# Drifter data analysis

Supplementary material for 'Efficient Curve Fitting with Penalized B-Splines for Oceanographic and Ecological Applications'

Dong-Young Lee and Jae-Hwan Jhong

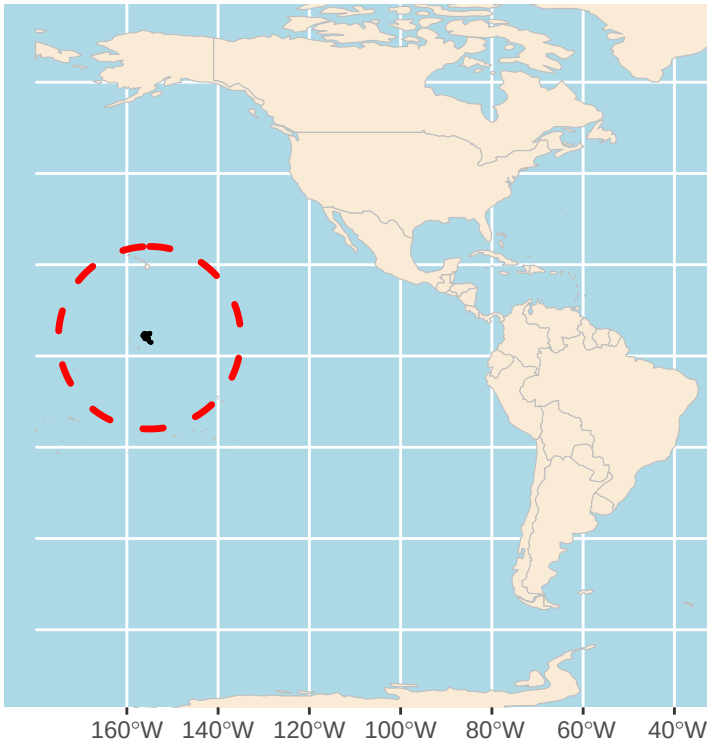
2025-05-29

```
rm(list = ls())

# install.packages(c('ggforce', 'cowplot', 'sphereplot',
# 'rworldmap', 'ggplot2', 'sf', 'ncdf4', 'dplyr', 'raster',
# 'gridExtra', 'genlasso', 'mgcv'))

library(ggforce)
library(cowplot)
library(sphereplot)
library(rworldmap)
library(ggplot2)
library(sf)
library(ncdf4)
library(ggplot2)
library(dplyr)
library(raster)
library(gridExtra)
library(stats)
library(genlasso)
library(mgcv)

source("admm_source.R")
source("splineBox.R")
```

```
#==============================================================================
# Data & Fitting (Integrated)
#==============================================================================
test1 = read.csv("Data/drifter.csv")
t1 = subset(test1,test1$WMO == 5102764)
t2 = subset(test1,test1$WMO == 5102765)
t3 = subset(test1,test1$WMO == 5102766)

n = nrow(t2)
order = 3
dimension = 40
t = seq(0, 1, length = n)

t1 = cbind(t1,t[1:nrow(t1)])
t2 = cbind(t2,t[1:nrow(t2)])
t3 = cbind(t3,t[1:nrow(t3)])
```

```r
t1 = t1[, -which(names(t1) == "WMO")]
t2 = t2[, -which(names(t2) == "WMO")]
t3 = t3[, -which(names(t3) == "WMO")]
names(t2) = names(t1)
names(t3) = names(t1)


t123 = rbind(t1,t2,t3)
colnames(t123)[3] = 't'
t123_sorted = t123[order(t123$t), ]

y = as.matrix(sapply(t123_sorted[, c("longitude", "latitude")], as.numeric))

n = nrow(t123_sorted)
t = t123_sorted$t

#============================================================================

knots = knots_quantile(t, dimension, order)
B = bsplines(t, knots, order)
D = bspline_jump(knots, order)
fit = bspline.curve.admm_lambdas(y, D, B,
                                 lambdas = NULL,
                                 lam_max = 100,
                                 lam_min = 1e-10,
                                 n_lambda = 200,
                                 max_iter = 1000,
                                 epsilon = 1e-8,
                                 eta_c = 1)

worldMap = getMap()
worldMap_sf = st_as_sf(worldMap)
mar = 0.2


y_df = as.data.frame(y)
y_df_sf = st_as_sf(y_df, coords = c("longitude", "latitude"), crs = 4326)

t_new = seq(0, 1, length = 5000)
B_new = bsplines(t_new, knots, order)
# best aic
fit_new = B_new %*% fit[[which.min(fit$aic)]]$xi
fit_new_df = as.data.frame(fit_new)
fit_new_df_sf = st_as_sf(fit_new_df, coords = c("longitude", "latitude"), crs = 4326)
# best bic
fit_new2 = B_new %*% fit[[which.min(fit$bic)]]$xi
fit_new_df2 = as.data.frame(fit_new2)
fit_new_df_sf2 = st_as_sf(fit_new_df2, coords = c("longitude", "latitude"), crs = 4326)

#============================================================================
# [Plot1(1,1)] World map
#============================================================================
nino4_center = c(-155, 4)  # Niño 4 region center (longitude, latitude)
```

```r
nino4_radius = 20          # Radius (unit is latitude/longitude)

worldmap = ggplot() +
  geom_sf(data = worldMap_sf, color = "grey", fill = "antiquewhite") +
  geom_sf(data = y_df_sf, size = 0.1) +
  geom_circle(
    aes(x0 = nino4_center[1], y0 = nino4_center[2], r = nino4_radius),
    inherit.aes = FALSE,  # Operates independently of existing data
    color = "red",        # Circle border color
    linetype = "dashed",  # Circle style
    # size = 1.2            # Circle thickness
    linewidth = 1.2
  ) +
  coord_sf(
    xlim = c(-180, -40),  # Longitude range
    ylim = c(-70, 70)     # Latitude range
  ) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA)) # Set background color
worldmap
```



