Cherry Blossom Prediction Competition

Sean Hardison

Kyoto, Japan

Environmental covariates

The environmental covariates I used in model development were drawn from the Japan Meteorological Agency (see (https://www.data.jma.go.jp/obd/stats/etrn/view/monthly_s3_en.php?block_no=47895&view=13)[https://www.data.jma.go.jp/obd/stats/etrn/view/monthly_s3_en.php?block_no=47895&view=13]).

```
ky_temps <-
  read_excel(here::here('data/kyoto_temps.xlsx')) %>%
  dplyr::select(year = Year,
                mar = Mar,
                feb = Feb,
                jmax = Jan_max,
                fmax = Feb_max,
                mmax = Mar_max,
                jpre = Jan_precip,
                fpre = Feb_precip,
                mpre = Mar_precip)
```

Data processing

```
# Identify the Kyoto data and draw a 200 km buffer around the site
ky_bf <- japan %>%
  filter(str_detect(location, "Kyoto")) %>%
  dplyr::select(long, lat) %>%
  distinct() %>%
  st_as_sf(.,coords = c("long","lat"),
           crs = 4326) %>%
  st_transform(st_crs("+proj=utm +zone=54 +datum=WGS84 +units=km +no_defs")) %>%
  st_buffer(.,dist = 200)

# Intersect the bloom data with the buffer to identify sites close to Kyoto
jp <-
  japan %>%
  mutate(bloom_date = as.Date(bloom_date)) %>%
  st_as_sf(.,coords = c("long","lat"),
           crs = 4326) %>%
  st_transform(st_crs("+proj=utm +zone=54 +datum=WGS84 +units=km +no_defs")) %>%
  st_intersection(.,ky_bf) %>%

  # devtools::install.github("seanhardison1/dream")
  dream::sfc_as_cols(names = c("longitude","latitude")) %>%
  st_set_geometry(NULL)
```

```r
# there are multiple bloom dates within years for distinct locations,
# so here I take the mean bloom date for these sites within years.
y <- 1950
jp_lats <- jp %>%
  filter(year >= y) %>%
  group_by(location) %>%
  summarise(longitude = mean(longitude),
            latitude = mean(latitude))

jp_sample <- jp %>%
  filter(year >= y) %>%
  group_by(location, bloom_date) %>%
  dplyr::summarise(bloom_doy = mean(bloom_doy, na.rm = T)) %>%
  left_join(.,jp_lats) %>%
  mutate(year = year(bloom_date)) %>%
  # turn into tsibble object and fill gaps
  tsibble(key = "location", index = "year") %>%
  fill_gaps()

# identify the sites with the longest running time series
sites <-
  jp_sample %>%
  as_tibble() %>%
  group_by(location) %>%
  dplyr::summarise(n = n()) %>%
  filter(n == 69) %>%
  pull(location)

# final data for use in model
jp_sample2 <- jp_sample %>%
  filter(location %in% sites) %>%
  dplyr::select(location, bloom_doy, year,
                latitude, longitude) %>%
  left_join(.,ky_temps)
```

Model fitting

Here I used a generalized additive model to evaluate how the date of first bloom varies within the region around Kyoto. I used a tensor product interaction term to incorporate data from nearby sites into bloom date prediction in Kyoto. The environmental covariates include February precipitation (`fpre`), February average daily maximum temperature (`fmax`), January average daily maximum temperature (`jmax`), and February daily mean temperature (`ftemp`).

```r
m <- gam(bloom_doy ~
           s(fpre) +
           s(fmax) +
           s(jmax) +
           s(feb) +
           te(longitude, latitude, year),
         data = jp_sample2)
```

