On two conjectures about perturbations of the stochastic growth rate – Supporting and Supplementary Material

Stefano Giaimo (giaimo@evolbio.mpg.de)
Evolutionary Theory Department
Max Planck Institute for Evolutionary Biology
August-Thienemann-Str. 2, 24306 Ploen, Germany

14th Oct 2022

Here, the code to generate Figure 1 in the manuscript is reported. Code was written in R 4.1.2 and executed on the platform: x86_64-apple-darwin17.0.

Code for Figure 1

The R environment gets cleaned:

```
rm(list=ls())
```

All global variables and functions are set. The environmental states (good year, bad year) of the model get labeled:

```
my.envs <- c(1,2) # 1 is a good year, 2 is bad year
my.envs
```

```
## [1] 1 2
```

The probabilities of the environmental states are attributed:

```
pr.good <- 0.1
pr.bad <- 1-pr.good
```

Number of time steps in the simulation of the demographic process:

```
Tlimit <- 50000
# this parameter was set to 50000 for the simulations
# in the manuscript
```

Number of stages in the matrix population model from Eq.11 in Claessen, *Am.Nat.* 2005, 165, pp. E27-E35.

```
nr.stages = 3
```

Projection matrices $\boldsymbol{B}$ and $\boldsymbol{G}$ taken from Eq.11 in Claessen, *Am.Nat.* 2005, 165, pp. E27-E35.

```
base.mats <- list(good =  matrix( c(  14.7,  0.0029,   0.042,
                                      1.49,  0.0012,   0.017,
                                      970,  0.78,     0.57) ,
                                ncol=nr.stages,nrow=nr.stages,byrow = TRUE),

                  bad = matrix( c(  0.019,       0,       0.0001,
                                    0.89,       0.0018,  0.026,
```

```
                                    388,       0.78,    0.57) ,
                            ncol=nr.stages,nrow=nr.stages,byrow = TRUE))
```

Average of $\boldsymbol{B}$ and $\boldsymbol{G}$:

```
M.mat <- (pr.good * base.mats[[1]]) + (pr.bad * base.mats[[2]])
```

Parameter γ to regulate variability in demographic parameters, this is parameter v in Claessen, *Am.Nat.* 2005, 165, pp. E27-E35.

```
v.par <- seq(0,0.05,length.out = 41)
```

