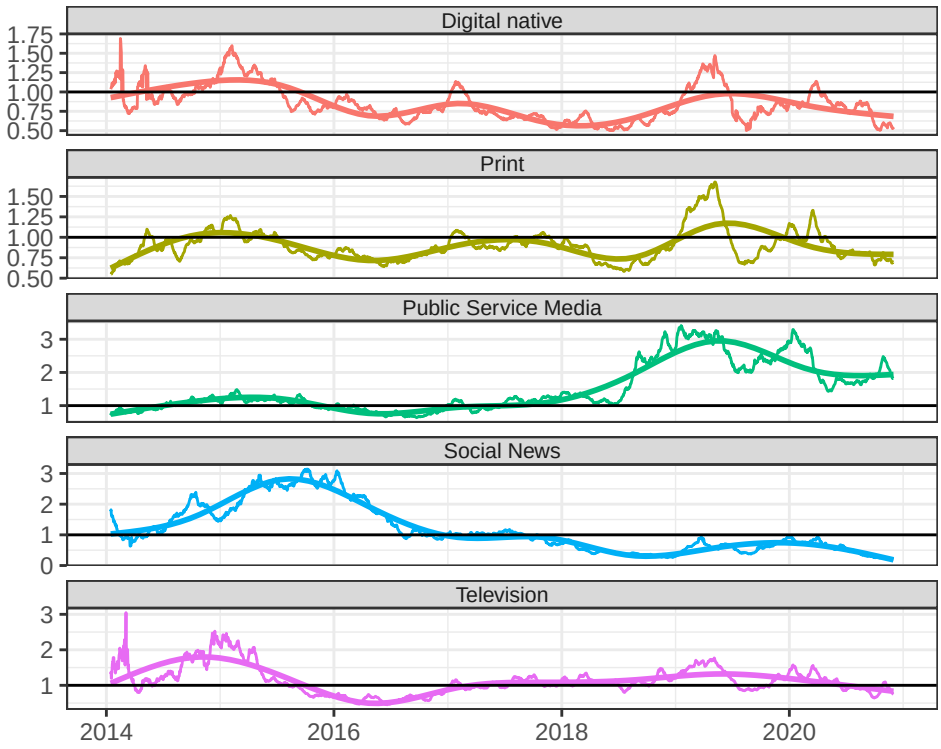
```
comments_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score,
```



Performance score on comments

Table 4

day	page_type	Performance.Score
2014-12-31	Digital native	1.2874698
2015-12-31	Digital native	0.8034407
2016-12-31	Digital native	0.8692248
2017-12-31	Digital native	0.5601729
2018-12-31	Digital native	0.7479741
2019-12-31	Digital native	0.8326415
2020-10-01	Digital native	0.5623352
2014-12-31	Print	1.0595244
2015-12-31	Print	0.7805958
2016-12-31	Print	0.8792900
2017-12-31	Print	0.9107715
2018-12-31	Print	0.9603575
2019-12-31	Print	1.1601157
2020-10-01	Print	0.6671005
2014-12-31	Public Service Media	1.1144453
2015-12-31	Public Service Media	0.9346946
2016-12-31	Public Service Media	1.0303060
2017-12-31	Public Service Media	1.3621467
2018-12-31	Public Service Media	2.7280356
2019-12-31	Public Service Media	2.9464965
2020-10-01	Public Service Media	1.8655316
2014-12-31	Social News	1.5217559
2015-12-31	Social News	2.7861317
2016-12-31	Social News	1.0284328
2017-12-31	Social News	0.7324497
2018-12-31	Social News	0.2922126
2019-12-31	Social News	0.9337616
2020-10-01	Social News	0.2738432
2014-12-31	Television	2.1111079
2015-12-31	Television	0.6731568
2016-12-31	Television	0.8871565
2017-12-31	Television	1.1222971
2018-12-31	Television	1.2290917
2019-12-31	Television	1.3684244
2020-10-01	Television	0.6553789



Performance score on comments

```
            colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(page_type~., ncol = 1, scales = "free_y") +
  theme(strip.text.x = element_text(size = 8, margin = margin(0.05,0,0.05,0, "cm"))) +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL, colour = "page type") +
  guides(colour = FALSE)
```

