CA Stream Gage Dashboard

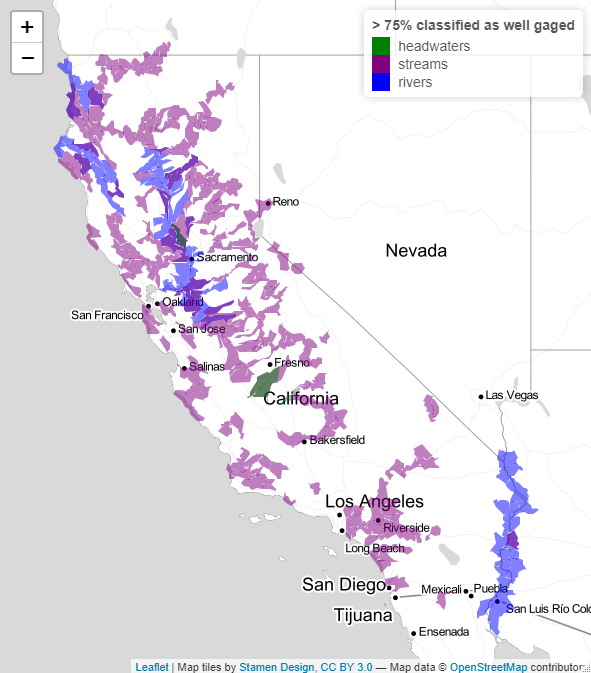
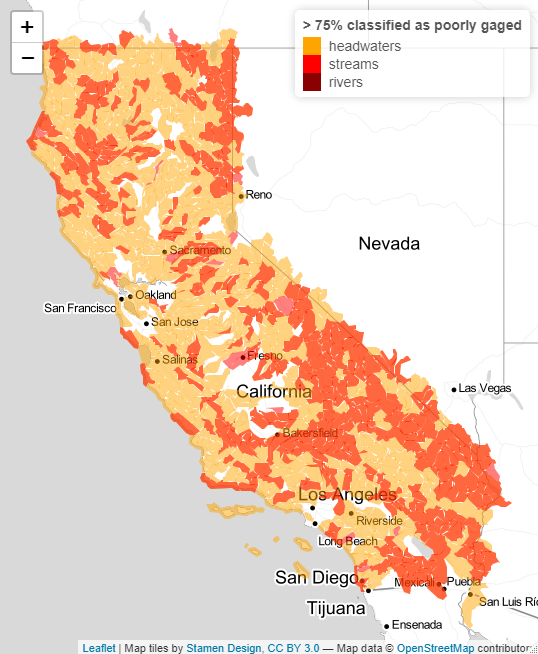
# Process

1. Access all the data from geoconnex, nhdplus, and CA State Google Drive
2. Create a single huc 10 and huc 12 file with columns needed for dashboard
3. Create a single gage file … right now just includes FIRO
4. Clean up text in stream gage file to make it easier to explore metadata

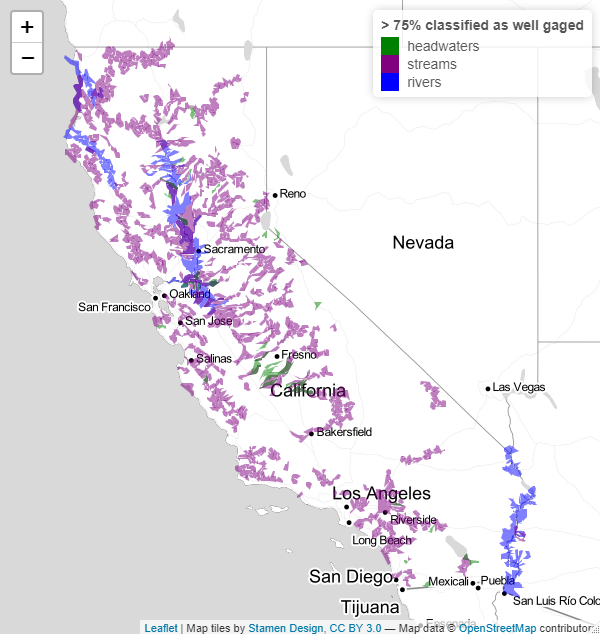
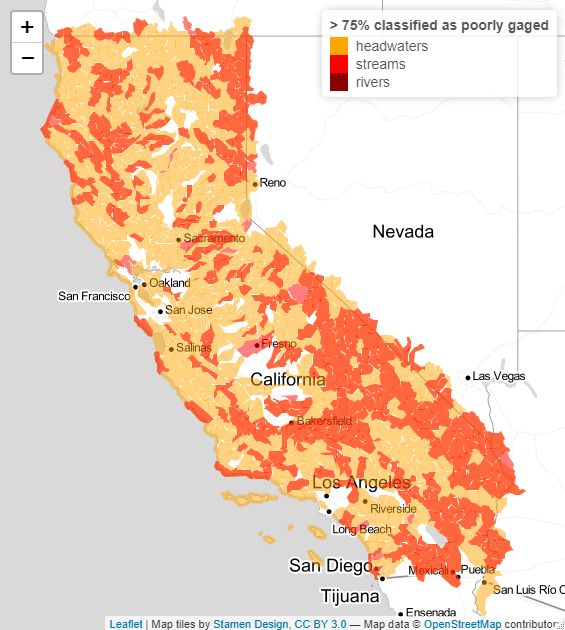
# Visualizations