Prediction data

I fitted the model using data through 2021. However, I assumed that the available data for February 2022 (through 2/26) would be representative of the month of February, and so used those data in the 2022 projection. I add those data to the prediction data here.

```
ndf <- tibble(longitude = 19.2106,
              latitude = 3887.377,
              year = 1953:2022,
              fpre = c(jp_sample2 %>%
                         filter(location == "Japan/Kyoto") %>% pull(fpre),17.0),
              fmax = c(jp_sample2 %>%
                         filter(location == "Japan/Kyoto") %>% pull(fmax),9.0),
              feb = c(jp_sample2 %>%
                        filter(location == "Japan/Kyoto") %>% pull(feb),4.2),
              jmax = c(jp_sample2 %>%
                         filter(location == "Japan/Kyoto") %>% pull(jmax),8.2))
```

Projecting environmental data

I used neural network autoregression to generate projections of environmental data, which I then bound to the prediction data created in the previous step. See https://otexts.com/fpp2/nnetar.html for description of the method.

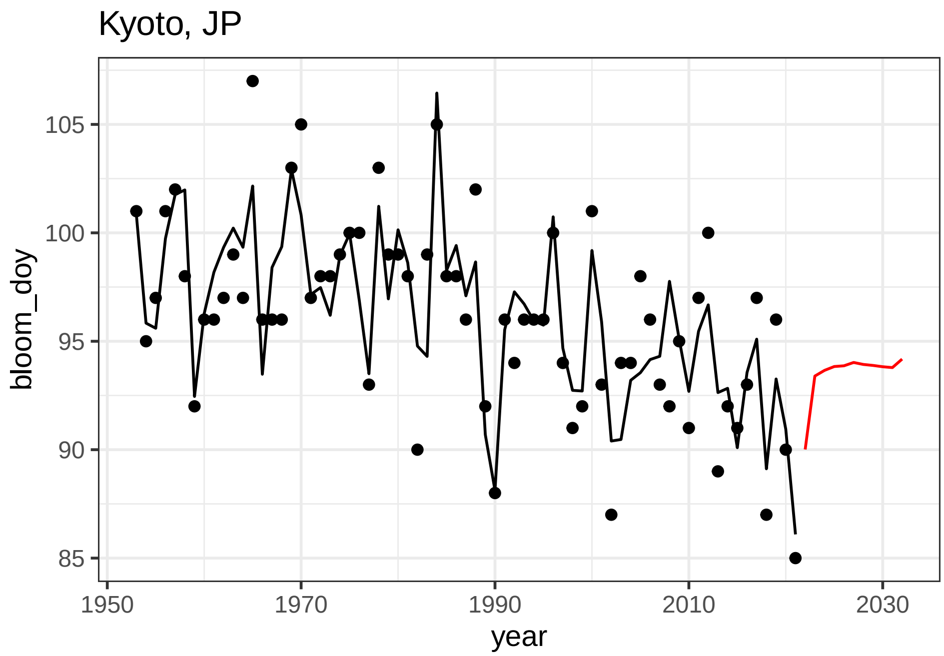
```
if (fc){
  base <- ndf %>% tsibble(index = "year")

  output_fpre <-
    base %>%
    model(
      fpre = NNETAR(fpre)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_fmax <-
    base %>%
    model(
      fmax = NNETAR(fmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_feb <-
    base %>%
    model(
      feb = NNETAR(feb)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_jmax <-
    base %>%
    model(
      jmax = NNETAR(jmax)
```

```r
    ) %>%
    forecast(h = 10) %>%
    tibble()

  jp_proj <-
    output_fpre %>%
    dplyr::select(year, fpre = .mean) %>%
    left_join(.,output_fmax %>%
                dplyr::select(year, fmax = .mean)) %>%
    left_join(.,output_feb %>%
                dplyr::select(year, feb = .mean)) %>%
    left_join(.,output_jmax %>%
                dplyr::select(year, jmax = .mean))

  save(jp_proj, file = here::here("data/jp_env_fc.rdata"))
} else {
  load(here::here("data/jp_env_fc.rdata"))
}
```