Generation of all projections matrices with intermediate variability, one matrix for bad years and one for good years for each value of γ, following Eqs. 6-7 in Claessen, *Am.Nat.* 2005, 165, pp. E27-E35:

```
# matrices are stored in a list
all.mats <- lapply(v.par, function(x) {
  l <- lapply(base.mats, function(y) {M.mat + x*(y - M.mat)})
  return(l)
})
all.mats
```

```
## [[1]]
## [[1]]$good
##          [,1]    [,2]    [,3]
## [1,]   1.4871 0.00029 0.00429
## [2,]   0.9500 0.00174 0.02510
## [3,] 446.2000 0.78000 0.57000
##
## [[1]]$bad
##          [,1]    [,2]    [,3]
## [1,]   1.4871 0.00029 0.00429
## [2,]   0.9500 0.00174 0.02510
## [3,] 446.2000 0.78000 0.57000
##
##
## [[2]]
## [[2]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.503616 0.0002932625 0.004337138
## [2,]   0.950675 0.0017393250 0.025089875
## [3,] 446.854750 0.7800000000 0.570000000
##
## [[2]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.485265 0.0002896375 0.004284763
## [2,]   0.949925 0.0017400750 0.025101125
## [3,] 446.127250 0.7800000000 0.570000000
##
##
## [[3]]
## [[3]]$good
##            [,1]        [,2]        [,3]
## [1,]   1.520132 0.000296525 0.004384275
## [2,]   0.951350 0.001738650 0.025079750
## [3,] 447.509500 0.780000000 0.570000000
```

```
##
## [[3]]$bad
##           [,1]        [,2]        [,3]
## [1,]   1.48343 0.000289275 0.004279525
## [2,]   0.94985 0.001740150 0.025102250
## [3,] 446.05450 0.780000000 0.570000000
##
##
## [[4]]
## [[4]]$good
##             [,1]         [,2]        [,3]
## [1,]   1.536648 0.0002997875 0.004431413
## [2,]   0.952025 0.0017379750 0.025069625
## [3,] 448.164250 0.7800000000 0.570000000
##
## [[4]]$bad
##             [,1]         [,2]        [,3]
## [1,]   1.481595 0.0002889125 0.004274288
## [2,]   0.949775 0.0017402250 0.025103375
## [3,] 445.981750 0.7800000000 0.570000000
##
##
## [[5]]
## [[5]]$good
##             [,1]       [,2]       [,3]
## [1,]   1.553164 0.00030305 0.00447855
## [2,]   0.952700 0.00173730 0.02505950
## [3,] 448.819000 0.78000000 0.57000000
##
## [[5]]$bad
##             [,1]       [,2]       [,3]
## [1,]   1.479759 0.00028855 0.00426905
## [2,]   0.949700 0.00174030 0.02510450
## [3,] 445.909000 0.78000000 0.57000000
##
##
## [[6]]
## [[6]]$good
##             [,1]         [,2]        [,3]
## [1,]   1.569681 0.0003063125 0.004525688
## [2,]   0.953375 0.0017366250 0.025049375
## [3,] 449.473750 0.7800000000 0.570000000
##
## [[6]]$bad
##             [,1]         [,2]        [,3]
## [1,]   1.477924 0.0002881875 0.004263813
## [2,]   0.949625 0.0017403750 0.025105625
## [3,] 445.836250 0.7800000000 0.570000000
##
##
## [[7]]
## [[7]]$good
##             [,1]        [,2]        [,3]
## [1,]   1.586197 0.000309575 0.004572825
```

```
## [2,]   0.954050 0.001735950 0.025039250
## [3,] 450.128500 0.780000000 0.570000000
##
## [[7]]$bad
##           [,1]        [,2]        [,3]
## [1,]   1.476089 0.000287825 0.004258575
## [2,]   0.949550 0.001740450 0.025106750
## [3,] 445.763500 0.780000000 0.570000000
##
##
## [[8]]
## [[8]]$good
##           [,1]         [,2]        [,3]
## [1,]   1.602713 0.0003128375 0.004619963
## [2,]   0.954725 0.0017352750 0.025029125
## [3,] 450.783250 0.7800000000 0.570000000
##
## [[8]]$bad
##           [,1]         [,2]        [,3]
## [1,]   1.474254 0.0002874625 0.004253338
## [2,]   0.949475 0.0017405250 0.025107875
## [3,] 445.690750 0.7800000000 0.570000000
##
##
## [[9]]
## [[9]]$good
##           [,1]      [,2]      [,3]
## [1,]   1.619229 0.0003161 0.0046671
## [2,]   0.955400 0.0017346 0.0250190
## [3,] 451.438000 0.7800000 0.5700000
##
## [[9]]$bad
##           [,1]      [,2]      [,3]
## [1,]   1.472419 0.0002871 0.0042481
## [2,]   0.949400 0.0017406 0.0251090
## [3,] 445.618000 0.7800000 0.5700000
##
##
## [[10]]
## [[10]]$good
##           [,1]         [,2]        [,3]
## [1,]   1.635745 0.0003193625 0.004714238
## [2,]   0.956075 0.0017339250 0.025008875
## [3,] 452.092750 0.7800000000 0.570000000
##
## [[10]]$bad
##           [,1]         [,2]        [,3]
## [1,]   1.470584 0.0002867375 0.004242863
## [2,]   0.949325 0.0017406750 0.025110125
## [3,] 445.545250 0.7800000000 0.570000000
##
##
## [[11]]
## [[11]]$good
```

```
##            [,1]        [,2]        [,3]
## [1,]   1.652261 0.000322625 0.004761375
## [2,]   0.956750 0.001733250 0.024998750
## [3,] 452.747500 0.780000000 0.570000000
##
## [[11]]$bad
##            [,1]        [,2]        [,3]
## [1,]   1.468749 0.000286375 0.004237625
## [2,]   0.949250 0.001740750 0.025111250
## [3,] 445.472500 0.780000000 0.570000000
##
##
## [[12]]
## [[12]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.668777 0.0003258875 0.004808513
## [2,]   0.957425 0.0017325750 0.024988625
## [3,] 453.402250 0.7800000000 0.570000000
##
## [[12]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.466914 0.0002860125 0.004232388
## [2,]   0.949175 0.0017408250 0.025112375
## [3,] 445.399750 0.7800000000 0.570000000
##
##
## [[13]]
## [[13]]$good
##            [,1]       [,2]       [,3]
## [1,]   1.685293 0.00032915 0.00485565
## [2,]   0.958100 0.00173190 0.02497850
## [3,] 454.057000 0.78000000 0.57000000
##
## [[13]]$bad
##            [,1]       [,2]       [,3]
## [1,]   1.465078 0.00028565 0.00422715
## [2,]   0.949100 0.00174090 0.02511350
## [3,] 445.327000 0.78000000 0.57000000
##
##
## [[14]]
## [[14]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.701810 0.0003324125 0.004902788
## [2,]   0.958775 0.0017312250 0.024968375
## [3,] 454.711750 0.7800000000 0.570000000
##
## [[14]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.463243 0.0002852875 0.004221913
## [2,]   0.949025 0.0017409750 0.025114625
## [3,] 445.254250 0.7800000000 0.570000000
##
##
```

```
## [[15]]
## [[15]]$good
##            [,1]        [,2]        [,3]
## [1,]   1.718326 0.000335675 0.004949925
## [2,]   0.959450 0.001730550 0.024958250
## [3,] 455.366500 0.780000000 0.570000000
##
## [[15]]$bad
##            [,1]        [,2]        [,3]
## [1,]   1.461408 0.000284925 0.004216675
## [2,]   0.948950 0.001741050 0.025115750
## [3,] 445.181500 0.780000000 0.570000000
##
##
## [[16]]
## [[16]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.734842 0.0003389375 0.004997063
## [2,]   0.960125 0.0017298750 0.024948125
## [3,] 456.021250 0.7800000000 0.570000000
##
## [[16]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.459573 0.0002845625 0.004211438
## [2,]   0.948875 0.0017411250 0.025116875
## [3,] 445.108750 0.7800000000 0.570000000
##
##
## [[17]]
## [[17]]$good
##            [,1]      [,2]      [,3]
## [1,]   1.751358 0.0003422 0.0050442
## [2,]   0.960800 0.0017292 0.0249380
## [3,] 456.676000 0.7800000 0.5700000
##
## [[17]]$bad
##            [,1]      [,2]      [,3]
## [1,]   1.457738 0.0002842 0.0042062
## [2,]   0.948800 0.0017412 0.0251180
## [3,] 445.036000 0.7800000 0.5700000
##
##
## [[18]]
## [[18]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.767874 0.0003454625 0.005091338
## [2,]   0.961475 0.0017285250 0.024927875
## [3,] 457.330750 0.7800000000 0.570000000
##
## [[18]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.455903 0.0002838375 0.004200963
## [2,]   0.948725 0.0017412750 0.025119125
## [3,] 444.963250 0.7800000000 0.570000000
```

```
##
##
## [[19]]
## [[19]]$good
##           [,1]        [,2]        [,3]
## [1,]   1.78439 0.000348725 0.005138475
## [2,]   0.96215 0.001727850 0.024917750
## [3,] 457.98550 0.780000000 0.570000000
##
## [[19]]$bad
##             [,1]        [,2]        [,3]
## [1,]   1.454068 0.000283475 0.004195725
## [2,]   0.948650 0.001741350 0.025120250
## [3,] 444.890500 0.780000000 0.570000000
##
##
## [[20]]
## [[20]]$good
##             [,1]         [,2]        [,3]
## [1,]   1.800906 0.0003519875 0.005185613
## [2,]   0.962825 0.0017271750 0.024907625
## [3,] 458.640250 0.7800000000 0.570000000
##
## [[20]]$bad
##             [,1]         [,2]        [,3]
## [1,]   1.452233 0.0002831125 0.004190488
## [2,]   0.948575 0.0017414250 0.025121375
## [3,] 444.817750 0.7800000000 0.570000000
##
##
## [[21]]
## [[21]]$good
##             [,1]       [,2]       [,3]
## [1,]   1.817422 0.00035525 0.00523275
## [2,]   0.963500 0.00172650 0.02489750
## [3,] 459.295000 0.78000000 0.57000000
##
## [[21]]$bad
##             [,1]       [,2]       [,3]
## [1,]   1.450397 0.00028275 0.00418525
## [2,]   0.948500 0.00174150 0.02512250
## [3,] 444.745000 0.78000000 0.57000000
##
##
## [[22]]
## [[22]]$good
##             [,1]         [,2]        [,3]
## [1,]   1.833939 0.0003585125 0.005279888
## [2,]   0.964175 0.0017258250 0.024887375
## [3,] 459.949750 0.7800000000 0.570000000
##
## [[22]]$bad
##             [,1]         [,2]        [,3]
## [1,]   1.448562 0.0002823875 0.004180013
```

```
## [2,]   0.948425 0.0017415750 0.025123625
## [3,] 444.672250 0.7800000000 0.570000000
##
##
## [[23]]
## [[23]]$good
##           [,1]        [,2]        [,3]
## [1,]   1.850455 0.000361775 0.005327025
## [2,]   0.964850 0.001725150 0.024877250
## [3,] 460.604500 0.780000000 0.570000000
##
## [[23]]$bad
##           [,1]        [,2]        [,3]
## [1,]   1.446727 0.000282025 0.004174775
## [2,]   0.948350 0.001741650 0.025124750
## [3,] 444.599500 0.780000000 0.570000000
##
##
## [[24]]
## [[24]]$good
##           [,1]         [,2]        [,3]
## [1,]   1.866971 0.0003650375 0.005374163
## [2,]   0.965525 0.0017244750 0.024867125
## [3,] 461.259250 0.7800000000 0.570000000
##
## [[24]]$bad
##           [,1]         [,2]        [,3]
## [1,]   1.444892 0.0002816625 0.004169538
## [2,]   0.948275 0.0017417250 0.025125875
## [3,] 444.526750 0.7800000000 0.570000000
##
##
## [[25]]
## [[25]]$good
##           [,1]      [,2]      [,3]
## [1,]   1.883487 0.0003683 0.0054213
## [2,]   0.966200 0.0017238 0.0248570
## [3,] 461.914000 0.7800000 0.5700000
##
## [[25]]$bad
##           [,1]      [,2]      [,3]
## [1,]   1.443057 0.0002813 0.0041643
## [2,]   0.948200 0.0017418 0.0251270
## [3,] 444.454000 0.7800000 0.5700000
##
##
## [[26]]
## [[26]]$good
##           [,1]         [,2]        [,3]
## [1,]   1.900003 0.0003715625 0.005468438
## [2,]   0.966875 0.0017231250 0.024846875
## [3,] 462.568750 0.7800000000 0.570000000
##
## [[26]]$bad
```

```
##            [,1]         [,2]        [,3]
## [1,]   1.441222 0.0002809375 0.004159063
## [2,]   0.948125 0.0017418750 0.025128125
## [3,] 444.381250 0.7800000000 0.570000000
##
##
## [[27]]
## [[27]]$good
##            [,1]        [,2]        [,3]
## [1,]   1.916519 0.000374825 0.005515575
## [2,]   0.967550 0.001722450 0.024836750
## [3,] 463.223500 0.780000000 0.570000000
##
## [[27]]$bad
##            [,1]        [,2]        [,3]
## [1,]   1.439387 0.000280575 0.004153825
## [2,]   0.948050 0.001741950 0.025129250
## [3,] 444.308500 0.780000000 0.570000000
##
##
## [[28]]
## [[28]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.933035 0.0003780875 0.005562713
## [2,]   0.968225 0.0017217750 0.024826625
## [3,] 463.878250 0.7800000000 0.570000000
##
## [[28]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.437552 0.0002802125 0.004148588
## [2,]   0.947975 0.0017420250 0.025130375
## [3,] 444.235750 0.7800000000 0.570000000
##
##
## [[29]]
## [[29]]$good
##            [,1]       [,2]       [,3]
## [1,]   1.949551 0.00038135 0.00560985
## [2,]   0.968900 0.00172110 0.02481650
## [3,] 464.533000 0.78000000 0.57000000
##
## [[29]]$bad
##            [,1]       [,2]       [,3]
## [1,]   1.435716 0.00027985 0.00414335
## [2,]   0.947900 0.00174210 0.02513150
## [3,] 444.163000 0.78000000 0.57000000
##
##
## [[30]]
## [[30]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.966068 0.0003846125 0.005656988
## [2,]   0.969575 0.0017204250 0.024806375
## [3,] 465.187750 0.7800000000 0.570000000
```

```
##
## [[30]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.433881 0.0002794875 0.004138113
## [2,]   0.947825 0.0017421750 0.025132625
## [3,] 444.090250 0.7800000000 0.570000000
##
##
## [[31]]
## [[31]]$good
##            [,1]        [,2]        [,3]
## [1,]   1.982584 0.000387875 0.005704125
## [2,]   0.970250 0.001719750 0.024796250
## [3,] 465.842500 0.780000000 0.570000000
##
## [[31]]$bad
##            [,1]        [,2]        [,3]
## [1,]   1.432046 0.000279125 0.004132875
## [2,]   0.947750 0.001742250 0.025133750
## [3,] 444.017500 0.780000000 0.570000000
##
##
## [[32]]
## [[32]]$good
##            [,1]         [,2]        [,3]
## [1,]   1.999100 0.0003911375 0.005751263
## [2,]   0.970925 0.0017190750 0.024786125
## [3,] 466.497250 0.7800000000 0.570000000
##
## [[32]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.430211 0.0002787625 0.004127638
## [2,]   0.947675 0.0017423250 0.025134875
## [3,] 443.944750 0.7800000000 0.570000000
##
##
## [[33]]
## [[33]]$good
##            [,1]      [,2]      [,3]
## [1,]   2.015616 0.0003944 0.0057984
## [2,]   0.971600 0.0017184 0.0247760
## [3,] 467.152000 0.7800000 0.5700000
##
## [[33]]$bad
##            [,1]      [,2]      [,3]
## [1,]   1.428376 0.0002784 0.0041224
## [2,]   0.947600 0.0017424 0.0251360
## [3,] 443.872000 0.7800000 0.5700000
##
##
## [[34]]
## [[34]]$good
##            [,1]         [,2]        [,3]
## [1,]   2.032132 0.0003976625 0.005845538
```

```
## [2,]   0.972275 0.0017177250 0.024765875
## [3,] 467.806750 0.7800000000 0.570000000
##
## [[34]]$bad
##           [,1]         [,2]        [,3]
## [1,]   1.426541 0.0002780375 0.004117163
## [2,]   0.947525 0.0017424750 0.025137125
## [3,] 443.799250 0.7800000000 0.570000000
##
##
## [[35]]
## [[35]]$good
##           [,1]        [,2]        [,3]
## [1,]   2.048648 0.000400925 0.005892675
## [2,]   0.972950 0.001717050 0.024755750
## [3,] 468.461500 0.780000000 0.570000000
##
## [[35]]$bad
##           [,1]        [,2]        [,3]
## [1,]   1.424706 0.000277675 0.004111925
## [2,]   0.947450 0.001742550 0.025138250
## [3,] 443.726500 0.780000000 0.570000000
##
##
## [[36]]
## [[36]]$good
##           [,1]         [,2]        [,3]
## [1,]   2.065164 0.0004041875 0.005939813
## [2,]   0.973625 0.0017163750 0.024745625
## [3,] 469.116250 0.7800000000 0.570000000
##
## [[36]]$bad
##           [,1]         [,2]        [,3]
## [1,]   1.422871 0.0002773125 0.004106688
## [2,]   0.947375 0.0017426250 0.025139375
## [3,] 443.653750 0.7800000000 0.570000000
##
##
## [[37]]
## [[37]]$good
##           [,1]       [,2]       [,3]
## [1,]   2.081681 0.00040745 0.00598695
## [2,]   0.974300 0.00171570 0.02473550
## [3,] 469.771000 0.78000000 0.57000000
##
## [[37]]$bad
##           [,1]       [,2]       [,3]
## [1,]   1.421035 0.00027695 0.00410145
## [2,]   0.947300 0.00174270 0.02514050
## [3,] 443.581000 0.78000000 0.57000000
##
##
## [[38]]
## [[38]]$good
```

```
##            [,1]         [,2]        [,3]
## [1,]   2.098197 0.0004107125 0.006034088
## [2,]   0.974975 0.0017150250 0.024725375
## [3,] 470.425750 0.7800000000 0.570000000
##
## [[38]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.419200 0.0002765875 0.004096213
## [2,]   0.947225 0.0017427750 0.025141625
## [3,] 443.508250 0.7800000000 0.570000000
##
##
## [[39]]
## [[39]]$good
##            [,1]        [,2]        [,3]
## [1,]   2.114713 0.000413975 0.006081225
## [2,]   0.975650 0.001714350 0.024715250
## [3,] 471.080500 0.780000000 0.570000000
##
## [[39]]$bad
##            [,1]        [,2]        [,3]
## [1,]   1.417365 0.000276225 0.004090975
## [2,]   0.947150 0.001742850 0.025142750
## [3,] 443.435500 0.780000000 0.570000000
##
##
## [[40]]
## [[40]]$good
##            [,1]         [,2]        [,3]
## [1,]   2.131229 0.0004172375 0.006128363
## [2,]   0.976325 0.0017136750 0.024705125
## [3,] 471.735250 0.7800000000 0.570000000
##
## [[40]]$bad
##            [,1]         [,2]        [,3]
## [1,]   1.415530 0.0002758625 0.004085738
## [2,]   0.947075 0.0017429250 0.025143875
## [3,] 443.362750 0.7800000000 0.570000000
##
##
## [[41]]
## [[41]]$good
##            [,1]      [,2]      [,3]
## [1,]   2.147745 0.0004205 0.0061755
## [2,]   0.977000 0.0017130 0.0246950
## [3,] 472.390000 0.7800000 0.5700000
##
## [[41]]$bad
##            [,1]      [,2]      [,3]
## [1,]   1.413695 0.0002755 0.0040805
## [2,]   0.947000 0.0017430 0.0251450
## [3,] 443.290000 0.7800000 0.5700000
```

