Untitled

Charles Jason Tinant

6/15/2018

1.0 General questions: What is the drought history of the Pine Ridge Reservation? Does the drought extent differ across the study area?

2.0 Data read in from NOAA website 3.0 Data munging to check, join and remove NAs 3.1 Updated unit vals - originally in tenths of mm; now in mm 3.2 Changed daily data into monthly and annual data 3.3 Cross-validated monthly data 3.4 Created plots 4.0 Describe results: The results generally show rainshadow effects from the Black Hills and southeast to northwest aridity trend at the annual scale.

```
knitr::opts_chunk$set(echo = TRUE)
# Get API key (aka, token) for downloading precip data at http://www.ncdc.noaa.gov/cdo-web/token
# token for NOAA API tied to jtinant@olc.edu
options(noaakey = "VpcuARumMpCfFyclKHPfvskEYnaiLJHD")
# see 'rnoaa' for details
```

```
# for header:, include=FALSE

# Sets up the library of packages
library("tidyverse")
library("here") # identifies where to save work
library("rio") # more robust I/O - to import and clean data
library("DataExplorer") # quick look at NA vals
library("rnoaa") #  R wrapper for NOAA data inc. NCDC
library("lubridate") # easier dates
library("corrplot") # correlation plots
library("broom") # tidies linear models

# mapping packages
library("maps") # outlines of continents, countries, states & counties
library("mapdata") # higher-resolution outlines
library('ggmap')
library('deldir') # for Vorononi tesselation - Theissen polygons

# library("janitor") # tools for examining and cleaning dirty data
# library("dataRetrieval") # USGS data import
# library("RColorBrewer") - there is a better one?
# library("workflowr") # creates a research website
# library("colorspace")
# library("bookdown") #
# library(unpivotr) # fix nasty Excel files
# library("friendlyeval")

# a useful description of commits:
# http://r-pkgs.had.co.nz/git.html
```

```
# Creates points for maps. No need to evaluate.
# The output files have been created.
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# 'station' is a variable for NOAA weather station locations
```

```
# 'gage' is a variable for USGS stream gages
# 'site' is a variable for OST monitoring stationsn

# 'station'
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# get station id with the mapping tool at
# https://www.ncdc.noaa.gov/cdo-web/datatools/findstation
# iterate across a list of station ids by purrr::map
# to get NOAA station meta data using rnoaa::ncdc_stationsn
# the output is a list of 7 x 2 x 9.n
# flatten into a dataframe by purrr::flatten
# reorder and rename columns by dplyr
# save the station df by rion
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~n
sta_input <- data.frame(name = c('RAPID CITY RGNL AP',
                                 'HOT SPRINGS',
                                'OELRICHS',
                                'INTERIOR 3 NE',
                                'COTTONWOOD 2 E',
                                'LONG VALLEY',
                                'HARRISON',
                                'AINSWORTH, NE',
                                'MURDO, SD US',
                                'MISSION 14 S, SD US',
                                'ORAL, SD US',
                                'KADOKA 0.3 N, SD US'),
                        id = c("GHCND:USW00024090",
                               "GHCND:US1SDFR0001",
                               "GHCND:USC00396212",
                               "GHCND:USC00394184",
                               "GHCND:USC00391972",
                               "GHCND:USC00394983",
                               "GHCND:USC00253615",
                               "GHCND:USC00250050",
                               "GHCND:USC00395891",
                               "GHCND:USC00395638",
                               "GHCND:USC00396304",
                               "GHCND:US1SDJK0006"),
                        stringsAsFactors = FALSE)

#station <- map(sta_input$id, ncdc_stations)
#station <- flatten_dfr(station)

#station <- station %>%
#  rename(lat = latitude) %>%
#  rename(lon = longitude) %>%
#  select(id, name, lat, lon, everything())

#export(station, "data/sta_meta_orig.csv")

# 'gage'
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# get gage ids by USGS watermapper
```

```r
# iterate across a list of gage ids by purrr::map_dfr
# to get gage metadata using dataRetrieval::readNWISsite
# reorder and rename columns by dplyr
# save the station df by rio
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

gage_id <- data.frame(name = c("WHITE R NR NE-SD STATE LINE",
                               "WHITE R NEAR OGLALA SD",
                               "WHITE CLAY CR NEAR OGLALA SD",
                               "WHITE R NEAR INTERIOR SD",
                               "WOUNDED KNEE CREEK AT WOUNDED KNEE SD",
                               "BEAR IN THE LODGE CR NEAR WANBLEE SD",
                               "WHITE R NEAR KADOKA SD",
                               "BLACK PIPE CREEK NR BELVIDERE SD",
                               "LITTLE WHITE R NEAR MARTIN SD",
                               "LAKE CR BELOW REFUGE NEAR TUTHILL SD",
                               "LITTLE WHITE R NEAR VETAL SD",
                               "SOUTH FORK BAD R NEAR COTTONWOOD SD",
                               "BAD R NEAR MIDLAND SD"),
                      id = c('06445685', '06446000', '06445980',
                             '06446500', '06446100', '06446700',
                             '06447000', '06447230', '06447500',
