Week1_lab

Juliette Verstaen

January 9, 2019

```
library(devtools)
devtools::install_github("BruceKendall/PVA")
```

```
## Skipping install of 'PVA' from a github remote, the SHA1 (fa5d2aed) has not changed since last insta
##   Use `force = TRUE` to force installation
```

```
devtools::install_github("BruceKendall/mpmtools")
```

```
## Skipping install of 'mpmtools' from a github remote, the SHA1 (ff1984db) has not changed since last 
##   Use `force = TRUE` to force installation
```

1. Enter Matrix model

creating matrix

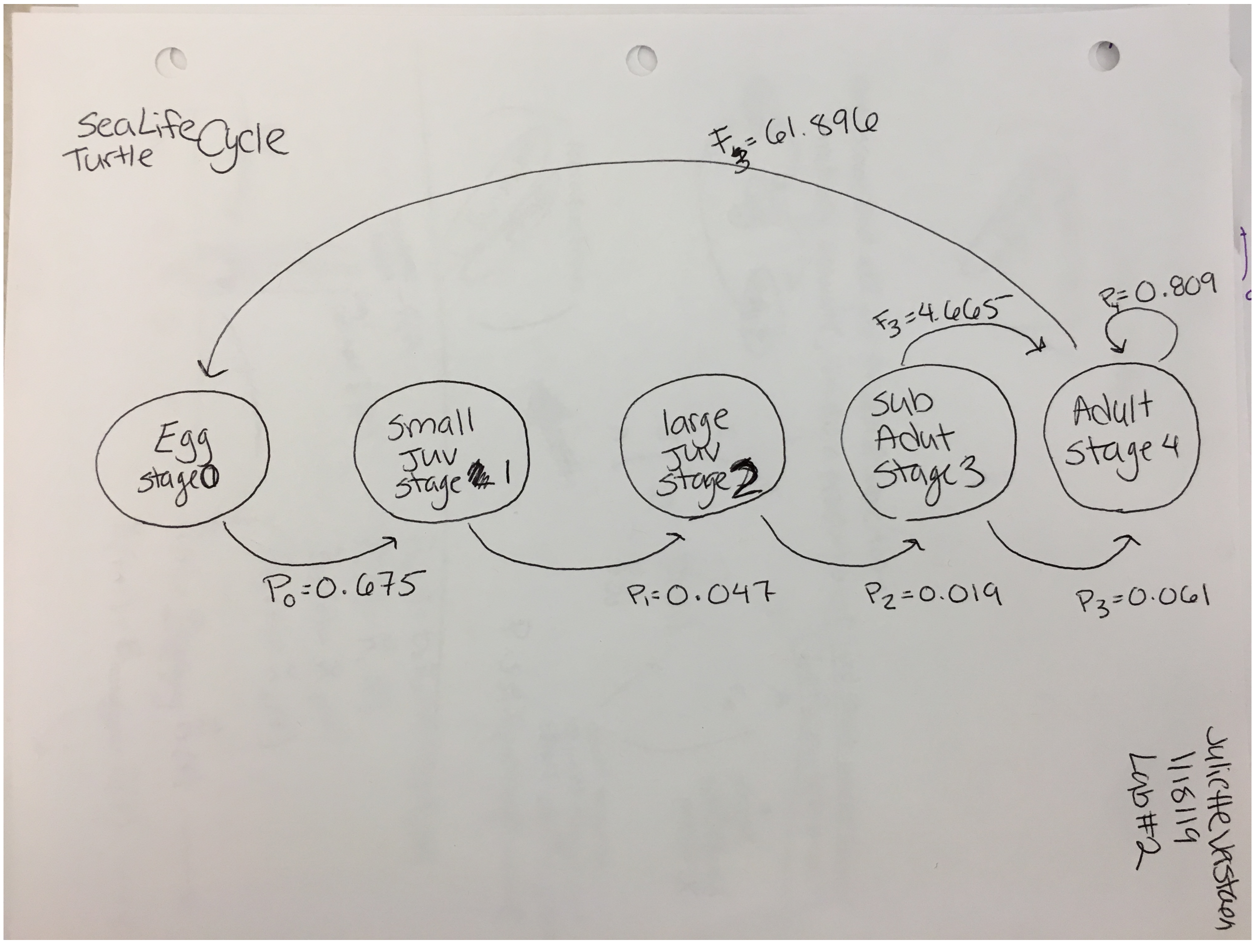
```
A <- matrix(c(0,     0,     0,     4.665, 61.896,
              0.675, 0.703, 0,     0,      0,
              0,     0.047, 0.657, 0,      0,
              0,     0,     0.019, 0.682,  0,
              0,     0,     0,     0.061,  0.809),
          nrow = 5, ncol = 5, byrow = TRUE)
```

giving matrix row and/or column names below is naming by column, if wanted to name by row would have to use byrow=FALSE

```
class_names <- c("Egg", "Sm Juv", "Lg Juv", "Subadult", "Adult")
A <- matrix(c(0,     0,     0,     4.665, 61.896,
              0.675, 0.703, 0,     0,      0,
              0,     0.047, 0.657, 0,      0,
              0,     0,     0.019, 0.682,  0,
              0,     0,     0,     0.061,  0.809),
          nrow = 5, ncol = 5, byrow = TRUE, dimnames = list(class_names, class_names))
```

Life Cycle

```
library(knitr)
include_graphics("life_cycle.png")
```



1.1 Matrix Conventions

- RC Cola: indexed first by *R*ow and then by *C*olumn
- *A* capital bold letters are matrices and *n* lower case are vectors
- use brackets to return the elements you want to look at. example = A[4,3] element in 4th row and 3rd column

```
A
```

```
##             Egg Sm Juv Lg Juv Subadult  Adult
## Egg       0.000  0.000  0.000    4.665 61.896
## Sm Juv    0.675  0.703  0.000    0.000  0.000
## Lg Juv    0.000  0.047  0.657    0.000  0.000
## Subadult  0.000  0.000  0.019    0.682  0.000
## Adult     0.000  0.000  0.000    0.061  0.809
```

```
A[4,3]
```

```
## [1] 0.019
```

```
A[,3]
```

```
##      Egg   Sm Juv   Lg Juv Subadult    Adult
##    0.000    0.000    0.657    0.019    0.000
```

```r
A[4,]
```

```
##       Egg   Sm Juv   Lg Juv Subadult    Adult
##     0.000    0.000    0.019    0.682    0.000
```

```r
A[, c(3,5)]
```