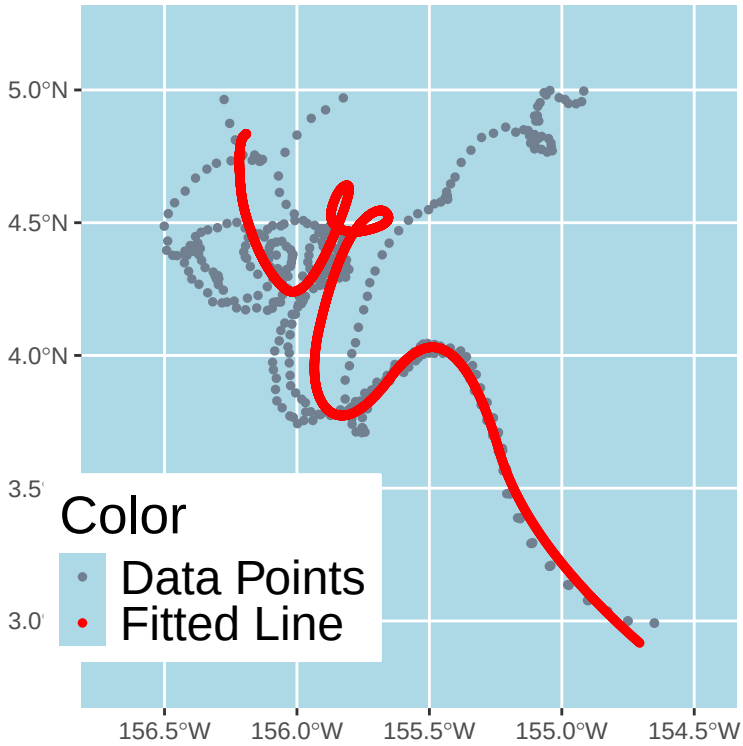

```r
#================================================================================
# [Plot3(1,1)] best aic
#================================================================================
zoommap1 = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y_df_sf, aes(color = "Data Points"), size = 1) +
  geom_sf(data = fit_new_df_sf, aes(color = "Fitted Line"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
```

```
                  max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2),
        legend.background = element_rect(fill = "white", color = "white"),
        legend.title = element_text(size = 20),
        legend.text = element_text(size = 18)) +
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
  scale_color_manual(values = c("Data Points" = "slategrey", "Fitted Line" = "red")) +
  labs(fill = "Map Fill",color = 'Color') # Modify legend title
```

```
## Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2
## 3.5.0.
## i Please use the `legend.position.inside` argument of `theme()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
zoommap1
```

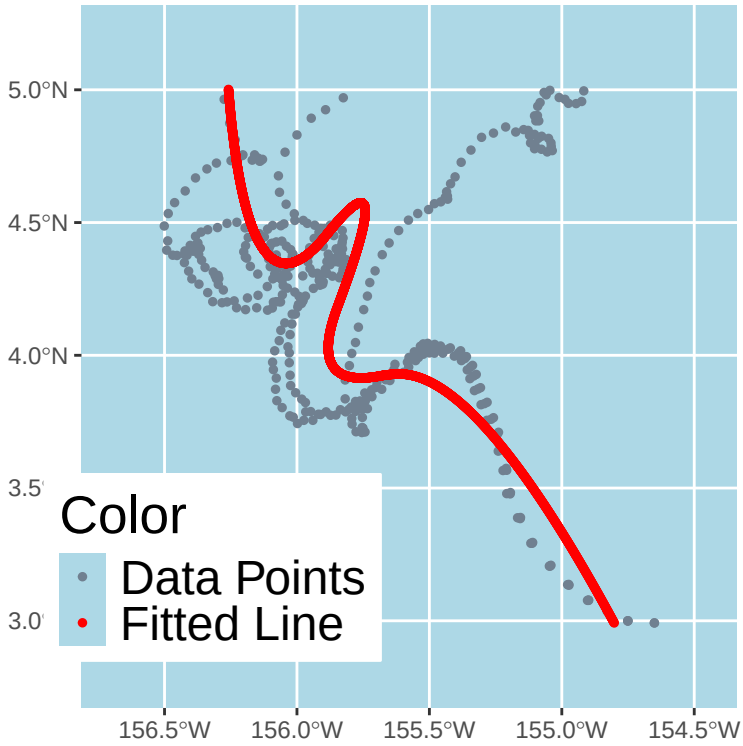
```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```




```
#=============================================================================
# [Plot3(1,2)] best bic
#=============================================================================
zoommap2 = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y_df_sf, aes(color = "Data Points"), size = 1) +
  geom_sf(data = fit_new_df_sf2, aes(color = "Fitted Line"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
```

```
          ylim = c(min(y_df$latitude) - mar,
                   max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2), # Move legend to the bottom right inside the plot
        legend.background = element_rect(fill = "white", color = "white"), # Set legend background colo
        legend.title = element_text(size = 20), # Legend title size
        legend.text = element_text(size = 18)) + # Legend text size
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
  scale_color_manual(values = c("Data Points" = "slategrey", "Fitted Line" = "red")) +
  labs(fill = "Legend", color = "Color") # Set legend title
zoommap2
```

