Step 03: Accessing and Analyzing Spatial SWE Data

Keith Jennings

Getting started

First, we need to load our packages like we did in our previous notebook.

```
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(cowplot)
```

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

```
theme_set(theme_cowplot())
library(knitr)
library(snotelr)
library(terra)  # working with raster and vector data
```

```
## terra 1.8.54
##
## Attaching package: 'terra'
##
## The following object is masked from 'package:knitr':
##
##     spin
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(sf)     # geospatial transforms and processing
```

```
## Linking to GEOS 3.13.0, GDAL 3.8.5, PROJ 9.5.1; sf_use_s2() is TRUE
```

```
library(exactextractr) # fast spatial summaries
```

Why do we need spatial SWE data if SNOTEL exists?

Let's explore the answer to this question with some numbers. First we'll import the SNOTEL and river basin metadata like we did in the previous notebook.

```
# snotel metadata
snotel_info <- snotel_info() %>%
  mutate(HUC = stringr::str_extract(string = description,
                                    pattern = "(?<=\\().*(?=\\))"))
# harbor dataset of river basin metadata
harbor_url <- "https://raw.githubusercontent.com/peckhams/nextgen_basin_repo/refs/heads/main/__Collated_
basin_info <- read_tsv(harbor_url)
# join the two into one dataset
all_info <- left_join(snotel_info,
                      basin_info,
                      by = "HUC")
```

In total, there are 920 SNOTEL stations, 543 of which started their service life 40 or more years ago. Not bad.

However, these SNOTEL stations are only in the mountains of the western US and Alaska, precluding other regions from analysis.

They also only cover a limited number of river basins, making it more difficult to explore snow-flow synchrony. For example, of the 671 unique CAMELS basins in HARBOR, only 24 have a SNOTEL station.

What's more, SNOTEL stations measure a fraction of their containing basins and often do not represent surrounding snow conditions in complex terrain (e.g., Molotch and Bales, 2006).

> Molotch, N.P. and Bales, R.C. (2006), SNOTEL representativeness in the Rio Grande headwaters on the basis of physiographics and remotely sensed snow cover persistence. Hydrol. Process., 20: 723-739. https://doi.org/10.1002/hyp.6128

Accessing spatial SWE data

To respond to these shortcomings, we can use spatial SWE data. While myriad products exist (all of which have their own pros and cons), we will focus here on the well validated, easily accessible University of Arizona product.

It's produced daily on a 4 km and 800 m grid.

There are multiple ways of accessing the data, but for expediency's sake, I downloaded the water year 2021 NetCDF file from their server: https://climate.arizona.edu/data/UA_SWE/WYData_4km/UA_SWE_Depth_WY2021.nc

It's included in the `/data` directory and we can import it now.

```r
# File path
nc_file <- "../data/UA_SWE_Depth_WY2021.nc"

# Read the NetCDF using terra
swe_raster <- rast(nc_file)
```

We can get some information about the data after we've imported the file.

```r
# Check netcdf file info
swe_raster
```

