Ecosystem-based forecasts of recruitment in two menhaden species.

Supporting Information 1

1 Preamble

This R-markdown document contains code to replicate the result and figures contained in the manuscript. The code makes use of several packages for cleaner code, handier plotting, etc., which we now load.

```
suppressPackageStartupMessages({
    library("dplyr")
    library("ggplot2")
    library("grid")
    library("gridExtra")
    library("stringr")
    library("purrr")
    library('png')
    library('rgl')
    library('parallel')
})
```

There is one final package, 'rEDM', which performs the key Empirical Dynamic Modeling calculations.

```
library('rEDM')
```

For readers unfamiliar with the package, it is best to begin with the tutorial included

```
vignette('rEDM_tutorial')
```

Additionally, there is a script included in the supplemental materials with help functions.

```
source("S3_help_functions.R")
```

2 Data

2.1 Load Data

```
load('S2_data_inputs.Rdata')
```

There are 5 separate data frames: - bio.Atl: JAI and LPUE for Atlantic menhaden. - bio.Gulf.nt: JAI and LPUE for Gulf menhaden where Texas data are NOT included. - bio.Gulf.yt: JAI and LPUE for Gulf menhaden where Texas data ARE included. - phys.Atl: SLP, SST, and river discharge EOF(1)'s for the Atlantic menhaden range. - phys.Gulf: SLP, SST, and river discharge EOF(1)'s for the Gulf menhaden range.

2.2 Basic Plots

Here, we want to plot the biological data as time series for the Atlantic and the Gulf. First, we scale LPUE and JAI by their standard deviations. Then we convert the data from wide to long form for plotting.

```
df.plot <- bind_rows(
    bio.Atl %>%
    mutate(LPUE = LPUE / sd(LPUE, na.rm = TRUE),
           JAI = JAI / sd(JAI, na.rm = TRUE)) %>%
        select(Year, LPUE, JAI) %>%
        tidyr::gather(key = var, value = value, LPUE, JAI) %>%
        mutate(region = "Atlantic"),
    bio.Gulf.yt %>%
        mutate(LPUE = LPUE / sd(LPUE, na.rm = TRUE),
               JAI = JAI / sd(JAI, na.rm = TRUE)) %>%
        select(Year, LPUE, JAI) %>%
        tidyr::gather(key = var, value = value, LPUE, JAI) %>%
    mutate(region = "Gulf"),
    bio.Gulf.nt %>%
        mutate(`JAI without Texas` = JAI / sd(JAI, na.rm = TRUE)) %>%
        select(Year, `JAI without Texas`) %>%
        tidyr::gather(key = var, value = value, `JAI without Texas`) %>%
    mutate(region = "Gulf")
)

f1_a <- df.plot %>%
    filter(region == "Atlantic") %>%
    ggplot(aes(x=Year,y=value,col=var)) +
    geom_line(lwd=1) +
    labs(y = "Normalized Abundance") +
    theme_bw() +
    theme(  legend.text = element_text(size = 8),
            legend.background = element_rect(color = 'black',size=.1),
            legend.key.height = unit(0.5, "cm"),
            legend.key.width = unit(0.5, "cm"),
            legend.justification=c(0.1,0.1), legend.position=c(0.1,0.1) )

f1_b <- bio.Atl %>%
    ggplot(aes(x=LPUE,y=JAI)) +
    geom_point() +
    theme_bw()

f1_c <- df.plot %>%
    filter(region == "Gulf") %>%
    ggplot(aes(x=Year,y=value,col=var)) +
    geom_line(lwd=1) +
    labs(y = "normalized abundance") +
    theme_bw() +
    theme(  legend.text = element_text(size = 8),
            legend.background = element_rect(color = 'black',size=.1),
            legend.key.height = unit(0.5, "cm"),
            legend.key.width = unit(0.5, "cm"),
            legend.justification=c(0.1,0.1), legend.position=c(0.1,0.1) )
```

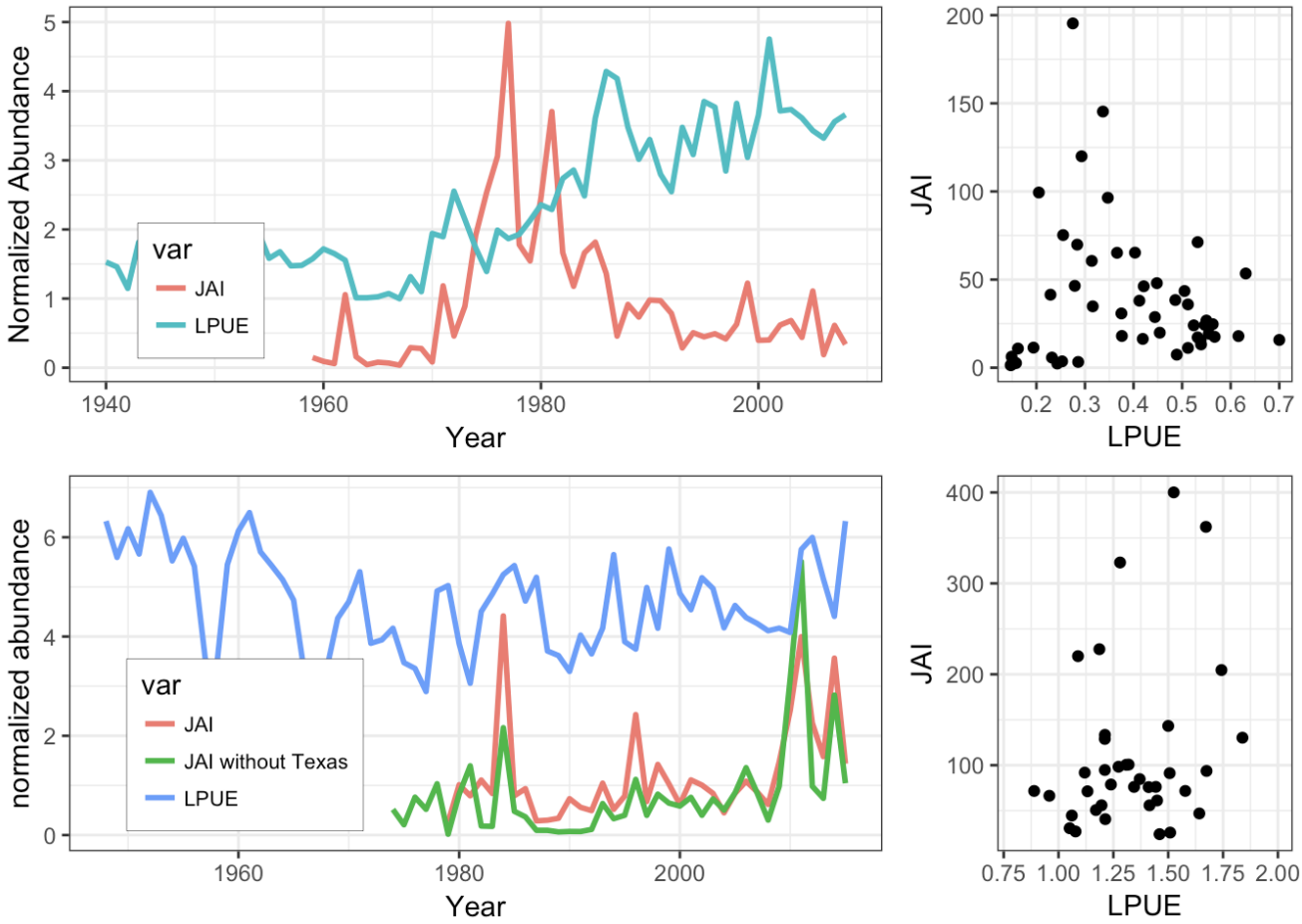
```
f1_d <- bio.Gulf.yt %>%
    ggplot(aes(x=LPUE,y=JAI)) +
    geom_point() +
    theme_bw()


grid.arrange(grobs=list(f1_a,f1_b,f1_c,f1_d), ncol=2,widths = c(2.2,1))
```

```
## Warning: Removed 19 rows containing missing values (geom_path).
```

```
## Warning: Removed 19 rows containing missing values (geom_point).
```

```
## Warning: Removed 57 rows containing missing values (geom_path).
```

```
## Warning: Removed 31 rows containing missing values (geom_point).
```



3 Univariate Analysis

3.1 Set-up univariate analysis function

The most basic procedure for univariate analysis is as follows: 1. Scale each time series by standard deviation. 2. Do simplex on whole time series (using E = 1:8). 3. Determine optimal E (by highest rho). 4. Do s-map using optimal E.

