Obtaining, cleaning and visualizing hydrological data with R

Alex Hurley
University of Birmingham

🏠 aglhurley.rbind.io

🐦 aglhurley

Using R
In
Hydrology

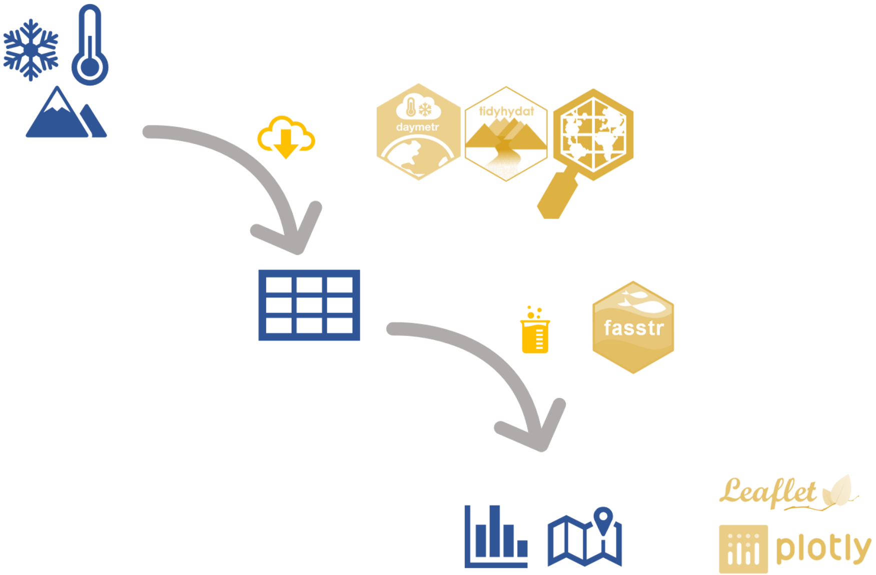
EGU 2019

Goals

- Introduce **useful packages**
- Highlight synergies
- Showcase capabilities (**processing**, **stats**)
- Visualize (**static** and **interacive** graphs)

Approach

- Download **hydrometric** and **ancillary** (gridded) data for stations defined with `AOI` in Glacier National Park (BC, Canada)
- Calculate *P-pET* (`raster`), *Runoff*, and flow statistics (`fasstr`)
- Visualize results (P-pET vs. R, interactive flow stats)



Get Started

Set-up: packages and utils

```r
install.packages("devtools")

# hydro and met
devtools::install_github("bcgov/fasstr")
install.packages("daymetr")
install.packages("tidyhydat")

# spatial
devtools::install_github("mikejohnson51/AOI")
install.packages("rgeos")
install.packages("raster")
install.packages("leaflet")

# general purpose and viz
install.packages("dplyr")
install.packages("purrr")
install.packages("ggplot2")
install.packages("plotly")
install.packages("DT")
```

- Relevant packages loaded when necessary
- Typically use `package::function()` for clarity

A: Define area of interest

Define area

- use AOI to define bounding box around Glacier National Park (100 by 100 km)

```r
# Specify Regions and have a peak
rockies <- AOI::getAOI(clip = list("Glacier National Park Canada",
                                   100,
                                   100),
                       km = TRUE)

class(rockies)
```

