Comment on 'Legacy nitrogen may prevent achievement of water quallity goals in the Gulf of Mexico'

Tristan Ballard (tballard@stanford.edu)

3/07/2019

First read in packages used later for plotting. If they are not installed, run install.packages('packageName').

```
library(ggplot2); library(lattice); library(dplyr); library(raster); library(sf); library(lwgeom); libra
#setwd('~/Desktop/')
```

Read in observed and Van Meter predicted MRB loading datasets and convert to consistent units.

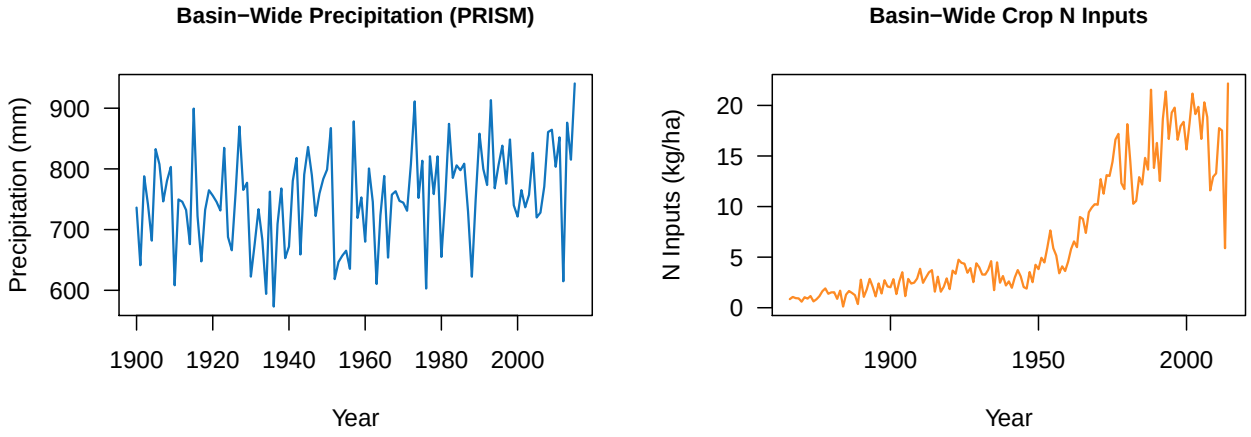
```
## 1955-2014 loads (tons) from https://cfpub.epa.gov/roe/indicator.cfm?i=33
mrb = c(473993.6096,332897.8374,475095.9203,406752.6557,
  367069.4698,356046.3626,302033.1373,304237.7587,192904.376,
  262349.9514,390217.9949,224871.3869,251326.8442,296521.5837,
  385808.752,408957.2771,392422.6163,607373.2067,865313.9152,
  833346.9043,873030.0902,545643.8064,616191.6925,855393.1187,
  1223564.899,872020,693720,1125900,1512600,1317520,971200,
  1095510,991670,616750,704730,1108400,1221200,991400,1792500,
  995500,931600,911290,1132300,1143040,1214720,591390,963100,
  878610,727310,937150,940650,662700,964400,1330090,1062860,
  1222200,1201630,622620,997320,801490)

## Convert from tons to kg/ha
## MRB area is 291451362.1ha, and .00110231 converts tons to kg
mrbKG = mrb / 291451362.1  / .00110231
mrb.dat = data.frame(year=1955:2014, mrb=mrbKG, observedLoadKtons = mrb/1000)

## Read in predicted Science fluxes
vm.pred = read.csv('./VM_predicted.csv', head=T) #kilotons
vm.pred$load = vm.pred$load / 291451362.1  / .00110231 * 1000 #kg/ha
mrb2 = left_join(mrb.dat, vm.pred)

#Calculate R2 values of Van Meter et al. model, 1955-2014
#cor(mrb2$mrb, mrb2$load)^2 #R2=0.6722
```

```
## Read in land use predictors from GBC
gbc.dat = read.csv('n_surplus.csv', head=T)

## convert from kg/ha of cropland to kg/ha of MRB by multiplying by fraction of MRB covered in cropland
crop.surplus = gbc.dat$crop_n * gbc.dat$frac_crop
surplus.df = data.frame(year=gbc.dat$year, crop.surplus)

## Note that if you do not scale the crop N term, the resulting R2 values are still very similar and co
```

```
## Read in basin-wide annual precipitation from PRISM dataset
precip.df=readRDS('precip.df.RDS') # [mm]
```

