Step 01: Getting SNOTEL Data

Keith Jennings

Setting up R

First we need to load our packages. If you're missing any of the required ones, you have a couple install options. First, you can use the RStudio package interface for the most common, CRAN-accessible packages:

Or, you can run `install.packages` for whichever package(s) you need. Example: `install.packages("snotelr")`.

Let's load our required packages.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.3.0
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(snotelr)
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
theme_set(theme_cowplot()) # setting the plot theme makes for cleaner figures
library(knitr)
```

What is SNOTEL?

SNOTEL, for the uninitiated, is the Snow Telemetry network run by the US Department of Agriculture's Natural Resources Conservation Service (NRCS). The NRCS has sited these stations across the mountain west to better monitor snow water equivalent (SWE, the liquid depth of water contained in a snowpack if it were completely melted) and related climate variables such as air temperature and precipitation.

Although it has its shortcomings, SNOTEL is the premier US snow dataset. Researchers use it for model development and validation, trend detection, process-based studies, and myriad other purposes. Water managers and other operations professionals rely on these data for water supply forecasts, making them incredibly valuable to decision-making in the arid and semi-arid west.

Exploring snotelr

The snotelr package for R provides the functions we need to easily, reproducibly access SNOTEL data. First, let's see what stations we're working with by using the `snotel_info()` function.

```
# Access the snotel metadata and save it as a dataframe
info <- snotel_info()

# View the top 5 lines
info %>% head()
```

```
##   network state            site_name
## 1    SNTL    AK     elmendorf field
## 2    SNTL    UT           elk ridge
## 3    SNTL    AK              hoonah
## 4    SNTL    AK pilgrim hot springs
## 5    SNTL    AK          seven mile
## 6    SNTL    CA          lost lakes
##                                     description      start        end
## 1                Outlet Ship Creek (190204010404) 2024-10-01 2025-06-13
## 2                 Cottonwood Creek (140802010402) 2024-10-01 2025-06-13
## 3 Port Fredrick-Frontal Icy Strait (190102110906) 2023-10-01 2025-06-13
## 4    Paystreak Creek-Pilgrim River (190501050702) 2024-07-01 2025-06-13
## 5                 Outlet Ray River (190804040306) 2024-09-01 2025-06-13
## 6     Upper West Fork Carson River (160502010301) 2024-09-01 2025-06-13
##   latitude longitude elev        county site_id
## 1    61.25   -149.82   52     Anchorage    1332
## 2    37.82   -109.77 2603      San Juan    1323
## 3    58.12   -135.41  463 Hoonah-angoon    1318
## 4    65.09   -164.92    6          Nome    1327
## 5    65.94   -149.86  201 Yukon-koyukuk    1330
## 6    38.65   -119.95 2633        Alpine    1331
```

Here we see information such as the network name, state, site name, start and end date of measurements, lat-lon, elevation, and the unique site id. We will use this information to downselect sites for further processing.

One of my favorite SNOTEL sites is Niwot in Colorado. We can view its metadata by filtering the info dataframe to just the single entry that contains the word "niwot" in the `site_name` column.

```
info %>% filter(str_detect(site_name, "niwot"))
```

```
##   network state site_name                        description      start
## 1    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
##          end latitude longitude elev  county site_id
## 1 2025-06-13    40.04   -105.55 3030 Boulder     663
```

We see it's in Boulder County, Colorado at an elevation of 3030 m and its record runs from 1979-10-01 to present day.

We can next use the `snotel_download()` function to download all of Niwot's data.

```
# Name the site_id
nwt_id = 663

# Download to a dataframe
df <- snotel_download(
  site_id = nwt_id,
  internal = TRUE    # save the file to R workspace (can also be saved locally)
)
```

```
## Downloading site: niwot , with id: 663
```

A quick look at the data reveals the information snotelr can provide from the SNOTEL network:

