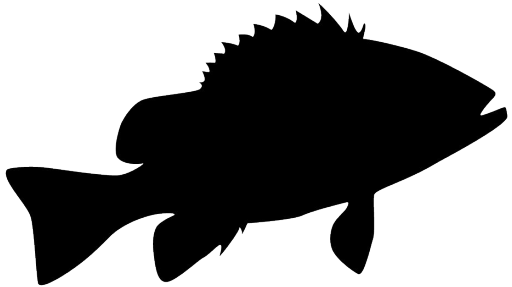
SHINY SHOW & TELL

fisheye



Ashley Vizek (she/her) | ECS in support of NWFSC

NOAA FISHERIES
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

About Literature Resources Contact us

fisheye

Fisheries Economic Explorer

An interactive tool for exploring data from the West Coast Groundfish Trawl Catch Share Program. Data summarized comes from the Economic Data Collection Program and PacFIN.

Choose an application to get started

Performance Metrics

Whiting Purchase & Production

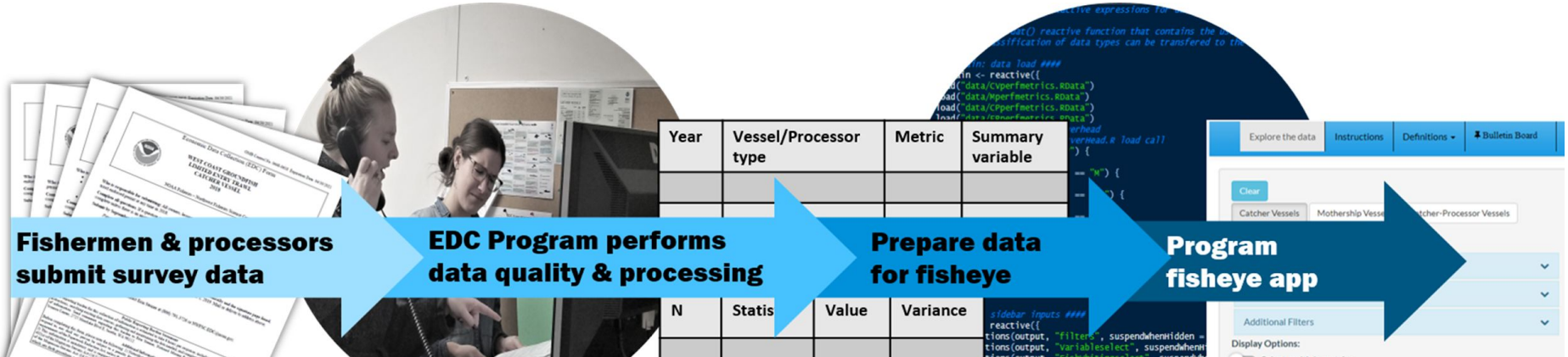
Landings Tracker

Why Shiny?

Primary goal of fisheye is to report EDC data while protecting confidentiality

Different use cases and user groups that request specific sectors or aggregations; we wanted to group by everything and filter by everything

Shiny provides an interactive approach to reporting data



Clear

Catcher Vessels | Mothership Vessels | Catcher-Processor Vessels | First Receivers and Shorebased Processors