Industry-wide (weighted average)

```
comments_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
                                   "2020-10-01"))) %>%
  dplyr::arrange(day) %>%
  kable(booktabs = T, caption="") %>%
```

Table 5

day	Performance.Score
2014-12-31	1.2072661
2015-12-31	1.0286788
2016-12-31	0.9094740
2017-12-31	0.8634352
2018-12-31	0.9372990
2019-12-31	1.1928726
2020-10-01	0.7033748

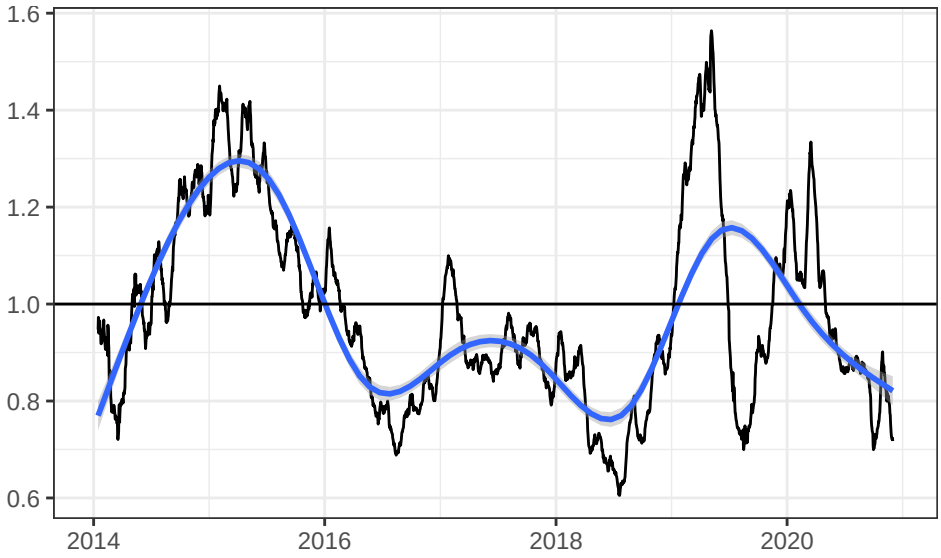


Figure 4: Performance score on comments

```
  kable_styling(latex_options = "striped")
```