Function to extract the dominant eigenvalue (stable growth rate) λ of population projection matrix

```
detlambda <- function(ppm) {(
  eig <- abs(eigen(ppm, only.values = TRUE)$values))
  eig <- eig[which.max(eig)]
  return(eig)
}
```

Function to extract the stable stage distribution (dominant right eigenvector) of a population projection matrix

```
detSSD <- function(ppm) {
  reig <-   Re(eigen(ppm)$vectors)
  reig <-   reig[,which.max(abs(eigen(ppm, only.values = TRUE)$values))]
  # the eigenvector is scaled so that its components
  # add up to 1
  reig <- reig/sum(reig)
  return(reig)
}
```

Function to extract the reproductive value (dominant left eigenvector) of a population projection matrix

```
detRV <- function(ppm) {
  leig <- Re(eigen(t(ppm))$vectors)
  leig <- leig[,which.max(abs(eigen(t(ppm), only.values = TRUE)$values))]
  # the eigenvector is scaled so that its dot product with
  # the stable stage distribution is equal to 1
  leig <- leig/(sum(leig * detSSD(ppm)))
  return(leig)
}
```

Function to compute the deterministic sensitivities of $\log \lambda$ from a population projection matrix

```
detsens <- function(ppm) {
  # the deterministic growth rate for the projection matrix as its leading eigenvalue
  my.lambda <- detlambda(ppm)
  # the stable stage distribution for the projection matrix
  my.SSD <- detSSD(ppm)
  # reproductive value for the projection matrix
  my.RV <- detRV(ppm)
  # determinstic sensitivities of growth rate on the log scale
  sD <- my.RV %*% t(my.SSD)/my.lambda
  return(sD)
}
```

Function to compute the second derivatives of $\log \lambda$ for a population projection matrix using methods in Shyu and Caswell, 2014, 5, Methods in Ecology and Evolution, pp. 473-482:

```
# returns a (nr.stages)X(nr.stages) Hessian matrix with the
# second derivatives of \log\lambda with respect to the entries
# of the ppm
secder <- function(ppm) {
  # the deterministic growth rate for the projection
  # matrix as its leading eigenvalue
  my.lambda <- detlambda(ppm)
  # identity matrix
  Imat <- diag(ncol(ppm))
  # the stable stage distribution for the projection matrix
  my.SSD <- detSSD(ppm)
```

```
  # reproductive value for the projection matrix
  my.RV <- detRV(ppm)
  # vector of 1s
  evec <- rep(1,ncol(ppm))

  # Eq. 21 in Shyu and Caswell (2014)
  dw.dA <- solve(my.lambda*Imat - ppm + my.SSD%*%t(evec)%*%ppm) %*%
    kronecker(t(my.SSD),(Imat-(my.SSD%*%t(evec))))

  # Eq. 22 in Shyu and Caswell (2014)
  dv.dA <- solve(my.lambda*Imat - t(ppm) + my.lambda*my.RV%*%t(my.SSD)) %*%
    ((kronecker((Imat-(my.RV%*%t(my.SSD))),(t(my.RV)))) -
       my.lambda* kronecker(my.RV,t(my.RV))%*%dw.dA)

  # Eq. 25b in Shyu and Caswell (2014)
  Bmat <- (kronecker(Imat,my.RV)%*%dw.dA) + (kronecker(my.SSD,Imat)%*%dv.dA)

  # Eq. 43 in Shyu and Caswell (2014)
  Hloglam <- (Bmat + t(Bmat))/(2*my.lambda) -
    kronecker(my.SSD%*%t(my.SSD),my.RV%*%t(my.RV))/(my.lambda^2)

  # Hessian of log\lambda
  return(Hloglam)
}
```

Function to compute the covariances between demographic contributions of stage j to stage i and the demographic contribution of stage l to stage k for all stages given two population projection matrices

```
covterm <- function(ppm1, # ppm for good years
                    ppm2  # ppm for bad years
) {
  # from the general formula Cov(X,Y)=E(XY)-E(X)E(Y)
  term1 <- pr.good * outer(X = c(ppm1), Y = c(ppm1), FUN = "*") +
    (pr.bad * outer(X = c(ppm2), Y = c(ppm2), FUN = "*"))

  term2 <- (pr.good * ppm1) + (pr.bad * ppm2)
  term2 <- outer(X = c(term2), Y = c(term2), FUN = "*")

  # returns a matrix where the covariance Cov(a_{i,j};a_{k,l})
  # between the contribution of stage j to stage i
  # and the contribution of stage l to stage k is found
  # in row 3*(j-1)+i and column 3*(l-1)+k,
  # recall that there are 3 stages in this model
  return(term1-term2)
}
```

Function to compute the small-noise approximation to the stochastic growth rate, Eq. 8 in the manuscript

```
sna <- function(ppm1, # ppm for good years
                ppm2  # ppm for bad years
) {
  # average projection matrix
  aveppm <- (pr.good * ppm1) + (pr.bad * ppm2)
  # growth rate in the average environment as
  # dominant eigenvalue of
```

```
  # average projection matrix
  lambdaD <- detlambda(aveppm)
  # covariance matrix
  Cmat <- covterm(ppm1,ppm2)
  # Jacobian of loglambdaD
  Dloglambda <- detsens(aveppm)
  env.var <- -(1/2)*c(c(Dloglambda)%*%Cmat%*%c(Dloglambda))
  return(log(lambdaD) + env.var)
}
```

Function to compute the small-noise approx to the environment-independent sensitivities in Eq. 13 in the manuscript using Eq. C-11 Shyu and Caswell, 2014, 5, Methods in Ecology and Evolution, pp. 473-482:

```
csna.bij <- function(ppm1, # ppm for good years
                     ppm2  # ppm for bad years
) {
  # average projection matrix
  aveppm <- (pr.good * ppm1) + (pr.bad * ppm2)
  # covariance matrix
  Cmat <- covterm(ppm1,ppm2)
  # growth rate in the average environment as dominant
  # eigenvalue of average projection matrix
  my.lambda <- detlambda(aveppm)
  # stable stage distribution in the average environment
  my.SSD <- detSSD(aveppm)
  # reproductive value in the average environment
  my.RV <- detRV(aveppm)
  # Jacobian of \lambda
  my.sens <- my.lambda*detsens(aveppm)
  # Hessian of \lambda
  Hlambda <- secder(aveppm)
  Hlambda <- my.lambda*(Hlambda +
                          kronecker(my.SSD%*%t(my.SSD),
                                    my.RV%*%t(my.RV))/(my.lambda^2))
  # Eq. C-11 in Appendix of Shyu and Caswell (2014)
  Dlambdas <- ((1/my.lambda) +
                 c(c(my.sens)%*%Cmat%*%(c(my.sens)))/my.lambda^3) *
    t(c(my.sens))
  Dlambdas <- Dlambdas - t(c(my.sens))%*%Cmat%*%Hlambda/my.lambda^2
  return(matrix(c(Dlambdas),ncol=nr.stages,nrow = nr.stages))
}
```

Function to compute Caswell's conjecture (Eq. 6 of the manuscript) for the specific case of $b^s_{i,j}(k)$ from the small-noise approximation to $b^s_{i,j}$ for both environmental states (BY, GY)

```
bijk <- function(ppm1, # ppm for good years
                 ppm2  # ppm for bad years
) {
  # average projection matrix
  aveppm <- (pr.good * ppm1) + (pr.bad * ppm2)
  # growth rate in the average environment as
  # dominant eigenvalue of average projection matrix
  my.lambda <- detlambda(aveppm)
  # environment-independent sensitivities of
  # the stochastic growth rate \ln\lambda_s
```

```
  bij <- csna.bij(ppm1,ppm2)
  # sensitivities of \ln\lambda_s in good years
  bijgood <- pr.good * bij
  # sensitivities of \ln\lambda_s in bad years
  bijbad <- pr.bad * bij
  return(list(good=bijgood,bad=bijbad))
}
```

Function to compute the small-noise approximation to $b^s_{i,j}(k)$, Eqs. 14-15 in the manuscript, for both environmental states (BY, GY)

```
sna.bijk <- function(ppm1, # ppm for good years
                     ppm2  # ppm for bad years
) {
  # average projection matrix
  aveppm <- (pr.good * ppm1) + (pr.bad * ppm2)
  # growth rate in the average environment as
  # dominant eigenvalue of average projection matrix
  my.lambda <- detlambda(aveppm)
  # environment-independent sensitivities of \ln\lambda_s
  bij <- csna.bij(ppm1,ppm2)
  # Jacobian of \ln\lambda
  my.sens <- detsens(aveppm)
  # sum in brackets of Eq. 14 in manuscript for good years
  sum.term.good <- sum((ppm1 - aveppm) * my.sens)
  # sum in brackets of Eq. 14 in manuscript for bad years
  sum.term.bad <- sum((ppm2 - aveppm) * my.sens)
  # sensitivities of \ln\lambda_s in good years
  bijgood <- pr.good * (bij - c(my.sens) * sum.term.good)
  # sensitivities of \ln\lambda_s in bad years
  bijbad <- pr.bad * (bij - c(my.sens) * sum.term.bad)

  return(list(good=bijgood,bad=bijbad))
}
```

Here are the simulations to estimate stochastic sensitivities following Caswell, 2001, Matrix Population Models, Sec. 14.4.1.1. A random sequence of environmental states is generated

```
seq.env <- sample(my.envs, size = Tlimit, prob = c(pr.good,pr.bad), replace = TRUE)
```

A function is defined to project the stage distribution (sd) over one time step using a given population projection matrix (ppm) and retrieve population growth over this time step:

```
update.sd <- function(sd,ppm) {
  new.sd <- ppm%*%sd
  new.Rt <- sum(new.sd)
  new.sd <- new.sd /new.Rt
  return(list(new.sd=new.sd, new.Rt=new.Rt))
}
```

A function to project backward in time the reproductive value (rv) over one time step using a given population projection matrix (ppm):

```
update.rv <- function(rv,ppm) {
  new.rv <- t(t(rv)%*%ppm)
  new.rv <- new.rv /sum(new.rv)
  return(new.rv)
```

```
}
```