Projection

Here I predict from the GAM given observed and projected environmental covariates.

```r
# bind prediction and projection data.frames
ndf2 <- ndf %>%
  bind_rows(
    jp_proj %>%
      mutate(latitude = unique(ndf$latitude),
             longitude = unique(ndf$longitude)))

# project from GAM
pred <-
  predict(m, newdata = ndf2, se.fit = T)
pred_df_jp <- tibble(bloom_doy = pred$fit,
                  year = ndf2$year)

# observed data from Kyoto
jp_obs <-
  jp_sample2 %>%
  filter(str_detect(location, "Kyoto"))

ggplot() +
  geom_point(data = jp_obs,
             aes(x = year, y = bloom_doy)) +
  geom_line(data = pred_df_jp %>%
              filter(year <= 2021), aes(x = year, y = bloom_doy)) +
  geom_line(data = pred_df_jp %>%
            filter(year > 2021), aes(x = year, y = bloom_doy),
            color = "red") +
  labs(title = "Kyoto, JP") +
  theme_bw()
```



Vancouver, British Columbia

I built my model for Vancouver, BC around data collected through the VCBF Neighborhood Blog (https://forums.botanicalgarden.ubc.ca/threads/kerrisdale.36008/). I used the "date of first post" as a proxy for the date of first bloom.

Environmental covariates

Even though the temporal extent of the bloom data were limited (2008-2021), I still used environmental covariates as predictors in the model. This code queries the Global Historical Climatology Network for Vancouver monthly weather data.

```
# query raw
if (fc){
  bc_temps <- ghcnd(stationid = "CA001108395", refresh = TRUE)
  save(bc_temps, file = here::here("data/bc_temps_raw.rdata"))
} else {
  load(here::here("data/bc_temps_raw.rdata"))
}

bc_temps2 <-
  bc_temps  %>%
  dplyr::select_at(vars(year, month, element, contains("VALUE"))) %>%
  rowwise() %>%
  mutate(val = mean(c_across(VALUE1:VALUE31), na.rm = T)) %>%
  ungroup() %>%
  dplyr::select(year, month, element, val) %>%
  spread(.,element, val) %>%
  dplyr::filter(month %in% 1:3) %>%
  dplyr::select(tmax = TMAX,
                tmin = TMIN,
                temp = TAVG,
```

```r
                precip = PRCP,
                year, month)

jan <- bc_temps2 %>% filter(month == 1) %>%
  rename_at(vars(1:4), function(x)paste0("j_",x)) %>%
  dplyr::select(-month)
feb <- bc_temps2 %>% filter(month == 2) %>%
  rename_at(vars(1:4), function(x)paste0("f_",x)) %>%
  dplyr::select(-month)
mar <- bc_temps2 %>% filter(month == 3) %>%
  rename_at(vars(1:4), function(x)paste0("m_",x)) %>%
  dplyr::select(-month)
bc_temps3 <- jan %>%
  left_join(.,feb, by = c("year")) %>%
  left_join(.,mar, by = c("year"))

# Bind the data with the BC bloom data drawn from the neighborhood forum
bc_blooms <- read_excel(here::here("data/bc_blooms.xlsx")) %>%
  mutate(bloom_doy = yday(date)) %>%
  left_join(.,bc_temps3)
```

Model fitting

I again chose a GAM with average daily max temperatures in February and March included as model covariates.

```r
m <- gam(bloom_doy ~ s(f_tmax, k = 3) + s(m_tmax),
              data = bc_blooms)
```