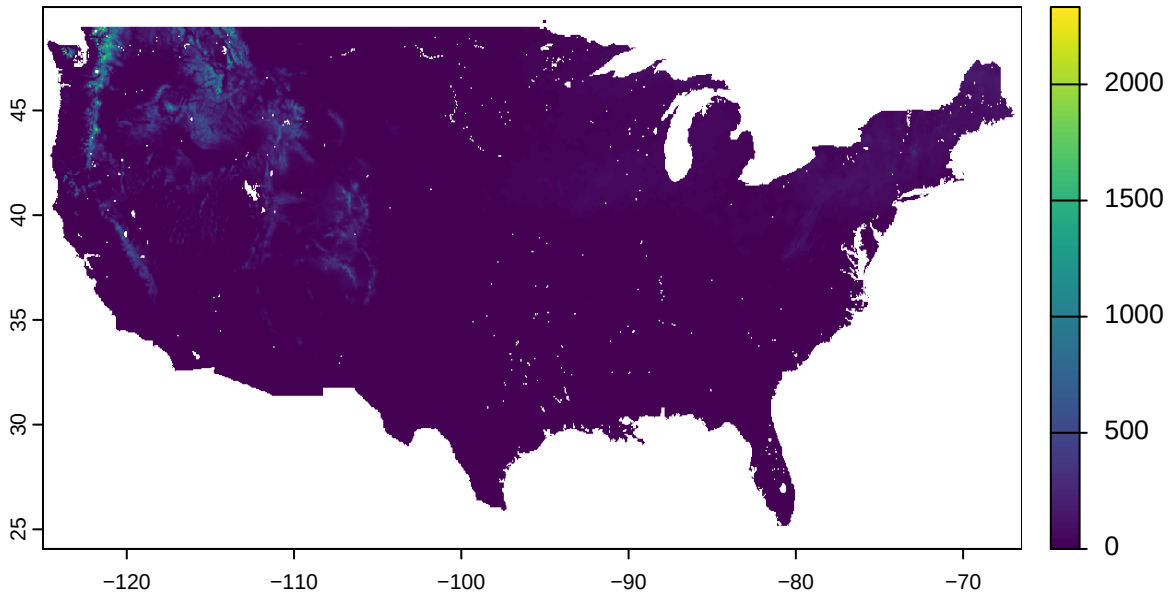
```
## class       : SpatRaster
## size        : 621, 1405, 730  (nrow, ncol, nlyr)
## resolution  : 0.04166667, 0.04166667  (x, y)
## extent      : -125.0208, -66.47917, 24.0625, 49.9375  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat NAD83 (EPSG:4269)
## sources     : UA_SWE_Depth_WY2021.nc:SWE  (365 layers)
##               UA_SWE_Depth_WY2021.nc:DEPTH  (365 layers)
## varnames    : SWE (Snow Water Equivalent)
##               DEPTH (Snow Depth)
## names       :           SWE_1,           SWE_2,           SWE_3,           SWE_4,           SWE_5,
## unit        : millimeters h20, millimeters h20, millimeters h20, millimeters h20, millimeters h20, mi
## time (days) : 2020-10-01 to 2021-09-30 (365 steps)
```

In the above we see important data points like the spatial resolution, the extent, the coordinate reference system (CRS), the variables (SWE and snow depth) and their units (mm), the layer names, and time.

Conveniently, each layer is designated by the variable (SWE and DEPTH) plus the day of water year.

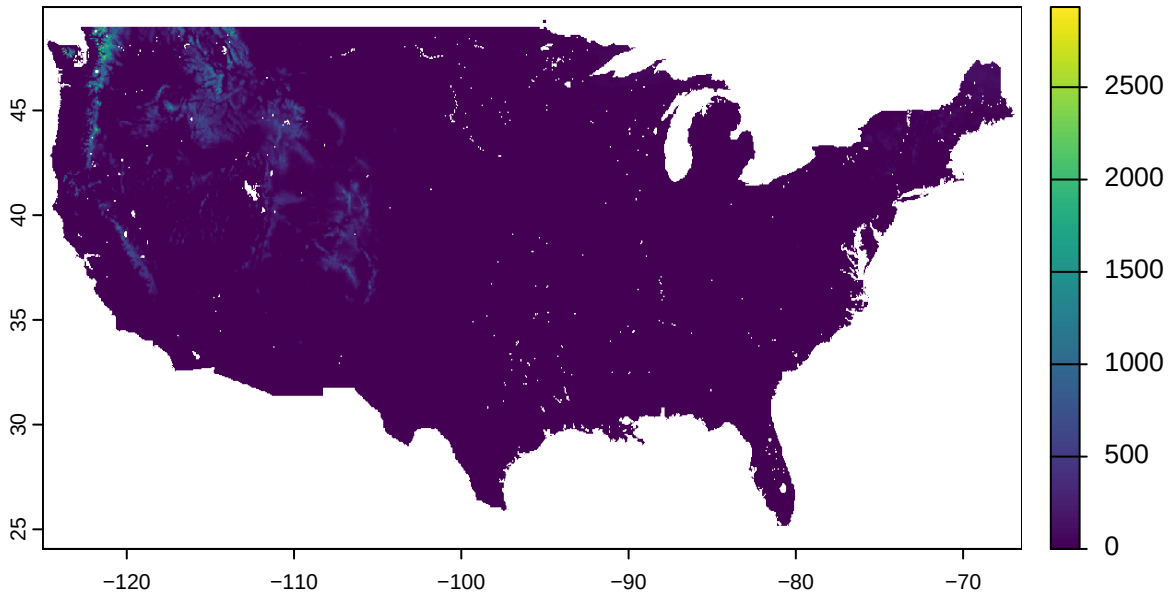
Let's look at some data now. We can start with just the numeric element.

```r
plot(swe_raster[[150]])
```