                             '06449000',  '06449100', '06440200',
                             '06441000'),
                      stringsAsFactors = FALSE)


gage <- map_dfr(gage_id$id, readNWISsite)
gage <- gage %>%
  select(site_no, station_nm, dec_lat_va, dec_long_va, everything())
export(gage, "data/gage_meta.csv")

# site
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
site <- import("data/Chemistry-1993-2013_17Mar21.csv")
eco <- import("data/MacroSummaries.csv")

eco <- eco %>%
  clean_names() %>%
  select(1:2) %>%
  rename(id = station) %>%
  distinct(id, .keep_all = TRUE)

site <- site %>%
  clean_names() %>%
  arrange(sample_sites) %>%
  filter(sample_sites != "Bear in the Lodge USGS1") %>%
  filter(sample_sites != "Bear in the Lodge USGS2") %>%
  filter(sample_sites != "Bear in the Lodge USGS3") %>%
  filter(sample_sites != "Black Pipe II") %>%
  filter(sample_sites != "Corn Creek I") %>%
  filter(sample_sites != "Little Corn Creek I") %>%
  filter(sample_sites != "Medicine Root II") %>%
```

```
  filter(sample_sites != "Porcupine Lagoon downstream") %>%
  filter(sample_sites != "Porcupine Lagoon upstream") %>%
  filter(sample_sites != "Pine Ridge Lift Station Downstream") %>%
  distinct(sample_sites, .keep_all = TRUE) %>%
  select(1:4) %>%
  select(-1) %>%
  rename(name = sample_sites) %>%
  rename(lat = latitude) %>%
  rename(lon = longitude) %>%
  mutate(lon = -1 * lon)

site_id <- data.frame(id = c("AMH1", "BEA1", "BEA2", "BEA3", "BEL1",
                             "BEL2", "BLP1", "BUZ1", "CHR1", "CHR2",
                             "CRA1", "EAN1", "EAN2", "LWR1", "LWR2",
                             "LWR3", "LWR4", "LON1", "LOD1", "MER1",
                             "MER2", "MER3", "MER4", "NFL1", "PAS1",
                             "PAS2", "PAS3", "POR1", "POR2", "POR3",
                             "POT1", "RED1", "WCC1", "WCC2", "WCC3",
                             "WHR1", "WHR2", "WHR3", "WHR4", "WHR5",
                             "WOL1", "WOK1", "WOK2", "WOK3", "WOK4"),
                      name = c("American Horse I",
                               "Bear Creek I",
                               "Bear Creek II",
                               "Bear Creek III",
                               "Bear in the Lodge I",
                               "Bear in the Lodge II",
                               "Black Pipe I",
                               "Buzzard Creek I",
                               "Cheyenne River I",
                               "Cheyenne River II",
                               "Craven Creek I",
                               "Eagle Nest I",
                               "Eagle Nest II",
                               "Little White River I",
                               "Little White River II",
                               "Little White River III",
                               "Little White River IV",
                               "Long Creek I",
                               "Lost Dog Creek I",
                               "Medicine Root I",
                               "Medicine Root II",
                               "Medicine Root III",
                               "Medicine Root IV",
                               "No Flesh Creek I",
                               "Pass Creek I",
                               "Pass Creek II",
                               "Pass Creek III",
                               "Porcupine Creek I",
                               "Porcupine Creek II",
                               "Porcupine Creek III",
                               "Potato Creek",
                               "Red Water Creek",
                               "White Clay Creek I",
```

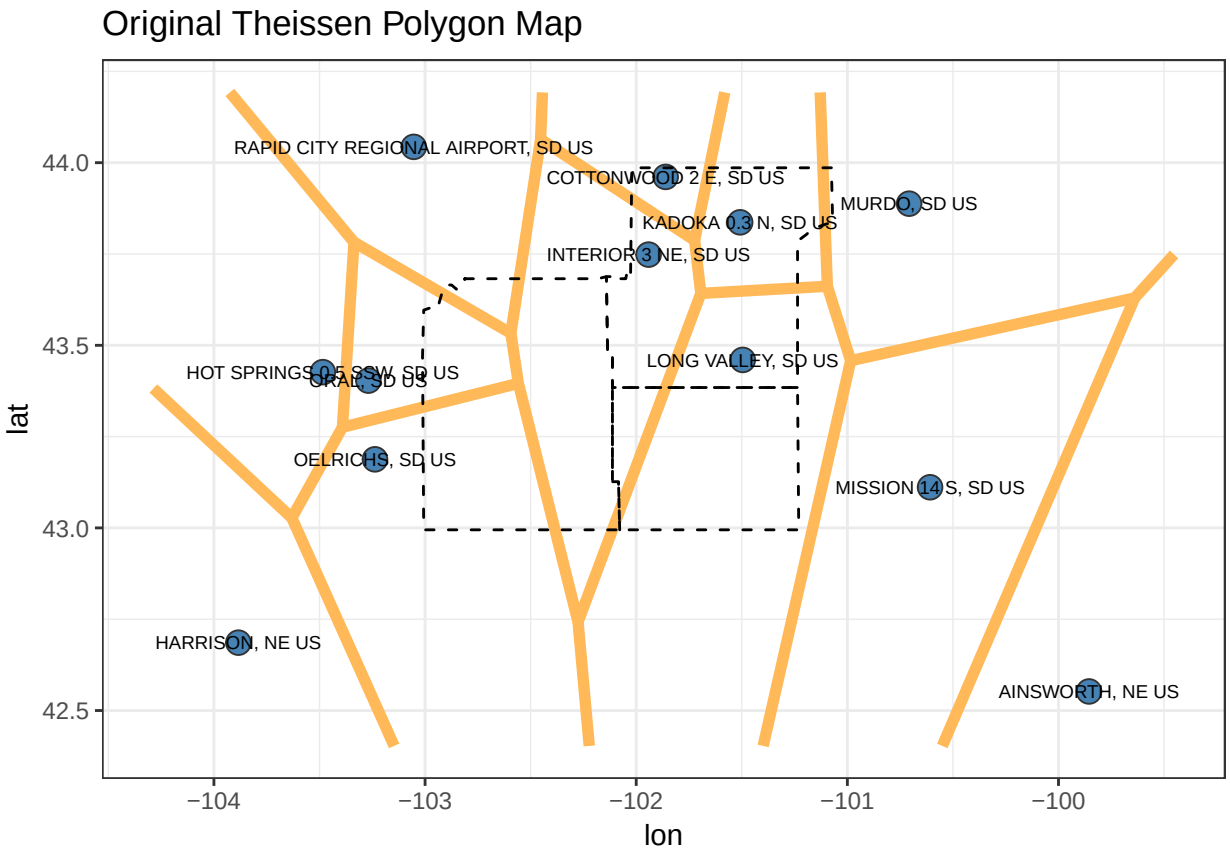
```
                                "White Clay Creek II",
                                "White Clay Creek III",
                                "White River I",
                                "White River II",
                                "White River III",
                                "White River IV",
                                "White River V",
                                "Wolf Creek I",
                                "Wounded Knee I",
                                "Wounded Knee II",
                                "Wounded Knee III",
                                "Wounded Knee IV"),
                      stringsAsFactors = FALSE)

site <- full_join(site, site_id, by = "name")

site <- site %>%
  drop_na()

site <- full_join(site, eco, by="id")

site <- site %>%
  replace_na(list(ecoregion = "Tablelands")) %>%
  filter(id != "CHR1") %>%
  filter(id != "CHR2") %>%
  mutate(ecoregion = case_when(
    id == "WHR1" ~ "Tablelands",
    id == "WHR2" ~ "Badlands",
    id == "WHR3" ~ "Badlands",
    id == "WHR4" ~ "Badlands",
    id == "WHR5" ~ "Badlands",
    TRUE ~ as.character(ecoregion)))

rm(site_id, eco)
# export(site, "data/site_meta.csv")
```

```
# import location data
sta_meta <- import("data/sta_meta_orig.csv")
sta_meta <- sta_meta %>%
  filter(id != "GHCND:US1SDCS0027") # filters out hermosa (short rec.)

# define the study area using data from the 'maps' package
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# import polygon data - counties
counties <- map_data("county")
counties <- subset(counties, region %in%
   c("south dakota", "nebraska"))
prr <- subset(counties, subregion %in%
   c("shannon", "jackson", "bennett"))

# create voroni line segments
voronoi <- deldir(sta_meta$lon, sta_meta$lat)
```

```
##
```

