# SURE Project

September 05, 2022

## Objective

Predict and project spatially the result from the models.

```
## Script by Anita Giraldo, 4 May 2022
## Last modified by Anita Giraldo, 4 May 2022

# Clear environment ----
# rm(list = ls())

# directories ----
m.dir <- here()
d.dir <- here('data')

## Load info on years RCCA ----
years <- read.csv(paste(d.dir, "RCCA_North_Coast_sites.csv", sep ='/')) # Rows: 25

# get the sites from with preMHW data ----
# 3 or more pre MHW surveys
ncsites <- years %>%
  mutate_at(vars(site_name), list(as.factor)) %>%
  # get only sites with PRE MHW data
  dplyr::filter(pre.mhw.years > 2) %>%
  droplevels() # Rows: 10
```

## 1. Load RCCA data

```
df <- read.csv(paste(d.dir, "RCCA_kelp_inverts_NC_depth-zones_wave_clim_temp_nit_subs_orbvel_npp.csv", s
  mutate_at(vars(site_name, month, year, transect, zone), list(as.factor)) %>%
  mutate(zone_new = case_when(
    transect == '1' ~ 'OUTER',
    transect == '2' ~ 'OUTER',
    transect == '3' ~ 'OUTER',
    transect == '4' ~ 'INNER',
    transect == '5' ~ 'INNER',
    transect == '6' ~ 'INNER')) %>%
  dplyr::select(-zone) %>%
  rename(zone = zone_new) %>%
  mutate_at(vars(zone), list(as.factor)) %>%
  relocate(zone, .after = transect) # Rows: 1,154

## get the sites for North Coast model ----
```

```r
df.nc <- df %>%
  dplyr::select(-c(latitude, longitude)) %>%
  right_join(ncsites, by = c('site_name')) %>%
  droplevels() %>% # glimpse()
  # dplyr::select(-c(total.years, pre.mhw.years, during.mhw.years, post.mhw.years)) %>%
  relocate(c(latitude, longitude), .after = zone) # Rows: 708

length(levels(df.nc$site_name)) # 10
```

```
## [1] 10
```

```r
levels(df.nc$site_name)
```

```
##  [1] "Fort Ross"            "Gerstle Cove"         "Mendocino Headlands"
##  [4] "Ocean Cove"           "Point Arena MPA (M2)" "Point Arena Ref"
##  [7] "Portuguese Beach"     "Stillwater Sonoma"    "Stornetta"
## [10] "Van Damme"
```

```r
any(is.na(df.nc$Max_Monthly_Anomaly_Temp)) # FALSE
```

```
## [1] FALSE
```

## 2. Choose variables and transform needed

```r
# names(df.nc)
dat1 <- df.nc %>%
  dplyr::select(
    # Factors
    latitude, longitude,
    site_name, year, transect, zone,
    # Bio vars
    den_NERLUE , den_MESFRAAD , den_STRPURAD , den_PYCHEL, den_HALRUF,
    # Nitrate vars
    Days_10N,
    Min_Monthly_Nitrate,
    Max_Monthly_Nitrate,
    Mean_Monthly_Nitrate,
    Mean_Monthly_Upwelling_Nitrate,
    Max_Monthly_Anomaly_Nitrate,
    Mean_Monthly_Summer_Nitrate,
    # Temperature vars
    Days_16C ,
    Mean_Monthly_Temp ,
    Mean_Monthly_Summer_Temp,
    MHW_Upwelling_Days  ,
    Min_Monthly_Anomaly_Temp,
    Max_Monthly_Anomaly_Upwelling_Temp,
    Min_Monthly_Temp,
    Mean_Monthly_Upwelling_Temp,
```

```
    #wh.95 ,   wh.max,
    npgo_mean , mei_mean,
    # substrate
    mean_depth, mean_prob_of_rock, mean_vrm, mean_slope,
    # waves
    wh_max, wh_mean, mean_waveyear, wh_95prc,
    # Orb vel
    UBR_Mean, UBR_Max,
    # NPP
    Mean_Monthly_NPP, Max_Monthly_NPP_Upwelling, Mean_Monthly_NPP_Upwelling, Min_Monthly_NPP,
  ) %>%
  # Bio transformations
  mutate(log_den_NERLUE = log(den_NERLUE + 1),
         log_den_MESFRAAD = log(den_MESFRAAD + 1),
         log_den_STRPURAD = log(den_STRPURAD + 1),
         log_den_PYCHEL = log(den_PYCHEL + 1),
         log_den_HALRUF = log(den_HALRUF + 1),
         log_mean_vrm = log(mean_vrm + 1)) %>%
  dplyr::select(-c(den_NERLUE,
                   den_MESFRAAD,
                   den_STRPURAD,
                   den_PYCHEL,
                   den_HALRUF,
                   mean_vrm)) %>%
  # Temperature transformations
  mutate(log_Days_16C = log(Days_16C + 1)) %>%
  dplyr::select(-c(Days_16C)) %>%
  # Orb vel transformations
  mutate(log_UBR_Mean = log(UBR_Mean + 1),
         log_UBR_Max = log(UBR_Max + 1)) %>%
  dplyr::select(-c(UBR_Mean,
                   UBR_Max)) %>%
  # NPP transformations
  mutate(log_Mean_Monthly_NPP_Upwelling = log(Mean_Monthly_NPP_Upwelling + 1),
         log_Min_Monthly_NPP = log(Min_Monthly_NPP + 1)) %>%
  dplyr::select(-c(Mean_Monthly_NPP_Upwelling,
                   Min_Monthly_NPP)) # Rows: 708

# log(x + 1) avoids log(0)

#### Drop NAs ----
dat2 <- dat1 %>%
  drop_na() # Rows: 686


# glimpse(dat2)
levels(dat2$year)
```

```
##  [1] "2006" "2007" "2008" "2009" "2010" "2011" "2012" "2013" "2014" "2015"
## [11] "2016" "2017" "2018" "2019" "2020" "2021"
```

## 3. Divide data into train and test

```
# Split data into a training set (75%), and a testing set (25%)
inTraining <- createDataPartition(dat2$log_den_NERLUE, p = 0.75, list = FALSE)
train.gam <- dat2[ inTraining,]
test.gam  <- dat2[-inTraining,]
```

## 4. Run GAM

```
gam1 <- gam(formula = log_den_NERLUE ~
              s(log_den_STRPURAD, k = 5, bs = "cr") + # purple sea urchins
              s(Max_Monthly_Nitrate, k = 5, bs = "cr") +
              s(wh_max, k = 5, bs = "cr") + # wave height
              s(log_UBR_Max, k = 4, bs = "cr") + # orbital velocity, continuous variables
              s(site_name, zone, bs = "re") + # zone is nested within site
              s(year, bs = "re"), # discrete variables, categorical
            # random factor
              family = tw(), data = dat2, method = "REML")

# k: the dimension of the basis used to represent the smooth terms
# bs = 'cr': cubic regression splines
# bs = 're': random effects, penalized by a ridge penalty
```

## 5. Check GAM

```
gam1$aic # model selection?
```

```
## [1] 1684.045
```

```
gam1$deviance # goodness of fit
```

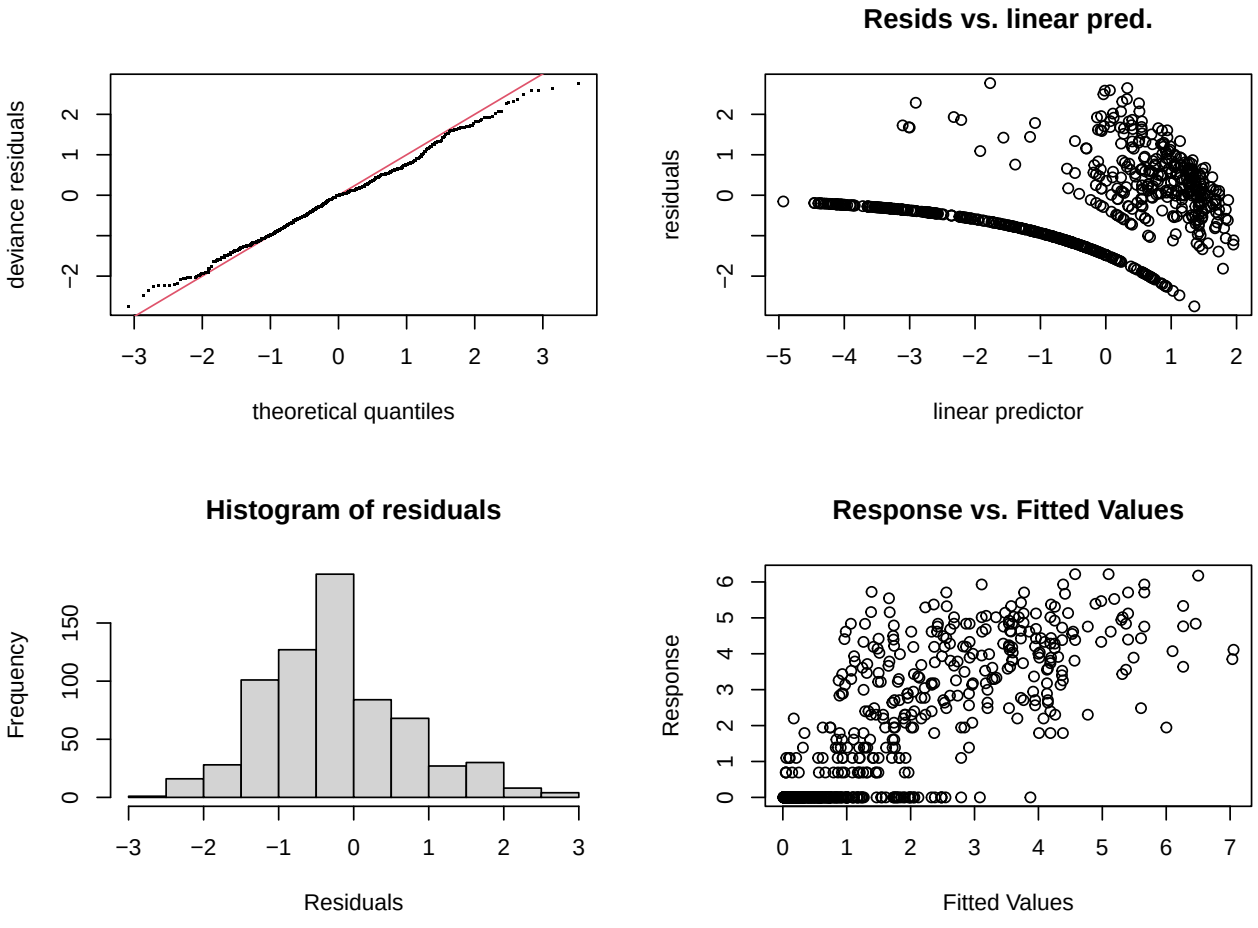
```
## [1] 650.3355
```

```
summary(gam1)
```

```
##
## Family: Tweedie(p=1.085)
## Link function: log
##
## Formula:
## log_den_NERLUE ~ s(log_den_STRPURAD, k = 5, bs = "cr") + s(Max_Monthly_Nitrate,
##     k = 5, bs = "cr") + s(wh_max, k = 5, bs = "cr") + s(log_UBR_Max,
##     k = 4, bs = "cr") + s(site_name, zone, bs = "re") + s(year,
##     bs = "re")
##
## Parametric coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5254     0.3789  -1.387    0.166
##
## Approximate significance of smooth terms:
##                            edf Ref.df      F  p-value
## s(log_den_STRPURAD)      2.358  2.680  8.543 6.17e-05 ***
## s(Max_Monthly_Nitrate)  3.294  3.651 11.159  < 2e-16 ***
## s(wh_max)                3.698  3.918 10.613  < 2e-16 ***
## s(log_UBR_Max)           2.667  2.884  7.611 0.000286 ***
## s(site_name,zone)       15.335 19.000  4.033 2.31e-05 ***
## s(year)                 12.784 15.000  9.229  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.623   Deviance explained = 64.5%
## -REML = 880.45  Scale est. = 1.3074    n = 686
```

```
gam.check(gam1) # model diagnostic plots
```




```
##
## Method: REML   Optimizer: outer newton
## full convergence after 8 iterations.
## Gradient range [-2.23194e-08,3.732632e-09]
```

```
## (score 880.4514 & scale 1.30745).
## Hessian positive definite, eigenvalue range [0.5091827,617.4705].
## Model rank =  52 / 52
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                            k'   edf k-index p-value
## s(log_den_STRPURAD)     4.00  2.36    0.80  <2e-16 ***
## s(Max_Monthly_Nitrate)  4.00  3.29    0.59  <2e-16 ***
## s(wh_max)               4.00  3.70    0.59  <2e-16 ***
## s(log_UBR_Max)          3.00  2.67    0.62  <2e-16 ***
## s(site_name,zone)      20.00 15.33      NA      NA
## s(year)                16.00 12.78      NA      NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

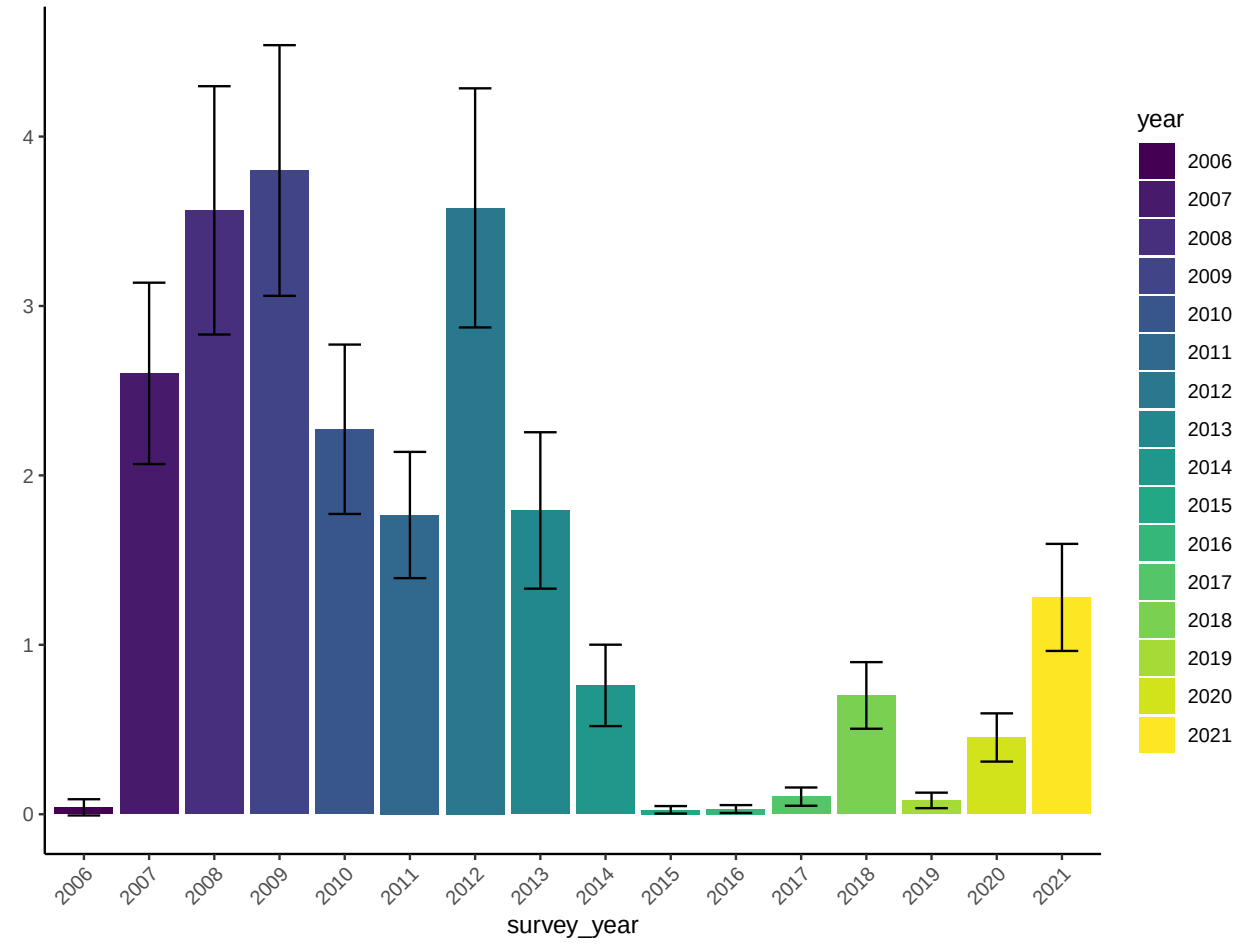
```
# The effective degrees of freedom (EDF) reflects the degree of non-linearity
# of a curve. As the edf increasingly exceeds 2, the degree of non-linearity
# progressively increases.

