Creating a reference population

22 November, 2017

# Overview

A reference population of known pallid sturgeon abundance was needed to evaluate varying monitoring designs and their ability to achieve the fundamental objectives identified during the first population assessment program workshop (See Section ??). The reference population provides data on the annual survival and movement of each fish in the population (including new recruits) at the bend-level. Since fish length may influence sampling effectiveness (e.g. gear selectivity), data on individuals (e.g., growth) was also simulated.

### Objectives

The objectives of this analysis were to: 1. Create a spatially explicit pallid sturgeon population with known bend-level annual abundances, and 2. Simulate reference population dynamics for 10 years with varying but known annual survival, growth, movement, and recruitment.

### Major Assumptions

1. Survival probability is homogeneous among all individuals and independent of space and time.
2. Individual fish growth follows a von Bertalanffy growth equation.
3. There is no movement within years.
4. Movement within a basin can occur between years.
5. Recruitment occurs at the basin-level and is stochastic.

Assumptions 1 and 2 are reasonable simplifications for the purposes of this evaluation and have been used in various fish studies (CITATIONS...NOT SURE IF THIS IS TRUE; NEED TO ASK MIKE AND LOOK AT LITERATURE). Assumptions 3 and 4 match the assumptions of many closed population estimators from mark-recapture estimation to robust design estimation. Making these assumptions allows for the evaluation of such estimators under the best possible movement conditions---if an estimator performs poorly under the best movement conditions, then its use in less favorable movement conditions is further questionable. Lastly, since recruitment is expected to vary from year to year and fish are expected to migrate to spawning habitats within basin (and not necessarily stay in particular bends), assumption 5 is appropriate for studying pallid sturgeon. For a more detailed examination of these assumptions see the "Discussion" section below.

# Methods

### Spatial extent and grain

This analysis was constrained to generate reference pallid sturgeon populations for segments and bends within the upper basin (segments 1-4) and lower basin (segments 7-10, 13 and 14) of the Missouri River. Bends of varying length were used to generate the spatial distribution of pallid sturgeon (Table 1).

### Population Initialization

The pallid sturgeon reference population was initialized using data from the PSPAP database and the pallid sturgeon literature. Population initialization required xx steps: 1. initializing bend-level abundance, 2. initializing lengths and growth parameters of individuals within each bend, 3. assigning demographic rates (i.e., sex, origin).

#### Bend Abundance

We populated each river bend with pallid sturgeon based on expected segment-level densities by origin (Table 2). The number of pallid sturgeon assigned to a bend was stochastically generated from a Poisson distribution given the segment density and bend length as:

where

* is the number of pallid sturgeon within each bend,
* is the segment and origin specific density in fish per RKM,
* is bend length in RKM,
* indexes segment within universe,
* indexes bend within segment, and
* indexes whether pallid sturgeon were hatchery or natural origin.

Generating bend abundances from a Poisson distribution allows for variation in bend densities among bends within segment while guaranteeing that the bend is populated with a whole number of fish. Additionally, one could derive a spatial Poisson distribution with the following assumptions:

1. The probability of 1 pallid being in a column of water with length and the width of the river at RKM is .
2. The probability of 2 or more pallid sturgeon in a column of water w/ length is negligible .
3. The number of pallid sturgeon in disjoint (non-overlapping) water columns and are independent. In other words the presence or absence of a fish in water column 1 does not effect the probability of a fish being present in water column 2, even if these two water columns are adjacent to each other.

#### Assigning length and growth parameters to individuals

We randomly assigned an initial length to each individual fish. Initial lengths were generated from a segment specific distribution constructed from recent (2015-??) length data in the PSPAP database. After removing outliers. (DESCRIBE FURTHER, CHAT WITH MIKE ABOUT REASONING.)