```
##      PLEASE NOTE:  The components "delsgs" and "summary" of the
##  object returned by deldir() are now DATA FRAMES rather than
##  matrices (as they were prior to release 0.0-18).
##  See help("deldir").
##
##      PLEASE NOTE: The process that deldir() uses for determining
##  duplicated points has changed from that used in version
##  0.0-9 of this package (and previously). See help("deldir").
```

```
#Plot the points, voronoi lines, and annotate
# grabbed this code off of a website by googling 'Voroni diag. R'
ggplot(data = sta_meta, aes(x = lon, y = lat)) +
  geom_point(
    fill = rgb(70, 130, 180, 255, maxColorValue = 255),
    pch = 21,
    size = 4,
    color = "#333333") +
  geom_segment(
    aes(x = x1, y = y1, xend = x2, yend = y2),
    size = 2,
    data = voronoi$dirsgs,
    linetype = 1,
    color = "#FFB958") +
  geom_polygon(data = prr, aes(x = long, y = lat, group = group),
               color = "black", linetype = "dashed", fill = "NA") +
  theme_bw() +
  geom_text(data = sta_meta, aes(label = name), size = 2.5) +
  ggtitle("Original Theissen Polygon Map")
```




```
# Results:
# We can drop Ainsworth, Harrison that are outside of study area &
# Oral would be better than Hot Springs for elev, location & coverage

#ggplot2::ggsave(filename = "theissen_init.png",
#                width = 6, height = 6, units = "in")
```

```
# DL Global Historical Climatology Network (GHCN) Daily Data
# Note: to flatten a list use: Reduce(rbind, list)

# load metadata
sta_meta <- import("data/sta_meta_orig.csv")

# make a table for manual import
# sta <- as.tibble(select(station_meta, 1:2))

# id                 name
# <chr>              <chr>
# GHCND:USW00024090 RAPID CITY REGIONAL AIRPORT, SD US
# GHCND:US1SDFR0001 HOT SPRINGS 0.5 SSW, SD US
# GHCND:USC00396212 OELRICHS, SD US
# GHCND:USC00394184 INTERIOR 3 NE, SD US
# GHCND:USC00391972 COTTONWOOD 2 E, SD US
# GHCND:USC00394983 LONG VALLEY, SD US
# GHCND:USC00253615 HARRISON, NE US
# GHCND:USC00250050 AINSWORTH, NE US
# GHCND:USC00395891 MURDO, SD US
```

```r
# GHCND:USC00395638 MISSION 14 S, SD US
# GHCND:US1SDCS0027 HERMOSA 10.3 ESE, SD US
# GHCND:USC00396304 ORAL, SD US
# GHCND:US1SDJK0006 KADOKA 0.3 N, SD US

# download data from NOAA
sta_rap <- meteo_tidy_ghcnd('USW00024090', keep_flags = FALSE,
                            var = "PRCP")
sta_hot <- meteo_tidy_ghcnd('US1SDFR0001', keep_flags = FALSE,
                            var = "PRCP")
sta_oel <- meteo_tidy_ghcnd('USC00396212', keep_flags = FALSE,
                            var = "PRCP")
sta_int <- meteo_tidy_ghcnd('USC00394184', keep_flags = FALSE,
                            var = "PRCP")
sta_cot <- meteo_tidy_ghcnd('USC00391972', keep_flags = FALSE,
                            var = "PRCP")
sta_lon <- meteo_tidy_ghcnd('USC00394983', keep_flags = FALSE,
                            var = "PRCP")
sta_har <- meteo_tidy_ghcnd('USC00253615', keep_flags = FALSE,
                            var = "PRCP")
sta_ain <- meteo_tidy_ghcnd('USC00250050', keep_flags = FALSE,
                            var = "PRCP")
sta_mur <- meteo_tidy_ghcnd('USC00395891', keep_flags = FALSE,
                            var = "PRCP")
sta_mis <- meteo_tidy_ghcnd('USC00395638', keep_flags = FALSE,
                            var = "PRCP")
sta_her <- meteo_tidy_ghcnd('US1SDCS0027', keep_flags = FALSE,
                            var = "PRCP")
sta_ora <- meteo_tidy_ghcnd('USC00396304', keep_flags = FALSE,
                            var = "PRCP")
sta_kad <- meteo_tidy_ghcnd('US1SDJK0006', keep_flags = FALSE,
                            var = "PRCP")

# save the data
export(sta_rap, file = "data/sta_rap.csv")
export(sta_hot, file = "data/sta_hot.csv")
export(sta_oel, file = "data/sta_oel.csv")
export(sta_int, file = "data/sta_int.csv")
export(sta_cot, file = "data/sta_cot.csv")
export(sta_lon, file = "data/sta_lon.csv")
export(sta_har, file = "data/sta_har.csv")
export(sta_ain, file = "data/sta_ain.csv")
export(sta_mur, file = "data/sta_mur.csv")
export(sta_mis, file = "data/sta_mis.csv")
export(sta_her, file = "data/sta_her.csv")
export(sta_ora, file = "data/sta_ora.csv")
export(sta_kad, file = "data/sta_kad.csv")
```

```r
# load metadata
sta_meta <- import("data/sta_meta_fin.csv")
sta_meta_orig <- as.tibble(import("data/sta_meta_orig.csv"))

# these were downloaded from NOAA on 2018-06-01
```

```
#   dropped in the subsequent analysis
# sta_her <- import(file = "data/sta_her.csv") # n = 304 obs so drop
# sta_hot <- import(file = "data/sta_hot.csv") # dropped by Theissen
# sta_lon <- import(file = "data/sta_lon.csv") # dropped by Theissen
# sta_har <- import(file = "data/sta_har.csv") # dropped by Theissen
# sta_ain <- import(file = "data/sta_ain.csv") # dropped by Theissen
# sta_kad <- import(file = "data/sta_kad.csv") # dropped by Theissen

# import prior saved files
sta_oel <- import(file = "data/sta_oel.csv") #
sta_cot <- import(file = "data/sta_cot.csv") #
sta_rap <- import(file = "data/sta_rap.csv") # needs fixed & saved
sta_int <- import(file = "data/sta_int.csv") #
sta_mis <- import(file = "data/sta_mis.csv") #
sta_ora <- import(file = "data/sta_ora.csv") #
sta_mur <- import(file = "data/sta_mur.csv") # need for NA vals


# join tables and rename prcp cols - data is in mm
# going from oldest to youngest; 'date' is chr
merge_sta1 <- full_join(sta_oel, sta_cot, by = "date")
merge_sta1 <- merge_sta1 %>%
  rename(oel = prcp.x) %>%
  rename(cot = prcp.y) %>%
  select(-id.x, -id.y)
rm(sta_oel, sta_cot)

merge_sta2 <- full_join(merge_sta1, sta_rap, by = "date")
merge_sta2 <- merge_sta2 %>%
  rename(rap = prcp) %>%
  select(-id)
rm(sta_rap)

merge_sta3 <- full_join(merge_sta2, sta_int, by = "date")
merge_sta3 <- merge_sta3 %>%
  rename(int = prcp) %>%
  select(-id)
rm(sta_int)

merge_sta4 <- full_join(merge_sta3, sta_mis, by = "date")
merge_sta4 <- merge_sta4 %>%
  rename(mis = prcp) %>%
  select(-id)
rm(sta_mis)

merge_sta5 <- full_join(merge_sta4, sta_ora, by = "date")
merge_sta5 <- merge_sta5 %>%
  rename(ora = prcp) %>%
  select(-id)
rm(sta_ora)

# final join & fix date & add year and month
sta <- full_join(merge_sta5, sta_mur, by = "date")
```

```r
sta <- sta %>%
  select(-starts_with("id")) %>%
  rename(mur = prcp)  %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

rm(sta_mur, merge_sta1, merge_sta2, merge_sta3, merge_sta4,
   merge_sta5)

# export(sta, file = "data/stations.csv")
```

```r
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Oral