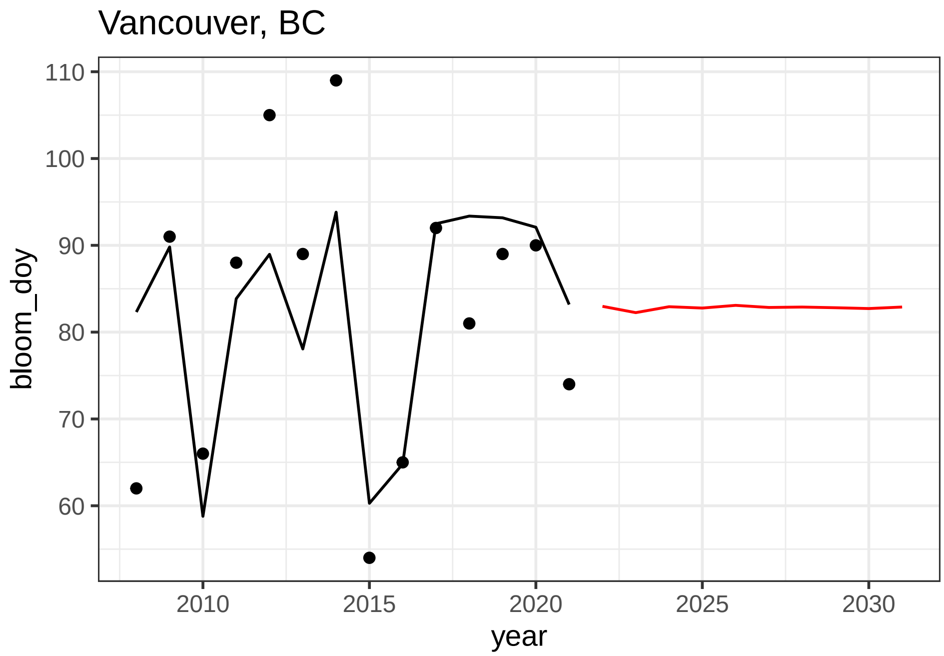
Projecting environmental data

```r
# need to turn data into tsibble first
base <- bc_blooms %>%
  tsibble(index = "year")

if (fc){
  output_m_tmax <-
    base %>%
    model(
      j_tmax = NNETAR(m_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_f_tmax <-
    base %>%
    model(
      f_tmax = NNETAR(f_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  bc_proj <-
    output_m_tmax %>%
```

```
    dplyr::select(year, m_tmax = .mean) %>%
    left_join(.,output_f_tmax %>%
                dplyr::select(year, f_tmax = .mean))
  save(bc_proj, file = here::here("data/bc_env_fc.rdata"))
} else {
  load(here::here("data/bc_env_fc.rdata"))
}
```

Projection

```
ndf <- bc_blooms %>%
  dplyr::select(m_tmax, f_tmax, year) %>%
  bind_rows(.,bc_proj)

pred <- predict(m, newdata = ndf)
pred_df_bc <- tibble(bloom_doy = pred,
                  year = ndf$year)

ggplot() +
  geom_point(data = bc_blooms, aes(y = bloom_doy, x = year)) +
  geom_line(data = pred_df_bc %>% filter(year <= 2021),
            aes(y = bloom_doy, x = year)) +
  geom_line(data = pred_df_bc %>% filter(year > 2021),
            aes(y = bloom_doy, x = year), color = "red") +
  labs(title = "Vancouver, BC") +
  theme_bw()
```



Washington, DC

Environmental covariates

I used the same process as the previous analysis to query environmental data for Washington, DC

```
# query raw
if (fc){
  dc_temps <- ghcnd(stationid = "USC00186350", refresh = TRUE)
  save(dc_temps, file = here::here("data/dc_temps_raw.rdata"))
} else {
  load(here::here("data/dc_temps_raw.rdata"))
}

dc_temps2 <-
  dc_temps  %>%
  dplyr::select_at(vars(year, month, element, contains("VALUE"))) %>%
  rowwise() %>%
  mutate(val = mean(c_across(VALUE1:VALUE31), na.rm = T)) %>%
  ungroup() %>%
  dplyr::select(year, month, element, val) %>%
  spread(.,element, val) %>%
  dplyr::filter(month %in% 1:3) %>%
  dplyr::select(tmax = TMAX,
                tmin = TMIN,
                temp = TOBS,
                precip = PRCP,
                year, month)

jan <- dc_temps2 %>% filter(month == 1) %>%
  rename_at(vars(1:4), function(x)paste0("j_",x)) %>%
  dplyr::select(-month)
feb <- dc_temps2 %>% filter(month == 2) %>%
  rename_at(vars(1:4), function(x)paste0("f_",x)) %>%
  dplyr::select(-month)
mar <- dc_temps2 %>% filter(month == 3) %>%
  rename_at(vars(1:4), function(x)paste0("m_",x)) %>%
  dplyr::select(-month)
dc_temps3 <- jan %>%
  left_join(.,feb, by = c("year")) %>%
  left_join(.,mar, by = c("year"))
```

