```r
library(tidync)
library(data.table)
library(tidyverse)
library(lubridate)
library(zyp)
library(sf)
library(raster)
library(terra)
library(RColorBrewer)
library(rasterVis)
library(xts)

proj_dir = "~/Dropbox/WB/GRACE_Ensemble/"

plot_trends <- function(R, horiz){

  #R = wb_region_list[2]

  region =
    wb_regions %>%
    filter(WB_REGION == R | FORMAL_EN == R) %>%
    st_transform(x, crs = st_crs(3857))

  if(nrow(region) > 0){

    bounds = st_bbox(region)
    # bounds[bounds > 180] = 179
    # bounds[bounds < -179] = 0

    region_trends_temp =
      gws_trends_spatial %>% st_transform(x, crs = st_crs(3857)) %>%
      st_within(st_make_valid(region))

    sel_logical = lengths(region_trends_temp) > 0

    region_trends =
      gws_trends_spatial[sel_logical, ] %>% st_transform(x, crs =
st_crs(3857)) %>%
      st_transform(x, crs = st_crs('+proj=longlat +datum=WGS84
+no_defs')) %>%
      st_cast("MULTIPOINT") %>%
      filter(!is.na(trends))# %>% st_shift_longitude()

    #Create a grid for interpolation
    region_out = region %>% st_transform(x, crs = st_crs(4326))

    bbox <- st_bbox(region_out) #%>% st_transform(x, crs =
st_crs(4326)))

    grd_template <- expand.grid(
```

```
      X = seq(from = bbox["xmin"], to = bbox["xmax"], by = 0.2),
      Y = seq(from = bbox["ymin"], to = bbox["ymax"], by = 0.2) # 20 m
resolution
    )

    #Projection for Raster
    crs_raster_format <- "+proj=longlat +datum=WGS84 +no_defs"

    #Create raster template
    grd_template_raster <- grd_template %>%
      dplyr::mutate(Z = 0) %>%
      raster::rasterFromXYZ(
        crs = crs_raster_format)

    # Inverse Distance Weighting
    fit_IDW <- gstat::gstat( # The setup here is quite similar to NN
      formula = trends ~ 1,
      data = as(region_trends, "Spatial"),
      nmax = 10, nmin = 3,
      set = list(idp = 0.5) # inverse distance power
    )

    interp_IDW <- interpolate(grd_template_raster, fit_IDW)

    ## crop and mask
    r2 <- crop(interp_IDW, extent(region_out))
    r3 <- terra::mask(r2, region %>% st_transform(x, crs =
st_crs(4326)))

    r3_df =
      as.data.frame(r3, xy = T) %>% rename(values = var1.pred) %>%
filter(!is.na(values)) %>%
      mutate(trends_cut = cut(values,
                              breaks = c(-Inf, -2, -1.5, -1, -0.5, 0,
0.5, 1, 1.5, 2, Inf))) %>%
      dplyr::select(-trends_cut) %>%
      mutate(values = ifelse(values < -2, -2, values)) %>%
      mutate(values = ifelse(values > 2, 2, values)) %>%
      rasterFromXYZ()


    color_pal =
c('#b2182b','#d6604d','#f4a582','#fddbc7','#d1e5f0','#92c5de','#4393c3
','#2166ac')

    out_file = paste0("Plots/GWS_",R,".png")
    out_title = paste('Groundwater Storage Trends in',
R,"(2002-2017)")

    # test =
```

```
    #   ggplot() +
    #   geom_sf(data = region_out, col = 'black') +
    #   geom_raster(r3_df, mapping = aes_string(x = 'x', y = 'y', fill
= 'values')) +
    #   theme_bw() +
    #   coord_sf(datum = st_crs(region_out)) +
    #


    png(file= out_file,
        width=1000, height=800)

    #pe <- as.polygons(ext(r3_df), crs = 'WGS84')

    plot(r3_df, legend=FALSE,
         breaks = c(-2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2),
         col = color_pal)