```
df %>% head()
```

```
##   network state site_name                        description      start
## 1    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
## 2    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
## 3    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
## 4    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
## 5    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
## 6    SNTL    CO    niwot  North Boulder Creek (101900050401) 1979-10-01
##          end latitude longitude elev  county site_id       date
## 1 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-01
## 2 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-02
## 3 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-03
## 4 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-04
## 5 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-05
## 6 2025-06-13    40.04   -105.55 3030 Boulder     663 1980-10-06
##   snow_water_equivalent snow_depth precipitation_cumulative temperature_max
## 1                     0         NA                         0              NA
## 2                     0         NA                         0              NA
## 3                     0         NA                         0              NA
## 4                     0         NA                         0              NA
## 5                     0         NA                         0              NA
## 6                     0         NA                         0              NA
##   temperature_min temperature_mean precipitation
## 1              NA               NA             0
## 2              NA               NA             0
## 3              NA               NA             0
## 4              NA               NA             0
## 5              NA               NA             0
## 6              NA               NA             0
```

In this file we see a fair amount of redundant information from the site metadata (we'll strip this out later) along with new data such as `snow_water_equivalent`, `snow_depth`, `temperature_min/max/mean`, and `precipitation`. In the next section we'll start working with these data.

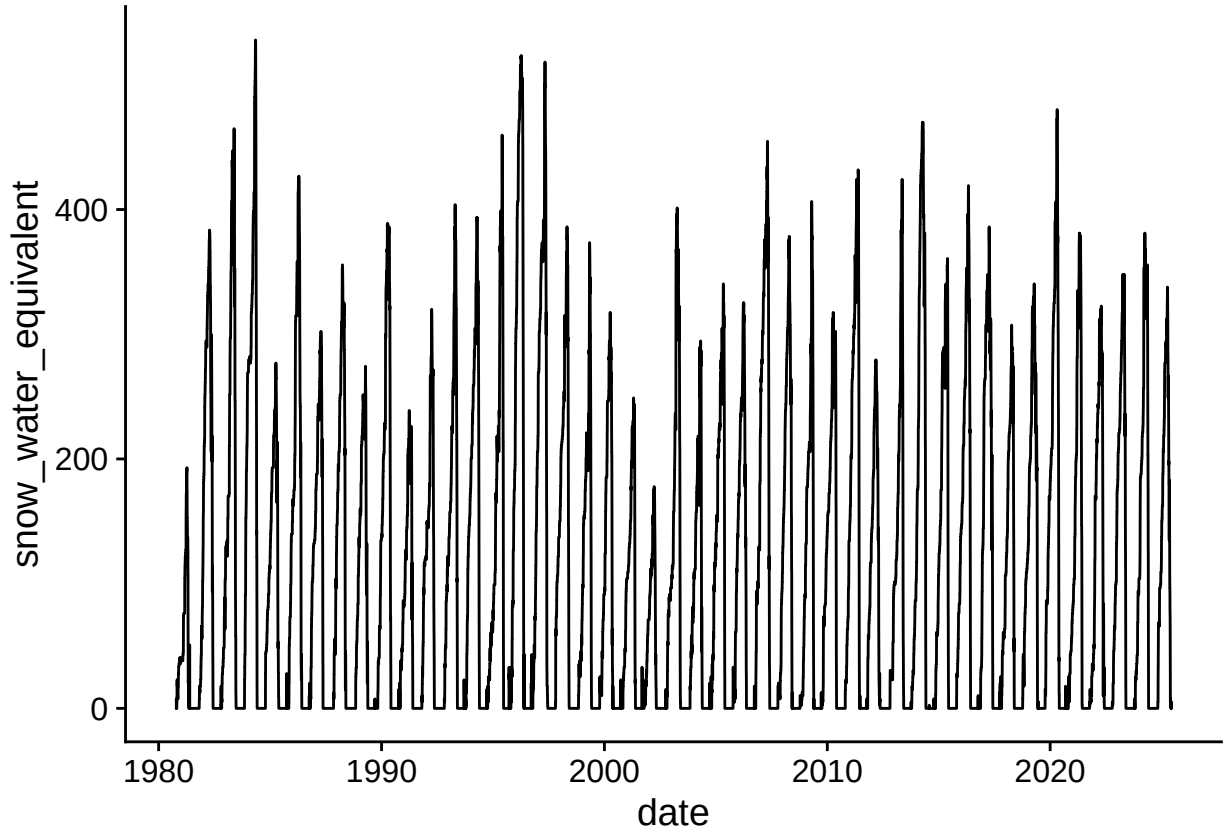
Processing and analyzing SNOTEL data

One of the first things we'll do is strip out the redundant information (keeping only the `site_id` column for later joining with the metadata) and process the date info.

```
# Select the columns we want
df <- df %>%
  select(site_id:precipitation)

# Format the date
df <- df %>%
  mutate(date = ymd(date))

# Create a water year function
# Requires the lubridate package loaded under tidyverse
wateryear <- function(date){
  ifelse(month(date) >= 10,
         year(date) + 1,
         year(date))
}

# Create a day of water year function
# Requires the lubridate package loaded under tidyverse
day_of_wateryear <- function(date){
  ifelse(year(date) %% 4 != 0,
         ifelse(month(date) >= 10,
                yday(date) - 273,
                yday(date) + 92),
         ifelse(month(date) >= 10,
                yday(date) - 274,
                yday(date) + 92))
}