Projecting environmental data

I included average maximum daily temperatures for both February and March in the model. Similar to my approach for the Kyoto model, I assumed that the data currently available for February was representative of the entire month and used that in the model. However, I still needed to project March data for the 2022 prediction, and February for >2022, so I did that here first. In the final projection, I replaced the projected data for February 2022 with the "true" value (through 2/26).

```
# project environmental data----
base <- dc_temps3 %>%
  tsibble(index = "year") %>%
  fill_gaps()

if (fc){
```

```
  output_mtmax <-
    base %>%
    model(
      m_tmax = NNETAR(m_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_ftmax <-
    base %>%
    model(
      f_tmax = NNETAR(f_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()
  save(output_ftmax, output_mtmax, file = here::here("data/dc_env_fc.rdata"))
} else {
  load(here::here("data/dc_env_fc.rdata"))
}

# tidy format
proj_df <-
  output_ftmax %>%
  dplyr::select(year, f_tmax = .mean) %>%
  left_join(.,output_mtmax %>%
              dplyr::select(year, m_tmax = .mean))
```

Model fitting

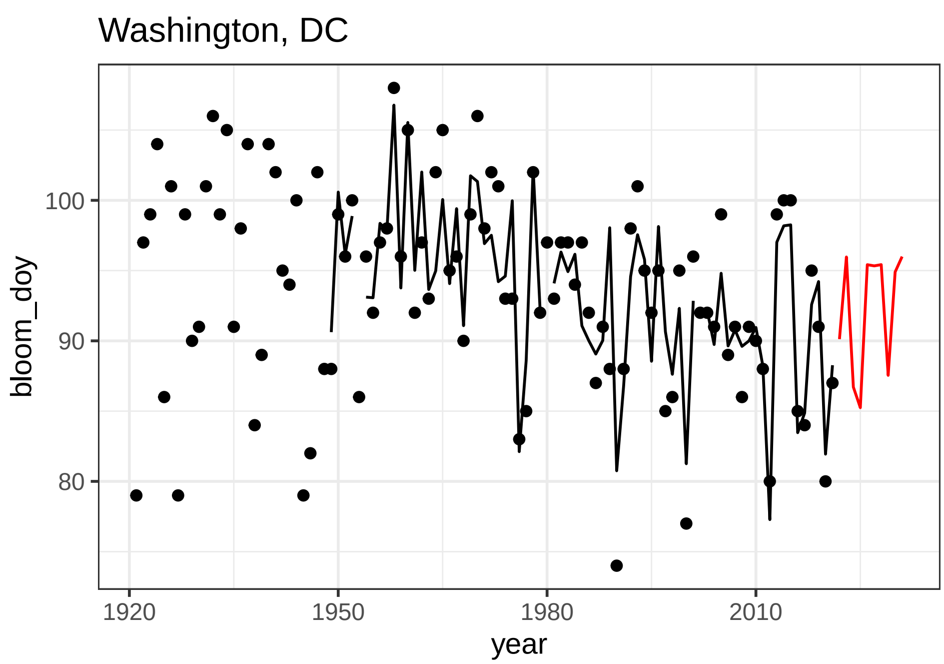
There was a strong temporal trend component in the data so I included a smoother for `year`, along with smooths with average maximum daily temperatures in March and February.

```
dc_sample <- dc %>%
  left_join(.,dc_temps3) %>%
  tsibble(index = "year")

m <- gam(bloom_doy ~
           s(year) +
           s(m_tmax) +
           s(f_tmax),
    data = dc_sample)
```