Additionally, we assigned individual von Bertalanffy growth parameters and , or the asymptotic length and Brody growth coefficient of a fish's growth trajectory, respectively, to each fish. Growth parameters were generated from a basin specific bivariate normal distribution fitted to the length data in the PSPAP database. (FURTHER DESCRIPTION HERE; NEED MIKE TO DO SINCE NOT SURE HOW DONE)

To avoid unrealistic growth parameter values we truncated each bivariate normal distribution (one for each basin) to the middle 80%. All points of the truncated distribution are contained on or within an ellipse centered at , the basin specific mean values for and . Since the R package truncates multivariate normal distributions to rectangular regions, as opposed to elliptical regions, we used a different approach to randomly draw points from the desired elliptical region.

In short, we first drew points from the middle 80% of a standard bivariate normal distribution and then transformed these points to the bivariate normal distribution of interest, giving us randomly drawn points that fall on or within the 80% ellipse. The truncation process was achieved using twice in R, once for the normal distribution associated with and then again for the normal distribution associated with conditional on . For more details see Appendix ??.

### Simulating Annual Transitions

After initializing the reference population, we track all individual fish for 10 years, recording individual survival status, bend location, and length on a yearly basis. Additionally, new recruits may be added to the population and tracked.

#### Survival & Growth

Survival is stochastic but simulated with a survival probability that is homogeneous among individuals with all fish in the river having an equal probability, , of surviving the year. Survival probability is independent of individual age, length, sex, origin, location, and year. In other words, each year individual fish survival is a Bernoulli trial with fixed probability of success, .

Annual growth is projected by individual von Bertalanffy growth curves as

where is the individual's length during year and each fish has its own randomly generated asymptotic length, , and Brody growth rate, , as described in the previous section on initializing the population.

#### Movement

Within basin movement occurs between years and is considered at the bend-level. Pallid sturgeon may move from one bend to another bend within the same segment, from one bend to another bend within a different segment (but same basin), or stay within the same bend. Movement probabilities are based on current bend location with the probability of being in bend the following year increasing as the distance between the fish's current bend and bend decreases. In particular,

and

where is the distance in RKM from the center of bend to the center of bend and and are basin specific parameters.

(NEED JUSTIFICATION FOR THIS...CHAT WITH MIKE/LOOK INTO LITERATURE)

To account for the uncertain nature of movement probabilities, we vary and from simulation to simulation, allowing the analysis of populations that range in having high site fidelity to low site fidelity.

#### Recruitment

Recruitment occurs randomly with a fixed expected frequency (e.g. once every 3 years). During years when recruitment occurs the number of recruits is drawn from a basin specific Poisson distribution.  
Each new recruit is tracked after being randomly assigned an age-0 location within basin, an age-0 length of 200mm, and von Bertalanffy growth parameters. Age-0 bend locations are generated from a discrete uniform distribution that includes all bends within basin, while growth parameters are generated from the same basin specific truncated bivariate normal distribution used during the initialization of the population. Initial length was chosen as 200mm, because (REASON HERE). Despite fixed initial lengths, the growth trajectories of recruits will differ due to variation in their individual growth parameters.

# Discussion

There were several assumptions made in the simulation of the reference populations. While the major assumptions are reasonable for the given objectives, a further discussion of some assumptions

Additionally, since it is unclear how closely assumptions 3 and 4 are met in reality, we explore and discuss the consequences of breaking these movement assumptions on estimator outputs in Appendix ??.

# Figures

FUNCTIONAL IMAGE HERE

# Tables

Table 1. Summary of bends within PSPAP Missouri river segments.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Basin | Segment | Number of bends | Minimum length (km) | Mean length (km) | Maximum length (km) |
| UB | 1 | 1 | 7.89 | 7.9 | 7.9 |
|  | 2 | 40 | 0.64 | 2.2 | 3.9 |
|  | 3 | 91 | 0.64 | 2.0 | 7.1 |
|  | 4 | 24 | 0.97 | 3.3 | 8.0 |
| LB | 7 | 34 | 0.16 | 2.6 | 7.9 |
|  | 8 | 61 | 1.13 | 4.0 | 9.2 |
|  | 9 | 80 | 1.13 | 4.4 | 11.7 |
|  | 10 | 39 | 1.61 | 4.7 | 10.3 |
|  | 13 | 45 | 1.77 | 4.1 | 10.6 |
|  | 14 | 56 | 1.45 | 3.9 | 19.0 |