```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```



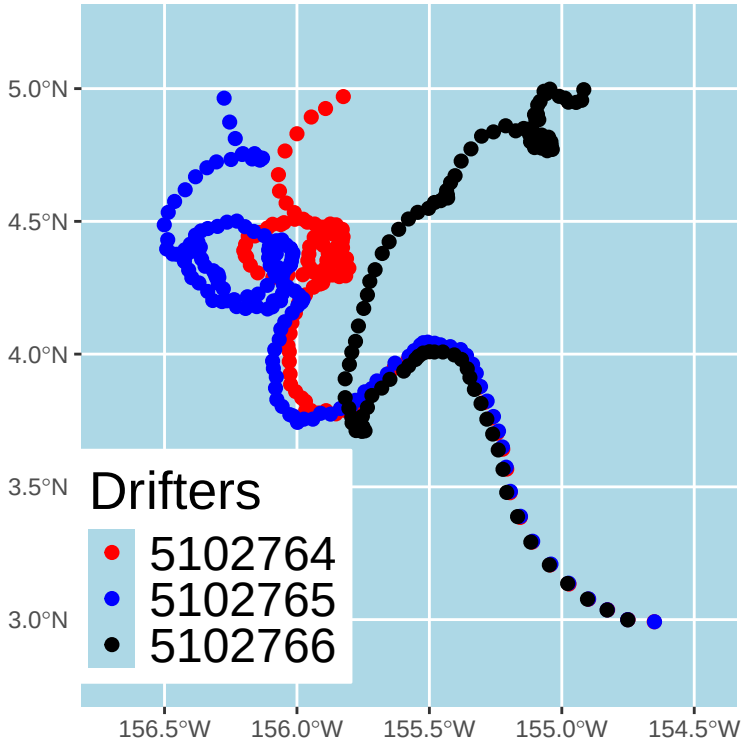

```
#=============================================================================
# Data & Fitting (Individual)
#=============================================================================
b1 = subset(test1,test1$WMO == 5102764)
b2 = subset(test1,test1$WMO == 5102765)
b3 = subset(test1,test1$WMO == 5102766)


n1 = nrow(b1)
n2 = nrow(b2)
n3 = nrow(b3)

time1 = seq(0, 1, length = n1)
time2 = seq(0, 1, length = n2)
time3 = seq(0, 1, length = n3)
```

```
b1 = cbind(b1,time1)
b2 = cbind(b2,time2)
b3 = cbind(b3,time3)
b1 = b1[, -which(names(b1) == "WMO")]
b2 = b2[, -which(names(b2) == "WMO")]
b3 = b3[, -which(names(b3) == "WMO")]


y1 = as.matrix(sapply(b1[,c("longitude", "latitude")], as.numeric))
y2 = as.matrix(sapply(b2[,c("longitude", "latitude")], as.numeric))
y3 = as.matrix(sapply(b3[,c("longitude", "latitude")], as.numeric))

order = 3
dimension = 10

knots1 = knots_quantile(time1, dimension, order)
B1 = bsplines(time1, knots1, order)
D1 = bspline_jump(knots1, order)
fit1 = bspline.curve.admm_lambdas(y1, D1, B1,
                                  lambdas = NULL,
                                  lam_max = 100,
                                  lam_min = 1e-10,
                                  n_lambda = 200,
                                  max_iter = 1000,
                                  epsilon = 1e-8,
                                  eta_c = 1)
knots2 = knots_quantile(time2, dimension, order)
B2 = bsplines(time2, knots2, order)
D2 = bspline_jump(knots2, order)
fit2 = bspline.curve.admm_lambdas(y2, D2, B2,
                                  lambdas = NULL,
                                  lam_max = 100,
                                  lam_min = 1e-10,
                                  n_lambda = 200,
                                  max_iter = 1000,
                                  epsilon = 1e-8,
                                  eta_c = 1)
knots3 = knots_quantile(time3, dimension, order)
B3 = bsplines(time3, knots3, order)
D3 = bspline_jump(knots3, order)
fit3 = bspline.curve.admm_lambdas(y3, D3, B3,
                                  lambdas = NULL,
                                  lam_max = 100,
                                  lam_min = 1e-10,
                                  n_lambda = 200,
                                  max_iter = 1000,
                                  epsilon = 1e-8,
                                  eta_c = 1)

worldMap = getMap()
worldMap_sf = st_as_sf(worldMap)

y1_df = as.data.frame(y1)
```

```r
y2_df = as.data.frame(y2)
y3_df = as.data.frame(y3)
y1_df_sf = st_as_sf(y1_df, coords = c("longitude", "latitude"), crs = 4326)
y2_df_sf = st_as_sf(y2_df, coords = c("longitude", "latitude"), crs = 4326)
y3_df_sf = st_as_sf(y3_df, coords = c("longitude", "latitude"), crs = 4326)

t_new = seq(0, 1, length = 5000)
B1_new = bsplines(t_new, knots1, order)
B2_new = bsplines(t_new, knots2, order)
B3_new = bsplines(t_new, knots3, order)

# best aic
fit1_new = B1_new %*% fit1[[which.min(fit1$aic)]]$xi
fit2_new = B2_new %*% fit2[[which.min(fit2$aic)]]$xi
fit3_new = B3_new %*% fit3[[which.min(fit3$aic)]]$xi
fit1_new_df = as.data.frame(fit1_new)
fit2_new_df = as.data.frame(fit2_new)
fit3_new_df = as.data.frame(fit3_new)
fit1_new_df_sf = st_as_sf(fit1_new_df, coords = c("longitude", "latitude"), crs = 4326)
fit2_new_df_sf = st_as_sf(fit2_new_df, coords = c("longitude", "latitude"), crs = 4326)
fit3_new_df_sf = st_as_sf(fit3_new_df, coords = c("longitude", "latitude"), crs = 4326)

# best bic (for reference)
fit1_new2 = B1_new %*% fit1[[which.min(fit1$bic)]]$xi
fit2_new2 = B2_new %*% fit2[[which.min(fit2$bic)]]$xi
fit3_new2 = B3_new %*% fit3[[which.min(fit3$bic)]]$xi
fit1_new_df2 = as.data.frame(fit1_new2)
fit2_new_df2 = as.data.frame(fit2_new2)
fit3_new_df2 = as.data.frame(fit3_new2)
fit1_new_df_sf2 = st_as_sf(fit1_new_df2, coords = c("longitude", "latitude"), crs = 4326)
fit2_new_df_sf2 = st_as_sf(fit2_new_df2, coords = c("longitude", "latitude"), crs = 4326)
fit3_new_df_sf2 = st_as_sf(fit3_new_df2, coords = c("longitude", "latitude"), crs = 4326)

#=============================================================================
# [Figure1(1,2)] basic point default
#=============================================================================
mar = 0.2

b123basic = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y1_df_sf, aes(color = "5102764"), size = 2) +
  geom_sf(data = y2_df_sf, aes(color = "5102765"), size = 2) +
  geom_sf(data = y3_df_sf, aes(color = "5102766"), size = 2) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
                    max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2), # Move legend to the bottom left inside the plot
        legend.background = element_rect(fill = "white", color = "white"), # Set legend background colo
        legend.title = element_text(size = 20), # Legend title size
        legend.text = element_text(size = 18)) + # Legend text size
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
```

```
  scale_color_manual(values = c("5102764" = "red",
                                "5102765" = "blue",
                                "5102766" = "black")) +
  labs(fill = "Map Fill", color = "Drifters") # Set legend title
b123basic
```