Projection

```
ndf <-
  tibble(f_tmax = dc_sample$f_tmax,
         m_tmax = dc_sample$m_tmax,
         bloom_doy = dc_sample$bloom_doy,
         year = dc_sample$year)  %>%
  dplyr::select(year, m_tmax, f_tmax) %>%
  bind_rows(.,proj_df) %>%
  # enter f_tmax for most of feb 2022
  mutate(f_tmax = ifelse(year == 2022, 111.5384616, f_tmax))
```

```r
pred <- predict(m, newdata = ndf, se.fit = T)
pred_df_dc <- tibble(fit = pred$fit,
                     se = pred$se.fit,
                     year = ndf$year)

ggplot() +
  geom_point(data = dc_sample, aes(y = bloom_doy, x = year)) +
  geom_line(data = pred_df_dc %>% filter(year <= 2021), aes(y = fit, x = year))+
  geom_line(data = pred_df_dc %>% filter(year > 2021), aes(y = fit, x = year),
            color = "red") +
  labs(title = "Washington, DC") +
  theme_bw()
```



Liestal, CH

Environmental covariates

I used environmental data from Basel, Switzerland in model building.

```r
bas_temps <-
  read_excel(here::here("data/basel_temps2.xlsx")) %>%
  dplyr::rename(temp = 2,
                gdd = 3,
                precip = 4,
                soil_moist = 5,
                wind_speed = 5,
                wind_dir = 6)

y <- str_sub(bas_temps$timestamp,1,4)
m <- str_sub(bas_temps$timestamp,5,6)
d <- str_sub(bas_temps$timestamp,7,8)
hr <- str_sub(bas_temps$timestamp,9,13)
```

```r
bas_temps2 <- bas_temps %>%
  mutate(date = as.Date(paste(y, m, d, sep = "-")))

mmtemps <- bas_temps2 %>%
  group_by(date) %>%
  dplyr::summarise(tmax = max(temp),
                   tmin = min(temp)) %>%
  group_by(ymon = yearmonth(date)) %>%
  dplyr::summarise(m_tmax = (mean(tmax) - 32)*(5/9),
                   m_tmin = (mean(tmin) - 32)*(5/9))

bas_temps3 <- bas_temps2 %>%
  group_by(ymon = yearmonth(date)) %>%
  dplyr::summarise(m_temp = mean(temp),
                   m_precip = mean(precip)) %>%
  left_join(.,mmtemps) %>%
  filter(month(ymon) %in% 1:3) %>%
  mutate(year = year(ymon),
         month = month(ymon)) %>%
  dplyr::select(-ymon)

jan <- bas_temps3 %>%
  filter(month == 1) %>%
  rename_at(vars(1:4), function(x)paste0("j_",x)) %>%
  dplyr::select(-month)

feb <- bas_temps3 %>%
  filter(month == 2) %>%
  rename_at(vars(1:4), function(x)paste0("f_",x)) %>%
  dplyr::select(-month)

mar <- bas_temps3 %>%
  filter(month == 3) %>%
  rename_at(vars(1:4), function(x)paste0("m_",x)) %>%
  dplyr::select(-month)

bas_temps4 <- left_join(feb,
                        jan) %>%
  left_join(.,mar)
```

Data processing

```r
# sw first
sw <-
  meteoswiss %>%
  mutate(bloom_date = as.Date(bloom_date)) %>%
  st_as_sf(.,coords = c("long","lat"),
           crs = 4326) %>%
  st_transform(st_crs("+proj=utm +zone=32 +datum=WGS84 +units=km +no_defs")) %>%
  dream::sfc_as_cols(names = c("longitude","latitude")) %>%
  st_set_geometry(NULL)

y <- 1984
```

```r
sw_lats <- sw %>%
  filter(year >= y) %>%
  group_by(location) %>%
  summarise(longitude = mean(longitude),
            latitude = mean(latitude))

sw_sample <- sw %>%
  filter(year >= y) %>%
  group_by(location, bloom_date) %>%
  dplyr::summarise(bloom_doy = mean(bloom_doy, na.rm = T)) %>%
  left_join(.,sw_lats) %>%
  mutate(year = year(bloom_date))

sw_sample2 <- sw_sample %>%
  dplyr::select(location, bloom_doy, year,
                latitude, longitude) %>%
  tsibble(index = "year", key = "location") %>%
  fill_gaps() %>%
  mutate(bloom_doy_l1 = lag(bloom_doy),
         bloom_doy_l2 = lag(bloom_doy, 2)) %>%
  left_join(.,bas_temps4) %>%
  na.exclude()
```