Plot the precipitation and Crop N surplus term

```
par(mfrow=c(1,2))
plot(precip.df$year, precip.df$precip, type='l', ylab='Precipitation (mm)', xlab='Year', las=1, col='#1
plot(surplus.df$year, surplus.df$crop.surplus, type='l', ylab='N Inputs (kg/ha)', xlab='Year', las=1, c
```



Read in actual pigment data from Rabalais et al. (MPB 2004). I also verify the R2 values they provide for their model vs. the pigments. I recreate their R2 values very nearly for the 1960-1997 period (noting that the N load estimates I have are from the GBC paper, so may be slightly different than those used in the updated Science paper) but are a bit lower than reported in the 1820-1930 period.

```
## Read in pigment data
pigment = read_excel('D50zeaBcaro.xlsx')
pigment$yearCentered = round(pigment$yearCentered)
pigment = data.frame(apply(pigment, 2, rev)) ## Order rows by increasing year
pigment = right_join(pigment, data.frame(yearCentered = 1773:2012)) ## add rows of NAs for missing years

## Read in BSi sediment core data from Turner, Milan, and Rabalais 'A retrospective analysis of trace m
BSie30 = read_csv('BSiE30.csv')
BSie30$year = round(BSie30$year)
names(BSie30) = c('year', 'BSie30')

## Check the Van Meter correlation numbers:
## 1820-1930
# yr.start = 1820; yr.end = 1930
# cor(pigment$zea[pigment$yearCentered %in% yr.start:yr.end], vm.pred$load[vm.pred$year %in% yr.start:y
# cor(pigment$bcaro[pigment$yearCentered %in% yr.start:yr.end], vm.pred$load[vm.pred$year %in% yr.start
# ## 1960-1997 (they claim 1960-2000, except no pigment data past 1997...)
# yr.start = 1960; yr.end = 2000
```

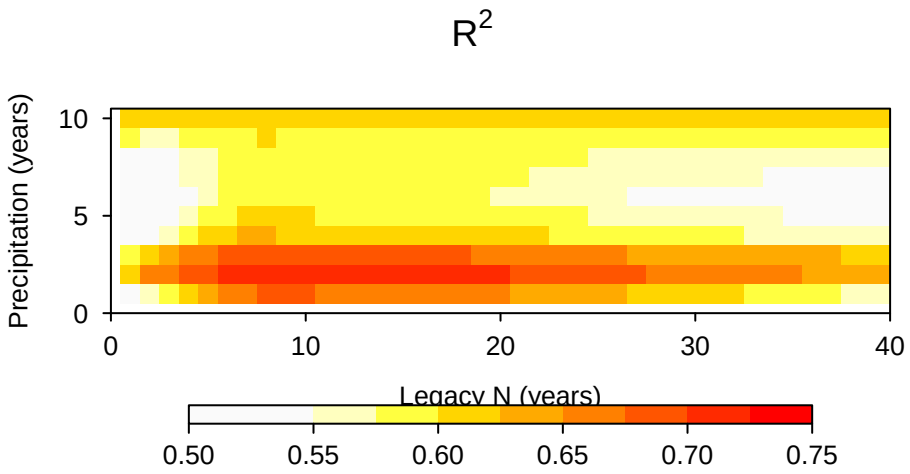
```
# cor(pigment$zea[pigment$yearCentered %in% yr.start:yr.end], vm.pred$load[vm.pred$year %in% yr.start:y
# cor(pigment$bcaro[pigment$yearCentered %in% yr.start:yr.end], vm.pred$load[vm.pred$year %in% yr.start
```

