



Hydrologic Data for Use in Calculating Flow Condition Metrics for Waters in and near National Park Service Units.

Periods of Record Through the End of the 2018 Water Year

Natural Resource Data Series NPS/NRSS/IMD/NRDS—2019/1232



ON THE COVER

Gunnison River flowing through a canyon at Curecanti National Recreation Area.
Photo by Victoria Stauffenberg.

Hydrologic Data for Use in Calculating Flow Condition Metrics for Waters in and near National Park Service Units.

Periods of Record Through the End of the 2018 Water Year

Natural Resource Data Series NPS/NRSS/IMD/NRDS—2019/1232

Joseph C. DeVivo

National Park Service
1201 Oakridge Drive, Suite 150
Fort Collins, CO 80525

August 2019

U.S. Department of the Interior
National Park Service
Natural Resource Stewardship and Science
Fort Collins, Colorado

The National Park Service, Natural Resource Stewardship and Science office in Fort Collins, Colorado, publishes a range of reports that address natural resource topics. These reports are of interest and applicability to a broad audience in the National Park Service and others in natural resource management, including scientists, conservation and environmental constituencies, and the public.

The Natural Resource Data Series is intended for the timely release of basic data sets and data summaries. Care has been taken to assure accuracy of raw data values, but a thorough analysis and interpretation of the data has not been completed. Consequently, the initial analyses of data in this report are provisional and subject to change.

All manuscripts in the series receive the appropriate level of peer review to ensure that the information is scientifically credible, technically accurate, appropriately written for the intended audience, and designed and published in a professional manner.

Data in this report were collected and analyzed using methods based on established, peer-reviewed protocols and were analyzed and interpreted within the guidelines of the protocols.

Views, statements, findings, conclusions, recommendations, and data in this report do not necessarily reflect views and policies of the National Park Service, U.S. Department of the Interior. Mention of trade names or commercial products does not constitute endorsement or recommendation for use by the U.S. Government.

This report is available in digital format from the [Natural Resource Publications Management website](#). If you have difficulty accessing information in this publication, particularly if using assistive technology, please email irma@nps.gov.

Please cite this publication as:

DeVivo, J. C. 2019. Hydrologic data for use in calculating flow condition metrics for waters in and near National Park Service units. Periods of record through the end of the 2018 water year. Natural Resource Data Series NPS/NRSS/IMD/NRDS—2019/1232. National Park Service, Fort Collins, Colorado.

Contents

	Page
Figures.....	iv
Tables.....	iv
Abstract.....	v
Background and Summary.....	1
Methods.....	2
Acquisition of Mean Daily Flow Data for Stations of Interest.....	2
Preparation of Mean Daily Flow Data for Analysis.....	2
Identification of Stations with Zero Flow Maxima	3
Interpolation of Missing Data.....	3
Derivation of Periods of Record to be used for Hydrologic Metric Calculation.....	4
Data Records.....	6
Data Quality Evaluation.....	7
Mean Daily Flows	8
Periods of Record for Analysis.....	8
Usage Notes	10
References.....	11
Appendix A. R Code Listing	13
Appendix B. R Session and Version Information.....	35

Figures

	Page
Figure 1. General workflow for acquisition of mean daily flow data and derivation of periods of record to be used for calculation of hydrologic metrics as a part of the IMD Environmental Setting monitoring protocol.	2
Figure 2. Mean daily streamflow for Androscoggin River near Gorham, NH.	4

Tables

	Page
Table 1. Stations with zero flows that were removed from analysis.....	3
Table 2. Base Periods of record for hydrological station 0105400 (Androscoggin River near Gorham, NH).	4
Table 3. Analysis Periods of record for hydrological station 0105400 (Androscoggin River near Gorham, NH).....	5
Table 4. Overview of data sets and their provenance.....	6
Table 5. Definition of qualification codes used to describe mean daily flow measurements and periods of record.....	7
Table 6. Number of mean daily flow records per qualification code within the <i>Mean Daily Flow Data For Use in Calculating Hydrologic Metrics</i> data set.....	8
Table 7. Number of records per qualification code for Periods of Record (PORs).	9

Abstract

This data report (and associated data sets) documents the acquisition and preparation of stream daily flow data for use in calculating a series of hydrologic metrics as a part of the NPS Park Environmental Setting monitoring protocol (DeVivo et al. 2018). Acquisition and preparation of data is a three-step process. First, mean daily flow data from stations of interest are acquired from the USGS National Water Information System (NWIS) at <https://waterdata.usgs.gov/nwis>. Second, the data are evaluated and processed to ensure their suitability for use in calculating hydrologic metrics by a) eliminating stations with maximum flows of 0, and b) interpolating data gaps of two days or less. Last, the period of record for each station is inspected to identify uninterrupted, whole-water-year sub-periods of two years or more. The resultant flow and period-of-record data will later be used to calculate roughly 200 hydrologic metrics describing flow conditions in and around national parks. Mean daily flow data were acquired from NWIS on 2019-07-10.

Background and Summary

The status and condition of park resources is affected by interconnected system drivers at regional- to global scales surrounding parks. Climate and surface water dynamics are inherently linked with landscape dynamics, and the suite of environmental setting metrics provide an important context (and potential covariates) for understanding patterns and changes in other ecological indicators monitored at parks. Within monitoring plans developed by NPS Inventory & Monitoring networks, linkages between landscape dynamics, climate, and hydrology measures were the most-cited drivers of the status, condition, and trends in water resources (water quality, quantity, and geomorphology), changing species distributions (aquatic and terrestrial, with a focus on exotics), and changes in landscape pattern (habitat fragmentation) and type (land cover, use, and conversion) (DeVivo et al 2018).

The National Park Service (NPS) Park Environmental Settings (PES) monitoring protocol produces nationally-standardized status and trend information related to Landscape Dynamics, Climate, and Surface Water Dynamics measures (DeVivo et al. 2018). This data report documents the acquisition and preparation of stream daily flow data for use in calculating a series of hydrologic metrics as a part of the national Park Environmental Setting monitoring protocol (DeVivo et al. 2018). Although other environmental factors and characteristics of a stream influence the overall condition of a stream and its biological communities, a stream's flow is thought of as the master variable (Power et al. 1995). The hydrologic output of a watershed is a function of land characteristics, human use, weather and climate conditions, urbanization, and soil characteristics. As a result, stream flow characteristics are some of the most appropriate and useful indicators for assessing aquatic ecosystem integrity, and for monitoring environmental changes over time. They also provide key support data for interpreting other indicators including water quality, threatened and endangered aquatic species, wetlands, and riparian habitat.

Methods

Procedures for acquiring and preparing the data for additional analyses are outlined in DeVivo 2018 (See Figure 1).

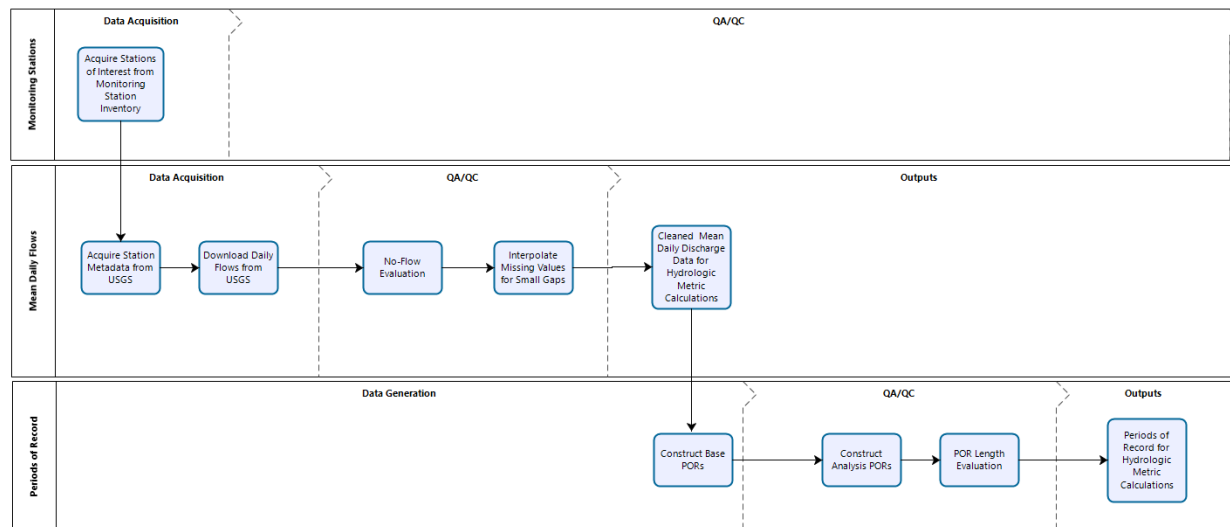


Figure 1. General workflow for acquisition of mean daily flow data and derivation of periods of record to be used for calculation of hydrologic metrics as a part of the IMD Environmental Setting monitoring protocol (DeVivo et al. 2018).

Acquisition of Mean Daily Flow Data for Stations of Interest

A total of 12,013 stations within HUC-12 watersheds that intersect with park boundaries (NPS 2019a) were evaluated for whether sufficient mean daily flow data exist for use in calculating hydrologic metrics at the conclusion of the 2018 water year. Station metadata were queried from the USGS National Water Information System <http://waterdata.usgs.gov/nwis> using the dataRetrieval R package (Hirsch and De Ciccio 2015) to identify stations with historic mean daily flow data (parameter code “00060” and statistical code “00003”). Mean daily flow data were available for 1,540 stations.

A total of 14,544,864 mean daily flow values (across all stations) were acquired on 2019-07-10. Data qualification codes from USGS were retained with the source data to inform use in subsequent analyses. Downloaded data have been retained for re-use as appropriate (NPS 2019b).

Preparation of Mean Daily Flow Data for Analysis

Preparation of data for metric calculation includes two steps. First, all stations where the maximum mean daily flow is 0 cfs were identified and removed from the data set. Second, because metric calculation requires continuous time series data, the period of record is evaluated for small data gaps (two days or less) and missing values are interpolated.

Identification of Stations with Zero Flow Maxima

Periods of record were evaluated to remove all stations with a maximum daily flow of 0 cfs. Upon inspection, seven stations were identified as having maximum daily flows of 0 cfs and removed from the data set (Table 1). As of the end of the 2018 water year, 1,534 stations were reported to have a maximum mean daily flow data greater than 0 cfs.