Metric

Filter by: fisheries, location, size

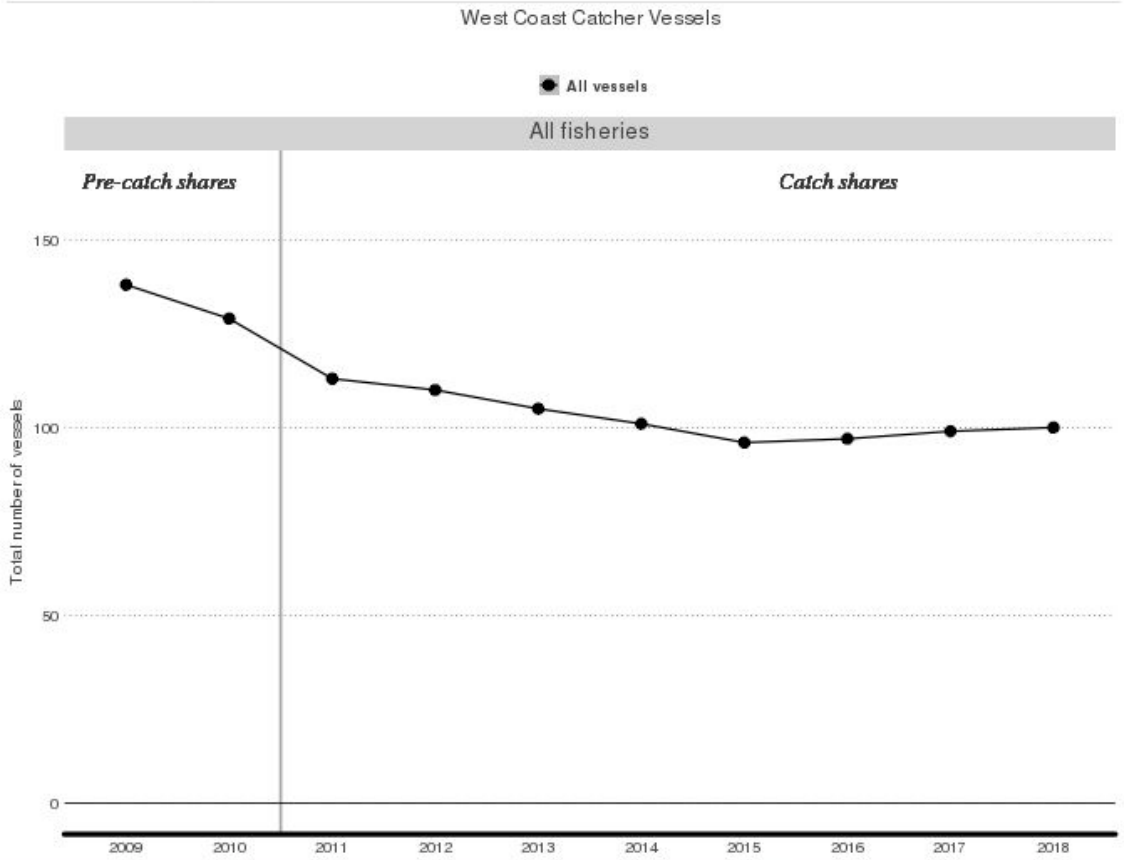
Additional Filters

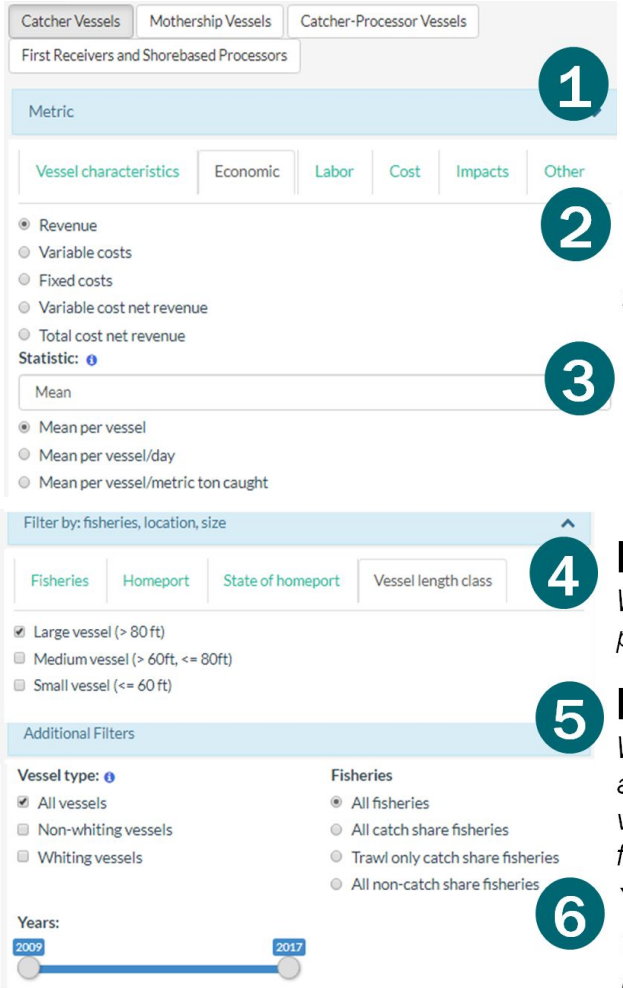
Display Options:

Select multiple metrics

Download Plot(s) | Download Data | Bookmark...

Visualize the Data | Dataset





1 **Sector**
Which type of operation would you like to explore?

2 **Metric**
Which measure would you like to display?

3 **Statistic**
How would you like the information summarized?

4 **Participant category**
Which category of vessels or processors would you like to see?

5 **Filter**
Would you like to compare whiting and non-whiting vessels/processors? Which set of fisheries would you like to explore?

6 **Year**
Would you like to change the range of years displayed?

Sidebar (& dataframe) structure

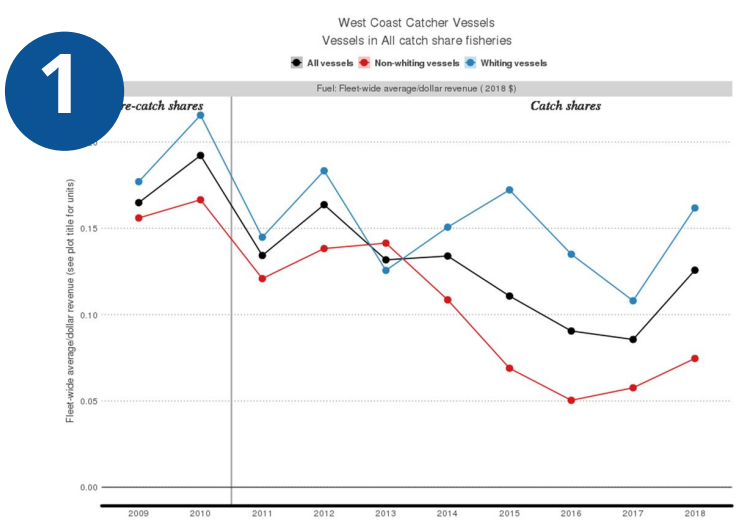
Fisheye tour

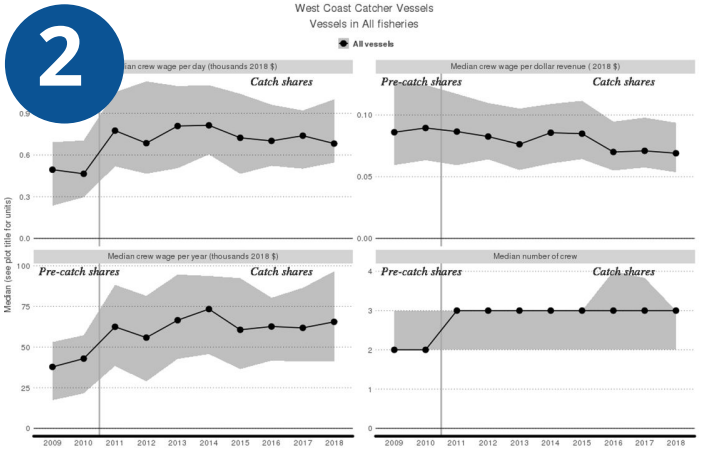
(1) How do fuel costs differ for whiting and non-whiting vessels in catch share fisheries?