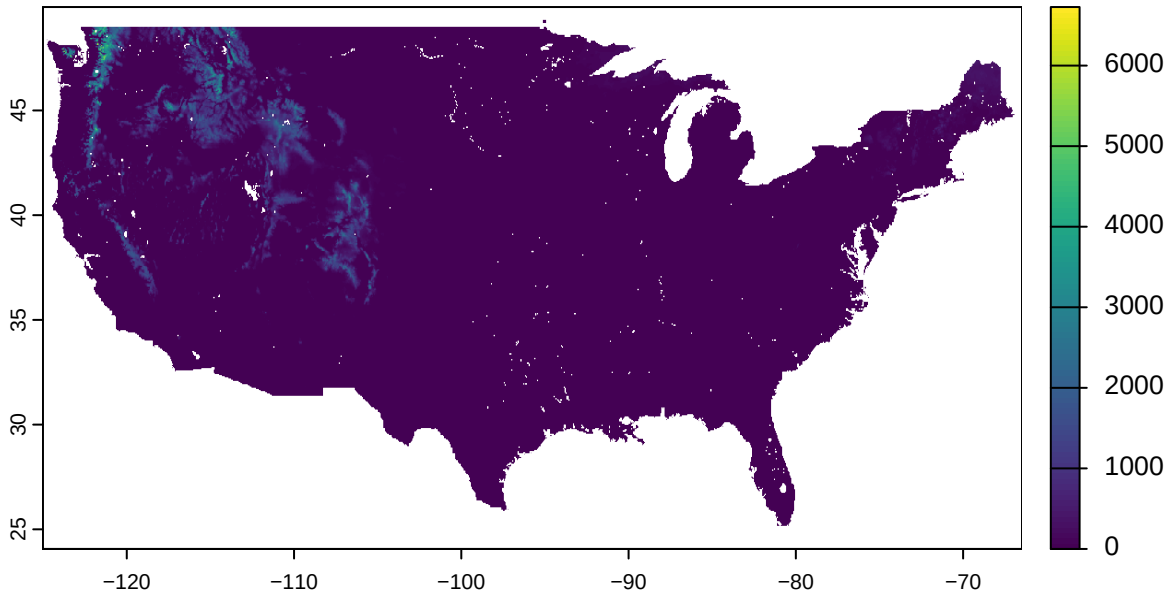
We can also use the layer name for SWE...

```
plot(swe_raster[["SWE_180"]])
```



. . . and snow depth.

```
plot(swe_raster[["DEPTH_180"]])
```



Analyzing spatial SWE data

However, looking at pretty maps of SWE and snow depth won't help our synchrony work.

We'll start the process by summarizing the 1 year of SWE data from our file to a single basin where we have SNOTEL observations. (Future work should extend this to all years and basins of interest.)

Our basin

I extracted our basin from the USGS GAGES II dataset here. Specifically, this is the basin for USGS gage 06187915, which is part of the GAGES II Reference subset. It also encompasses SNOTEL station 670, which has expressed statistically significant declines in maximum SWE, snow cover duration, and the percent of total melt that occurs before max SWE.

We'll import this basin polygon and match the CRS to the SWE data.

```
# Name the file
basin_file <- "../data/usgs_06187915_basin.shp"

# Load basin shapefile
basin <- st_read(basin_file)
```

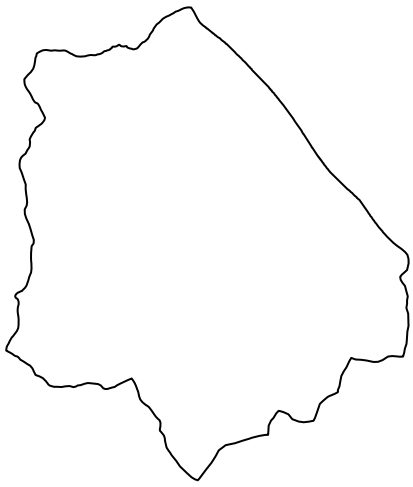
```
## Reading layer 'usgs_06187915_basin' from data source
##   '/Users/ksjennin/Documents/projects/cz_synchrony/snow-analysis/data/usgs_06187915_basin.shp'
```

```
##   using driver 'ESRI Shapefile'
## Simple feature collection with 1 feature and 3 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -1099531 ymin: 2520762 xmax: -1089344 ymax: 2532371
## Projected CRS: NAD83 / Conus Albers
```

```
# Ensure coordinate reference systems match
basin <- st_transform(basin, crs(swe_raster))
```