# Add a water year column to the dataset
df <- df %>%
  mutate(wyear = wateryear(date),
         dowy = day_of_wateryear(date))
```

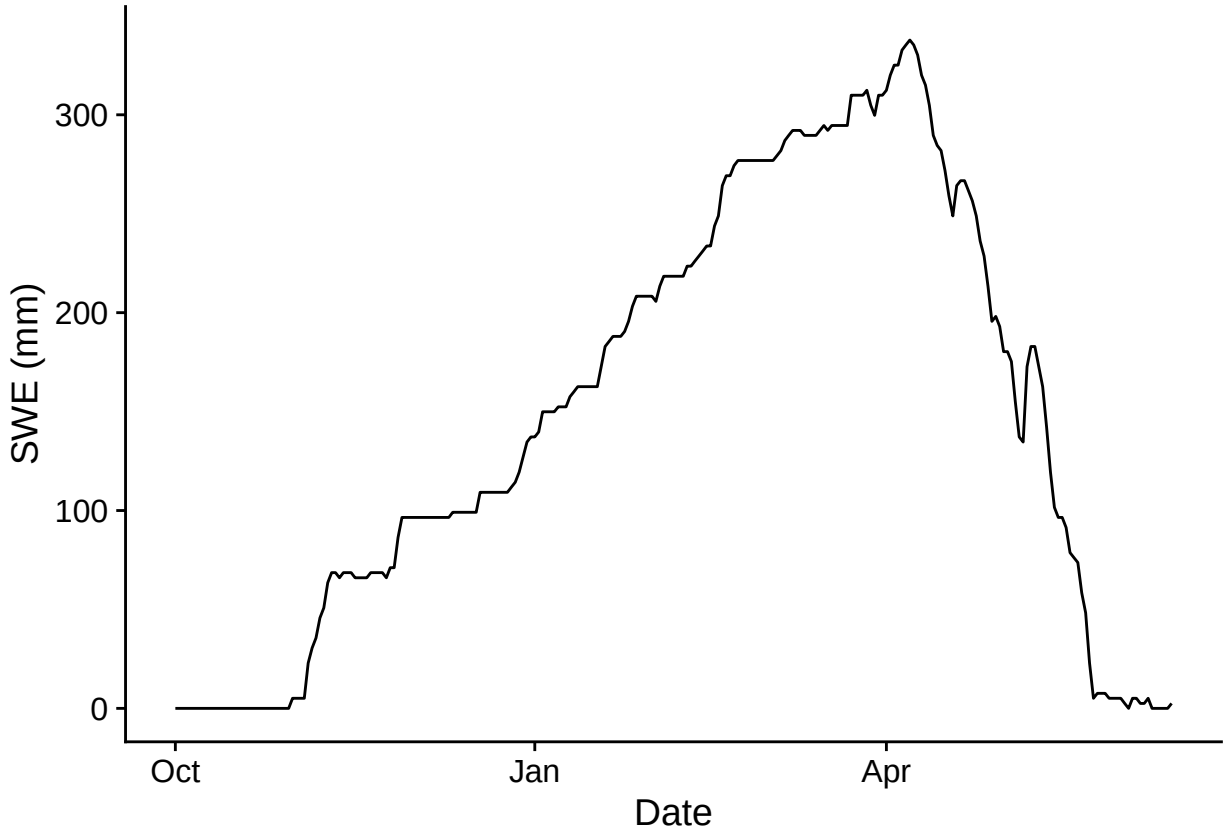
Now that we've formatted the date, we can plot SWE data.

```
ggplot(df, aes(date, snow_water_equivalent)) +
  geom_line()
```



Hmm, that's no good unless you like looking at squiggly migraine lines. Let's plot a single year of data instead and add some axes info.

```
plot_yr = 2025
df %>%
  filter(wyear == plot_yr) %>%
  ggplot(aes(date, snow_water_equivalent)) +
  geom_line() +
  labs(x = "Date",
       y = "SWE (mm)")
```



This is what snow hydrologists call a niveograph (like a hydrograph for snow) where the date is on the x-axis and SWE is on the y-axis.

A niveograph tells us all sorts of information about a snowpack.

The above plot from Jennings and Molotch (2019) shows a few simple metrics that we can compute from SWE data on a niveograph. A few others include the accumulation and melt seasons (pre- and post-peak SWE, respectively), snowmelt center of mass (when 50% of peak SWE has melted), and snow-on and snow-off dates (when the period of consistent, or continuous, snow cover begins and ends). We'll dive more into these metrics later.

Suffice it to say we will see inter-year differences. Take, for example, 2011 vs. 2012:

```
plot_yrs = c(2011,2012)
df %>%
  filter(wyear %in% plot_yrs) %>%
  ggplot(aes(dowy, snow_water_equivalent, color = as.factor(wyear))) +
  geom_line() +
  labs(x = "Day of Water Year",
       y = "SWE (mm)") +
  scale_color_manual(values = c("#0072B2", "#E69F00"),
                     name = "Water\nYear") +
  theme(legend.position = c(0.8, 0.8))
```