Table 2. Segment and river level densities by origin: hatchery (H) and natural/wild (W). Minimum and maximum densities were taken from data across a few recent years (when available), while mean densities are those reported in the literature from the most recent year's data.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Basin | Segments | Fish Type | Minimum Density (fish/rkm) | Maximum Density (fish/rkm) | Mean Density (fish/rkm) | Most Recent Year | References |
| UB | 1-4 | H | 91.57 | 91.57 | 91.57 | 2013 | [1] |
|  |  | W | 0.28 | 0.37 | 0.28 | 2008\* | [2]-[4] |
| LB | 7-9 | H | 28.62 | 32.30 | 32.30 | 2010 | [5] |
|  |  | W | 5.43 | 8.88 | 5.70 | 2010 | [5] |
|  | 10, 13, 14 | H | 5.53 | 10.17 | 5.53 | 2013 | [6] |
|  |  | W | 0.56 | 0.93 | 0.93 | 2013 | [6] |

\*Estimated year of data collection based on reference date.

# Code

Box 1. Function used to simulate reference populations given pallid sturgeon density, survival, and spatial structure.

## function(inputs,...)  
## {  
## segs<-inputs$segs  
## bends<-inputs$bends  
## fish\_density<-inputs$fish\_density  
## nyears<-inputs$nyears  
## phi<-inputs$phi  
## initial\_length<-inputs$initial\_length  
## mv\_beta0<-inputs$mv\_beta0  
## mv\_beta1<-inputs$mv\_beta1  
## dis<- inputs$dis  
## #direct<- inputs$direct #UNUSED  
##   
## # this function allocates fish to bends within a segment  
## # probabilistically given bend weights and determines the survival  
## # of fish within bend over nyears given survival probabilities  
##   
## # inputs  
## ## segment: segment [1,2,3,4,7,8,9,10,13,14]  
## ## fish\_density: initial density data from load-and-clean; fish/rkm  
## ## type: input for fish type [hatchery, natural]  
## ## bends: bend data from load-and-clean  
## ## nyears: number of years to simulate the population for  
## ## phi: a matrix of survival probabilities by segment (rows) and   
## ## year (cols)  
## ## lower: a list of growth values for the lower basin:  
## ## $ln\_Linf\_mu: mean ln(Linf) value for growth equation  
## ## $ln\_k\_mu: mean ln(k) value for growth equation  
## ## $vcv: variance and covariances for Linf and k on a natural log scale  
## ## upper: a list of growth values for the upper basin:  
## ## $ln\_Linf\_mu: mean ln(Linf) value for growth equation  
## ## $ln\_k\_mu: mean ln(k) value for growth equation  
## ## $vcv: variance and covariances for Linf and k on a natural log scale  
## ## initial\_length: functions to simulate initial length given an  
## ## empirical distribution of segment specific lengths  
## ## mv\_beta0:  
## ## mv\_beta1:  
## ## dis:  
##   
## # outputs  
## ## out: a list of 3 objects:  
## ## $out: a matrix where each row is a bend and   
## ## each column represents a year; number  
## ## $bendMeta: a dataframe including the information in "bends"  
## ## with expected initial segment density (from init\_dens),   
## ## initial bend abundance, and segment index columns added  