Here's what it looks like:

```
plot(basin$geometry)
```



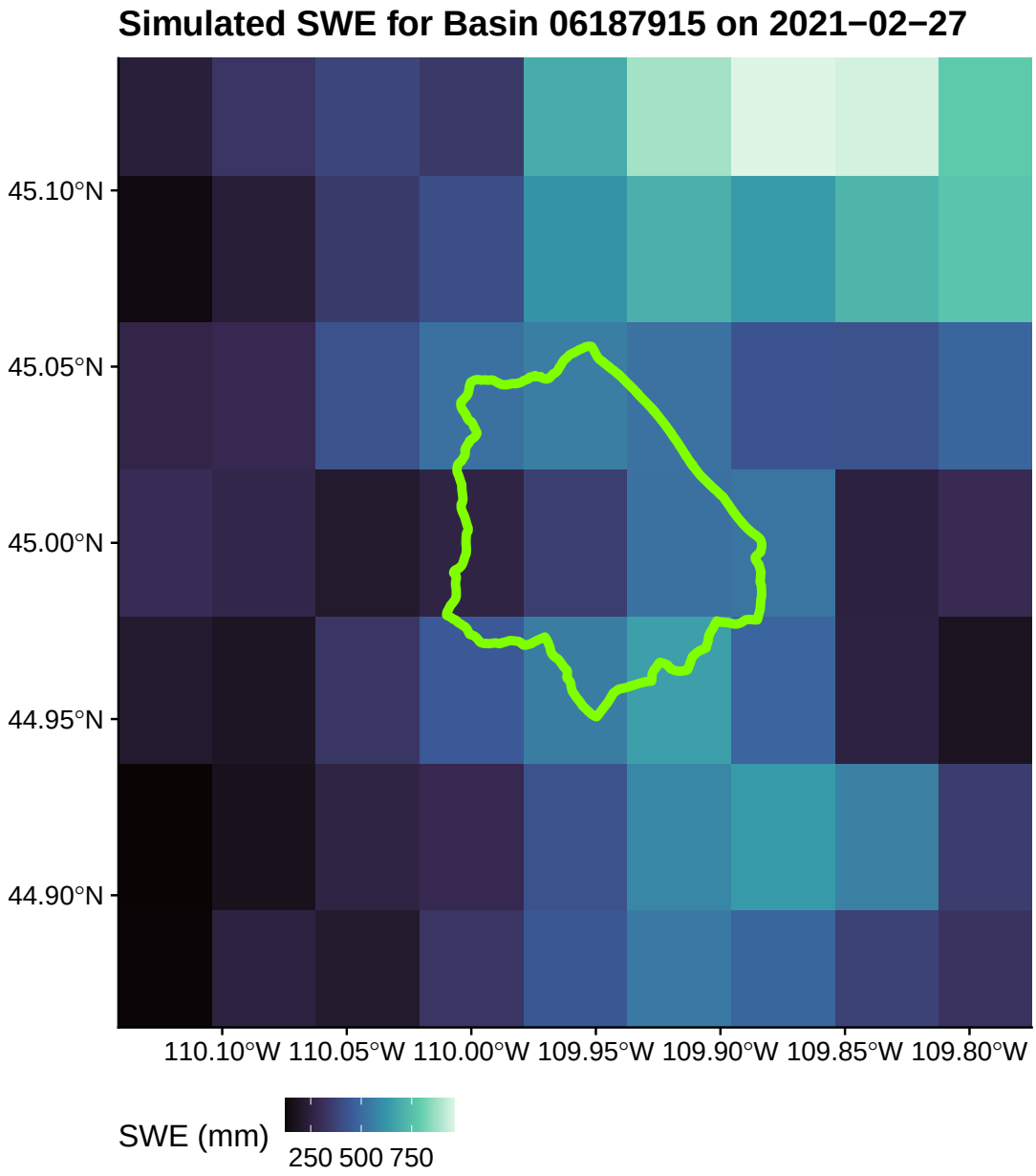
Wow, not very interesting. Let's continue.