# Residual plotting aims to show that there is something wrong with the model
# assumptions.
# The key assumptions are
# 1. The assumed mean variance relationship is correct, so that scaled residuals
# have constant variance.
# 2. The response data are independent, so that the residuals appear approximately
# so.

# visualize responses
par(mfrow = c(3, 3), mar = c(2, 4, 3, 1))
visreg(gam1)
dev.off()
```

```
## null device
##           1
```

## 6. Predict to compare to observed

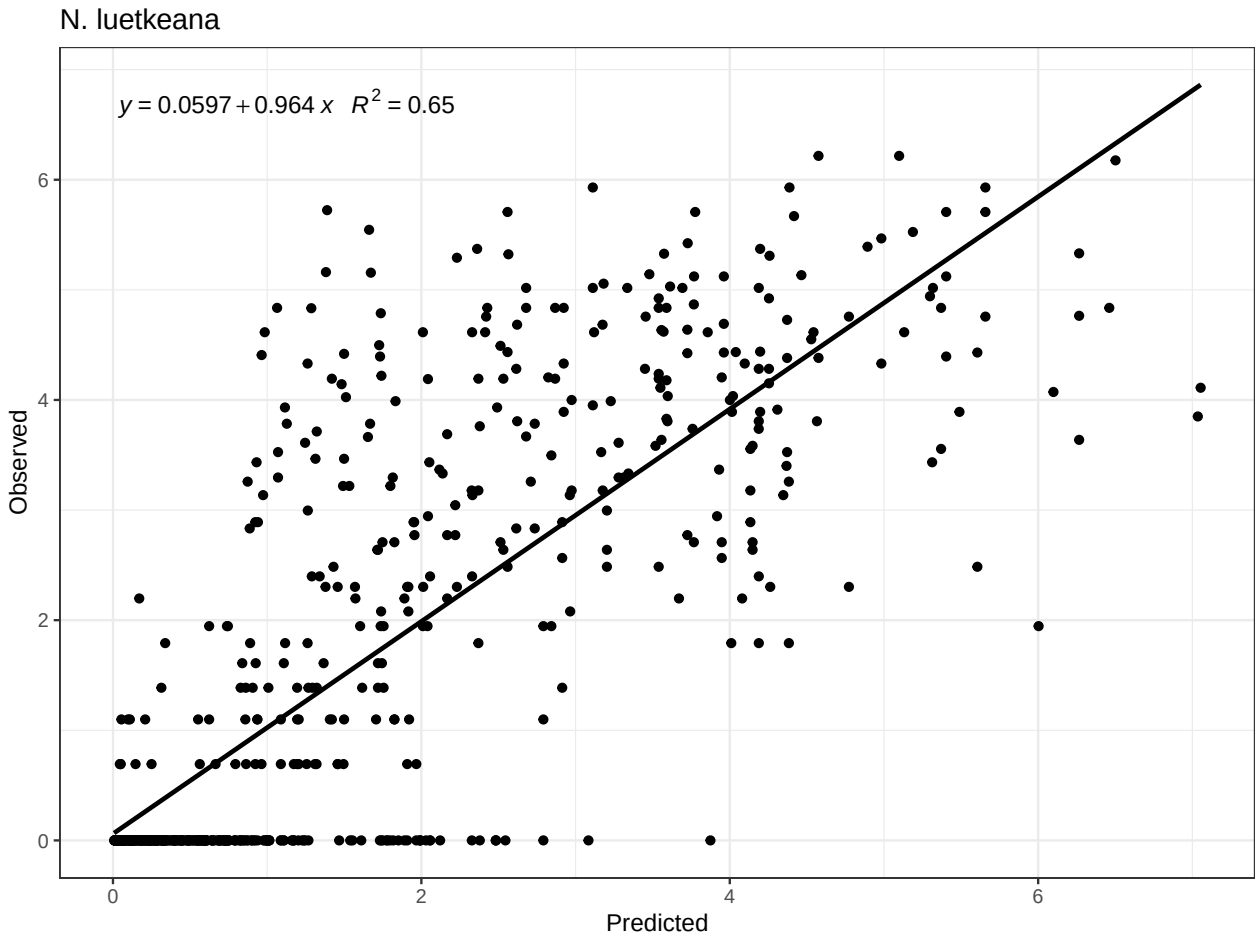
```
testdata <- dat2 %>%
  dplyr::select("log_den_STRPURAD",
                "log_mean_vrm",
                "log_UBR_Max",
                "Max_Monthly_Nitrate",
                "wh_max",
                "log_den_NERLUE",
                "site_name", "zone", "year")

# head(testdata)

# fit the data
```

```
fits <- predict.gam(gam1, newdata = testdata, type = 'response', se.fit = T)


## predict average kelp per year --
predicts.year <- testdata %>%
  data.frame(fits) %>%
  group_by(year) %>% #only change here
  summarise(response = mean(fit, na.rm = T), se.fit = mean(se.fit, na.rm = T)) %>%
  ungroup()


ggmod.year <- ggplot(aes(x = year, y = response, fill = year), data = predicts.year) +
  ylab(" ")+
  xlab('survey_year')+
  scale_fill_viridis(discrete = T) +
  geom_bar(stat = "identity")+
  geom_errorbar(aes(ymin = response - se.fit, ymax = response + se.fit),width = 0.5) +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 45, h = 1))

ggmod.year
```



## 7. Plot observed vs. predicted

```
predicts.all <-  testdata %>%
  data.frame(fits) %>%
  #group_by(survey_year) %>% #only change here
  #summarise(response=mean(fit), se.fit = mean(se.fit)) %>%
  ungroup()


# Plot observed vs. predicted --
library(ggpmisc)

my.formula <- y ~ x

p <- ggplot(predicts.all, aes(x = fit, y = log_den_NERLUE)) +
  geom_smooth(method = "lm", se=FALSE, color="black", formula = my.formula) +
  stat_poly_eq(formula = my.formula,
               aes(label = paste(..eq.label.., ..rr.label.., sep = "~~~")),
               parse = TRUE) +
  geom_point() +
  #scale_color_viridis(discrete = T) +
  labs(x = 'Predicted', y = 'Observed', title = 'N. luetkeana') +
  theme_bw()
p
```




**Predict best model across all years and site that I have data for**

```
# gam1
# Max_Monthly_Nitrate
# log_UBR_Max
# wh_max
# log_den_STRPURAD

### Get depth ----

depth.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors/depth"

# names(dat2)

# depth <- rast(paste(depth.dir, "depth_mean_nc.all_wInterp_300m_30m.tif", sep ='/'))
# aggregate from 300x300 resolution to 900x900 (factor = 3)
# depth2 <- aggregate(depth, fact = 3, fun = mean, na.rm = TRUE)
# writeRaster(depth2, file.path(depth.dir, 'depth_mean_nc.all_wInterp_900m_30m.tif'), overwrite = TRUE)
# disaggregate from 300x300 resolution to 150x150 (factor = 2)
# depth3 <- disagg(depth, fact = 2)
# writeRaster(depth3, file.path(depth.dir, 'depth_mean_nc.all_wInterp_150m_30m.tif'), overwrite = TRUE)
depth <- rast(paste(depth.dir, "depth_mean_nc.all_wInterp_900m_30m.tif", sep ='/'))
```

```r
# plot(depth)

n.extent <- ext(depth)

# depth # EPSG:26910

crs1 <- "epsg:4326"
d2 <- project(depth, crs1) # depth # EPSG:4326

n.extent <- ext(d2)
```

```r
## Get rock ----
sub.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors/rock"

# dir(sub.dir)

# rock <- rast(paste(sub.dir, "prob_rock_nc_MASKED_LatLong_all_300m_wInterp.tif", sep ='/'))
# aggregate from 300x300 resolution to 900x900 (factor = 3)
# rock2 <- aggregate(rock, fact = 3, fun = mean, na.rm = TRUE)
# writeRaster(rock2, file.path(sub.dir, 'prob_rock_nc_MASKED_LatLong_all_900m_wInterp.tif'), overwrite
# disaggregate from 300x300 resolution to 150x150 (factor = 2)
# rock3 <- disagg(rock, fact = 2)
# writeRaster(rock3, file.path(sub.dir, 'prob_rock_nc_MASKED_LatLong_all_150m_wInterp.tif'), overwrite
rock <- rast(paste(sub.dir, "prob_rock_nc_MASKED_LatLong_all_900m_wInterp.tif", sep ='/'))
# rock

# # crop to NC --
rock2 <- crop(rock, ext(d2))
# plot(rock2)

rock3 <- resample(rock2, d2)
# plot(rock3)

# Resample transfers values between non matching Raster objects (in terms of origin)
# and resolution).
```

```r
### Get Env predictors  ----

re.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors"


### Get nitrate predictors  ----
max_nit <- rast(paste(re.dir, "Nitrate", "Max_Monthly_Nitrate.tif", sep ='/'))
# max_nit # multiple layers

# # crop to NC --
max_nit2 <- crop(max_nit, n.extent)
# plot(max_nit2[[1]])

# resample predictors to bathy ----
max_nit3 <- resample(max_nit2, d2)

# mask predictors to bathy ----
```

```
max_nit4 <- mask(max_nit3, d2)
# plot(max_nit4[[1]])
```

```
### Get Wave predictors  ----

w.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors"

## Max Wave height --

# load raster data --

wave.dir <- paste(w.dir, "waves", "wh_max", sep  = '/')

# load raster data --

n.files <- dir(wave.dir)
# list files in source --
n.files <- list.files(wave.dir, pattern = '.tif$', full.names = TRUE)
# n.files
# length(n.files)
# list names to load onto the Environment --
names.list <- list.files(wave.dir, pattern = '.tif$')
names.list <- str_replace(names.list, ".tif$", "")
# length(names.list)

# load csv files as a list --
tfiles <- lapply(n.files, rast) # this is a list
# tfiles[[1]]

# stack them ---
whmax.stack <- c()

# use do call to create a raster otherwise it creates a list
whmax.stack <- do.call("c", tfiles)
# plot(whmax.stack[[1]])


# # crop to NC --
whmax.stack2 <- crop(whmax.stack, n.extent)
# plot(whmax.stack2[[1]])

# resample predictors to bathy ----
whmax.stack3 <- resample(whmax.stack2, d2) # align origin, aggregate or disaggregate
                                           # to have same resolution and coordinate
                                           # system

# mask predictors to bathy ----
whmax.stack4 <- mask(whmax.stack3, d2) # make NA do not match to the d2 extent
# plot(whmax.stack4[[1]])
```

```
## Mean UBR MAX ----

w2.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors/orbital_vel"
```

```r
# load raster data --

ubr <- rast(paste(w2.dir, "UBR_Max_30m_NC.tif", sep ='/'))
# plot(ubr[[1]])

ubr <- project(ubr, rock)

# plot(ubr[[1]])

ubr1 <- classify(ubr, cbind(0, NA)) # assign the raster values 0 are reclassified
                                    # to take values NA

# plot(ubr1[[1]])


# # crop to NC --
ubr2 <- crop(ubr1, n.extent)
# plot(ubr2[[1]])

# resample predictors to bathy ----
ubr3 <- resample(ubr2, d2)

# mask predictors to bathy ----
ubr4 <- mask(ubr3, d2)
# plot(ubr4[[1]])

ubr5 <- log(ubr4)
```

```r
### Get urchins ----

# load purple urchin predictions ----
urch.dir <- "/Volumes/GoogleDrive/My Drive/SURE_Project/Spatial_data/Predictors/urchins/log_sp_predicti

# load raster data --

u.files <- dir(urch.dir)
u.files <- list.files(urch.dir, pattern = '.tif')
# u.files
# length(u.files)

##

# stack rasters --

preds1 <- c(max_nit4[[1]], whmax.stack4[[1]], ubr5[[1]])
# names(preds1)
```

```r
# get year raster ----

# make rasters for site, year, zone

# assign the raster values in the range 0-Inf are reclassifed to take values 1998
# max_nit4 has 24 layers
```

```r
year1998 <- classify(max_nit4[[1]], cbind(0, Inf, 1998), right = FALSE)
# plot(year1998)
names(year1998) <- 'year'

year.list <- paste(1998:2021)
length(year.list)
```

```
## [1] 24
```

```r
preds2 <- c(preds1, year1998)
# names(preds2)

names(preds2) <- c("Max_Monthly_Nitrate" ,
                   "wh_max",
                   "log_UBR_Max",
                   "year")
```

```r
# sites ----

rdf <- as.data.frame(year1998, xy = T)
# head(rdf)
rdf$site <- rdf$y
rdf <- rdf[,-3] # remove the column 'year'
# head(rdf)

site.raster <- rast(rdf, type = 'xyz', crs = "EPSG:4326", extent = ext(year1998))
# site.raster

# plot(site.raster)

ext(year1998)
```

```
## SpatExtent : -124.57660297128, -122.333702693402, 37.1392272098406, 42.0114450419344 (xmin, xmax, ym
```