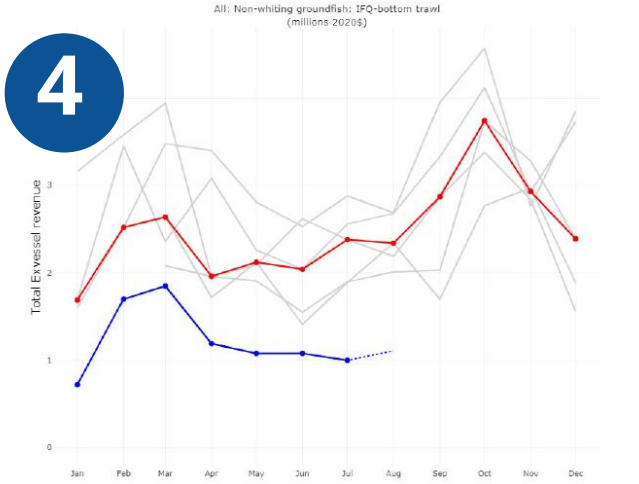
(2) How has crew wage changed over time for vessels in catch share fisheries?

(3) How does whiting production value differ by sector?

(4) How much revenue did the IFQ groundfish fishery generate in July this year?







Challenges

We want to group by everything and filter by everything…

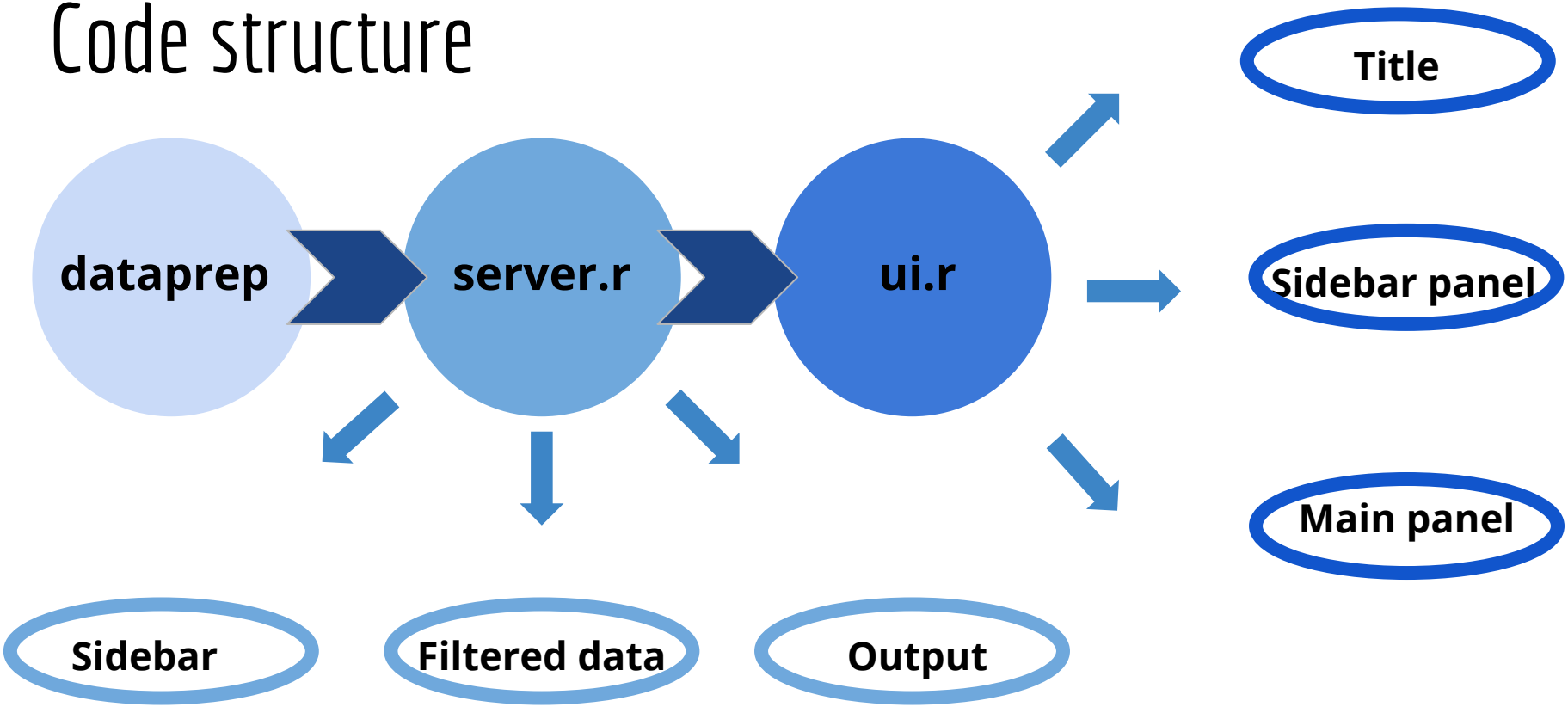
It is more difficult to program the **SIDEBAR** with each additional filter, especially when unique cases are added