Basin, U of A SWE, and SNOTEL become one

First we'll provide a better visualization of the basin and the UofA SWE product.

```
# Grab one day of the data
plot_swe <- swe_raster[["SWE_150"]]

# Convert basin to SpatVector
```

```r
basin_vect <- vect(basin)

# Crop and mask the SWE raster to the basin area buffer
buffered_basin <- terra::buffer(basin_vect, width = 10000)  # 20 km buffer
plot_swe <- crop(plot_swe, buffered_basin)

# Convert the SWE into a dataframe to plot in ggplot
swe_df <- as.data.frame(plot_swe, xy = TRUE, na.rm = TRUE)
names(swe_df)[3] <- "swe_mm"

# Plot
ggplot() +
  geom_raster(data = swe_df, aes(x = x, y = y, fill = swe_mm)) +
  scale_fill_viridis_c(option = "G", name = "SWE (mm)") +
  geom_sf(data = basin, fill = NA, color = "chartreuse", lwd = 2) +
  coord_sf(xlim = range(swe_df$x), ylim = range(swe_df$y)) +
  theme(axis.title = element_blank(), legend.position = "bottom") +
  labs(title = paste0("Simulated SWE for Basin 06187915 on ", time(plot_swe)))
```



There we can see the basin outline for 06187915 and the simulated SWE from 2021-02-27.

The other thing we'll do is summarize the SWE data within the basin and compare it to the SNOTEL data.

```
# Summarize each day of SWE across the basin using exactextractr
basin_swe <- exact_extract(swe_raster, basin, fun = "mean")

# Transpose to get one row per day
basin_swe <- as.data.frame(t(basin_swe)) %>%
  rename(value = V1) %>%
  mutate(variable = str_extract(row.names(.), "(?<=\\.)[^_]+"),
         dowy = as.numeric(str_extract(row.names(.), "(?<=_)\\d+")),
         wyear = 2021)

# Quick look at data
```

```
basin_swe %>%
  head()
```

```
##            value variable dowy wyear
## mean.SWE_1     0      SWE    1  2021
## mean.SWE_2     0      SWE    2  2021
## mean.SWE_3     0      SWE    3  2021
## mean.SWE_4     0      SWE    4  2021
## mean.SWE_5     0      SWE    5  2021
## mean.SWE_6     0      SWE    6  2021
```

Let's now import the SNOTEL SWE data for that year and join it to the spatial SWE data.