```r
ext(site.raster)
```

```
## SpatExtent : -124.50939997, -122.39250532, 37.17282871, 42.00304467 (xmin, xmax, ymin, ymax)
```

```r
site.raster2 <- extend(site.raster, year1998)

preds3 <- c(preds2, site.raster2)
names(preds3) <- c("Max_Monthly_Nitrate"  ,
                   "wh_max",
                   "log_UBR_Max",
                   "year" ,
                   "site_name")
```

```r
# zone ----

zone.raster <- d2
names(zone.raster) <- 'zone'
```

```
# plot(zone.raster)
# levels(dat2$zone)

rec.m <-  c(-Inf, -10, 2,
            -10, 0.1, 1)

rclmat <- matrix(rec.m, ncol = 3, byrow = TRUE)
#      [,1]  [,2] [,3]
# [1,] -Inf -10.0    2
# [2,]  -10   0.1    1

zone.raster2 <- classify(zone.raster, rclmat, right = FALSE)
# plot(zone.raster2)

preds4 <- c(preds3, zone.raster2)
# names(preds4)

names(preds4) <- c("Max_Monthly_Nitrate" ,
                   "wh_max",
                   "log_UBR_Max",
                   "year",
                   "site_name",
                   "zone")
```

## LOOP to predict each year using data frame

```
nereo.mod <- gam1
summary(nereo.mod)
```

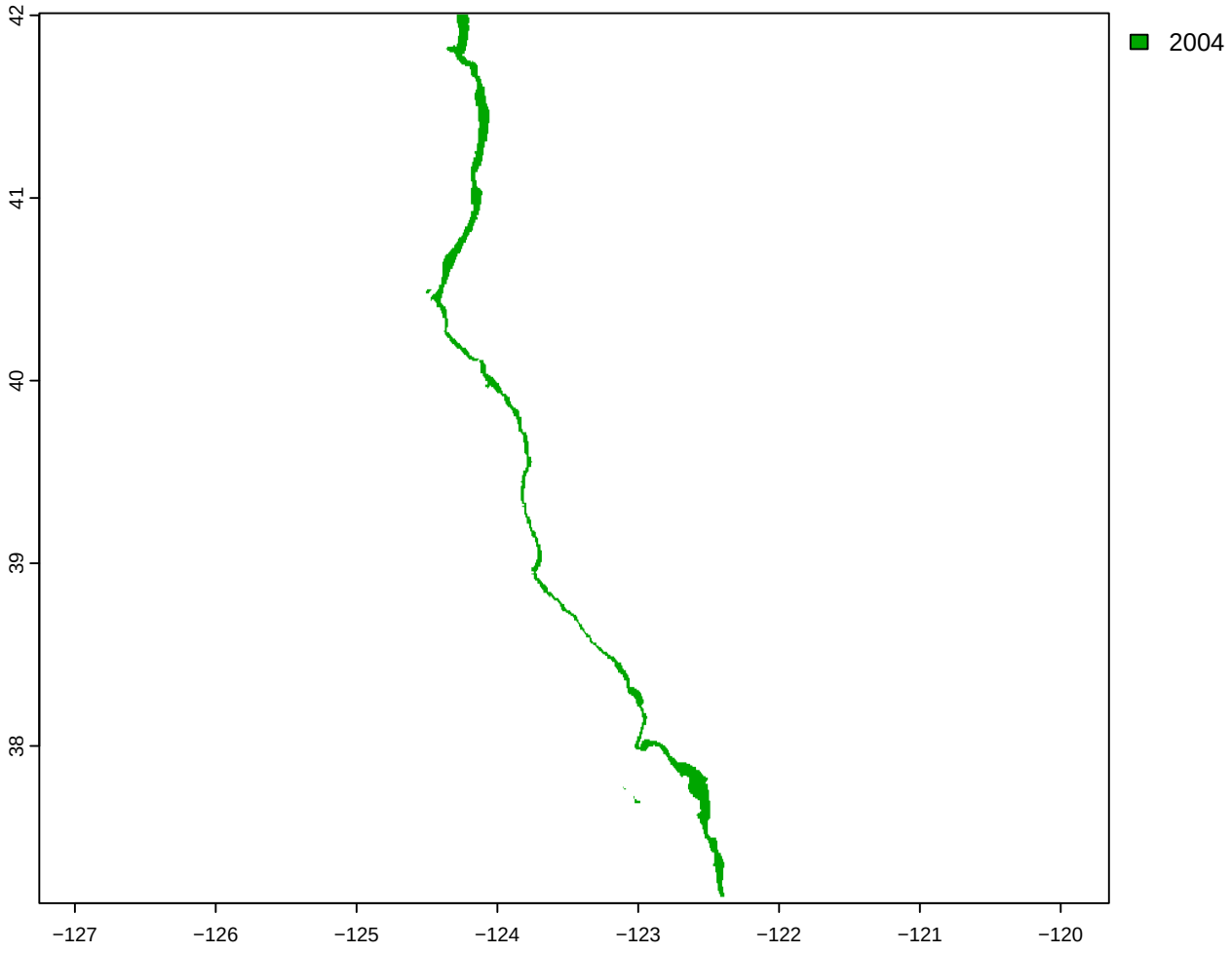
```
##
## Family: Tweedie(p=1.085)
## Link function: log
##
## Formula:
## log_den_NERLUE ~ s(log_den_STRPURAD, k = 5, bs = "cr") + s(Max_Monthly_Nitrate,
##     k = 5, bs = "cr") + s(wh_max, k = 5, bs = "cr") + s(log_UBR_Max,
##     k = 4, bs = "cr") + s(site_name, zone, bs = "re") + s(year,
##     bs = "re")
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5254     0.3789  -1.387    0.166
##
## Approximate significance of smooth terms:
##                            edf Ref.df      F  p-value
## s(log_den_STRPURAD)      2.358  2.680  8.543 6.17e-05 ***
## s(Max_Monthly_Nitrate)   3.294  3.651 11.159  < 2e-16 ***
## s(wh_max)                3.698  3.918 10.613  < 2e-16 ***
## s(log_UBR_Max)           2.667  2.884  7.611 0.000286 ***
## s(site_name,zone)       15.335 19.000  4.033 2.31e-05 ***
## s(year)                 12.784 15.000  9.229  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.623   Deviance explained = 64.5%
## -REML = 880.45  Scale est. = 1.3074    n = 686
```

```
# make list of years --
year.list <- paste(2004:2021)
length(year.list)
```

```
## [1] 18
```

```
# make template raster of year ----
year.raster <- classify(d2, cbind(-Inf, 0.1, 2004), right=FALSE)
plot(year.raster)
```




```
names(year.raster) <- 'year'

# make zone raster ----
```

```r
# outputs dir ----
# * use an output directory of yours
o2.dir <- here('spatial_data')

preds.dir <- paste(o2.dir, "sp_predictions_900m_resolution", sep ='/')
preds.dir
```

```
## [1] "/Users/chuntingzheng/Desktop/Git_Repositories/Chunting_Spatial_Analyses/spatial_data/sp_predicti
```

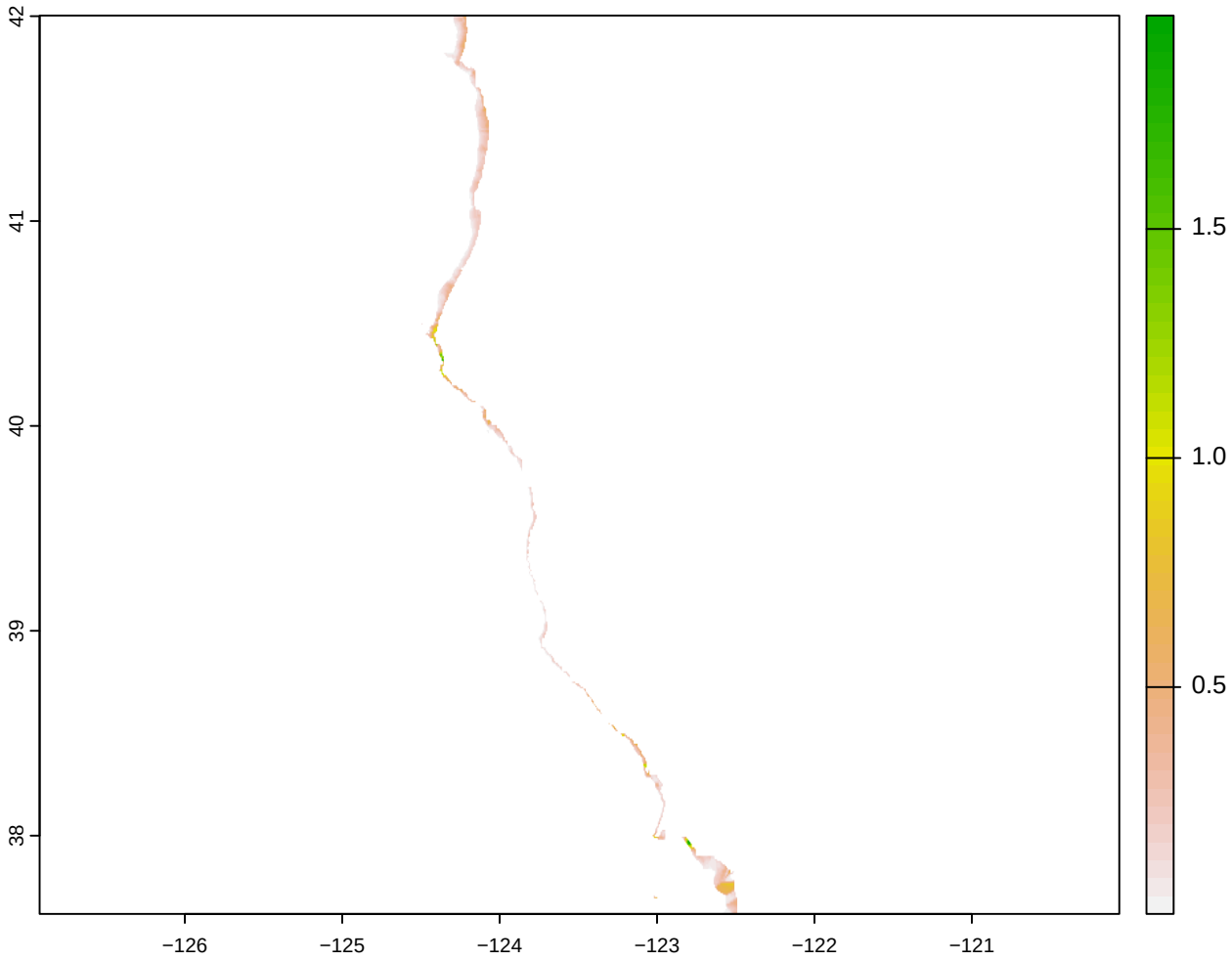
```r
# output for rasters scaled by rock
# * use an output directory of yours
rock.preds.dir <- paste(o2.dir, "sp_predictions_rock_900m_resolution", sep ='/')
rock.preds.dir
```

```
## [1] "/Users/chuntingzheng/Desktop/Git_Repositories/Chunting_Spatial_Analyses/spatial_data/sp_predicti
```

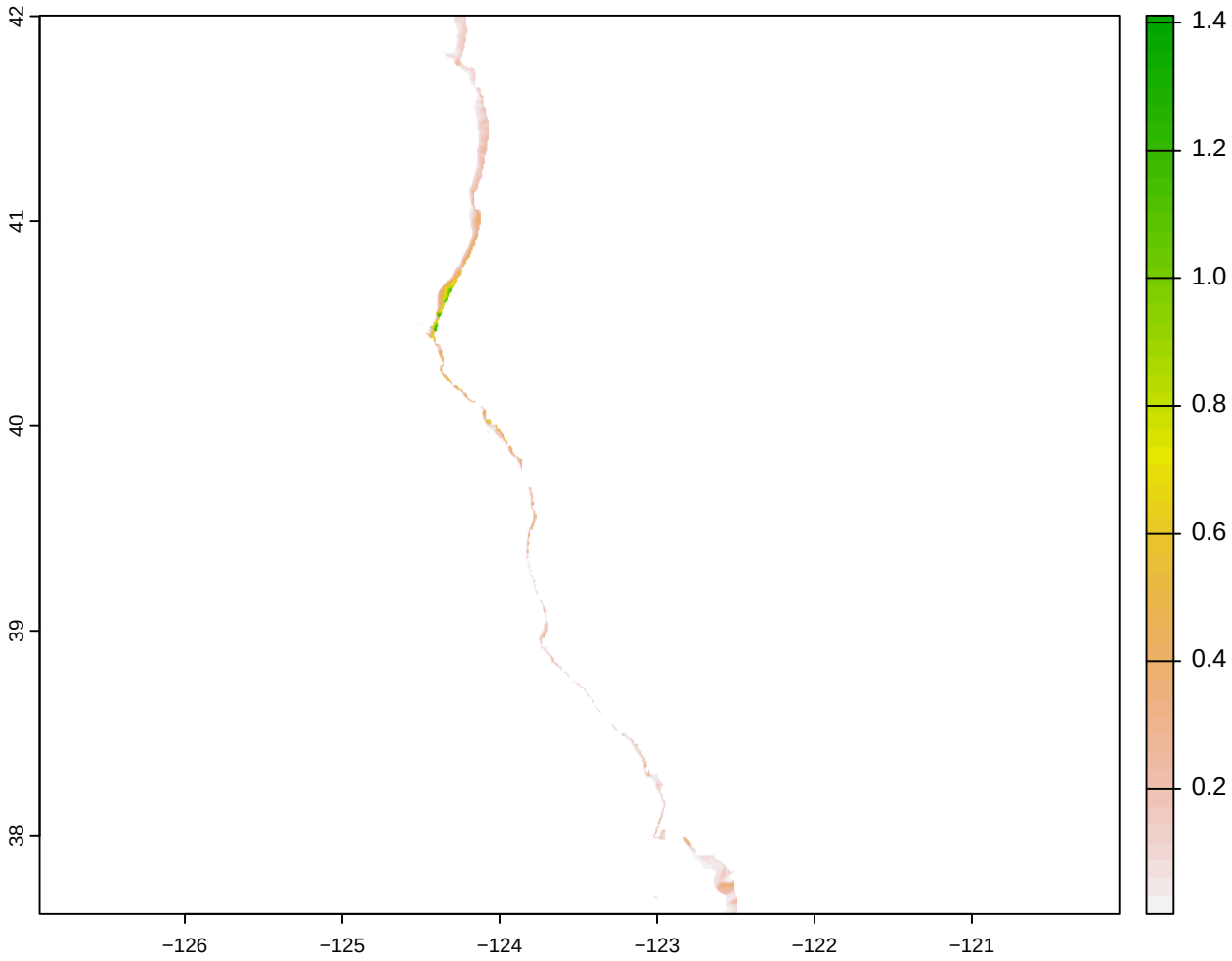
```r
for (i in 1:length(year.list)) {