Figure 1A

```
### Two-variable models with cropN and Precip estimating observed loads
n.lag.crop = 40
n.lag.precip = 10
two.way.R2<- two.way.R2.bcaro<- two.way.R2.zea <- matrix(rep(NA,n.lag.crop*n.lag.precip), nrow=n.lag.cr
n.obs = length(mrb.dat$mrb)
for (i in 1:n.lag.crop){
  ## Loop through n surplus variables
  surplus.loop = rep(NA, n.obs)
    for (u in 1:n.obs){
      surplus.loop[u] = sum(surplus.df$crop.surplus[c(89+u-i+1):c(89+u)])
    }
  for (j in 1:n.lag.precip){
    ## Loop through precipitaion variables
    precip.loop = rep(NA, n.obs)
    for (v in 1:n.obs){
      precip.loop[v] = sum(precip.df$precip[c(55+v-j+1):c(55+v)])
    }
    ## Fit the regression and extract model R2
    fit=lm(mrb.dat$mrb ~ surplus.loop + precip.loop)
    two.way.R2[i,j] = cor(fit$fitted.values, mrb.dat$mrb)^2
   # two.way.R2.bcaro[i,j] = cor(fit$fitted.values[6:41], pigment$bcaro)^2
  #  two.way.R2.zea[i,j] = cor(fit$fitted.values[6:41], pigment$zea)^2
  }
}
max(two.way.R2) #highest R2 value from 2-variable models
```

```
## [1] 0.7221609
```

```
## Plot the 2-way R2 values
twoWayPlot = function(two.way.plot, min.plot, max.plot, n.col){
  two.way.plot[two.way.plot<min.plot]=min.plot
  levelplot(two.way.plot, col.regions = c('gray98','gray98',rev(heat.colors(n.col-1))[-3]), xlab=list("
      at=c(seq(min.plot,max.plot,length.out=n.col+1)),xlim=c(0,40), ylim=c(0,10.5),
      colorkey = list(space='bottom', tck=.8, width=.8, height=.8),
      scales = list(x=list(at=c(0,10,20,30,40), labels=c(0,10,20,30,40)), y=list(at=c(0,5,10), labels=c
}
twoWayPlot(two.way.R2, min.plot=0.5, max.plot=0.75, n.col=10)
```

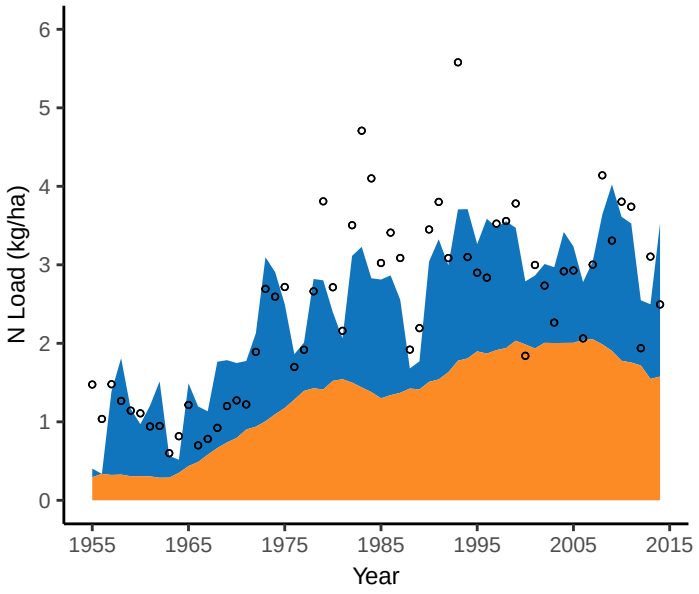



```
#twoWayPlot(two.way.R2.bcaro, min.plot=.55, max.plot=0.82, n.col=10)
#twoWayPlot(two.way.R2.zea, min.plot=.55, max.plot=0.8, n.col=10)
```