Table 1. Stations with zero flows that were removed from analysis.

Station Number	Station Name
06394600	HELL CANYON NEAR JEWELL CAVE NEAR CUSTER SD
06394605	HELL CANYON NEAR CUSTER SD
08313204	MORTANDAD CANYON NR LOS ALAMOS, NM
08313253	CANON DE VALLE ABV NM HWY 501 NR LOS ALAMOS, NM
08313265	WATER CANYON BLW NM HWY 4 NR WHITE ROCK, NM
1025125372	SPLIT WASH AT ANTLER RIDGE, NTS, NV
10252330	WHEATON WASH NR MOUNTAIN PASS CA

Interpolation of Missing Data

Periods of record were examined to determine whether sufficient data existed preceding and following data gaps to interpolate missing values. Time series were evaluated to insert missing dates, and any data gaps of two days or less were interpolated using the “fillMissing” function within the smwrBase package (Lorenz, 2015). As described in the package documentation:

The method used to interpolate missing values is based on tsSmooth constructed using StructTS on x with type set to trend. The smoothing method basically uses the information (slope) from two values previous to missing values and the two values following missing values to smoothly interpolate values accounting for any change in slope. Beauchamp (1989) used time-series methods for synthesizing missing streamflow records. The group that is used to define the statistics that control the interpolation is very simply defined by span rather than the more in-depth measures described in Elshorbagy and others (2000).

The “span” parameter was set to 3 resulting in the use of simple linear interpolation to replace missing values; the “max.fill” parameter was set to 2.

Across all stations, 7,429 mean daily flow values were missing from the source USGS data, and a total of 2,487,226 data points were missing across all stations’ periods of record. A total of 1,723 values were interpolated. All remaining records with flow values of “NA” were removed from the dataset. The resultant dataset contains 14,526,327 mean daily flow values from stations in or near NPS units.

Derivation of Periods of Record to be used for Hydrologic Metric Calculation

For each station of interest, the period of record for flow data was analyzed to identify data suitable for analysis. In general, periods of record must span at least two water years and not contain any missing data. For any given station, particularly those with long periods of record, there may be many “sub” periods of record where there are continuous data spanning two or more water years. For all continuous records, a series of PORs for analysis were identified beginning with the first two years, and then annually-augmented periods as appropriate for the period of record.

For example, for site number 0105400 (Androscoggin River near Gorham, NH, the station period of record is from 10/1/1913 to 11/13/2018. The period of record contains two discontinuities (Figure 2):



Figure 2. Mean daily streamflow for Androscoggin River near Gorham, NH.

The period of record contains three “Base PORs”, or periods of continuous data without data gaps (Table 2). Base PORs may contain values that are estimated by the data provider, or are interpolated during QC processes described above. Base PORs are trimmed to the start/end of water years (October 1 to September 30). Station PORs may contain several or no Base PORs (as would be the case where the station POR does not span two entire water years).

Table 2. Base Periods of record for hydrological station 0105400 (Androscoggin River near Gorham, NH).

Start Date	End Date
1913-10-01	1917-09-30
1918-10-01	1921-09-30
1928-10-01	2017-09-30

Each Base POR is then used to generate several Analysis PORs, which will be used for analysis of hydrologic metrics. Analysis PORs start at the beginning of their respective Base POR and contain continuous data spanning two or more whole water years (Table 3). Each Base POR may have multiple Analysis PORs of increasing length depending on the number of water years contained in the Base POR.

Table 3. Analysis Periods of record for hydrological station 0105400 (Androscoggin River near Gorham, NH). Analysis PORs are only shown for the 1913-1917 Base POR.

Start Date	End Date
1913-10-01	1914-09-30
1913-10-01	1915-09-30
1913-10-01	1916-09-30
1913-10-01	1917-09-30

Across all stations with mean daily flow data, 1,501 Base PORs were identified at 1,180 stations. Stations have as many as five Base PORs, however 353 stations do not have two or more continuous water years of data within their period of record and therefore calculation of hydrologic metrics is not possible at those locations. A total of 33,320 Analysis PORs were identified for use in calculation of hydrologic metrics.

Data Records

Three data sets were acquired or generated as a part of this effort (Table 4):

1. **Mean Daily Flow Data from USGS.** This data set contains the raw mean daily flow data and associated data qualification codes for stations of interest. These data are preserved as a record to support reproducible generation of the following data sets and/or if needed to support other uses (NPS 2019b). Available at <https://irma.nps.gov/DataStore/Reference/Profile/2263556>.
2. **Mean Daily Flow Data For Use in Calculating Hydrologic Metrics.** This data set contains mean daily flow data that have been evaluated and processed to ensure suitability for use in calculating hydrologic metrics. This data set includes interpolated values to fill (some) data gaps and some lumping of USGS qualification codes to inform subsequent use in analysis and reporting (NPS 2019c). Available at <https://irma.nps.gov/DataStore/Reference/Profile/2263592>.
3. **Periods of Record for Analysis.** This data set contains all periods of record to be used when calculating hydrologic metrics at stations of interest. Periods of record are qualified based on length of data availability and quality of underlying mean daily flow data (NPS 2019d). Available at <https://irma.nps.gov/DataStore/Reference/Profile/2263593>.

Table 4. Overview of data sets and their provenance.

Source Data	Resultant Data	Number of Records / Measure- ments	Temporal Range
USGS National Water Information System (http://www.waterdata.usgs.gov/nwis)	NPS_IMD_ESProtocol_Hydrology_2018_ All_Daily_Flows_2263556-data.csv	14,544,864	1874-04-01 to 2019-07-10
NPS_IMD_ESProtocol_Hydrology_2018_ All_Daily_Flows_2263556-data.csv	NPS_IMD_ESProtocol_Hydrology_2018_ Clean_Daily_Flows_2263592-data.csv	14,526,327	1874-04-01 to 2019-07-10
NPS_IMD_ESProtocol_Hydrology_2018_ Clean_Daily_Flows_2263592-data.csv	NPS_IMD_ESProtocol_Hydrology_2018_ Analysis_PORs_2263593-data.csv	33,320	1874-10-01 to 2018-09-30

Data Quality Evaluation

Daily flow data for use in calculating hydrologic metrics (NPS 2019c) were qualified based on qualification codes provided by USGS and results of additional processing described above (Table 5). Periods of record for analysis were qualified based on the length of the period of record and qualification codes associated with the mean daily flow measurements.

Table 5. Definition of qualification codes used to describe mean daily flow measurements and periods of record.

Qualification Code	Definition	Applies To
MPN	Measurement is not provisional	Mean Daily Flow Measurements
MPY	USGS Qualification Code (qualcode) contains "P" denoting that the measurement is provisional	Mean Daily Flow Measurements
MEN	Measurement is not estimated	Mean Daily Flow Measurements
MEY	USGS Qualification Code (qualcode) contains "e" denoting that the measurement is estimated	Mean Daily Flow Measurements
MfMN	Measurement was not interpolated by NPS	Mean Daily Flow Measurements
MfMY	Measurement was interpolated by NPS	Mean Daily Flow Measurements
MCN	USGS qualification code (qualcode) does contains "<" or ">" denoting that data are not censored	Mean Daily Flow Measurements
MCY	USGS qualification code (qualcode) contains "<" or ">" denoting that it is known to be greater or less than the reported value and is therefore considered censored	Mean Daily Flow Measurements
PLL	The period of record is considered long enough to support trend analysis in most situations (20 years or longer). Note that the amount of data necessary for analysis will vary by station, measure, and may be longer in some instances.	Periods of Record
PLS	The period of record is considered short (less than five years) and does not meet the recommended minimum of five years to conduct monotonic trend analysis on monthly variables	Periods of Record
PLI	The period of record is between 5 and 20 years and data may not be suitable for some analyses.	Periods of Record
PPN	The period of record does not contain provisional data	Periods of Record
PPY	The period of record contains provisional data	Periods of Record

Table 5 (continued). Definition of qualification codes used to describe mean daily flow measurements and periods of record.

Qualification Code	Definition	Applies To
PIN	The period of record does not contain interpolated data	Periods of Record
PIY	The period of record contains interpolated data	Periods of Record
PEN	The period of record does not contain estimated data	Periods of Record
PEY	The period of record contains estimated data	Periods of Record
PCN	The period of record does not contain censored data	Periods of Record
PCY	The period of record contains censored data	Periods of Record

Mean Daily Flows

Most mean daily flow data acquired from USGS have been accepted (99.2%), with 0.8% of data still provisional (Table 6). The data set contains 431,488 estimated values (these may be accepted or provisional), and an additional 1,723 values were interpolated by NPS as a part of the processes described above. Less than 1% of the data set are known to be censored (accepted by USGS for publication, but qualified as different than the reported value; 6,482 values total).

Table 6. Number of mean daily flow records per qualification code within the *Mean Daily Flow Data For Use in Calculating Hydrologic Metrics* data set (NPS 2019c).

Category	Total	No Qualifier Code	Estimated (by USGS)	Interpolated (by NPS)	Censored
Accepted (by USGS)	14,406,603 (99.2%)	13,974,550 (96.2%)	425,571 (2.9%)	0 (0.0%)	6,482 (0.0%)
Provisional (by USGS)	118,001 (0.8%)	112,084 (0.8%)	5,917 (0.0%)	0 (0.0%)	0 (0.0%)
Other	1,723 (0.0%)	0 (0.0%)	0 (0.0%)	1,723 (0.0%)	0 (0.0%)
Total	14,526,327	14,086,634	431,488	1,723	6,482

Periods of Record for Analysis

Analysis PORs were qualified based on the number of continuous water years and characteristics of the underlying mean daily flow data. Additionally, periods of record were qualified based on whether

they contained provisional data, estimated or interpolated data, or censored data (data reported by USGS as known to be above or below the recorded value). The frequency of qualification codes for analysis periods of record are presented in Table 7. Overall, 722 of the 1,509 stations have periods of record >20 years in length, with 502 of those having 20+ years of uninterrupted data. Provisional data are included in less than one percent of all Analysis PORs.

Table 7. Number of records per qualification code for Periods of Record (PORs).