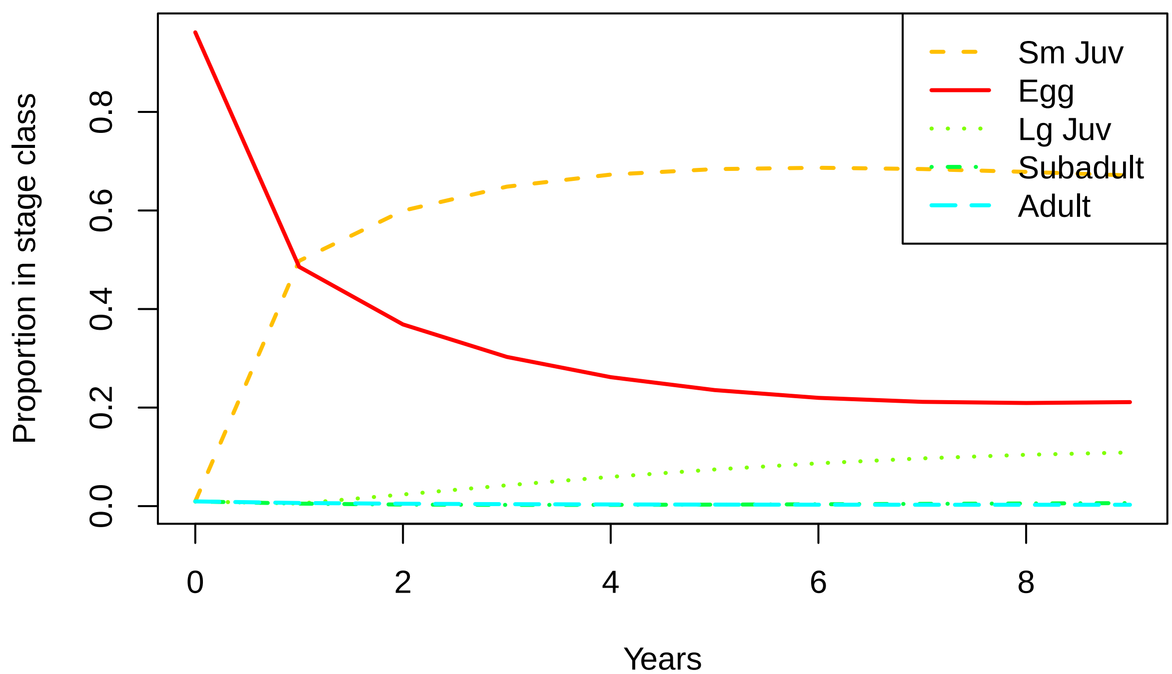
```
##            Lg Juv  Adult
## Egg         0.000 61.896
## Sm Juv      0.000  0.000
## Lg Juv      0.657  0.000
## Subadult    0.019  0.000
## Adult       0.000  0.809
```

```r
A[3:4, 3]
```

```
##   Lg Juv Subadult
##    0.657    0.019
```

2. Projecting the population matrix

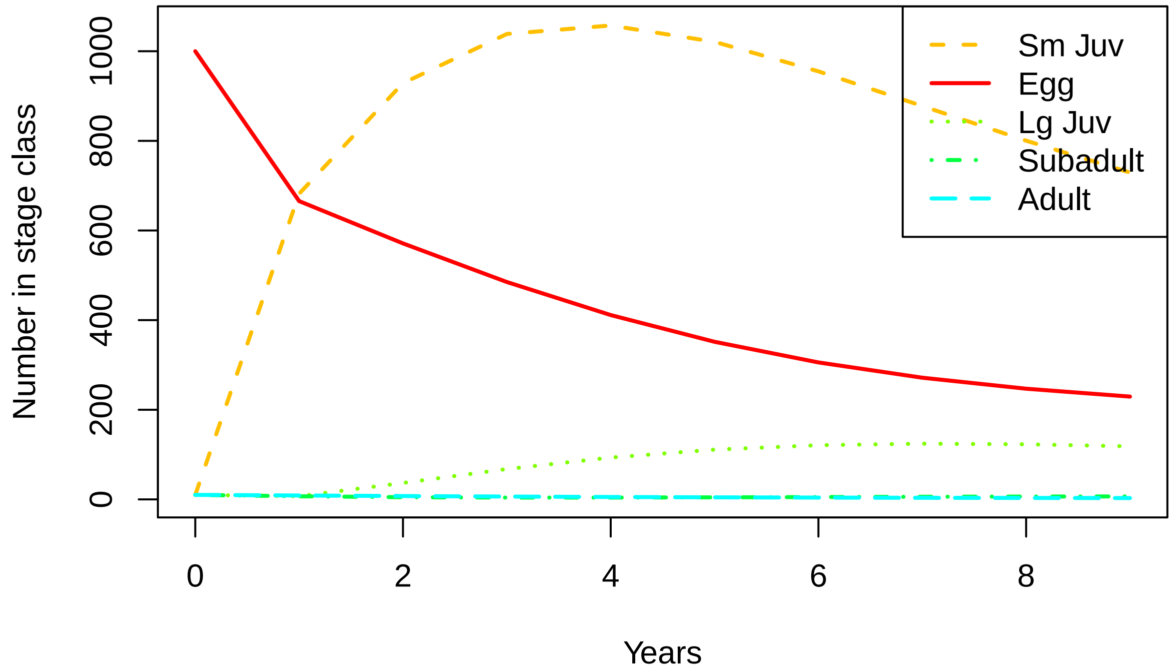
```r
library(popbio)

# Initial abundance
n_0 <-c(1000, 10, 10, 10, 10)

# Project the matrix
pop <- pop.projection(A, n_0, iterations = 10)

# Plot each stage through time
stage.vector.plot(pop$stage.vector)
```



```r
#these are proportions in each stage through time
```