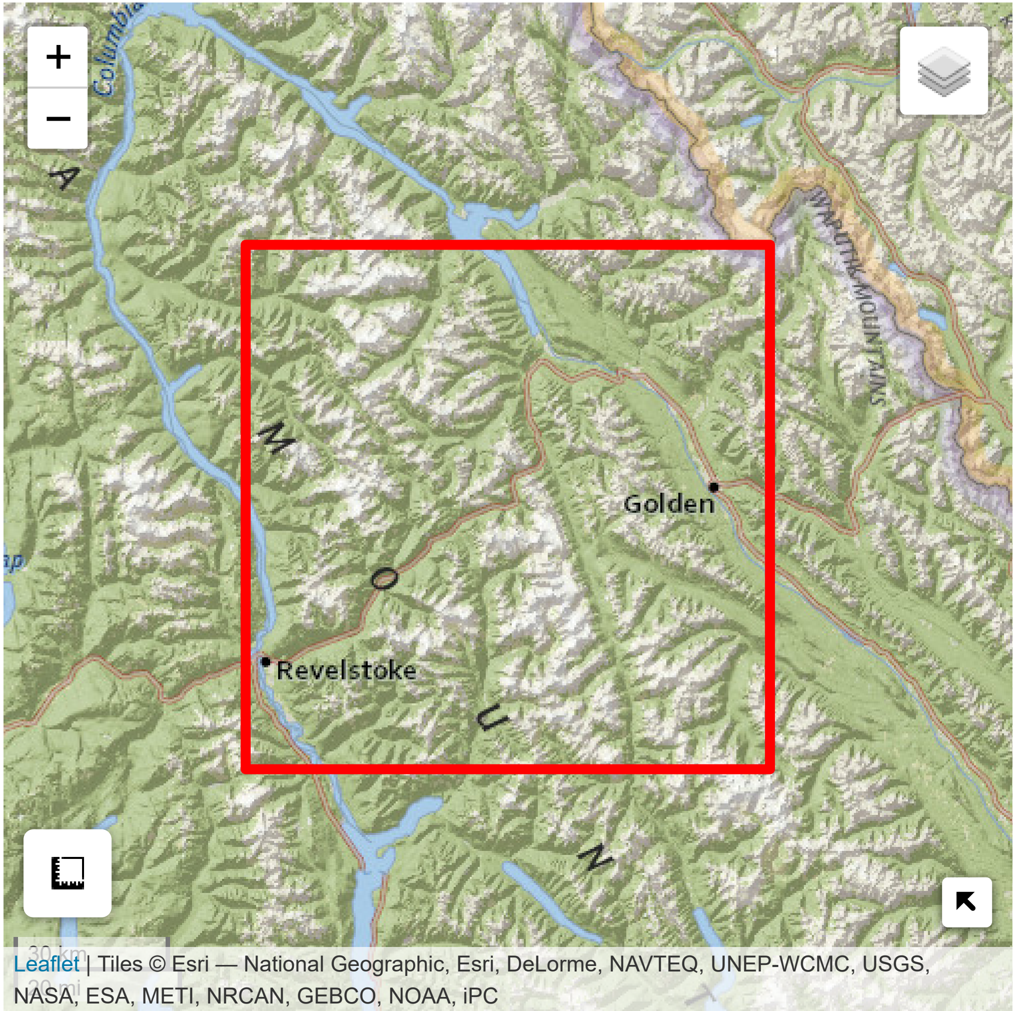
```
## [1] "SpatialPolygons"
## attr(,"package")
## [1] "sp"
```

- rockies@bbox useful for other functions!

Define area

```
library(dplyr)
library(leaflet)

rockies %>% AOI::check()
```

Define area



Leaflet | Tiles © Esri — National Geographic, Esri, DeLorme, NAVTEQ, UNEP-WCMC, USGS, NASA, ESA, METI, NRCAN, GEBCO, NOAA, iPC

B: Climate Data

Download climate data

- daymetr: gridded climate data for North America as NetCDF
- downloads files to defined path
- provided in LCC projection

```
# for rockies
path_rockies <- "./03_get-clean-viz_files/data/rockies"

# download data from daymet service
params <- c("dayl", "tmin", "tmax", "prcp")
params %>%
    purrr::walk(
        ~daymetr::download_daymet_ncss(location = c(rockies@bbox[[4]],
                                           rockies@bbox[[1]],
                                           rockies@bbox[[2]],
                                           rockies@bbox[[3]]), # top left to bottom righ
                              start = 2010,
                              end = 2011,
                              param = .x,
                              frequency = "daily",
                              path = path_rockies))
```

Load climate rasters

- Define projections

```
# Projected CS
proj4.Lambert <- "+proj=lcc +lat_1=25 +lat_2=60 +lat_0=42.5 +lon_0=-100 +x_0=0 +y_0=0 +a
# Geographic CS
proj4.WGS <- "+init=epsg:4326"
```