  # 1. get urchins
  urchin.rast <- rast(paste(urch.dir, u.files[i], sep ='/'))
  urchin.rast2 <- resample(urchin.rast, d2)

  # 2. stack with predictors for that year
  #env.raster <- c(d2, max_nit4[[i+6]], whmax.stack4[[i]], wymean.stack4[[i]])

  #env.raster <- c(d2, mean_up_T4[[i+6]], max_nit4[[i+6]], whmax.stack4[[i]], wymean.stack4[[i]])

  # V3
  env.raster <- c(max_nit4[[i+6]], whmax.stack4[[i]], ubr5[[i]])

  preds1 <- c(urchin.rast2, env.raster)

  # 3. get year and stack it
  year.no <- as.numeric(year.list[i])
  year.r <- classify(year.raster, cbind(-Inf, 0, year.no), right=FALSE)

  preds2 <- c(preds1, year.r)

  # 3. stack zone
  preds3 <- c(preds2, zone.raster2)

  # 4. stack site
  preds4 <- c(preds3, site.raster2)

  # name predictors
  names(preds4) <- c("log_den_STRPURAD",
                     "Max_Monthly_Nitrate"  ,
                     "wh_max",
                     "log_UBR_Max",
                     "year",
                     "zone",
                     "site_name")
```

```
  df4 <- as.data.frame(preds4, xy = T) %>%
    mutate_at(vars(year, zone, site_name), list(as.factor)) %>%
    mutate(zone = recode_factor(zone, '1' = 'INNER', '2' = 'OUTER')) %>%
    glimpse()

  # 5. predict
  year.pred.df <- predict.gam(nereo.mod, newdata = df4, type = 'response', se.fit = T)
  head(year.pred.df)

  # join with df for lats and lons
  preds.all <-  df4 %>%
    data.frame(year.pred.df) %>%
    dplyr::select(x, y, fit) %>%
    glimpse()

  # 6. Rasterize
  crs.p <- "epsg:4326"
  year.prediction <- rast(preds.all, type = 'xyz', crs = crs.p, digits = 6)
  plot(year.prediction)

  # 7. save raw raster
  # name.raster <- paste(year.no, "Log_Nereo_GIVE_IT_A_NAME.tif", sep = '_')
  name.raster <- paste(year.no, "Log_Nereo_NC.tif", sep = '_')
  # writeRaster(year.prediction, paste(preds.dir, name.raster, sep = '/'))

  # 8. scale by rock
  rock4 <- resample(rock3, year.prediction)
  year.prediction2 <- rock4 * year.prediction

  # 9. save rastr scaled by rock
  # name.raster.rock <- paste(year.no, "Log_Nereo_rock__GIVE_IT_A_NAME.tif", sep = '_')
  name.raster.rock <- paste(year.no, "Log_Nereo_rock_NC.tif", sep = '_')
  # writeRaster(year.prediction2, paste(rock.preds.dir, name.raster.rock, sep = '/'))
}
```

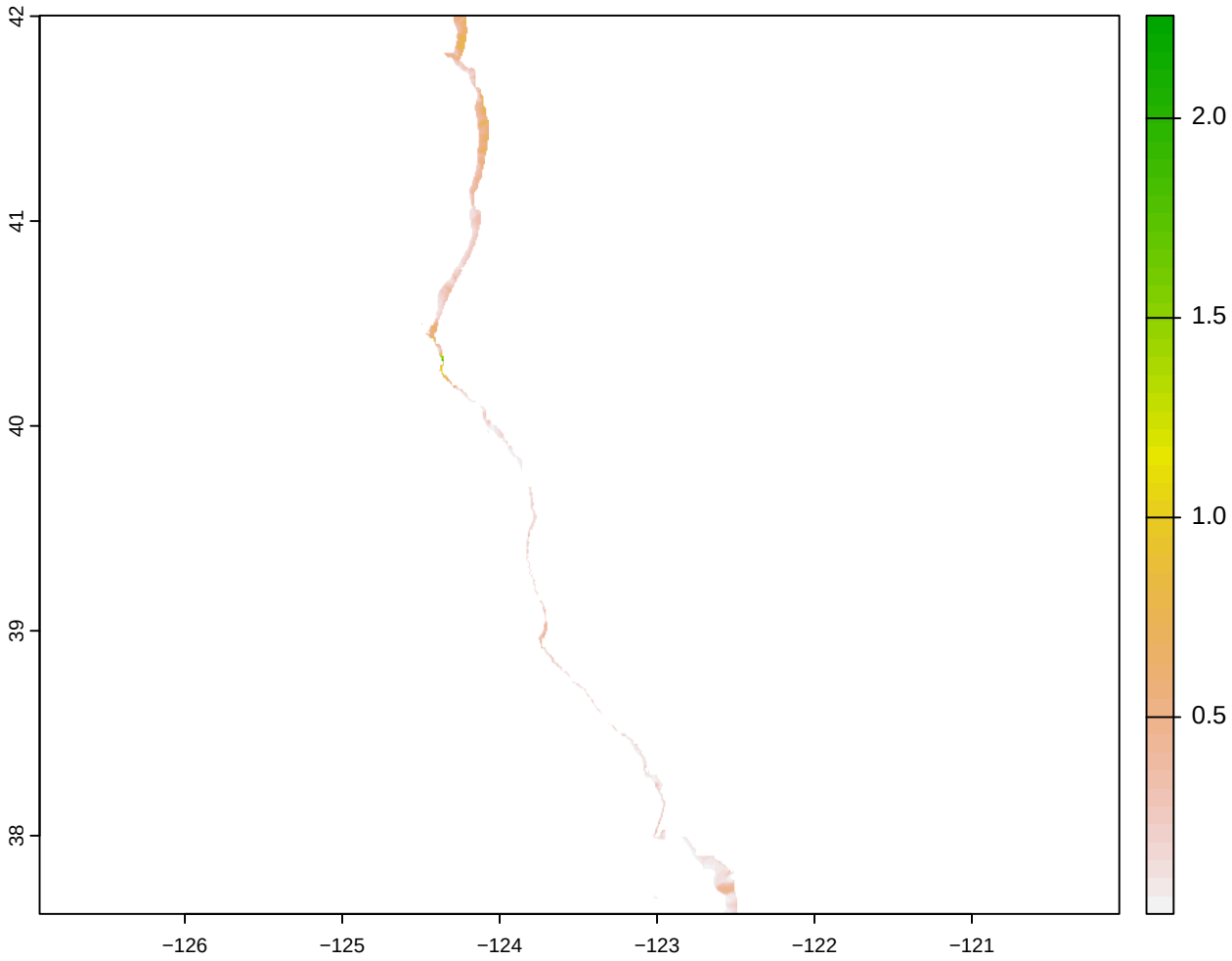
```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 8.165331, 7.645896, 7.144061, 6.724667, 6.758957, ~
## $ Max_Monthly_Nitrate <dbl> 15.88245, 15.91557, 15.95458, 15.99759, 16.04059, ~
## $ wh_max              <dbl> 7.346423, 7.346423, 7.346423, 7.346423, 7.346423, ~
## $ log_UBR_Max         <dbl> 4.403312, 4.442105, 4.492454, 4.519158, 4.556305, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.06476205, 0.07876568, 0.09720306, 0.11404618, 0.12144096, 0.1043~
```



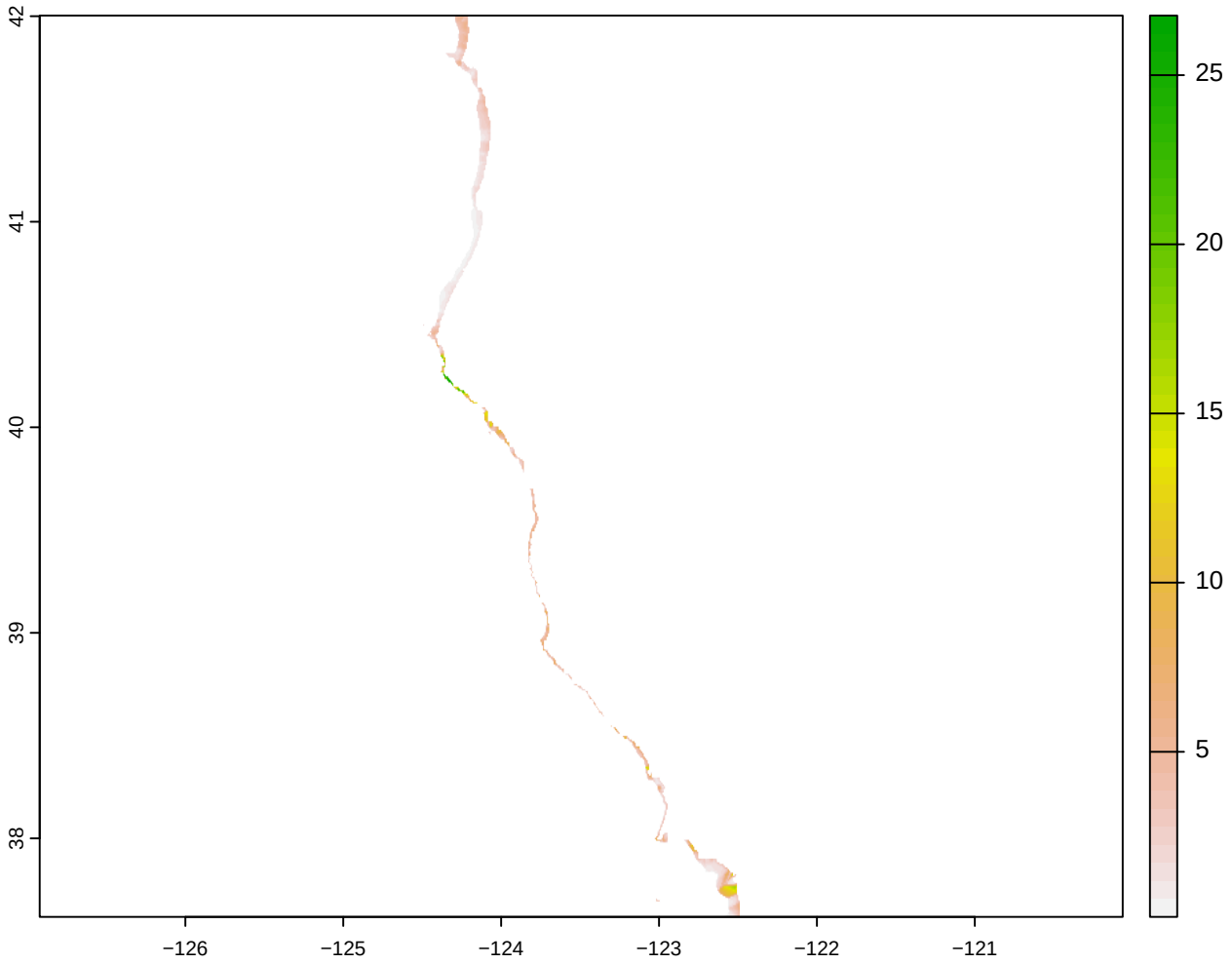

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 5.098059, 4.758600, 4.432178, 4.159541, 4.168014, ~
## $ Max_Monthly_Nitrate <dbl> 11.58586, 11.59447, 11.60152, 11.60751, 11.61350, ~
## $ wh_max              <dbl> 6.833786, 6.833786, 6.833786, 6.833786, 6.833786, ~
## $ log_UBR_Max         <dbl> 4.243152, 4.287691, 4.343748, 4.373198, 4.410262, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.03636405, 0.04315182, 0.05117806, 0.05682922, 0.05987507, 0.0571~
```