Retrieval of the stochastich growth rate and its sensitivity matrix for each value of γ:

```
vdata <- lapply(1:length(v.par), function(x) {

  # extract the projection matrices for good
  # and bad years for the given value of v
  my.mats <- list(all.mats[[x]]$good, all.mats[[x]]$bad)

  # compose the average projection matrix
  aveM <- (pr.good * my.mats[[1]]) +
    (pr.bad * my.mats[[2]])

  # set the initial stage distribution, i.e. the stage distribution
  # at time 0 as equal to the stable stage distribution
  # for the average projection matrix
  sd0 <- detSSD(aveM)

  # set the final reproductive value vector, i.e.
  # the stage-specific reproductive value at time Tlimit,
  # as proportional to the reproductive value vector of
  # the average projection matrix
  rvT <- detRV(aveM)
  # then the vector is normalized so that its components
  # add up to 1 and put in column form
  rvT <- (t(t(rvT / sum(rvT))))

  # generate a sequence of population projection matrices
  # corresponding to the sequence of environments
  seq.mats <- sapply(seq.env, function(x) my.mats[[x]],
                     simplify = FALSE)

  # initialize the sequence of stage distributions as
  # a list where the element j is the stage distribution at time j-1
  seq.sd <- list(sd0)

  # initialize the sequence of time-specific growth rates
  seq.Rt <- c()

  # generate the sequences of stage distributions and
  # time-specific growth rates
  # let j work as a time index j=0,1,...,(Tlimit-1)
  j <- 0
  while (j <= (Tlimit-1)) {
    # the stage distribution in the next time step and the realized growth
    # from the current to the next time step are computed from the current
    # stage distribution seq.sd[j+1] and population projection matrix
    my.sdRt <- update.sd( sd = seq.sd[[j+1]], ppm = seq.mats[[j+1]])
    # the sequences of stage distributions and growth rates are
    # updated accordingly
    seq.sd <- append(seq.sd, list(my.sdRt[["new.sd"]]))
    seq.Rt <- c(seq.Rt, my.sdRt[["new.Rt"]])
    # the time index is updated
```

```
  j <- j+1
}

# generate the backward sequence of reproductive values
seq.rv <- Reduce(f = update.rv, x = rev(seq.mats),
                 init  = rvT, accumulate = TRUE)
# the sequence is then put in forward time as a list
seq.rv <- rev(seq.rv)

# the reproductive values relative to the total reproductive value
# in the population
seq.rrv <- mapply(function(x,y) x/c(t(x)%*%y), seq.rv, seq.sd, SIMPLIFY = FALSE)

# estimate of the stochastic growth rate from Eq. 14.61 in Caswell 2001
loglambda_s <- sum(log(seq.Rt))/Tlimit

# stochastic sensitivities using Eq. 14.103 in Caswell 2001
seS <- lapply(1:Tlimit, function(x) {
  term <- ((seq.rrv[[x+1]] %*% t(seq.sd[[x]]))) / (seq.Rt[x])
  return(term)
} )
seS <- (Reduce(f = "+", seS))/Tlimit

# set a sequence of 0 and 1 (indicator function) so that 0
# corresponds to a good year and
# 1 to a bad year in the environmental sequence (seq.env)
ind.fun <- (seq.env-1)

# compute the environment-specific stochastic sensitivities
# for bad years from Eq. 11 in Caswell 2005, 47,
# Aust. N. Z. J. Stat. pp.75-85
badseS <- lapply(1:Tlimit, function(x) {
  term <- ind.fun[x] * ((seq.rrv[[x+1]] %*% t(seq.sd[[x]]))) / (seq.Rt[x])
  return(term)
} )
badseS <- (Reduce(f = "+", badseS))/Tlimit

# set a sequence of 0 and 1 (indicator function) so
# that 0 corresponds to a bad year and
# 1 to a good year in the environmental sequence (seq.env)
ind.fun <- as.numeric(!(ind.fun))

# compute the environment-specific stochastic sensitivities
# for good years
goodseS <- lapply(1:Tlimit, function(x) {
  term <- ind.fun[x] * ((seq.rrv[[x+1]] %*% t(seq.sd[[x]]))) /
    (seq.Rt[x])
  return(term)
} )
goodseS <- (Reduce(f = "+", goodseS))/Tlimit

# returns the estimated stochastic elasticities and growth rate
return(list(stocgrowth = loglambda_s,
```

```
              overall=seS,
              good=goodseS,
              bad=badseS))
})
vdata
```

```
## [[1]]
## [[1]]$stocgrowth
## [1] 0.9121887
##
## [[1]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2630931599 0.7203018012 61.44187120
## [2,] 0.0002158645 0.0005909982  0.05041225
## [3,] 0.0005907428 0.0016173477  0.13796004
##
## [[1]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.616725e-02 7.164122e-02 6.111008509
## [2,] 2.146988e-05 5.878068e-05 0.005014002
## [3,] 5.875528e-05 1.608614e-04 0.013721506
##
## [[1]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2369259142 0.6486605841 55.33086269
## [2,] 0.0001943946 0.0005322175  0.04539825
## [3,] 0.0005319875 0.0014564863  0.12423854
##
##
## [[2]]
## [[2]]$stocgrowth
## [1] 0.9121809
##
## [[2]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2630950624 0.7203227340 61.4430666
## [2,] 0.0002158677 0.0005910197  0.0504136
## [3,] 0.0005907462 0.0016173923  0.1379625
##
## [[2]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.596905e-02 7.110006e-02 6.064754463
## [2,] 2.130735e-05 5.833689e-05 0.004976071
## [3,] 5.831023e-05 1.596462e-04 0.013617640
##
## [[2]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2371260098 0.6492226734 55.37831210
## [2,] 0.0001945603 0.0005326829  0.04543753
## [3,] 0.0005324360 0.0014577461  0.12434489
##
##
## [[3]]
## [[3]]$stocgrowth
```

```
## [1] 0.9121668
##
## [[3]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2630991512 0.7203677729 61.44563763
## [2,] 0.0002158745 0.0005910662  0.05041652
## [3,] 0.0005907536 0.0016174884  0.13796787
##
## [[3]]$good
##              [,1]         [,2]       [,3]
## [1,] 2.577373e-02 7.056851e-02 6.01925464
## [2,] 2.114735e-05 5.790148e-05 0.00493880
## [3,] 5.787152e-05 1.584523e-04 0.01351545
##
## [[3]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2373254247 0.6497992660 55.42638299
## [2,] 0.0001947271 0.0005331647  0.04547772
## [3,] 0.0005328820 0.0014590361  0.12445242
##
##
## [[4]]
## [[4]]$stocgrowth
## [1] 0.9121464
##
## [[4]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2631053827 0.7204364436 61.44955723
## [2,] 0.0002158848 0.0005911369  0.05042096
## [3,] 0.0005907647 0.0016176348  0.13797601
##
## [[4]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.558120e-02 7.004631e-02 5.974490795
## [2,] 2.098982e-05 5.747422e-05 0.004902173
## [3,] 5.743901e-05 1.572792e-04 0.013414882
##
## [[4]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2375241777 0.6503901334 55.47506644
## [2,] 0.0001948950 0.0005336627  0.04551879
## [3,] 0.0005333257 0.0014603557  0.12456113
##
##
## [[5]]
## [[5]]$stocgrowth
## [1] 0.9121198
##
## [[5]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2631137147 0.7205282877 61.4547991
## [2,] 0.0002158987 0.0005912316  0.0504269
## [3,] 0.0005907797 0.0016178307  0.1379869
##
```

```
## [[5]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.539143e-02 0.0695332329 5.930445297
## [2,] 2.083471e-05 0.0000570549 0.004866175
## [3,] 5.701258e-05 0.0001561263 0.013315911
##
## [[5]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2377222872 0.6509950548 55.52435380
## [2,] 0.0001950640 0.0005341767  0.04556073
## [3,] 0.0005337671 0.0014617045  0.12467099
##
##
## [[6]]
## [[6]]$stocgrowth
## [1] 0.9120871
##
## [[6]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2631241064 0.7206428622 61.46133776
## [2,] 0.0002159160 0.0005913497  0.05043432
## [3,] 0.0005907983 0.0016180751  0.13800049
##
## [[6]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.520434e-02 6.902905e-02 5.887101061
## [2,] 2.068196e-05 5.664331e-05 0.004830788
## [3,] 5.659209e-05 1.549931e-04 0.013218494
##
## [[6]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2379197711 0.6516138169 55.57423670
## [2,] 0.0001952340 0.0005347064  0.04560353
## [3,] 0.0005342062 0.0014630820  0.12478199
##
##
## [[7]]
## [[7]]$stocgrowth
## [1] 0.9120484
##
## [[7]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2631365179 0.7207797389 61.46914859
## [2,] 0.0002159367 0.0005914907  0.05044317
## [3,] 0.0005908205 0.0016183670  0.13801671
##
## [[7]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.501987e-02 6.853353e-02 5.844441547
## [2,] 2.053152e-05 5.623924e-05 0.004795999
## [3,] 5.617742e-05 1.538791e-04 0.013122596
##
## [[7]]$bad
##              [,1]         [,2]        [,3]
```

```
## [1,] 0.2381166468 0.6522462135 55.62470704
## [2,] 0.0001954052 0.0005352515  0.04564717
## [3,] 0.0005346431 0.0014644879  0.12489412
##
##
## [[8]]
## [[8]]$stocgrowth
## [1] 0.9120037
##
## [[8]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2631509104 0.7209385037 61.47820772
## [2,] 0.0002159607 0.0005916544  0.05045344
## [3,] 0.0005908463 0.0016187056  0.13803553
##
## [[8]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.483798e-02 6.804646e-02 5.802450735
## [2,] 2.038334e-05 5.584251e-05 0.004761792
## [3,] 5.576845e-05 1.527840e-04 0.013028181
##
## [[8]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2383129309 0.6528920451 55.67575698
## [2,] 0.0001955773 0.0005358119  0.04569165
## [3,] 0.0005350779 0.0014659216  0.12500735
##
##
## [[9]]
## [[9]]$stocgrowth
## [1] 0.9119532
##
## [[9]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2631672464 0.7211187561 61.4884920
## [2,] 0.0002159879 0.0005918402  0.0504651
## [3,] 0.0005908756 0.0016190900  0.1380569
##
## [[9]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.465861e-02 6.756764e-02 5.761113105
## [2,] 2.023735e-05 5.545292e-05 0.004728154
## [3,] 5.536506e-05 1.517071e-04 0.012935215
##
## [[9]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2385086400 0.6535511189 55.72737892
## [2,] 0.0001957505 0.0005363873  0.04573695
## [3,] 0.0005355105 0.0014673829  0.12512168
##
##
## [[10]]
## [[10]]$stocgrowth
## [1] 0.9118969
```

```
##
## [[10]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2631854893 0.7213201090 61.49997912
## [2,] 0.0002160183 0.0005920478  0.05047813
## [3,] 0.0005909083 0.0016195194  0.13808076
##
## [[10]]$good
##              [,1]         [,2]       [,3]
## [1,] 2.448170e-02 6.709686e-02 5.72041362
## [2,] 2.009353e-05 5.507031e-05 0.00469507
## [3,] 5.496714e-05 1.506481e-04 0.01284367
##
## [[10]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2387037899 0.6542232484 55.77956550
## [2,] 0.0001959247 0.0005369774  0.04578306
## [3,] 0.0005359411 0.0014688713  0.12523709
##
##
## [[11]]
## [[11]]$stocgrowth
## [1] 0.9118348
##
## [[11]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2632056034 0.7215421876 61.51264728
## [2,] 0.0002160518 0.0005922767  0.05049249
## [3,] 0.0005909443 0.0016199930  0.13810707
##
## [[11]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.430721e-02 6.663393e-02 5.680337687
## [2,] 1.995181e-05 5.469448e-05 0.004662527
## [3,] 5.457457e-05 1.496065e-04 0.012753500
##
## [[11]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2388983960 0.6549082534 55.83230959
## [2,] 0.0001961000 0.0005375822  0.04582997
## [3,] 0.0005363697 0.0014703865  0.12535357
##
##
## [[12]]
## [[12]]$stocgrowth
## [1] 0.9117671
##
## [[12]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2632275543 0.7217846292 61.52647547
## [2,] 0.0002160884 0.0005925267  0.05050818
## [3,] 0.0005909836 0.0016205100  0.13813580
##
## [[12]]$good
```

```
##              [,1]         [,2]        [,3]
## [1,] 2.413508e-02 6.617867e-02 5.640871184
## [2,] 1.981216e-05 5.432528e-05 0.004630513
## [3,] 5.418724e-05 1.485820e-04 0.012664688
##
## [[12]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2390924734 0.6556059593 55.88560429
## [2,] 0.0001962762 0.0005382014  0.04587766
## [3,] 0.0005367964 0.0014719280  0.12547111
##
##
## [[13]]
## [[13]]$stocgrowth
## [1] 0.9116938
##
## [[13]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2632513082 0.7220470830 61.54144329
## [2,] 0.0002161280 0.0005927973  0.05052515
## [3,] 0.0005910262 0.0016210697  0.13816689
##
## [[13]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.396527e-02 6.573089e-02 5.602000396
## [2,] 1.967452e-05 5.396254e-05 0.004599015
## [3,] 5.380506e-05 1.475741e-04 0.012577198
##
## [[13]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2392860367 0.6563161973 55.93944289
## [2,] 0.0001964534 0.0005388347  0.04592614
## [3,] 0.0005372211 0.0014734956  0.12558969
##
##
## [[14]]
## [[14]]$stocgrowth
## [1] 0.9116149
##
## [[14]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2632768325 0.7223292091 61.5575310
## [2,] 0.0002161705 0.0005930882  0.0505434
## [3,] 0.0005910719 0.0016216713  0.1382003
##
## [[14]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.379773e-02 0.0652904049 5.563712026
## [2,] 1.953885e-05 0.0000536061 0.004568021
## [3,] 5.342792e-05 0.0001465824 0.012491003
##
## [[14]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2394791001 0.6570388042 55.99381893
```

```
## [2,] 0.0001966317 0.0005394821  0.04597538
## [3,] 0.0005376440 0.0014750889  0.12570930
##
##
## [[15]]
## [[15]]$stocgrowth
## [1] 0.9115306
##
## [[15]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2633040951 0.7226306788 61.5747193
## [2,] 0.0002162160 0.0005933991  0.0505629
## [3,] 0.0005911207 0.0016223141  0.1382360
##
## [[15]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.363242e-02 0.0648570568 5.525993171
## [2,] 1.940512e-05 0.0000532558 0.004537519
## [3,] 5.305571e-05 0.0001456066 0.012406072
##
## [[15]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2396716774 0.6577736219 56.04872611
## [2,] 0.0001968108 0.0005401433  0.04602538
## [3,] 0.0005380650 0.0014767076  0.12582993
##
##
## [[16]]
## [[16]]$stocgrowth
## [1] 0.911441
##
## [[16]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2633330652 0.7229511734 61.59298966
## [2,] 0.0002162643 0.0005937297  0.05058363
## [3,] 0.0005911726 0.0016229975  0.13827395
##
## [[16]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.346928e-02 6.443068e-02 5.488831307
## [2,] 1.927328e-05 5.291151e-05 0.004507498
## [3,] 5.268834e-05 1.446462e-04 0.012322380
##
## [[16]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2398637820 0.6585204975 56.10415835
## [2,] 0.0001969910 0.0005408182  0.04607613
## [3,] 0.0005384843 0.0014783513  0.12595157
##
##
## [[17]]
## [[17]]$stocgrowth
## [1] 0.9113459
##
```

```
## [[17]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2633637124 0.7232903844 61.61232402
## [2,] 0.0002163154 0.0005940796  0.05060557
## [3,] 0.0005912275 0.0016237208  0.13831411
##
## [[17]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.330829e-02 6.401110e-02 5.452214279
## [2,] 1.914329e-05 5.257308e-05 0.004477947
## [3,] 5.232572e-05 1.437010e-04 0.012239898
##
## [[17]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2400554271 0.6592792831 56.16010974
## [2,] 0.0001971721 0.0005415065  0.04612762
## [3,] 0.0005389018 0.0014800198  0.12607421
##
##
## [[18]]
## [[18]]$stocgrowth
## [1] 0.9112457
##
## [[18]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2633960074 0.7236480131 61.6327048
## [2,] 0.0002163693 0.0005944485  0.0506287
## [3,] 0.0005912853 0.0016244833  0.1383564
##
## [[18]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.314938e-02 6.359818e-02 5.416130284
## [2,] 1.901511e-05 5.224036e-05 0.004448856
## [3,] 5.196775e-05 1.427706e-04 0.012158602
##
## [[18]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2402466254 0.6600498353 56.21657456
## [2,] 0.0001973542 0.0005422082  0.04617984
## [3,] 0.0005393176 0.0014817127  0.12619784
##
##
## [[19]]
## [[19]]$stocgrowth
## [1] 0.9111402
##
## [[19]]$overall
##              [,1]         [,2]       [,3]
## [1,] 0.2634299216 0.7240237699 61.6541151
## [2,] 0.0002164258 0.0005948362  0.0506530
## [3,] 0.0005913460 0.0016252845  0.1384009
##
## [[19]]$good
##              [,1]         [,2]        [,3]
```

```
## [1,] 2.299253e-02 6.319175e-02 5.380567860
## [2,] 1.888870e-05 5.191323e-05 0.004420213
## [3,] 5.161435e-05 1.418547e-04 0.012078465
##
## [[19]]$bad
##              [,1]        [,2]        [,3]
## [1,] 0.2404373893 0.660832016 56.27354727
## [2,] 0.0001975371 0.000542923  0.04623278
## [3,] 0.0005397317 0.001483430  0.12632244
##
##
## [[20]]
## [[20]]$stocgrowth
## [1] 0.9110295
##
## [[20]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2634654270 0.7244173742 61.67653836
## [2,] 0.0002164851 0.0005952424  0.05067845
## [3,] 0.0005914096 0.0016261236  0.13844747
##
## [[20]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.283770e-02 6.279168e-02 5.345515873
## [2,] 1.876403e-05 5.159155e-05 0.004392009
## [3,] 5.126542e-05 1.409529e-04 0.011999464
##
## [[20]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2406277309 0.6616256896 56.33102249
## [2,] 0.0001977211 0.0005436509  0.04628644
## [3,] 0.0005401442 0.0014851708  0.12644800
##
##
## [[21]]
## [[21]]$stocgrowth
## [1] 0.9109138
##
## [[21]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.263502496 0.7248285541 61.69995851
## [2,] 0.000216547 0.0005956668  0.05070504
## [3,] 0.000591476 0.0016270002  0.13849610
##
## [[21]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.268483e-02 6.239783e-02 5.310963505
## [2,] 1.864106e-05 5.127521e-05 0.004364235
## [3,] 5.092088e-05 1.400649e-04 0.011921574
##
## [[21]]$bad
##              [,1]         [,2]       [,3]
## [1,] 0.2408176619 0.6624307275 56.3889950
## [2,] 0.0001979059 0.0005443916  0.0463408
```

```
## [3,] 0.0005405551 0.0014869354  0.1265745
##
##
## [[22]]
## [[22]]$stocgrowth
## [1] 0.910793
##
## [[22]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2635411034 0.7252570462 61.72436000
## [2,] 0.0002166114 0.0005961091  0.05073274
## [3,] 0.0005915451 0.0016279137  0.13854677
##
## [[22]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.253391e-02 6.201004e-02 5.276900244
## [2,] 1.851976e-05 5.096406e-05 0.004336881
## [3,] 5.058065e-05 1.391904e-04 0.011844772
##
## [[22]]$bad
##              [,1]        [,2]        [,3]
## [1,] 0.2410071938 0.663247004 56.44745976
## [2,] 0.0001980917 0.000545145  0.04639586
## [3,] 0.0005409644 0.001488723  0.12670200
##
##
## [[23]]
## [[23]]$stocgrowth
## [1] 0.9106673
##
## [[23]]$overall
##              [,1]        [,2]        [,3]
## [1,] 0.2635812222 0.725702595 61.74972773
## [2,] 0.0002166784 0.000596569  0.05076155
## [3,] 0.0005916169 0.001628864  0.13859944
##
## [[23]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.238488e-02 6.162820e-02 5.243315872
## [2,] 1.840009e-05 5.065801e-05 0.004309937
## [3,] 5.024464e-05 1.383292e-04 0.011769036
##
## [[23]]$bad
##              [,1]        [,2]        [,3]
## [1,] 0.2411963378 0.664074396 56.50641186
## [2,] 0.0001982783 0.000545911  0.04645161
## [3,] 0.0005413722 0.001490534  0.12683041
##
##
## [[24]]
## [[24]]$stocgrowth
## [1] 0.9105366
##
## [[24]]$overall
```

```
##              [,1]         [,2]        [,3]
## [1,] 0.2636228275 0.7261649524 61.77604700
## [2,] 0.0002167479 0.0005970464  0.05079145
## [3,] 0.0005916913 0.0016298491  0.13865409
##
## [[24]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.223772e-02 6.125217e-02 5.210200455
## [2,] 1.828203e-05 5.035692e-05 0.004283395
## [3,] 4.991278e-05 1.374809e-04 0.011694344
##
## [[24]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2413851047 0.6649127867 56.56584655
## [2,] 0.0001984659 0.0005466895  0.04650805
## [3,] 0.0005417785 0.0014923683  0.12695975
##
##
## [[25]]
## [[25]]$stocgrowth
## [1] 0.9104011
##
## [[25]]$overall
##              [,1]        [,2]        [,3]
## [1,] 0.2636658950 0.726643879 61.80330356
## [2,] 0.0002168199 0.000597541  0.05082241
## [3,] 0.0005917684 0.001630870  0.13871068
##
## [[25]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.209239e-02 0.0608818170 5.177544334
## [2,] 1.816553e-05 0.0000500607 0.004257246
## [3,] 4.958499e-05 0.0001366452 0.011620675
##
## [[25]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2415735052 0.6657620617 56.62575922
## [2,] 0.0001986544 0.0005474803  0.04656516
## [3,] 0.0005421834 0.0014942248  0.12709001
##
##
## [[26]]
## [[26]]$stocgrowth
## [1] 0.9102607
##
## [[26]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2637104006 0.7271391409 61.83148353
## [2,] 0.0002168943 0.0005980526  0.05085443
## [3,] 0.0005918480 0.0016319257  0.13876919
##
## [[26]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.194885e-02 6.051703e-02 5.145338113
```

```
## [2,] 1.805056e-05 4.976923e-05 0.004231482
## [3,] 4.926119e-05 1.358219e-04 0.011548007
##
## [[26]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2417615496 0.6666221105 56.68614542
## [2,] 0.0001988437 0.0005482833  0.04662294
## [3,] 0.0005425868 0.0014961037  0.12722118
##
##
## [[27]]
## [[27]]$stocgrowth
## [1] 0.9101156
##
## [[27]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2637563210 0.7276505129 61.86057345
## [2,] 0.0002169710 0.0005985808  0.05088748
## [3,] 0.0005919301 0.0016330156  0.13882958
##
## [[27]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.180707e-02 6.015769e-02 5.113572655
## [2,] 1.793711e-05 4.948241e-05 0.004206095
## [3,] 4.894131e-05 1.350108e-04 0.011476321
##
## [[27]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2419492479 0.6674928261 56.74700079
## [2,] 0.0001990339 0.0005490984  0.04668139
## [3,] 0.0005429888 0.0014980048  0.12735326
##
##
## [[28]]
## [[28]]$stocgrowth
## [1] 0.9099657
##
## [[28]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2638036335 0.7281777757 61.89056022
## [2,] 0.0002170502 0.0005991256  0.05092157
## [3,] 0.0005920147 0.0016341394  0.13889183
##
## [[28]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.166702e-02 5.980367e-02 5.082239067
## [2,] 1.782513e-05 4.920014e-05 0.004181077
## [3,] 4.862528e-05 1.342115e-04 0.011405597
##
## [[28]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2421366101 0.6683741047 56.80832115
## [2,] 0.0001992250 0.0005499255  0.04674049
## [3,] 0.0005433895 0.0014999279  0.12748623
```

```
##
##
## [[29]]
## [[29]]$stocgrowth
## [1] 0.9098112
##
## [[29]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2638523160 0.7287207167 61.92143112
## [2,] 0.0002171316 0.0005996867  0.05095666
## [3,] 0.0005921018 0.0016352965  0.13895591
##
## [[29]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.152867e-02 5.945487e-02 5.051328697
## [2,] 1.771460e-05 4.892231e-05 0.004156419
## [3,] 4.831303e-05 1.334239e-04 0.011335815
##
## [[29]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2423236456 0.6692658462 56.87010243
## [2,] 0.0001994170 0.0005507644  0.04680024
## [3,] 0.0005437888 0.0015018726  0.12762009
##
##
## [[30]]
## [[30]]$stocgrowth
## [1] 0.9096521
##
## [[30]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2639023467 0.7292791296 61.95317378
## [2,] 0.0002172153 0.0006002639  0.05099275
## [3,] 0.0005921913 0.0016364866  0.13902179
##
## [[30]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.139198e-02 5.911118e-02 5.020833123
## [2,] 1.760549e-05 4.864883e-05 0.004132115
## [3,] 4.800449e-05 1.326476e-04 0.011266958
##
## [[30]]$bad
##              [,1]        [,2]        [,3]
## [1,] 0.2425103639 0.670167953 56.93234066
## [2,] 0.0001996098 0.000551615  0.04686064
## [3,] 0.0005441868 0.001503839  0.12775484
##
##
## [[31]]
## [[31]]$stocgrowth
## [1] 0.9094884
##
## [[31]]$overall
##              [,1]        [,2]        [,3]
```

```
## [1,] 0.2639537046 0.729852814 61.98577617
## [2,] 0.0002173012 0.000600857  0.05102983
## [3,] 0.0005922831 0.001637709  0.13908946
##
## [[31]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.125693e-02 5.877248e-02 4.990744145
## [2,] 1.749778e-05 4.837961e-05 0.004108157
## [3,] 4.769960e-05 1.318825e-04 0.011199007
##
## [[31]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2426967741 0.6710803322 56.99503203
## [2,] 0.0001998035 0.0005524773  0.04692167
## [3,] 0.0005445835 0.0015058266  0.12789045
##
##
## [[32]]
## [[32]]$stocgrowth
## [1] 0.9093202
##
## [[32]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2640063690 0.7304415767 62.01922659
## [2,] 0.0002173894 0.0006014658  0.05106788
## [3,] 0.0005923772 0.0016389637  0.13915889
##
## [[32]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.112348e-02 5.843869e-02 4.961053780
## [2,] 1.739143e-05 4.811456e-05 0.004084539
## [3,] 4.739828e-05 1.311283e-04 0.011131945
##
## [[32]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2428828851 0.6720028916 57.05817281
## [2,] 0.0001999980 0.0005533512  0.04698334
## [3,] 0.0005449789 0.0015078354  0.12802694
##
##
## [[33]]
## [[33]]$stocgrowth
## [1] 0.9091474
##
## [[33]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2640603198 0.7310452283 62.05351366
## [2,] 0.0002174798 0.0006020901  0.05110689
## [3,] 0.0005924736 0.0016402499  0.13923004
##
## [[33]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.099161e-02 5.810968e-02 4.931754254
## [2,] 1.728642e-05 4.785359e-05 0.004061253
```

```
## [3,] 4.710048e-05 1.303848e-04 0.011065754
##
## [[33]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2430687055 0.6729355436 57.12175941
## [2,] 0.0002001934 0.0005542365  0.04704564
## [3,] 0.0005453731 0.0015098651  0.12816429
##
##
## [[34]]
## [[34]]$stocgrowth
## [1] 0.9089703
##
## [[34]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2641155374 0.7316635861 62.08862633
## [2,] 0.0002175723 0.0006027297  0.05114684
## [3,] 0.0005925723 0.0016415674  0.13930291
##
## [[34]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.086129e-02 5.778538e-02 4.902837994
## [2,] 1.718273e-05 4.759661e-05 0.004038292
## [3,] 4.680614e-05 1.296517e-04 0.011000418
##
## [[34]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2432542439 0.6738782027 57.18578834
## [2,] 0.0002003895 0.0005551331  0.04710855
## [3,] 0.0005457661 0.0015119157  0.12830249
##
##
## [[35]]
## [[35]]$stocgrowth
## [1] 0.9087888
##
## [[35]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2641720027 0.7322964727 62.12455382
## [2,] 0.0002176669 0.0006033845  0.05118774
## [3,] 0.0005926732 0.0016429158  0.13937746
##
## [[35]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.073249e-02 5.746569e-02 4.874297622
## [2,] 1.708032e-05 4.734354e-05 0.004015651
## [3,] 4.651518e-05 1.289290e-04 0.010935919
##
## [[35]]$bad
##              [,1]        [,2]        [,3]
## [1,] 0.2434395086 0.674830786 57.25025620
## [2,] 0.0002005866 0.000556041  0.04717209
## [3,] 0.0005461580 0.001513987  0.12844154
##
```