* Highlight HUC 10 or HUC 12 watersheds that have more than 50% of headwaters, streams, and rivers classified as well gaged
  + Headwaters = stream order 1 to 3
  + Streams = stream order 4 to 6
  + Rivers = stream order 7 to 9
* Tally the river segments classified as poorly, somewhat, and well gaged by classification
* Calculate the percent of each segment classified as well gaged, somewhat gaged, and poorly gaged.

Plot of rivers, streams, and headwaters where more than 75% of those segments classified as well gaged

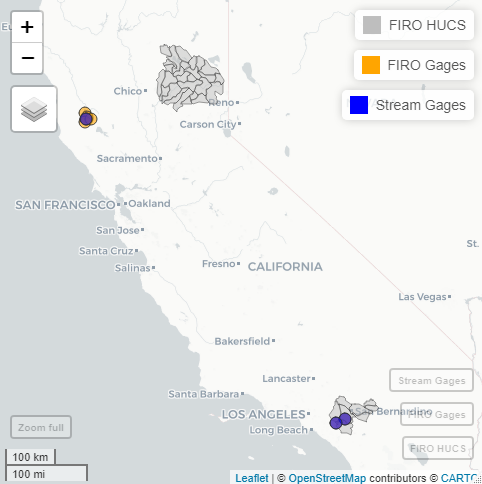
 

HUC12

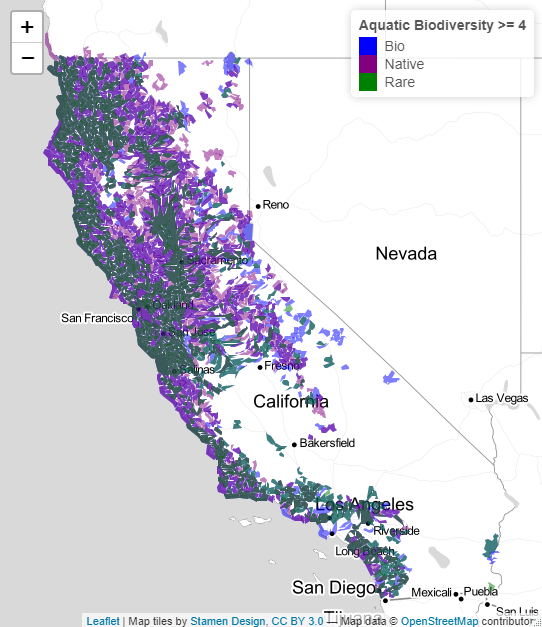
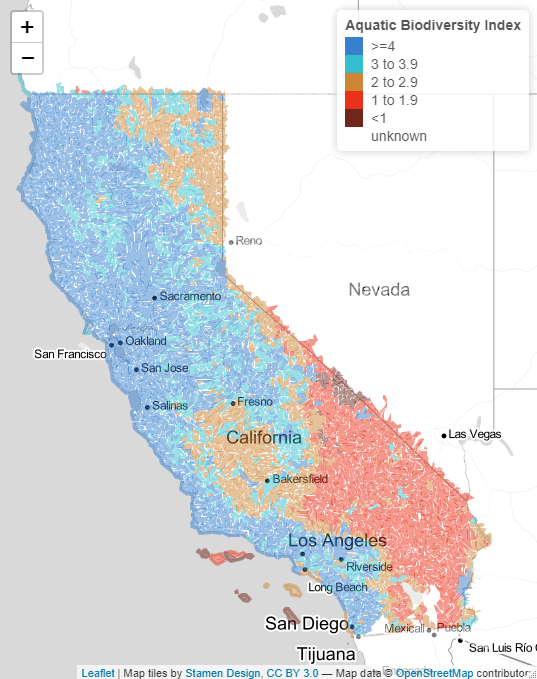
 

FIRO Gages –and HUCs (note no gages highlighted in the HUCS to the right… not in FIRO gage file.