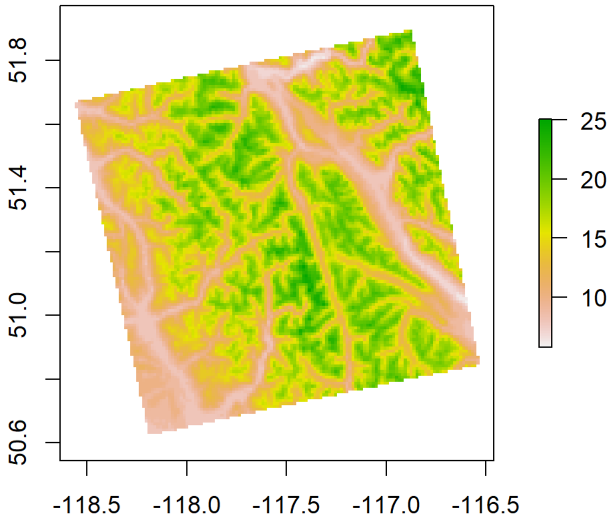
Load climate rasters

- read individual NetCDFs, stack as rasters
- reproject

```r
path_rockies <- "./03_get-clean-viz_files/data/rockies"

# load data, reproject
params <- c("dayl", "tmin", "tmax", "prcp")
rockies_stacks <- params %>%
    purrr::map( function(x){

        list.files(path_rockies,
                   pattern = x,
                   full.names = TRUE) %>%

            raster::stack() %>%
            raster::`projection<-`(., proj4.Lambert) %>%
            raster::projectRaster(crs = proj4.WGS)
        }
    ) %>%
    setNames(params)
```

Load climate rasters

```
# Jan 1, 2010 + 399 days
raster::plot(rockies_stacks$prcp[[400]])
```



Calculate P - pET

- based on `ET.Hamon()` from `Ecohydrology` 📦
- applied to basic raster math

```r
# custom function to calculate Hamon's PET
et.ham <- function(tmin,tmax,dayl){
    # modified from Evapotranspiration package
    Ta <- (tmax + tmin)/2
    vs_Tmax <- 0.6108 * exp(17.27 * tmax/(tmax + 237.3))
    vs_Tmin <- 0.6108 * exp(17.27 * tmin/(tmin + 237.3))
    vas <- (vs_Tmax + vs_Tmin)/2
    ET_Hamon.Daily <- 0.55 * 25.4 * (dayl/12)^2 * (216.7 *
                                                          vas * 10/(Ta + 273.3))/100
    return(ET_Hamon.Daily)
}


# calculate PET over all days (2 years total)
et_rockies <- et.ham(tmin = rockies_stacks$tmin,
                     tmax = rockies_stacks$tmax,
                     dayl = rockies_stacks$dayl / 3600)

# remove calc. artefacts (set to NA from previous raster)
et_rockies <- raster::mask(et_rockies, rockies_stacks$dayl[[1]])
```