## ## $individual\_meta: a dataframe including the information in "bendMeta"  
## ## but expanded to include one row for each individual fish  
## ## $Z: a matrix with entries for individual survival status (0=Dead,  
## ## 1=Alive) where each row represents a fish living and each column  
## ## represents a year   
## ## $BND: a matrix where each row is an individual fish (matching up with  
## ## the entries from "individual\_meta"), each column is a year, and each  
## ## entry is a number from 1 to 471 which indicates the bend the fish was  
## ## living in during the given year; NAs are entered for dead fish  
## ## $l: a matrix where each row is an individual fish (matching up with  
## ## the entries from "individual\_meta"), each column is a year, and each  
## ## entry is the length of the fish during the given year; NAs are entered  
## ## for dead fish  
## ## $inputs: a list of the inputs used to create the reference population  
##   
##   
## # assumptions  
## ## no movement within years  
## ## no recruitment  
## ## survival homogenous for individuals  
##   
## ## ERROR HANDLING  
## ### PHI  
## if(dim(phi)[1]!=length(segs) |  
## dim(phi)[2]!=nyears-1)  
## {return(print("Survival(phi) needs to be a matrix \n  
## with rows equal to the number of segments \n  
## and the same number of columns as years-1 to simulate"))}  
## ### INITIAL FISH DENSITY  
## if(nrow(fish\_density)!=length(segs))  
## {return(print("Initial fish density (fish\_density) needs to be \n  
## a dataframe of densities by segment with number of \n  
## rows equal to the number of segments."))}  
## ### GROWTH  
## if(dim(inputs$lower$vcv)[1]!=2|dim(inputs$lower$vcv)[2]!=2)  
## {return("The lower variance covariance \n  
## matrix needs to be 2x2 square")}  
## if(dim(inputs$upper$vcv)[1]!=2|dim(inputs$upper$vcv)[2]!=2)  
## {return("The upper variance covariance \n  
## matrix needs to be 2x2 square")}  
## #if(dim(vbgf\_vcv)[1]!=2|dim(vbgf\_vcv)[2]!=2){return("The variance covariance \n  
## # matrix needs to be 2x2 square")}  
## #if(dim(vbgf\_vcv)[3]!=10){return("There needs to be 10 2x2 variance covariance \n  
## # matrices for each segment")}  
## #if(length(Linf)!=10){return("Linf needs to be a vector of 10 values for each segment")}  
## #if(length(k)!=10){return("k needs to be a vector of 10 values for each segment")}  
## ## END: ERROR HANDLING   
##   
##   
## # GET BEND INFORMATION  
## tmp<- subset(bends, b\_segment %in% segs)  
## tmp<- tmp[order(tmp$id),]## CRITICAL  
## bends\_in\_segs<-aggregate(bend\_num~b\_segment,tmp,length)   
## bends\_in\_segs$phi\_indx<-1:nrow(bends\_in\_segs)  
##   
## tmp<-merge(tmp, bends\_in\_segs[,-2],by="b\_segment",all.x=TRUE)  
##   
## # ADD INITIAL DENSITIES TO TMP  
## tmp<-merge(tmp, fish\_density,by=c("rpma", "b\_segment"),all.x=TRUE)  
##   
## # INITIAL ABUNDANCES  
## ## PULL NUMBER FROM A POISSON AFTER ADJUSTING  
## ## DENSITY FOR BEND SIZE  