Figure 1B

```
i = 8 #Cumulative crop N # of years
j = 2 #Cumulative precipitaiton # of years

idx.start = which(precip.df$year == 1954)
## Set up desired predictors for 2-variable model
surplus.loop = rep(NA, n.obs)
  for (u in 1:n.obs){
    surplus.loop[u] = sum(surplus.df$crop.surplus[c(89+u-i+1):c(89+u)])
  }
precip.loop = rep(NA, n.obs)
  for (v in 1:n.obs){
    precip.loop[v] = sum(precip.df$precip[c(idx.start+v-j+1):c(idx.start+v)])
  }

## Fit the highest R2 model
#fit = lm(mrb.dat$mrb ~ surplus.loop + precip.loop)
#summary(fit)

## Minimum deviate precipitation for visualization
precip.md=precip.loop-min(precip.loop)
fit = lm(mrb.dat$mrb ~ surplus.loop + precip.md)
#summary(fit) #minimum deviating only changes intercept

## Make stacked time series
bottom = coef(fit)[1] + coef(fit)[2]*surplus.loop
top = coef(fit)[3]*precip.md

## Set up dataframes for plotting in ggplot
dat.plot = data.frame(pred=c(bottom,top), sector=c(rep('z', length(bottom)), rep('a', length(top))), Ye

#pdf('~/Desktop/stacked.crop.pdf', width=3.3, height=2.78, useDingbats=F)
ggplot(data=dat.plot, aes(x=Year, y=pred, fill=sector)) +
```

```
    geom_area() +
    geom_point(data=dat.plot, aes(x=Year, y=load), size=.9, stroke=.4, shape=1) +
    guides(fill=F) +
    labs(y='N Load (kg/ha)') +
    scale_fill_manual(values=c('#1175BE','#FD8B25')) +
    scale_x_continuous(breaks=seq(1955, 2015, by=10)) +
    scale_y_continuous(limits=c(0, 6), breaks=seq(0, 6, by=1)) +
    theme(axis.title=element_text(size=9),axis.text.x=element_text(size=7.5), axis.text.y=element_text(s
      panel.background = element_blank(), axis.line = element_line(colour = "black"))
```




```
#dev.off()
```

Figure 2

```
## Set up dataframe for plotting with units consistent with Van Meter et al (2018)
fig2.pred = read.csv('VM_predicted.csv', head=T) #ktons
fig2.df = left_join(fig2.pred, surplus.df)
```

```
## Joining, by = "year"
```

```
n = nrow(fig2.df)
## Add in the cumulative N terms
surplus.loop = rep(NA, n)
  i = 8
  for (u in i:n){
    surplus.loop[u] = sum(fig2.df$crop.surplus[c(u-i+1):c(u)])
  }
fig2.df = data.frame(fig2.df, cropN.cum8=surplus.loop)
fig2.df$crop.surplus = fig2.df$crop.surplus * 291451362.1  * .00110231 / 1000 #convert to ktons
fig2.df$cropN.cum8 = fig2.df$cropN.cum8 * 291451362.1  * .00110231 / 1000 #convert to ktons

## Add in the 2yr precip term
fig2.df = left_join(fig2.df, precip.df)
```

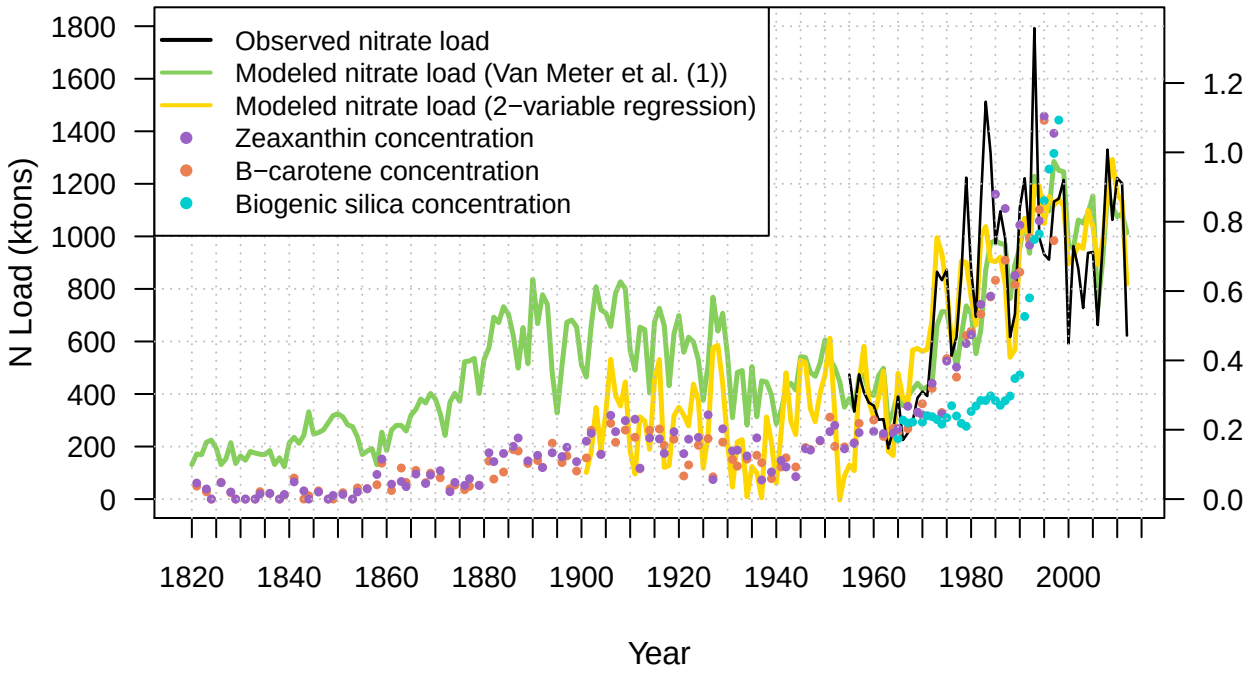
```
## Joining, by = "year"
```

```
precip.loop = rep(NA, n)
  j = 2
  for (u in i:n){
    precip.loop[u] = sum(fig2.df$precip[c(u-j+1):c(u)])
  }
fig2.df = data.frame(fig2.df, precip.cum2=precip.loop)

## Add in our model predictions
fit = lm(c(mrb.dat$mrb * 291451362.1 * .00110231 / 1000) ~ cropN.cum8 + precip.cum2, data=fig2.df[156:2
fig2.df = data.frame(fig2.df, preds=predict(fit, fig2.df))
```