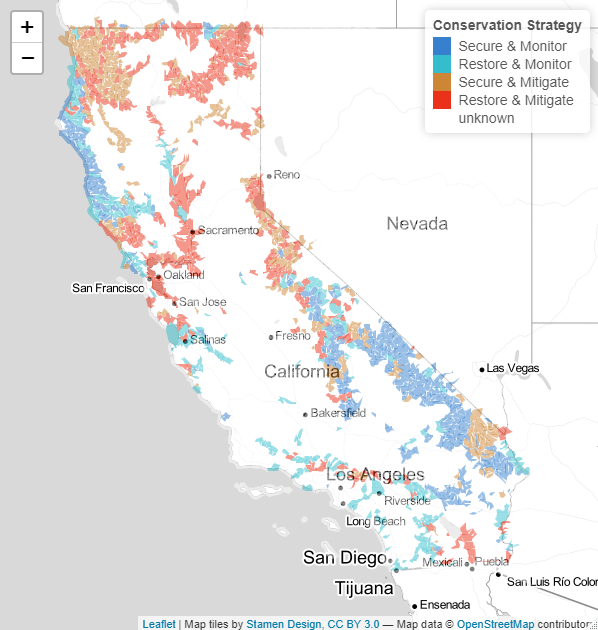
Also assumed those gages with FIRO should be classified as floodmgmt and active (all 3 were NA in the original shapefile).



Aquatic Biodiversity – selecting HUCs with the ranking for bio, native, and rare >= 4 (left) and the bio diversity index on right

Conservation Status



The wetlands and natural vegetation files were too large to do anything. I could dissolve the fields but all efforts to simplify broke arcgis, R, and mapshaper.html. California will have to hand a simpler file or calculate percent wetlands in each HUC10 and HUC12. The files as provided would be too large to load on a dashboard.

# Metadata on Gages

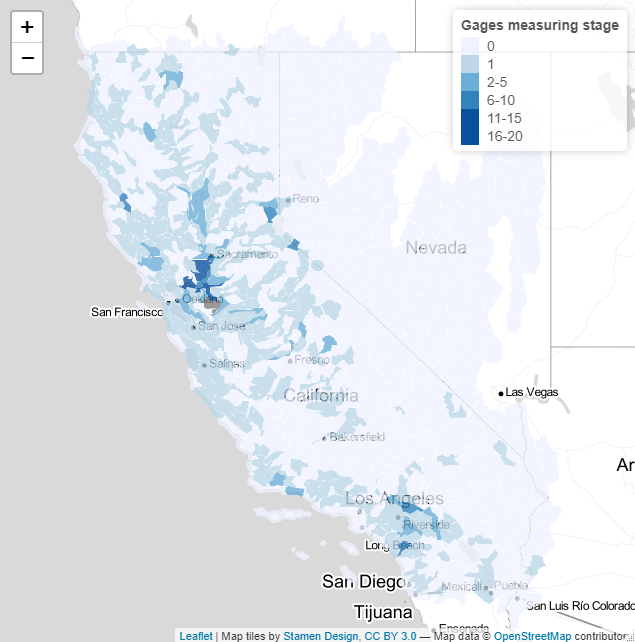
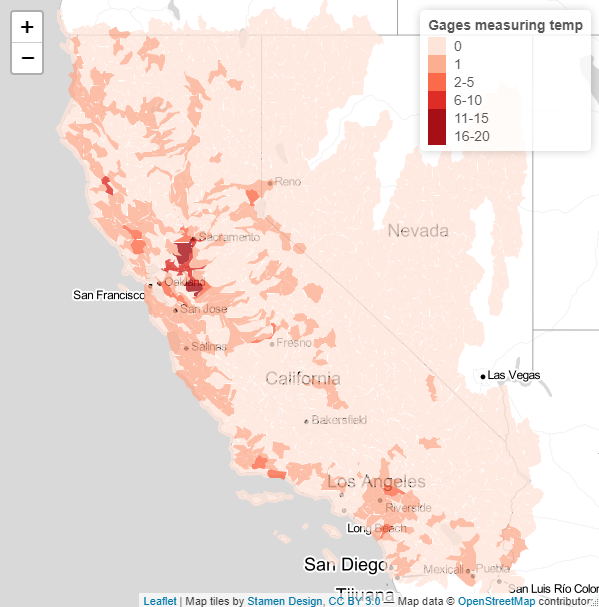
Clean up operator column