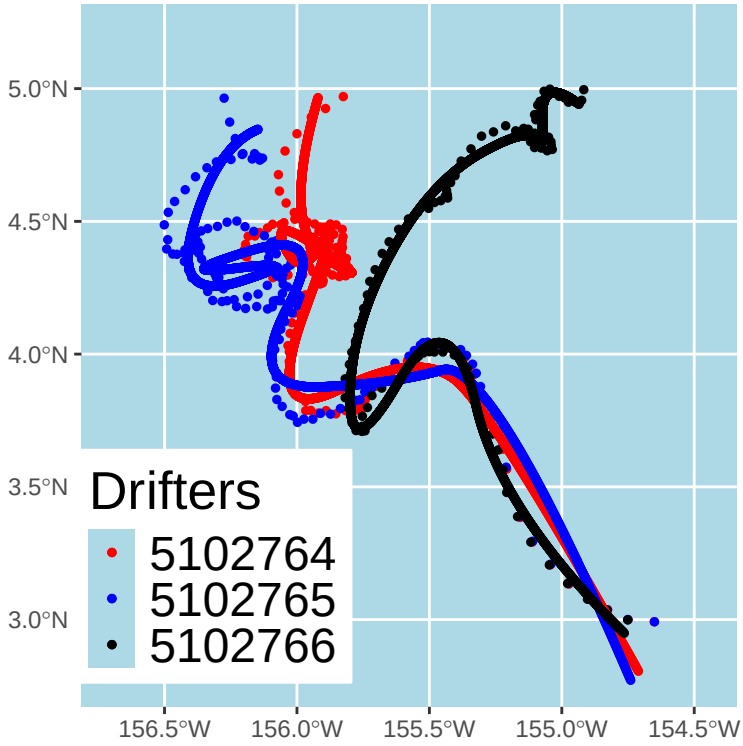
```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```




```
#=========================================================================
# [Figure 2(1,1) ~ 2(2,2)] Total 4 plots
# Plot b123 fitted lines together (AIC)
#=========================================================================
b123fitted = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y1_df_sf, aes(color = "5102764"), size = 1) +
  geom_sf(data = y2_df_sf, aes(color = "5102765"), size = 1) +
  geom_sf(data = y3_df_sf, aes(color = "5102766"), size = 1) +
  geom_sf(data = fit1_new_df_sf, aes(color = "5102764"), size = 1) +
  geom_sf(data = fit2_new_df_sf, aes(color = "5102765"), size = 1) +
  geom_sf(data = fit3_new_df_sf, aes(color = "5102766"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
                    max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2), # Move legend to the bottom right inside the plot
        legend.background = element_rect(fill = "white", color = "white"), # Set legend background colo
        legend.title = element_text(size = 20), # Legend title size
        legend.text = element_text(size = 18)) + # Legend text size
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
```

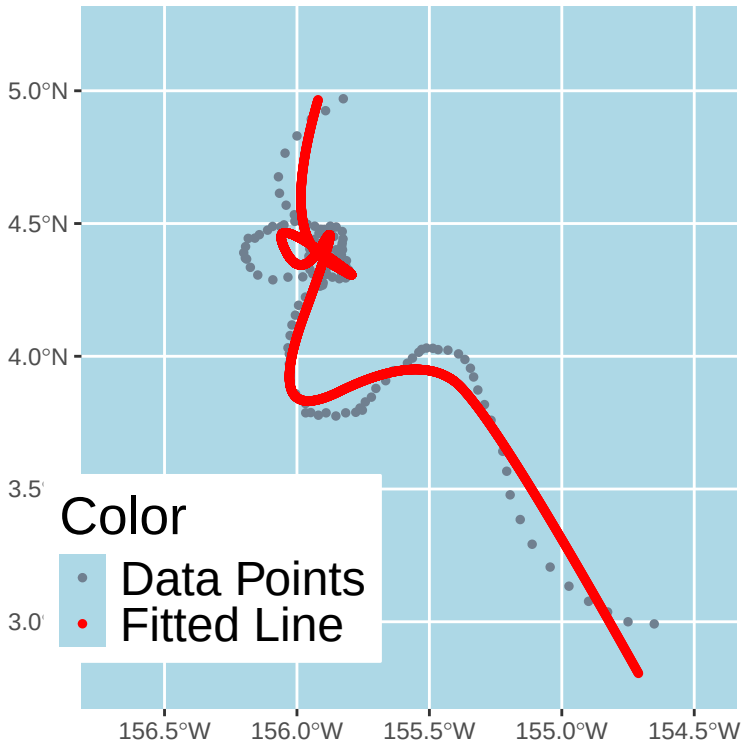
```
  scale_color_manual(values = c("5102764" = "red",
                                "5102765" = "blue",
                                "5102766" = "black")) +
  labs(color = "Drifters") # Set single legend title
b123fitted
```

```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```



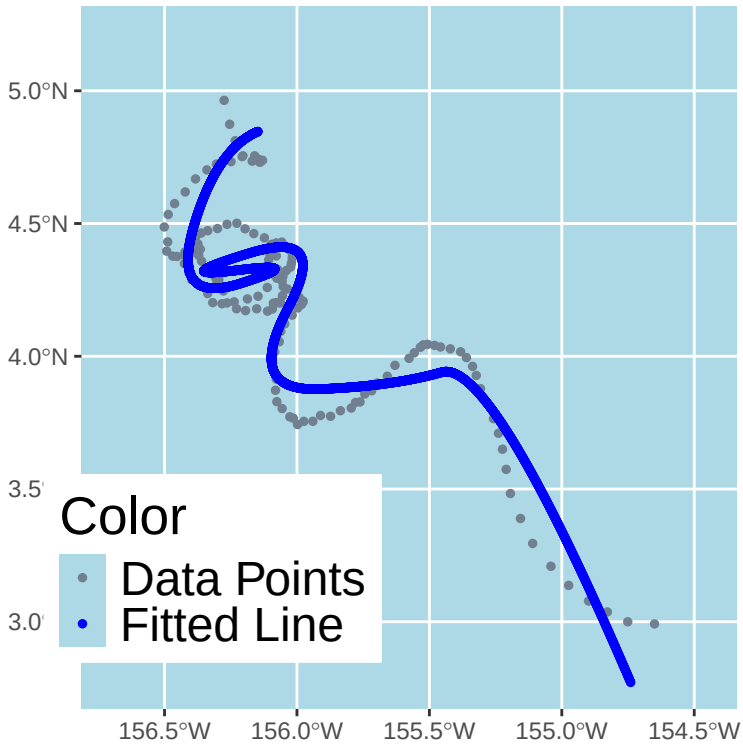

```
# b1 fitted line (AIC)
b1fitted = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y1_df_sf, aes(color = "Data Points"), size = 1) +
  geom_sf(data = fit1_new_df_sf, aes(color = "Fitted Line"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
                    max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2),
        legend.background = element_rect(fill = "white", color = "white"),
        legend.title = element_text(size = 20),
        legend.text = element_text(size = 18)) +
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
  scale_color_manual(values = c("Data Points" = "slategrey", "Fitted Line" = "red")) +
  labs(fill = "Map Fill", color = 'Color') # Set legend title
b1fitted
```

```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```



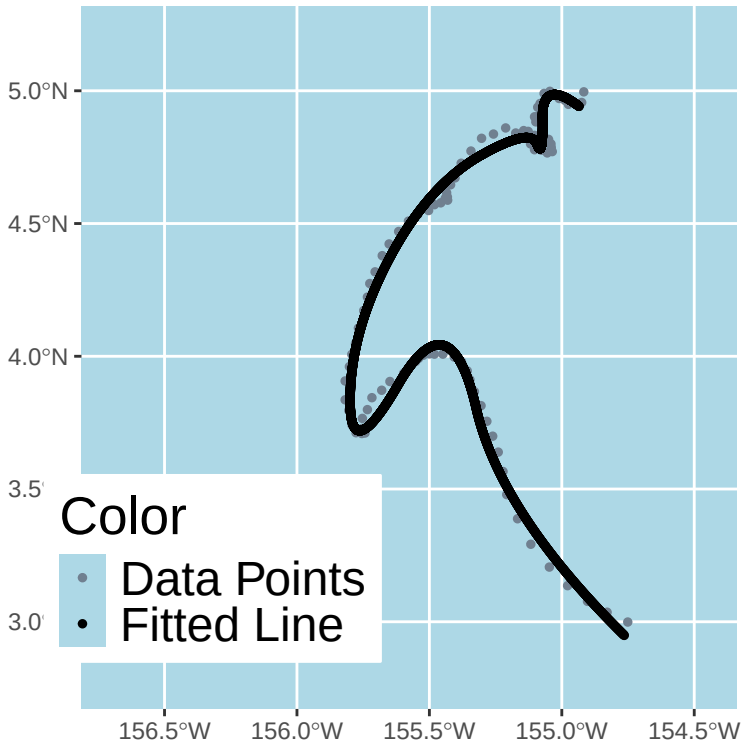