Calculate P - pET

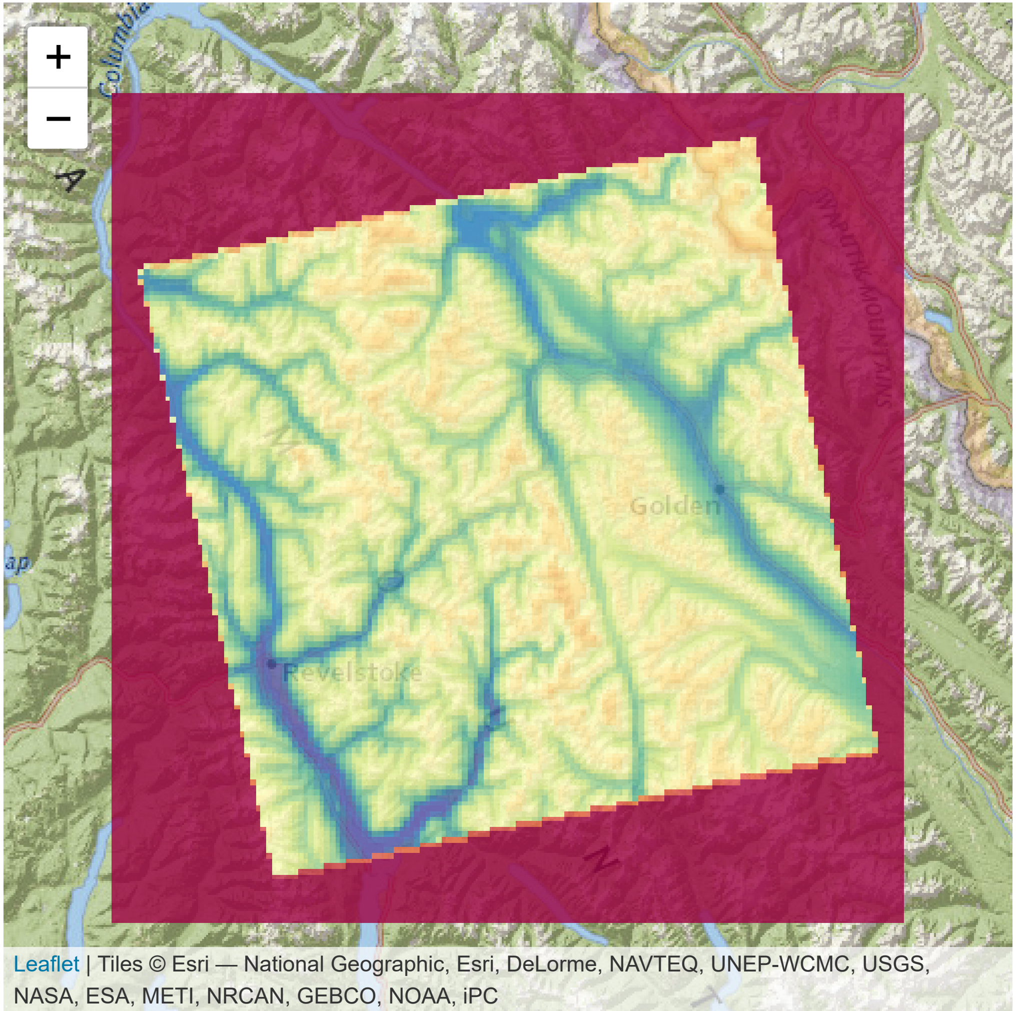
- aggregate to monthly values

```
# set up indices for aggregating over months
year_mon <- seq(as.Date("2010-01-01"),
                as.Date("2011-12-31"),
                by = "1 day") %>%
    format("%Y-%m")
months <- as.numeric(as.factor(year_mon))


# monthly totals of et
et_monthly <- raster::stackApply(et_rockies, months, fun = sum)

# monthly totals of p
p_monthly <- raster::stackApply(rockies_stacks$prcp, months, fun = sum)

# rough balance: p - pet
pet_monthly <- p_monthly - et_monthly

# mask NA values for clean plotting
pet_monthly <- raster::mask(pet_monthly, rockies_stacks$dayl[[1]])
```

Calculate P - pET

```
# interactive for June
leaflet() %>%
    addPolygons(data = rockies, fillColor = "transparent", color = col, weight = 4) %>%
    addProviderTiles("Esri.NatGeoWorldMap",
                     group = "Terrain") %>%
    addRasterImage(x = pet_monthly[[6]], opacity = .8)
```

Calculate P - pET



Leaflet | Tiles © Esri — National Geographic, Esri, DeLorme, NAVTEQ, UNEP-WCMC, USGS, NASA, ESA, METI, NRCAN, GEBCO, NOAA, iPC

B: Flow data

Identify viable stations

- use bounding box from AOI to filter for stations in tidyhydat data base (lat, lon).

```
# requires download
tidyhydat::hy_set_default_db(hydat_path = "D:/ext_data_R/Hydat.sqlite3")

rockies_stns <- tidyhydat::allstations %>%
    filter(between(LONGITUDE,
                   rockies@bbox[1,1],
                   rockies@bbox[1,2]),

           between(LATITUDE,
                   rockies@bbox[2,1],
                   rockies@bbox[2,2]),

           HYD_STATUS == "ACTIVE")
```

STATION_NUMBER	STATION_NAME	HYD_STATUS
08NA002	COLUMBIA RIVER AT NICHOLSON	ACTIVE
08NA006	KICKING HORSE RIVER AT GOLDEN	ACTIVE
08NB005	COLUMBIA RIVER AT DONALD	ACTIVE
08NB012	BLAEBERRY RIVER ABOVE WILLOWBANK CREEK	ACTIVE

Download hydrometric data

- fasstr::screen_flow_data() for missing data checks

```
rockies_q <- tidyhydat::hy_daily_flows(station_number =  rockies_stns$STATION_NUMBER,
                                       start_date = "2000-01-01",
                                       end_date = "2015-12-31")

check_me <- rockies_q %>%
    fasstr::screen_flow_data()
```