# Variable naming convention:
# sta           precipitation station
# _har          Harrision precip station - used to fill NA vals
# _meta         metadata
# _raw          the "mostly" raw dataset
# _geXX         data greater than or equal to year XX
# _geXXmYY      data greater than or equal to year XX and month YY
# _ltXXmYY      data less than year XX and month YY
# _clean        intermediate df - clean part of NA split ;-}
# _dirty        intermediate df - NA part of NA split ;-}
# _transZ        final df - after cleaning

# NAs are caused in part by different start dates
# OELRICHS    1893 - 2018  -fixed with Murdo
# COTTONWOOD  1909 - 2018  -fixed with Interior & Murdo
# RAPID CITY  1948 - 2018  -fixed with Interior
# INTERIOR    1949 - 2018  -fixed with Cottonwood & Murdo
# ORAL        1971 - 2018  -fixed with Oelrichs & Harrison
# LONG VALLEY 1927 - 2012  - Removed from further analysis
# MISSION     1951 - 2018  - Removed from further analysis

# load metadata & data
sta_meta  <- as.tibble(import("data/sta_meta_fin.csv"))
sta_raw   <- import("data/stations.csv") # N = 45,784
sta_har   <- as.tibble(import("data/sta_har.csv"))
  sta_har <- sta_har %>%
    mutate(date = ymd(date)) # harrison used to fill NA

# fix date & add year and month
sta_raw <- sta_raw %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())
```

```r
# remove NA in 2018 data - old code not listed in naming convention
# chopping off everything after 2018-05-31; N= 45,684
sta_not_2018    <- sta_raw %>% filter(year != 2018)
  sta_2018_filt <- sta_raw %>% filter(year == 2018 & month != 6)
  sta_raw       <- bind_rows(sta_not_2018, sta_2018_filt) %>%
    arrange(desc(date))
rm(sta_2018_filt, sta_not_2018)

# check NA in original
intro_raw <- as.tibble(introduce(sta_raw)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_raw

# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840


# NAs are caused in part by different start dates
# OELRICHS    1893 - 2018
# COTTONWOOD  1909 - 2018
# RAPID CITY  1948 - 2018
# INTERIOR    1949 - 2018
# MISSION     1951 - 2018
# ORAL        1971 - 2018 - in.progress


# Clean Oral & part of Oelrichs 1971-05-01 to present
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1971-05-01
  sta_ge72      <- sta_raw %>% filter(year >= 1972) #
  sta_71m05     <- sta_raw %>% filter(year == 1971 & month >= 5)
sta_ge71m05     <- bind_rows(sta_ge72, sta_71m05) # this is active
  rm(sta_ge72, sta_71m05)

  sta_le71      <- sta_raw %>% filter(year < 1971)
  sta_71m01_m05 <- sta_raw %>% filter(year == 1971 & month < 5)
sta_lt71m05     <- bind_rows(sta_le71, sta_71m01_m05) # this is not
  rm(sta_le71, sta_71m01_m05)

#   2. filter NA vals from Oral
#     Before = 368 NA <- filling with Oelrichs
#     After: 10 concurrent NA values

sta_clean <- sta_ge71m05 %>%
  filter(!is.na(ora)) # this is not active

sta_dirty <- sta_ge71m05 %>%
  filter(is.na(ora)) %>%
  mutate(ora = oel) # fix oral with oelrichs
```

```r
sta_ge71m05 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge71m05 %>%
  filter(is.na(ora)) # check is :-] NA is 1 obs

#   3. filter NA vals from Oelrichs (oel)
#     Before = 368 NA <- filling with Oral
#     After: 10 concurrent NA values

sta_clean <- sta_ge71m05 %>%
  filter(!is.na(oel)) # this is not active

sta_dirty <- sta_ge71m05 %>%
  filter(is.na(oel)) %>%
  mutate(oel = ora) # fix oelrichs with oral

sta_ge71m05 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge71m05 %>%
   filter(is.na(ora)) # check is :-] down to 1 NA
rm(sta_NA)

#   4. fix Oelrichs & Oral NA with Harrision
#     Before: 1 concurrent NA values

sta_ge71m05 <- left_join(sta_ge71m05, sta_har, by = "date")
  rm(sta_har)

sta_clean <- sta_ge71m05 %>%
  filter(!is.na(oel)) # this is not active

sta_dirty <- sta_ge71m05 %>%
  filter(is.na(oel)) %>%
  mutate(oel = prcp) %>%
  mutate(ora = prcp)

sta_ge71m05 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge71m05 %>%
   filter(is.na(ora)) # check is :-] zero NA vals
rm(sta_NA)

#   4. Put the pieces back together
sta_ge71m05 <- sta_ge71m05 %>%
  select(-prcp, -id)
sta_trans <- bind_rows(sta_ge71m05, sta_lt71m05)
rm(sta_ge71m05,sta_lt71m05)

# Check work
intro_raw
```

```
#A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

intro_trans <- as.tibble(introduce(sta_trans)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans

# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# export the work to a file
# export(sta_trans, file = "data/stations_trans1.csv")
```

```
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Mission
# Variable naming convention - see munge-precip-data-oral code chunk


# load metadata & data
sta_meta  <- as.tibble(import("data/sta_meta_fin.csv"))
sta_trans <- import("data/stations_trans1.csv")
sta_lon   <- import("data/sta_lon.csv")

# fix date & add year and month
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

sta_lon <- sta_lon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  select(date, everything())


# Clean Mission using Long Valley station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1951-08-01
  sta_ge52      <- sta_trans %>% filter(year >= 1952)
  sta_51m08     <- sta_trans %>% filter(year == 1951 & month >= 8)
sta_ge51m08     <- bind_rows(sta_ge52, sta_51m08) # this is active
  rm(sta_ge52, sta_51m08)

  sta_le51      <- sta_trans %>% filter(year < 1951)
  sta_51m01_m08 <- sta_trans %>% filter(year == 1951 & month < 8)
```

```r
sta_lt51m08     <- bind_rows(sta_le51, sta_51m01_m08) # this is not
  rm(sta_le51, sta_51m01_m08)

#   2. Attach Long Valley to df
sta_ge51m08 <- left_join(sta_ge51m08, sta_lon, by = "date")
  rm(sta_lon)

#   3. filter NA vals from Mission with Long Valley
#     Before = 858 NA <- filling with Long Valley
#     After: 14 NA values
sta_clean <- sta_ge51m08 %>%
  filter(!is.na(mis)) # this is not active

sta_dirty <- sta_ge51m08 %>%
  filter(is.na(mis)) %>%
  mutate(mis = prcp) # fix Mission with Long Valley

sta_ge51m08 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge51m08 %>%
  filter(is.na(mis)) # check is :-] # down to 6 vals
rm(sta_NA)

#   4. Use the Interior vals to fill missing vals
#     Before = 6 NA <- filling with Interior
#     After: XX  NA values

sta_clean <- sta_ge51m08 %>%
  filter(!is.na(mis)) # this is not active

sta_dirty <- sta_ge51m08 %>%
  filter(is.na(mis)) %>%
  mutate(mis = int) # fix Mission with Interior

sta_ge51m08 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge51m08 %>%
   filter(is.na(mis)) # check is :-] down to 2 NA vals
 rm(sta_NA)

#   5. Use the Oelrich vals to fill missing vals
#     Before = 2 NA <- filling with Interior
#     After: zero NA values

sta_clean <- sta_ge51m08 %>%
  filter(!is.na(mis)) # this is not active

sta_dirty <- sta_ge51m08 %>%
  filter(is.na(mis)) %>%
  mutate(mis = oel) # fix Mission with Oelrichs
```

```r
sta_ge51m08 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge51m08 %>%
   filter(is.na(mis)) # check is :-] down to 2 NA vals
 rm(sta_NA)

#   6. Put the pieces back together
sta_ge51m08 <- sta_ge51m08 %>%
  select(-prcp, -id)
sta_trans2 <- bind_rows(sta_ge51m08, sta_lt51m08)
rm(sta_ge51m08,sta_lt51m08)

# Check work
# A tibble: 1 x 4 (RAW)
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

intro_trans <- as.tibble(introduce(sta_trans)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans
# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

intro_trans2 <- as.tibble(introduce(sta_trans2)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans2
# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

# export the work to a file
# export(sta_trans2, file = "data/stations_trans2.csv")
```

```r
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Interior
# Variable naming convention - see munge-precip-data-oral code chunk


# load metadata & data
sta_meta  <- as.tibble(import("data/sta_meta_fin.csv"))
sta_trans <- import("data/stations_trans2.csv")
sta_mur   <- import("data/sta_mur.csv")

# fix date & add year and month
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
```

```r
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

sta_mur <- sta_mur %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

# Clean Interior using Cottonwood station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1949-11-01
  sta_ge50      <- sta_trans %>% filter(year >= 1950)
  sta_49m11     <- sta_trans %>% filter(year == 1949 & month >= 11)
sta_ge49m11     <- bind_rows(sta_ge50, sta_49m11) # this is active
  rm(sta_ge50, sta_49m11)

  sta_le50      <- sta_trans %>% filter(year < 1949)
  sta_49m01_m11 <- sta_trans %>% filter(year == 1949 & month < 11)
sta_lt49m11     <- bind_rows(sta_le50, sta_49m01_m11) # this is not
  rm(sta_le50, sta_49m01_m11)

#   2. filter NA vals from Interior with Cottonwood
#     Before = 2,578 NA <- filling with Cottonwood
#     After: 11 NA values
sta_clean <- sta_ge49m11 %>%
  filter(!is.na(int)) # this is not active

sta_dirty <- sta_ge49m11 %>%
  filter(is.na(int)) %>%
  mutate(int = cot) # fix Interior with Cottonwood

sta_ge49m11 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge49m11 %>%
  filter(is.na(int)) # check is :-] # down to 11 vals
  rm(sta_NA)

#   3. Fill NA with Rapid
#     Before: 11 NA <- filling with Murdo
#     After:  zero NA values

sta_clean <- sta_ge49m11 %>%
  filter(!is.na(int)) # this is not active

sta_dirty <- sta_ge49m11 %>%
  filter(is.na(int)) %>%
  mutate(int = rap) # fix Interior with Interior
```

```r
sta_ge49m11 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge49m11 %>%
   filter(is.na(int)) # check is :-] down to zero vals
 rm(sta_NA)

#   4. filter NA vals from Cottonwood with Interior
#     Before: 850 NA <- filling with Interior
#     After: 11 NA values
sta_clean <- sta_ge49m11 %>%
  filter(!is.na(cot)) # this is not active

sta_dirty <- sta_ge49m11 %>%
  filter(is.na(cot)) %>%
  mutate(cot = int) # fix Cottonwood with Interior

sta_ge49m11 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge49m11 %>%
  filter(is.na(cot)) # check is :-] # down to zero vals
  rm(sta_NA)

#   3. Put the pieces back together
sta_trans3 <- bind_rows(sta_ge49m11, sta_lt49m11)
rm(sta_ge49m11, sta_lt49m11)

# Check work
# A tibble: 1 x 4 - Raw
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

# A tibble: 1 x 4 - Trans1
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# A tibble: 1 x 4 - Trans2
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

intro_trans3 <- as.tibble(introduce(sta_trans3)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans3

# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106210             456840
```

```
# export the work to a file
# export(sta_trans3, file = "data/stations_trans3.csv")
```

```
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Rapid City NA with Interior & Cottonwood
# Rationale - this is the closest to the project area;
#  remember the point estimates are for the project area.

# Variable naming convention - see munge-precip-data-oral code chunk

# load metadata & data
sta_meta  <- as.tibble(import("data/sta_meta_fin.csv"))
sta_trans <- import("data/stations_trans3.csv")

# fix date & add year and month
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

# Clean Rapid City using Interior & Cottonwood station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1948-05-01
  sta_ge49      <- sta_trans %>% filter(year >= 1949) # yr above
  sta_48m05     <- sta_trans %>% filter(year == 1948 & month >= 5)
sta_ge48m05     <- bind_rows(sta_ge49, sta_48m05) # this is active
  rm(sta_ge49, sta_48m05)

  sta_lt48     <- sta_trans %>% filter(year < 1948)
  sta_48m01_m05 <- sta_trans %>% filter(year == 1948 & month < 5)
sta_lt09m06     <- bind_rows(sta_lt48, sta_48m01_m05) # this is not
  rm(sta_lt48, sta_48m01_m05)

#   2. filter NA vals from Rapid City and fill with Interior
#     Before: 6 NA <- filling with Interior
#     After: zero NA values
sta_clean <- sta_ge48m05 %>%
  filter(!is.na(rap)) # this is not active

sta_dirty <- sta_ge48m05 %>%
  filter(is.na(rap)) %>%
  mutate(rap = int) # fix Rapid City with Interior

sta_ge48m05 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge48m05 %>%
  filter(is.na(rap)) # check is :-] # down to zero vals
  rm(sta_NA)
```

```
#   4. Put the pieces back together
#sta_ge48m05 <- sta_ge48m05 %>%
#  select(-prcp, -id)
sta_trans4 <- bind_rows(sta_ge48m05, sta_lt09m06)
rm(sta_ge48m05, sta_lt48m05)

# Check work
# A tibble: 1 x 4 - Raw
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

# A tibble: 1 x 4 - Trans1
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# A tibble: 1 x 4 - Trans2
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

# A tibble: 1 x 4 - Trans3
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106210             456840

intro_trans4 <- as.tibble(introduce(sta_trans4)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans4

# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106204             456840

# export the work to a file
# export(sta_trans4, file = "data/stations_trans4.csv")
```

```
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Cottonwood with Rapid City
# Rationale - Refactored - RC has similarity to Cottonwood

# Variable naming convention - see munge-precip-data-oral code chunk

# load metadata & data
sta_meta      <- as.tibble(import("data/sta_meta_fin.csv"))
sta_trans     <- import("data/stations_trans4.csv")
sta_meta_orig <- as.tibble(import("data/sta_meta_orig.csv"))

# fix date & add year and month
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
```

```
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

# Clean Cottonwood precip NA using Murdo station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1909-06-01
  sta_ge09      <- sta_trans %>% filter(year >= 1910) # yr above
  sta_09m06     <- sta_trans %>% filter(year == 1909 & month >= 6)
sta_ge09m06     <- bind_rows( sta_ge09, sta_09m06) # this is active
  rm(sta_ge09, sta_09m06)

  sta_lt09     <- sta_trans %>% filter(year < 1909)
  sta_09m01_m06 <- sta_trans %>% filter(year == 1909 & month < 6)
sta_lt09m06     <- bind_rows(sta_lt09, sta_09m01_m06) # this is not
  rm(sta_lt09, sta_09m01_m06)

#Check on split :-)
count_check <- bind_rows(sta_ge09m06, sta_lt09m06)
rm(count_check)

#   2. filter NA vals from Cottonwood & fill with Murdo
#     Before: 96 NA <- filling with Murdo
#     After: 12 NA values
sta_clean <- sta_ge09m06 %>%
  filter(!is.na(cot)) # this is not active