```
## Warning: A numeric ‘legend.position‘ argument in ‘theme()‘ was deprecated in ggplot2
## 3.5.0.
## i Please use the ‘legend.position.inside‘ argument of ‘theme()‘ instead.
```

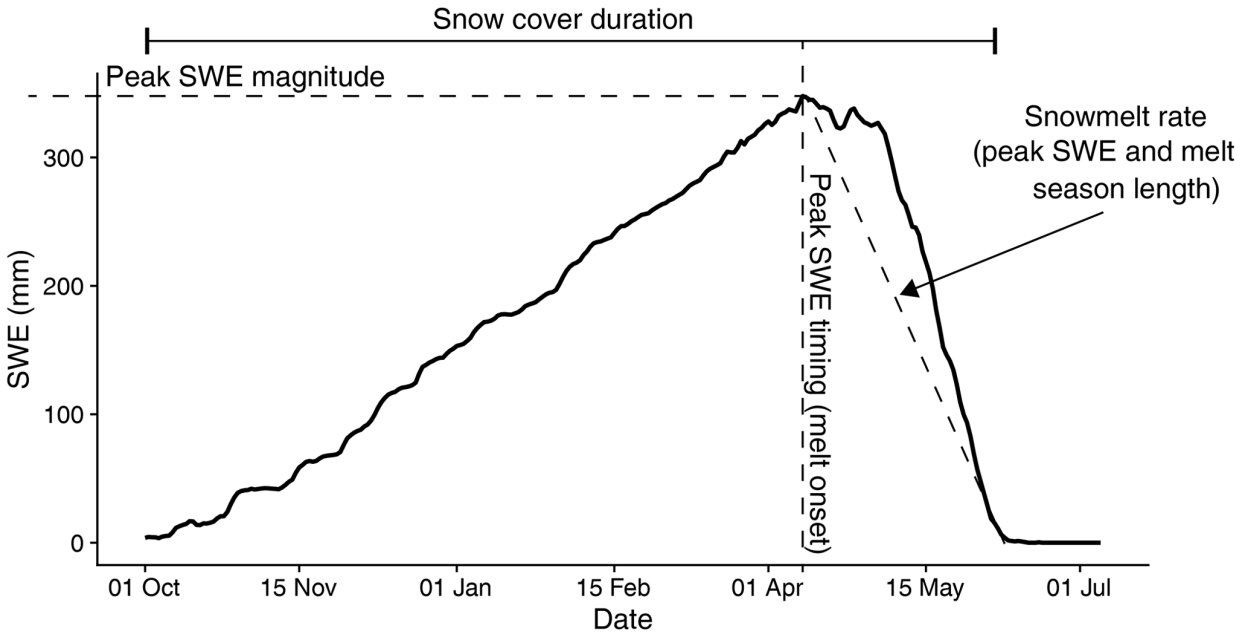
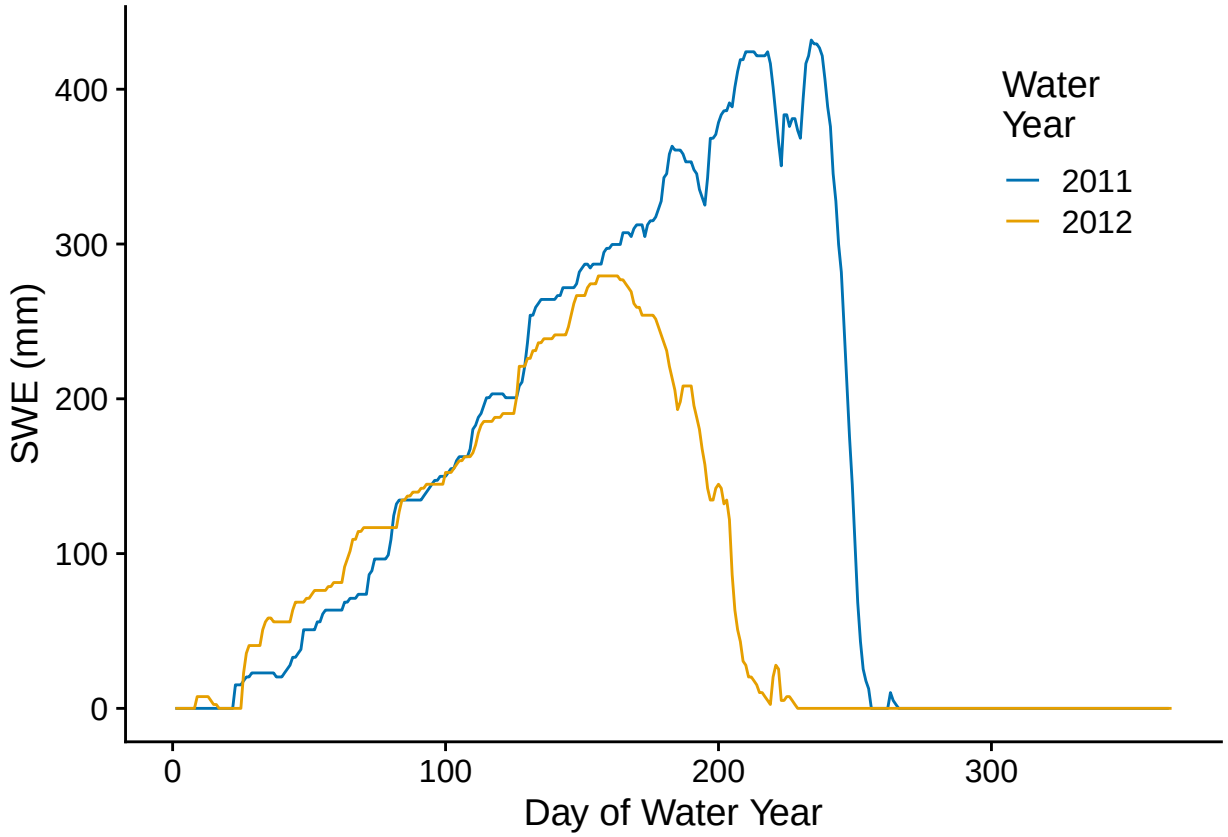


Figure 1: image

```
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



A glance at the above plot tells us a couple obvious things:

1. 2011 had a higher, later peak SWE than 2012
2. 2011 had more snow covered days than 2012

Rather than just visually assessing the data we can quantify it:

```
df %>%
  filter(wyear %in% plot_yrs) %>%
  group_by(wyear) %>%
  summarize(max_swe = max(snow_water_equivalent),
            max_swe_dowy = which.max(snow_water_equivalent),
            snow_off_dowy = min(which(snow_water_equivalent == 0 & dowy > max_swe_dowy)),
            melt_season_days = as.numeric(snow_off_dowy - max_swe_dowy),
            melt_rate = max_swe / melt_season_days) %>%
  mutate(across(where(is.numeric), ~ round(.x, 1))) %>%
  kable(col.names = c("Water Year", "Max SWE (mm)", "Max SWE DOWY",
                      "Snow Off DOWY", "Melt Season (d)", "Melt Rate (mm/d)"))
```

Water Year	Max SWE (mm)	Max SWE DOWY	Snow Off DOWY	Melt Season (d)	Melt Rate (mm/d)
2011	431.8	234	256	22	19.6
2012	279.4	156	229	73	3.8

In the next notebook, we'll take this form of analysis a step further and produce a series of metrics for multiple sites using R functions we create. Before that, though, we'll look at some other SNOTEL data that is available via `snotelr` and a URL-builder approach.