However, when time series exhibit high auto-correlation in time, it can be difficult to distinguish meaningful nonlinear predictability from the temporal auto-correlation. Thus, it can be useful to use univariate EDM to predict the first differences of the time series (which cannot be predicted from auto-correlation), then transform back to the raw values. In this case, we use the functions simplex_deltas and smap_deltas furnished in the help_functions instead of the basic simplex and smap functions in the rEDM package.

```
do_univariate_analysis <- function(ts, E.list = 1:8, predict_diff = FALSE, ...)
{
    ts <- ts / sd(ts, na.rm = TRUE)

    if(predict_diff){
        simplex_out <-  simplex_deltas(ts, E=E.list, ...)$delta_stats
    }else{
        simplex_out <- simplex(ts, E = E.list, ...)
    }

    E.star <- simplex_out$E[which.max(simplex_out$rho)]

    smap_out <- if(predict_diff){
        s_map_deltas(ts, E=E.star, ...)$delta_stats
    }else{
        s_map(ts, E = E.star, ...)
    }

    return(list(simplex = simplex_out,smap = smap_out))
}
```

3.2 Do univariate analysis on each abundance time series

We apply the do_univariate_analysis function now to each of the four biological time series of interest. Note that since the adult indices (LPUE) show very strong auto-correlation, we use the first-difference approach for these. To make plotting with `ggplot2` easier later on, we set the results up to go into ‘long’ form, where the columns denote the species, variable, and method details.

```
results <- list(do_univariate_analysis(bio.Atl$JAI, silent = TRUE) %>%
                        lapply(function(df) mutate(df,species = "Atlantic", vari
able = "JAI", method = "normal")),
                    do_univariate_analysis(bio.Gulf.nt$JAI, silent = TRUE) %>%
                        lapply(function(df) mutate(df,species = "Gulf", variable
= "JAI without Texas", method = "normal")),
                    do_univariate_analysis(bio.Gulf.yt$JAI, silent = TRUE) %>%
                        lapply(function(df) mutate(df,species = "Gulf", variable
= "JAI", method = "normal")),
                    do_univariate_analysis(bio.Atl$LPUE, predict_diff = TRUE, si
lent = TRUE) %>%
                        lapply(function(df) mutate(df,species = "Atlantic", vari
able = "LPUE", method = "diff")),
                    do_univariate_analysis(bio.Gulf.yt$LPUE, predict_diff = TRUE
, silent = TRUE) %>%
                        lapply(function(df) mutate(df,species = "Gulf", variable
= "LPUE", method = "diff")))
```

Repackage the results to separate `simplex` and `s-map` output. For each `s-map` model output, add columns for delta rho, delta MAE, delta RMSE.

```
results <- list(
    simplex = do.call(rbind, lapply(results,function(L) L$simplex)),
    smap = do.call(rbind,lapply(results, function(L) {
        L$smap %>% mutate(drho = rho - first(rho),
                  dmae = mae - first(mae),
                  drmse = rmse - first(rmse))
    }))
)
```

3.3 Run Surrogates on Everything

EDM analyses generally do not have parametric statistical descriptions of confidence intervals for calculating significance. Instead, these can be simulted by running analysis on appropraite surrogate time series. Here, develop null distributions for the s-map nonlinearity test. Recall that s-maps indicate nonlinear dynamics when nonlinear models, $\theta > 0$, have greater forecast accuracy than the equivalent linear S-map, $\theta = 0$. That is, if we define

$$\Delta\rho = \max_{\theta}(\rho) - \rho\Big|_{\theta=0}$$

$$\Delta\rho = \min_{\theta}(mae) - mae\Big|_{\theta=0}.$$

Nonlinearity is indicated if $\Delta\rho$ is positive or alternatively if $\Delta\mae$ is negative. However, for relatively short time series, a time series with linear dynamics could potentially produce a

small improvement for nonlinear θ due. Thus, the null hypothesis we test to establish significance can be stated as “what is the probability that a time series with the same basic characteristics as X would spuriously produce a given $\Delta\rho$?”.

Thus we compute a null distribution for $\Delta\rho$ using surrogate time series that have the same linear Fourier spectrum, but randomized phases (Ebisuzakia 1999). This is accomplished with the ‘rEDM’ function, make_surrogate_data(). We then run the same univariate analysis on each surrogate, and recorded the S-map $\Delta\rho$.

```
do_null_s_map <- function(x, n.surr = 100, predict_diff = FALSE, n.core = 1, ...)
{
    do.call(rbind,
            mclapply(1:n.surr,function(idx) {
                x.surr <- make_surrogate_data(x[!is.na(x)],method='ebisuzaki')
                df.out <- do_univariate_analysis(x.surr,predict_diff = predict_di
ff, ...)

                return(mutate(df.out$smap,idx=idx))
            }, mc.cores = n.core)
    )
}
```

Note that if the package `parallel` is not available, `mclapply` can be changed to `lapply`.

We can now apply this function to the JAI and LPUE time series for both coasts. Note that this section is set with “eval=FALSE” because the computations are intensive and should not be rerun every time the markdown is compiled. However, when the chunks are run, they create the .Rdata file that contains the results.

```
# ```{r}
number_of_surrogates <- 500
results.null <- bind_rows(do_null_s_map(bio.Atl$JAI, silent = TRUE, n.surr = numb
er_of_surrogates) %>%
                          mutate(species = "Atlantic", variable = "JAI", method =
"normal"),
                      do_null_s_map(bio.Gulf.nt$JAI, silent = TRUE, n.surr = numbe
r_of_surrogates) %>%
                          mutate(species = "Gulf", variable = "JAI without Texas",
method = "normal"),
                      do_null_s_map(bio.Gulf.yt$JAI, silent = TRUE, n.surr = numbe
r_of_surrogates) %>%
                          mutate(species = "Gulf", variable = "JAI", method = "nor
mal"),
                      do_null_s_map(bio.Atl$LPUE, predict_diff = TRUE, silent = TR
UE, n.surr = number_of_surrogates) %>%
                          mutate(species = "Atlantic", variable = "LPUE", method =
"diff"),
                      do_null_s_map(bio.Gulf.yt$LPUE, predict_diff = TRUE, silent
= TRUE, n.surr = number_of_surrogates) %>%
                          mutate(species = "Gulf", variable = "LPUE", method = "di
ff"))

results.null <- results.null %>% filter(theta==0) %>%
         rename(rho0=rho,mae0=mae,rmse0=rmse) %>%
         select(idx,species,variable,method,rho0,mae0,rmse0) %>%
         right_join(results.null,by=c("idx","species","variable","method") ) %>%
         mutate(drho=rho-rho0,dmae=mae-mae0,drmse=rmse-rmse0)

save(results.null,file="./univariate_ebi_null.Rdata")
```

3.4 Plots

Next, we make plots that show simplex, S-map (including null results) for each variable (JAI/LPUE for Gulf/Atlantic).

First we set up lists.

```
load("univariate_ebi_null.Rdata")


plot_rows <- list( c("Atlantic","JAI","normal"),
                   c("Gulf","JAI","normal"),
                   c("Atlantic","LPUE","diff"),
                   c("Gulf","LPUE","diff") )

g_rho <- vector(mode='list')
g_mae <- vector(mode='list')
```