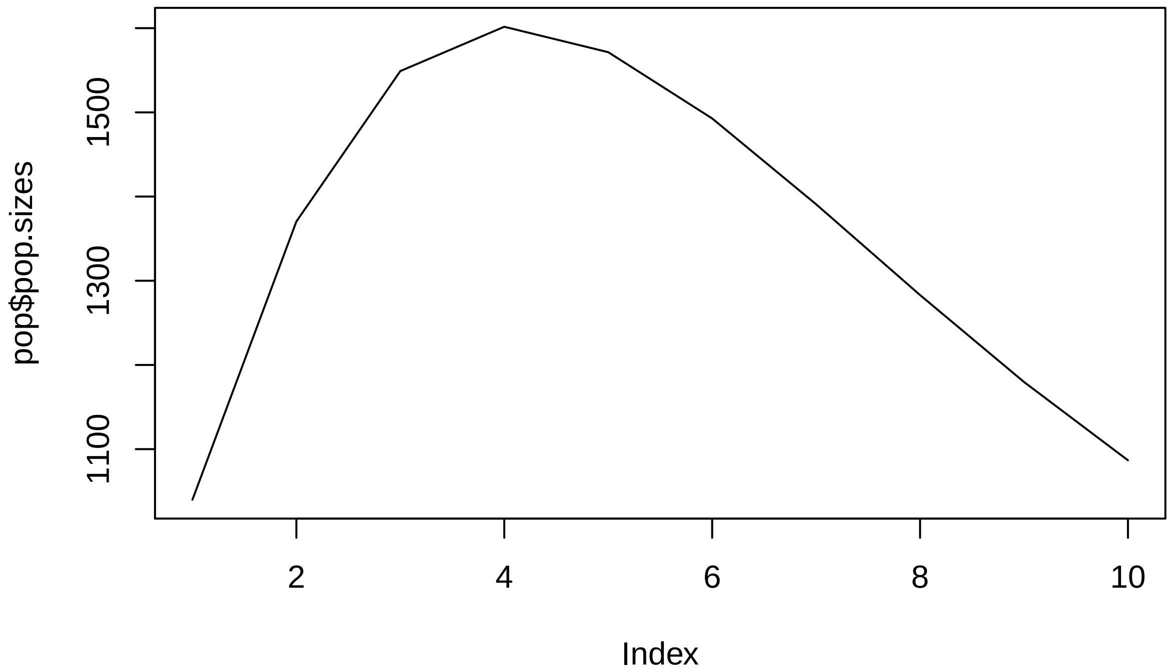
Actual abundances through time

```r
stage.vector.plot(pop$stage.vector, proportions = FALSE)
```

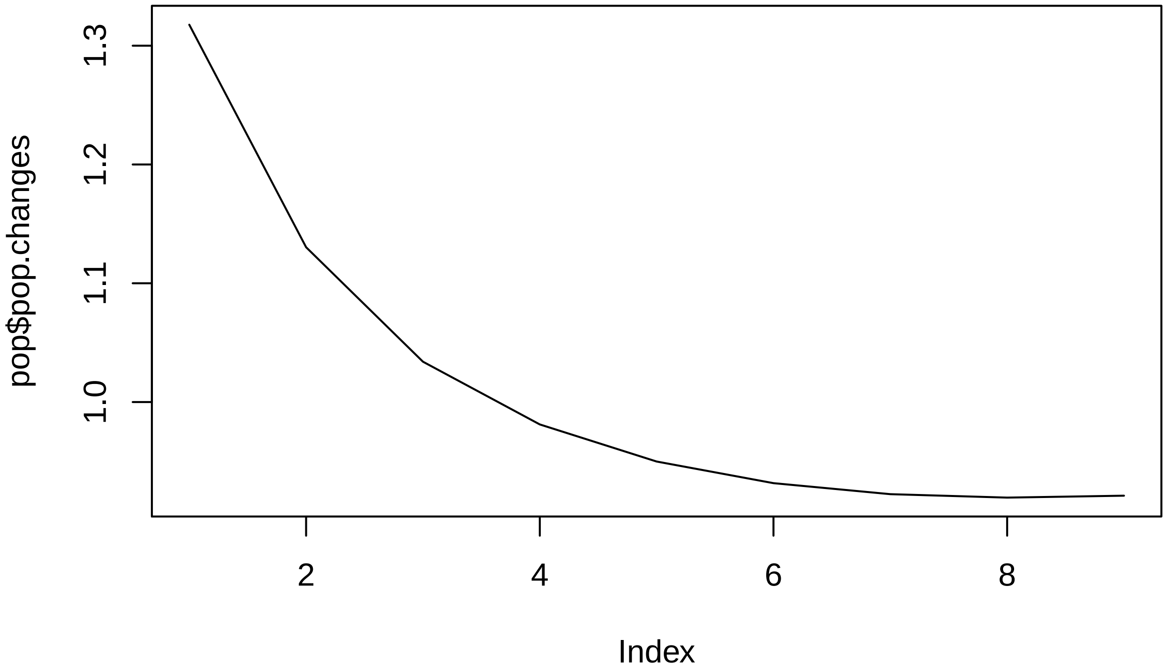


This graph shows the number of turtles in each stage class over 10 years.

```r
plot(pop$pop.sizes, type = "l")
```



```r
plot(pop$pop.changes, type = "l")
```