## tmp$N\_ini<-rpois(nrow(tmp),  
## lambda=tmp$expected\_dens\*tmp$length.rkm)  
## tmp<-tmp[order(tmp$id),] #CRITICAL  
## ## ASSIGN A BEND TO EACH INVIDUAL   
## ### EXPAND BENDS FOR EACH FISH  
## individual\_meta<- as.data.frame(  
## lapply(tmp,function(x) rep(x,tmp$N\_ini)))  
## ### ADD INDIVIDUAL FISH ID  
## individual\_meta$fish\_id<-1:nrow(individual\_meta)  
## ## ASSIGN GROWTH PARAMETERS TO EACH INDIVIDUAL  
## ln\_Linf<-ifelse(segs %in% c(1:4), inputs$upper$ln\_Linf\_mu,  
## inputs$lower$ln\_Linf\_mu) # a vector of mean Linf values, one entry for each segment  
## ln\_k<-ifelse(segs %in% c(1:4), inputs$upper$ln\_k\_mu, inputs$lower$ln\_k\_mu)  
## # a vector of mean k values, one entry for each segment  
## ln\_B<-array(0,dim=c(2,2,length(segs)))  
## for(i in 1:length(segs))  
## {  
## if(segs[i] %in% c(1:4))   
## {  
## ln\_B[,,i]<-eigen(inputs$upper$vcv)$vectors%\*%  
## matrix(c(sqrt(eigen(inputs$upper$vcv)$values[1]),0,0,  
## sqrt(eigen(inputs$upper$vcv)$values[2])),2,2)  
## }  
## if(segs[i] %in% c(7:10,13,14))  
## {  
## ln\_B[,,i]<-eigen(inputs$lower$vcv)$vectors%\*%  
## matrix(c(sqrt(eigen(inputs$lower$vcv)$values[1]),0,0,  
## sqrt(eigen(inputs$lower$vcv)$values[2])),2,2)  
## }  
## } # an array of variance and covariances for Linf and k  
## # each matrix is for a segment  
## ln\_vals<-lapply(1:nrow(individual\_meta),function(m)  
## {#DRAWN FROM MIDDLE 80% OF BIVARIATE NORMAL (ELLIPSE)  
## z1<-rtruncnorm(1, -sqrt(-2\*log(0.2)), sqrt(-2\*log(0.2)), mean=0, sd=1)  
## z2<-rtruncnorm(n=1, a=-sqrt(-2\*log(0.2)-z1^2), b=sqrt(-2\*log(0.2)-z1^2),   
## mean=0, sd=1)  
## X<-t(ln\_B[,,individual\_meta$phi\_indx[m]]%\*%c(z1,z2)+  
## c(ln\_Linf[individual\_meta$phi\_indx[m]],ln\_k[individual\_meta$phi\_indx[m]]))  
## return(X)  
## })  
## ln\_vals<-do.call("rbind",ln\_vals)  
## individual\_meta$Linf<-exp(ln\_vals[,1])   
## individual\_meta$k<-exp(ln\_vals[,2])  
## individual\_meta$yr\_ini<-0  
##   
## ## Z: INDIVIDUAL SURVIVAL MATRIX WHERE EACH ROW IS A SINGLE FISH   
## ### IN THE GIVEN BEND AND EACH COLUMN IS A YEAR (0=Dead, 1=Alive)   
## Z<-matrix(0,nrow=nrow(individual\_meta),ncol=nyears)   
## Z[,1]<-1  
##   
## ## l: LENGTH FOR INDIVIDUALS  
## l<-matrix(0,nrow=nrow(individual\_meta),ncol=nyears)  
## for(i in unique(individual\_meta$b\_segment))  
## {  
## ## MAKE A QUICK FUNCTION OF THE INVERSE CUMULATIVE DISTRIBUTION  
## l\_ini<-approxfun(  
## initial\_length[initial\_length$segment==i,]$quantile,  
## initial\_length[initial\_length$segment==i,]$val,  
## rule=2)  
## indx<- which(individual\_meta$b\_segment==i)  
## l[indx,1]<-l\_ini(runif(length(indx)))  
## }  
## ### FIX ANY LENGTHS > THAN LINF  
## l[,1]<- ifelse(l[,1]>= individual\_meta$Linf,individual\_meta$Linf\*0.95,l[,1])   
##   
## ## BND: BEND LOCATION FOR INDIVIDUALS   
## BND<-matrix(0,nrow=nrow(individual\_meta),ncol=nyears)   
## BND[,1]<- individual\_meta$id  
##   
## ###############################################################  
## ## POPULATION DYNAMICS  
## ## 1. SUVIVAL  
## ## 2. GROWTH  
## ## 3. MOVEMENT  
## ## 4. RECRUITMENT  
## ###############################################################  
## bends2segs<-ddply(individual\_meta, .