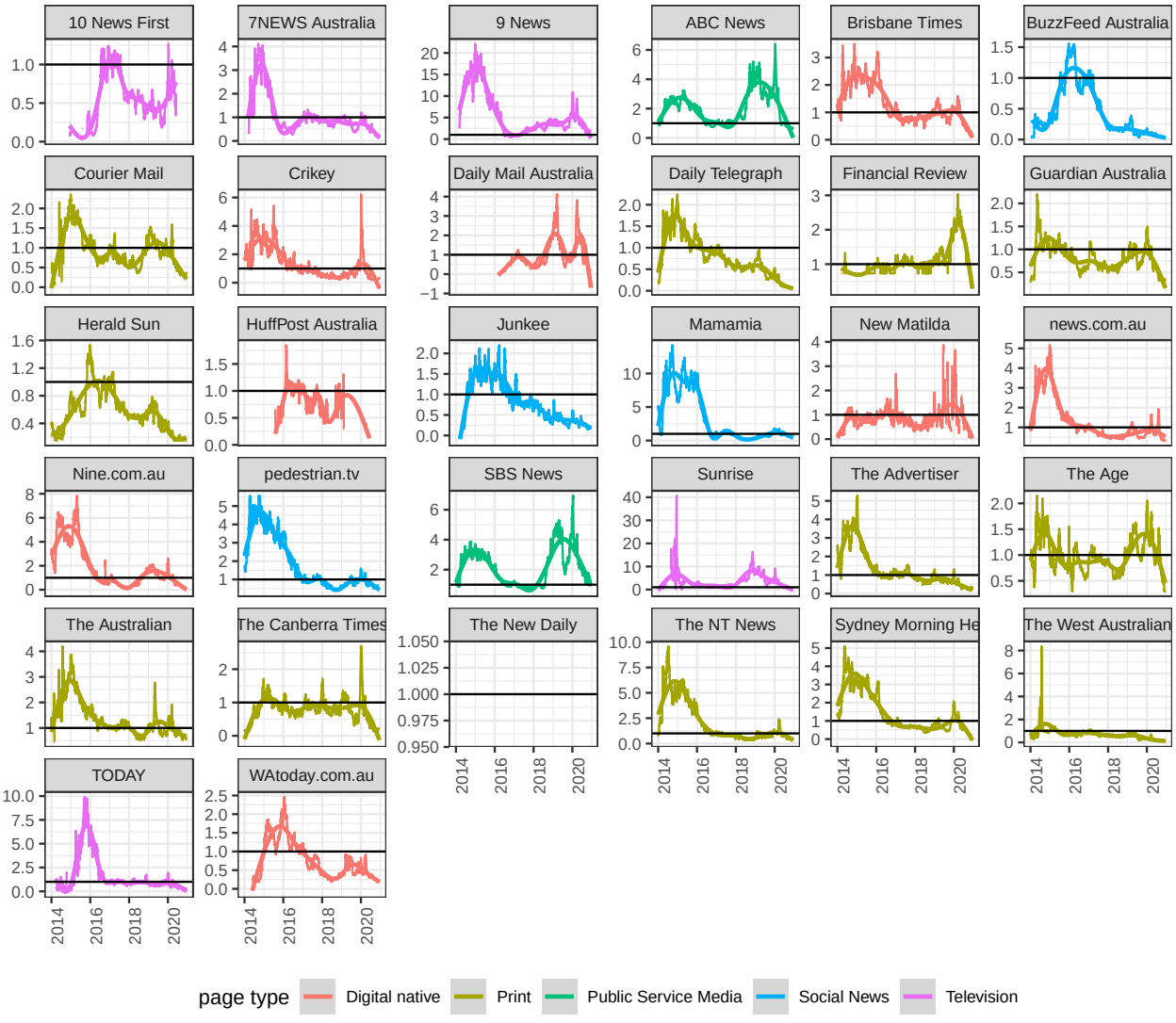
```
comments_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score)) +
  geom_line() +
  geom_smooth() +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL, colour = "page type")
```

Shares

```
{
  sink("/dev/null");
  cl <- makeCluster(10)
```

```
  clusterEvalQ(cl, {
    library(dplyr)
    library(magrittr)
  })
}

clusterExport(cl, "benchmarking")

benchmark <-
  dat %>%
  dplyr::filter(posix >= benchmarking_timeframe[1] &
                  posix <= benchmarking_timeframe[2]) %>%
  dplyr::group_by(Page.Name, Type) %>%
  dplyr::summarise(benchmark.score =
                     benchmarking(Shares))

dat_wt_benchmark <-
  merge(dat, benchmark,
        by = c("Page.Name","Type"))

this_seq <- seq(date_limits[1],
                date_limits[2], by = "day")
clusterExport(cl, "this_seq")

dat_wt_benchmark <-
  dat_wt_benchmark %>%
  dplyr::filter(Type %in% c("Link")) %>%
  dplyr::mutate(day = as.Date(posix)) %>%
  dplyr::select(Page.Name, page_type, Shares, benchmark.score, day)
clusterExport(cl, "dat_wt_benchmark")

par_res <-
  parLapply(cl, 1:length(this_seq), fun = computePerfScoreShares)

shares_ma30_dat <-
  bind_rows(par_res)

stopCluster(cl)

save(shares_ma30_dat, file = "shares_ma30_dat.RData")
```

```
load("shares_ma30_dat.RData")

shares_ma30_dat %>%
  ggplot(aes(x = day, y = Performance.Score, colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(Page.Name~., scales = "free_y", ncol = 6) +
  geom_hline(yintercept = 1) +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=90, hjust=1),
        legend.position = 'bottom') +
  labs(x = NULL, y = NULL, colour = "page type")
```



Performance score on shares

Clustered (weighted average)

```
shares_ma30_dat <-
  merge(shares_ma30_dat,
        weighting_by_type_on_links.df %>%
          dplyr::select(Page.Name, day, global_weight, page_type_weight),
        by = c("Page.Name", "day"))

shares_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
                                   "2020-10-01"))) %>%
  dplyr::arrange(page_type, day) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
shares_ma30_dat %>%
  dplyr::group_by(day, page_type) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = page_type_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score,
             colour = page_type)) +
  geom_line() +
  geom_smooth() +
  facet_wrap(page_type~., ncol = 1, scales = "free_y") +
  theme(strip.text.x = element_text(size = 8, margin = margin(0.05,0,0.05,0, "cm"))) +
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL, colour = "page type") +
  guides(colour = FALSE)
```