Download hydrometric data

- aggregate flow over months

```r
rockies_q_month <- rockies_q %>%
    filter(Date >= "2010-01-01",
           Date <= "2011-12-31") %>%

    # cleaning
    fasstr::fill_missing_dates() %>%
    # add info (table joining)
    fasstr::add_basin_area() %>%

    # calcs
    mutate(r_mm_day = Value / Basin_Area_sqkm * 86400 / 1e6 * 1e3) %>%

    # aggregate
    group_by(STATION_NUMBER,
             year_mon = format(Date, "%Y-%m")) %>%
    summarise(r_mm_month = sum(r_mm_day, na.rm = TRUE)) %>%

    mutate(date_time = lubridate::ymd(paste0(year_mon, "-01"),
                                      tz = "MST"))
```

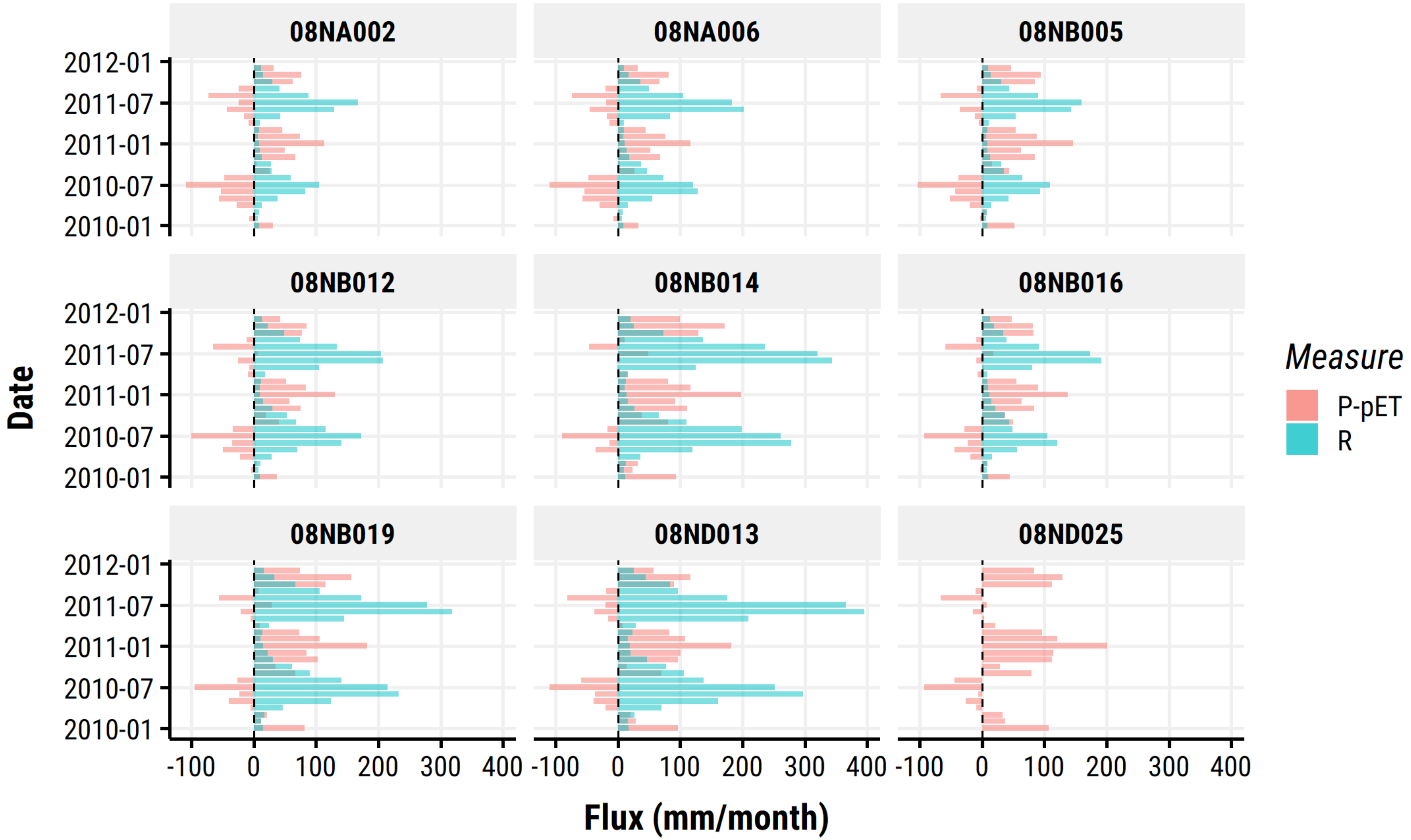
Extract P-pET for stations

```r
# extract data from raster at station locations
# Typically would use catchment shape files
pet_stns <- raster::extract(pet_monthly,
                y = rockies_stns[ ,c("LONGITUDE","LATITUDE")]) %>%

    as.data.frame() %>%

    bind_cols(rockies_stns[ ,"STATION_NUMBER"]) %>%

    setNames(c(year_mon %>% unique(), "stn")) %>%

    tidyr::gather(-stn, key = "year_mon", value = "p_pet_mm") %>%

    mutate(date_time = lubridate::ymd(paste0(year_mon, "-01"), tz = "MST"))
```