Model fitting

This model was unique in that that was a strong correlation structure in the residuals. I addressed this by including lag-1 and lag-2 versions of the bloom DOY variable as covariates. I also included a spatiotemporal smoother to incorporate data from nearby locations into the model, as well as environmental covariates.

```r
m <- gam(bloom_doy ~
           s(bloom_doy_l1) +
           s(bloom_doy_l2) +
           s(f_m_precip) +
           s(f_m_tmax) +
           s(j_m_tmax) +
           s(f_m_temp) +
           te(longitude, latitude, year),
         data = sw_sample2)
# acf(m$residuals)
# summary(m)
# appraise(m)
```

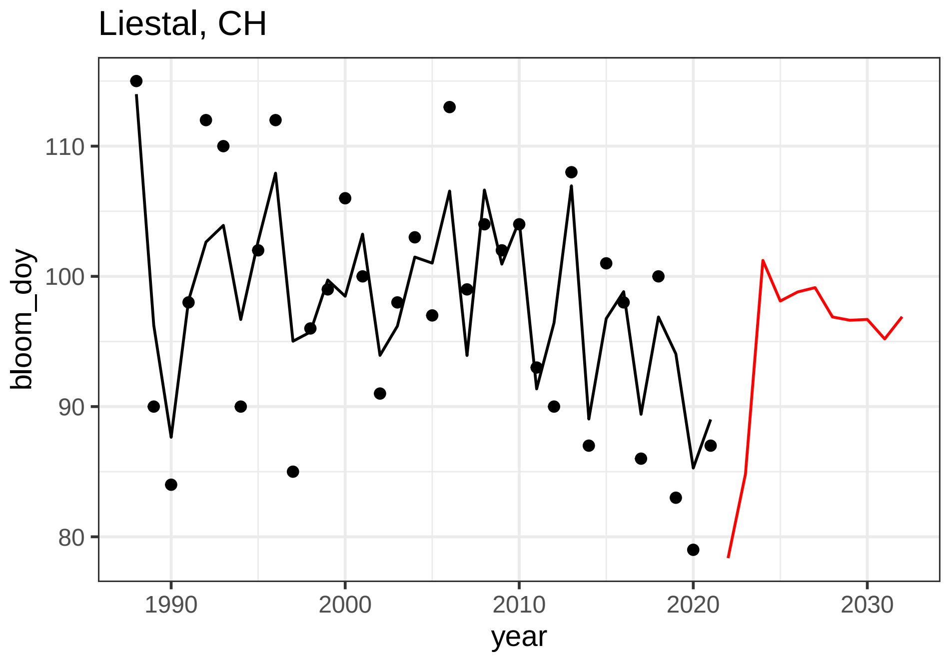
Projection

Projection from this model was tricky because of the lagged bloom variables. I addressed this by projecting the lagged bloom variable using neural network autoregression, and then lagging this term to generate a projected bloom date that could be used as a covariate for projection.

```r
y <- 1988
pred_df1 <- sw_sample2 %>%
  filter(location == "Switzerland/Liestal",
         year >= y)
prec_2022 <- bas_temps4 %>% filter(year == 2022) %>% pull(f_m_precip)
tmax_2022 <- bas_temps4 %>% filter(year == 2022) %>% pull(f_m_tmax)
temp_2022 <- bas_temps4 %>% filter(year == 2022) %>% pull(f_m_temp)
```

```r
jmax_2022 <- bas_temps4 %>% filter(year == 2022) %>% pull(j_m_tmax)

# prediction data through 2022 given that those data are mostly available
ndf <- tibble(longitude = 404.3579,
              latitude = 5259.444,
              year = y:2022,
              bloom_doy_l1 = c(pred_df1 %>% pull(bloom_doy_l1), 87),
              bloom_doy_l2 = c(pred_df1 %>% pull(bloom_doy_l2), 79),
              f_m_precip = c(pred_df1 %>% pull(f_m_precip), prec_2022),
              f_m_tmax = c(pred_df1 %>% pull(f_m_tmax), tmax_2022),
              f_m_temp = c(pred_df1 %>% pull(f_m_temp), temp_2022),
              j_m_tmax = c(pred_df1 %>% pull(j_m_tmax), jmax_2022))

# projecting covariates beyond 2022
if (fc){
  base <- ndf %>% tsibble(index = "year")

  output_bdl1 <-
    base %>%
    model(
      bloom_doy_l1 = NNETAR(bloom_doy_l1)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_fmpr <-
    base %>%
    model(
      f_m_precip = NNETAR(f_m_precip)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_jmtmax <-
    base %>%
    model(