Other SNOTEL variables

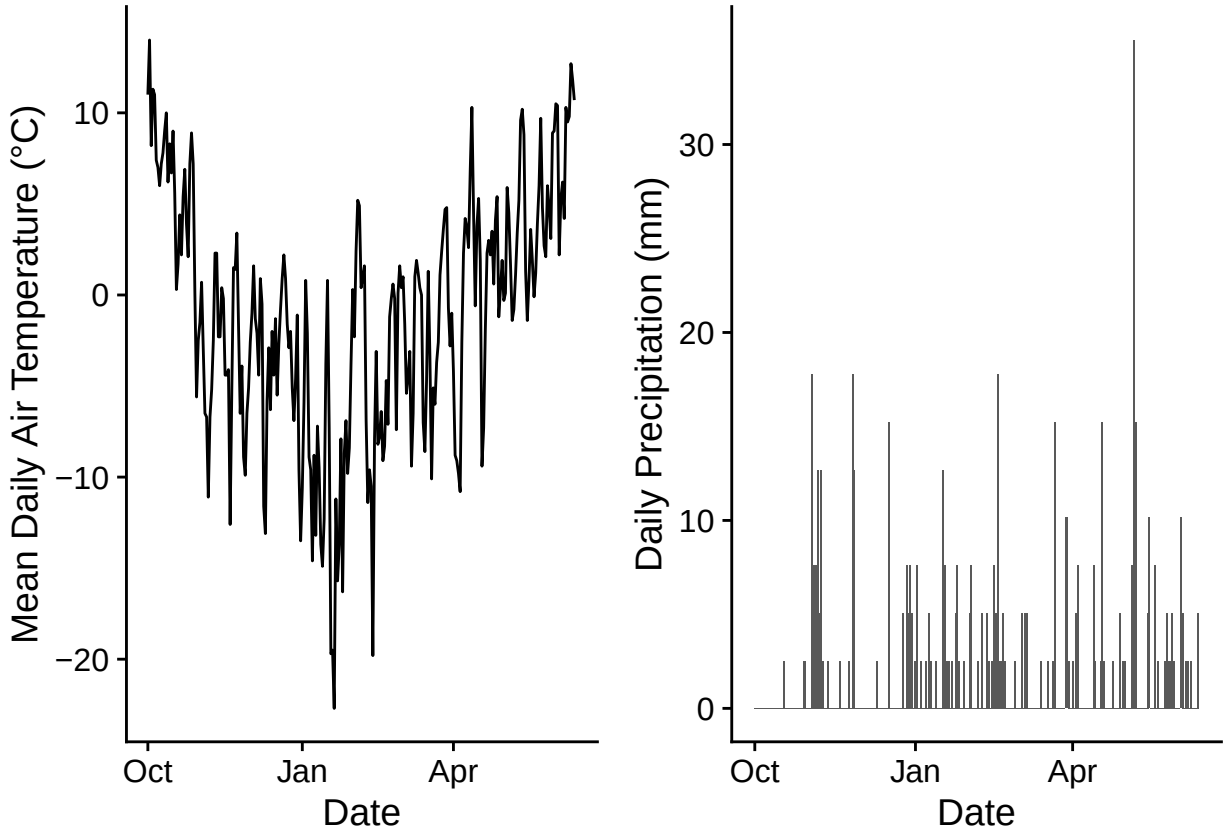
From snotelr

The `snotelr` package also retrieves ancillary met data like air temperature and precipitation (both incremental and cumulative).

```
plot_grid(
  df %>%
  filter(wyear == plot_yr) %>%
  ggplot(aes(date, temperature_mean)) +
  geom_line() +
  labs(x = "Date",
       y = "Mean Daily Air Temperature (°C)"),
  df %>%
  filter(wyear == plot_yr) %>%
  ggplot(aes(date, precipitation)) +
  geom_bar(stat = "identity") +
  labs(x = "Date",
       y = "Daily Precipitation (mm)"),
  ncol = 2
)
```