Industry-wide (weighted average)

```
shares_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  dplyr::filter(day %in% as.Date(c("2014-12-31",
                                   "2015-12-31",
                                   "2016-12-31",
                                   "2017-12-31",
                                   "2018-12-31",
                                   "2019-12-31",
```

Table 6

day	page_type	Performance.Score
2014-12-31	Digital native	3.0079323
2015-12-31	Digital native	1.6911738
2016-12-31	Digital native	1.0420439
2017-12-31	Digital native	0.4146908
2018-12-31	Digital native	0.9681289
2019-12-31	Digital native	1.1887792
2020-10-01	Digital native	0.3421648
2014-12-31	Print	2.1593141
2015-12-31	Print	1.4440760
2016-12-31	Print	0.9699189
2017-12-31	Print	0.7249104
2018-12-31	Print	0.8811932
2019-12-31	Print	1.2141308
2020-10-01	Print	0.3364081
2014-12-31	Public Service Media	2.5729321
2015-12-31	Public Service Media	2.2139192
2016-12-31	Public Service Media	1.0658887
2017-12-31	Public Service Media	0.9318365
2018-12-31	Public Service Media	3.5835385
2019-12-31	Public Service Media	5.6496360
2020-10-01	Public Service Media	0.8506018
2014-12-31	Social News	3.9779363
2015-12-31	Social News	5.5176709
2016-12-31	Social News	1.1361182
2017-12-31	Social News	0.6076637
2018-12-31	Social News	0.3724386
2019-12-31	Social News	1.1507439
2020-10-01	Social News	0.3098219
2014-12-31	Television	7.0739236
2015-12-31	Television	2.7157273
2016-12-31	Television	1.0282249
2017-12-31	Television	1.5291002
2018-12-31	Television	2.2719112
2019-12-31	Television	3.2606435
2020-10-01	Television	0.4670009