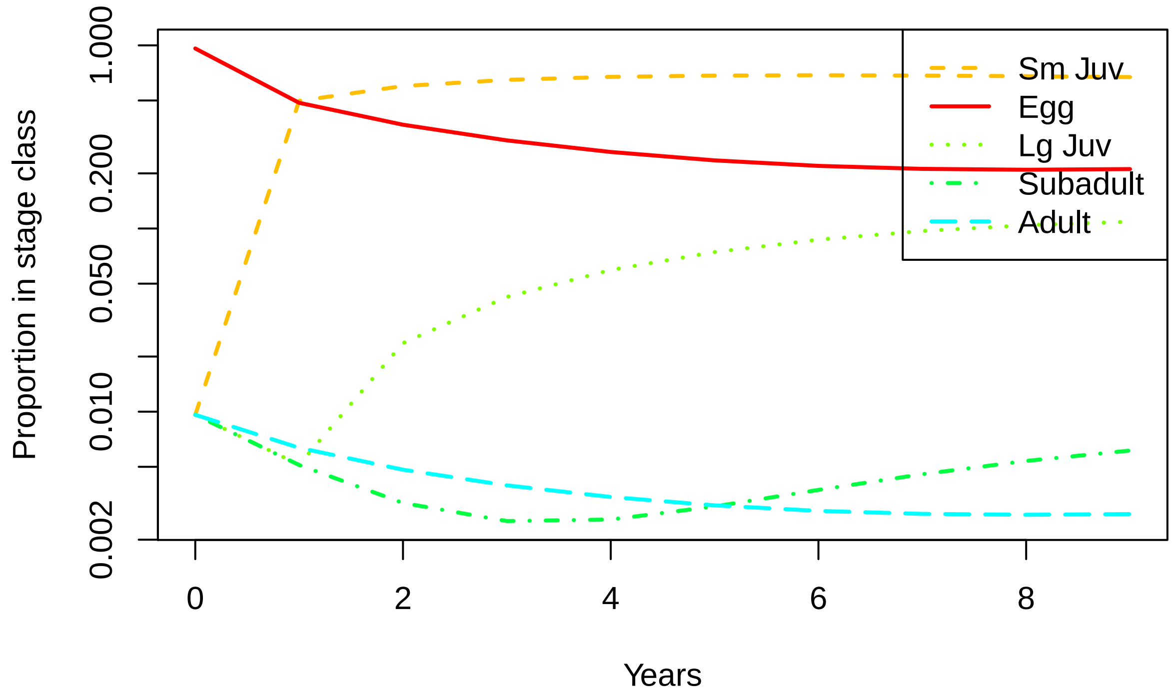


The population size is going down, which we could tell from the previous graphs that are seperated out by stage classes The changes in population is going down also whcih corresponds to the decrease in populaiton size

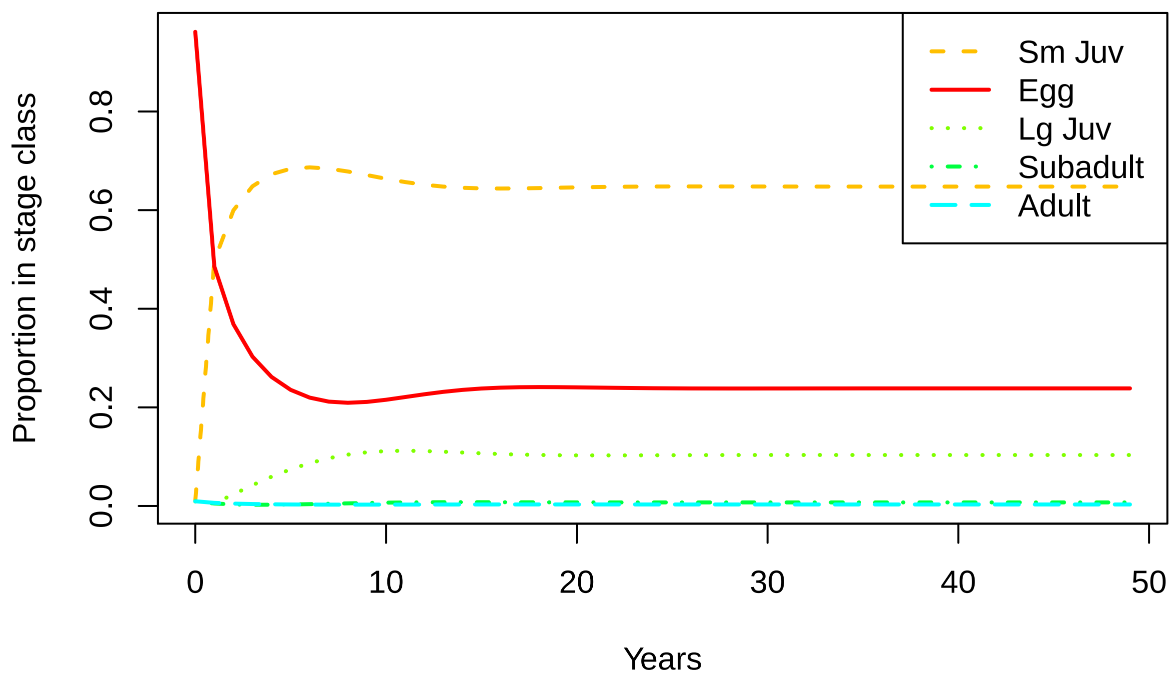
plot proportions in log to see if reaching stable

- they are not fully straight yet at 10 years so not stable
- when project out to 50 years you can tell that year 10 is when they start to be stable