```
# b2 fitted line (AIC)
b2fitted = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y2_df_sf, aes(color = "Data Points"), size = 1) +
  geom_sf(data = fit2_new_df_sf, aes(color = "Fitted Line"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
                    max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2),
        legend.background = element_rect(fill = "white", color = "white"),
        legend.title = element_text(size = 20),
        legend.text = element_text(size = 18)) +
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
  scale_color_manual(values = c("Data Points" = "slategrey", "Fitted Line" = "blue")) +
  labs(fill = "Map Fill", color = 'Color') # Set legend title
b2fitted
```

```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```




```
# b3 fitted line (AIC)
b3fitted = ggplot() +
  geom_sf(data = worldMap_sf, aes(fill = "World Map"), color = "grey", fill = "antiquewhite") +
  geom_sf(data = y3_df_sf, aes(color = "Data Points"), size = 1) +
  geom_sf(data = fit3_new_df_sf, aes(color = "Fitted Line"), size = 1) +
  coord_sf(xlim = c(min(y_df$longitude) - mar,
                    max(y_df$longitude) + mar),
           ylim = c(min(y_df$latitude) - mar,
                    max(y_df$latitude) + mar)) +
  theme(panel.background = element_rect(fill = "lightblue", color = NA),
        legend.position = c(0.2, 0.2),
        legend.background = element_rect(fill = "white", color = "white"),
        legend.title = element_text(size = 20),
        legend.text = element_text(size = 18)) +
  scale_fill_manual(values = c("World Map" = "antiquewhite")) +
  scale_color_manual(values = c("Data Points" = "slategrey", "Fitted Line" = "black")) +
  labs(fill = "Map Fill", color = 'Color') # Set legend title
b3fitted
```

```
## Warning: No shared levels found between `names(values)` of the manual scale and the
## data's fill values.
```

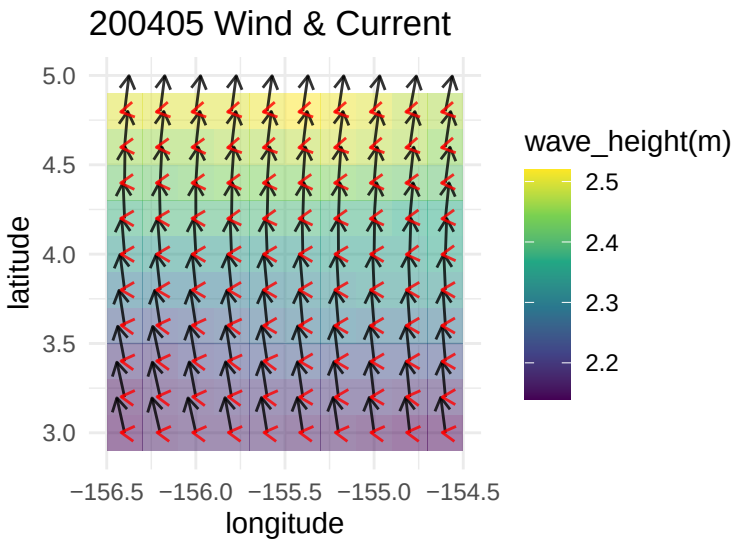



```
#==============================================================================
# Figure 2
#==============================================================================
# 200405 Wind & Current
#==============================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739711784083.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
wind_dir = ncvar_get(nc, "VMDR_WW")  # Wind Wave Direction
wave_dir = ncvar_get(nc, "VMDR")     # Mean Wave Direction
stokes_x = ncvar_get(nc, "VSDX")     # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")     # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0") # Significant Wave Height

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$wind_dir = as.vector(wind_dir[,,1])   # Wind direction (azimuth)
df$wave_dir = as.vector(wave_dir[,,1])   # Wave direction
df$stokes_x = as.vector(stokes_x[,,1])   # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])   # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1]) # Wave height

# Convert wind direction to vector (u, v)
df$wind_u = cos(df$wind_dir * pi / 180)  # x component
```

```r
df$wind_v = sin(df$wind_dir * pi / 180)  # y component

# Map visualization
p1 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = wave_height), alpha = 0.5) +  # Background color: wave height
  scale_fill_viridis_c(name = "wave_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * wind_u, yend = lat + 0.2 * wind_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "black", alpha = 0.8, size = 0.5) +  # Wind direction (black arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200405 Wind & Current", x = "longitude", y = "latitude") +
  theme_minimal()
```