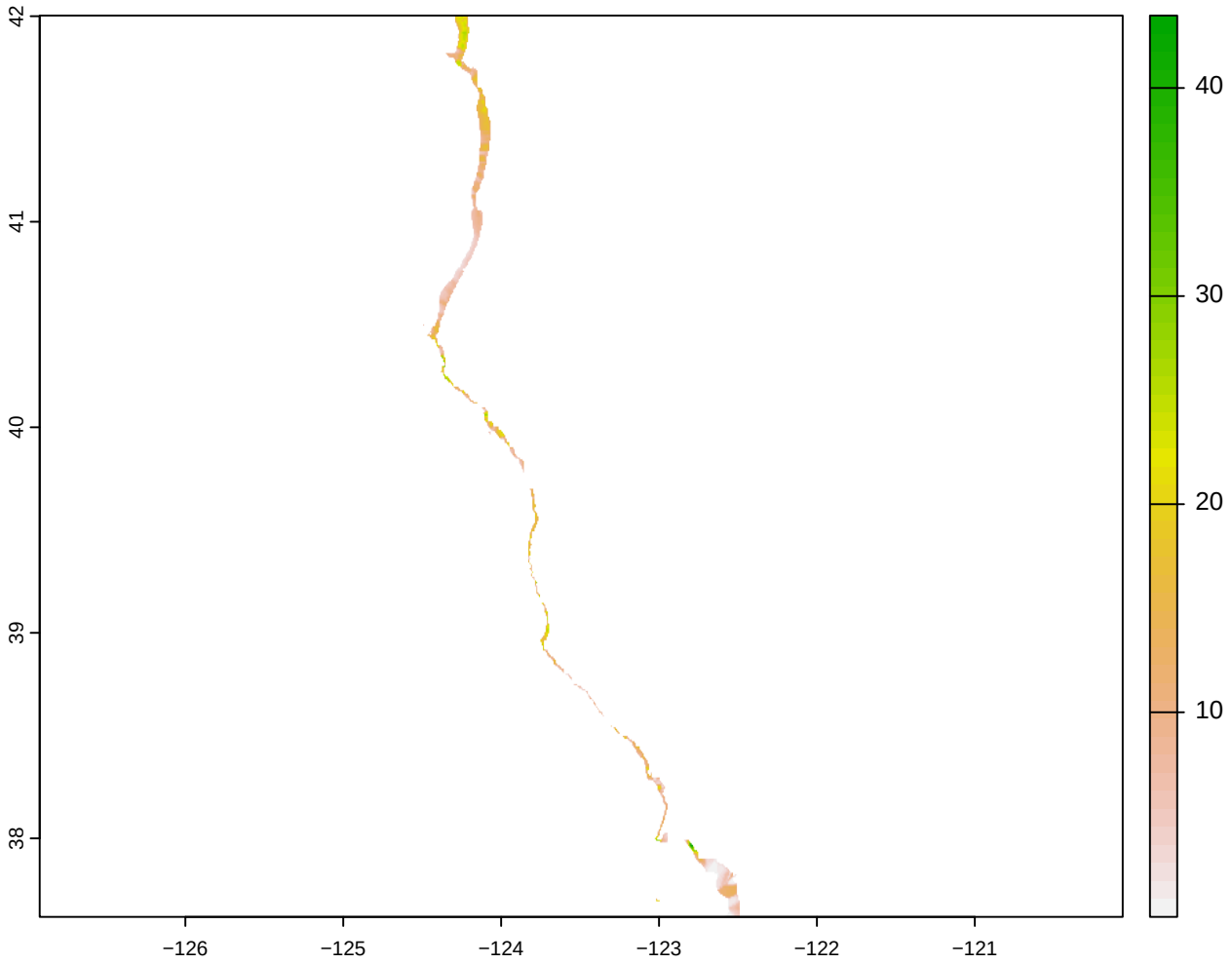

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 0.30573493, 0.28500643, 0.26541036, 0.24927713, 0.~
## $ Max_Monthly_Nitrate <dbl> 16.88462, 16.85879, 16.82238, 16.77880, 16.73522, ~
## $ wh_max              <dbl> 7.301894, 7.301894, 7.301894, 7.301894, 7.301894, ~
## $ log_UBR_Max         <dbl> 4.313672, 4.358007, 4.413838, 4.445281, 4.485267, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.3080737, 0.3244859, 0.3454846, 0.3535218, 0.3663546, 0.4099440, ~
```



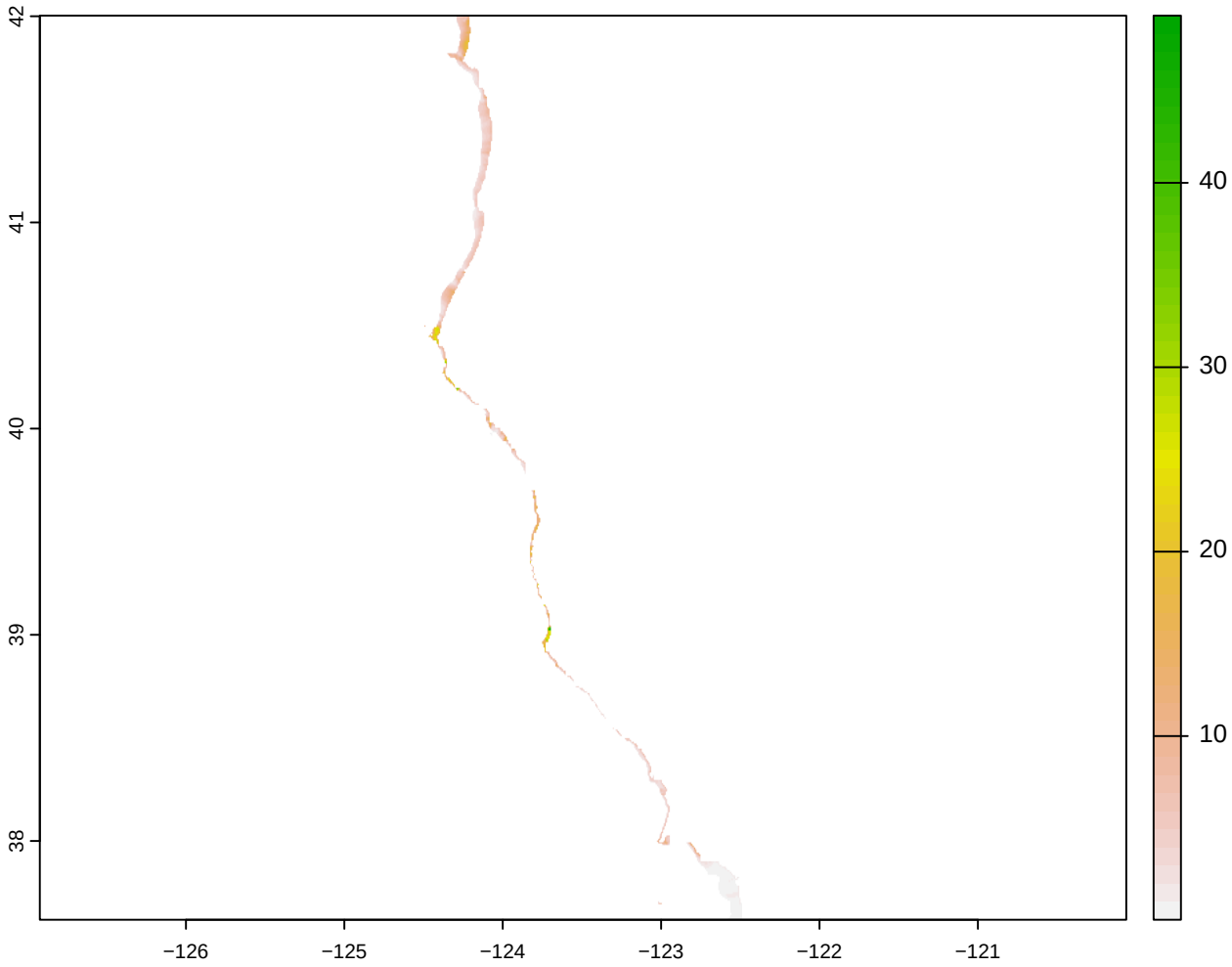

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.2203684, 2.0810645, 1.9472947, 1.8362591, 1.8489~
## $ Max_Monthly_Nitrate <dbl> 21.40735, 21.41267, 21.41452, 21.41403, 21.41354, ~
## $ wh_max              <dbl> 7.625381, 7.625381, 7.625381, 7.625381, 7.625381, ~
## $ log_UBR_Max         <dbl> 4.468370, 4.507249, 4.556448, 4.580810, 4.619103, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 2.962738, 3.139598, 3.363316, 3.473362, 3.655786, 4.105460, 5.3385~
```



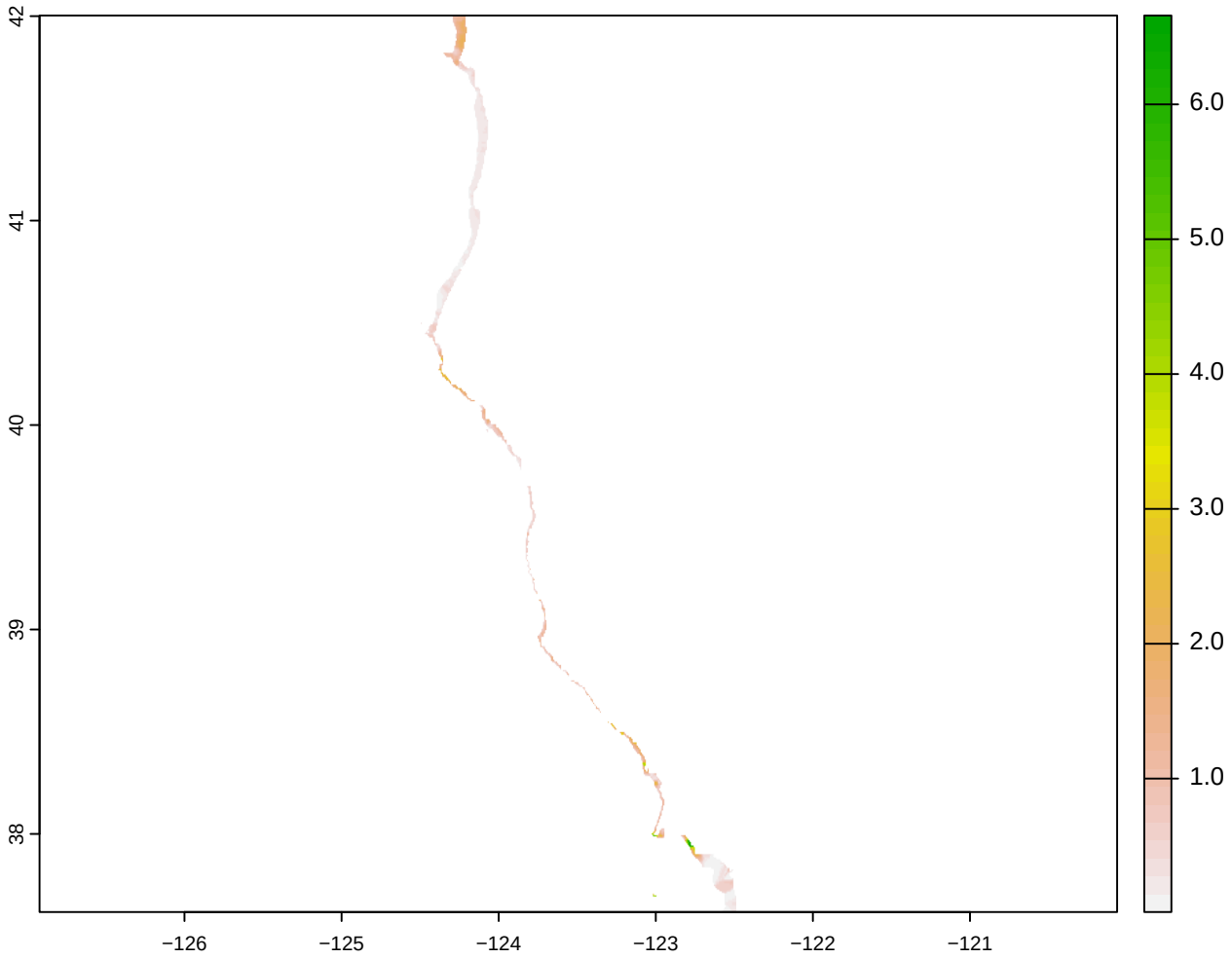

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.8991938, 2.7158563, 2.5291266, 2.3657858, 2.3625~
## $ Max_Monthly_Nitrate <dbl> 23.50820, 23.51453, 23.51774, 23.51884, 23.51994, ~
## $ wh_max              <dbl> 9.624915, 9.624915, 9.624915, 9.624915, 9.624915, ~
## $ log_UBR_Max         <dbl> 4.693219, 4.732087, 4.783358, 4.808147, 4.844074, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 13.66941, 14.53743, 15.60646, 16.14909, 16.82563, 18.21741, 22.265~
```



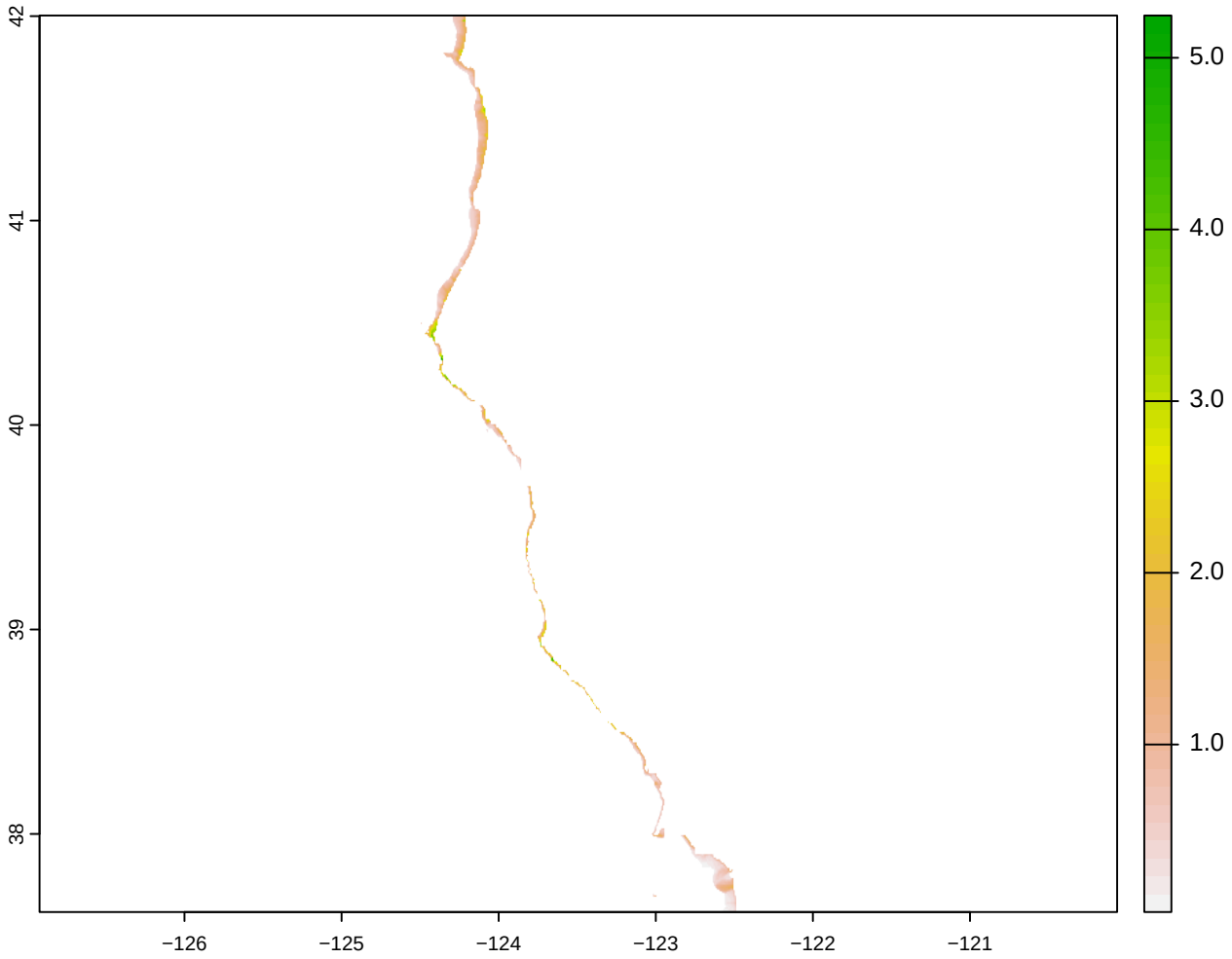

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.4332292, 2.2718120, 2.1140137, 1.9802941, 1.9805~
## $ Max_Monthly_Nitrate <dbl> 25.03871, 25.06623, 25.09579, 25.12674, 25.15768, ~
## $ wh_max              <dbl> 6.147168, 6.147168, 6.147168, 6.147168, 6.147168, ~
## $ log_UBR_Max         <dbl> 4.054003, 4.105019, 4.166689, 4.201367, 4.246289, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 2.888750, 3.166508, 3.523572, 3.757143, 4.066697, 4.821175, 7.2223~
```