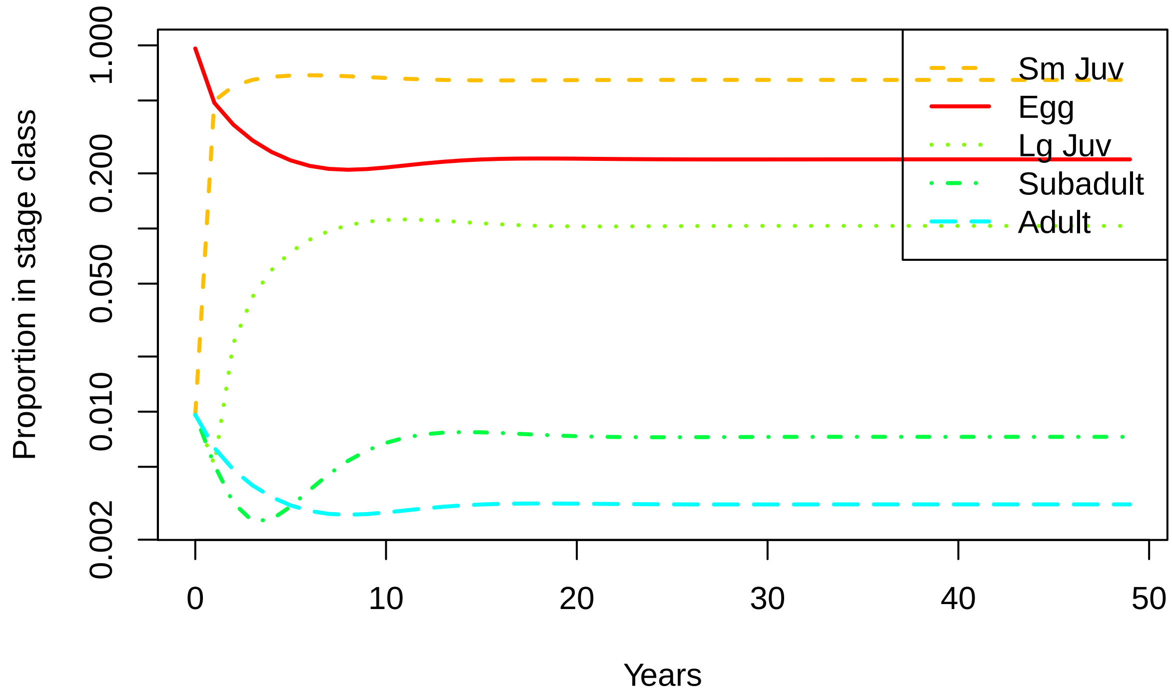
```r
stage.vector.plot(pop$stage.vector, log = "y")
```




```r
pop_50 <- pop.projection(A, n_0, iterations = 50)

# Plot each stage through time
stage.vector.plot(pop_50$stage.vector)
```




```r
stage.vector.plot(pop_50$stage.vector, log = "y")
```



- Plotting out to 50 years we can see that the populations become more stable at each time stage

-It seems that 10 years out is when the populaiton begins to stable

- The log version of out graph confirms that sometime after 10 years the population is reaching a stable state because they are straight

3. Analyzing the population matrix

SSD (stable stage distribution)

```
lambda(A)
```

```
## [1] 0.9515489
```

```
stable.stage(A)
```

```
##         Egg      Sm Juv      Lg Juv    Subadult       Adult
## 0.238508404 0.647732505 0.103356123 0.007285382 0.003117586
```

```
pop_50
```

```
## $lambda
## [1] 0.9515473
##
## $stable.stage
##         Egg      Sm Juv      Lg Juv    Subadult       Adult
## 0.238507035 0.647734069 0.103356048 0.007285280 0.003117568
##
## $stage.vectors
##             0      1          2           3           4           5
## Egg      1000 665.61  571.19685  485.036481  411.289802  351.626280
## Sm Juv     10 682.03  928.75384 1038.471823 1057.445316 1021.004674
## Lg Juv     10   7.04   36.68069   67.750644   93.320349  111.011399
```

```
## Subadult   10   7.01   4.91458    4.048677    4.048460    4.534136
## Adult      10   8.70   7.46591    6.339711    5.375795    4.595974
##                  6          7          8          9         10         11
## Egg      305.624171 271.522516 246.937023 229.398165 216.689085 207.013033
## Sm Juv   955.114025 877.741474 800.329955 729.314449 667.551819 615.554061
## Lg Juv   120.921709 124.335922 122.942550 118.388763 112.059197 104.997828
## Subadult   5.201497   5.844934   6.348627   6.665672   6.795375   6.763570
## Adult      3.994726   3.549024   3.227702   2.998477   2.832374   2.705908
##                 12         13         14         15         16         17
## Egg      199.036955 191.857183 184.924804 177.957788 170.857081 163.635883
## Sm Juv   572.468302 536.795161 506.870597 481.154272 458.372960 437.564721
## Lg Juv    97.914614  91.235911  85.171366  79.780506  75.030043  70.838267
## Subadult   6.607714   6.366838   6.075666   5.761860   5.445418   5.139346
## Adult      2.601658   2.507812   2.417197   2.326128   2.233311   2.138919
##                 18         19         20         21         22         23
## Egg      156.365578 149.138074 142.042252 135.151381 128.518302 122.175557
## Sm Juv   418.062220 399.444506 381.477688 364.057335 347.159488 330.802974
## Lg Juv    67.106284  63.737753  60.649595  57.776235  55.069681  52.497277
## Subadult   4.850961   4.583375   4.336879   4.110094   3.900832   3.706692
## Adult      2.043886   1.949412   1.856660   1.766588   1.679885   1.596978
##                 24         25         26         27         28         29
## Egg      116.138261 110.408094 104.977401  99.832788  94.957964  90.335776
## Sm Juv   315.022992 299.854489 285.323169 271.441933 258.210811 245.618826
## Lg Juv    50.038451  47.681343  45.419803  43.251000  41.173678  39.187014
## Subadult   3.525412   3.355062   3.194097   3.041351   2.895970   2.757352
## Adult      1.518063   1.443163   1.372178   1.304932   1.241212   1.180795
##                 30         31         32          33          34
## Egg       85.949527  81.783712  77.824334  74.0589335  70.4764600
## Sm Juv   233.646683 222.269549 211.459499 201.1874534 191.4245598
## Lg Juv    37.289953  35.480893  33.757616  32.1173500  30.5569093
## Subadult   2.625067   2.498805   2.378322   2.2634102   2.1538754
## Adult      1.123462   1.069009   1.017256   0.9680375   0.9212104
##                  35          36          37          38          39
## Egg       67.0670671  63.8218873  60.7328261  57.7923908  54.9935583
## Sm Juv   182.1430760 173.3168527 164.9215214 156.9344872 149.3348083
## Lg Juv    29.0728437  27.6615829  26.3195520  25.0432572  23.8293409
## Subadult   2.0495243   1.9501596   1.8555789   1.7655763   1.6799449
## Adult      0.8766456   0.8342273   0.7938496   0.7554146   0.7188306
##                  40          41          42          43          44
## Egg       52.3296821  49.7944288  47.3817401  45.0858102  42.9010741
## Sm Juv   142.1030221 135.2209599 128.6715743 122.4387913 116.5073922
## Lg Juv    22.6746129  21.5760627  20.5308583  19.5363379  18.5899972
## Subadult   1.5984799   1.5209810   1.4472542   1.3771137   1.3103819
## Adult      0.6840106   0.6508718   0.6193352   0.5893247   0.5607676
##                  45          46          47         48         49
## Egg       40.8222019  38.8440948  36.9618823 35.1709173 33.4667716
## Sm Juv   110.8629217 105.4916202 100.3803730 95.5166728 90.8885901
## Lg Juv    17.6894756  16.8325428  16.0170868 15.2411035 14.5026886
## Subadult   1.2468904   1.1864793   1.1289972  1.0743007  1.0222541
## Adult      0.5335943   0.5077381   0.4831353  0.4597253  0.4374501
##
## $pop.sizes
##  [1] 1040.0000 1370.3900 1549.0119 1601.6473 1571.4797 1492.7725 1390.8561
##  [8] 1282.9939 1179.7859 1086.7655 1005.9278  937.0344  878.6292  828.7629
```