Runoff vs. P-pET

```r
library(ggplot2)
pet_r_plot <- pet_stns %>%
    filter(stn %in% rockies_q_month$STATION_NUMBER) %>%

    ggplot(aes(x = date_time,
               y = p_pet_mm)) +
    # add climate data
    geom_bar(stat = "identity",
             aes(fill = "P-pET"),
             alpha = 0.5) +
    # add R data
    geom_bar(inherit.aes = FALSE,
             data = rockies_q_month %>%
                 rename(stn = STATION_NUMBER),
             aes(x = date_time,
                 y = r_mm_month,
                 fill = "R"),
             stat = "identity",
             alpha = 0.5) +
    geom_hline(yintercept = 0, linetype = 2) +
    # misc
    theme_pub(base_size = 18) +
    labs(x = "Date",
         y = "Flux (mm/month)",
         fill = "Measure") +
    coord_flip() +
    facet_wrap(~stn)
```

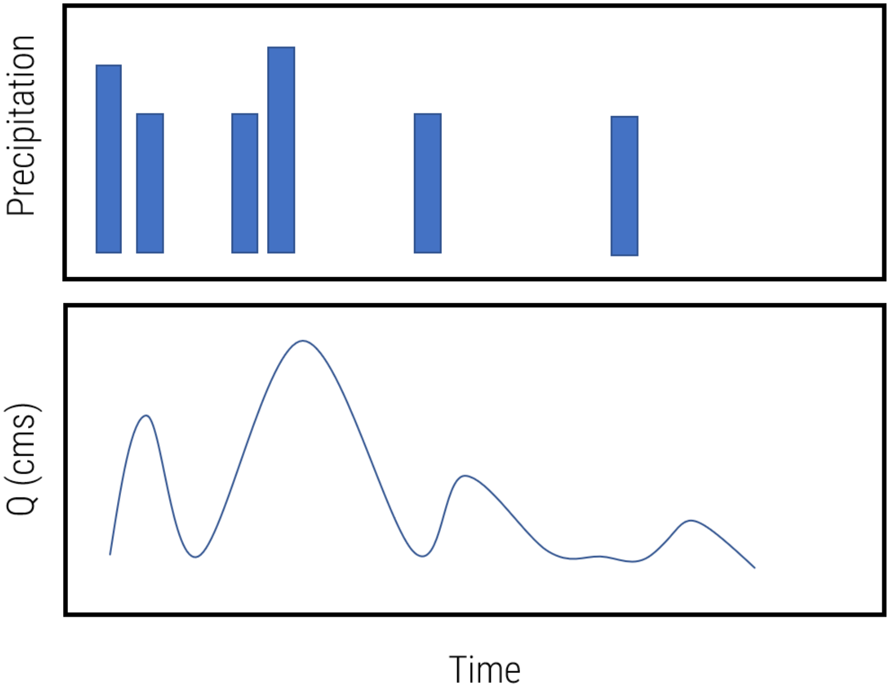
Runoff vs. P-pET



C: Flow stats in interactive visualization

Outcome:

Interactive visualization with two panels showing:



for Beaver River station

Prepare flow data

- `fasstr::calc_daily_stats()` for a range of flow statistics over chosen interval, e.g. (min, max, Q5, Q25, Q75, Q95, mean/median)

```
# choose a station
beaver_river <- rockies_stns %>%
    filter(STATION_NUMBER == "08NB019")

# calculate flow stats
beaver_q_stats <- fasstr::calc_daily_stats(rockies_q %>%
                                               filter(STATION_NUMBER == beaver_river$STA

# add 2011 flow data for interactive viz
beaver_q_stats <- beaver_q_stats %>%
    left_join(rockies_q %>%
                  filter(STATION_NUMBER == beaver_river$STATION_NUMBER,
                         Date >= "2010-01-01",
                         Date < "2011-01-01") %>%
                  mutate(DayofYear = lubridate::yday(Date)),
              by = "DayofYear")
```