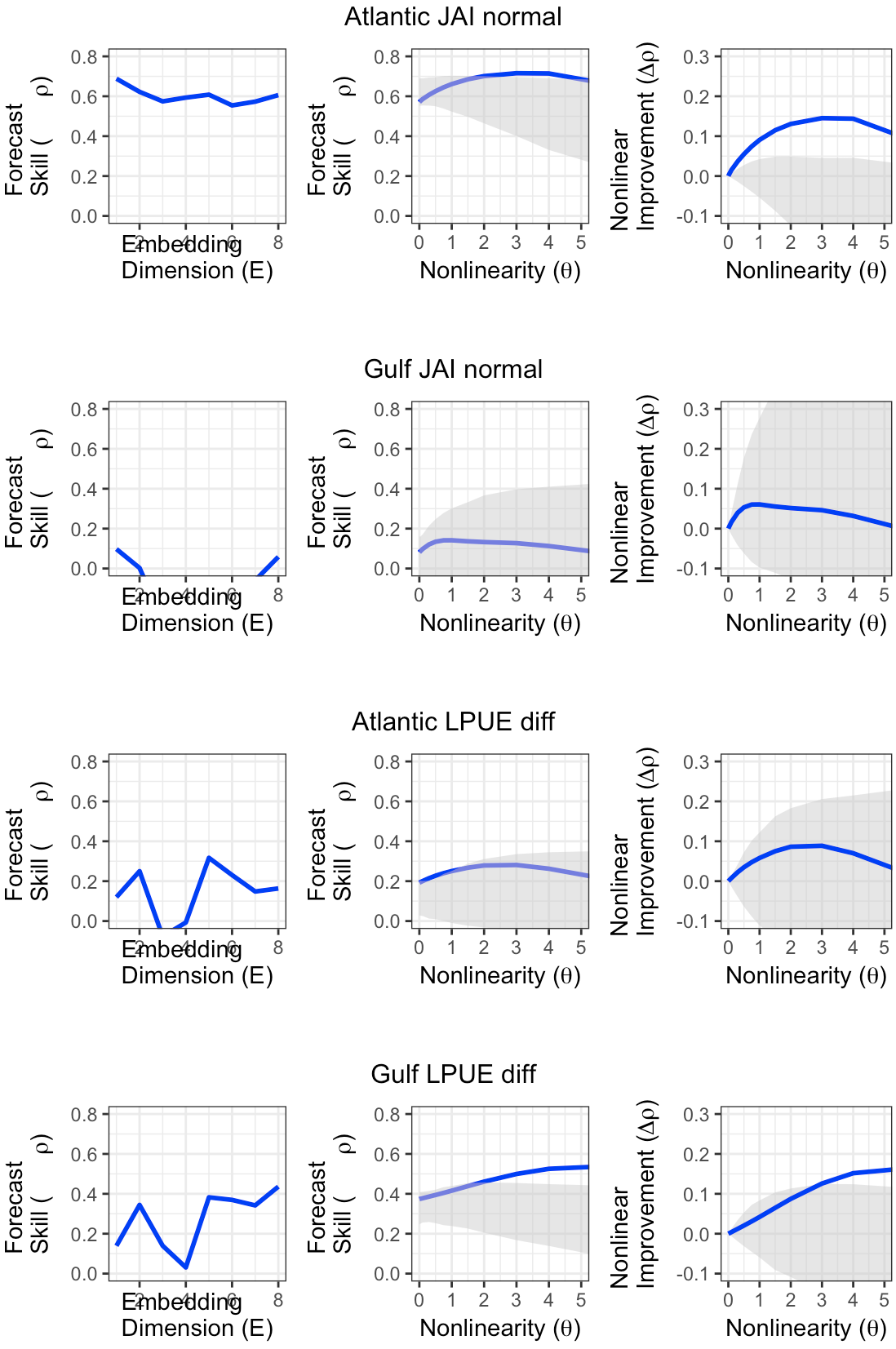
Next we make three panels for each of the four biological varriables. (A) shows simplex results, forecast skill (ρ) as a function of embedding dimension (E). (B) shows S-map result, forecast skill (ρ) as a function of S-map nonlinearity ($theta$). (C) is similar, but shows the nonlinear forecast improvement ($\Delta\rho$) as a function of (ρ). The null distribution created by surrogates is included as a shaded region.

```
for(rowdex in 1:length(plot_rows)){

    row.species <- plot_rows[[rowdex]][1]
    row.variable <- plot_rows[[rowdex]][2]
    row.method <- plot_rows[[rowdex]][3]

    h_A <- results$simplex %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=E,y=rho)) + geom_line(lwd = 1,col="blue") +
        coord_cartesian(ylim= c(0,.8)) +
        xlab(expression("Embedding \nDimension (E)")) +
        ylab(expression(paste("Forecast \nSkill (",rho,")"))) +
        theme_bw()

    h_B1 <- results$smap %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=theta,y=rho)) + geom_line(lwd = 1,col="blue") +
        xlab(expression(paste("Nonlinearity (",theta,")"))) +
        ylab(expression(paste("Forecast \nSkill (",rho,")"))) +
        theme_bw()

    h_C1 <- results$smap %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=theta,y=drho)) + geom_line(lwd = 1,col="blue") +
        coord_cartesian(ylim= c(-.1,.3)) +
        xlab(expression(paste("Nonlinearity (",theta,")"))) +
        ylab(expression(paste("Nonlinear \nImprovement (",Delta*rho,")"))) +
        theme_bw()

    row.uni.null <- results.null %>%
        filter(species==row.species,variable==row.variable,method==row.method)


    h_A <- h_A +
        theme(plot.margin=unit(c(0.5,0.5,2.0,1.0), "lines"),
              axis.title.x = element_text(vjust = 1,hjust=0.5))

    h_B <- h_B1 + stat_summary(data=row.uni.null, geom="ribbon", fill="grey80", a
lpha = .5,
                 fun.ymin = function(x) quantile(x, 0.05),
                 fun.ymax = function(x) quantile(x, 0.95)) +
        coord_cartesian(ylim= c(0,.8), xlim = c(0,5)) +
        theme_bw() +
        theme(plot.margin=unit(c(0.5,0.5,2.0,1.0), "lines"))
```

```
    h_C <- h_C1 + stat_summary(data=row.uni.null, geom="ribbon", fill="grey80", a
lpha = .5,
                  fun.ymin = function(x) quantile(x, 0.05),
                  fun.ymax = function(x) quantile(x, 0.95)) +
        coord_cartesian(ylim = c(-.1,.3), xlim=c(0,5)) +
        theme_bw() +
        theme(plot.margin=unit(c(0.5,0.5,2.0,1.0), "lines"))

    h_C$layers <- rev(h_C$layers)

    g_rho[[rowdex]] <- grid.arrange(h_A, h_B, h_C, nrow=1,
                                          top=do.call(paste,as.list(plot_rows[[r
owdex]])))
}
```

Finally we arrange the pannels using `grid.arrange` .

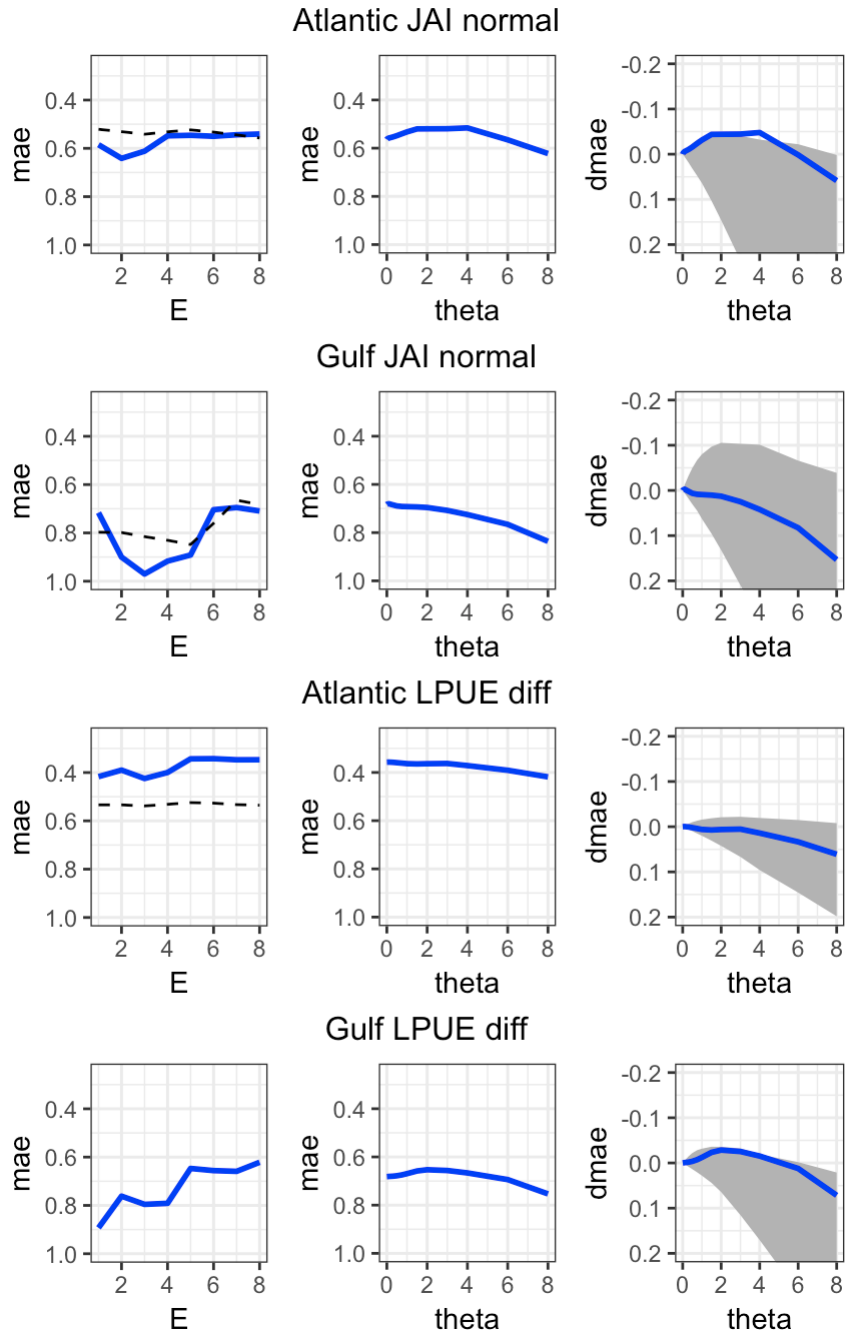
```
do.call(grid.arrange,c(g_rho,ncol=1))
```



We create a second set of figures that show results using (normalized) MAE to quantify forecast skill (or rather forecast error) instead of Pearson's correlation.

```
for(rowdex in 1:length(plot_rows)){

    row.species <- plot_rows[[rowdex]][1]
    row.variable <- plot_rows[[rowdex]][2]
    row.method <- plot_rows[[rowdex]][3]

    h_A <- results$simplex %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=E,y=mae)) + geom_line(lwd = 1,col='blue') +
        geom_line(aes(y=const_pred_mae),lty=2) +
                scale_y_reverse() +
        coord_cartesian(ylim= c(1,.25)) +
        theme_bw()

    h_B <- results$smap %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=theta,y=mae)) + geom_line(lwd = 1,col='blue') +
                scale_y_reverse() +
        coord_cartesian(ylim= c(1,.25)) +
        theme_bw()

    h_C1 <- results$smap %>%
        filter(species==row.species,variable==row.variable,method==row.method) %>
%
        ggplot(aes(x=theta,y=dmae)) + geom_line(lwd = 1,col='blue') +
        scale_y_reverse() +
        coord_cartesian(ylim= c(-.2,.2)) +
        theme_bw()

    row.uni.null <- results.null %>%
        filter(species==row.species,variable==row.variable,method==row.method)

    h_C <- h_C1 + stat_summary(data=row.uni.null, geom="ribbon", fill="grey70",
                 fun.ymin = function(x) quantile(x, 0.05),
                 fun.ymax = function(x) quantile(x, 0.95)) +
        theme_bw()

    h_C$layers <- rev(h_C$layers)

    g_mae[[rowdex]] <- grid.arrange(h_A, h_B, h_C, nrow=1,
                                        top=do.call(paste,as.list(plot_rows[[r
owdex]])))
}
```

```
grid.arrange(grobs=g_mae,ncol=1)
```