The more groups that are added, the larger the dataframe; and eventually **APP PERFORMANCE** is impacted

Designing/organizing the **USER INTERFACE** effectively was a challenge; especially with multiple layers of filters and groups

Confidentiality, confidentiality, confidentiality

Code structure



Basic code examples

Sidebar

```
output$sectorInput <- renderUI({
  checkboxGroupInput("sectorInput","Sector",
                     choices = unique(sumTab$Sector),
                     selected = c("Catcher-Processor",
                                  "Mothership",
                                  "Shoreside"))
})
```

Basic code examples

Filtered data

```
filtered <- reactive({
  if(input$tab_type == "Summary") {
      data %>%
        filter(Metric %in% c(input$yaxisInput),
               Statistic == input$statInput,
               Sector %in% c(input$sectorInput))
  } else {
    data %>%
      filter(Metric %in% c(input$yaxis2Input),
             Statistic == input$stat2Input,
             Sector %in% c(input$sector2Input))
  }
})
```

Basic code examples

Output

```
output$productplot <- renderPlot({
  if(is.null(filtered())) {
    return()
  }
    ggplot(filtered(),
           aes(x = Year,
               y = Value,
               group = Sector)) +
      scale_fill_manual(values = lineColor) +
      scale_color_manual(values = lineColor) +
      theme_minimal() +
      theme(text = element_text(size = 14),
            axis.text = element_text(size = 12),
            strip.text = element_text(size = 14)) +
      geom_point(aes(color = Sector), size = point_size) +
      geom_line(aes(color = Sector), size = line_size) +
      geom_ribbon(aes(ymax = upper,
                      ymin = lower,
                      fill = Sector), alpha = .25) +
      facet_wrap(~ylab, scales = 'free_y', ncol = 2) +
      labs(y = input$stat2Input) +
      scale_x_continuous(breaks= pretty_breaks())
}, height = 800, width = 1100)
```

Basic code examples

```
navbarPage(id = "page", collapsible = T, inverse = F,
           title = "",
           tabPanel("Explore the data", value = "results",
                    sidebarLayout(
                      sidebarPanel(uiOutput("tabs"),
                                   uiOutput("download_Table")),
                      mainPanel(
                        tabsetPanel(type = "tabs",
                                    tabPanel("Plot",
                                             conditionalPanel(
                                               condition = "input.tab_type ==  'Summary'",
                                               plotOutput("yearplot")),
                                             conditionalPanel(
                                               condition = "input.tab_type == 'By product type'",
                                               plotOutput("productplot")), style = "min-height: 1000px"),
                                    tabPanel("Table", DT::dataTableOutput("table"))
                                    )
                      )
                    )),
```

Title

Sidebar panel

Main panel

Notes about server deployment

- Open-source shiny server
  - Previously used Shiny Server Pro though found we didn't utilize/require the services provided to warrant cost.
  - Support from NWFSC IT

- We use PuTTY and Command Line or R Terminal to manage and transfer files.

Let’s talk about Shiny!