sta_dirty <- sta_ge09m06 %>%
  filter(is.na(cot)) %>%
  mutate(cot = mur) # fix Cottonwood with Murdo

sta_ge09m06 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge09m06 %>%
  filter(is.na(cot)) # check is :-] # down to 12 vals
  rm(sta_NA)

# ~~~~~~~~~~~~~~~~~~~~~~~
#   3. filter NA vals from Cottonwood & fill with Oelrichs
#     Before: 12 NA <- filling with Murdo
#     After: zero NA values
sta_clean <- sta_ge09m06 %>%
  filter(!is.na(cot)) # this is not active

sta_dirty <- sta_ge09m06 %>%
  filter(is.na(cot)) %>%
  mutate(cot = oel) # fix Cottonwood with Murdo

sta_ge09m06 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)
```

```r
sta_NA <- sta_ge09m06 %>%
  filter(is.na(cot)) # check is :-] # down to zero vals
  rm(sta_NA)

#   4. Put the pieces back together
sta_trans5 <- bind_rows(sta_ge09m06, sta_lt09m06)
rm(sta_ge09m06, sta_lt09m06)

# Check work
# A tibble: 1 x 4 - Raw
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

# A tibble: 1 x 4 - Trans1
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# A tibble: 1 x 4 - Trans2
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

# A tibble: 1 x 4 - Trans3
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106210             456840

# A tibble: 1 x 4 - Trans4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106204             456840

intro_trans5 <- as.tibble(introduce(sta_trans5)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans5

# A tibble: 1 x 4 - Trans5
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106108             456840

# export the work to a file
# export(sta_trans5, file = "data/stations_trans5.csv")
```

```r
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - clean Oelrichs with Murdo &
#   define oldest date with continuous data

# Variable naming convention - see munge-precip-data-oral code chunk
```

```r
# load metadata & data
sta_meta      <- as.tibble(import("data/sta_meta_fin.csv"))
sta_meta_orig <- as.tibble(import("data/sta_meta_orig.csv"))
sta_trans     <- import("data/stations_trans5.csv")

# fix date & add year and month
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

# Clean Oelrichs precip NA using Murdo station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   In this case it is the filler, Murdo, rather than the recipient
#   1. Split the raw data into two parts at 1907-12-01
  sta_ge08      <- sta_trans %>% filter(year >= 1908) # yr above
  sta_07m12     <- sta_trans %>% filter(year == 1907 & month >= 12)
sta_ge07m12     <- bind_rows( sta_ge08, sta_07m12) # this is active
  rm(sta_ge08, sta_07m12)

  sta_lt07     <- sta_trans %>% filter(year < 1907)
  sta_07m1_m12 <- sta_trans %>% filter(year == 1907 & month < 12)
sta_lt07m12     <- bind_rows(sta_lt07, sta_07m1_m12) # this is not
  rm(sta_lt07, sta_07m1_m12)

#Check on split :-)
count_check <- bind_rows(sta_ge07m12, sta_lt07m12)
rm(count_check)

#   2. filter NA vals from Oelrichs & fill with Murdo
#     Before: 1478 NA <- filling with Murdo
#     After: 186 NA values
sta_clean <- sta_ge07m12 %>%
  filter(!is.na(oel)) # this is not active

sta_dirty <- sta_ge07m12 %>%
  filter(is.na(oel)) %>%
  mutate(oel = mur) # oelrichs with Murdo

sta_ge07m12 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

  #Check on split :-)
count_check <- bind_rows(sta_ge07m12, sta_lt07m12)
rm(count_check)

sta_NA <- sta_ge07m12 %>%
  filter(is.na(oel)) %>%
  arrange() # check is :-] # down to 186 vals; fill with Cottonwood
rm(sta_NA)
```

```
#   3. filter NA vals from Oelrichs & fill with Cottonwood
#     Before: 186 NA <- filling with Cottonwood
#     After: 22 NA values
sta_clean <- sta_ge07m12 %>%
  filter(!is.na(oel)) # this is not active

sta_dirty <- sta_ge07m12 %>%
  filter(is.na(oel)) %>%
  mutate(oel = cot) # oelrichs with Cottonwood

sta_ge07m12 <- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

sta_NA <- sta_ge07m12 %>%
  filter(is.na(oel)) %>%
  arrange() # check is :-] # down to 22 vals;
# Oldest is 1909-05-28, so this is the end of the record
# The end of the complete record is 1909-06-01

#   4. Put the pieces back together & cut to end of complete record
sta_trans6 <- bind_rows(sta_ge07m12, sta_lt07m12)
rm(sta_ge07m12, sta_lt07m12, sta_NA)

# Check work
# A tibble: 1 x 4 - Raw
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

# A tibble: 1 x 4 - Trans1
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# A tibble: 1 x 4 - Trans2
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

# A tibble: 1 x 4 - Trans3
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106210             456840

# A tibble: 1 x 4 - Trans4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106204             456840

# A tibble: 1 x 4 - Trans5
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106108             456840
```

```r
intro_trans6 <- as.tibble(introduce(sta_trans6)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans6

# A tibble: 1 x 4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               104652             456840

# export the work to a file
# export(sta_trans6, file = "data/stations_trans6.csv")
```

```r
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - Append Long Valley with Mission
# Variable naming convention - see munge-precip-data-oral code chunk

# load metadata & data
# Note: the metadata was changed in this step.  A once only change.
#sta_meta      <- as.tibble(import("data/sta_meta_fin.csv"))
#sta_meta_orig <- as.tibble(import("data/Archived/station_meta2.csv")
sta_trans     <- as.tibble(import("data/stations_trans6.csv"))
sta_lon       <- as.tibble(import("data/sta_lon.csv"))

# add Long Valley & remove Mission metadata
#sta_meta_lon <- sta_meta_orig %>%
#  filter(name == "LONG VALLEY, SD US")
#sta_meta2    <- bind_rows(sta_meta, sta_meta_lon)
#sta_meta2    <- sta_meta2 %>%
#  filter(name != "MISSION 14 S, SD US")
#export(sta_meta2, file = "data/sta_meta_fin2.csv")
sta_meta      <- as.tibble(import("data/sta_meta_fin2.csv"))


# fix date, add year and month, and join Long Valley
sta_trans <- sta_trans %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

sta_lon <- sta_lon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  select(date, everything())

sta_trans <- left_join(sta_trans, sta_lon)
rm(sta_lon)

sta_trans <- sta_trans %>%
  select(-id) %>%
  rename(lon = prcp)
```

```
# Clean Long Valley precip NA using Mission station data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Using nearest year as split-point
#   1. Split the raw data into two parts at 1927-07-01
  sta_ge28      <- sta_trans %>% filter(year >= 1928) # yr above
  sta_27m07    <- sta_trans %>% filter(year == 1927 & month >= 7)
sta_ge27m07    <- bind_rows( sta_ge28, sta_27m07) # this is active
  rm(sta_ge28, sta_27m07)

  sta_lt27     <- sta_trans %>% filter(year < 1927)
  sta_27m1_m7 <- sta_trans %>% filter(year == 1927 & month < 7)
sta_lt27m07     <- bind_rows(sta_lt27, sta_27m1_m7) # this is not
  rm(sta_lt27, sta_27m1_m7)