4 CCM

To keep the code compact, we define a function that performs CCM analysis for a matrix of "predictor"/"target" variable pairs. The first step is to performing cross-mapping (with full library) from "predictor" to "target" using different E at a prediction time of $tp = -1$. E^* is selected to maximize the $tp = -1$ cross-map skill (ρ).. Next, cross-map skill is measured using this E^* with $tp = 0$. This method provides a way to determine cross-map E for a number of combinations with less risk of spurious results than fitting E on $tp = 0$.

```
do_ccm_runs <- function(block, ccm_runs,
                            E_list = 1:8, tp_fit = -1, tp_pred = 0,
                            lib_sizes = seq(from = 10, to = NROW(block), by = 5)
,
                            random_libs = TRUE, replace = FALSE,
                            silent = TRUE, ...)
{
    return(do.call(rbind, lapply(1:NROW(ccm_runs), function(i) {
        out.temp <- do.call(rbind,
                    lapply(E_list, function(E) {
                        ccm(block, lib_column = ccm_runs$from[i], target_column =
ccm_runs$to[i],
                            E = E, random_libs = FALSE, lib_sizes = NROW(block), t
p = tp_fit, ...,
                            silent = silent)
                    }))

        E.star <- out.temp[which.max(out.temp$rho), 'E']

        ccm(block, lib_column = ccm_runs$from[i],
            target_column = ccm_runs$to[i],
            E = E.star,
            lib_sizes = lib_sizes,
            random_libs = random_libs, replace = replace,
            tp = tp_pred, silent = silent, ...)
    })))
}
```

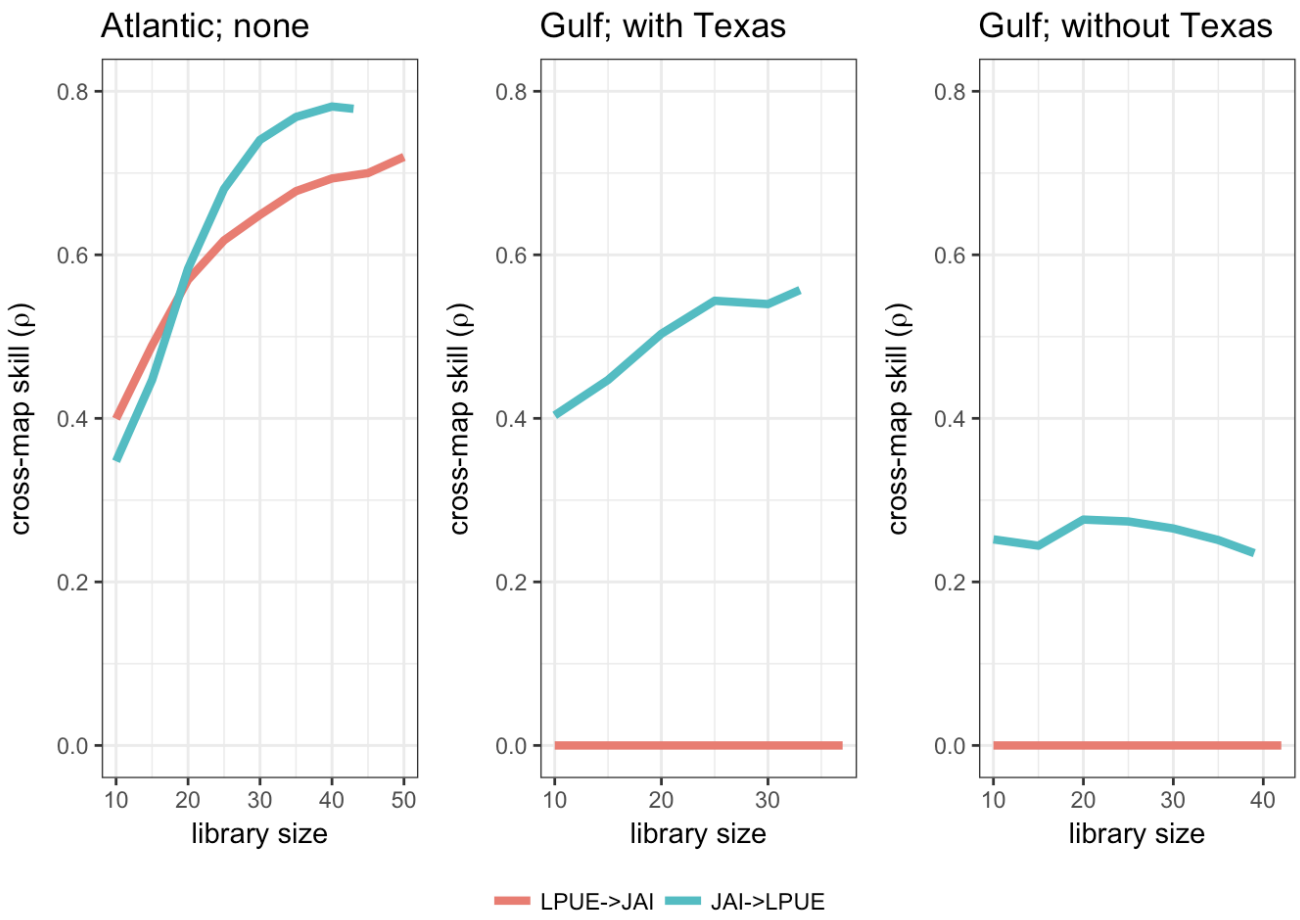
4.1 Between biological variables

We investigate the causal relationship between LPUE (adults) and JAI (juveniles). Measurements of bidirectional vs. unidirectional causality should inform us about the importance of deterministic biological drivers vs. stochastic influences on recruitment and the adult population.

```
# set up which effects to test using CCM
vars <- c("JAI", "LPUE")
ccm_runs_ex1 <- expand.grid(from = vars, to = vars)

# don't run CCM from a variable to itself
ccm_runs_ex1 <- ccm_runs_ex1[ccm_runs_ex1$from != ccm_runs_ex1$to,]

results_CCM_ex1 <- do.call(rbind,
                           list(do_ccm_runs(bio.Atl, ccm_runs_ex1, silent = TRUE)
%>%
                                    mutate(species = "Atlantic",label="none"),
                                do_ccm_runs(bio.Gulf.nt, ccm_runs_ex1, silent = T
RUE) %>%
                                    mutate(species = "Gulf",label="without Texas"
),
                                do_ccm_runs(bio.Gulf.yt, ccm_runs_ex1, silent = T
RUE) %>%
                                    mutate(species = "Gulf", label="with Texas"))
                           )
```

4.1.1 PLOTS

```
df.plot <- results_CCM_ex1 %>%
  mutate(experiment = interaction(species,label, sep="; ")) %>%
  mutate(ccm_label = interaction(lib_column,target_column,sep="->")) %>%
  select(species,label,experiment,ccm_label,lib_size,rho,mae,rmse) %>%
  group_by(species,label,experiment,ccm_label,lib_size) %>%
  summarise_at(vars(rho,mae,rmse),funs(pmax(0,mean(.,na.rm=TRUE))))

labs_panels <- unique(df.plot$experiment)
n_panels <- length(labs_panels)
h_panels <- vector(mode="list",n_panels)

for(i_panel in 1:n_panels){

  h_panels[[i_panel]] <- df.plot %>%
    filter(experiment == labs_panels[[i_panel]]) %>%
    ggplot(aes(x=lib_size,y=rho,color=ccm_label)) + geom_line(lwd=1.5) +
    ylim(c(0,.8)) +
    labs(title=labs_panels[[i_panel]],x='library size',y=expression(paste('cross-
map skill (',rho,')')),col="") +
    theme_bw() +
    theme(legend.position = "bottom")

} # i_pannel


g_legend<-function(a.gplot){
    g <- ggplotGrob(a.gplot + theme(legend.position = "bottom"))$grobs
    legend <- g[[which(sapply(g, function(x) x$name) == "guide-box")]]
  return(legend)}

mylegend<-g_legend(a.gplot=h_panels[[2]])
lheight <- sum(mylegend$height)

grid.arrange(do.call(arrangeGrob, c(lapply(h_panels,
                                function(h_i) h_i + theme(legend.position="none"
)),
                           nrow=1)),
             mylegend, nrow=2,heights = unit.c(unit(1, "npc") - lheight, lheight)
)
```



CCM analysis between JAI and LPUE suggests that including Texas does indeed give a better measure of recruitment insofar as it shows stronger causal relationship than the JAI created without Texas data.

The lack of prediction in the other direction (LPUE predicts JAI, i.e. effect of JAI on LPUE) is a bit puzzling. If JAI is strongly influenced by exogenous stochastic drivers, it will be difficult to predict current JAI from past LPUE, since the LPUE time series cannot contain current information about a stochastic driver. However, lag prediction should be possible.

