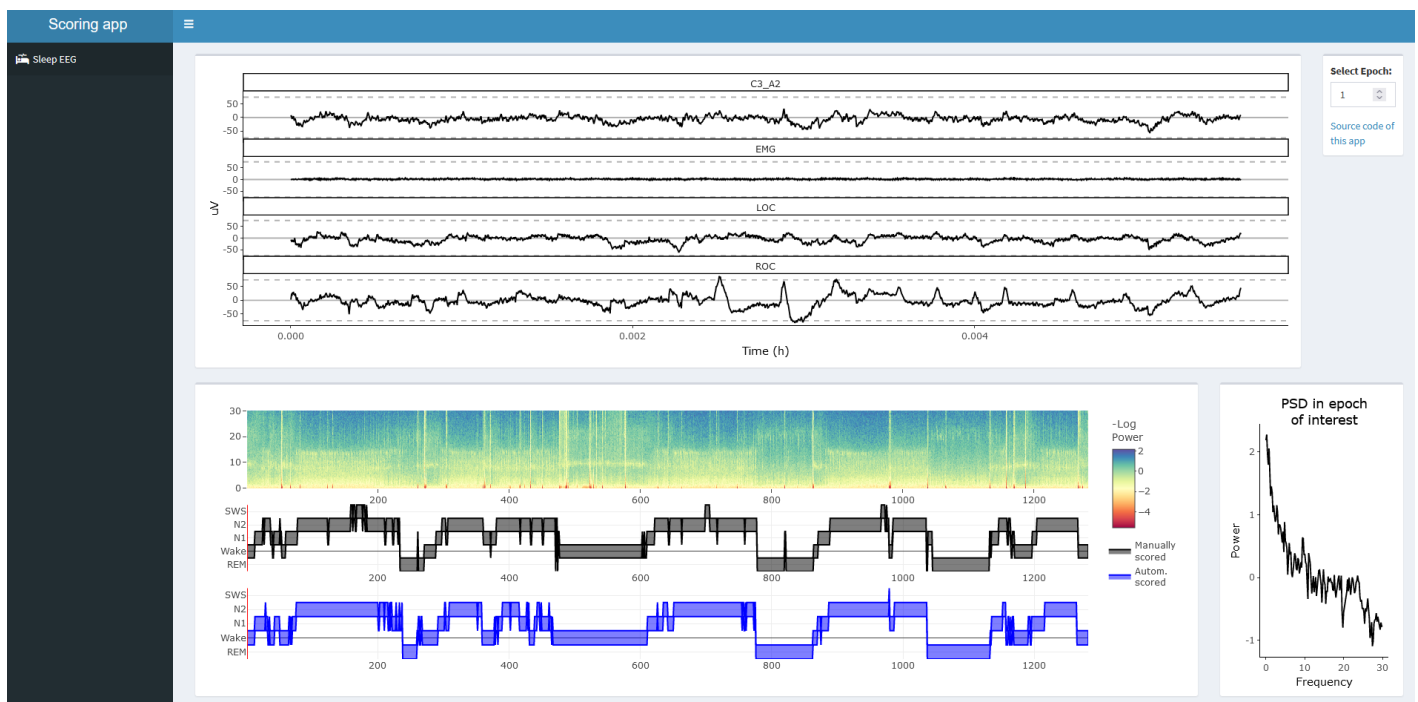


Automatic sleep stage scoring shiny app

Source code of the shiny app at

https://shiny.analysed.ch/scoring_app/



```
library(shiny)
library(shinydashboard)
library(plotly)
library(dplyr)

# Load spectra
spectra <- vroom::vroom('spectra.txt')

# stages
stages <- vroom::vroom('stages.txt')

# signal
signal <- vroom::vroom('signal.txt')
```

```

ui <- dashboardPage(
  dashboardHeader(title = ' Scoring app' ),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Sleep EEG", tabName = "spe_tra", icon = icon("procedures"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem("spe_tra",
        box(plotlyOutput("eeg_plot"), width = 11),
        box(numericInput("Epoch_sl", "Select Epoch:", 1, min = 1, max = 1282), width =
1,
          tags$a(href="https://blog.analysed.ch/books/analysedch-apps/page/automatic-
sleep-stage-scoring-shiny-app", "Source code of this app")),
        box(plotlyOutput("plot_old"), width = 10),
        box(plotlyOutput("plot_spec"), width = 2)
      )
    )
  )
)

server <- function(input, output){

  output$eeg_plot <- renderPlotly({
    ggplotly(ggplot(signal %>% dplyr::filter(Epoch == input$Epoch_sl), mapping = aes(x =
time, Amplitude)) +
      geom_line()+ facet_wrap(vars(Channel), nrow = 4)+ geom_hline(yintercept = 0,
alpha = 0.3)+
      geom_hline(yintercept = c(-75,75), lty = 7, alpha = 0.3)+
      labs(x = 'Time (h)', y = 'uV' ) +
      scale_colour_viridis_d(begin = 0.1, end = 0.7) + theme_classic())
  })

  output$plot_old <- renderPlotly({

    fig <- plot_ly(spectra, x = ~Epoch, y = ~Freq, z = ~Power, colors = "Spectral")
%>%
    add_heatmap() %>% colorbar(title = "-Log\nPower")

    fig1 <- plot_ly(stages, x = ~Epoch, name = 'Manually\nscored') %>%
      add_ribbons(ymin = ~man_scored - 0.5, ymax = ~man_scored + 0.5, color= I("black"))
  })
}

```

```

%>%
  add_segments(x = input$Epoch_sl, xend = input$Epoch_sl, y = -1.5, yend = 3.5,
showlegend = FALSE, color=I('red')) %>%
  plotly::layout(
    yaxis = list(
      range=c(-1.5,4),
      ticktext = list("REM", "Wake", "N1", "N2", "SWS"),
      tickvals = list(-1, 0, 1, 2, 3),
      tickmode = "array"
    )
  )

fig2 <- plot_ly(stages, x = ~Epoch, name = 'Autom.\nscored') %>%
  add_ribbons(ymin = ~Stages - 0.5, ymax = ~Stages + 0.5, color = I("blue")) %>%
  add_segments(x = input$Epoch_sl, xend = input$Epoch_sl, y = -1.5, yend = 3.5,
showlegend = FALSE, color=I('red')) %>%
  plotly::layout(
    yaxis = list(
      range=c(-1.5,4),
      ticktext = list("REM", "Wake", "N1", "N2", "SWS"),
      tickvals = list(-1, 0, 1, 2, 3),
      tickmode = "array"
    )
  )
  subplot(fig, fig1, fig2 , nrows = 3)
})

output$plot_spec <- renderPlotly({
  ggplotly(ggplot(spectra %>% dplyr::filter(Epoch == input$Epoch_sl), mapping = aes(x =
Freq, y=- Power)) +
    geom_line() + labs(x = 'Frequency', y = 'Power', title=' PSD in epoch\nof
interest') +
    scale_colour_viridis_d(begin = 0.1, end = 0.7) + theme_classic() +
theme(plot.title = element_text(hjust = 0.5)))
})
}

shinyApp(ui, server)

```