Prepare flow data

Show 10 ▾ entries Search:

	STATION_NUMBER.x	Date.x	DayofYear	Mean
1	08NB019	Jan-01	1	8.50687506
2	08NB019	Jan-02	2	8.514375060
3	08NB019	Jan-03	3	8.566250056
4	08NB019	Jan-04	4	8.535625129
5	08NB019	Jan-05	5	8.601249903
6	08NB019	Jan-06	6	8.445625036
7	08NB019	Jan-07	7	8.601874977
8	08NB019	Jan-08	8	8.653749972
9	08NB019	Jan-09	9	8.718125045
10	08NB019	Jan-10	10	8.682500004

Showing 1 to 10 of 365 entries

Prepare precipitation data

- Equivalent to earlier raster extraction

```
# extract P data and make df
p_beaver <- raster::extract(rockies_stacks$prcp,
                                        y = beaver_river[ ,c("LONGITUDE","LATITUDE")]) %>
    as.data.frame() %>%

    bind_cols(beaver_river %>%
                  select(STATION_NUMBER,
                         STATION_NAME)) %>%

    setNames(c(seq(as.Date("2010-01-01"),
                 as.Date("2011-12-31"),
                 by = "1 day") %>%
                   as.character(),
             "stn",
             "name")) %>%

    tidyr::gather(-stn, -name, key = "date", value = "p_mm_day") %>%

    mutate(date_time = lubridate::ymd(date, tz = "MST"))
```

Set-up interactive plots

Flow stats

```r
library(plotly)
q_beaver <- plot_ly(data = beaver_q_stats,
                    x = ~DayofYear) %>%

    # Add max range
    add_ribbons(ymin = ~Minimum,
                ymax = ~Maximum,
                color = I("gray80"),
                name = "Max. Range (2000 - 2015)") %>%

    # add percentiles
    add_ribbons(ymin = ~P5,
                ymax = ~P95,
                color = I("steelblue1"),
                name = "Q5-Q95 (2000 - 2015)") %>%
    # add 20
    add_lines(y = ~Value,
              color = I("darkorange"),
              name = "Beaver River (2010)") %>%
    layout(yaxis = list(title = "mean daily Q (Cm/s)"))
```