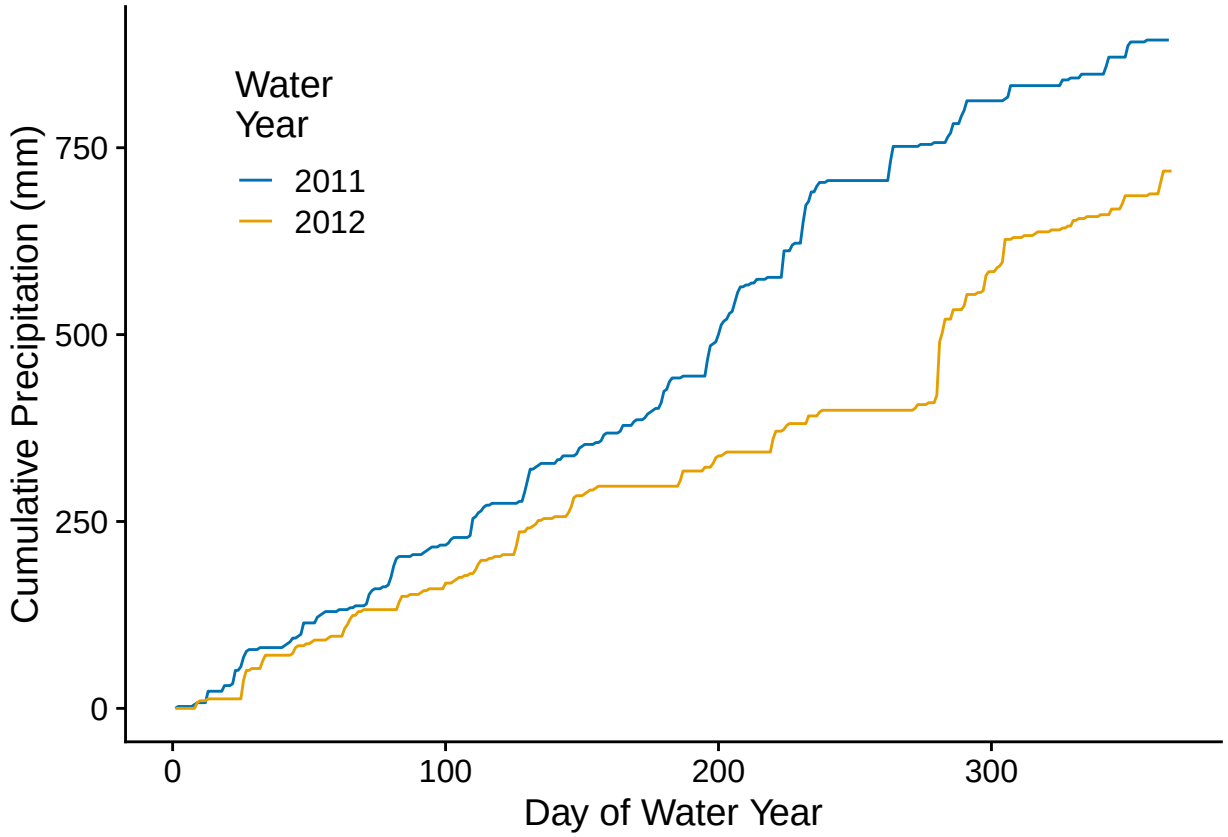
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_bar()`).
```



In fact, we can view the cumulative precip data from the two snow years plotted above to see the difference between a drought and non-drought year.

```
df %>%
  filter(wyear %in% plot_yrs) %>%
  ggplot(aes(dowy, precipitation_cumulative, color = as.factor(wyear))) +
  geom_line() +
  labs(x = "Day of Water Year",
       y = "Cumulative Precipitation (mm)") +
  scale_color_manual(values = c("#0072B2", "#E69F00"),
                     name = "Water\nYear") +
  theme(legend.position = c(0.1, 0.8))
```



In 2012 there were long stretches of zero precipitation. It was a *dry* snow drought. There are also *warm* snow droughts. We can explore this concept further in the synchrony work if needed.

SNOTEL data URLs

Another, non-snotelr way to get SNOTEL data is to build a URL that accesses it. (You can also use the SNOTEL API if you want to play around with different URL structures.)

```
# This functions downloads all of the available sensor data
# If you want just the regular SNOTEL met and snow data, replace "report=ALL" with "report=STAND"
allDownload <- function(ID, WYEAR){
  dl1 = "https://wcc.sc.egov.usda.gov/nwcc/view?intervalType=Historic+&report=ALL&timeseries=Daily&forma
  dl2 = "&year="
  dl3 = "&month=WY"
  read.csv(url(paste0(dl1,
                      ID,
                      dl2,
                      WYEAR,
                      dl3)),
           stringsAsFactors = F,
           skip = 6, na.strings = "-99.9")
}
```

Here, all we need to do is supply the site ID and the water year(s):

```
new_site = 784
new_years = c(2015, 2019)
new_df <- map_dfr(new_years, ~ allDownload(ID = new_site, WYEAR = .x))
```

A look at the data shows some new columns, with very ugly names:

```
new_df %>%
  head()
```

```
##   Site.Id       Date Time WTEQ.I.1..in. PREC.I.1..in. TOBS.I.1..degC.
## 1     784 2014-10-01                  0           0.0             3.2
## 2     784 2014-10-02                  0           0.0             5.5
## 3     784 2014-10-03                  0           0.0             8.8
## 4     784 2014-10-04                  0           0.1            13.4
## 5     784 2014-10-05                  0           0.1            11.7
## 6     784 2014-10-06                  0           0.1            12.3
##   TMAX.D.1..degC. TMIN.D.1..degC. TAVG.D.1..degC. SNWD.I.1..in.
## 1            11.8             3.0             7.2             0
## 2            10.2             0.6             4.9             0
## 3            19.0             5.5            12.3             0
## 4            20.5             8.8            14.1             0
## 5            21.6            11.2            15.8             0
## 6            21.5            10.7            14.8             0
##   SMS.I.1..2..pct....silt. SMS.I.1..8..pct....silt. SMS.I.1..20..pct....silt.
## 1                     34.2                     18.6                      1.9
## 2                     32.7                     18.3                      2.2
## 3                     31.9                     18.8                      2.0
## 4                     31.1                     19.3                      2.1
## 5                     29.5                     19.8                      2.2
## 6                     28.4                     20.2                      2.2
##   STO.I.1..2..degC. STO.I.1..8..degC. STO.I.1..20..degC. SAL.I.1..2..gram.
## 1               8.1              10.2               10.8              0.4
## 2               7.5              10.1               10.7              0.4
## 3               9.1              10.6               10.6              0.4
## 4              10.1              11.2               10.7              0.4
## 5              10.9              11.9               10.9              0.4
## 6              11.1              12.3               11.1              0.4
##   SAL.I.1..8..gram. SAL.I.1..20..gram. RDC.I.1..2..unit. RDC.I.1..8..unit.
## 1               0.3                0.1            21.63            10.84
## 2               0.3                0.1            20.10            10.73
## 3               0.3                0.1            19.34            10.94
## 4               0.3                0.1            18.57            11.15
## 5               0.3                0.1            17.26            11.39
## 6               0.3                0.1            16.44            11.57
##   RDC.I.1..20..unit. BATT.I.1..volt. WDIRV.D.1..degr. WSPDX.D.1..mph.
## 1               4.35           12.74              293            23.8
## 2               4.44           12.71              104            15.0
## 3               4.36           12.75              114             8.4
## 4               4.41           12.76              234             7.2
## 5               4.42           12.77              324            10.0
## 6               4.43           12.75              160            10.2
##   WSPDV.D.1..mph.  X
## 1             6.9 NA
```