We define a function that performs the basic do_ccm_runs() function across multiple lags. Note that for unlagged CCM, we used tp = -1 to select E to then measure CCM at tp = 0. Here, we use tp = lag - 1 to measure CCM at tp = 0. Also, we only need to measure CCM at full library to examine optimal prediction time.

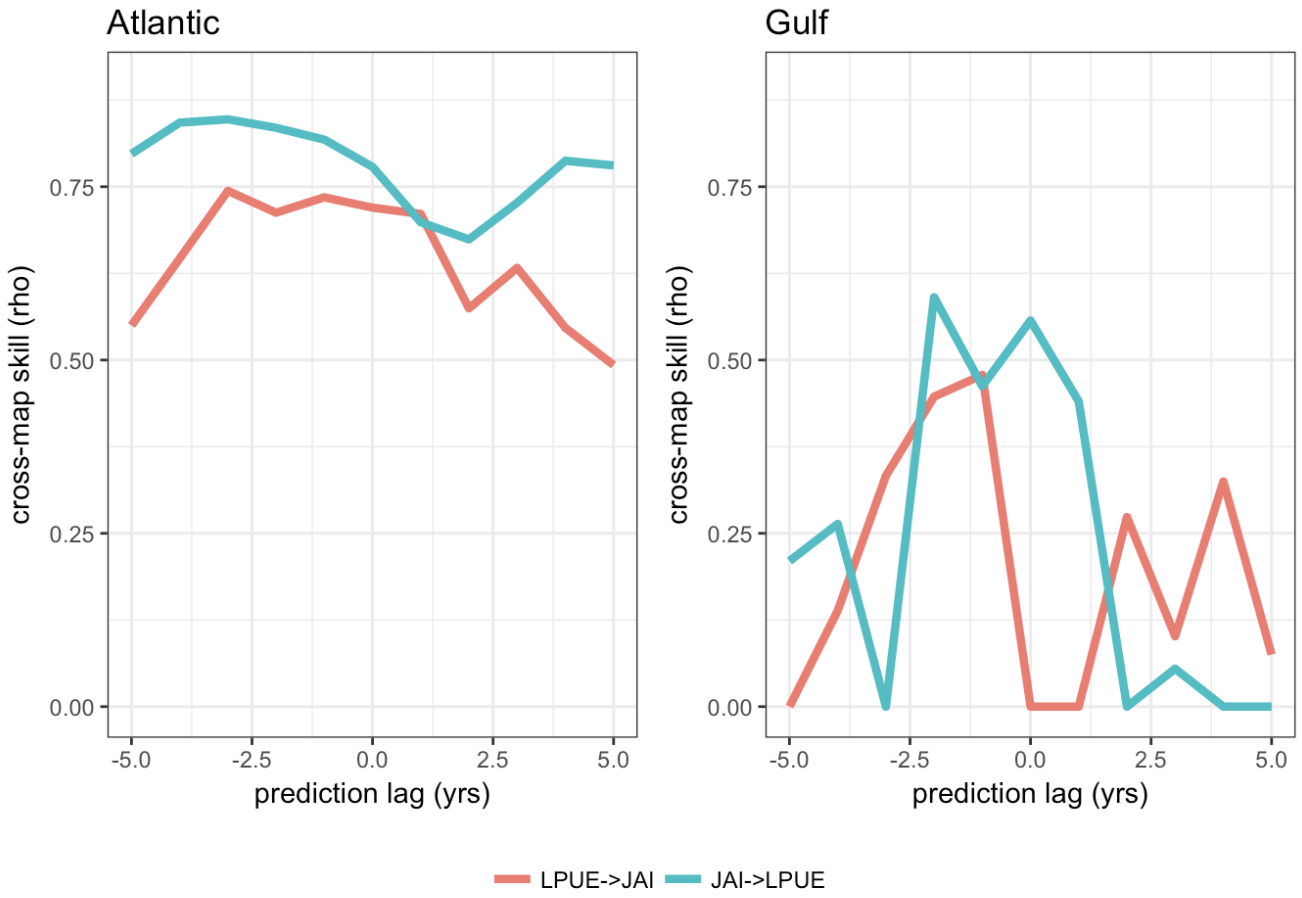
```
do_ccm_lag_analysis <- function(block, ccm_runs, lag_list = seq(-5,5,by=1), ...){
    do.call(rbind,
            lapply( lag_list, function(lag){
                do_ccm_runs(block, ccm_runs, tp_fit = (lag-1),tp_pred = lag,
                            lib_sizes = NROW(block),random_libs = FALSE, num_samp
les = 1, ...)
            }))}
```

Now we do ccm with varried time lag for the same pairings as above, i.e. ccm_runs_ex1.

```
results_lags_ex1 <- do.call(rbind,list(do_ccm_lag_analysis(bio.Atl,
                                            ccm_runs_ex1, lag_list = seq(-5,5,by
=1), silent = TRUE) %>%
                              mutate(species = "Atlantic",label="none"),
                          do_ccm_lag_analysis(bio.Gulf.yt,
                                            ccm_runs_ex1, lag_list = seq(-5,5,by
=1), silent = TRUE) %>%
                              mutate(species = "Gulf",label="with Texas")))
```

Now we plot the ccm skill as a function of time lag.

```
h_lags <- vector(mode="list",2)
L_species <- c("Atlantic","Gulf")
L_var <- c("JAI","LPUE")

for(i_species in 1:length(L_species)){

    species_i <- L_species[[i_species]]

    df.i <- results_lags_ex1 %>%
      filter(species==species_i) %>%
        mutate(ccm_label = interaction(lib_column,target_column,sep="->"))

    phys_i <- df.i$target_colum[1]
    title_i <- species_i
    h_lags[[i_species]] <- ggplot(df.i,aes(x=tp,y=pmax(0,rho),color=ccm_label)) +
geom_line(lwd=1.5) +
      labs(title=title_i,x='prediction lag (yrs)',y='cross-map skill (rho)') +
      ylim(c(0,0.9)) +
      theme_bw() +
      labs(col="") +
      theme(legend.position = "bottom")

  }

g_legend<-function(a.gplot){
    g <- ggplotGrob(a.gplot + theme(legend.position = "bottom"))$grobs
    legend <- g[[which(sapply(g, function(x) x$name) == "guide-box")]]
  return(legend)}

mylegend<-g_legend(a.gplot=h_lags[[1]])
lheight <- sum(mylegend$height)*1.5

grid.arrange(do.call(arrangeGrob, c(lapply(h_lags,
                                   function(h_i) h_i + theme(legend.position="none"
)),
                            nrow=1)),
             mylegend, nrow=2,heights = unit.c(unit(1, "npc") - lheight, lheight)
)
```



Cross-map shows that indeed there is evidence of dynamic causality from Gulf JAI to Gulf Lpue at a negative time-lag, consistent with Gulf recruitment being strongly driven by stochastic factors.

4.2 Environmental Drivers

We next use the same basic do_ccm_runs() to examine possible environmental drivers. We can use the same code, but with a different set of “from” and “to” variables.

```
ccm_runs_ex2 <- expand.grid(from = c("JAI","LPUE"), to = c("SLP","SST","streamflo
w"))

block.temp <- full_join(bio.Atl,phys.Atl,by="Year")

results_CCM_ex2 <- do.call(rbind,list(
                    do_ccm_runs(full_join(bio.Atl,phys.Atl,by="Year"), ccm_runs_e
x2, silent = TRUE) %>%
                        mutate(species = "Atlantic",label="none"),
                    do_ccm_runs(full_join(bio.Gulf.yt,phys.Gulf,by="Year"), ccm_r
uns_ex2, silent = TRUE) %>%
                        mutate(species = "Gulf",label="with Texas")))
```

4.2.0.1 NULL Analysis without Lags

We now develop null distributions for the environmental CCM analysis. Here we use Ebisuzaki surrogates, which preserves the distribution of values of the time series, but destroys any dynamic relationship with the real data. There is a slight complication, which is that the code to generate Ebisuzaki surrogates in the 'rEDM' package uses fft() and cannot deal with NAs in the data. Thus we need to write a quick intermediate function to ignore the NAs.

```
make_surrogate_ignoreNA <- function(y,number_of_surrogates=1){
        I_good <- which(is.finite(y))

        Y <- matrix(NA,nrow = length(y),ncol=number_of_surrogates)
        Y[I_good,] <- make_surrogate_data(y[I_good],num_surr = number_of_surrogat
es,method='ebisuzaki')
        return(Y)}
```

For ease, we define a function that repeats the do_ccm_runs over a given number of surrogate realizations.

```
do_null_ccm_runs <- function(block, ccm_runs, n.surr = 100, n.core = 1, ...)
{
    do.call(rbind,
            mclapply(1:n.surr,function(idx) {

                # If this code is adapted to do ccm runs where "from" and "to" ar
en't disjoint,
                # this next bit of code won't work quite as intended.

                    block <- block %>%
                        mutate_at(as.character(unique(ccm_runs[,'to'])),funs(make
_surrogate_ignoreNA))

                df.out <- do_ccm_runs(block, ccm_runs, ...)
                return(mutate(df.out,idx=idx))
            }, mc.cores = n.core)
    )
}
```

As with the univariate surrogates, the following section is set with 'eval=FALSE' because the computations are intensive and should not be rerun every time the markdown is compiled. However, when the chunk is run, it creates the .Rdata file that contains the results and is used to generate the figures.

