```
# Functions
# by: Mikaela Provost, modified from Lewis
# last edited: 2017-9-20



assemble_leslie <- function(SR = 'Beverton_Holt', mat_a_array = mat, W_a_array = W, year = j, n = n, H = H,
                            W_rec = W_rec, W_rec_sd = W_rec_sd, W_esc = W_esc, W_esc_sd = W_esc_sd, M = M,
                            f_W_slope = f_W_slope, alpha = alpha, beta = beta, eta = eta) {

  # Generate empty a_max by a_max matrix to put survivals and fertilities into
  A = matrix(0, a_max, a_max)

  # calculate fishery selectivity by size, sigmoid or dome-shaped selectivity commented out
  # because large growth was leading to catches going to zero with too many escaping catch
  Sel_W <- pnorm(W_a_array[,year], W_rec, W_rec_sd) #- pnorm(W_a_array[,year], W_esc, W_esc_sd)

  # harvest at size OR AGE given harvest rate H
  H_W = Sel_W * H

  # total mortality = Z, which is sum of natural and harvest mortalities at age, given noise in M generated by M()
  #Z = M() + H_W
  # NEW FORM- draw Z from preallocated series
  Z = M[year] + H_W

  # survival from t to t+1 as function of Z
  p = exp(-Z[-a_max])

  # store catch C in terms of biomass harvested for time j
  C = sum((n[,year-1] * W_a_array[,year]) * (1 - exp(-Z)) * (H_W / Z))

  # insert values of survival transition probabilities p[i] for all ages aside from age 1 (recruits)
  # into  subdiagonal of matrix A
  for(i in 2:a_max-1){
    A[i+1,i] = p[i]
  }

  # caclulate fecundity at age from fecundity at current size for each cohort and take product of fecundity
  # and proportion mature to get realized fertility m
  m = f_W_slope * W_a_array[,year] * mat_a_array[,year]


  # Following Botsford et aW_a_array 2014 and previous cohort resonance papers, use a pre-breeding census birth pulse
approach
  # as in Levin and Goodyear 1980, where F(i) = l(1) * m(i), where l(1) is stock-recruitment function and off-
diagonal
  # entries are just p(i)'s

  # calculate and store E, total eggs or early-stage larvae in year j
  E = sum(m * n[,year-1])

  if(SR == 'Beverton_Holt') {

    # Beverton-holt density-dependence, where R(t+1) = (E(t) * alpha * exp(-eta)) * exp(-beta * E(t)),
    # define f[i]'s
    #A[1,1:a_max] <- ((alpha * exp(eta())) / (1 + (beta * E))) * m
    # NEW FORM- draw from preallocated time series of forcing in recruitment
    A[1,1:a_max] <- ((alpha * exp(eta[year])) / (1 + (beta * E))) * m

    # currently only alternative is Ricker, but should build in more options
  } else {

    #A[1,1:a_max] <- ((alpha * exp(eta())) * exp(beta * E)) * m
    # NEW FORM- draw from preallocated time series of forcing in recruitment
    A[1,1:a_max] <- ((alpha * exp(eta[year])) * exp(beta * E)) * m

  }
  # return list including leslie matrix A, Egg production E, catch C
  return(list(A=A, E=E, C=C))
}




is_mature <- function(mat_function = 'logistic', mat_a_array = mat, W_a_array = W, age = i,
                        year = j, location = W_mat, scale = W_mat_sd * (sqrt(3)/pi)) {

    # # logistic ogive, given scale parameter for slope...should be set by user given data
    if(mat_function == 'logistic') {
      if(mat_a_array[age-1,year-1] >= plogis(W_a_array[age,year], location, scale)) {
        mat_a_array[age,year] <- mat_a_array[age-1,year-1]
      }
      else mat_a_array[age,year] <- plogis(W_a_array[age,year], location, scale)

    } else {
```

```
    # knife-edged
    if(mat_a_array[age-1,year-1] == 1) mat_a_array[age,year] <- 1
    else mat_a_array[age,year] <- as.numeric(W_a_array[age,year] >= W_mat)
  }

  return(mat_a_array[age,year])
}
```