```
## 2             3.2 NA
## 3             1.9 NA
## 4             1.8 NA
## 5             2.2 NA
## 6             1.9 NA
```

The columns starting with `SMS` and `STO` correspond to soil moisture and temperature, respectively, from different levels beneath the ground surface. The numbers (e.g., 2, 8, 20) in the column names correspond to the sensors depths in inches. Yes, inches.

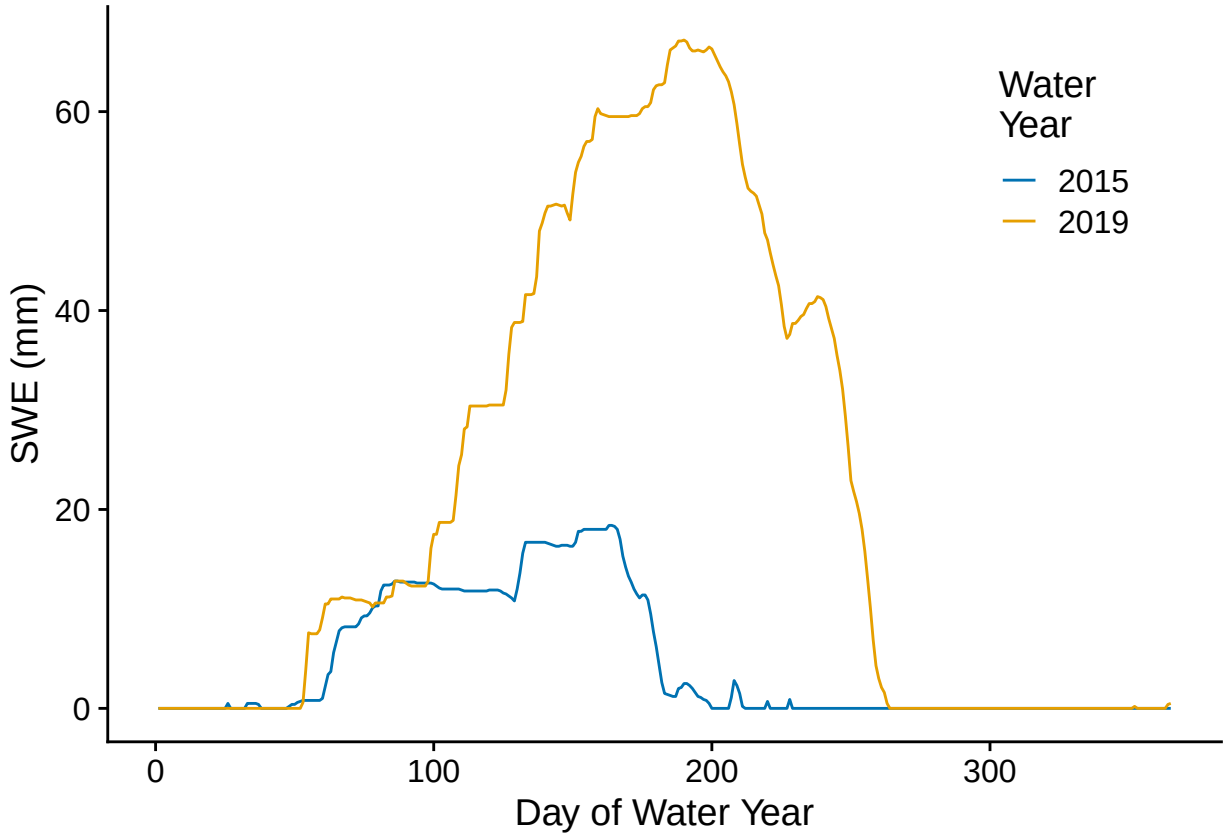
Like we did previously, we'll add some date information and visualize the data.

```
# Format the date and add wy info
new_df <- new_df %>%
  mutate(date = ymd(Date),
         wyear = wateryear(date),
         dowy = day_of_wateryear(date))
```

Because this is a new site, Palisades Tahoe in CA's Sierra Nevada, we'll look at SWE from a good (2019) and bad (2015) snow year.