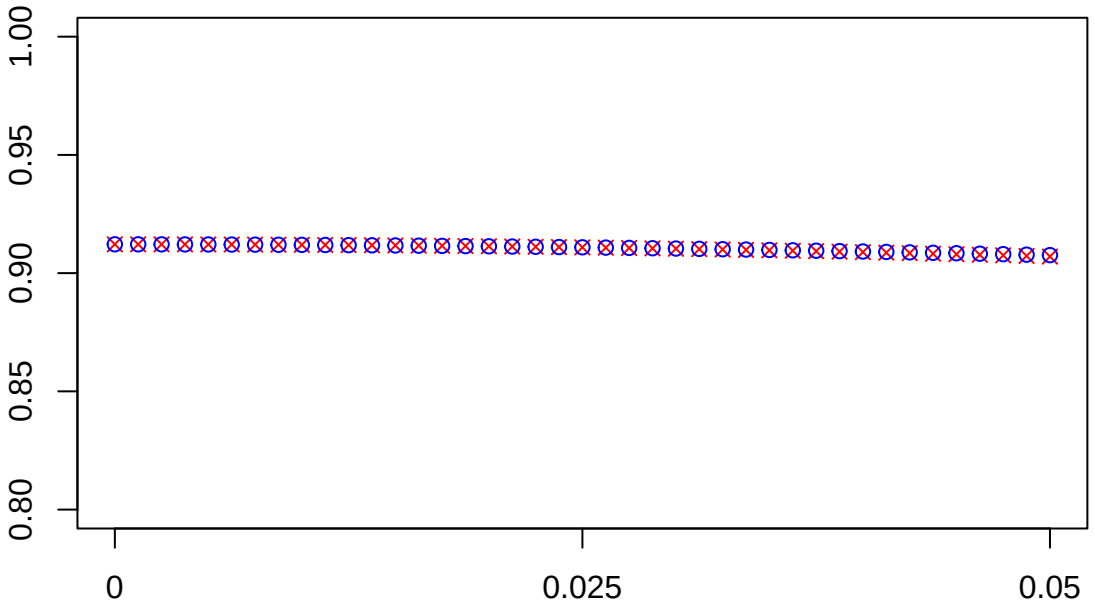
```
##
## [[36]]
## [[36]]$stocgrowth
## [1] 0.9086029
##
## [[36]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2642296968 0.7329437157 62.16128567
## [2,] 0.0002177636 0.0006040544  0.05122955
## [3,] 0.0005927762 0.0016442947  0.13945367
##
## [[36]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.060519e-02 0.0571505008 4.846125950
## [2,] 1.697919e-05 0.0000470943 0.003993322
## [3,] 4.622756e-05 0.0001282162 0.010872244
##
## [[36]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2436245076 0.6757932149 57.31515972
## [2,] 0.0002007844 0.0005569601  0.04723623
## [3,] 0.0005465486 0.0015160785  0.12858143
##
##
## [[37]]
## [[37]]$stocgrowth
## [1] 0.9084127
##
## [[37]]$overall
##              [,1]        [,2]        [,3]
## [1,] 0.2642886015 0.733605148 62.19881169
## [2,] 0.0002178624 0.000604739  0.05127229
## [3,] 0.0005928814 0.001645704  0.13953153
##
## [[37]]$good
##              [,1]         [,2]       [,3]
## [1,] 2.047935e-02 5.683974e-02 4.81831597
## [2,] 1.687930e-05 4.684881e-05 0.00397130
## [3,] 4.594322e-05 1.275134e-04 0.01080937
##
## [[37]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2438092489 0.6767654107 57.38049572
## [2,] 0.0002009831 0.0005578902  0.04730099
## [3,] 0.0005469382 0.0015181904  0.12872215
##
##
## [[38]]
## [[38]]$stocgrowth
## [1] 0.9082182
##
## [[38]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2643486991 0.7342806068 62.23712197
```

```
## [2,] 0.0002179632 0.0006054384  0.05131592
## [3,] 0.0005929887 0.0016471426  0.13961101
##
## [[38]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.035496e-02 5.653331e-02 4.790860863
## [2,] 1.678063e-05 4.660699e-05 0.003949578
## [3,] 4.566210e-05 1.268202e-04 0.010747298
##
## [[38]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2439937401 0.6777472990 57.44626111
## [2,] 0.0002011826 0.0005588314  0.04736634
## [3,] 0.0005473266 0.0015203224  0.12886371
##
##
## [[39]]
## [[39]]$stocgrowth
## [1] 0.9080195
##
## [[39]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2644099719 0.7349699348 62.27620686
## [2,] 0.0002180660 0.0006061522  0.05136045
## [3,] 0.0005930981 0.0016486109  0.13969209
##
## [[39]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.023198e-02 5.623113e-02 4.763753962
## [2,] 1.668316e-05 4.636878e-05 0.003928151
## [3,] 4.538414e-05 1.261365e-04 0.010685998
##
## [[39]]$bad
##              [,1]         [,2]       [,3]
## [1,] 0.2441779889 0.6787388073 57.5124529
## [2,] 0.0002013829 0.0005597835  0.0474323
## [3,] 0.0005477140 0.0015224744  0.1290061
##
##
## [[40]]
## [[40]]$stocgrowth
## [1] 0.9078167
##
## [[40]]$overall
##              [,1]         [,2]        [,3]
## [1,] 0.2644724030 0.7356729788 62.31605697
## [2,] 0.0002181708 0.0006068805  0.05140586
## [3,] 0.0005932095 0.0016501083  0.13977475
##
## [[40]]$good
##              [,1]         [,2]        [,3]
## [1,] 2.011040e-02 5.593311e-02 4.736988782
## [2,] 1.658686e-05 4.613409e-05 0.003907013
## [3,] 4.510929e-05 1.254621e-04 0.010625460
```

```
##
## [[40]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2443620025 0.6797398657 57.57906819
## [2,] 0.0002015840 0.0005607464  0.04749884
## [3,] 0.0005481003 0.0015246462  0.12914929
##
##
## [[41]]
## [[41]]$stocgrowth
## [1] 0.9076096
##
## [[41]]$overall
##              [,1]        [,2]        [,3]
## [1,] 0.2645359758 0.736389590 62.35666317
## [2,] 0.0002182776 0.000607623  0.05145214
## [3,] 0.0005933230 0.001651635  0.13985897
##
## [[41]]$good
##              [,1]         [,2]        [,3]
## [1,] 1.999019e-02 5.563918e-02 4.710558991
## [2,] 1.649172e-05 4.590286e-05 0.003886159
## [3,] 4.483750e-05 1.247968e-04 0.010565671
##
## [[41]]$bad
##              [,1]         [,2]        [,3]
## [1,] 0.2445457882 0.6807504063 57.64610418
## [2,] 0.0002017859 0.0005617201  0.04756598
## [3,] 0.0005484855 0.0015268377  0.12929330
```