(id), summarize,  
## rpma=mean(rpma),  
## b\_id=mean(b\_id),  
## phi\_indx=mean(phi\_indx))  
## #ADD IN BENDS WHICH INITIALLY HAD 0 FISH  
## bends2segs<-merge(data.frame(id=tmp$id[which(tmp$N\_ini==0)],  
## rpma=tmp$rpma[which(tmp$N\_ini==0)],  
## b\_id=tmp$b\_id[which(tmp$N\_ini==0)],  
## phi\_indx=tmp$phi\_indx[which(tmp$N\_ini==0)]),  
## bends2segs,  
## by=c("id","rpma","b\_id","phi\_indx"),all=TRUE)  
##   
##   
## # BEGIN POPULATION SIMULATION   
## ## 4. RECRUITMENT  
## ### HOW MANY RECRUITS AGE-1   
## r\_freq\_upper<-rbinom(nyears,1,1/inputs$upper$r\_freq)  
## r\_freq\_lower<-rbinom(nyears,1,1/inputs$lower$r\_freq)  
## recruits\_upper<- r\_freq\_upper\*rpois(nyears,exp(inputs$upper$r\_beta0))  
## recruits\_lower<- r\_freq\_lower\*rpois(nyears,exp(inputs$lower$r\_beta0))  
## r\_dat<-data.frame(rpma=c(rep(2,nyears),rep(4,nyears)), year=rep(1:nyears,2),  
## r\_year=c(r\_freq\_upper,r\_freq\_lower),   
## age\_0=c(recruits\_upper,recruits\_lower))  
##   
## for(i in 2:nyears)  
## {# loop over each year  
## ### RECRUIT TO POPULATION  
## if(sum(recruits\_upper[i-1],recruits\_lower[i-1])>0)  
## {  
## new\_recruits<- data.frame(  
## rpma=c(rep(2,recruits\_upper[i-1]),rep(4,recruits\_lower[i-1])))  
## ### ASSIGN AN INITIAL SEGMENT AND BEND LOCATION  
## recruit\_loc<- lapply(1:nrow(new\_recruits),function(x)  
## {  
## bend<- sample(tmp[which(tmp$rpma==new\_recruits$rpma[x]),]$b\_id,1)  
## #MAKE PROBABILITY BASED ON BEND LENGTH, LOCATION, IRC HABITAT AVAILABILITY ETC???  
## segment<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$b\_segment  
## phi\_indx<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$phi\_indx  
## id<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$id  
## return(list(b\_id=bend,b\_segment=segment,phi\_indx=phi\_indx, id=id))  
## })  
## recruit\_loc<-as.data.frame(do.call("rbind",recruit\_loc))  
## new\_recruits$b\_segment<- unlist(recruit\_loc$b\_segment)  
## new\_recruits$b\_id<- unlist(recruit\_loc$b\_id)  
## new\_recruits$phi\_indx<- unlist(recruit\_loc$phi\_indx)  
## new\_recruits$id<-unlist(recruit\_loc$id)  
##   
## ### ASSIGN GROWTH PARAMETES FOR NEW RECRUITS  
## ln\_vals<-lapply(1:nrow(individual\_meta),function(m)  
## {#DRAWN FROM MIDDLE 80% OF BIVARIATE NORMAL (ELLIPSE)  
## z1<-rtruncnorm(1, -sqrt(-2\*log(0.2)), sqrt(-2\*log(0.2)), mean=0, sd=1)  
## z2<-rtruncnorm(n=1, a=-sqrt(-2\*log(0.2)-z1^2), b=sqrt(-2\*log(0.2)-z1^2),   
## mean=0, sd=1)  
## X<-t(ln\_B[,,individual\_meta$phi\_indx[m]]%\*%c(z1,z2)+  
## c(ln\_Linf[individual\_meta$phi\_indx[m]],ln\_k[individual\_meta$phi\_indx[m]]))  
## return(X)  
## })  
## ln\_vals<-do.call("rbind",ln\_vals)  
## new\_recruits$Linf<-exp(ln\_vals[,1])  