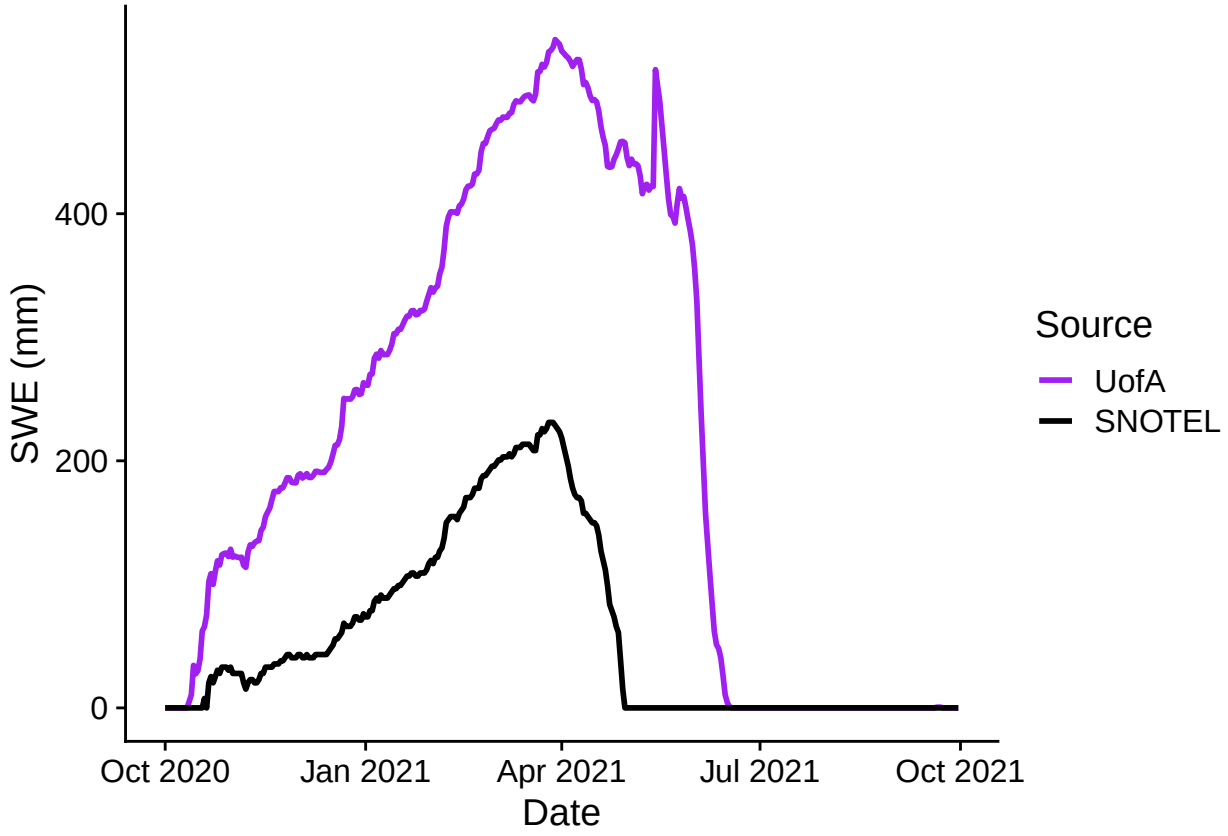
```
# Import SNOTEL data and filter to just time and basin of interest
snotel_swe <- readRDS("../data/snotel_camels_subset.RDS") %>%
  filter(wyear == "2021" & site_id == 670)

# Join to a version of the basin SWE data
all_swe <- left_join(
  basin_swe %>% filter(variable == "SWE") %>%
    select(swe_mm = value, dowy),
  snotel_swe,
  by = "dowy"
)

all_swe %>%
  head
```

```
##   swe_mm.x dowy site_id       date swe_mm.y snow_depth_mm ppt_mm tair_av_degC
## 1        0    1     670 2020-10-01        0             0      0          8.4
## 2        0    2     670 2020-10-02        0             0      0          7.3
## 3        0    3     670 2020-10-03        0             0      0          7.0
## 4        0    4     670 2020-10-04        0             0      0          9.5
## 5        0    5     670 2020-10-05        0             0      0          9.0
## 6        0    6     670 2020-10-06        0             0      0          9.4
##   wyear
## 1  2021
## 2  2021
## 3  2021
## 4  2021
## 5  2021
## 6  2021
```

First thing we'll do is plot the two SWE traces.

```
all_swe %>%
  select(date, swe_mm.x, swe_mm.y) %>%
  pivot_longer(cols = -date) %>%
  ggplot(aes(date, value, color = name)) +
  geom_line(lwd = 1) +
  scale_color_manual(breaks = c("swe_mm.x", "swe_mm.y"),
                     values = c("purple", "black"),
```