Type	Available Data ¹	Base PORs ²		Analysis PORs ²	
	Stations	Stations	PORs	Stations	PORs
PORs <= 5 years	383	289	322	1,180	4,126
PORs >5 and <= 20 Years	437	563	654	978	12,733
PORs >20 Years	721	502	525	502	16,461
PORs with Provisional Data	–	76	76	76	107
PORs with Estimated Data	–	715	814	715	12,161
PORs with Interpolated Data	–	69	76	69	452
PORs with Censored Data	–	18	18	18	201
Total	1,541	1,180	1,501	1,180	33,320

¹ From the *Mean Daily Flow Data from USGS* data set (NPS 2019b).

² From the *Periods of Record for Analysis* data set (NPS 2019d).

Usage Notes

These data sets are intended and suitable for use by the NPS Inventory and Monitoring Division when calculating hydrologic metrics for specific gaging stations as described in the national Environmental Setting monitoring protocol (DeVivo et al., 2018). The U.S. Geological Survey maintains the authoritative data, which is continually updated and amended over time. Individuals looking to conduct work outside the scope of the NPS Environmental Setting monitoring protocol are strongly encouraged to acquire the most-recent data from the USGS National Water Information System (NWIS) at <http://waterdata.usgs.gov/nwis>.

References

- DeVivo, J. C. 2018. Adding new stations and periods of record for use in calculation of hydrologic measures. Inventory & Monitoring Program Standard Operating Procedure NPS/IMD/SOP—3.3.3. National Park Service, Fort Collins, Colorado.
- DeVivo, J. C., L. Nelson, M. Kinseth, T. Philippi, and W. B. Monahan. 2018. Protocol for monitoring environmental setting of National Park Service units: landscape dynamics, climate, and hydrology. Natural Resource Report NPS/IMD/NRR—2018/1844. National Park Service, Fort Collins, Colorado.
- Hirsch, R. M., and De Cicco, L. A. 2015. User guide to Exploration and Graphics for RivEr Trends (EGRET) and dataRetrieval—R packages for hydrologic data (version 2.0, February 2015): U.S. Geological Survey Techniques and Methods book 4, chap. A10, 93 p., <http://dx.doi.org/10.3133/tm4A10>.
- Lettenmaier, D. P., L. L. Conquest and J. P. Hughes. 1982. Routine Streams and Rivers Water Quality Trend Monitoring Review. C.W. Harris Hydraulics Laboratory, Technical Report No. 75, Seattle, WA.
- Lorenz, D.L. 2015. smwrBase—An R package for managing hydrologic data, version 1.1.1. U.S. Geological Survey OpenFile Report 2015–1202, 7 p. (Also available at <https://pubs.usgs.gov/of/2015/1202/ofr20151202.pdf>).
- National Park Service (NPS). 2019a. Hydrology Stations for NPS Inventory and Monitoring Inventory 2.0 Monitoring Locations. *National Park Service Data Store* <https://irma.nps.gov/DataStore/Reference/Profile/2264917>.
- National Park Service (NPS). 2019b. Mean daily flows for stations within hydrologic unit codes intersecting with park boundaries. Complete periods of record from the USGS National Water Information System through the end of the 2018 water year. *National Park Service Data Store* <https://irma.nps.gov/DataStore/Reference/Profile/2263556>.
- National Park Service (NPS). 2019c. Mean daily flows for use in calculating hydrologic metrics as a part of the NPS Park Environmental Settings Monitoring Protocol based on data available through the end of 2018 water year. *National Park Service Data Store* <https://irma.nps.gov/DataStore/Reference/Profile/2263592>.
- National Park Service (NPS). 2019d. Periods of record for use in calculating hydrologic metrics as a part of the NPS Park Environmental Settings Monitoring Protocol based on data available through the end of 2018 water year. *National Park Service Data Store* <https://irma.nps.gov/DataStore/Reference/Profile/2263593>.
- Power, M. E., A. Sun, M. Parker, W. E. Dietrich, and J. T. Wootton. 1995. Hydraulic food-chain models: an approach to the study of food web dynamics in large rivers. *BioScience* 45:159-167.

Appendix A. R Code Listing

```
# This setup code loads both reproducible reporting packages
# (delete those not needed) and packages for the actual project.
# Note that it also generates the start of a BibTeX literature cited
# including the citations for R and all used packages

# reproducible reporting packages
RRpackages <- c('markdown',      # links to Sundown rendering library
               'rmarkdown',      # newer rendering via pandoc
               'pander',         # alternative renderer for markdown,
                                # plus better tables than just knitr
               'knitr',
               'bookdown',       # for larger documents
                                # https://bookdown.org/
                                # https://bookdown.org/yihui/bookdown/
               'bookdownplus',   # more templates for bookdown plus simplifying
                                # scripts
                                # https://bookdown.org/baydap/bookdownplus/
               "dataMaid",       # for makeCodebooks
               "R.rsp",          # dynamic generation of scientific reports
               "kimisc",         #
               "papeR",          # stat tables
               "texreg",         # formatting regression results for LaTeX
                                # or html
               "rmdHelpers",     # misc from Mark Peterson
                                # thisFileName() thisFile_knit()
               'yaml',          # format data into markdown
               'rmdformats',     # templates including automatic ToC,
                                # also use_bookdown()
               'htmltools',      #
#               'kfigr',          # anchors for figures (not needed with bookdown)
               "bibtex",
               "RefManageR",     # BibTeX reference manager
               "knitcitations" #
               )

inst <- RRpackages %in% installed.packages()
if (length(RRpackages[!inst]) > 0) {
  install.packages(RRpackages[!inst], dep = TRUE)
}
lapply(RRpackages, library, character.only = TRUE)

#
# Now repeat for packages used in the analyses
pkgList <- c("devtools",
            "smwrBase",
            "RODBC",
            "dataRetrieval",
            "waterData",
            "EflowStats",
            "flextable",
            "tibble",
```

```

        'lubridate',
        'english',
        'reshape2',
        'svMisc',
        "zoo",
        "EML",
        "dplyr")
inst <- pkgList %in% installed.packages()
if (length(pkgList[!inst]) > 0) {
  install.packages(pkgList[!inst], dep = TRUE,
    repos = "https://cloud.r-project.org")
}

lapply(pkgList, library, character.only = TRUE, quietly = TRUE)

# create stub of citations for packages
pkgBibTex <- lapply(c("base", pkgList, R packages), citation)

# pkgBibTex <- do.call()

knitr::opts_chunk$set(
  root.dir = params$projectDir, # from YAML parameter, knitr instead of setwd()
  echo = TRUE,
  comment = " ",
  # dev = "svg",
  fig.path = "figures/",
  tidy.opts = list(width.cutoff = 60),
  tidy = TRUE
)
# if ggplot, update theme to default to centered titles
if ("ggplot2" %in% .packages()) {
  theme_update(plot.title = element_text(hjust = 0.5))
}

#functions
my.max <- function(x) ifelse( !all(is.na(x)), max(x, na.rm=T), NA)

# The following reads data files generated by the RMD file associated with SOP 3.3.3.
# These are not needed to generate the data report, but are provided to expedite
processing during development.
# Need to update this so that it's based on the final station IDs after being
rewritten to the database

load(file="data/AllDailyFlowData.Rdata")
load(file="data/InterpolatedDailyFlowData.Rdata")
load(file="data/AnalysisPORs.Rdata")
load(file="data/mstr_stationQuals.Rdata")
load(file="data/DailyFlowsMetadata.Rdata")
load(file="data/Output_MeanDailyFlows.Rdata")
load(file="data/Output_AnalysisPORs.Rdata")
load(file="data/QualificationCodes.RData")
load(file="data/tlu_PORs.RData")
load(file="data/BasePORs.RData")

```

```

load(file="data/AllDailyFlowDataWithoutZeroFlowStations.RData")
load(file= "data/AllDailyFlowDataNoGaps.RData")
load(file= "data/stationinventory.RData")
ReportNumbers<-read.csv(file="metadata/ReportNumbers.csv", stringsAsFactors = FALSE)
ProjectMetadata<-read.csv(file="metadata/ProjectMetadata.csv", stringsAsFactors =
FALSE)

# Create Reference URLs
DataPublicationReportRefID<-subset(ReportNumbers$ReferenceID,
ReportNumbers$Parameter=="DataPublicationReportRefID")
FlowRefID<-subset(ReportNumbers$ReferenceID, ReportNumbers$Parameter=="FlowRefID")
CleanFlowRefID<-subset(ReportNumbers$ReferenceID,
ReportNumbers$Parameter=="CleanFlowRefID")
AnalysisPORsRefID<-subset(ReportNumbers$ReferenceID,
ReportNumbers$Parameter=="AnalysisPORsRefID")

DataPublicationReportURL<-
paste0("https://irma.nps.gov/DataStore/Reference/Profile/",DataPublicationReportRefID)
FlowURL<-paste0("https://irma.nps.gov/DataStore/Reference/Profile/",FlowRefID)
CleanFlowURL<-
paste0("https://irma.nps.gov/DataStore/Reference/Profile/",CleanFlowRefID)
AnalysisPORsURL<-
paste0("https://irma.nps.gov/DataStore/Reference/Profile/",AnalysisPORsRefID)

ch <- odbcConnect("Hydro")
QualificationCodes<- as.data.frame(sqlFetch(ch,"dbo.vw_QualificationCodes"))
tlu_PORs<- as.data.frame(sqlFetch(ch,"dbo.tlu_PORs"))
odbcClose(ch)

#write data frame to file so that you don't have to do this again!
save(QualificationCodes,file="data/QualificationCodes.RData")
save(tlu_PORs,file="data/tlu_PORs.RData")

# This will be updated to pull from published data set once the water inventory data
set is published.

ch2 <- odbcConnect("Monitoring Stations Inventory")
stationinventory<-
as.data.frame(sqlFetch(ch2,"dbo.USGS_NWIS_HydroQA_MonitoringLocations_AOA_HUC12_201907
02"))
odbcClose(ch2)

save(stationinventory,file="data/stationinventory.RData")

stationinventory2<-subset(stationinventory,MonitoringLocationTypeName=="Stream")
stationinventory2$staid<-as.character((gsub(".*-
","",stationinventory2$MonitoringLocationIdentifier)))

# get station metadata for all stations in the master database and determine which