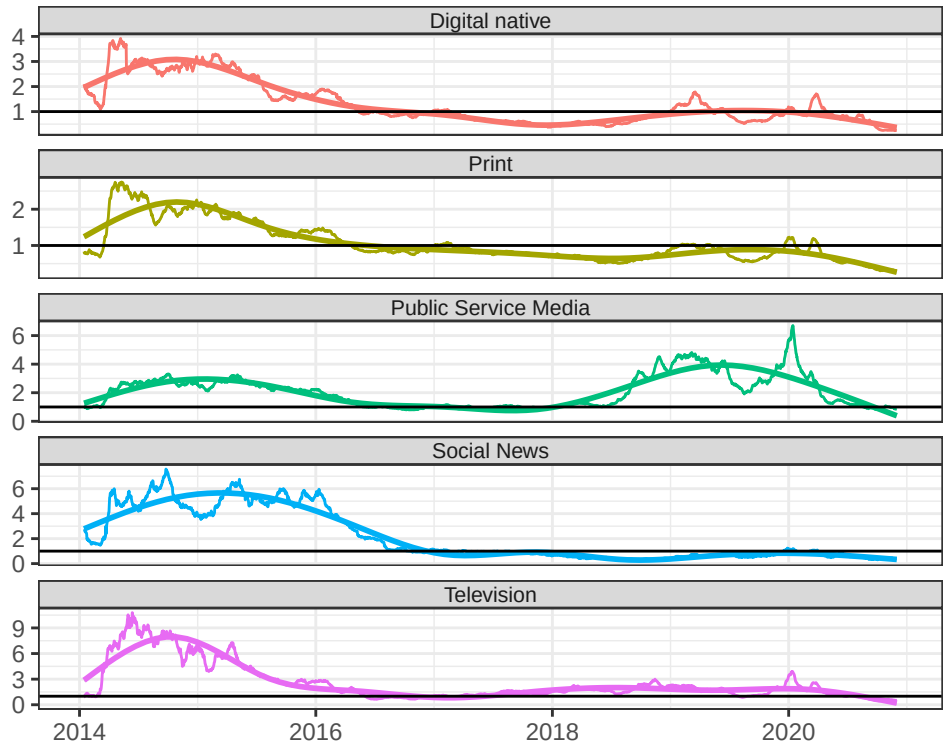


Figure 6: Performance score on shares

```
                                    "2020-10-01"))) %>%
  dplyr::arrange(day) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
shares_ma30_dat %>%
  dplyr::group_by(day) %>%
  dplyr::summarize(Performance.Score =
                     weighted.mean(Performance.Score,
                                   w = global_weight, na.rm = T)) %>%
  ggplot(aes(x = day,
             y = Performance.Score)) +
  geom_line() +
  geom_smooth() +
```

Table 7

day	Performance.Score
2014-12-31	2.7693053
2015-12-31	2.2390738
2016-12-31	1.0241962
2017-12-31	0.7659782
2018-12-31	1.2152879
2019-12-31	1.8064132
2020-10-01	0.4020031

Figure 5: Performance score on shares

Table 8

w.mean.2020	w.mean.2014	w.mean.2020/w.mean.2014
0.6733552	1.60851	0.4186205

```
  theme(legend.position = 'bottom') +
  geom_hline(yintercept = 1) +
  labs(x = NULL, y = NULL)
```

Stats

```
interactions_ma30_dat %>%
  dplyr::summarize(w.mean.2020 =
                     weighted.mean(Performance.Score[day == "2020-11-15"],
                                   w = global_weight[day == "2020-11-15"],
                                   na.rm = T),
                   w.mean.2014 = weighted.mean(Performance.Score[day == "2014-11-15"],
                                               w = global_weight[day == "2014-11-15"],
                                               na.rm = T)) %>%
  dplyr::mutate(w.mean.2020 / w.mean.2014) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
comments_ma30_dat %>%
  dplyr::summarize(w.mean.2020 =
                     weighted.mean(Performance.Score[day == "2020-11-15"],
                                   w = global_weight[day == "2020-11-15"],
                                   na.rm = T),
                   w.mean.2014 = weighted.mean(Performance.Score[day == "2014-11-15"],
                                               w = global_weight[day == "2014-11-15"],
                                               na.rm = T)) %>%
```

Table 9

w.mean.2020	w.mean.2014	w.mean.2020/w.mean.2014
0.8117675	1.261284	0.6436042

Table 10

w.mean.2020	w.mean.2014	w.mean.2020/w.mean.2014
0.4186723	2.741994	0.152689

```
  dplyr::mutate(w.mean.2020 / w.mean.2014)  %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
shares_ma30_dat %>%
  dplyr::summarize(w.mean.2020 =
                     weighted.mean(Performance.Score[day == "2020-11-15"],
                                   w = global_weight[day == "2020-11-15"],
                                   na.rm = T),
                   w.mean.2014 = weighted.mean(Performance.Score[day == "2014-11-15"],
                                               w = global_weight[day == "2014-11-15"],
                                               na.rm = T)) %>%
  dplyr::mutate(w.mean.2020 / w.mean.2014) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

```
shares_ma30_dat %>%
  dplyr::group_by(page_type) %>%
  dplyr::summarize(w.mean.2020 =
                     weighted.mean(Performance.Score[day == "2020-11-15"],
                                   w = page_type_weight[day == "2020-11-15"],
                                   na.rm = T),
                   w.mean.2014 = weighted.mean(Performance.Score[day == "2014-11-15"],
                                               w = page_type_weight[day == "2014-11-15"],
                                               na.rm = T)) %>%
  dplyr::mutate(w.mean.2020 / w.mean.2014) %>%
  kable(booktabs = T, caption="") %>%
  kable_styling(latex_options = "striped")
```

Table 11

page_type	w.mean.2020	w.mean.2014	w.mean.2020/w.mean.2014
Digital native	0.2546962	2.977737	0.0855335
Print	0.3207186	2.005623	0.1599098
Public Service Media	0.9590814	2.638820	0.3634508
Social News	0.4597690	4.732159	0.0971584
Television	0.5958861	4.491700	0.1326638