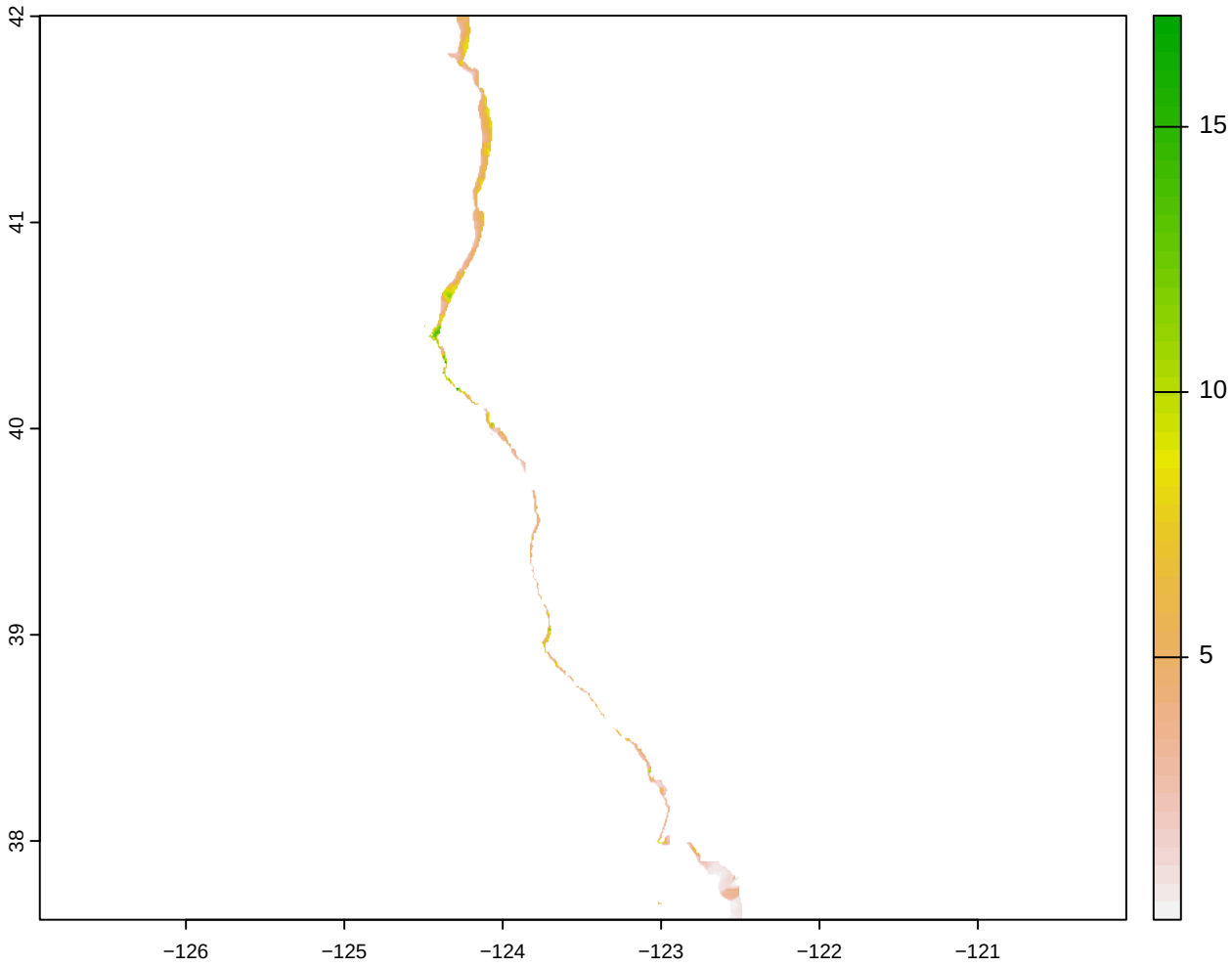

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.0601974, 1.9268007, 1.7990639, 1.6930345, 1.7012~
## $ Max_Monthly_Nitrate <dbl> 18.35394, 18.33418, 18.30607, 18.27232, 18.23856, ~
## $ wh_max              <dbl> 7.241228, 7.241228, 7.241228, 7.241228, 7.241228, ~
## $ log_UBR_Max         <dbl> 4.313672, 4.358007, 4.413838, 4.445281, 4.485267, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.6554790, 0.6936757, 0.7422494, 0.7633864, 0.7956447, 0.8953463, ~
```



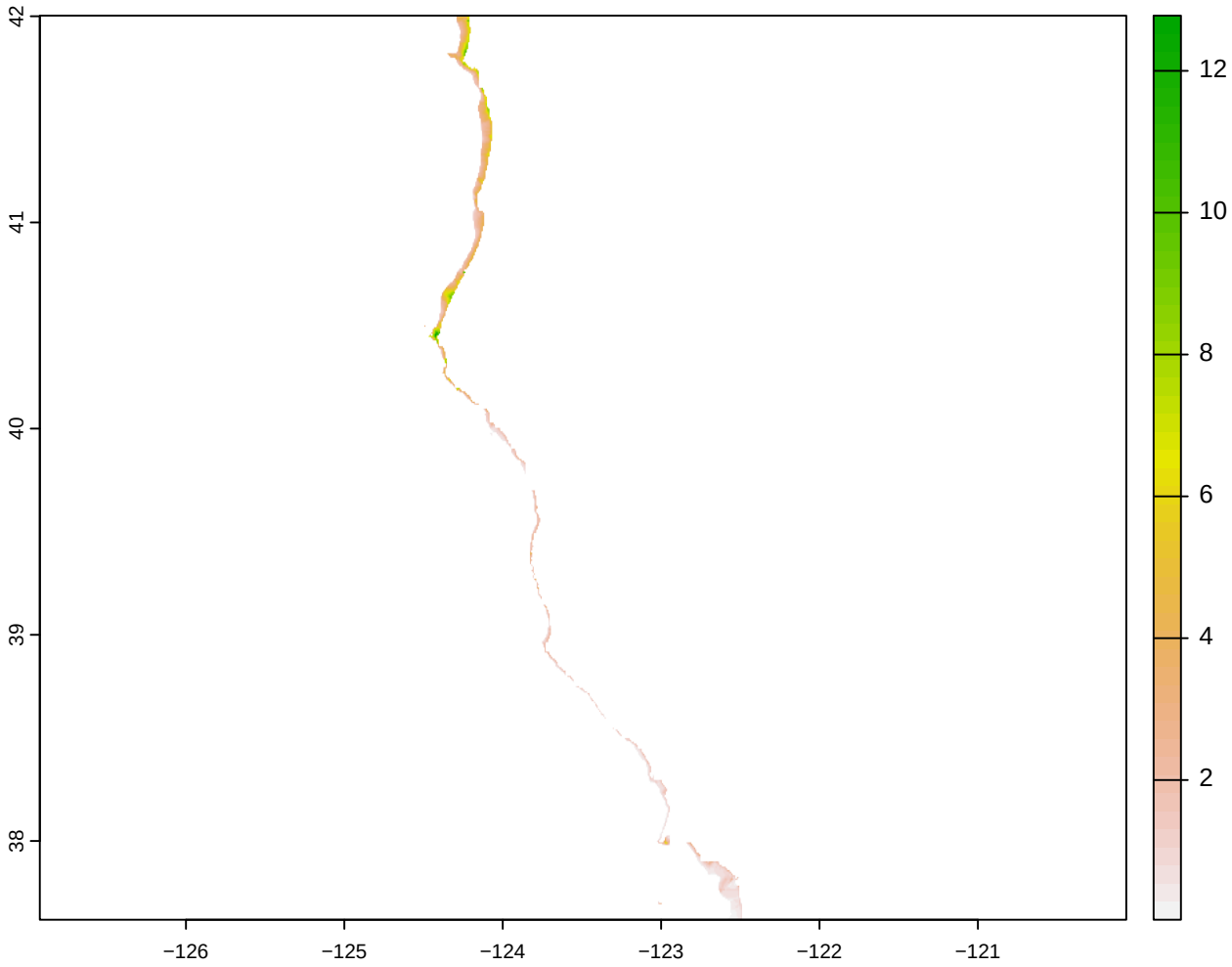

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.6841660, 2.5094998, 2.3399932, 2.1974931, 2.2034~
## $ Max_Monthly_Nitrate <dbl> 20.21718, 20.21745, 20.21789, 20.21845, 20.21901, ~
## $ wh_max              <dbl> 5.278422, 5.278422, 5.278422, 5.278422, 5.278422, ~
## $ log_UBR_Max         <dbl> 4.002067, 4.048822, 4.101814, 4.129939, 4.170015, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.3429097, 0.3712029, 0.4047024, 0.4236774, 0.4505480, 0.5173591, ~
```



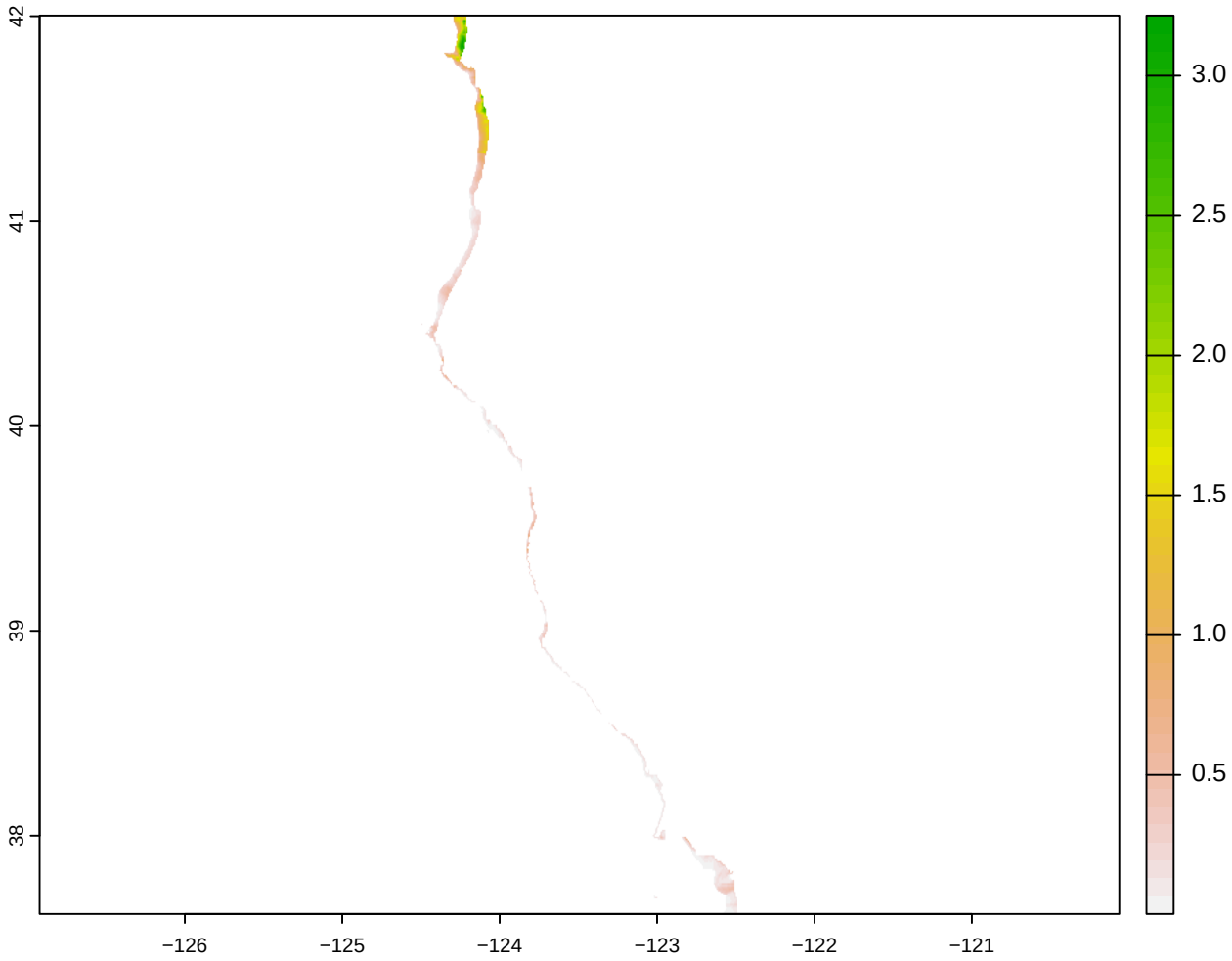

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 2.5592339, 2.3932226, 2.2292352, 2.0892174, 2.0904~
## $ Max_Monthly_Nitrate <dbl> 23.12979, 23.14237, 23.15684, 23.17256, 23.18829, ~
## $ wh_max              <dbl> 6.554802, 6.554802, 6.554802, 6.554802, 6.554802, ~
## $ log_UBR_Max         <dbl> 4.368648, 4.405648, 4.455777, 4.479410, 4.513760, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 2.893647, 3.085068, 3.344720, 3.483031, 3.675663, 4.131837, 5.5092~
```



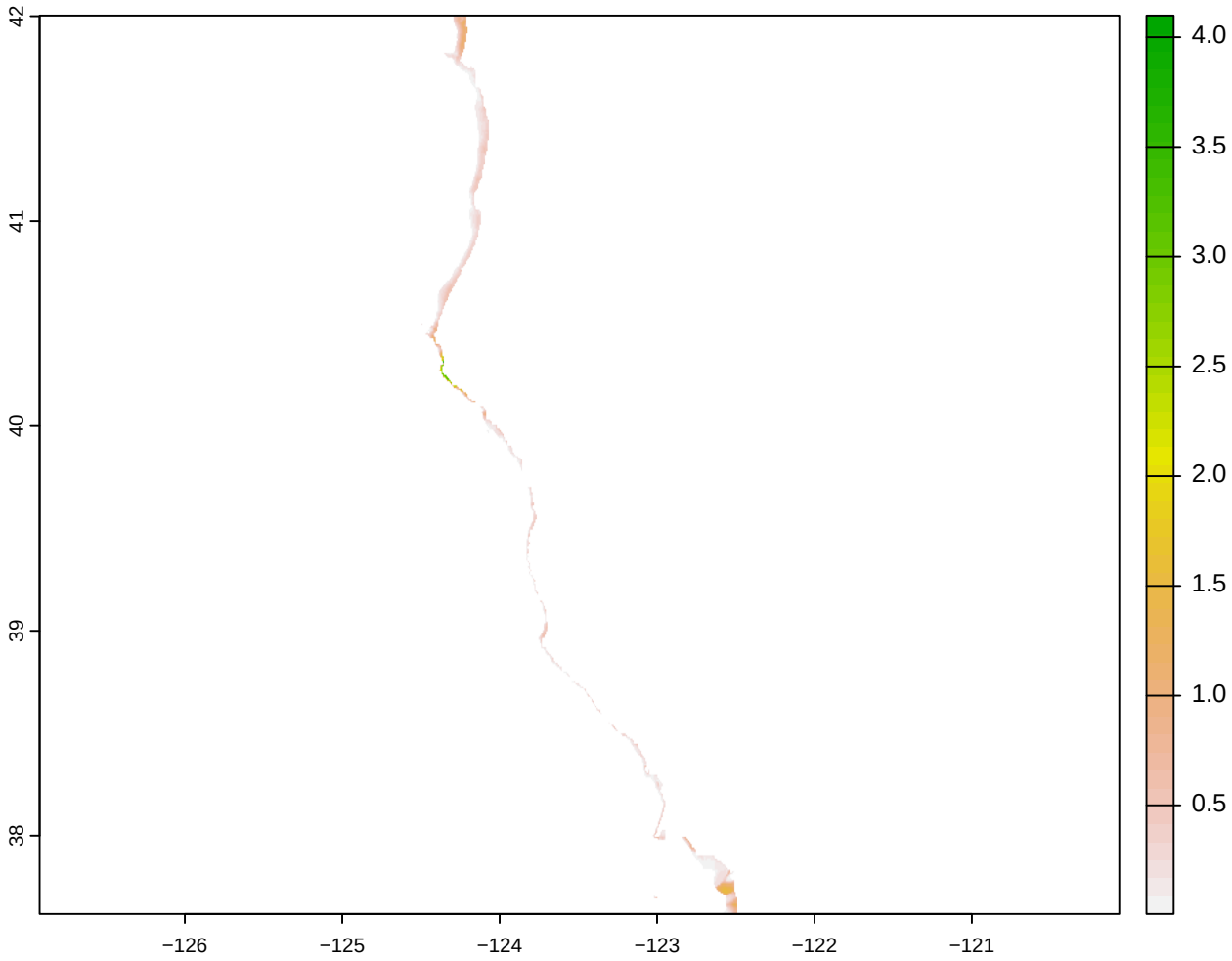

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 1.9137805, 1.7917955, 1.6704209, 1.5662628, 1.5679~
## $ Max_Monthly_Nitrate <dbl> 24.63958, 24.65014, 24.65831, 24.66486, 24.67142, ~
## $ wh_max              <dbl> 5.560881, 5.560881, 5.560881, 5.560881, 5.560881, ~
## $ log_UBR_Max         <dbl> 4.180888, 4.219778, 4.270270, 4.295372, 4.331035, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 2.299592, 2.446320, 2.643182, 2.744948, 2.902099, 3.338907, 4.6045~
```