```

ones have daily flow data.

```
sitesWithDailyDischargeData <- whatNWISdata(siteNumber = stationinventory2$staid,  
parameterCd="00060", statCd ="00003")  
sitesWithDailyDischargeData<-unique(subset(sitesWithDailyDischargeData, data_type_cd  
== 'dv', select=-c(loc_web_ds)))  
sitesWithDailyDischargeData<-unique(subset(sitesWithDailyDischargeData,  
select=c("site_no", "station_nm", "begin_date", "end_date")))  
colnames(sitesWithDailyDischargeData)[1]<-"staid"
```

#extract prior water year from station POR data.

```
MaxWaterYear<-ifelse(  
  
  leap_year(max(as_date(sitesWithDailyDischargeData$end_date)))== TRUE,  
  
  ifelse(  
    yday(max(as_date(sitesWithDailyDischargeData$end_date))) < 275,  
    paste(year(max(as_date(sitesWithDailyDischargeData$end_date)))-1),  
    paste(year(max(as_date(sitesWithDailyDischargeData$end_date))))  
  ),  
  
  ifelse(  
    yday(max(as_date(sitesWithDailyDischargeData$end_date))) < 274,  
    paste(year(max(as_date(sitesWithDailyDischargeData$end_date)))-1),  
    paste(year(max(as_date(sitesWithDailyDischargeData$end_date))))  
  )  
)  
# get daily flow data for all stations with daily flow data (this step takes awhile).  
  
# Run this code separately before knitting the data document; it will write the  
"AllDailyFlowData.Rdata" file that will be used when generating the data report.  
  
# Initialize empty AllDailyFlowData table  
AllDailyFlowData<- setNames(data.frame(matrix(ncol = 5, nrow = 0)), c("agency_cd",  
"site_no", "Date", "X_00060_00003", "X_00060_00003_cd"))  
  
# Fetch daily flow data for all sites with daily discharge data  
loopcount<-nrow(sitesWithDailyDischargeData)  
for (i in 1:loopcount) {  
  progress(i/loopcount*100)  
  dailyQ <- readNWISdv(siteNumber = sitesWithDailyDischargeData[i,1], parameterCd =  
"00060", startDate = sitesWithDailyDischargeData[i,3], endDate =  
sitesWithDailyDischargeData[i,4])  
  dailyQ <-subset(dailyQ,select=1:5)  
  colnames(dailyQ)<-c("agency_cd", "site_no", "Date", "X_00060_00003",  
"X_00060_00003_cd")  
  AllDailyFlowData<-rbind(AllDailyFlowData,dailyQ)  
  if (i == loopcount) cat("Done!\n")  
}  
#write systime to metadata table
```

```

DailyFlowsMetadata<-as.data.frame(cbind(DataAcquisitionDateTime=as_date(now())))

#write data frame to file so that you don't have to do this again!
save(AllDailyFlowData,file="data/AllDailyFlowData.RData")
save(DailyFlowsMetadata,file="data/DailyFlowsMetadata.RData")

# This identifies all sites with maximum flows of 0 across the site's entire period of
record.
# (this causes an error when doing the interpolation and the sites aren't useful for
metric calculation anyway)
# sites identified in this step will be used to qualify stations PORs later on.

AllDailyFlowData2<-AllDailyFlowData[c(2,4,3,5)]
colnames(AllDailyFlowData2)<-c("staid", "val", "dates", "qualcode")

zerotest<-subset(AllDailyFlowData2 %>% group_by(staid) %>% summarise(Value =
max(my.max(val))), Value==0)
zerotestTable<-subset(left_join(zerotest,sitesWithDailyDischargeData, by="staid"),
select=c(staid,station_nm))
names(zerotestTable)<-c("Station Number", "Station Name")

AllDailyFlowDataWithoutZeroFlowStations<-
anti_join(AllDailyFlowData2,zerotest,by="staid")
AllDailyFlowDataWithoutZeroFlowStations$dates<-
as_date(AllDailyFlowDataWithoutZeroFlowStations$dates)
AllDailyFlowDataWithoutZeroFlowStations$val<-
as.numeric(AllDailyFlowDataWithoutZeroFlowStations$val)

T1<-theme_vanilla(regulartable(zerotestTable))
T1<-autofit(T1)
T1<-align(T1, align = "left", part="all")
T1

save(AllDailyFlowDataWithoutZeroFlowStations,file=
"data/AllDailyFlowDataWithoutZeroFlowStations.RData")

##### Step 1. Insert missing dates with flow value = NA. This will add additional
data gaps where data can be interpolated.

rm(AllDailyFlowDataNoGaps)
AllDailyFlowDataNoGaps<- setNames(data.frame(matrix(ncol = 4, nrow = 0)), c("staid",
"val", "dates", "qualcode"))

stations<-unique(subset(AllDailyFlowDataWithoutZeroFlowStations,select="staid"))
stationcount<-nrow(stations)

for (i in 1:stationcount) {
  progress(i/stationcount*100)
  temp<-
subset(AllDailyFlowDataWithoutZeroFlowStations,staid==as.character(stations[i,1]))
  temp<-unique(temp)

```

```

    temp1.zoo<-zoo(temp[,-3],temp[,3])
    temp2 <- merge(temp1.zoo,zoo(,seq(start(temp1.zoo),end(temp1.zoo),by="day")),
all=TRUE)
    temp3<-as.data.frame(temp2)
    temp3$staid<-as.character(stations[i,1])
    temp3$dates = rownames(temp3)
    temp3<-temp3[,c(1,2,4,3)]
    AllDailyFlowDataNoGaps<-rbind(AllDailyFlowDataNoGaps,temp3)
  }

AllDailyFlowDataNoGaps$dates<-as_date(AllDailyFlowDataNoGaps$dates)
AllDailyFlowDataNoGaps$val<-as.numeric(AllDailyFlowDataNoGaps$val)

DailyFlowsMetadata$MissingValCount<-nrow(subset(AllDailyFlowDataNoGaps, is.na(val)))

#write data frames to file so that you don't have to do this again!
save(AllDailyFlowDataNoGaps,file= "data/AllDailyFlowDataNoGaps.RData")
save(DailyFlowsMetadata,file= "data/DailyFlowsMetadata.Rdata")


# Run this code separately before knitting the data document; it will write the
"AllInterpolatedFlowData.Rdata" file that will be used when generating the data
report.

# Identify subset of data where stations have at least one missing value, but less
than 40% missing values overall.

obscounts<-AllDailyFlowDataNoGaps%>%
  group_by(staid) %>%
  summarise(totalobs=n()) %>%
  as.data.frame()

NAs<- subset(AllDailyFlowDataNoGaps,is.na(val))
NAcounts<- NAs%>%
  group_by(staid) %>%
  summarise(NAobs=n(),na.rm=TRUE) %>%
  as.data.frame()

NAstats<-left_join(obscounts,NAcounts)
NAstats$percentNA<-100*NAstats$NAobs/NAstats$totalobs

StationsToInterpolate<-subset(NAstats,percentNA<40)
DataToInterpolate<-semi_join(AllDailyFlowDataNoGaps,StationsToInterpolate)
DataNotInterpolated<-anti_join(AllDailyFlowDataNoGaps,StationsToInterpolate)

# Initialize dataframe to store flow data with interpolated values. This seeds the
table with the zero-flow data from the prior step, which are omitted from
interpolation procedures.

rm(AllInterpolatedFlowData_temp)
AllInterpolatedFlowData_temp<-setNames(data.frame(matrix(ncol = 5, nrow = 0)),
c("staid", "val", "dates", "val2","qualcode"))

```

```

AllInterpolatedFlowData_temp$staid<-as.character(AllInterpolatedFlowData_temp$staid)
AllInterpolatedFlowData_temp$dates<-as.Date(AllInterpolatedFlowData_temp$dates)
AllInterpolatedFlowData_temp$val<-as.numeric(AllInterpolatedFlowData_temp$val)
AllInterpolatedFlowData_temp$qualcode<-
as.character(AllInterpolatedFlowData_temp$qualcode)
AllInterpolatedFlowData_temp$val2<-as.numeric(AllInterpolatedFlowData_temp$val2)
AllInterpolatedFlowData_temp<-as.data.frame(AllInterpolatedFlowData_temp)

# Begin interpolations

stations<-unique(subset(DataToInterpolate,select="staid"))
stationcount<-nrow(stations)

# for (i in 1:1) {
for (i in 1:stationcount) {
  progress(i/stationcount*100)
  # subset AllDailyFlowData by station and order by date
  temp<-as.data.frame(subset(DataToInterpolate,staid==as.character(stations[i,1])))

  #interpolate missing data for gaps of two days or less

  temp2<-as.data.frame(fillMissing(temp$val, span=3, max.fill=2))
  temp$val2<-temp2$fillMissing(temp$val, span = 3, max.fill = 2)`

  # Append missing data to daily flow data frame
  AllInterpolatedFlowData_temp<-rbind(AllInterpolatedFlowData_temp,temp)
  cat("\n")

  if (i == stationcount) cat("Done!\n")
}

AllInterpolatedFlowData_temp$qualcode<-
ifelse(is.na(AllInterpolatedFlowData_temp$val)&!is.na(AllInterpolatedFlowData_temp$val
2),"fM",as.character(AllInterpolatedFlowData_temp$qualcode))

AllInterpolatedFlowData_temp$val<-AllInterpolatedFlowData_temp$val2
AllInterpolatedFlowData_temp<-AllInterpolatedFlowData_temp[,c(1:4)]

AllInterpolatedFlowData<-rbind(AllInterpolatedFlowData_temp,DataNotInterpolated)

AllInterpolatedFlowData<-subset(AllInterpolatedFlowData, !is.na(val))

#write data frame to file so that you don't have to do this again!
save(AllInterpolatedFlowData,file="data/InterpolatedDailyFlowData.RData")

# The following creates the data frame of mean daily flow data for publication.
rm(Output_MeanDailyFlows)
Output_MeanDailyFlows<-AllInterpolatedFlowData