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Includes** | **Active** | **Inactive** |
| California | CA, DWR | 197 | 16 |
| City or County | City, County | 12 | 1 |
| Conservation Districts | Conservation, Consv | 2 | 0 |
| Energy Utilities | Pacific Gas, Power | 31 | 0 |
| Federal (non USGS) | US, Reclamation | 41 | 0 |
| Irrigation Districts | Irrigation, ID | 11 | 0 |
| Other | OTHER, unknown | 109 | 16 |
| USGS | USGS | 670 | 1407 |
| Water Utilities | Utility, Authority, Water | 23 | 0 |

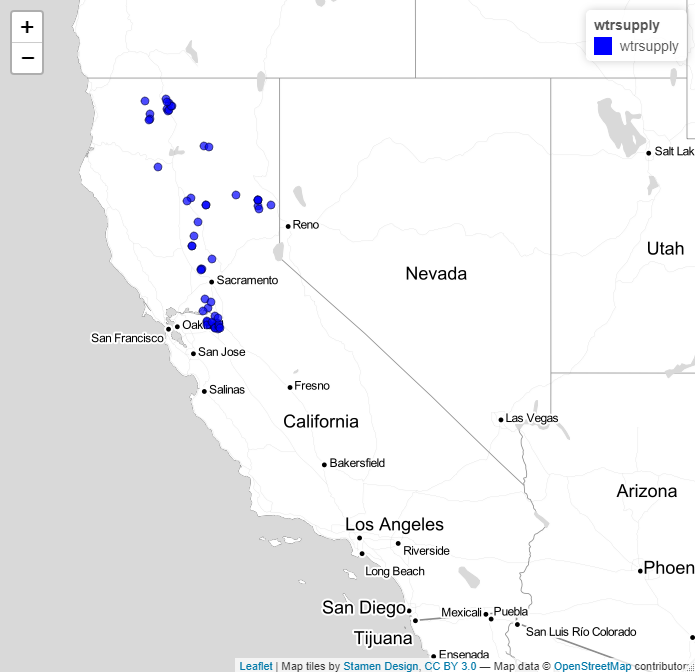
Operator by Stream Type – Active only

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Operator** | **Artificial Path** | **Canal/Ditch** | **Coastline** | **Connector** | **Pipeline** | **Stream - intermittent** | **Stream - Perennial** |
| California | 54 | 27 | 2 | 3 | 1 | 23 | 87 |
| City or County | 2 | 0 | 0 | 0 | 0 | 0 | 10 |
| Conservation Districts | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Energy Utilities | 1 | 1 | 0 | 1 | 0 | 0 | 28 |
| Federal (non USGS) | 15 | 1 | 0 | 1 | 0 | 6 | 18 |
| Irrigation Districts | 0 | 1 | 0 | 0 | 0 | 2 | 8 |
| Other | 5 | 3 | 0 | 0 | 1 | 25 | 75 |
| USGS | 88 | 41 | 0 | 4 | 6 | 142 | 385 |
| Water Utilities | 1 | 3 | 0 | 0 | 1 | 7 | 10 |

We can calculate the number of gages measuring different things and highlight HUC accordingly



Make circle size based on the number of purposes a gage is fulfilling and highlight certain purposes



\*Note that 83 sites have not been assigned to any purpose

When select a gage we can create a table showing it’s purpose… or when a huc is selected it grabs all the gages

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Site** | **Flood** | **Ecosystem** | **Groundwater** | **Public Safety** | **Water Supply** |
| ANH | Yes | No | No | Yes | No |

We can say how much data has been collected for an individual gage

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute | Stage | Flow | Water Quality | Temperature |
| Collected (Y/N)? | Yes | No | Yes | Yes |
| Days of Data | 13,514 | 0 | 8,752 | 9,496 |
| Years of Data | 37 | 0 | 24 | 26 |
| Real Time (Y/N)? | Yes | No | Yes | Yes |

# Using NLDI to grab gages upstream and downstream of segments



# Notes

* Geojson files are still large even after simplification. This makes it very slow to load in leaflet (app.R). About 1 minute.
* Mapboxer is a new R package that uses mapboxGL – this greatly increases the load time but still take 5 to 10 seconds on each load and 2-3 seconds for each update. Reading in mapbox vector tilesets does not appear to be an option yet.
* Fastest speeds and best performance are in javascript using mapbox tilesets.