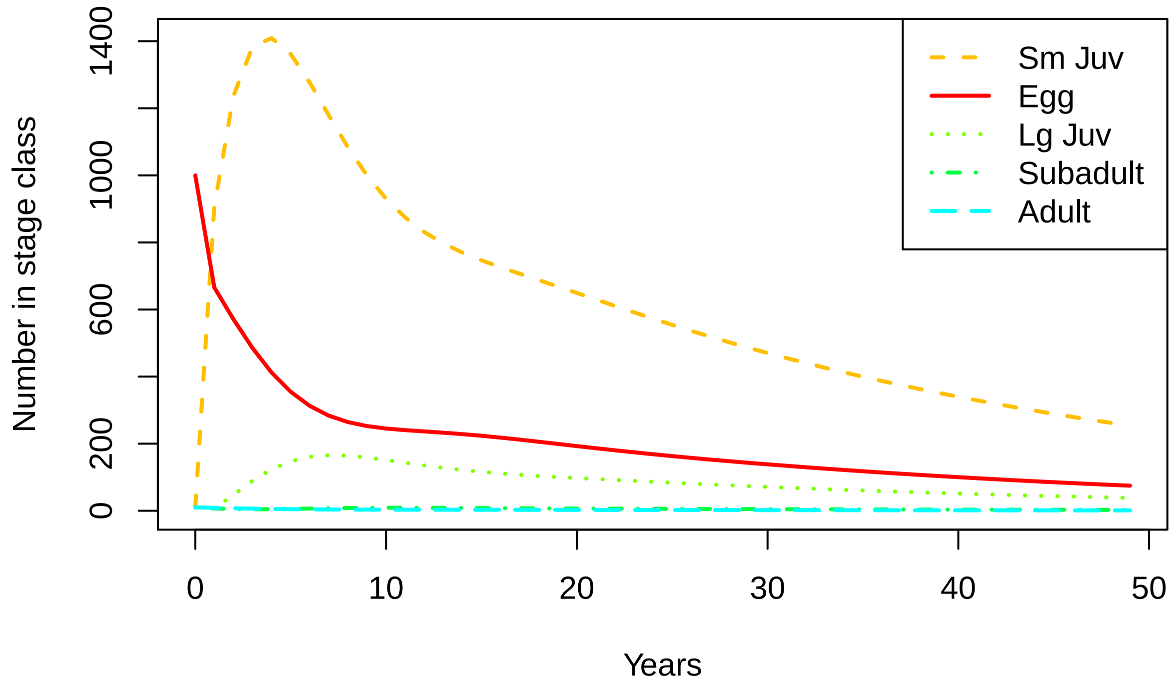
```
## [15]  785.4596  746.9806  711.9388  679.3171  648.4289  618.8531  590.3631
## [22]  562.8616  536.3282  510.7795  486.2432  462.7422  440.2866  418.8720
## [29]  398.4796  379.0798  360.6347  343.1020  326.4370  310.5952  295.5330
## [36]  281.2092  267.5847  254.6233  242.2911  230.5565  219.3898  208.7633
## [43]  198.6508  189.0274  179.8696  171.1551  162.8625  154.9715  147.4627
## [50]  140.3178
##
## $pop.changes
##  [1] 1.3176827 1.1303438 1.0339800 0.9811646 0.9499152 0.9317268 0.9224490
##  [8] 0.9195569 0.9211549 0.9256163 0.9315125 0.9376702 0.9432453 0.9477495
## [15] 0.9510108 0.9530888 0.9541791 0.9545305 0.9543885 0.9539632 0.9534161
## [22] 0.9528597 0.9523637 0.9519630 0.9516682 0.9514730 0.9513620 0.9513160
## [29] 0.9513153 0.9513425 0.9513837 0.9514286 0.9514705 0.9515055 0.9515321
## [36] 0.9515505 0.9515616 0.9515669 0.9515680 0.9515664 0.9515634 0.9515598
## [43] 0.9515563 0.9515532 0.9515509 0.9515492 0.9515481 0.9515475 0.9515473
```

Comparing lambda and the population vector are different because:

1. lambda and stable.stage project out to as far as needed to reach stable asymptotic (goes out as many years as needs to)
2. when we call to pop_ something it is only looking at until the number of iterations/years that we ran it