```

```

# Test for Provisional data (yes [MPY],no [MPN])
Output_MeanDailyFlows$ProvisionalQual<-
ifelse(grepl("P",Output_MeanDailyFlows$qualcode,ignore.case = FALSE)==TRUE,
      "MPY",
      "MPN")

# Test for Estimated Values (yes [MEY],no [MEN])
Output_MeanDailyFlows$EstimatedQual<-
ifelse(grepl("e",Output_MeanDailyFlows$qualcode,ignore.case = FALSE)==TRUE,
      "MEY",
      "MEN")

# Test for Interpolated Values (yes [MfMY],no [MfMN])
Output_MeanDailyFlows$InterpolatedQual<-ifelse(Output_MeanDailyFlows$qualcode=="fM",
      "MfMY",
      "MfMN")

# Test for Data known to be above/below recorded value (censored) (yes [MCY],no [MCN])
Output_MeanDailyFlows$CensoredDataQual<-ifelse(Output_MeanDailyFlows$qualcode=="A >" |
Output_MeanDailyFlows$qualcode=="A <",
      "MCY",
      "MCN")

# Clean up field data formats
Output_MeanDailyFlows<-as.data.frame(Output_MeanDailyFlows)
Output_MeanDailyFlows$ProvisionalQual<-factor(Output_MeanDailyFlows$ProvisionalQual)
Output_MeanDailyFlows$EstimatedQual<-factor(Output_MeanDailyFlows$EstimatedQual)
Output_MeanDailyFlows$InterpolatedQual<-factor(Output_MeanDailyFlows$InterpolatedQual)
Output_MeanDailyFlows$CensoredDataQual<-factor(Output_MeanDailyFlows$CensoredDataQual)

#write data frame to file so that you don't have to do this again!
save(Output_MeanDailyFlows,file="./data/Output_MeanDailyFlows.RData", compress=TRUE)

BasePORTable <- matrix(c("1913-10-01","1917-09-30","1918-10-01","1921-09-30","1928-10-
01","2017-09-30"),ncol=2,byrow=TRUE)

T2<-theme_vanilla(regulartable(as.data.frame(BasePORTable)))
T2<-autofit(T2)
T2<-set_header_labels(T2,V1="Start Date",V2="End Date")
T2<-align(T2, align = "center", part="all")
T2

AnalysisPORTable <- matrix(c("1913-10-01","1914-09-30","1913-10-01","1915-09-
30","1913-10-01","1916-09-30","1913-10-01","1917-09-30"),ncol=2,byrow=TRUE)

T3<-theme_vanilla(regulartable(as.data.frame(AnalysisPORTable)))
T3<-autofit(T3)
T3<-set_header_labels(T3,V1="Start Date",V2="End Date")
T3<-align(T3, align = "center", part="all")
T3

```

```

# Get number of stations with daily flow data, including those with interpolated data
interpolatedstations<-unique(subset(Output_MeanDailyFlows,select="staid"))
interpolatedstationcount<-nrow(interpolatedstations)

# Initialize BasePORs table
rm(BasePORs)
BasePORs<- data.frame(
  site_no = character(),
  POR_begin_date = as.Date(character()),
  POR_end_date = as.Date(character()),
  Year_Type = as.Date(character()),
  AnalysisStartYear = as.Date(character()),
  AnalysisEndYear = as.Date(character()),
  AnalysisStartDate = as.Date(character()),
  AnalysisEndDate = as.Date(character()),
  stringsAsFactors = FALSE
)

for (i in 1:interpolatedstationcount) {

  site_no<-interpolatedstations[i,1]

  # get daily flow data for a site
  dailyQ <-
subset(Output_MeanDailyFlows,staid==site_no,select=c("staid","val","dates"))

  # get first and last date of any data gaps, generate data gap table. If it contains
  any records, the full POR is not usable and must be split into sub-PORs for analysis
  startgaps<-dailyQ[diff(dailyQ$dates)>1,]
  endgaps<-dailyQ[c(1, diff(dailyQ$dates))>1,]

  rm(dfgaps)
  dfgaps<-setNames(data.frame(matrix(ncol=2,nrow=0)), c("start","end"))

  for (k in 1:nrow(endgaps)){

    dfgaps<-rbind(dfgaps,data.frame(start=startgaps[k,]$dates,end=endgaps[k,]$dates))

  }

  # get first and last date of full POR
  startdate<-min(dailyQ$dates)
  enddate<-max(dailyQ$dates)

  # construct continuous sub PORs based on start and end dates of data gaps
  rm(PORs)
  PORs<-setNames(data.frame(matrix(ncol=2,nrow=0)), c("start","end"))

  if (is.na(dfgaps[1,1])) {

```

```

PORs<-rbind(PORs,data.frame(start=startdate, end=enddate))

} else if (nrow(dfgaps)==1) {

  # construct first sub-POR
  PORs<-rbind(PORs,data.frame(start=startdate, end=min(dfgaps$start)))

  # construct last sub POR
  PORs<-rbind(PORs,data.frame(start=max(dfgaps$end), end=enddate))

} else {

  # construct first sub-POR
  PORs<-rbind(PORs,data.frame(start=startdate, end=min(dfgaps$start)))

  # get intermediate sub-PORs if there is more than one data gap

  for (j in 1:(nrow(dfgaps)-1)) {

    PORs<-rbind(PORs,data.frame(start=dfgaps$end[j], end=dfgaps$start[j+1]))

  }

  # construct last sub POR
  PORs<-rbind(PORs,data.frame(start=max(dfgaps$end), end=enddate))

}

rm(POR_out)
POR_out<-
as.data.frame(cbind(site_no,POR_begin_date=as.character(PORs$start),POR_end_date=as.ch
aracter(PORs$end),Year_Type="water",AnalysisStartYear="", AnalysisEndYear="",
AnalysisStartDate="", AnalysisEndDate=""))

# if start date is a leap year then assign start year based on day of year

POR_out[,7]<-ifelse(

  leap_year(as.Date(POR_out[,2]))== TRUE,

  ifelse(
    yday(POR_out[,2]) > 275,
    paste(year(POR_out[,2])+1,"-10-01", sep=""),
    paste(year(POR_out[,2]),"-10-01", sep="")
  ),

  ifelse(
    yday(POR_out[,2]) > 274,
    paste(year(POR_out[,2])+1,"-10-01", sep=""),
    paste(year(POR_out[,2]),"-10-01", sep="")
  )
)

```

```

    )
  )

#repeat for end dates.

POR_out[,8]<-ifelse(

  leap_year(as.Date(POR_out[,3]))== TRUE,

  ifelse(
    yday(POR_out[,3]) < 275,
    paste(year(POR_out[,3])-1,"-09-30", sep=""),
    paste(year(POR_out[,3]),"-09-30", sep="")
  ),

  ifelse(
    yday(POR_out[,3]) < 274,
    paste(year(POR_out[,3])-1,"-09-30", sep=""),
    paste(year(POR_out[,3]),"-09-30", sep="")
  )
)

# Get Analysis Water Years for based on Analysis Start and End Dates
POR_out[,5]<-as.character(waterYear(POR_out[,7]))
POR_out[,6]<-as.character(waterYear(POR_out[,8]))

BasePORs<-rbind(BasePORs,POR_out)

}

# Remove PORs less than two years in length
BasePORs<-subset(BasePORs,as.numeric(AnalysisEndYear)-
as.numeric(AnalysisStartYear)>=2)
colnames(BasePORs)[2] <- "StationPORStartDate"
colnames(BasePORs)[3] <- "StationPOREndDate"
colnames(BasePORs)[5] <- "BasePORStartYear"
colnames(BasePORs)[6] <- "BasePOREndYear"
colnames(BasePORs)[7] <- "BasePORStartDate"
colnames(BasePORs)[8] <- "BasePOREndDate"

#write BasePORs data frame to file so that you don't have to do this again!
save(BasePORs,file="data/BasePORs.RData",compress=TRUE)
tlu_PORsTemp<-subset(tlu_PORs,YearType=="WY", select=c("POR_ID", "StartYear",
"EndYear"))
colnames(tlu_PORsTemp)[2]<-"BasePORStartYear"
tlu_PORsTemp$BasePORStartYear<-as.character(tlu_PORsTemp$BasePORStartYear)
AnalysisPORs<-inner_join(BasePORs,tlu_PORsTemp)
colnames(AnalysisPORs)[10]<-"AnalysisPOREndYear"
AnalysisPORs$AnalysisPORStartYear<-AnalysisPORs$BasePORStartYear
AnalysisPORs<-AnalysisPORs[,c(1:9,11,10)]
AnalysisPORs$AnalysisPORStartDate<-AnalysisPORs$BasePORStartDate
AnalysisPORs$AnalysisPOREndDate<-paste(AnalysisPORs$AnalysisPOREndYear,"-09-30",

```

```

sep="")

#remove Analysis PORs less than 2 years long
AnalysisPORs<-subset(AnalysisPORs,as.numeric(AnalysisPOREndYear)-
as.numeric(AnalysisPORStartYear)>=2)

#remove Analysis PORs that extend past the Base POR end year
AnalysisPORs<-
subset(AnalysisPORs,as.numeric(AnalysisPOREndYear)<=as.numeric(BasePOREndYear))

#write AnalysisPORs data frame to file so that you don't have to do this again!
save(AnalysisPORs,file="data/AnalysisPORs.RData",compress=TRUE)
AnalysisPORs2<-AnalysisPORs
AnalysisPORs2$staid<-as.character(AnalysisPORs2$site_no)

# Qualify POR based on the number of water years (<5 years [PLS], 5-15 years [PLI],
>15 years [PLL])
AnalysisPORs2$LengthQual<-"PLI"
AnalysisPORs2$LengthQual<-ifelse(as.numeric(AnalysisPORs2$AnalysisPOREndYear)-
as.numeric(AnalysisPORs2$AnalysisPORStartYear)>20,"PLL",AnalysisPORs2$LengthQual)
AnalysisPORs2$LengthQual<-ifelse(as.numeric(AnalysisPORs2$AnalysisPOREndYear)-
as.numeric(AnalysisPORs2$AnalysisPORStartYear)<5,"PLS",AnalysisPORs2$LengthQual)

# Contains Provisional Data (yes [PPY], no [PPN])
QualifiedFlowRecords_ProvisionalQual<-subset(Output_MeanDailyFlows,
ProvisionalQual=="MPY")

testProvisionalPORs<-
inner_join(AnalysisPORs2,QualifiedFlowRecords_ProvisionalQual,by="staid")
testProvisionalPORs<-subset(testProvisionalPORs,dates<=AnalysisPOREndDate &
dates>=AnalysisPORStartDate)

summaryStats_testProvisionalPORs<-testProvisionalPORs %>%
  group_by(staid,POR_ID) %>%
  summarize(PPY=n())

AnalysisPORs2<-full_join(AnalysisPORs2,summaryStats_testProvisionalPORs,by=c("staid",
"POR_ID"))

AnalysisPORs2$ProvisionalQual<-ifelse(is.na(AnalysisPORs2$PPY)==TRUE,
  "PPN",
  "PPY")

# Contains Estimated data (yes [PEY],no [PEN])
QualifiedFlowRecords_EstimatedQual<-subset(Output_MeanDailyFlows,
EstimatedQual=="MEY")

testEstimatedPORs<-
inner_join(AnalysisPORs2,QualifiedFlowRecords_EstimatedQual,by="staid")
testEstimatedPORs<-subset(testEstimatedPORs,dates<=AnalysisPOREndDate &
dates>=AnalysisPORStartDate)