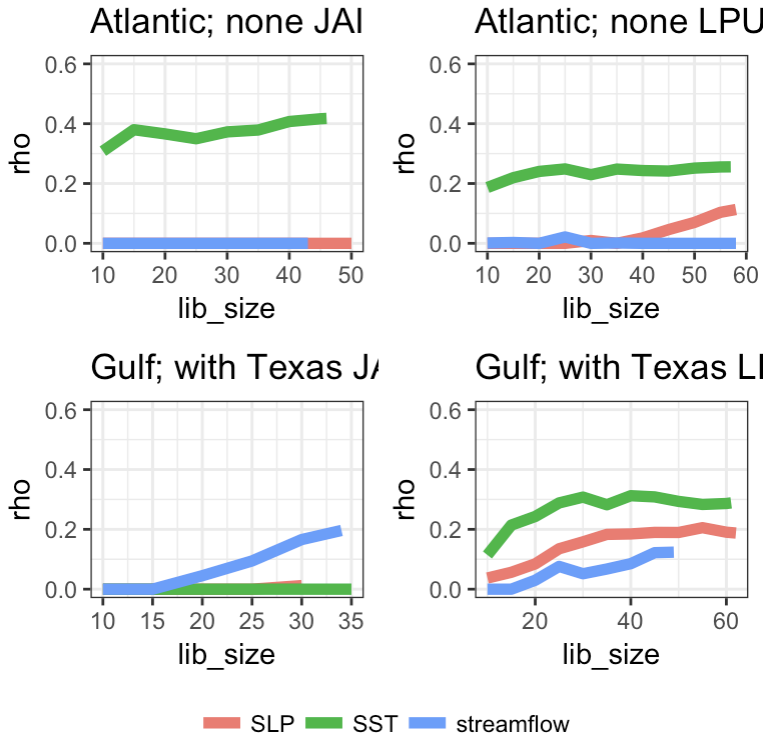
```
# ```{r}
number_of_surrogates <- 500
results_null_CCM_ex2 <- do.call(rbind,list(do_null_ccm_runs(full_join(bio.Atl,phy
s.Atl,by="Year"), ccm_runs_ex2,
                                                    n.surr = number_of_surrogates, sile
nt = TRUE) %>%
                         mutate(species = "Atlantic",label="none"),
                     do_null_ccm_runs(full_join(bio.Gulf.yt,phys.Gulf,by="Year"),
ccm_runs_ex2,
                                      n.surr = number_of_surrogates, silent = TRUE
) %>%
                         mutate(species = "Gulf",label="with Texas")))

save(results_null_CCM_ex2,file='./env_ccm_null.Rdata')
```

4.2.1 PLOT

```
load('./env_ccm_null.Rdata')
```

```
df.plot <- results_CCM_ex2 %>%
    mutate(experiment = interaction(species,label, sep="; ")) %>%
    # mutate(ccm_label = interaction(lib_column,target_column,sep="->")) %>%
    select(species,lib_column,target_column,experiment,lib_size,rho,mae,rmse) %>%
    group_by(species,lib_column,target_column,experiment,lib_size) %>%
    summarise_at(vars(rho,mae,rmse),funs(pmax(0,median(.,na.rm=TRUE))))

labs_panels <- unique(df.plot$experiment)
n_panels <- length(labs_panels)
h_panels <- vector(mode="list",n_panels)

h_ccm_ex2 <- vector(mode="list",4)
L_species <- c("Atlantic","Gulf")
L_var <- c("JAI","LPUE")

for(i_species in 1:length(L_species)){
    for(i_var in 1:length(L_var)){


        i_1d <- length(L_species)*(i_species-1) + i_var
        var_i <- L_var[[i_var]]
        species_i <- L_species[[i_species]]
        ## JAI
        h_ccm_ex2[[i_1d]] <- df.plot %>%
            filter(species == species_i) %>%
            filter(lib_column == var_i) %>%
            # mutate(ccm_label = interaction(lib_column,target_column,sep="->"))
%>%
            ggplot(aes(x=lib_size,y=rho,color=target_column)) + geom_line(lwd=2)
+
            ylim(c(0,.6)) +
            labs(title=paste(labs_panels[[i_species]],var_i),
                 col="") +
            theme_bw() +
            theme(legend.position = "bottom")


    } # i_var
} # i_species
```

```
g_legend<-function(a.gplot){
    g <- ggplotGrob(a.gplot + theme(legend.position = "bottom"))$grobs
    legend <- g[[which(sapply(g, function(x) x$name) == "guide-box")]]
  return(legend)}

mylegend<-g_legend(a.gplot=h_ccm_ex2[[1]])
lheight <- sum(mylegend$height)

# do.call(grid.arrange,c(h,nrow = 2))
grid.arrange(do.call(arrangeGrob, c(lapply(h_ccm_ex2,
                                   function(h_i) h_i + theme(legend.position="none"
)),
                            nrow=2)),
             mylegend, nrow=2,heights = unit.c(unit(1, "npc") - lheight, lheight)
)
```



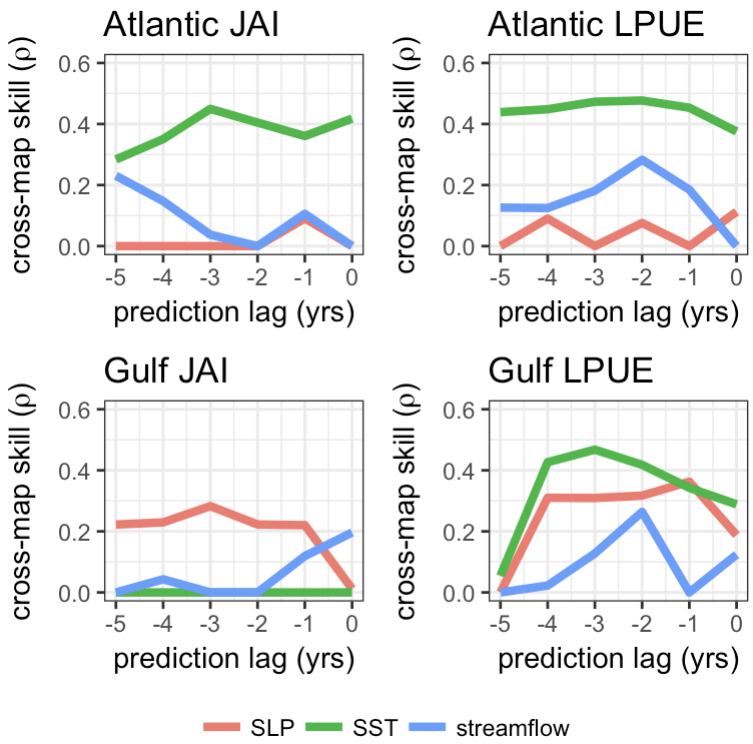
4.3 Lag Analysis

Since environmental variables are often best understood as stochastic, we must allow for the possibility of CCM only at a negative time lag. Thus we do the analysis for the same pairings as above, i.e. ccm_runs_ex2.

```
results_CCM_lags <- do.call(rbind,list(do_ccm_lag_analysis(full_join(bio.Atl,phys
.Atl,by="Year"),
                                                ccm_runs_ex2, lag_list = seq(-5,0,by
=1), silent = TRUE) %>%
                               mutate(species = "Atlantic",label="none"),
                           do_ccm_lag_analysis(full_join(bio.Gulf.yt,phys.Gulf,by="
Year"),
                                                ccm_runs_ex2, lag_list = seq(-5,0,by
=1), silent = TRUE) %>%
                               mutate(species = "Gulf",label="with Texas")))
```

```
h_ccm_lags <- vector(mode="list",4)
L_species <- c("Atlantic","Gulf")
L_var <- c("JAI","LPUE")

for(i_species in 1:length(L_species)){
  for(i_var in 1:length(L_var)){

    i_1d <- length(L_species)*(i_species-1) + i_var
    var_i <- L_var[[i_var]]
    species_i <- L_species[[i_species]]

    df.i <- results_CCM_lags %>%
      filter(species==species_i) %>%
      filter(lib_column==var_i)

    phys_i <- df.i$target_colum[1]
    title_i <- paste(species_i,var_i)
    h_ccm_lags[[i_1d]] <- ggplot(df.i,aes(x=tp,y=pmax(0,rho),color=target_column)
) + geom_line(lwd=1.5) +
      labs(title=title_i,
           col="",
           x="prediction lag (yrs)",
           y=expression(paste("cross-map skill (",rho,")"))) +
      ylim(c(0,0.6)) +
      theme_bw() +
      theme(legend.position = "bottom")

  }}
```

```
g_legend<-function(a.gplot){
    g <- ggplotGrob(a.gplot + theme(legend.position = "bottom"))$grobs
    legend <- g[[which(sapply(g, function(x) x$name) == "guide-box")]]
  return(legend)}

mylegend<-g_legend(a.gplot=h_ccm_lags[[1]])
lheight <- sum(mylegend$height)

grid.arrange(do.call(arrangeGrob, c(lapply(h_ccm_lags,
                                   function(h_i) h_i + theme(legend.position="none"
)),
                           nrow=2)),
             mylegend, nrow=2,heights = unit.c(unit(1, "npc") - lheight, lheight)
)
```



5 Multivariate EDM

```
results_multi_ex1 <- NULL
```

5.1 LPUE to predict JAI

CCM suggests that in the Atlantic, recruitment (JAI) can be predicted from stock (LPUE). This is particularly interesting, because conventional methods for prediction recruitment from stock via a Ricker curve (or other) have faired poorly with Menhaden. The conventional

approach, however, assumes that the single-species dynamics are independent of other factors (or slightly more complicated situations like the environmental Ricker), i.e. that the effect of stock on recruitment can be understood without accounting for the ecosystem context of the population.