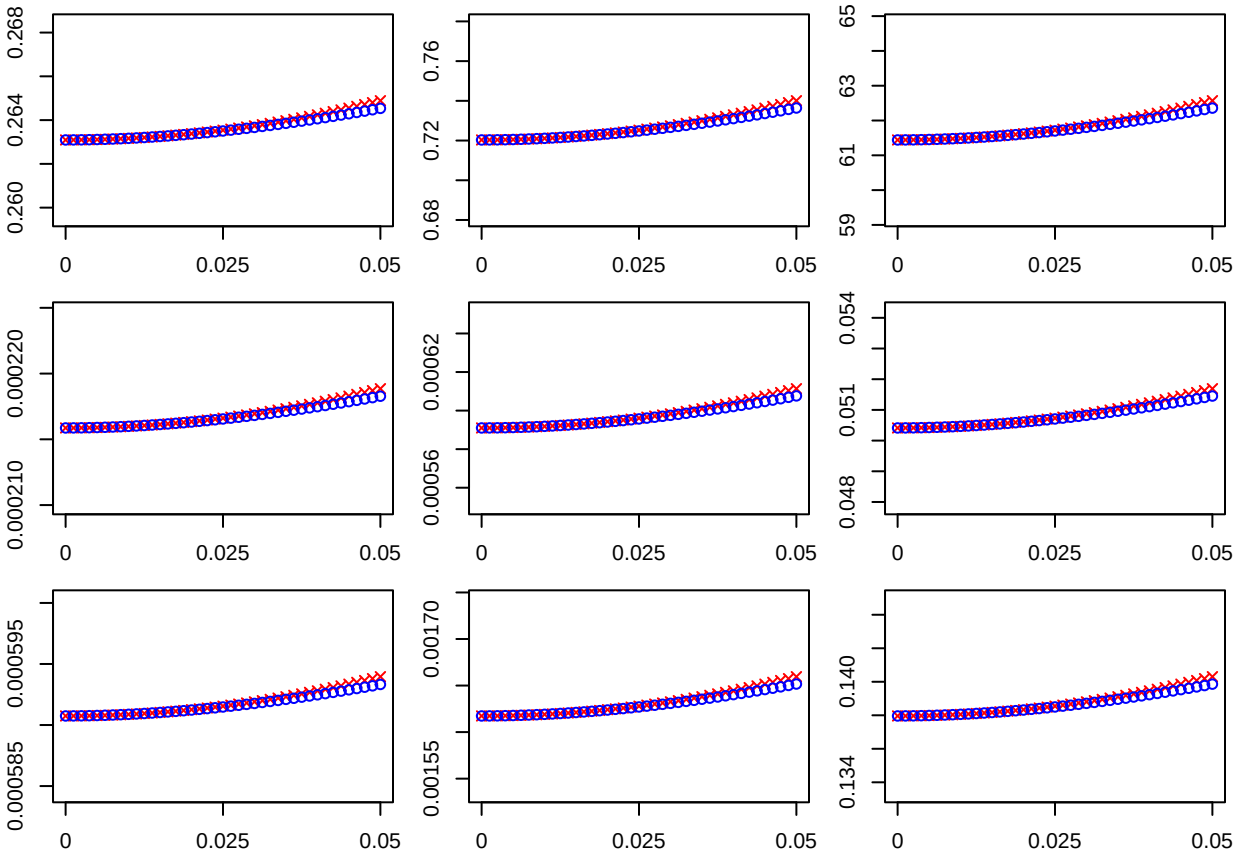
As a check, we compare the stochastic growth rate as computed from the small noise approximation and as estimated from simulations:

```
# extracts data about the growth rate both as approximated via
# small noise and as estimated from simulations
sgrdat <- sapply(1:length(v.par), function(x) {
  goodM <- (all.mats[[x]])$good
  badM <- (all.mats[[x]])$bad
  snasgr <- sna(goodM,badM)
  estsgr <- (vdata[[x]])$stocgrowth
  dat <- cbind(snasgr,estsgr)
  return(dat)
}, simplify = TRUE)
rownames(sgrdat) <- c("sna","est")

# Plots stochastic growth both as approximated via
# small noise and as estimated from simulations
ylimits <- c(0.8,1)
plot(v.par,sgrdat["sna",],ylim=ylimits,xlab="",ylab="",
     pch = 4, col="red", xaxt='n')
axis(1,at = c(v.par[1],tail(v.par,n=1)/2,tail(v.par,n=1)),
     c(v.par[1],tail(v.par,n=1)/2,tail(v.par,n=1)))
points(v.par,sgrdat["est",], pch = 1, col="blue")
```