```

```

summaryStats_testEstimatedPORs<-testEstimatedPORs %>%
  group_by(staid,POR_ID) %>%
  summarize(PEY=n())

AnalysisPORs2<-full_join(AnalysisPORs2,summaryStats_testEstimatedPORs,by=c("staid",
"POR_ID"))

AnalysisPORs2$EstimatedQual<-ifelse(is.na(AnalysisPORs2$PEY)==FALSE,
  "PEY",
  "PEN")

# Contains Interpolated data (yes [PEY],no [PEN])
QualifiedFlowRecords_InterpolatedQual<-subset(Output_MeanDailyFlows,
InterpolatedQual=="MfMY")

testInterpolatedPORs<-
inner_join(AnalysisPORs2,QualifiedFlowRecords_InterpolatedQual,by="staid")
testInterpolatedPORs<-subset(testInterpolatedPORs,dates<=AnalysisPOREndDate &
dates>=AnalysisPORStartDate)

summaryStats_testInterpolatedPORs<-testInterpolatedPORs %>%
  group_by(staid,POR_ID) %>%
  summarize(PfMY=n())

AnalysisPORs2<-full_join(AnalysisPORs2,summaryStats_testInterpolatedPORs,by=c("staid",
"POR_ID"))

AnalysisPORs2$InterpolatedQual<-ifelse(is.na(AnalysisPORs2$PfMY)==FALSE,
  "PIY",
  "PIN")

# Contains Data known to be above/below recorded value (yes [PCY],no [PCN])
QualifiedFlowRecords_CensoredDataQual<-subset(Output_MeanDailyFlows,
CensoredDataQual=="MCY")

testCensoredDataPORs<-
inner_join(AnalysisPORs2,QualifiedFlowRecords_CensoredDataQual,by="staid")
testCensoredDataPORs<-subset(testCensoredDataPORs,dates<=AnalysisPOREndDate &
dates>=AnalysisPORStartDate)

summaryStats_testCensoredDataPORs<-testCensoredDataPORs %>%
  group_by(staid,POR_ID) %>%
  summarize(PCY=n())

AnalysisPORs2<-full_join(AnalysisPORs2,summaryStats_testCensoredDataPORs,by=c("staid",
"POR_ID"))

AnalysisPORs2$CensoredDataQual<-ifelse(is.na(AnalysisPORs2$PCY)==FALSE,
  "PCY",
  "PCN")

```

```

# clean up and save data set
AnalysisPORs2<-subset (AnalysisPORs2,select=-c (staid,PPY,PEY,PfMY,PCY))

AnalysisPORs2$LengthQual<-factor (AnalysisPORs2$LengthQual)
AnalysisPORs2$ProvisionalQual<-factor (AnalysisPORs2$ProvisionalQual)
AnalysisPORs2$EstimatedQual<-factor (AnalysisPORs2$EstimatedQual)
AnalysisPORs2$InterpolatedQual<-factor (AnalysisPORs2$InterpolatedQual)
AnalysisPORs2$CensoredDataQual<-factor (AnalysisPORs2$CensoredDataQual)

Output_AnalysisPORs<-AnalysisPORs2

#clean up field formats
Output_AnalysisPORs$site_no<-as.character (Output_AnalysisPORs$site_no)
Output_AnalysisPORs$StationPORStartDate<-
as.Date (Output_AnalysisPORs$StationPORStartDate)
Output_AnalysisPORs$StationPOREndDate<-as.Date (Output_AnalysisPORs$StationPOREndDate)
Output_AnalysisPORs$BasePORStartDate<-as.Date (Output_AnalysisPORs$BasePORStartDate)
Output_AnalysisPORs$AnalysisPORStartDate<-
as.Date (Output_AnalysisPORs$AnalysisPORStartDate)
Output_AnalysisPORs$AnalysisPOREndYear<-
as.character (Output_AnalysisPORs$AnalysisPOREndYear)

#write AnalysisPORs data frame to file so that you don't have to do this again!
save (Output_AnalysisPORs,file="data/output_AnalysisPORs.RData",compress=TRUE)

# Create Stats for Text
BasePORCount<-BasePORs %>% group_by (site_no) %>% summarise (count=n ())
BasePORCountMax<-max (BasePORCount$count)
BasePORCountMin<-min (BasePORCount$count)

AnalysisPORCount<-Output_AnalysisPORs %>% group_by (site_no) %>% summarise (count=n ())
AnalysisPORCountMax<-max (AnalysisPORCount$count)
AnalysisPORCountMin<-min (AnalysisPORCount$count)

StationsWithBasePORs<-nrow (unique (subset (BasePORs,select="site_no")))
StationsWithoutBasePORs<-nrow (unique (subset (Output_MeanDailyFlows,select="staid")))-
nrow (unique (subset (BasePORs,select="site_no"))))

# Generate Table of Qualification codes
dataset1<-c ("USGS National Water Information System
(http://www.waterdata.usgs.gov/nwis)",
"NPS_IMD_ESProtocol_Hydrology_2018_All_Daily_Flows_2263556-data.csv",
formatC (nrow (AllDailyFlowData), big.mark=","),
paste0 (min (AllDailyFlowData$Date),
" to ",
max (AllDailyFlowData$Date)))
)

dataset2<-c ("NPS_IMD_ESProtocol_Hydrology_2018_All_Daily_Flows_2263556-data.csv",
"NPS_IMD_ESProtocol_Hydrology_2018_Clean_Daily_Flows_2263592-data.csv",

```

```

formatC(nrow(Output_MeanDailyFlows), big.mark=","),
paste0(min(Output_MeanDailyFlows$dates),
       " to ",
       max(Output_MeanDailyFlows$dates))
)

dataset3<-c("NPS_IMD_ESProtocol_Hydrology_2018_Clean_Daily_Flows_2263592-data.csv",
            "NPS_IMD_ESProtocol_Hydrology_2018_Analysis_PORs_2263593-data.csv",
            formatC(nrow(Output_AnalysisPORs), big.mark=","),
            paste0(min(Output_AnalysisPORs$AnalysisPORStartDate),
                   " to ",
                   max(Output_AnalysisPORs$AnalysisPOREndDate)))

alldatasets<-as.data.frame(rbind(dataset1,dataset2,dataset3))

# Create Table
datasetTable<-theme_vanilla(regulartable(alldatasets))
datasetTable<-autofit(datasetTable)
datasetTable<-set_header_labels(datasetTable,V1="Source Data",V2="Resultant Data",
V3="Number of Records / Measurements",V4="Temporal Range")
datasetTable<-align(datasetTable, align = "left", part="body")
datasetTable<-align(datasetTable, align = "center", part="header")
datasetTable<-align(datasetTable, j=3,align = "right")
datasetTable

# Generate Table of Qualification codes
factors1 <- read.csv("metadata/factors_AnalysisPORs.csv", stringsAsFactors = FALSE)
factors1$appliedto <- "Periods of Record"
factors2 <- read.csv("metadata/factors_CleanDailyFlowsData.csv", stringsAsFactors =
FALSE)
factors2$appliedto <- "Mean Daily Flow Measurements"
allfactors<-rbind(factors1,factors2)

dqcodes<-rbind(allfactors[36:43,], allfactors[3:13,])
dqcodes<-dqcodes[c(-1)]

# Create Table
Tdqs<-theme_vanilla(regulartable(dqcodes))
Tdqs<-autofit(Tdqs)
Tdqs<-set_header_labels(Tdqs,code="Qualification
Code",definition="Definition",appliedto="Applies To")
Tdqs<-align(Tdqs, align = "left", part="body")
Tdqs<-align(Tdqs, align = "left", part="header")
Tdqs

# Generate Stats for Table and Text

FlowRecordCount<-nrow(Output_MeanDailyFlows)
AcceptedSubset<-subset(subset(Output_MeanDailyFlows,
                             Output_MeanDailyFlows$qualcode=="A" |

```

```

        Output_MeanDailyFlows$qualcode=="A >" |
        Output_MeanDailyFlows$qualcode=="A [4]" |
        Output_MeanDailyFlows$qualcode=="A R" |
        Output_MeanDailyFlows$qualcode=="A <" |
        Output_MeanDailyFlows$qualcode=="A e"))
ProvisionalSubset<-subset(Output_MeanDailyFlows,
Output_MeanDailyFlows$ProvisionalQual=="MPY")
OtherSubset<-subset(Output_MeanDailyFlows,
        Output_MeanDailyFlows$qualcode!="A" &
        Output_MeanDailyFlows$qualcode!="A >" &
        Output_MeanDailyFlows$qualcode!="A <" &
        Output_MeanDailyFlows$qualcode!="A e" &
        Output_MeanDailyFlows$qualcode!="A [4]" &
        Output_MeanDailyFlows$qualcode!="A R" &
        Output_MeanDailyFlows$ProvisionalQual!="MPY")

## Overall Counts
AcceptedTotal<-nrow(AcceptedSubset)
AcceptedTotalPercent<-round((100*AcceptedTotal/FlowRecordCount),digits=1)
AcceptedTableText<-paste(formatC(AcceptedTotal, big.mark=","), "
(",formatC(AcceptedTotalPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ", sep="")

ProvisionalTotal<-nrow(ProvisionalSubset)
ProvisionalTotalPercent<-round((100*ProvisionalTotal/FlowRecordCount),digits=1)
ProvisionalTableText<-paste(formatC(ProvisionalTotal, big.mark=","), "
(",formatC(ProvisionalTotalPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ", sep="")

OtherTotal<-nrow(OtherSubset)
OtherTotalPercent<-round((100*OtherTotal/FlowRecordCount),digits=1)
OtherTableText<-paste(formatC(OtherTotal, big.mark=","), "
(",formatC(OtherTotalPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ", sep="")

TotalTotal<-AcceptedTotal+ProvisionalTotal+OtherTotal
TotalTableText<-paste(formatC(TotalTotal, big.mark=","), "          ")

Fc1=c(AcceptedTableText,ProvisionalTableText,OtherTableText,TotalTableText)

## Overall Counts of Records without Qualifications
AcceptedNotFlaggedTotal<-AcceptedTotal-nrow(subset(AcceptedSubset,
AcceptedSubset$EstimatedQual=="MEY" | AcceptedSubset$InterpolatedQual=="MfMY" |
AcceptedSubset$CensoredDataQual=="MCY"))
AcceptedNotFlaggedPercent<-
round((100*AcceptedNotFlaggedTotal/FlowRecordCount),digits=1)
AcceptedNotFlaggedTableText<-paste(formatC(AcceptedNotFlaggedTotal, big.mark=","), "
(",formatC(AcceptedNotFlaggedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ", sep="")

ProvisionalNotFlaggedTotal<-ProvisionalTotal-nrow(subset(ProvisionalSubset,
ProvisionalSubset$EstimatedQual=="MEY" | ProvisionalSubset$InterpolatedQual=="MfMY" |
ProvisionalSubset$CensoredDataQual=="MCY"))