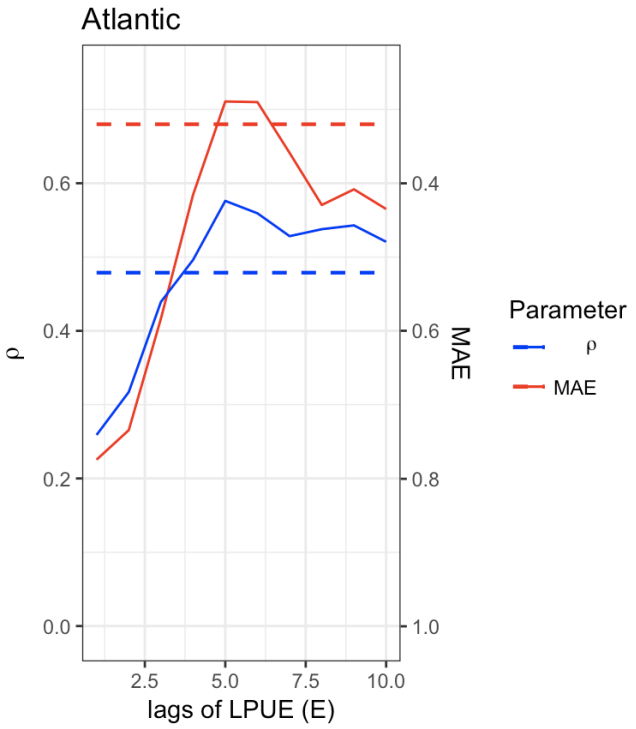
Here we follow up to (1) examine how effective it is and (2) contrast with typical Stock-Recruitment curve approaches to prediction by looking at dimensionality. To do this, we define another function:

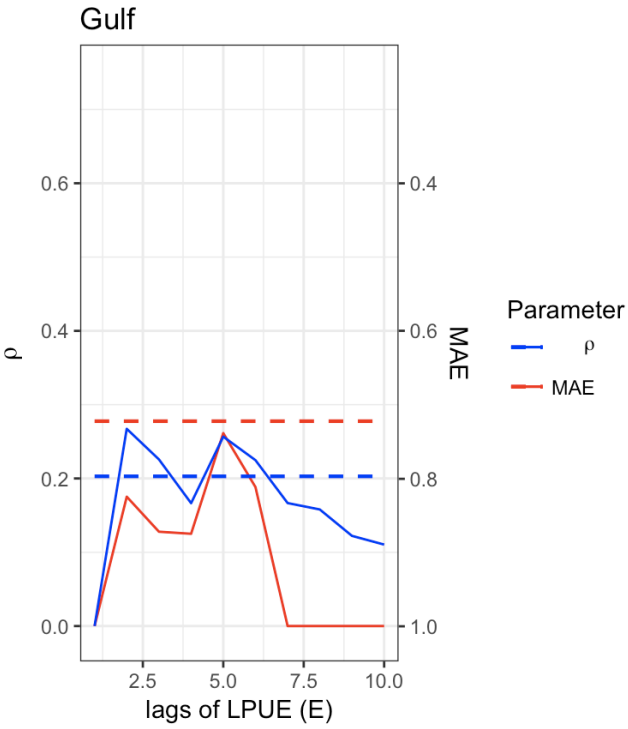
```
do_multivariate_E_analysis <- function(data,pred_col = 1,lag_col = 2,other_col =
NULL, E = 1:8, first_column_time = FALSE, ...){

    if(first_column_time){
        t <- data[,1]
        data <- data[,-1]
    }

    if(is.character(pred_col)) pred_col = match(pred_col,names(data))
    if(is.character(lag_col)) lag_col = match(lag_col,names(data))
    if(is.character(other_col)) other_col = match(other_col,names(data))

    block <- make_block(data = data, cols = c(pred_col,rep(lag_col,max(E)),other_
col),
                        delays = c(0,0:-(max(E)-1),rep(0,length(other_col)))) %>%
        as.data.frame() %>%
        mutate_all(funs((. - mean(.,na.rm=TRUE))/sd(.,na.rm=TRUE)))

    if(first_column_time){
        block <- block %>% mutate(time = t) %>% select(time,everything())
    }

    L_columns <- lapply(E,function(x) c(1+(1:x), 1+max(E)+seq(from=1,by = 1,lengt
h.out = length(other_col)) ))

    out <- block_lnlp(block, first_column_time = first_column_time,
                      method = "simplex", num_neighbors = "e+1",
                      columns = L_columns, target_column = 1, stats_only = TRUE,
                      theta = NULL) %>%
        mutate(embedding=str_count(embedding,",")+1) %>%
    rename(E=embedding)


}
```

```
E.list <- 1:10

results_multi_ex1 <- do.call(rbind,
                             list(do_multivariate_E_analysis(data=bio.Atl,pred_col
= "JAI",lag_col = "LPUE",
                                                             first_column_time = TR
UE, E = E.list) %>%
                                      mutate(species = "Atlantic",pred_vars="LPUE")
,
                                    do_multivariate_E_analysis(data=bio.Gulf.yt,pred_
col = "JAI",lag_col = "LPUE",
                                                             first_column_time = TR
UE, E = E.list) %>%
                                      mutate(species = "Gulf", pred_vars="LPUE"))
                             )
```

Make plots

```
h_mex1 <- vector(mode='list',2)

for(i_species in 1:2){
    species_i <- unique(results_multi_ex1$species)[i_species]

    df.i <- results_multi_ex1 %>%
        filter(species == species_i)
    p <- ggplot(df.i, aes(x = E))

    p <- p + geom_line(aes(y = pmax(0,rho), colour = "rho"))
    p <- p + geom_line(aes(y = 1 - pmin(1,mae), colour = "MAE"))

    # add  constant predictors
    p <- p + geom_line(aes(y = pmax(0,const_pred_rho), colour = "rho"),lty=2,lwd=
0.75)
    p <- p + geom_line(aes(y = 1 - pmin(1,const_pred_mae), colour = "MAE"),lty=2,
lwd=0.75)

    # now adding the secondary axis, following the example in the help file ?scal
e_y_continuous
    # and, very important, reverting the above transformation
    p <- p + scale_y_continuous(limits = c(-0.01,.75),sec.axis = sec_axis(~1-., n
ame = "MAE"))

    # modifying colours and theme options
    p <- p + scale_colour_manual(values = c("blue", "red"),labels=expression(rho,
"MAE"))
    p <- p + labs(y = expression(rho),
                  x = "lags of LPUE (E)",
                  title = species_i,
                  colour = "Parameter")
    p <- p + theme(legend.position = c(0.8, 0.2)) + theme_bw()

    h_mex1[[i_species]] <- p
  }


do.call(grid.arrange,c(h_mex1,nrow = 1))
```


Atlantic
ρ
MAE
0.0
0.2
0.4
0.6
0.4
0.6
0.8
1.0
2.5
5.0
7.5
10.0
lags of LPUE (E)
Parameter
ρ
MAE


Gulf
ρ
MAE
0.0
0.2
0.4
0.6
0.4
0.6
0.8
1.0
2.5
5.0
7.5
10.0
lags of LPUE (E)
Parameter
ρ
MAE

```
h_A <- bio.Atl %>%
    mutate(JAI = lead(JAI,1)) %>%
    ggplot(aes(x=LPUE,y=JAI)) + geom_point() +
    labs(x  = "Stock(t) (LPUE)",y = "Recruits(t+1) (JAI)") +
    theme_bw()

h_B <- results_multi_ex1 %>%
    filter(species=="Atlantic") %>%
    filter(pred_vars==c("LPUE")) %>%
    ggplot(aes(x=E,y=mae)) + geom_line(lwd=1,col="blue") +
    geom_line(aes(y=const_pred_mae),col="grey60",lty=2) +
    labs(x = "lags of LPUE (E)") +
    theme_bw()

block_3d <- bio.Atl %>%
    mutate(JAI = lead(JAI,1)) %>%
    mutate(LPUE_1 = LPUE) %>%
    mutate(LPUE_2 = lag(LPUE,3)) %>%
    mutate_all(funs((. - min(.,na.rm=TRUE))/(max(.,na.rm=TRUE) - min(.,na.rm=TRUE
))))
rgl::plot3d(x=block_3d$LPUE_1,y=block_3d$LPUE_2,z=block_3d$JAI)
rgl::lines3d(x=block_3d$LPUE_1,y=block_3d$LPUE_2,z=block_3d$JAI)


rgl.snapshot("3D S-R Atl.png")
rgl.close()

## imports the png files
png.i <- readPNG("3D S-R Atl.png")
h_C <- rasterGrob(png.i, interpolate=TRUE)

grid.arrange(h_A,h_B,h_C,nrow=1)
```