As another check, we plots the environment-independent sensitivities both as approximated via small noise and as estimated from simulations:

```
eisensdat <- sapply(1:length(v.par), function(x) {
  goodM <- (all.mats[[x]])$good
  badM <- (all.mats[[x]])$bad
  snaeisens <- (csna.bij(goodM,badM))
  esteisens <- ((vdata[[x]])$overall)
  dat <- list(sna=snaeisens,est=esteisens)
  return(dat)
}, simplify = FALSE)
eisensdat <- sapply(1:length(v.par),
                    function(x) cbind(c(eisensdat[[x]]$sna),
                                      c(eisensdat[[x]]$est)),
                    simplify = TRUE)

par(mfcol=c(3,3),mar=c(2,2,1,1))
invisible(sapply(1:(nr.stages^2), function(x) {
  my.min <- min(eisensdat[x, ],eisensdat[x+(nr.stages^2), ])
  my.max <- max(eisensdat[x, ],eisensdat[x+(nr.stages^2), ])
  ylimits <- c(my.min-2*(my.max-my.min),my.max+2*(my.max-my.min))
  # plots environment-independent sensitivities as approximated
  # via small noise
  plot(v.par, eisensdat[x, ], ylim=ylimits, ylab="", xlab="",
       pch = 4, col="red", xaxt='n')
  axis(1,at = c(v.par[1],tail(v.par,n=1)/2,tail(v.par,n=1)),
       labels = c(v.par[1],tail(v.par,n=1)/2,tail(v.par,n=1)))
  # add to plot the environment-independent sensitivities
  # as estimated from simulations
  points(v.par, eisensdat[(x+(nr.stages^2)), ], pch = 1, col="blue")
}, simplify = TRUE))
```



From the data generated above, we extract environment-specific sensitivities for both environmental states both as approximated via small noise and as estimated from simulations. Sensitivities for good years:

```
# Good years sensitivities
goodsensdat <- sapply(1:length(v.par), function(x) {
  goodM <- (all.mats[[x]])$good
  badM <- (all.mats[[x]])$bad
  conjgoodsens1 <- bijk(goodM,badM)$good
  conjgoodsens2 <- pr.good*((vdata[[x]])$overall)
  snagoodsens <- (sna.bijk(goodM,badM))$good
  estgoodsens <- ((vdata[[x]])$good)
  dat <- list(conj1= conjgoodsens1,
              conj2= conjgoodsens2,
              sna=snagoodsens,
              est=estgoodsens)
  return(dat)
}, simplify = FALSE)
goodsensdat <- sapply(1:length(v.par),
                      function(x) {
                      dat <-cbind(c(goodsensdat[[x]]$conj1),
                                  c(goodsensdat[[x]]$conj2),
                                  c(goodsensdat[[x]]$sna),
                                  c(goodsensdat[[x]]$est))
                        return(dat)},
                      simplify = TRUE)
```