Set-up interactive plots

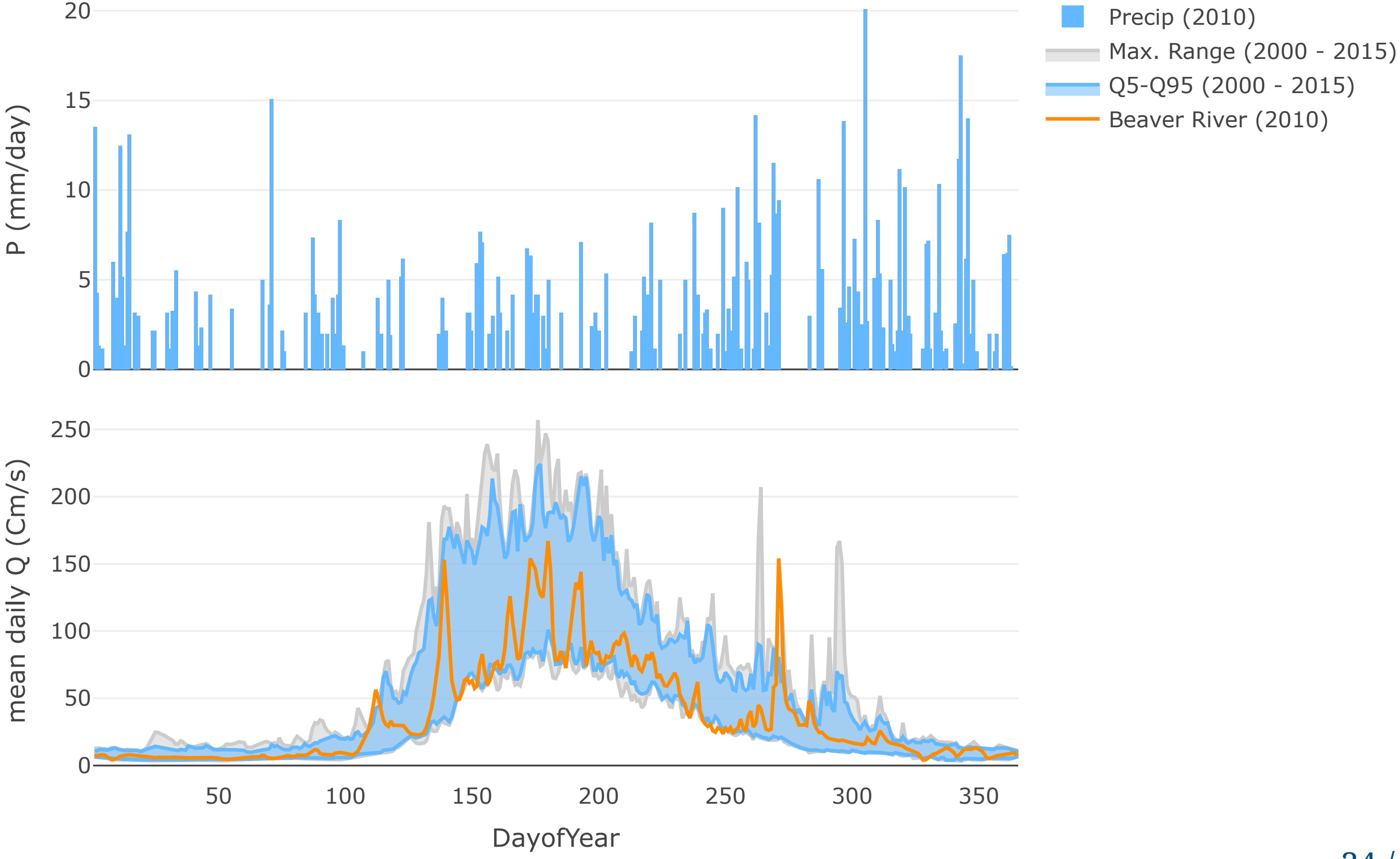
Precip

```
precip_beaver <- p_beaver %>%
    filter(date_time >= "2010-01-01",
           date_time < "2011-01-01") %>%
    mutate(DayofYear = lubridate::yday(date_time)) %>%

    # set up plot
    plot_ly(data = .,
            type = "bar",
            name = "Precip (2010)") %>%
    add_bars(x = ~DayofYear,
             y = ~p_mm_day,
             color = I("steelblue1")) %>%
    layout(yaxis = list(title = "P (mm/day)"),
           width = 800)
```

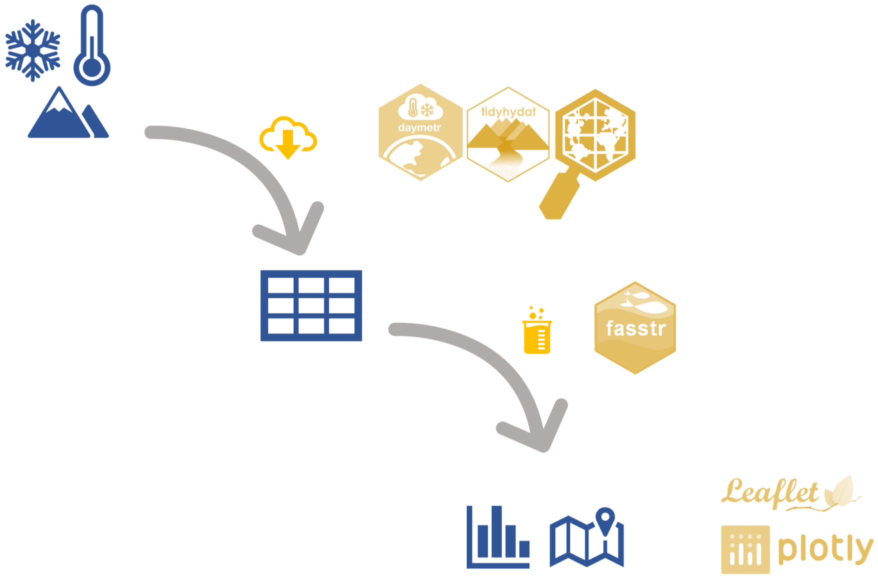
Generate final visualization

```
subplot(precip_beaver,
        q_beaver,
        nrows = 2,
        shareX = TRUE,
        titleY = TRUE)
```

Generate final visualization



Summary



Used range of general-purpose and hydrologic packages in concert to:

- obtain spatial and time series data (climate, hydrometric)
- wrangle, clean and pre-process data
- make a range of static and interactive visualization (maps, figures)