Informating management decisions for turtle conservation

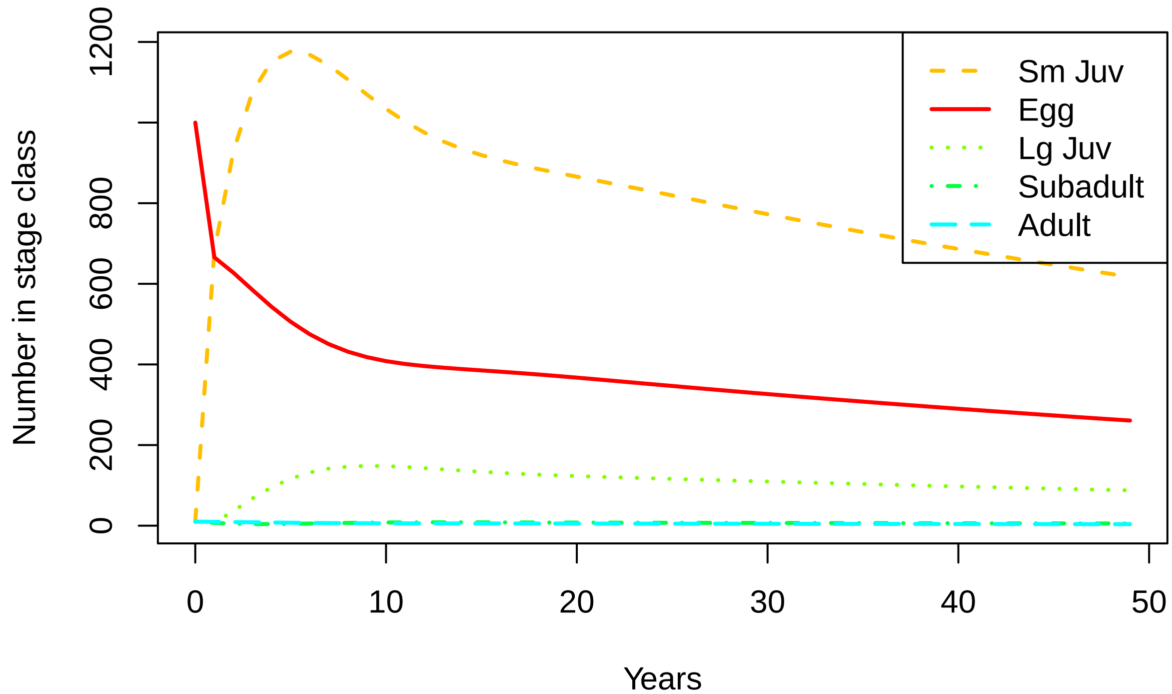
```r
#increase egg survival
class_names <- c("Egg", "Sm Juv", "Lg Juv", "Subadult", "Adult")
egg_increase <- matrix(c(0,     0,     0,     4.665, 61.896,
              0.9, 0.703, 0,     0,      0,
              0,     0.047, 0.657, 0,      0,
              0,     0,     0.019, 0.682,  0,
              0,     0,     0,     0.061,  0.809),
          nrow = 5, ncol = 5, byrow = TRUE, dimnames = list(class_names, class_names))

# Initial abundance
n_0_egg <-c(1000, 10, 10, 10, 10)

# Project the matrix
pop_egg <- pop.projection(egg_increase, n_0_egg, iterations = 50)

# Plot each stage through time
stage.vector.plot(pop_egg$stage.vector, proportions = FALSE)
```




```
lambda(egg_increase)
```

```
## [1] 0.9680462
```

```
###increasing egg survival does not seem to matter much


#increase egg survival
class_names <- c("Egg", "Sm Juv", "Lg Juv", "Subadult", "Adult")
adults_increase <- matrix(c(0,     0,     0,     4.665, 61.896,
                        0.675, 0.703, 0,     0,      0,
                         0,     0.047, 0.657, 0,      0,
                         0,     0,     0.019, 0.682,  0,
                         0,     0,     0,     0.061,  0.9),
          nrow = 5, ncol = 5, byrow = TRUE, dimnames = list(class_names, class_names))

# Initial abundance
n_0_adults <-c(1000, 10, 10, 10, 10)

# Project the matrix
pop_adults <- pop.projection(adults_increase, n_0_adults, iterations = 50)

# Plot each stage through time
stage.vector.plot(pop_adults$stage.vector, proportions = FALSE)
```