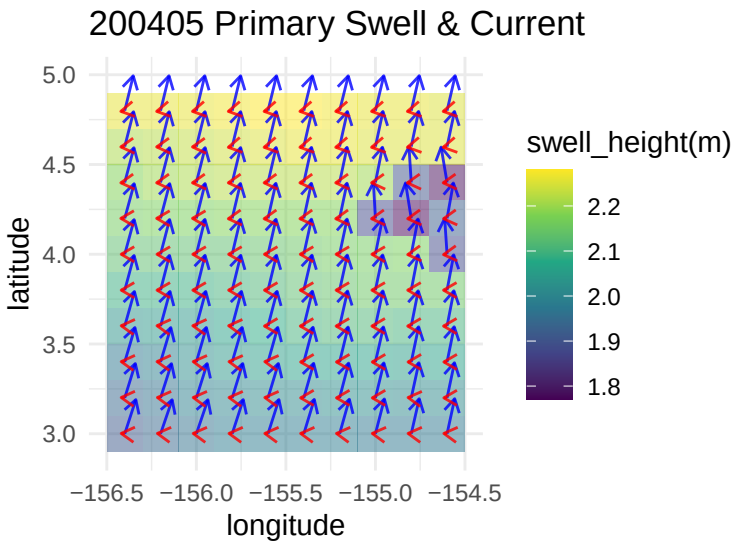
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
p1
```



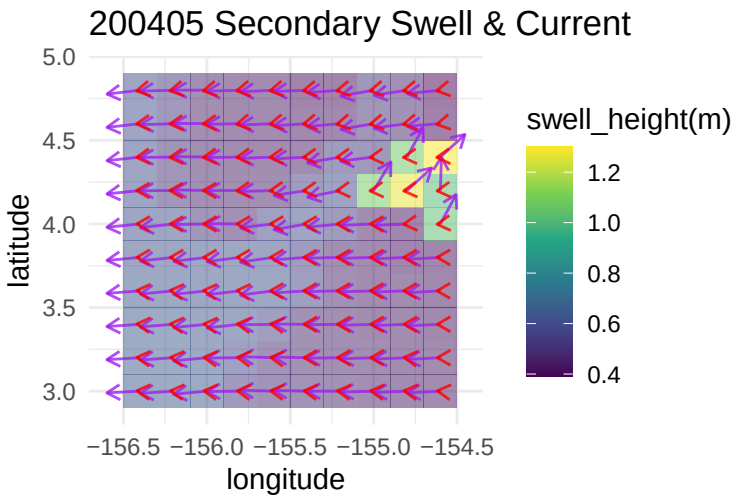

```r
#=============================================================================
# 200405 Primary Swell & Current
#=============================================================================
nc_file =  "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739713922090.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW1")     # Primary swell direction
swell_height = ncvar_get(nc, "VHM0_SW1") # Primary swell height
swell_period = ncvar_get(nc, "VTM01_SW1")# Primary swell period
stokes_x = ncvar_get(nc, "VSDX")          # Stokes Drift X (current)
```

```r
stokes_y = ncvar_get(nc, "VSDY")          # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")       # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])     # Primary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Primary swell height
df$swell_period = as.vector(swell_period[,,1]) # Primary swell period
df$stokes_x = as.vector(stokes_x[,,1])       # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])       # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p2 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: primary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "blue", alpha = 0.8, size = 0.5) +  # Primary swell direction (blue arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200405 Primary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p2
```



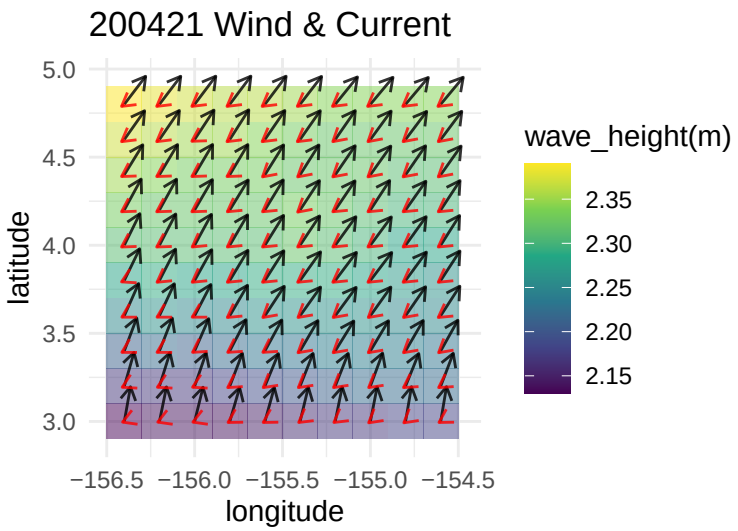

```r
#=========================================================================
# 200405 Secondary Swell & Current
```

```r
#==========================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739714468705.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW2")    # Secondary swell direction
swell_height = ncvar_get(nc, "VHM0_SW2") # Secondary swell height
swell_period = ncvar_get(nc, "VTM01_SW2")# Secondary swell period
stokes_x = ncvar_get(nc, "VSDX")         # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")         # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")      # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])    # Secondary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Secondary swell height
df$swell_period = as.vector(swell_period[,,1]) # Secondary swell period
df$stokes_x = as.vector(stokes_x[,,1])      # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])      # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p3 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: secondary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "purple", alpha = 0.8, size = 0.5) +  # Secondary swell direction (purple arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200405 Secondary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p3
```



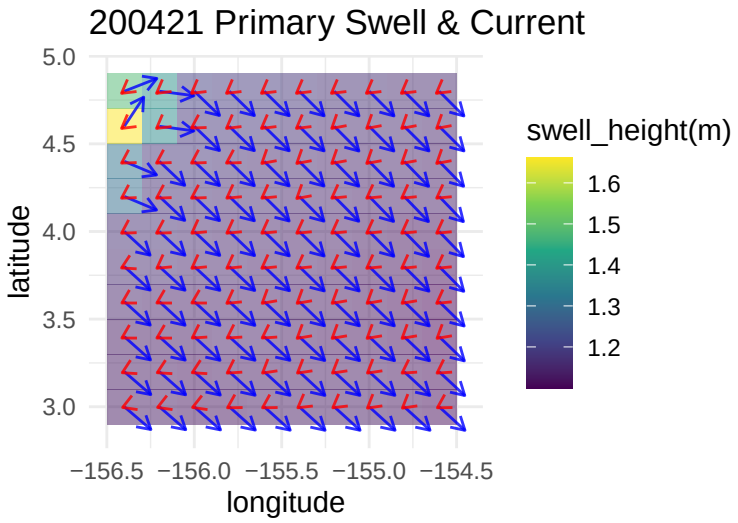