    title(main = out_title, line = 1, adj = 0.45, cex.main=2)


    if(horiz == T) {
      loc.1 = c(0.07, 0.34, 0.2, 0.24) #Change these values to adjust
the legend position
    } else if (horiz == F) {
      loc.1 = c(0.2, 0.24, 0.18, 0.45)
    }

    if(R %like% 'Yemen'){
      loc.1 = c(0.27, 0.54, 0.2, 0.24) #Change these values to adjust
the legend position
    }

    plot(r3_df, legend.only = T, smallplot = loc.1, horizontal =
horiz,
         breaks = c(-2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2), lab.breaks
= c('< -2', -1.5, -1, -0.5, 0, 0.5, 1, 1.5, ">2"),
         col = color_pal, legend.args = list(text = 'Groundwater
Storage\n Trends (cm/yr)', side = 3,
                                             font = 2, line = 1, cex =
1.5))

    region = st_transform(region, crs = st_crs(4326))
    plot(region$geometry, add = T, lwd = 1)

    dev.off()
  }
}

#Load the datasets
```

```
#The GRACE Mascon dataset
tws_mascon =
  tidync(paste0(proj_dir,'GRACE_data/JPL_Mascons/
GRCTellus.JPL.200204_202106.GLO.RL06M.MSCNv02CRI.nc')) %>%
  activate("D0,D1,D2") %>%
  hyper_tibble() %>% as.data.table() %>%
  .[order(lat, lon)] %>% #ensures the order is the same
  group_by(lat,lon) %>%
  mutate(cell_id = cur_group_id())  %>% as.data.table()

#Locations of the mascons
mascons =
  tidync('~/Dropbox/WB/GRACE_Analysis/JPL_MSCNv02_PLACEMENT.nc') %>%
  activate("D1") %>%
  hyper_tibble() %>% as.data.table()

#Adjustment factors
clm_factors =
  tidync('~/Dropbox/WB/GRACE_Analysis/
CLM4.SCALE_FACTOR.JPL.MSCNv02CRI.nc') %>%
  activate("D0,D1") %>%
  hyper_tibble() %>% as.data.table()

#Land Boundary
land_filter =
  tidync('~/Dropbox/WB/GRACE_Analysis/LAND_MASK.CRI.nc') %>%
  activate("D0,D1") %>%
  hyper_tibble() %>% as.data.table()

#Dates
dates1 =
  tws_mascon[,.(time)]%>% distinct() %>%
  mutate(dates = as.Date(time, origin = '2002-01-01')) %>%
  mutate(ym = substr(ymd(dates), 1, 7))

dates1 = dates1[!duplicated(dates1$ym)]

########################
######### Analysis
########################
#Add dates to the TWS df; also scale the values
tws_mascon =
  tws_mascon %>% merge(dates1[,.(ym, time)], by = 'time') %>%
  merge(clm_factors, by = c("lat", "lon")) %>% merge(land_filter, by =
c("lat", "lon")) %>%
  .[, lwe_corr := scale_factor*lwe_thickness]

#Convert from long to wide format
tws_lwe =
```

```
  tws_mascon[,.(lwe_corr, lon, lat, cell_id, ym)] %>%
  dcast(lon + lat + cell_id ~ ym, value.var = "lwe_corr")

#Estimate the new baseline (2002-2020)
bsl_colnum = 164 #199 = 2020; 165 = 2012; 118 = 2012
mean_tws = apply(tws_lwe[, 4:bsl_colnum], 1, mean)

#Estimate the new baseline
sd_tws = apply(tws_lwe[, 4:bsl_colnum], 1, sd)

#Remove the new baseline (currently not using data from GRACE-FO)
tws_lwe_base =
  tws_lwe[, 4:199] - mean_tws

#Estimate the new baseline
#sd_tws2 = apply(tws_lwe_base, 1, sd)

tws_lwe_base =
  cbind(tws_lwe[,1:3], tws_lwe_base) %>%
  mutate(lon = ifelse(lon>180, lon-360, lon))

# fwrite(tws_lwe_base,
#        paste0(proj_dir, "GRACE_Data/JPL_Mascons/
GRACE_JPL_TWS_220502.csv"))

# test =
#   tidync('/Users/tejasvi/Dropbox/Mac/Downloads/GLDAS/
GLDAS_NOAH025_M.A200309.021.nc4.SUB.nc4')


#Use the methodology developed by Shamsuddhua and Taylor (2020)