```

```

ProvisionalNotFlaggedPercent<-
round((100*ProvisionalNotFlaggedTotal/FlowRecordCount),digits=1)
ProvisionalNotFlaggedTableText<-paste(formatC(ProvisionalNotFlaggedTotal,
big.mark=","), "
(",formatC(ProvisionalNotFlaggedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

OtherNotFlaggedTotal<-OtherTotal-nrow(subset(OtherSubset,
OtherSubset$EstimatedQual=="MEY" | OtherSubset$InterpolatedQual=="MfMY" |
OtherSubset$CensoredDataQual=="MCY"))
OtherNotFlaggedPercent<-round((100*OtherNotFlaggedTotal/FlowRecordCount),digits=1)
OtherNotFlaggedTableText<-paste(formatC(OtherNotFlaggedTotal, big.mark=","), "
(",formatC(OtherNotFlaggedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

NotFlaggedTotal<-
AcceptedNotFlaggedTotal+ProvisionalNotFlaggedTotal+OtherNotFlaggedTotal
NotFlaggedTotalTableText<-paste(formatC(NotFlaggedTotal, big.mark=","), "
")

Fc2=c(AcceptedNotFlaggedTableText,ProvisionalNotFlaggedTableText,OtherNotFlaggedTableT
ext,NotFlaggedTotalTableText)

## Overall Counts of Records with values estimated by USGS
AcceptedEstimatedTotal<-nrow(subset(AcceptedSubset,
AcceptedSubset$EstimatedQual=="MEY"))
AcceptedEstimatedPercent<-round((100*AcceptedEstimatedTotal/FlowRecordCount),digits=1)
AcceptedEstimatedTableText<-paste(formatC(AcceptedEstimatedTotal, big.mark=","), "
(",formatC(AcceptedEstimatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

ProvisionalEstimatedTotal<-nrow(subset(ProvisionalSubset,
ProvisionalSubset$EstimatedQual=="MEY"))
ProvisionalEstimatedPercent<-
round((100*ProvisionalEstimatedTotal/FlowRecordCount),digits=1)
ProvisionalEstimatedTableText<-paste(formatC(ProvisionalEstimatedTotal, big.mark=","),
" (",formatC(ProvisionalEstimatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

OtherEstimatedTotal<-nrow(subset(OtherSubset, OtherSubset$EstimatedQual=="MEY"))
OtherEstimatedPercent<-round((100*OtherEstimatedTotal/FlowRecordCount),digits=1)
OtherEstimatedTableText<-paste(formatC(OtherEstimatedTotal, big.mark=","), "
(",formatC(OtherEstimatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

EstimatedTotal<-AcceptedEstimatedTotal+ProvisionalEstimatedTotal+OtherEstimatedTotal
EstimatedTotalTableText<-paste(formatC(EstimatedTotal, big.mark=","), "
")

Fc3=c(AcceptedEstimatedTableText,ProvisionalEstimatedTableText,OtherEstimatedTableText
,EstimatedTotalTableText)

## Overall Counts of Records with values interpolated by NPS
AcceptedInterpolatedTotal<-nrow(subset(AcceptedSubset,

```

```

AcceptedSubset$InterpolatedQual=="MfMY"))
AcceptedInterpolatedPercent<-
round((100*AcceptedInterpolatedTotal/FlowRecordCount),digits=1)
AcceptedInterpolatedTableText<-paste(formatC(AcceptedInterpolatedTotal, big.mark=","),
" (",formatC(AcceptedInterpolatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

ProvisionalInterpolatedTotal<-nrow(subset(ProvisionalSubset,
ProvisionalSubset$InterpolatedQual=="MfmY"))
ProvisionalInterpolatedPercent<-
round((100*ProvisionalInterpolatedTotal/FlowRecordCount),digits=1)
ProvisionalInterpolatedTableText<-paste(formatC(ProvisionalInterpolatedTotal,
big.mark=","), "
(",formatC(ProvisionalInterpolatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

OtherInterpolatedTotal<-nrow(subset(OtherSubset,
OtherSubset$InterpolatedQual=="MfMY"))
OtherInterpolatedPercent<-round((100*OtherInterpolatedTotal/FlowRecordCount),digits=1)
OtherInterpolatedTableText<-paste(formatC(OtherInterpolatedTotal, big.mark=","), "
(",formatC(OtherInterpolatedPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

InterpolatedTotal<-
AcceptedInterpolatedTotal+ProvisionalInterpolatedTotal+OtherInterpolatedTotal
InterpolatedTotalTableText<-paste(formatC(InterpolatedTotal, big.mark=","), "
")

Fc4=c(AcceptedInterpolatedTableText,ProvisionalInterpolatedTableText,OtherInterpolated
TableText,InterpolatedTotalTableText)

## Overall Counts of Records with values flagged as Censored based on USGS
qualification codes
AcceptedCensoredTotal<-nrow(subset(AcceptedSubset,
AcceptedSubset$CensoredDataQual=="MCY"))
AcceptedCensoredPercent<-round((100*AcceptedCensoredTotal/FlowRecordCount),digits=1)
AcceptedCensoredTableText<-paste(formatC(AcceptedCensoredTotal, big.mark=","), "
(",formatC(AcceptedCensoredPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

ProvisionalCensoredTotal<-nrow(subset(ProvisionalSubset,
ProvisionalSubset$CensoredDataQual=="MfmY"))
ProvisionalCensoredPercent<-
round((100*ProvisionalCensoredTotal/FlowRecordCount),digits=1)
ProvisionalCensoredTableText<-paste(formatC(ProvisionalCensoredTotal, big.mark=","), "
(",formatC(ProvisionalCensoredPercent,width=4,format="f",digits=1,drop0trailing =
FALSE), "%) ",sep="")

OtherCensoredTotal<-nrow(subset(OtherSubset, OtherSubset$CensoredDataQual=="MfmY"))
OtherCensoredPercent<-round((100*OtherCensoredTotal/FlowRecordCount),digits=1)
OtherCensoredTableText<-paste(formatC(OtherCensoredTotal, big.mark=","),
" (",

```

```

formatC(OtherCensoredPercent,digits=1,width=4,format="f",drop0trailing = FALSE),
        "%) ",
        sep="")

CensoredTotal<-AcceptedCensoredTotal+ProvisionalCensoredTotal+OtherCensoredTotal
Censored1TableText<-paste(formatC(CensoredTotal, big.mark=","), "      ")

Fc5=c(AcceptedCensoredTableText,ProvisionalCensoredTableText,OtherCensoredTableText,Ce
nsored1TableText)

# Generate Table of QualCounts
qualcounttable = data.frame(Cat=c("Accepted (by USGS)", "Provisional (by
USGS)", "Other", "Total"),Fc1,Fc2,Fc3,Fc4,Fc5)
rownames(qualcounttable)<-c("Accepted (by USGS)", "Provisional (by
USGS)", "Other", "Total")

#Print Table to Report
T4<-theme_vanilla(regulartable(qualcounttable))
T4<-autofit(T4)
T4<-set_header_labels(T4,Cat="",Fc1="Total",Fc2="No Qualifier Code",Fc3="Estimated (by
USGS)",Fc4="Interpolated (by NPS)",Fc5="Censored")
T4<-align(T4, align = "right", part="body")
T4<-align(T4, align = "center", part="header")
T4<-align(T4, j=1,align = "left")
T4

## Generate Stats for Table and Text

## Available Data from USGS. Note that is includes data gaps and potentially multiple
PORs per station.
StationPORsSubset <- AllDailyFlowData %>%
  group_by(site_no) %>%
  summarize(begin_date = min(Date),end_date = max(Date)) %>%
  arrange(site_no,begin_date,end_date)

colnames(StationPORsSubset)[1]<-"staid"

StationPORsSubset$LengthQual<-as.numeric(difftime(as_date(StationPORsSubset$end_date),
as_date(StationPORsSubset$begin_date), unit="weeks"))/52.25

StationPLSubset<-subset(StationPORsSubset, StationPORsSubset$LengthQual<=5)
StationPLISubset<-subset(StationPORsSubset, StationPORsSubset$LengthQual>5 &
StationPORsSubset$LengthQual<=20)
StationPLLSubset<-subset(StationPORsSubset, StationPORsSubset$LengthQual>20)

# Station Counts
StationPLSCount<-nrow(unique(subset(StationPLSubset,select=staid)))
StationPLICount<-nrow(unique(subset(StationPLISubset,select=staid)))
StationPLLCount<-nrow(unique(subset(StationPLLSubset,select=staid)))
StationPPYTableText<-NA