```
#================================================================================
# 200421 Wind & Current
#================================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739805996585.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
wind_dir = ncvar_get(nc, "VMDR_WW")  # Wind Wave Direction
wave_dir = ncvar_get(nc, "VMDR")     # Mean Wave Direction
stokes_x = ncvar_get(nc, "VSDX")     # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")     # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0") # Significant Wave Height

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$wind_dir = as.vector(wind_dir[,,1])   # Wind direction (azimuth)
df$wave_dir = as.vector(wave_dir[,,1])   # Wave direction
df$stokes_x = as.vector(stokes_x[,,1])   # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])   # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1]) # Wave height

# Convert wind direction to vector (u, v)
df$wind_u = cos(df$wind_dir * pi / 180)  # x component
df$wind_v = sin(df$wind_dir * pi / 180)  # y component

# Map visualization
p4 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = wave_height), alpha = 0.5) +  # Background color: wave height
  scale_fill_viridis_c(name = "wave_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * wind_u, yend = lat + 0.2 * wind_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "black", alpha = 0.8, size = 0.5) +  # Wind direction (black arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
```

```
              arrow = arrow(length = unit(0.2, "cm")),
              color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200421 Wind & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p4
```



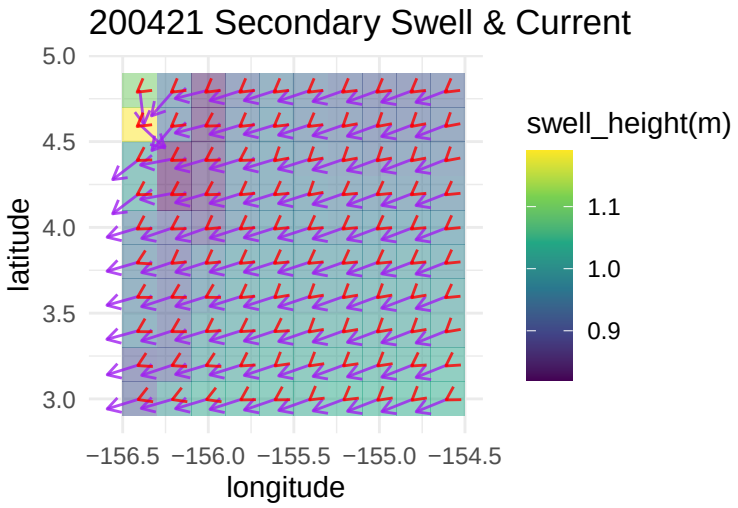

```
#================================================================================
# 200421 Primary Swell & Current
#================================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739806290060.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW1")     # Primary swell direction
swell_height = ncvar_get(nc, "VHM0_SW1") # Primary swell height
swell_period = ncvar_get(nc, "VTM01_SW1")# Primary swell period
stokes_x = ncvar_get(nc, "VSDX")          # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")          # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")       # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])     # Primary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Primary swell height
df$swell_period = as.vector(swell_period[,,1]) # Primary swell period
df$stokes_x = as.vector(stokes_x[,,1])       # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])       # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
```

```r
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p5 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: primary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "blue", alpha = 0.8, size = 0.5) +  # Primary swell direction (blue arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200421 Primary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p5
```



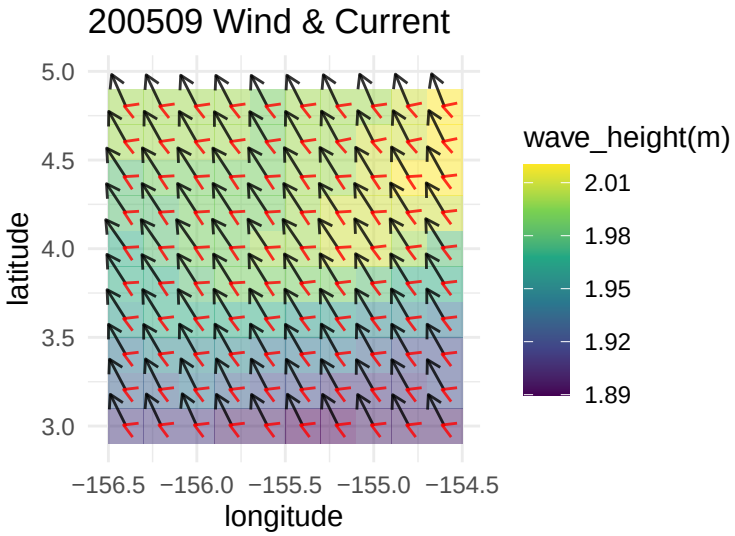

```r
#==============================================================================
# 200421 Secondary Swell & Current
#==============================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739806388750.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW2")     # Secondary swell direction
swell_height = ncvar_get(nc, "VHM0_SW2") # Secondary swell height
swell_period = ncvar_get(nc, "VTM01_SW2")# Secondary swell period
stokes_x = ncvar_get(nc, "VSDX")          # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")          # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")       # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
```

```r
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])     # Secondary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Secondary swell height
df$swell_period = as.vector(swell_period[,,1]) # Secondary swell period
df$stokes_x = as.vector(stokes_x[,,1])       # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])       # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p6 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: secondary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "purple", alpha = 0.8, size = 0.5) +  # Secondary swell direction (purple arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200421 Secondary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p6
```

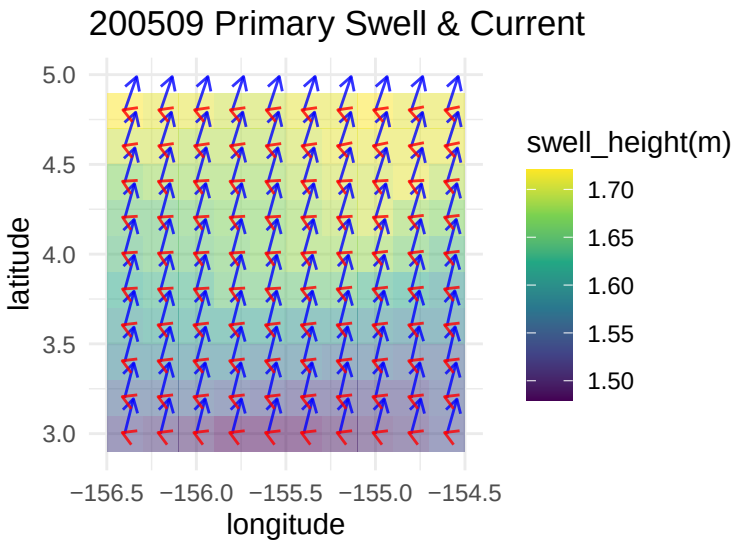