#Check on split :-)
count_check <- bind_rows(sta_ge27m07, sta_lt27m07)
rm(count_check)

#   2. filter NA vals from Long Valley & fill with Mission
#     Before: 2904 NA <- filling with Mission
#     After: 423 NA values
sta_clean <- sta_ge27m07%>%
  filter(!is.na(lon)) # this is not active

sta_dirty <- sta_ge27m07%>%
  filter(is.na(lon)) %>%
  mutate(lon = mis) # Long Valley with Mission

sta_ge27m07<- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)

# Check on split :-)
count_check <- bind_rows(sta_ge27m07, sta_lt27m07)
rm(count_check)

sta_NA <- sta_ge27m07 %>%
  filter(is.na(lon)) %>%
  arrange() # check is :-] # down to 423 vals; fill with Cottonwood
rm(sta_NA)

#   3. filter NA vals from Long Valley & fill with Cottonwood
#     Before: 423 NA <- filling with Mission
#     After: zero NA values
sta_clean <- sta_ge27m07%>%
  filter(!is.na(lon)) # this is not active

sta_dirty <- sta_ge27m07%>%
  filter(is.na(lon)) %>%
  mutate(lon = cot) # Long Valley with Mission

sta_ge27m07<- bind_rows(sta_clean, sta_dirty)
  rm(sta_clean, sta_dirty)
```

```
  #Check on split :-)
count_check <- bind_rows(sta_ge27m07, sta_lt27m07)
rm(count_check)

sta_NA <- sta_ge27m07 %>%
  filter(is.na(lon)) %>%
  arrange() # check is :-] # down to zero vals

#   4. Put the pieces back together & cut to end of complete record
sta_trans7 <- bind_rows(sta_ge27m07, sta_lt27m07)
rm(sta_ge27m07, sta_lt27m07, sta_NA)

# Check work
# A tibble: 1 x 4 - Raw
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               111229             456840

# A tibble: 1 x 4 - Trans1
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               110496             456840

# A tibble: 1 x 4 - Trans2
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               109638             456840

# A tibble: 1 x 4 - Trans3
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106210             456840

# A tibble: 1 x 4 - Trans4
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106204             456840

# A tibble: 1 x 4 - Trans5
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               106108             456840

# A tibble: 1 x 4 - Trans6
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      10               104652             456840

intro_trans7 <- as.tibble(introduce(sta_trans7)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_trans7
```

```
# A tibble: 1 x 4 - Trans7 - note the extra column
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 45684      11               117128             502524

# export the work to a file
# export(sta_trans7, file = "data/stations_trans7.csv")

# This is to cut back to complete part of Oelrichs
  sta_ge10      <- sta_trans7 %>% filter(year >= 1910) # yr above
  sta_09m06     <- sta_trans7 %>% filter(year == 1909 & month >= 6)
sta_fin     <- bind_rows( sta_ge10, sta_09m06) # this is active
  rm(sta_ge10, sta_09m06)

intro_sta_fin <- as.tibble(introduce(sta_fin)) %>%
  select(-(3:5)) %>%
  select(-5)
intro_sta_fin

# A tibble: 1 x 4 - Final
#   rows columns total_missing_values total_observations
#  <int>   <int>                <int>              <int>
# 39812      11                76046             437932

# export(sta_fin, file = "data/stations_final.csv")
```

```
# General Purpose: fill NA prior to upscaling to monthly data
# Specific purpose - Check Mission & Long Valley covariance
# Variable naming convention - see munge-precip-data-oral code chunk

# load metadata & data
sta_meta      <- as.tibble(import("data/sta_meta_fin2.csv"))
sta_fin     <- as.tibble(import("data/stations_final.csv"))

# Split data to the end of Mission - the purpose here is
#   to look at a double mass plot with Mission & Long Valley

sta_test <- sta_fin %>%
  filter(year > 1951) %>%
  filter(year < 2012)

# gather values & create groups
sta_gath <- gather(sta_test, key = "station", value = "prcp", -date,
                   -year, -month, factor_key = TRUE)

sta_group <- sta_gath %>%
  group_by(year, month, station)

# sum daily precip over a month
sta_gath_mon <- sta_group %>%
  summarize(prcp_tenths = sum(prcp)) %>%
  mutate(prcp_mm = prcp_tenths/10) %>%
  select(-prcp_tenths)
```

```r
# spread result - now in months
sta_mon <- sta_gath_mon %>%
  spread(station, prcp_mm) %>%
  mutate(day = 1) %>%
  mutate(date = make_date(year = year, month = month, day = day)) %>%
  select(date, year, month, everything()) %>%
  select(-day) %>%
  ungroup()
rm(sta_gath, sta_group, sta_gath_mon)

# filter Mission, Long Valley, Interior
sta_gath2 <- gather(sta_mon, key = "station", value = "prcp", -date,
                    -year, -month, -lon, factor_key = TRUE) %>%
  filter(station == "int" |
           station == "mis")

# plot the graphs of Interior and Mission
# see which precip is more similiar.

ggplot(sta_gath2, aes(prcp, lon)) +
  geom_point() +
  geom_smooth(method = "lm", aes(color = "red")) +
  facet_grid(.~station) +
   scale_x_sqrt() +
  scale_y_sqrt() +
  geom_smooth() +
  ggtitle("Long Valley similarity to nearest stations")
```

```r
# continued from above
lm_both <- lm(data = sta_mon, sqrt(lon) ~ sqrt(mis) + sqrt(int))
lm_both_tidy <- tidy(lm_both)
lm_both_glance <- glance(lm)

lm_int <- lm(data = sta_mon, sqrt(lon) ~ sqrt(int))
lm_int_tidy <- tidy(lm_int)
lm_int_glance <- glance(lm_int)

lm_mis <- lm(data = sta_mon, sqrt(lon) ~ sqrt(mis))
lm_mis_tidy <- tidy(lm_mis)
lm_mis_glance <- glance(lm_mis)

# combine models
lm_glance <- bind_rows(lm_both_glance, lm_int_glance)
lm_glance <- bind_rows(lm_glance, lm_mis_glance)

lm_glance <- as.tibble(lm_glance) %>%
  mutate(name = c("both", "int", "mis")) %>%
  select(name, everything())
lm_glance
```

```r
# continued from above
# result: drop the long valley data and use Interior for southeast
# remove not-needed stations from metadata
sta_meta <- sta_meta %>%
```

```
  filter(name != "LONG VALLEY, SD US")

# export(sta_meta, file = "data/sta_meta_fin3.csv")
```

```
# continued from above
# General Purpose: prepare data for drought index
# Specific purpose: convert daily precip to monthly precip

# load metadata & data
sta_meta  <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_day   <- as.tibble(import("data/stations_final2.csv"))

# remove Murdo, Mission, Long Valley - see above
#sta_day   <- sta_day %>%
#  select(-c(lon, mur, mis))
#export(sta_day, file = "data/stations_final2.csv")

# fix date & add year and month
sta_day <- sta_day %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  mutate(year = year(date)) %>%
  mutate(month = month(date)) %>%
  select(date, year, month, everything())

# gather daily values
sta_gath <- gather(sta_day, key = "station", value = "prcp", -date,
                   -year, -month, factor_key = TRUE)

# create groups
sta_group <- sta_gath %>%
  group_by(year, month, station)

# sum daily precip over a month
sta_gath_mon <- sta_group %>%
  summarize(prcp_tenths = sum(prcp)) %>%
  mutate(prcp_mm = prcp_tenths/10) %>%
  select(-prcp_tenths)