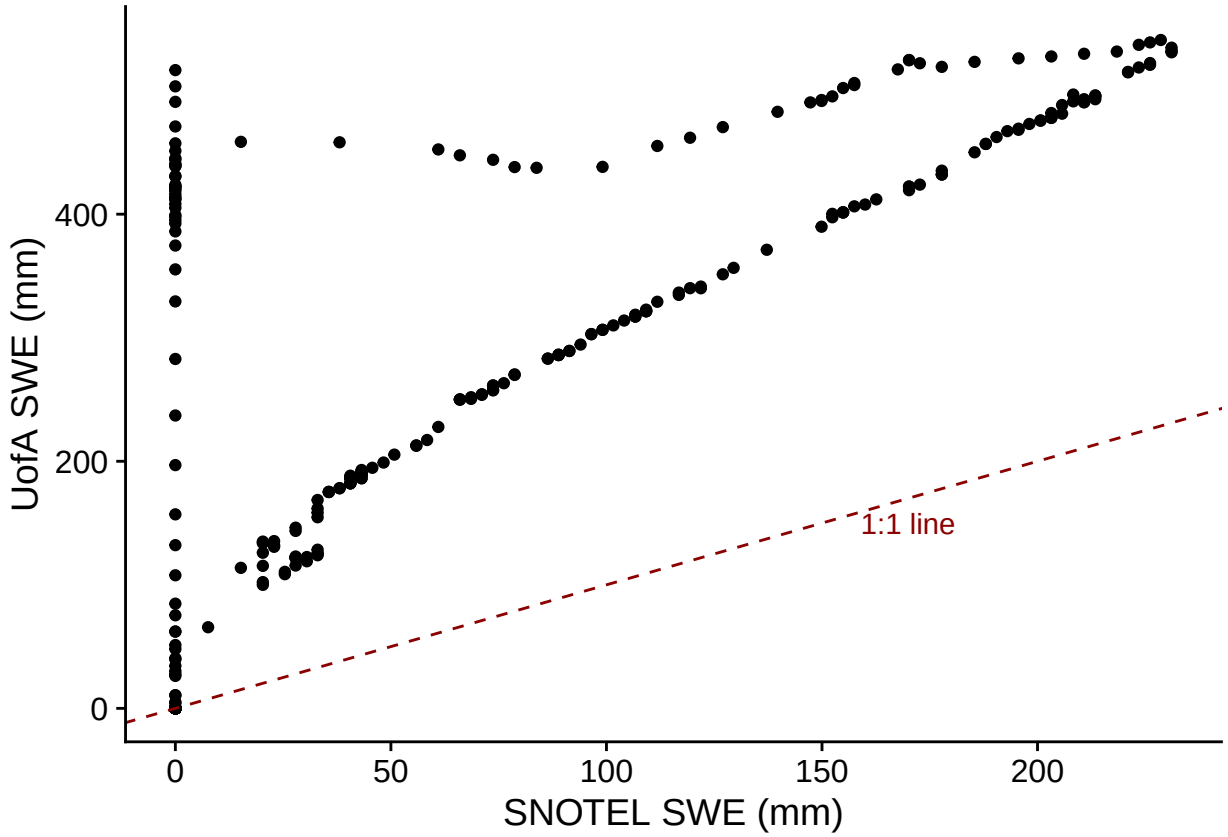
```
                    labels = c("UofA", "SNOTEL"),
                    name = "Source") +
  labs(x = "Date", y = "SWE (mm)")
```



There are some obvious major differences (SWE magnitude, snow cover duration, snow off date, etc.), which we can explore at a later time. Right now I'll note that the UofA SWE product assimilates SNOTEL observed SWE and divergences may arise from point vs. basin elevational differences, the assimilation protocol of the product, and the snowmelt model used to create it. (And, this is only one location so we shouldn't infer too much about model performance.)

We can also look at a scatterplot.

```
ggplot(all_swe, aes(swe_mm.y, swe_mm.x)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, lty = "dashed", color = "darkred") +
  annotate(geom = "text", x = 170, y = 150, label = "1:1 line", color = "darkred") +
  labs(x = "SNOTEL SWE (mm)", y = "UofA SWE (mm)")
```



It doesn't look great, but just for laughs, and for our future work, we'll compute a few performance metrics.

```
# Functions for performance metrics
nse <- function(sim, obs) {
  1 - sum((sim - obs)^2) / sum((obs - mean(obs))^2)
}
kge <- function(sim, obs) {
  r <- cor(sim, obs)
  alpha <- sd(sim) / sd(obs)
  beta <- mean(sim) / mean(obs)
  1 - sqrt((r - 1)^2 + (alpha - 1)^2 + (beta - 1)^2)
}
mean_absolute_bias <- function(sim, obs){
  mean(sim) - mean(obs)
}
mean_relative_bias <- function(sim, obs){
  ((mean(sim) - mean(obs)) / mean(obs)) * 100
}

# Compute performance on the data
swe_performance <- all_swe %>%
  summarize(nse = nse(swe_mm.x, swe_mm.y),
            kge = kge(swe_mm.x, swe_mm.y),
            mab = mean_absolute_bias(swe_mm.x, swe_mm.y),
            mrb = mean_relative_bias(swe_mm.x, swe_mm.y))
```

```
# View the metrics
swe_performance
```

```
##         nse       kge      mab      mrb
## 1 -7.499323 -2.222629 159.5016 276.8558
```

That's it for this portion of the analysis. At this point we've set the stage for future work with code that we can extend and repurpose.

What next?

1. Generate research questions with the main project and other sub-projects
2. Identify snow data needs
3. Build out a targets-based workflow for data access, processing, and analysis (likely expanding to all long-term SNOTEL stations and spatial SWE from selected basins with USGS stream gages)
4. Download and process all necessary data
5. Compute performance metrics for modeled SWE products we want to use
6. Perform bias correction as necessary
7. Calculate snow cover properties for point and spatial SWE data to relate to streamflow data in synchrony framework