# For accumulated variables such as Qs_acc, the monthly mean surface
runoff is the
# average 3-hour accumulation over all 3-hour intervals in April 1979.
To compute
# monthly accumulation, use this formula:
#
#   Qs_acc (April){kg/m2} = Qs_acc (April){kg/m2/3hr} * 8{3hr/day} *
30{days}


gldas_noah =
  tidync(paste0(proj_dir, 'GLDAS_data/NOAH/gldas_noah_merge.nc4')) %>%
  hyper_tibble() %>% as.data.table() %>%
  .[,Qs_month := Qs_acc * 8 * 30] %>% #Convert accumulated runoff into
monthly runoff (see formula above)
  .[, total_water_without_runoff :=
      (SWE_inst + SoilMoi0_10cm_inst + SoilMoi10_40cm_inst +
      SoilMoi40_100cm_inst + SoilMoi100_200cm_inst + CanopInt_inst)/
```

```
10] %>%
  .[, total_water :=
      (Qs_month + SWE_inst + SoilMoi0_10cm_inst + SoilMoi10_40cm_inst
+
      SoilMoi40_100cm_inst + SoilMoi100_200cm_inst + CanopInt_inst)/
10]
# convert mass to volume: vol = kg / (1000 kg/m3) [Density of water]
-> vol = (1/1000) m3
# Therefore: kg/m2 -> (1/1000) [m3/m2] -> (1/10) [cm]

gldas_dates =
  copy(gldas_noah[,.(time)]) %>% distinct() %>%
  .[,ym := substr(as.Date(time, origin = '2000-01-01'), 1, 7)]

gldas_noah =
  gldas_noah %>% merge(gldas_dates, by = 'time') %>%
  .[ym %in% dates1$ym]

##########Without runoff
# gldas_noah_spread =
#   gldas_noah[,.(total_water_without_runoff, lon, lat, ym)] %>%
#   dcast(lon + lat ~ ym, value.var = "total_water_without_runoff")
#
# #Estimate the new baseline land water content (2002-2020)
# mean_gldas = apply(gldas_noah_spread[, 3:198], 1, mean)
#
# #Remove the new baseline (currently not using data from GRACE-FO)
# gldas_lwc_base =
#   gldas_noah_spread[, 3:198] - mean_gldas
#
# gldas_noah_base_rem =
#   cbind(gldas_noah_spread[,1:2], gldas_lwc_base) %>%
#   merge(tws_lwe_base[,.(lon,lat,cell_id)], by = c("lat", "lon"),
all.x = F)
#
# #fwrite(gldas_noah_base_rem,
"GLDAS_2002_2017_SWS_without_runoff.csv")

###########With runoff
gldas_noah_spread =
  gldas_noah[,.(total_water, lon, lat, ym)] %>%
  dcast(lon + lat ~ ym, value.var = "total_water")

#Estimate the new baseline land water content
mean_gldas = apply(gldas_noah_spread[, 3:(bsl_colnum-1)], 1, mean)

#Remove the new baseline (currently not using data from GRACE-FO)
gldas_lwc_base =
  gldas_noah_spread[, 3:198] - mean_gldas
```

```
gldas_noah_base_rem =
  cbind(gldas_noah_spread[,1:2], gldas_lwc_base) %>%
  merge(tws_lwe_base[,.(lon,lat,cell_id)], by = c("lat", "lon"), all.x
= F)

# fwrite(gldas_noah_base_rem,
#        paste0(proj_dir, 'GLDAS_data/NOAH/
GLDAS_2002_2017_SWS_with_runoff_220508.csv'))

###Create Groundwater Storage Anomaly

sws_anomaly =
  gldas_noah_base_rem %>% .[order(lon,lat)]

tws_anomaly =
  tws_lwe_base[cell_id %in% sws_anomaly$cell_id] %>% .[order(lon,lat)]


#Checks to ensure the datasets align
identical(sws_anomaly$cell_id, tws_anomaly$cell_id)
identical(colnames(sws_anomaly)[3:193], colnames(tws_anomaly)[4:194])
identical(dim(tws_anomaly[ ,4:194]), dim(sws_anomaly[ ,3:193]))

gws_anomaly =
  tws_anomaly[ ,4:194] - sws_anomaly[ ,3:193]

gws_anomaly = cbind(tws_anomaly[,1:3], gws_anomaly)

#WRITE OUTPUT CSV
fwrite(sws_anomaly,
       paste0(proj_dir, "GRACE_Data/JPL_Mascons/
GLDAS_NOAH_02_20_SWS_wRunoff_BSL2017.csv"))

fwrite(tws_anomaly,
       paste0(proj_dir, "GRACE_Data/JPL_Mascons/
GRACE_TWS_scaled_BSL2017.csv"))

fwrite(gws_anomaly,
       paste0(proj_dir, "GRACE_Data/JPL_Mascons/
GRACE_GWS_2002_2020_wRunoff_BSL2017.csv"))


####Trend Analysis

#Estimate the trends using Mann-kendall
tws_trends =
  apply(tws_lwe_base[, 3:ncol(tws_lwe_base)], 1, zyp.yuepilon)

tws_trends_estimates =
  tws_lwe[,1:3] %>%
```

```
  mutate(trends = tws_trends[2,], sig = tws_trends[6,]) %>%
  mutate(lon2 = ifelse(lon>180, lon-360, lon))

#fwrite(tws_trends_estimates, "GRACE_TWS_trends.csv")

#Convert monthly to annual
gws_annual =
  gws_anomaly %>% melt(id.vars = c("lon", "lat", "cell_id"),
                       measure.vars = 4:ncol(gws_anomaly),
                       variable.name = "ym", value.name = "gws") %>%
  .[, year := substr(ym, 1,4)] %>%
  .[, an_mean := mean(gws), .(lat, lon, year)] %>%
  .[,.(lat, lon, cell_id, year, an_mean)] %>% distinct() %>%
  dcast(lon + lat + cell_id ~ year, value.var = "an_mean") %>% .
[order(cell_id)]

gws_trends =
  apply(gws_annual[, 4:ncol(gws_annual)], 1, zyp.yuepilon)

gws_trends_estimates =
  gws_annual[,1:3] %>%
  mutate(trends = gws_trends[2,], sig = gws_trends[6,]) %>%
  mutate(lon = ifelse(lon>180, lon-360, lon))

#fwrite(gws_trends_estimates, "Output/
GRACE_GWS_annual_trends_220203.csv")

test = fread(paste0(proj_dir, "GRACE_Data/JPL_Mascons/
GLDAS_NOAH_02_20_SWS_wRunoff_220508.csv"))
```