Plot plot plot!

```
## Combine data into single data frame for plotting
yr.start = 1820; yr.end = 2015
df.plot = left_join(pigment, fig2.df, by=c('yearCentered'='year'))
#df.plot = left_join(df.plot, basin.input, by=c('yearCentered'='year'))
df.plot = left_join(df.plot, mrb.dat, by=c('yearCentered'='year'))
#df.plot = left_join(df.plot, BSid50, by=c('yearCentered'='year'))
df.plot = left_join(df.plot, BSie30, by=c('yearCentered'='year'))

df.plot = df.plot[df.plot$yearCentered %in% yr.start:yr.end,]


## Scale pigment data relative to the standard deviations of the pigments and of observed loading
sdLoads = sd(df.plot$observedLoadKtons[df.plot$yearCentered %in% 1955:1997])
sdZea = sd(df.plot$zea[df.plot$yearCentered %in% 1955:1997], na.rm=T)
scalingPig = sdLoads / sdZea

lwd.plot = 2.5
ymax = 1800 #y axis parameter
#pdf('~/Desktop/Fig2.pdf', width=8.5, height=4.4, useDingbats=F)
  plot(df.plot$yearCentered, df.plot$load, ylim=c(0, ymax), las=1, ylab='N Load (ktons)',xlab='Year', c
    lines(df.plot$yearCentered, df.plot$preds, col='gold', lwd=lwd.plot) # Our model predictions
    ## Add in observed loads
    lines(df.plot$yearCentered, df.plot$observedLoadKtons, col='black', lwd=1.3)
    ## Plot the pigments:
    points(df.plot$yearCentered, df.plot$bcaro*scalingPig, pch=16, col='#EA7F51', cex=.62)
    points(df.plot$yearCentered, df.plot$zea*scalingPig, pch=16, col='#9B61C4', cex=.62)
    points(df.plot$yearCentered, df.plot$BSie30*scalingPig, pch=16, col='cyan3', cex=.62)
    axis(side=1, at=seq(yr.start, yr.end, 5), cex.axis=.9)
    axis(side=2, at=seq(0, ymax, 200), las=1, cex.axis=.9)
    abline(h=seq(0, ymax, 200), v=seq(yr.start, yr.end, 5), lty=3, col='grey')
    ## Add legend:
    legend('topleft', bg='white',
         c('Observed nitrate load','Modeled nitrate load (Van Meter et al. (1))','Modeled nitrate load
         cex=.85, pch=c(NA,NA,NA,16,16,16), lty=c(1,1,1,NA,NA,NA), lwd=c(2,2,2,NA,NA,NA),
         col=c('black','#87CF5C','gold','#9B61C4','#EA7F51','cyan3'))
  ## Secondary y-axis for pigments
    par(new=T)
      plot(x=0,y=0, pch=NA, ylim=c(0,ymax/scalingPig), axes=F, xlab='',ylab='')
```

```
      axis(side=4, at=seq(0,ymax/scalingPig,.2),las=1, cex.axis=.9)
```




```
#dev.off()
```