```
new_df %>%
  ggplot(aes(dowy, WTEQ.I.1..in., color = as.factor(wyear))) +
  geom_line() +
  labs(x = "Day of Water Year",
       y = "SWE (mm)") +
  scale_color_manual(values = c("#0072B2", "#E69F00"),
                     name = "Water\nYear") +
  theme(legend.position = c(0.8, 0.8))
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

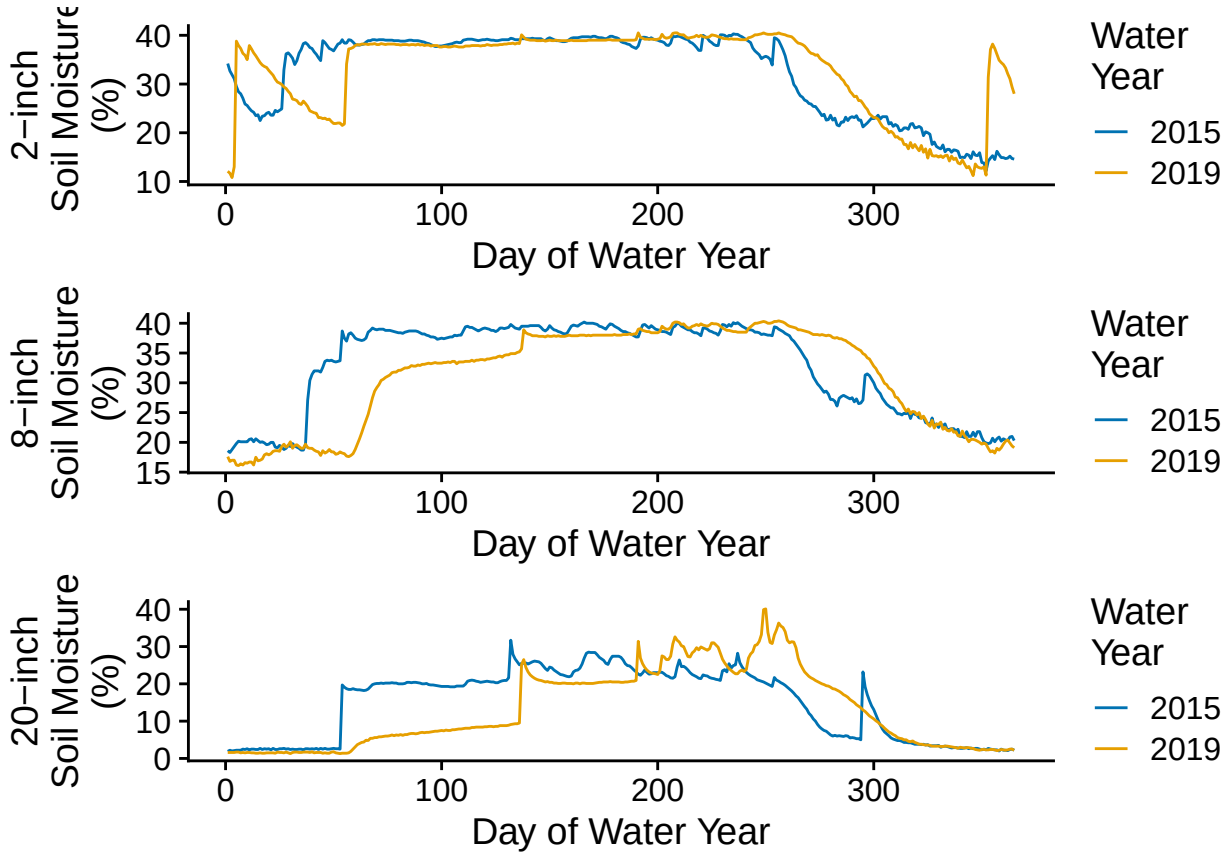


Fortunately, we lived in Reno in 2019 and not 2015.

We can also plot the different depths of soil moisture.

```
# Plot the soil moisture traces
plot_grid(
  new_df %>%
    ggplot(aes(dowy, SMS.I.1..2..pct....silt., color = as.factor(wyear))) +
    geom_line() +
    scale_color_manual(values = c("#0072B2", "#E69F00"),
                       name = "Water\nYear") +
    labs(x = "Day of Water Year", y = "2-inch\nSoil Moisture\n(%)"),
  new_df %>%
    ggplot(aes(dowy, SMS.I.1..8..pct....silt., color = as.factor(wyear))) +
    geom_line() +
    scale_color_manual(values = c("#0072B2", "#E69F00"),
                       name = "Water\nYear") +
    labs(x = "Day of Water Year", y = "8-inch\nSoil Moisture\n(%)"),
  new_df %>%
    ggplot(aes(dowy, SMS.I.1..20..pct....silt., color = as.factor(wyear))) +
    geom_line() +
    scale_color_manual(values = c("#0072B2", "#E69F00"),
                       name = "Water\nYear") +
    labs(x = "Day of Water Year", y = "20-inch\nSoil Moisture\n(%)"),
  ncol = 1
)
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



Let's move on now to computing snow metrics across sites and water years for our synchrony work.

Further reading and info

- snotelr R package
- SNOTEL website
- Jennings, K. S., & Molotch, N. P. (2019). The sensitivity of modeled snow accumulation and melt to precipitation phase methods across a climatic gradient. Hydrology and Earth System Sciences, 23(9), 3765-3786. https://doi.org/10.5194/hess-23-3765-2019