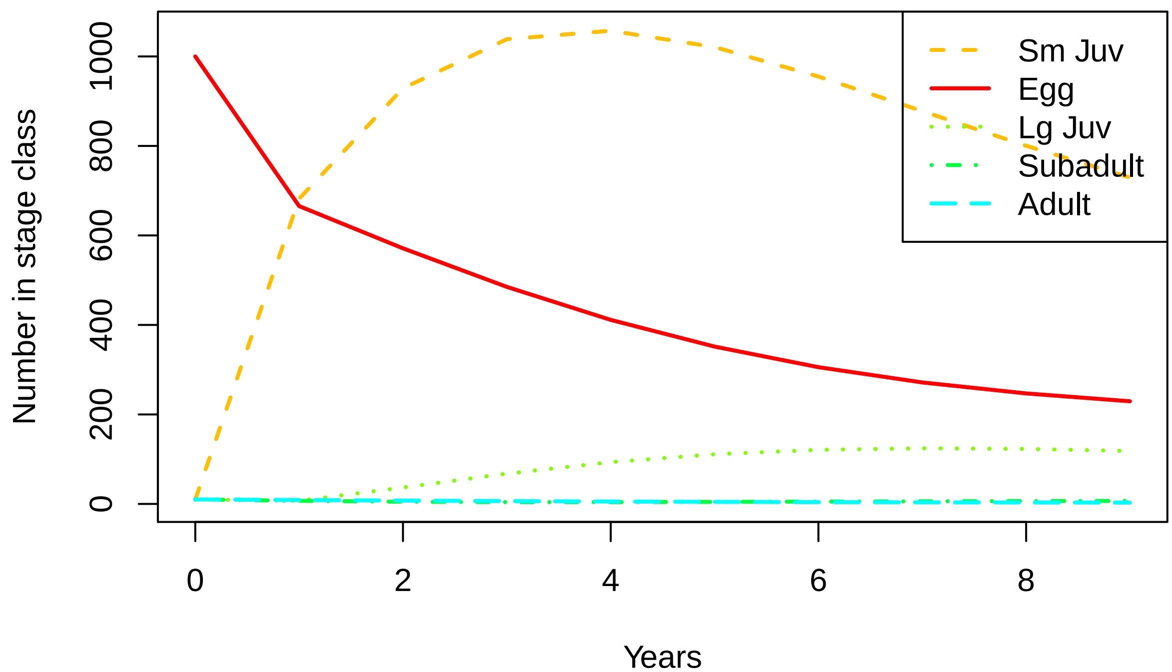



```r
lambda(adults_increase)
```

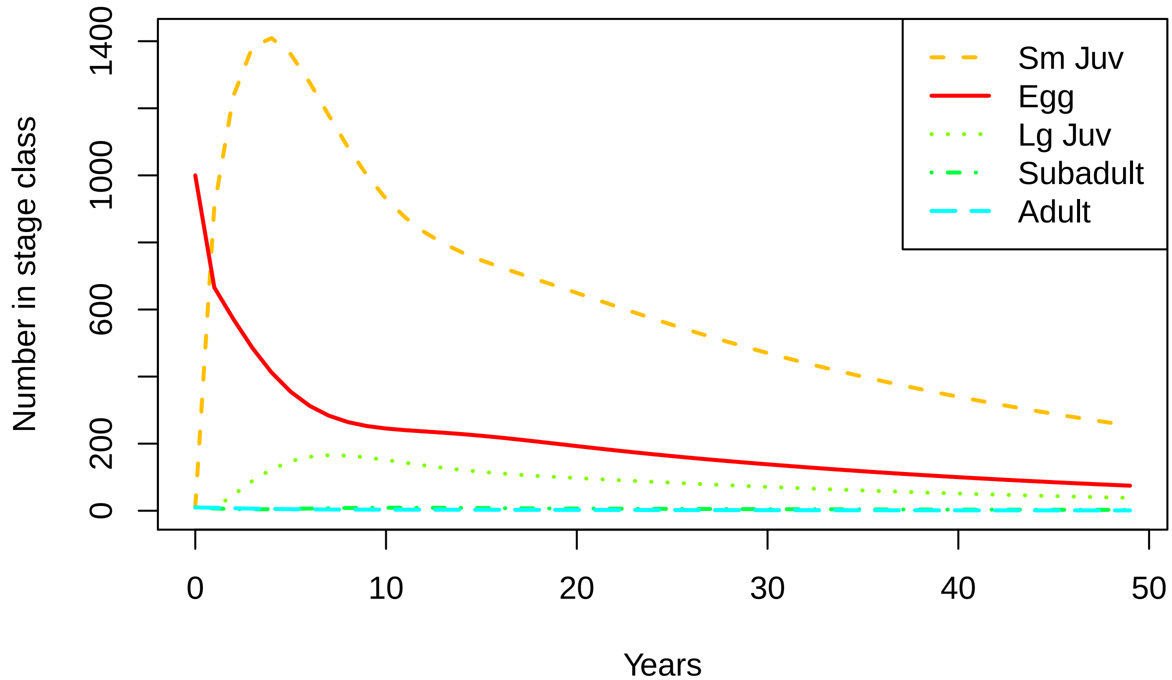
```
## [1] 0.988256
```

Increasing the survival of the eggs does not make a big difference in the numbers of total turtles. However, changing the survival rate of the adult turtles does.

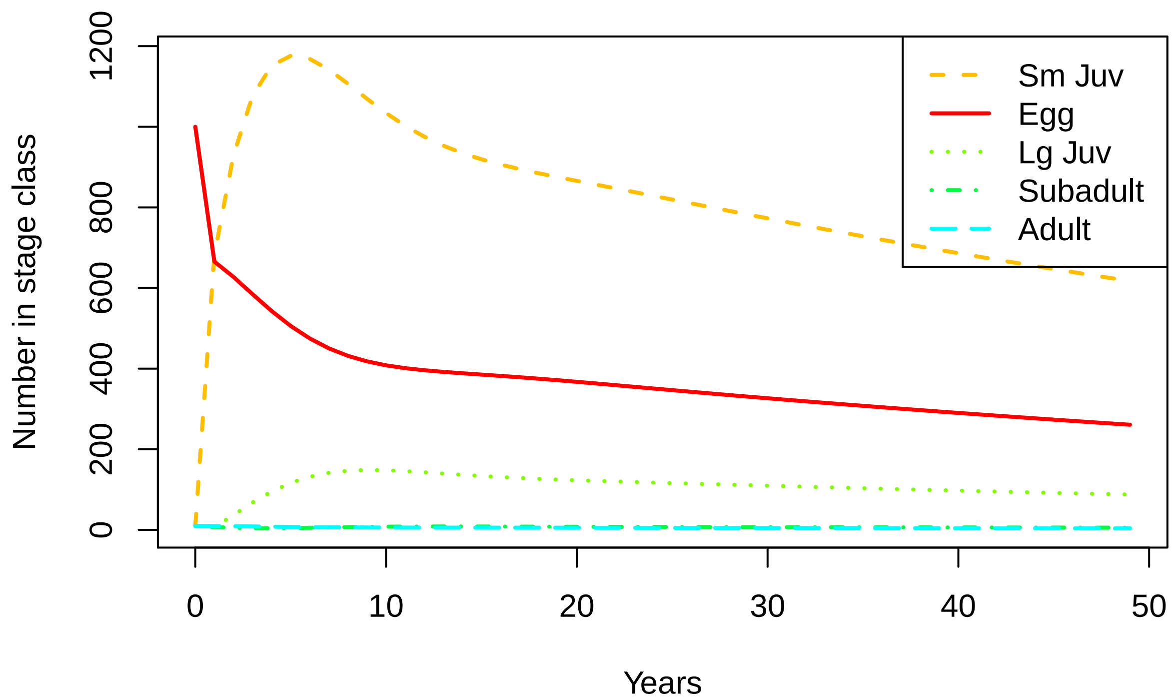
```r
#normal
stage.vector.plot(pop$stage.vector, proportions = FALSE)
```




```r
#increase egg survival
stage.vector.plot(pop_egg$stage.vector, proportions = FALSE)
```




```
#increase adult
stage.vector.plot(pop_adults$stage.vector, proportions = FALSE)
```



In order to construct a viable conservation sea turtle plan, we need to increase the survival of the adults. This can be done by decreasing deaths. Many sea turtle deaths are now caused by human activity such as net entanglement, prop strikes, and runoff pollution/algage blooms. In order to properly write a conservation plan, the writter will need to know which area the conservation action will occur and what the main causes of death are.