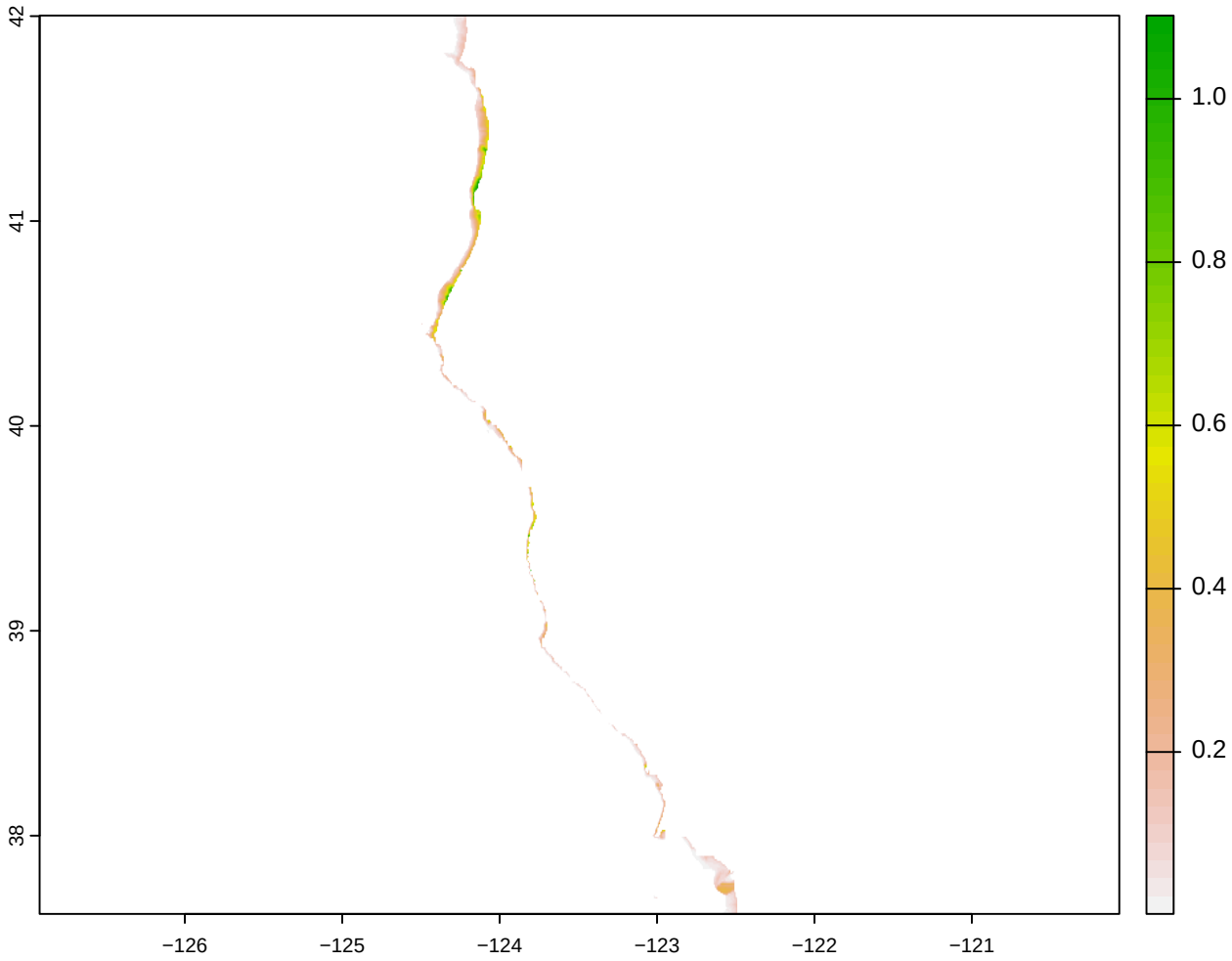

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 3.967592, 3.700086, 3.436109, 3.210230, 3.201737, ~
## $ Max_Monthly_Nitrate <dbl> 18.98434, 18.97235, 18.95427, 18.93205, 18.90983, ~
## $ wh_max              <dbl> 7.151553, 7.151553, 7.151553, 7.151553, 7.151553, ~
## $ log_UBR_Max         <dbl> 4.403312, 4.442105, 4.492454, 4.519158, 4.556305, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.8127533, 0.8944598, 0.9885701, 1.0421751, 1.0876524, 1.1324559, ~
```



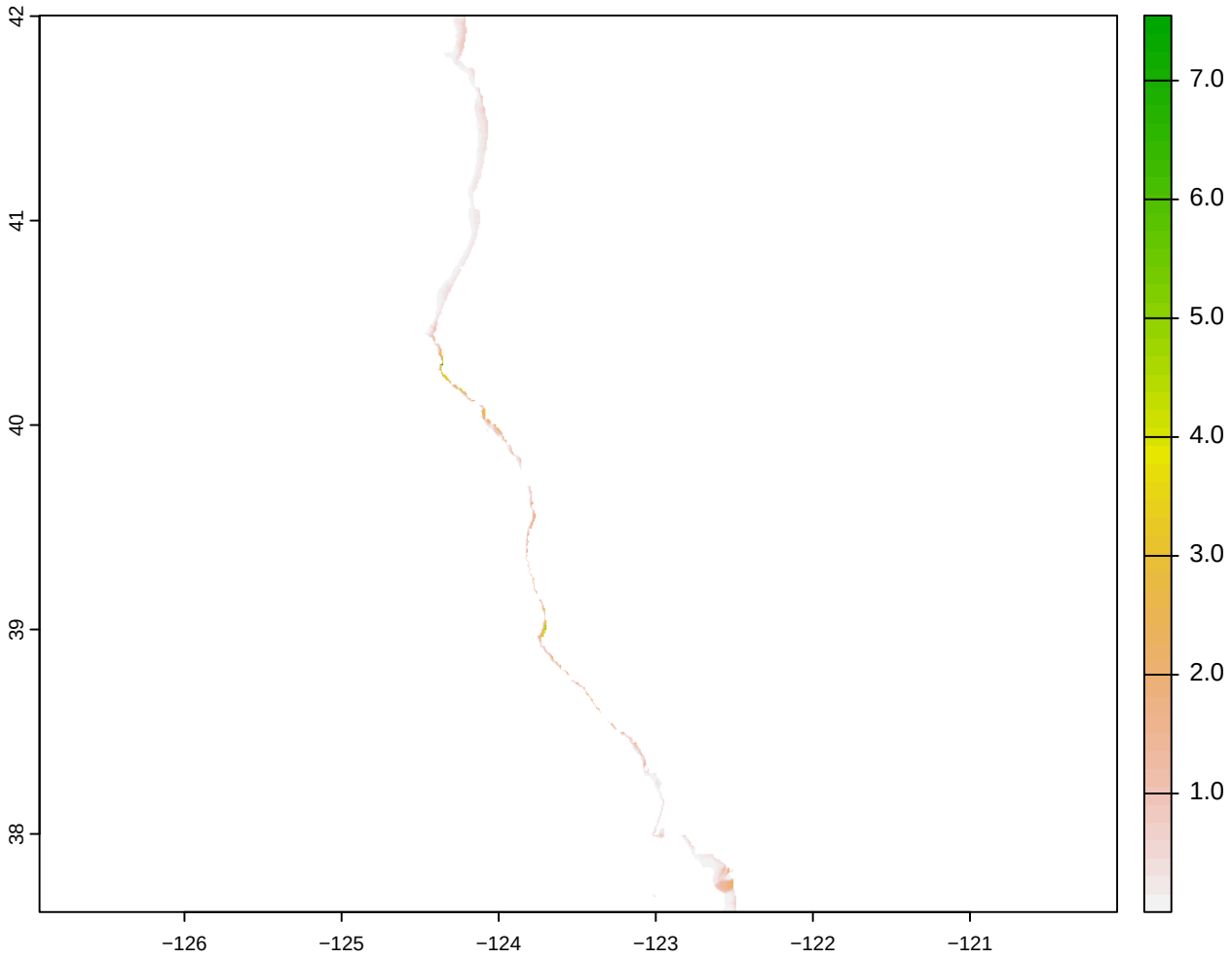

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 6.687811, 6.246604, 5.821805, 5.466880, 5.481265, ~
## $ Max_Monthly_Nitrate <dbl> 16.90357, 16.88388, 16.85292, 16.81433, 16.77575, ~
## $ wh_max              <dbl> 8.630524, 8.630524, 8.630524, 8.630524, 8.630524, ~
## $ log_UBR_Max         <dbl> 4.548507, 4.592100, 4.643646, 4.669160, 4.708314, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.2834324, 0.3351063, 0.4025213, 0.4590970, 0.4710557, 0.3833472, ~
```