# spread result - now in months
#   ...and take a bow, because this is MAGIC!  Thnx Tidyverse.
sta_mon <- sta_gath_mon %>%
  spread(station, prcp_mm) %>%
  mutate(day = 1) %>%
  mutate(date = make_date(year = year, month = month, day = day)) %>%
  select(date, year, month, everything()) %>%
  select(-day) %>%
  ungroup()

rm(sta_day, sta_gath, sta_gath_mon, sta_group)
# export(sta_mon, file = "data/stations_monthly.csv")
```

```
# General Purpose: check a short-term record: Oral
```

```r
# Variable naming convention - see munge-precip-data-oral code chunk

# load metadata & data
sta_meta    <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_mon     <- as.tibble(import("data/stations_monthly.csv"))

# Split data to the end of Oral - the purpose here is
#   to look at a double mass plot with Oelrichs

sta_test <- sta_mon %>%
  filter(year > 1971)

# check a linear model
lm <- lm(data = sta_mon, sqrt(ora) ~ sqrt(oel))
lm.tidy <- tidy(lm)
lm.glance <- glance(lm)

# plot the graphs of Oral & Oelrichs

ggplot(sta_test, aes(ora, oel)) +
  geom_point() +
  geom_smooth(method = "lm", aes(color = "red")) +
   scale_x_sqrt() +
  scale_y_sqrt() +
  geom_smooth() +
  theme_bw() +
  ggtitle("Double mass plots")
# it's ok, keep Oral...
```

```r
# General Purpose: prepare data for drought index
# Specific purpose: graphical EDA

sta_meta    <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_mon     <- as.tibble(import("data/stations_monthly.csv"))

# fix date & add year and month
sta_mon <- sta_mon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date))

# gather monthly values & order them
sta_gath_mon <- gather(sta_mon, key = "station", value = "prcp",
                       -date, -year, -month, factor_key = TRUE)

# x$name <- factor(x$name, levels = x$name[order(x$val)])

# plot
ggplot(sta_gath_mon, aes(date, prcp)) +
  geom_line() +
  facet_grid(station ~ .) +
  theme_classic() +
  labs(title = "Monthly precipitation depths",
       subtitle = "Pine Ridge Reservation, SD for 1909-2018") +
       xlab("") +
```

```
       ylab("mm")

#ggplot2::ggsave(filename = "precip_mon.png",
#                 width = 6, height = 6, units = "in")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: convert monthly precip to yearly prcp

# load metadata & data
sta_meta    <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_mon     <- as.tibble(import("data/stations_monthly.csv"))

# fix date & add year and month
sta_mon <- sta_mon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  select(date, year, month, everything())

# gather monthly values
sta_gath <- gather(sta_mon, key = "station", value = "prcp", -date,
                   -year, -month, factor_key = TRUE)

# create groups
sta_group <- sta_gath %>%
  group_by(year,  station)

# sum monthly precip over a year
sta_gath_yr <- sta_group %>%
  summarize(prcp = sum(prcp))

# spread result - now in years
sta_yr <- sta_gath_yr %>%
  spread(station, prcp) %>%
  filter(year != 1909) %>%
  filter(year != 2018) %>%
  ungroup()

rm(sta_mon, sta_gath, sta_gath_yr, sta_group)
# export(sta_yr, file = "data/stations_yearly.csv")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: graphical EDA - yearly

# NEED TO FIX - screwed up variables

sta_meta    <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_yr     <- as.tibble(import("data/stations_yearly.csv"))

# gather monthly values
sta_gath <- gather(sta_yr, key = "station", value = "prcp",
                   -year, factor_key = TRUE)

# plot
ggplot(sta_gath, aes(year, prcp)) +
```

```
  geom_line() +
  facet_grid(station ~ .) +
  theme_classic() +
  labs(title = "Annual precipitation depths",
       subtitle = "Pine Ridge Reservation, SD for 1910-2017") +
       xlab("Date") +
       ylab("Depth, mm")

# ggplot2::ggsave(filename = "precip_yr.png",
#                 width = 6, height = 6, units = "in")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: create summaries of data
sta_meta  <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_yr    <- as.tibble(import("data/stations_yearly.csv"))

# gather and summarize yearly values
# Next step - do by water year???
sta_gath_yr <- gather(sta_yr, key = "station", value = "prcp",
                   -year, factor_key = TRUE)

sta_summary_yr <- as.tibble(sta_gath_yr) %>%
  group_by(station) %>%
  summarise(mean = mean(prcp, na.rm = TRUE),
            med = median(prcp, na.rm = TRUE),
            IQR = IQR(prcp, na.rm = TRUE),
            min = min(prcp, na.rm = TRUE),
            max = max(prcp, na.rm = TRUE)) %>%
  arrange(desc(med))

sta_summary_yr
# export(sta_summary_yr, file = "data/sta_summary_yr.csv")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: create summaries of data
sta_meta  <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_mon   <- as.tibble(import("data/stations_monthly.csv"))

# fix dates
sta_mon <- sta_mon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date)) %>%
  select(date, year, month, everything())

# gather and summarize monthly values
sta_gath_mon <- gather(sta_mon, key = "station", value = "prcp",
                       -date, -year, -month, factor_key = TRUE)

sta_summary_mon <- as.tibble(sta_gath_mon) %>%
  group_by(station, month) %>%
  summarise(mean = mean(prcp, na.rm = TRUE),
            med = median(prcp, na.rm = TRUE),
            IQR = IQR(prcp, na.rm = TRUE),
            min = min(prcp, na.rm = TRUE),
```

```
            max = max(prcp, na.rm = TRUE)) %>%
  arrange(month) %>%
  arrange(station)

sta_summary_mon
# export(sta_summary_mon, file = "data/sta_summary_mon.csv")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: graphical EDA
sta_meta    <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_mon     <- as.tibble(import("data/stations_monthly.csv"))

# fix date & add year and month
sta_mon <- sta_mon %>%
  mutate(date = ymd(date)) %>%
  arrange(desc(date))

# gather monthly values
sta_gath_mon <- gather(sta_mon, key = "station", value = "prcp",
                       -date, -year, -month, factor_key = TRUE)

ggplot(sta_gath_mon, aes(month, prcp, group = month)) +
  geom_boxplot() +
  facet_wrap(~station) +
  theme_classic() +
  labs(title = "Monthly precipitation depths",
       subtitle = "Pine Ridge Reservation, SD for 1910-2017") +
       xlab("Date") +
       ylab("Depth, mm")

#ggplot2::ggsave(filename = "precip_yr.png",
#                width = 6, height = 6, units = "in")
```

```
# General Purpose: prepare data for drought index
# Specific purpose: graphical EDA - correlation plot

sta_meta  <- as.tibble(import("data/sta_meta_fin3.csv"))
sta_yr   <- as.tibble(import("data/stations_yearly.csv"))

# fix date & add year and month
sta_yr <- sta_yr %>%
  arrange(year)

# need to have a correlation matrix without any NA vals
# gather yearly values
sta_gath <- gather(sta_yr, key = "station", value = "prcp",
                   -year, factor_key = TRUE)

# filter NAs
sta_gath_72 <- sta_gath %>%
  filter(year > 1972)

# spread remaining matrix & arrange from west to east
sta_72 <- sta_gath_72 %>%
```

```
  spread(station, prcp) %>%
  select(oel, ora, rap, int, cot)

# create a correlation matrix and plot it
sta_M <- cor(sta_72)
corrplot.mixed(sta_M,  order = "hclust", addrect = 2, upper = "ellipse", lower = "number", title = "Pre
```