Calculate NANI over the MRB 1987-2012 (at available 5yr census intervals).

```
## 3112 counties by 2110 HUC8 watersheds
## Each entry is the proportion of the county that each huc8 makes up, so rowsums ~ 1.
props = read_xlsx('HUC8_county_conversion.xlsx')
huc8 = data.frame('areakm2' = as.numeric(props[1,-c(1,2)])) #km2
props = props[-1,]
rownames(huc8) = gsub("X", "", names(props)[-c(1,2)])
props$FIPS = gsub("W0" ,"", props$FIPS)
props$FIPS = gsub("W" ,"", props$FIPS)
props$Area_km2 = as.numeric(props$Area_km2)
props = data.frame(apply(props, 2, as.numeric))

#### Read in NANI data and clean up
## County NANI (kg/km2/yr) Downloaded from https://www.sciencedirect.com/science/article/pii/S004896971
nani = read_xlsx('county_nani.xlsx', sheet = "County", n_max = 3114)
```

```
## New names:
## * `` -> `..1`
## * `` -> `..2`
## * `` -> `..3`
## * `` -> `..4`
## * `` -> `..5`
## * ... and 107 more
```

```
nani = nani[-2, nani[1,] %in% c("FIPScode", "NANI 1987", "NANI 1992", "NANI 1997", "NANI 2002", "NANI 20
names(nani) = nani[1,]
```

```
## Warning: Must use a character vector as names.
## This warning is displayed once per session.
```

```
nani = nani[-1,]
nani = data.frame(apply(nani, 2, as.numeric))

identical(props$FIPS, nani$FIPScode) ## TRUE = Good
```

```
## [1] TRUE
```

```
## Convert kg/km2 to kg
nani$NANI.1987kg = props$Area_km2 * nani$NANI.1987
nani$NANI.1992kg = props$Area_km2 * nani$NANI.1992
nani$NANI.1997kg = props$Area_km2 * nani$NANI.1997
nani$NANI.2002kg = props$Area_km2 * nani$NANI.2002
nani$NANI.2007kg = props$Area_km2 * nani$NANI.2007
nani$NANI.2012kg = props$Area_km2 * nani$NANI.2012

#### Now calculate kg N for each HUC8
huc8$'naniKg1987' = colSums(props[,-c(1,2)] * nani$NANI.1987kg)
huc8$'naniKg1992' = colSums(props[,-c(1,2)] * nani$NANI.1992kg)
huc8$'naniKg1997' = colSums(props[,-c(1,2)] * nani$NANI.1997kg)
huc8$'naniKg2002' = colSums(props[,-c(1,2)] * nani$NANI.2002kg)
huc8$'naniKg2007' = colSums(props[,-c(1,2)] * nani$NANI.2007kg)
huc8$'naniKg2012' = colSums(props[,-c(1,2)] * nani$NANI.2012kg)

row.names(huc8)[nchar(row.names(huc8))==7] = paste0("0", row.names(huc8))[nchar(row.names(huc8))==7]

## HUC8 shapefile
shape.huc8 = shapefile("WBDHU8_Reg_1_18.shp")

identical(shape.huc8$HUC8, rownames(huc8)) ## IF TRUE, Proceed, otherwise fix HUC8 ordering
```

```
## [1] TRUE
```

```
MARBarea = sum(shape.huc8$AreaSqKm[(shape.huc8$HUC2 %in% c("05","06","07","08","10","11"))]) # 3283723
colSums(huc8[shape.huc8$HUC2 %in% c("05","06","07","08","10","11"),]) / MARBarea
```

```
##     areakm2  naniKg1987  naniKg1992  naniKg1997  naniKg2002  naniKg2007
##    1.000793 2322.489113 2462.063909 2692.754209 2731.000544 2730.899122
##  naniKg2012
## 3138.722388
```