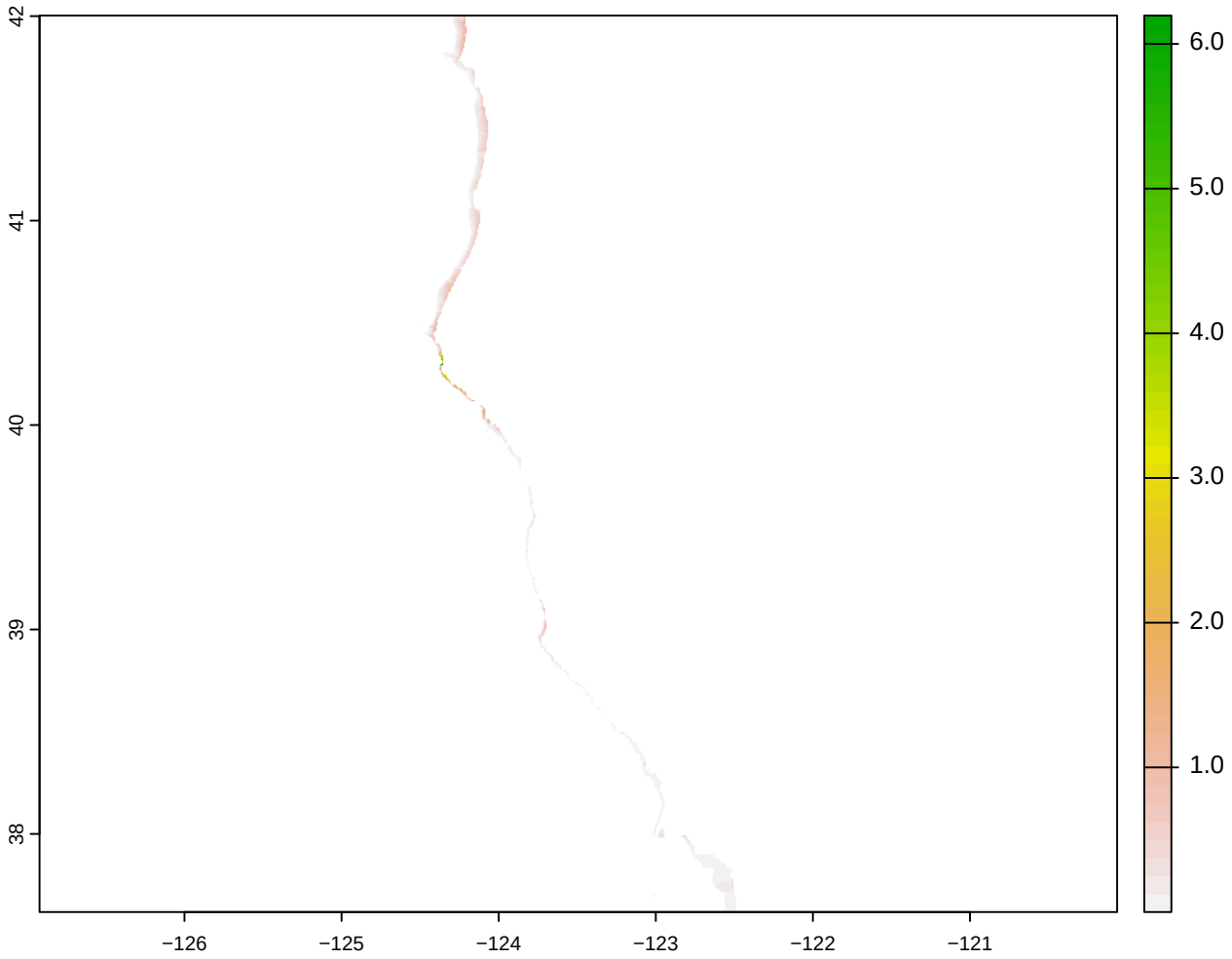

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 6.925117, 6.467491, 6.025046, 5.653919, 5.664871, ~
## $ Max_Monthly_Nitrate <dbl> 12.08996, 12.06946, 12.03194, 11.98290, 11.93386, ~
## $ wh_max              <dbl> 5.770716, 5.770716, 5.770716, 5.770716, 5.770716, ~
## $ log_UBR_Max         <dbl> 4.180888, 4.219778, 4.270270, 4.295372, 4.331035, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.01129444, 0.01345351, 0.01648298, 0.01929378, 0.02028382, 0.0173~
```



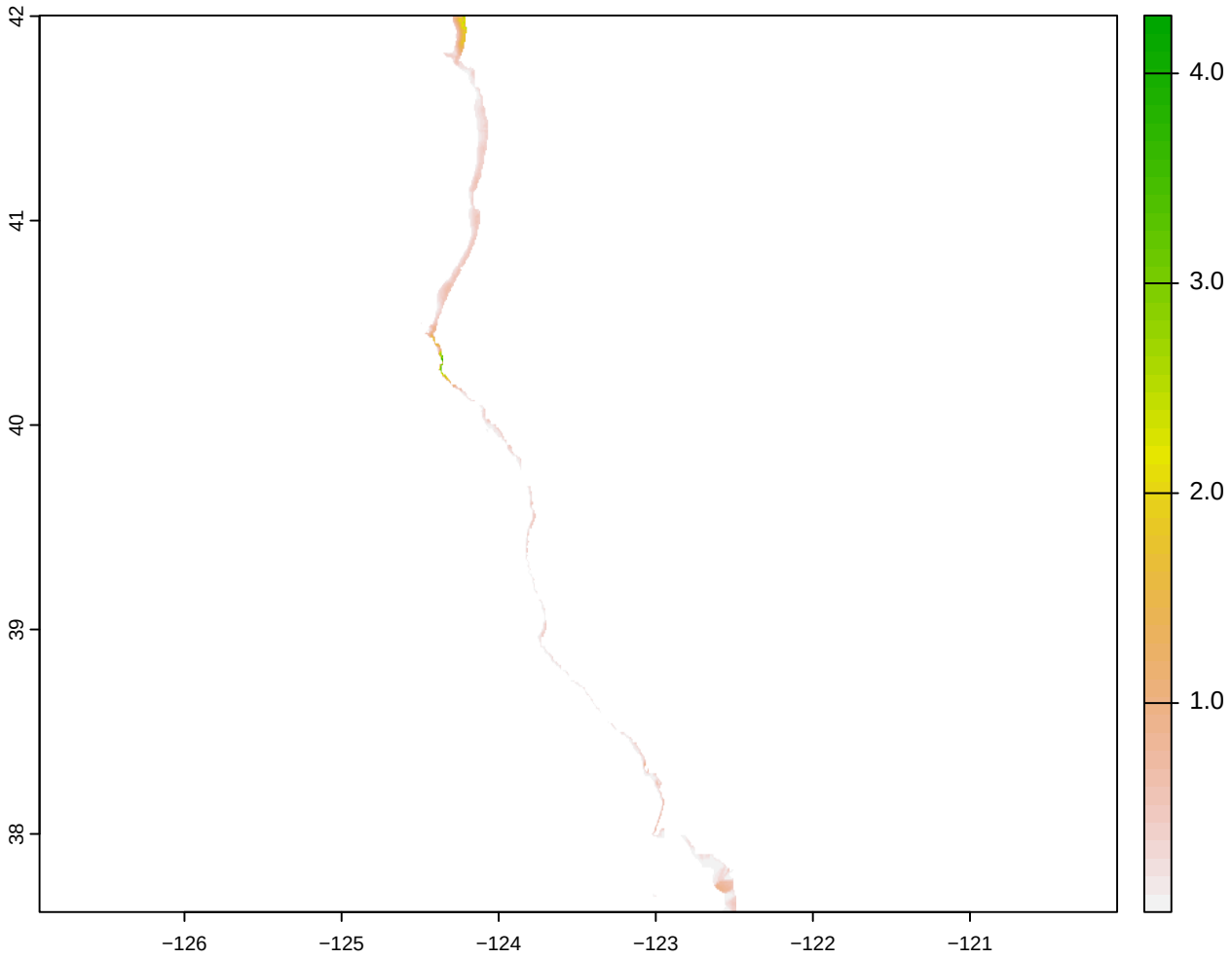

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 9.385114, 8.789563, 8.181453, 7.648340, 7.633313, ~
## $ Max_Monthly_Nitrate <dbl> 17.33315, 17.33548, 17.30701, 17.25769, 17.20837, ~
## $ wh_max              <dbl> 6.878233, 6.878233, 6.878233, 6.878233, 6.878233, ~
## $ log_UBR_Max         <dbl> 4.368648, 4.405648, 4.455777, 4.479410, 4.513760, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.07623093, 0.09287424, 0.11383885, 0.13053521, 0.13418349, 0.1080~
```



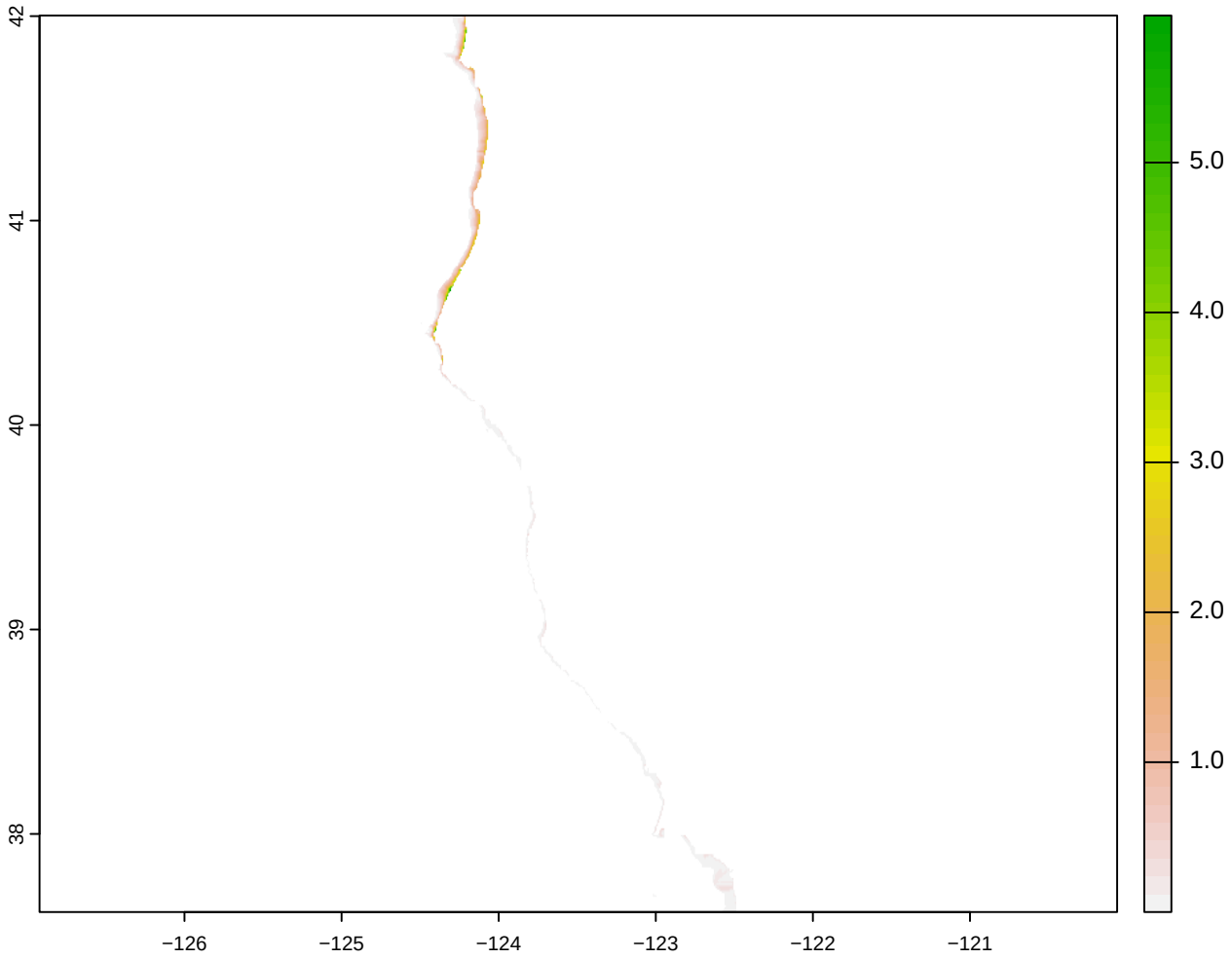

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 10.095670, 9.481309, 8.868011, 8.342246, 8.379427,~
## $ Max_Monthly_Nitrate <dbl> 17.44757, 17.41323, 17.34809, 17.26211, 17.17612, ~
## $ wh_max              <dbl> 7.654572, 7.654572, 7.654572, 7.654572, 7.654572, ~
## $ log_UBR_Max         <dbl> 4.468370, 4.507249, 4.556448, 4.580810, 4.619103, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.1078260, 0.1296841, 0.1556225, 0.1747838, 0.1747467, 0.1319062, ~
```



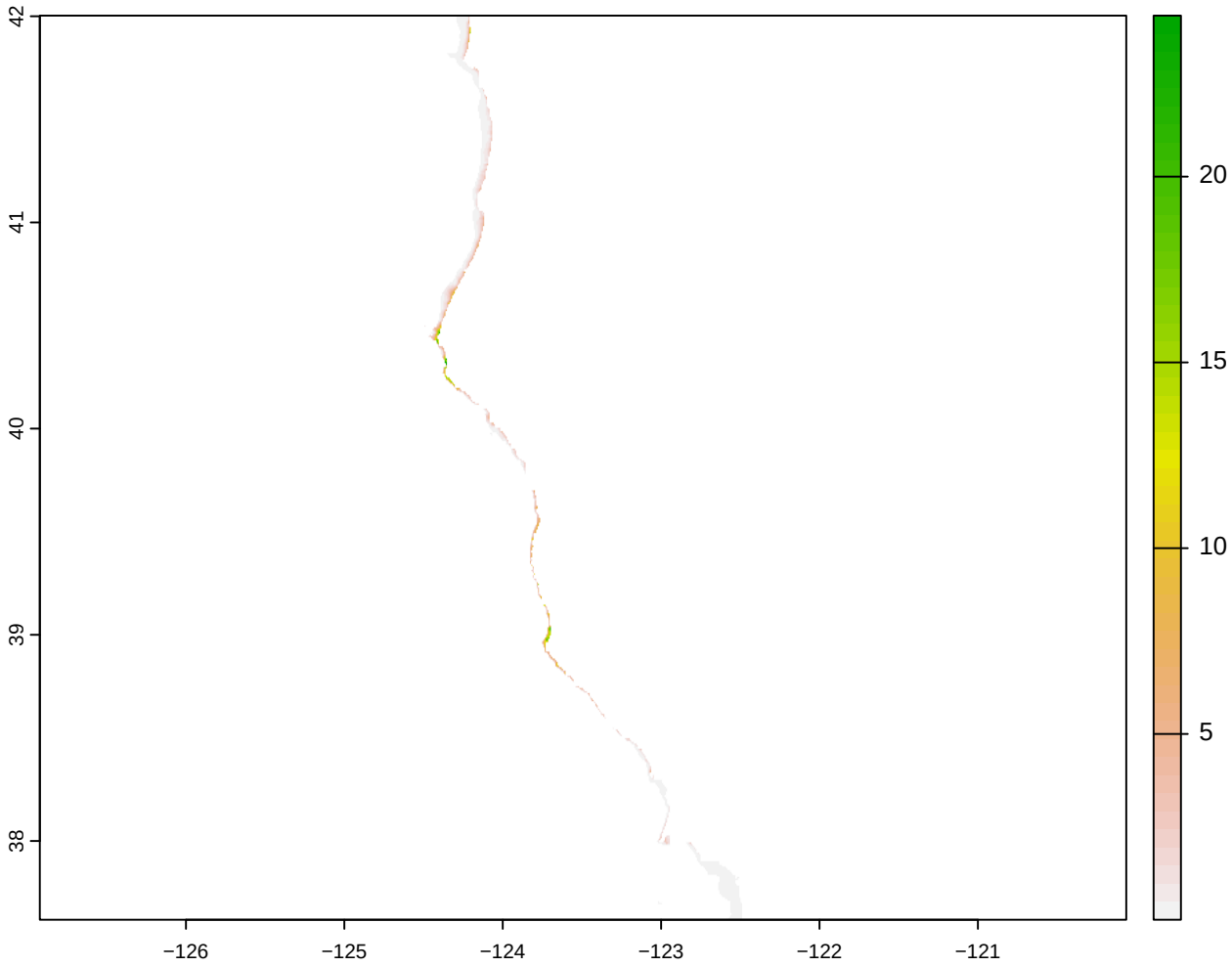

```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 7.600497, 7.159626, 6.720915, 6.346608, 6.399035, ~
## $ Max_Monthly_Nitrate <dbl> 18.80846, 18.75844, 18.68539, 18.59677, 18.50814, ~
## $ wh_max              <dbl> 8.463658, 8.463658, 8.463658, 8.463658, 8.463658, ~
## $ log_UBR_Max         <dbl> 4.489139, 4.530936, 4.585124, 4.610135, 4.647731, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.4959051, 0.5719198, 0.6638437, 0.7279437, 0.7243660, 0.5867601, ~
```




```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 21.484024, 19.948782, 18.420515, 17.098902, 16.941~
## $ Max_Monthly_Nitrate <dbl> 21.35366, 21.35780, 21.34906, 21.33161, 21.31415, ~
## $ wh_max              <dbl> 6.300334, 6.300334, 6.300334, 6.300334, 6.300334, ~
## $ log_UBR_Max         <dbl> 4.260127, 4.299006, 4.348894, 4.374132, 4.410054, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.01377183, 0.02109010, 0.03262116, 0.04614760, 0.05020716, 0.0295~
```




```
## Rows: 2,495
## Columns: 9
## $ x                   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2~
## $ y                   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, ~
## $ log_den_STRPURAD    <dbl> 34.472588, 32.056618, 29.662643, 27.602850, 27.419~
## $ Max_Monthly_Nitrate <dbl> 24.35116, 24.31230, 24.24364, 24.15481, 24.06597, ~
## $ wh_max              <dbl> 8.412910, 8.412910, 8.412910, 8.412910, 8.412910, ~
## $ log_UBR_Max         <dbl> 4.563695, 4.600451, 4.650694, 4.673129, 4.708073, ~
## $ year                <fct> 2004, 2004, 2004, 2004, 2004, 2004, 2004, 2004, 20~
## $ zone                <fct> OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, OUTER, O~
## $ site_name           <fct> 41.9988444785755, 41.9988444785755, 41.99884447857~
## Rows: 2,495
## Columns: 3
## $ x   <dbl> -124.2868, -124.2784, -124.2700, -124.2616, -124.2532, -124.2448, ~
## $ y   <dbl> 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.99884, 41.998~
## $ fit <dbl> 0.006206254, 0.011449411, 0.021139254, 0.034473698, 0.036481410, 0~
```


42
41
40
39
38
−126
−125
−124
−123
−122
−121
20
15
10
5