```r
#==========================================================================
# 200509 Wind & Current
#==========================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739806673914.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
wind_dir = ncvar_get(nc, "VMDR_WW")  # Wind Wave Direction
```

```r
wave_dir = ncvar_get(nc, "VMDR")     # Mean Wave Direction
stokes_x = ncvar_get(nc, "VSDX")     # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")     # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0") # Significant Wave Height

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$wind_dir = as.vector(wind_dir[,,1])    # Wind direction (azimuth)
df$wave_dir = as.vector(wave_dir[,,1])    # Wave direction
df$stokes_x = as.vector(stokes_x[,,1])    # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])    # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1]) # Wave height

# Convert wind direction to vector (u, v)
df$wind_u = cos(df$wind_dir * pi / 180)  # x component
df$wind_v = sin(df$wind_dir * pi / 180)  # y component


# Map visualization
p7 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = wave_height), alpha = 0.5) +  # Background color: wave height
  scale_fill_viridis_c(name = "wave_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * wind_u, yend = lat + 0.2 * wind_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "black", alpha = 0.8, size = 0.5) +  # Wind direction (black arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200509 Wind & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p7
```

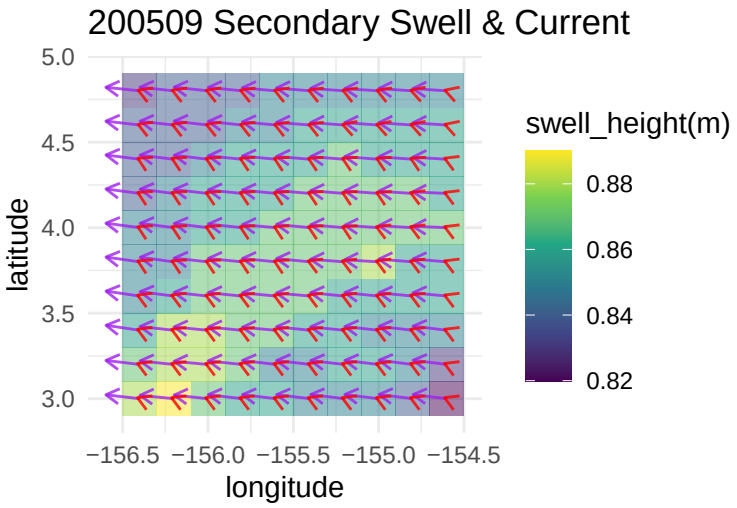


```r
#==============================================================================
# 200509 Primary Swell & Current
#==============================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739807093609.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW1")     # Primary swell direction
swell_height = ncvar_get(nc, "VHM0_SW1") # Primary swell height
swell_period = ncvar_get(nc, "VTM01_SW1")# Primary swell period
stokes_x = ncvar_get(nc, "VSDX")          # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")          # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")       # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])     # Primary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Primary swell height
df$swell_period = as.vector(swell_period[,,1]) # Primary swell period
df$stokes_x = as.vector(stokes_x[,,1])       # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])       # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p8 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: primary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "blue", alpha = 0.8, size = 0.5) +  # Primary swell direction (blue arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200509 Primary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p8
```




```
#================================================================================
# 200509 Secondary Swell & Current
#================================================================================
nc_file = "Data/cmems_mod_glo_wav_my_0.2deg_PT3H-i_1739807415115.nc"
nc = nc_open(nc_file)

# Get variables
lon = ncvar_get(nc, "longitude")
lat = ncvar_get(nc, "latitude")
swell_dir = ncvar_get(nc, "VMDR_SW2")     # Secondary swell direction
swell_height = ncvar_get(nc, "VHM0_SW2") # Secondary swell height
swell_period = ncvar_get(nc, "VTM01_SW2")# Secondary swell period
stokes_x = ncvar_get(nc, "VSDX")          # Stokes Drift X (current)
stokes_y = ncvar_get(nc, "VSDY")          # Stokes Drift Y (current)
wave_height = ncvar_get(nc, "VHM0")       # Total wave height (for reference)

# Close file
nc_close(nc)

# Convert to dataframe
df = expand.grid(lon = lon, lat = lat)
df$swell_dir = as.vector(swell_dir[,,1])     # Secondary swell direction (azimuth)
df$swell_height = as.vector(swell_height[,,1]) # Secondary swell height
df$swell_period = as.vector(swell_period[,,1]) # Secondary swell period
df$stokes_x = as.vector(stokes_x[,,1])       # Stokes Drift X component
df$stokes_y = as.vector(stokes_y[,,1])       # Stokes Drift Y component
df$wave_height = as.vector(wave_height[,,1])  # Total wave height

# Convert swell direction to vector (u, v)
df$swell_u = cos(df$swell_dir * pi / 180)  # x component
df$swell_v = sin(df$swell_dir * pi / 180)  # y component

# Map visualization
p9 = ggplot(df, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = swell_height), alpha = 0.5) +  # Background color: secondary swell height
  scale_fill_viridis_c(name = "swell_height(m)") +
```

```
  geom_segment(aes(xend = lon + 0.2 * swell_u, yend = lat + 0.2 * swell_v),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "purple", alpha = 0.8, size = 0.5) +  # Secondary swell direction (purple arrow)
  geom_segment(aes(xend = lon + 0.2 * stokes_x, yend = lat + 0.2 * stokes_y),
               arrow = arrow(length = unit(0.2, "cm")),
               color = "red", alpha = 0.8, size = 0.5) +  # Stokes Drift (current direction, red arrow)
  coord_quickmap() +
  labs(title = "200509 Secondary Swell & Current", x = "longitude", y = "latitude") +
  theme_minimal()
p9
```




```
#================================================================================
# Figure 2
#================================================================================
# p0 = grid.arrange(p1,p2,p3,p4,p5,p6,p7,p8,p9, nrow = 3, ncol = 3)
```