## new\_recruits$k<-exp(ln\_vals[,2])   
## new\_recruits$fish\_id<-(max(individual\_meta$fish\_id)+1):(max(individual\_meta$fish\_id)+sum(recruits\_upper[i-1],recruits\_lower[i-1]))  
## new\_recruits$yr\_ini<-i-1  
## individual\_meta<-rbind.fill(individual\_meta,new\_recruits)   
##   
## ## UPDATE MATRICES: Z, L, BND  
## ### MATRICES TO APPEND TO OTHER  
## BND\_recruits<-l\_recruits<-Z\_recruits<- matrix(0,nrow=sum(recruits\_upper[i-1],recruits\_lower[i-1]),ncol=nyears)  
##   
## # ALIVE OR NOT  
## Z\_recruits[,i-1]<-1 # had to be alive in previous year to recruit  
## Z<-rbind(Z,Z\_recruits)  
##   
## # LENGTH AT AGE 0  
## l\_recruits[,i-1]<-200 ## 250mm calibrates to ~ 325 mm at age-1  
## l<-rbind(l,l\_recruits)  
## ## ADD VARIATION IN AGE 0 SIZE???  
##   
## ## BEND  
## BND\_recruits[,i-1]<-new\_recruits$id  
## BND<-rbind(BND,BND\_recruits)  
## }  
##   
## for(m in 1:nrow(Z))  
## {# loop over individuals  
## #INDEX FOR LOCATION OF FISH IN PREVIOUS TIME STEP   
## seg\_indx<-ifelse(Z[m,i-1]>0,bends2segs$phi\_indx[which(  
## bends2segs$id==BND[m,i-1])],1)# Using 1 when FALSE is   
## # arbitrary and only a placeholder  
## # since fish is dead at this point  
## b\_indx<-ifelse(Z[m,i-1]>0,bends2segs$b\_id[which(  
## bends2segs$id==BND[m,i-1])],1)# Using 1 when FALSE is   
## # arbitrary and only a placeholder  
## # since fish is dead at this point  
## ## 1. SURVIVAL  
## Z[m,i]<- rbinom(1,  
## size=1,  
## prob=phi[seg\_indx,i-1]\*Z[m,i-1])  
## ## 2. FABENS MODEL FOR GROWTH (VBGF)  
## l[m,i]<-(l[m,i-1] + (individual\_meta$Linf[m]-l[m,i-1])\*(1-exp(-individual\_meta$k[m])))\*Z[m,i]  
## # 0 growth if dead   
## ## 3. MOVEMENT MODEL  
## if(individual\_meta$rpma[m]==2 & Z[m,i]>0)  
## {  
## y<- exp(mv\_beta0[1]+  
## mv\_beta1[1]\*dis$rpma2[b\_indx,])  
## y[which(dis$rpma2[b\_indx,]==0)]<-1  
## p<- y/sum(y)  
## BND[m,i]<- sample(x=subset(tmp,rpma==2)$id,  
## size=1,  
## prob=p)  
## } # end if   
## if(individual\_meta$rpma[m]==4 & Z[m,i]>0)# needs to be alive  
## {  
## y<- exp(mv\_beta0[2]+  
## mv\_beta1[2]\*dis$rpma4[b\_indx,])  
## y[which(dis$rpma4[b\_indx,]==0)]<-1  
## p<- y/sum(y)  
## BND[m,i]<- sample(x=subset(tmp,rpma==4)$id,  
## size=1,  
## prob=p)  
## } # end if  
## }  
## } # END POPOULATION SIMULATION  
##   
## ## AGE 0's IN FINAL YEAR  
## if(sum(recruits\_upper[nyears],recruits\_lower[nyears])>0)  
## {  
## ### ASSIGN A LENGTH AND GROWTH PARAMETERS  
## new\_recruits<- data.frame(  
## rpma=c(rep(2,recruits\_upper[nyears]),rep(4,recruits\_lower[nyears])))  
## ### ASSIGN A SEGMENT AND BEND  
## recruit\_loc<- lapply(1:nrow(new\_recruits),function(x)  
## {  
## bend<- sample(tmp[which(tmp$rpma==new\_recruits$rpma[x]),]$b\_id,1)  
## #MAKE PROBABILITY BASED ON BEND LENGTH, LOCATION, IRC HABITAT AVAILABILITY ETC???  