```

```

StationPEYTableText<-NA
StationPIYTableText<-NA
StationPCYTableText<-NA

StationPORTotalSites<-nrow(unique(subset(StationPORsSubset,select=staid)))

Pc1=c(StationPLSCount,StationPLICount,StationPLLCount,StationPPYTableText,StationPEYTableText,StationPIYTableText, StationPCYTableText,StationPORTotalSites)

## Base PORs
BasePORsSubset <- Output_AnalysisPORs %>%
  group_by(site_no, BasePORStartYear, BasePOREndYear) %>%
  filter(AnalysisPOREndYear == max(AnalysisPOREndYear)) %>%
  arrange(site_no,BasePORStartYear,BasePOREndYear)

BasePORsSubset<-
unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear,BasePOREndYear,LengthQual,ProvisionalQual,EstimatedQual,InterpolatedQual,CensoredDataQual)))

# Base POR Station Counts by Qualification Code
BasePLSCount<-nrow(unique(subset(BasePORsSubset,select=site_no,LengthQual=="PLS")))
BasePLICount<-nrow(unique(subset(BasePORsSubset,select=site_no,LengthQual=="PLI")))
BasePLLCount<-nrow(unique(subset(BasePORsSubset,select=site_no,LengthQual=="PLL")))
BasePPYCount<-
nrow(unique(subset(BasePORsSubset,select=site_no,ProvisionalQual=="PPY")))
BasePEYCount<-nrow(unique(subset(BasePORsSubset,select=site_no,EstimatedQual=="PEY")))
BasePIYCount<-
nrow(unique(subset(BasePORsSubset,select=site_no,InterpolatedQual=="PIY")))
BasePCYCount<-
nrow(unique(subset(BasePORsSubset,select=site_no,CensoredDataQual=="PCY")))
BasePORTotalSites<-nrow(unique(subset(BasePORsSubset,select=site_no)))

Pc3=c(BasePLSCount,BasePLICount,BasePLLCount,BasePPYCount,BasePEYCount,BasePIYCount,BasePCYCount,BasePORTotalSites)

# BasePOR POR Counts by qualification code
BasePLSPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),LengthQual=="PLS")))
BasePLIPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),LengthQual=="PLI")))
BasePLLPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),LengthQual=="PLL")))
BasePPYPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),ProvisionalQual=="PPY")))
BasePEYPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),EstimatedQual=="PEY")))
BasePIYPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),InterpolatedQual=

```

```

="PIY"))))
BasePCYPORCount<-
nrow(unique(subset(BasePORsSubset,select=c(site_no,BasePORStartYear),CensoredDataQual=
="PCY"))))
BasePORTotalPORs<-nrow(BasePORsSubset)

Pc4=c(BasePLSPORCount,BasePLIPORCount,BasePLLPORCount,BasePPYPORCount,BasePEYPORCount,
BasePIYPORCount,BasePCYPORCount,BasePORTotalPORs)

## Analysis PORs
AnalysisPORsSubset<-
unique(subset(Output_AnalysisPORs,select=c(site_no,AnalysisPORStartYear,AnalysisPOREnd
Year,LengthQual,ProvisionalQual,EstimatedQual,InterpolatedQual,CensoredDataQual)))

# AnalysisPOR Station Counts by Qualification Code
AnalysisPLSCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,LengthQual=="PLS"))))
AnalysisPLICount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,LengthQual=="PLI"))))
AnalysisPLLCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,LengthQual=="PLL"))))
AnalysisPPYCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,ProvisionalQual=="PPY"))))
AnalysisPEYCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,EstimatedQual=="PEY"))))
AnalysisPIYCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,InterpolatedQual=="PIY"))))
AnalysisPCYCount<-
nrow(unique(subset(AnalysisPORsSubset,select=site_no,CensoredDataQual=="PCY"))))
AnalysisPORTotalSites<-nrow(unique(subset(AnalysisPORsSubset,select=site_no)))

Pc5=c(AnalysisPLSCount,AnalysisPLICount,AnalysisPLLCount,AnalysisPPYCount,AnalysisPEYCo
unt,AnalysisPIYCount,AnalysisPCYCount,AnalysisPORTotalSites)

# AnalysisPOR POR Counts by qualification code
AnalysisPLSPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),LengthQual=
="PLS"))))
AnalysisPLIPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),LengthQual=
="PLI"))))
AnalysisPLLPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),LengthQual=
="PLL"))))
AnalysisPPYPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),Provisional
Qual=="PPY"))))
AnalysisPEYPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),EstimatedQu
al=="PEY"))))
AnalysisPIYPORCount<-
nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),Interpolate
dQual=="PIY"))))
AnalysisPCYPORCount<-

```

```

nrow(unique(subset(AnalysisPORsSubset,select=c(site_no,AnalysisPOREndYear),CensoredDataQual=="PCY")))
AnalysisPORTotalPORs<-nrow(AnalysisPORsSubset)

Pc6=c(AnalysisPLSPORCount,AnalysisPLIPORCount,AnalysisPLLPORCount,AnalysisPPYPORCount,
AnalysisPEYPORCount,AnalysisPIYPORCount,AnalysisPCYPORCount,AnalysisPORTotalPORs)

# Generate Table of QualCounts for PORs
PORqualcounttable = data.frame(cat=c("PORs <5 years","PORs 5-20 Years","PORs >20
Years","PORs with Provisional Data", "PORs with Estimated Data","PORs with
Interpolated Data", "PORs with Censored Data","Total"),Pc1,Pc3,Pc4,Pc5,Pc6)

T5<-theme_vanilla(regulartable(PORqualcounttable))
T5<-colformat_int(T5, col_keys=c("Pc1","Pc3","Pc4","Pc5","Pc6"),big.mark=",")
T5<-
set_header_labels(T5,cat="",Pc1="Stations",Pc3="Stations",Pc4="PORs",Pc5="Stations",Pc
6="PORs")
T5<-add_header(T5,cat="",Pc1="Available Data1",Pc3="Base PORs2", Pc4="Base PORs2",
Pc5="Analysis PORs2", Pc6="Analysis PORs2" )
T5<-merge_h(T5,part="header")
T5<-align(T5, align = "right", part="body")
T5<-align(T5, align = "center", part="header")
T5<-align(T5, j=1,align = "left")
autofit(T5)

sessionInfo()
Sys.time()

```

Appendix B. R Session and Version Information

```
sessionInfo()
R version 3.6.1 (2019-07-05)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] stats      graphics  grDevices  utils      datasets  methods
base

other attached packages:
[1] EML_2.0.0          zoo_1.8-6          svMisc_1.1.0
[4] reshape2_1.4.3     english_1.2-3      tibble_2.1.3
[7] flextable_0.5.5    EflowStats_5.0.1   waterData_1.0.8
[10] dataRetrieval_2.7.5 RODBC_1.3-15       smwrBase_1.1.5
[13] lubridate_1.7.4    devtools_2.1.0     usethis_1.5.1
[16] knitr_1.23          RefManageR_1.2.12  bibtex_0.4.2
[19] htmltools_0.3.6    rmdformats_0.3.5   yaml_2.2.0
[22] rmdHelpers_1.2     dplyr_0.8.3        texreg_1.36.23
[25] papeR_1.0-4        xtable_1.8-4       car_3.0-3
[28] carData_3.0-2      kimisc_0.4         R.rsp_0.43.1
[31] dataMaid_1.3.1     bookdownplus_1.5.7 bookdown_0.12
[34] knitr_1.23         pander_0.6.3       rmarkdown_1.14
[37] markdown_1.0

loaded via a namespace (and not attached):
[1] uuid_0.1-2          readxl_1.3.1       backports_1.1.4
[4] xaringan_0.11       plyr_1.8.4         imputeTS_3.0
[7] lazyeval_0.2.2      jqr_1.1.0          ggplot2_3.2.0
[10] digest_0.6.20       magick_2.1         gdata_2.18.0
[13] magrittr_1.5        memoise_1.1.0      emld_0.2.0
[16] openxlsx_4.1.0.1    remotes_2.1.0      readr_1.3.1
[19] gmodels_2.18.1      officer_0.3.5      R.utils_2.9.0
```

```

[22] xts_0.11-2          forecast_8.7         tseries_0.10-47
[25] prettyunits_1.0.2   colorspace_1.4-1     haven_2.1.1
[28] xfun_0.8            callr_3.3.1         crayon_1.3.4
[31] jsonlite_1.6        zeallot_0.1.0       glue_1.3.1
[34] gtable_0.3.0        V8_2.3              questionr_0.7.0
[37] R.cache_0.13.0      pkgbuild_1.0.3      quantmod_0.4-15
[40] DEoptimR_1.0-8      abind_1.4-5         scales_1.0.0
[43] stinepack_1.4       miniUI_0.1.1.1      Rcpp_1.0.2
[46] foreign_0.8-71      httr_1.4.0          RColorBrewer_1.1-2
[49] pkgconfig_2.0.2     R.methodsS3_1.7.1   nnet_7.3-12
[52] tidyselect_0.2.5    rlang_0.4.0         later_0.8.0
[55] munsell_0.5.0       cellranger_1.1.0    tools_3.6.1
[58] cli_1.1.0           evaluate_0.14        stringr_1.4.0
[61] processx_3.4.1      fs_1.3.1            jsonld_2.1
[64] zip_2.0.3           robustbase_0.93-5   purrr_0.3.2
[67] nlme_3.1-140        mime_0.7             formatR_1.7
[70] R.oo_1.22.0         RcppRoll_0.3.0      xml2_1.2.0
[73] compiler_3.6.1      rstudioapi_0.10     curl_4.0
[76] testthat_2.2.0      stringi_1.4.3       highr_0.8
[79] ps_1.3.0            desc_1.2.0          gdtools_0.1.9
[82] forcats_0.4.0       lattice_0.20-38     urca_1.3-0
[85] vctrs_0.2.0         pillar_1.4.2        lmtest_0.9-37
[88] data.table_1.12.2   lmom_2.8            httpuv_1.5.1
[91] R6_2.4.0            latticeExtra_0.6-28 promises_1.0.1
[94] gridExtra_2.3       rio_0.5.16          sessioninfo_1.1.1
[97] MASS_7.3-51.4       gtools_3.8.1        assertthat_0.2.1
[100] pkgload_1.0.2       rprojroot_1.3-2     withr_2.1.2
[103] fracdiff_1.4-2      parallel_3.6.1      hms_0.5.0
[106] quadprog_1.5-7      grid_3.6.1          timeDate_3043.102
[109] TTR_0.23-4          base64enc_0.1-3     shiny_1.3.2
Sys.time()
[1] "2019-08-02 11:24:45 MDT"

```

The Department of the Interior protects and manages the nation's natural resources and cultural heritage; provides scientific and other information about those resources; and honors its special responsibilities to American Indians, Alaska Natives, and affiliated Island Communities.

NPS 909/159885, August 2019

National Park Service
U.S. Department of the Interior



[Natural Resource Stewardship and Science](#)

1201 Oakridge Drive, Suite 150
Fort Collins, CO 80525