      j_m_tmax = NNETAR(j_m_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_fmtmax <-
    base %>%
    model(
      f_m_tmax = NNETAR(f_m_tmax)
    ) %>%
    forecast(h = 10) %>%
    tibble()

  output_fmtem <-
    base %>%
    model(
      f_m_temp = NNETAR(f_m_temp)
```

```
    ) %>%
    forecast(h = 10) %>%
    tibble()

  sw_proj <-
    output_bdl1 %>%
    dplyr::select(year, bloom_doy_l1 = .mean) %>%
    left_join(.,output_fmpr %>%
                dplyr::select(year, f_m_precip = .mean)) %>%
    left_join(.,output_jmtmax %>%
                dplyr::select(year, j_m_tmax = .mean)) %>%
    left_join(.,output_fmtmax %>%
                dplyr::select(year, f_m_tmax = .mean)) %>%
    left_join(.,output_fmtem %>%
                dplyr::select(year, f_m_temp = .mean))

  save(sw_proj, file = here::here("data/sw_env_fc.rdata"))
} else {
  load(here::here("data/sw_env_fc.rdata"))
}

ndf2 <- ndf %>%
  bind_rows(
  sw_proj %>%
    mutate(bloom_doy_l2 = lag(bloom_doy_l1),
         latitude = unique(ndf$latitude),
         longitude = unique(ndf$longitude))
  ) %>%
  # fixing the lag-2 projection term
  mutate(bloom_doy_l2 = ifelse(year == 2023, 87, bloom_doy_l2))

# do the prediction

pred <- predict(m, newdata = ndf2, se.fit = T)

pred_df_ch <- tibble(bloom_doy = pred$fit,
                  year = ndf2$year)

ggplot() +
  geom_point(data = pred_df1, aes(y = bloom_doy, x = year)) +
  geom_line(data = pred_df_ch %>% filter(year <= 2021),
          aes(x = year, y = bloom_doy)) +
  geom_line(data = pred_df_ch %>% filter(year > 2021),
          aes(x = year, y = bloom_doy), color = "red") +
  labs(title = "Liestal, CH") +
  theme_bw()
```



All projections

```r
dc <-
  pred_df_dc %>%
  filter(year > 2021) %>%
  pull(fit) %>%
  round()

ch <-
  pred_df_ch %>%
  filter(year > 2021, year < 2032) %>%
  pull(bloom_doy) %>%
  round()

bc <-
  pred_df_bc %>%
  filter(year > 2021, year < 2032) %>%
  pull(bloom_doy) %>%
  round()

jp <-
  pred_df_jp %>%
  filter(year > 2021, year < 2032) %>%
  pull(bloom_doy) %>%
  round()

submission <- tibble(year = 2022:2031,
                     kyoto = jp,
                     liestal = ch,
                     washingtondc = dc,
                     vancouver = bc)
```

```r
write.csv(submission,
          here::here("data/hardison_submission.csv"),
          row.names = F)
```