## segment<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$b\_segment  
## phi\_indx<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$phi\_indx  
## id<- tmp[which(tmp$b\_id==bend & tmp$rpma==new\_recruits$rpma[x]),]$id  
## return(list(b\_id=bend,b\_segment=segment,phi\_indx=phi\_indx, id=id))  
## })  
## recruit\_loc<-as.data.frame(do.call("rbind",recruit\_loc))  
## new\_recruits$b\_segment<- unlist(recruit\_loc$b\_segment)  
## new\_recruits$b\_id<- unlist(recruit\_loc$b\_id)  
## new\_recruits$phi\_indx<- unlist(recruit\_loc$phi\_indx)  
## new\_recruits$id<-unlist(recruit\_loc$id)  
##   
## ### GROWTH PARAMETES FOR NEW RECRUITS  
## ln\_vals<-lapply(1:nrow(individual\_meta),function(m)  
## {#DRAWN FROM MIDDLE 80% OF BIVARIATE NORMAL (ELLIPSE)  
## z1<-rtruncnorm(1, -sqrt(-2\*log(0.2)), sqrt(-2\*log(0.2)), mean=0, sd=1)  
## z2<-rtruncnorm(n=1, a=-sqrt(-2\*log(0.2)-z1^2), b=sqrt(-2\*log(0.2)-z1^2),   
## mean=0, sd=1)  
## X<-t(ln\_B[,,individual\_meta$phi\_indx[m]]%\*%c(z1,z2)+  
## c(ln\_Linf[individual\_meta$phi\_indx[m]],ln\_k[individual\_meta$phi\_indx[m]]))  
## return(X)  
## })  
## ln\_vals<-do.call("rbind",ln\_vals)  
## new\_recruits$Linf<-exp(ln\_vals[,1])  
## new\_recruits$k<-exp(ln\_vals[,2])   
## new\_recruits$fish\_id<-(max(individual\_meta$fish\_id)+1):(max(individual\_meta$fish\_id)+sum(recruits\_upper[nyears],recruits\_lower[nyears]))  
## new\_recruits$yr\_ini<-nyears  
## individual\_meta<-rbind.fill(individual\_meta,new\_recruits)   
##   
## ## UPDATE MATRICES: Z, L, BND  
## ### MATRICES TO APPEND TO OTHER  
## BND\_recruits<-l\_recruits<-Z\_recruits<- matrix(0,nrow=sum(recruits\_upper[nyears],recruits\_lower[nyears]),ncol=nyears)  
##   
## # ALIVE OR NOT  
## Z\_recruits[,nyears]<-1 # had to be alive in previous year to recruit  
## Z<-rbind(Z,Z\_recruits)  
##   
## # LENGTH AT AGE 0  
## l\_recruits[,nyears]<-200 ## 250mm calibrates to ~ 325 mm at age-1  
## l<-rbind(l,l\_recruits)  
## ## ADD VARIATION IN AGE 0 SIZE???  
##   
## ## BEND  
## BND\_recruits[,nyears]<-new\_recruits$id  
## BND<-rbind(BND,BND\_recruits)  
## }  
##   
## ## PROCESS POPULATION AND RETURN   
## l[l==0]<-NA  
## BND[BND==0]<-NA  
## # MATRIX OF BEND LEVEL ABUNDANCES TO RETURN  
## out<-aggregate(Z[,1],  
## by=list(id=BND[,1]),  
## sum)  
## names(out)[2]<-"yr\_1"  
## for(i in 2:nyears)  
## {  
## app<-aggregate(Z[,i],  
## by=list(id=BND[,i]),  
## sum)  
## names(app)[2]<-paste("yr",i,sep="\_")  
## out<-merge(out,app, by="id", all=TRUE)  
## }  
## ## IN CASE THERE WERE BENDS WITH NO FISH EVERY YEAR  
## out<-merge(out,data.frame(id=tmp$id), by="id", all=TRUE)  
## out[is.na(out)]<-0 # NAs for no fish in bend  
## if(nrow(out)!=nrow(tmp))  
## {  
## return(print("ERROR IN BEND ABUNDANCE MERGE"))  
## } #ERROR HANDLING FOR DOUBLE CHECKING  
## out<-out[order(out$id),]  
## #if(is.null(Linf)){l<-0}  
## out<-list(out=as.matrix(out[,-c(1)]),   
## bendMeta=tmp,  
## individual\_meta=individual\_meta,  
## Z=Z,  
## BND=BND,  
## l=l,  
## r\_dat=r\_dat,  
## inputs=inputs)  
## return(out)# return relevant stuff  
## }