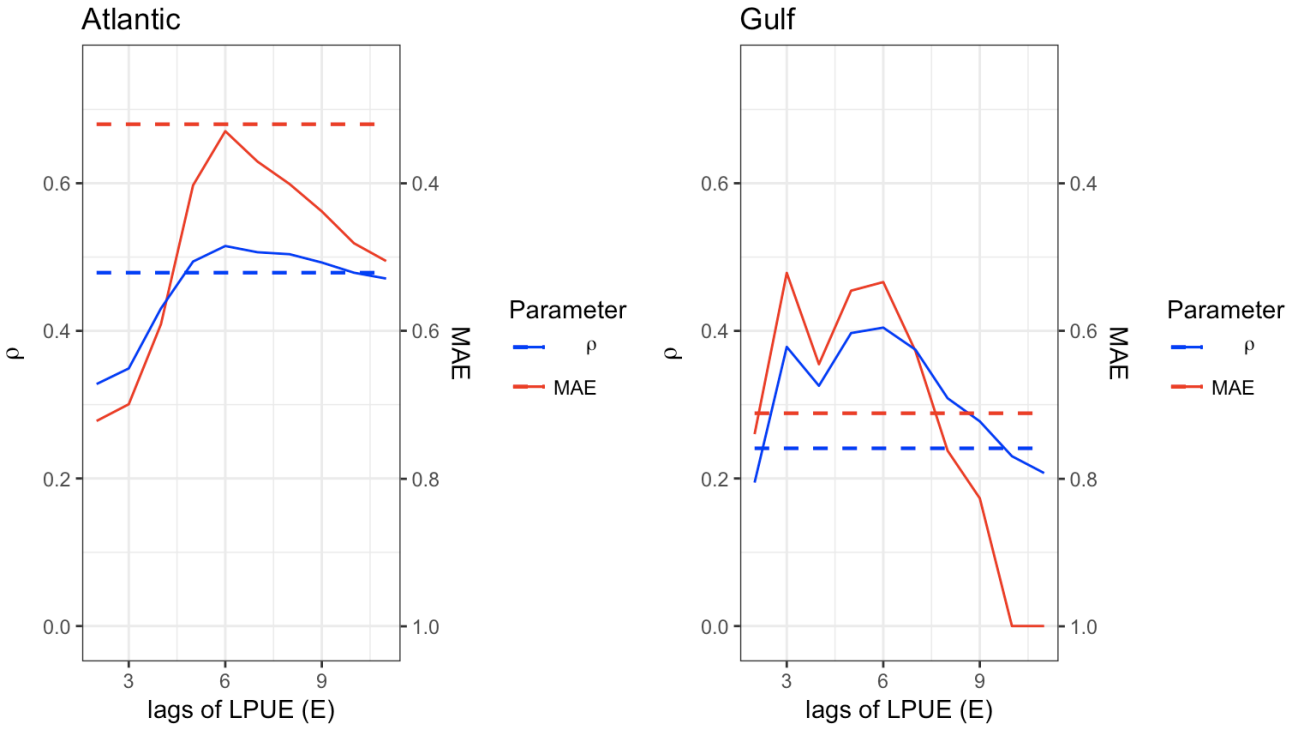
```
h_A <- bio.Gulf.yt %>%
    mutate(JAI = lead(JAI,1)) %>%
    ggplot(aes(x=LPUE,y=JAI)) + geom_point() +
    labs(x  = "Stock(t) (LPUE)",y = "Recruits(t+1) (JAI)") +
    theme_bw()

h_B <- results_multi_ex1 %>%
    filter(species=="Gulf") %>%
    filter(pred_vars==c("LPUE")) %>%
    ggplot(aes(x=E,y=mae)) + geom_line(lwd=1,col="blue") +
    geom_line(aes(y=const_pred_mae),col="grey60",lty=2) +
    labs(x = "lags of LPUE (E)") +
    theme_bw()

grid.arrange(h_A,h_B,nrow=1)
```

5.2 Add in Environment

```
E.list <- 1:10

results_multi_ex2 <- do.call(rbind,
                             list(do_multivariate_E_analysis(data=full_join(bio.Atl
,phys.Atl,by="Year"),
                                                             pred_col = 'JAI',lag_c
ol = 'LPUE',other_col='SST',
                                                             first_column_time = TR
UE, E = E.list) %>%
                                      mutate(species = "Atlantic",pred_vars=paste("
LPUE","SST",sep = ";")),
                                   do_multivariate_E_analysis(data=full_join(bio.Gul
f.yt,phys.Gulf,by="Year"),
                                                             pred_col = 'JAI',lag_c
ol = 'LPUE',other_col='SST',
                                                             first_column_time = TR
UE, E = E.list) %>%
                                      mutate(species = "Gulf", pred_vars=paste("LPU
E","SST",sep = ";")),
                             do_multivariate_E_analysis(data=full_join(bio.Gulf.yt,
phys.Gulf,by="Year"),
                                                             pred_col = 'JAI',lag_c
ol = 'LPUE',other_col='SLP',
                                                             first_column_time = TR
UE, E = E.list) %>%
                                      mutate(species = "Gulf", pred_vars=paste("LPU
E","SLP",sep = ";")))
                             )
```

Make some plots.

```
h_mex2 <- vector(mode='list',2)
plot_spec <- list(list('Atlantic','LPUE;SST'),list('Gulf','LPUE;SLP'))
for(i_plot in 1:length(plot_spec)){
    species_i <- plot_spec[[i_plot]][[1]]
    predvars_i <- plot_spec[[i_plot]][[2]]
    df.i <- results_multi_ex2 %>%
        filter(species == species_i) %>%
        filter(pred_vars == predvars_i)

    p <- ggplot(df.i, aes(x = E))

    p <- p + geom_line(aes(y = pmax(0,rho), colour = "rho"))
    p <- p + geom_line(aes(y = 1 - pmin(1,mae), colour = "MAE"))

    # add  constant predictors
    p <- p + geom_line(aes(y = pmax(0,const_pred_rho), colour = "rho"),lty=2,lwd=
0.75)
    p <- p + geom_line(aes(y = 1 - pmin(1,const_pred_mae), colour = "MAE"),lty=2,
lwd=0.75)



    # now adding the secondary axis, following the example in the help file ?scal
e_y_continuous
    # and, very important, reverting the above transformation
    p <- p + scale_y_continuous(limits = c(-0.01,.75),sec.axis = sec_axis(~1-., n
ame = "MAE"))



    # modifying colours and theme options
    p <- p + scale_colour_manual(values = c("blue", "red"),labels=expression(rho,
"MAE"))
    p <- p + labs(y = expression(rho),
                  x = "lags of LPUE (E)",
                  title = species_i,
                  colour = "Parameter")
    p <- p + theme(legend.position = c(0.8, 0.2)) + theme_bw()

    h_mex2[[i_plot]] <- p
  }


do.call(grid.arrange,c(h_mex2,nrow = 1))
```



We can also make plots that overlay the two multivariate experiments (EX1: only lags of LPUE, EX2: include environment).

```
h_mex2 <- vector(mode='list',2)
plot_spec <- list(list('Atlantic','LPUE;SST'),list('Gulf','LPUE;SLP'))
for(i_plot in 1:length(plot_spec)){
    species_i <- plot_spec[[i_plot]][[1]]
    predvars_i <- plot_spec[[i_plot]][[2]]
    df.i.ex1 <- results_multi_ex1 %>%
        filter(species == species_i) %>%
        filter(pred_vars == 'LPUE')

    df.i.ex2 <- results_multi_ex2 %>%
        filter(species == species_i) %>%
        filter(pred_vars == predvars_i)


    p <- ggplot(df.i.ex2, aes(x = E))
    p <- p + geom_line(aes(y = pmax(0,rho), colour = "rho"))

    # add the MAE line, transformed to match roughly the range of the temperature
    p <- p + geom_line(aes(y = 1 - pmin(1,mae), colour = "MAE"))

    # add  constant predictors
    p <- p + geom_line(aes(y = pmax(0,rho),colour = "rho"), data = df.i.ex1 ,lty=
2,lwd=0.75)
    p <- p + geom_line(aes(y = 1 - pmin(1,mae),colour = "MAE"), data = df.i.ex1,
,lty=2,lwd=0.75)



    # now adding the secondary axis, following the example in the help file ?scal
e_y_continuous
    # and, very important, reverting the above transformation
    p <- p + scale_y_continuous(limits = c(-0.01,.75),sec.axis = sec_axis(~1-., n
ame = "MAE"))



    # modifying colours and theme options
    p <- p + scale_colour_manual(values = c("blue", "red"),labels=expression(rho,
"MAE"))
    p <- p + labs(y = expression(rho),
                  x = "lags of LPUE (E)",
                  title = species_i,
                  colour = "Parameter")
    p <- p + theme(legend.position = c(0.8, 0.2)) + theme_bw()

    h_mex2[[i_plot]] <- p
  }
```

```
do.call(grid.arrange,c(h_mex2,nrow = 1))
```