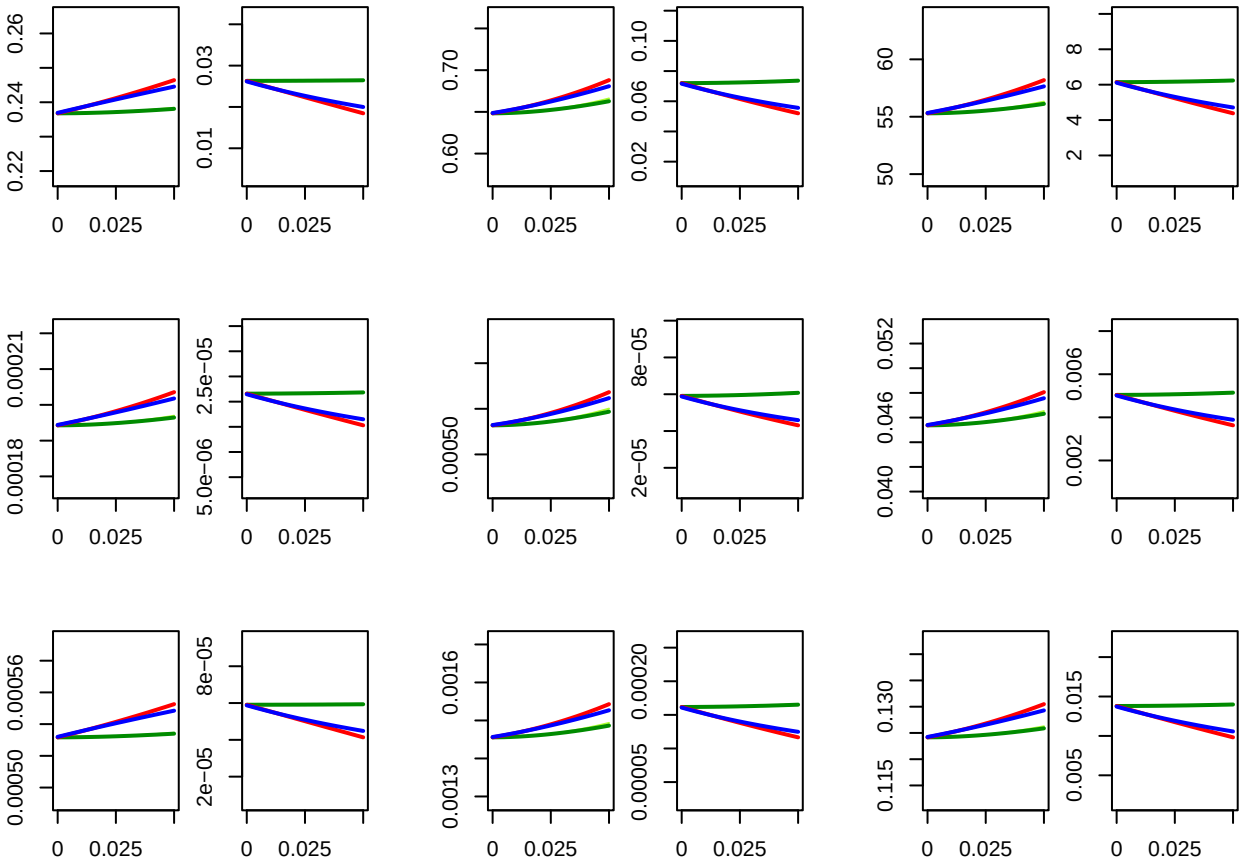
Sensitivities for bad years:

```
# Bad years sensitivities
badsensdat <- sapply(1:length(v.par), function(x) {
  goodM <- (all.mats[[x]])$good
  badM <- (all.mats[[x]])$bad
  conjbadsens1 <- bijk(goodM,badM)$bad
  conjbadsens2 <- pr.bad*((vdata[[x]])$overall)
  snabadsens <- (sna.bijk(goodM,badM))$bad
  estbadsens <- ((vdata[[x]])$bad)
  dat <- list(conj1= conjbadsens1,
              conj2= conjbadsens2,
              sna=snabadsens,
              est=estbadsens)
  return(dat)
}, simplify = FALSE)
badsensdat <- sapply(1:length(v.par),
                     function(x) {
                       dat <- cbind(c(badsensdat[[x]]$conj1),
                                    c(badsensdat[[x]]$conj2),
                                    c(badsensdat[[x]]$sna),
                                    c(badsensdat[[x]]$est))
                       return(dat)},
                     simplify = TRUE)
```

Generation of Figure 1

```
par(mar=c(2,2,0.4,0.5))
layout(matrix(c(1, 9+1, 0, 4, 9+4, 0,  7, 9+7,
                0,   0, 0, 0,    0,0,  0,   0,
                2, 9+2, 0, 5, 9+5, 0,  8, 9+8,
                0,   0, 0, 0,    0,0,  0,   0,
                3, 9+3, 0,  6, 9+6, 0, 9, 9+9),
              ncol = 8, nrow=5, byrow=TRUE),
       widths = c(1,1,0.3,1,1,0.3,1,1),
       heights = c(1,0.3,1,0.3,1))


# plot sensitivities for bad years
invisible(sapply(1:(nr.stages^2), function(x) {
  yldat <- c(badsensdat[x, ],
             badsensdat[x+(nr.stages^2), ],
             badsensdat[x+2*(nr.stages^2), ])
  my.min <- min(yldat)
  my.max <- max(yldat)
  ylimits <- c(my.min-2*(my.max-my.min),my.max+2*(my.max-my.min))
  # plot sensitivities of \ln\lambda_{s} to additive changes
  # in bad years only according to conjecture (Eq. 6
  # in the manuscript) with b_{i,j} approximated with small noise
  plot(v.par, badsensdat[x, ], ylim=ylimits, ylab="", xlab="",
       col="greenyellow", type = "l", xaxt = "n")
  axis(1, at = c(v.par[1], mean(v.par), tail(v.par, 1)),
       labels = c(v.par[1], mean(v.par), tail(v.par, 1)))
  # add to plot the sensitivities of \ln\lambda_{s} to additive
  # changes in bad years according to conjecture (Eq. 6
  # in the manuscript) with b_{i,j} estimated via simulations
```

```
  lines(v.par, badsensdat[(x+(nr.stages^2)), ],
        col="green4", lwd=2)
  # add to plot the sensitivities of \ln\lambda_{s} to
  # additive changes in bad years only according to
  # small noise approximation (Eq. 15 in the manuscript)
  lines(v.par, badsensdat[(x+2*(nr.stages^2)), ],
        col="red", lwd=2)
  # add to plot the sensitivities of \ln\lambda_{s} to
  # additive changes in bad years only according to
  # simulations
  lines(v.par, badsensdat[(x+3*(nr.stages^2)), ],
        col="blue", lwd=2)
}, simplify = TRUE))

# plot sensitivities for good years
invisible(sapply(1:(nr.stages^2), function(x) {
  yldat <- c(goodsensdat[x, ],
             goodsensdat[x+(nr.stages^2), ],
             goodsensdat[x+2*(nr.stages^2), ],
             goodsensdat[x+3*(nr.stages^2), ])
  my.min <- min(yldat)
  my.max <- max(yldat)
  ylimits <- c(my.min-2*(my.max-my.min),my.max+2*(my.max-my.min))
  # plot sensitivities of \ln\lambda_{s} to additive changes
  # in good years only according to conjecture (Eq. 6
  # in the manuscript) with b_{i,j} approximated with small noise
  plot(v.par, goodsensdat[x, ], ylim=ylimits, ylab="", xlab="",
       col="greenyellow", type = "l", xaxt="n")
  axis(1, at = c(v.par[1], mean(v.par), tail(v.par, 1)),
       labels = c(v.par[1], mean(v.par), tail(v.par, 1)))
  # add to plot the sensitivities of \ln\lambda_{s} to
  # additive changes in good years according to conjecture (Eq. 7
  # in the manuscript) with b_{i,j} estimated via simulations
  lines(v.par, goodsensdat[(x+(nr.stages^2)), ],
        col="green4", lwd=2)
  # add to plot the sensitivities of \ln\lambda_{s} to
  # additive changes in good years only according to
  # small noise approximation (Eq. 15 in the manuscript)
  lines(v.par, goodsensdat[(x+2*(nr.stages^2)), ],
        col="red", lwd=2)
  # add to plot the sensitivities of \ln\lambda_{s} to
  # additive changes in good years only according to
  # simulations
  lines(v.par, goodsensdat[(x+3*(nr.stages^2)), ],
        col="blue", lwd=2)
}, simplify = TRUE))
```



Code for Figure 2

The R environment gets cleaned:

```
rm(list=ls())
```

The files 020306appxa.txt (https://www.journals.uchicago.edu/doi/suppl/10.1086/378648/suppl_file/020306appxa.txt) and 020306appxb.txt (https://www.journals.uchicago.edu/doi/suppl/10.1086/378648/suppl_file/020306appxb.txt) are in the Appendix of Tuljapurkar et al (2003), *AmNat* 162(4):pp. 489−502. These files contain the seven 8×8 projection matrices for the tropical understory shrub (*Ardisia escallonioides*) corresponding to the stages: seed (1), seedling (2), juvenile (3), pre-rep (4), small rep (5), med rep (6), large rep (7), v. large rep (8). Make sure that these files are in the working directory.

Set the number of stages:

```
nr.stages <- 8
```

Extract, stores in a list and report all 7 projection matrices, one for each of the 7 environmental states corresponding to different percentages of percent of forest canopy openness: 65% (1), 55% (2), 45% (3), 35% (4), 25% (5), 15% (6), <5% (7):

```
mats <- read.delim("020306appxa.txt", header = FALSE, skip = 12)
mats <- mats[-(9*(1:6)),3:10]
rownames(mats) <- NULL
mats <- sapply(1:(nr.stages-1), function(x) {
  m <- mats[((nr.stages*(x-1))+1):(nr.stages*x),]
  m <- as.matrix(m)
```

```
}, simplify = FALSE)
mats
```

```
## [[1]]
##    V3   V4    V5    V6     V7      V8      V9    V10
## 1 0.0 0.01 11.40 110.0 790.70 1450.60 3216.20 4066.9
## 2 0.1 0.00  0.00   0.0   0.00    0.00    0.00    0.0
## 3 0.0 0.70  0.57   0.2   0.01    0.00    0.00    0.0
## 4 0.0 0.00  0.12   0.4   0.11    0.00    0.00    0.0
## 5 0.0 0.00  0.01   0.4   0.10    0.10    0.00    0.0
## 6 0.0 0.00  0.00   0.0   0.50    0.48    0.00    0.0
## 7 0.0 0.00  0.00   0.0   0.13    0.29    0.33    0.0
## 8 0.0 0.00  0.00   0.0   0.03    0.14    0.67    1.0
##
## [[2]]
##     V3  V4   V5    V6     V7     V8     V9     V10
## 9  0.0 0.0 1.10 58.90 190.30 481.50 702.00 1508.10
## 10 0.1 0.0 0.00  0.00   0.00   0.00   0.00    0.00
## 11 0.0 0.7 0.70  0.10   0.00   0.00   0.00    0.00
## 12 0.0 0.0 0.08  0.49   0.07   0.00   0.00    0.00
## 13 0.0 0.0 0.01  0.27   0.50   0.03   0.03    0.00
## 14 0.0 0.0 0.00  0.12   0.23   0.51   0.21    0.00
## 15 0.0 0.0 0.00  0.00   0.07   0.33   0.38    0.00
## 16 0.0 0.0 0.00  0.00   0.03   0.10   0.28    0.94
##
## [[3]]
##     V3  V4   V5    V6     V7     V8     V9     V10
## 17 0.0 0.7 1.30 62.10 306.90 579.70 890.60 1843.00
## 18 0.1 0.0 0.00  0.00   0.00   0.00   0.00    0.00
## 19 0.0 0.7 0.43  0.00   0.00   0.00   0.00    0.00
## 20 0.0 0.0 0.22  0.55   0.00   0.00   0.00    0.00
## 21 0.0 0.0 0.00  0.27   0.43   0.00   0.10    0.00
## 22 0.0 0.0 0.00  0.05   0.45   0.64   0.15    0.13
## 23 0.0 0.0 0.00  0.00   0.06   0.31   0.50    0.13
## 24 0.0 0.0 0.00  0.00   0.00   0.03   0.20    0.75
##
## [[4]]
##     V3  V4   V5    V6    V7    V8     V9    V10
## 25 0.0 0.0 1.80 14.30 18.10 61.20 125.20 179.40
## 26 0.1 0.0 0.00  0.00  0.00  0.00   0.00   0.00
## 27 0.0 0.7 0.52  0.04  0.00  0.00   0.00   0.08
## 28 0.0 0.0 0.19  0.32  0.06  0.00   0.00   0.00
## 29 0.0 0.0 0.03  0.16  0.42  0.05   0.02   0.00
## 30 0.0 0.0 0.03  0.36  0.38  0.43   0.10   0.00
## 31 0.0 0.0 0.00  0.00  0.02  0.30   0.24   0.00
## 32 0.0 0.0 0.00  0.00  0.00  0.16   0.56   0.92
##
## [[5]]
##     V3   V4   V5    V6     V7     V8     V9     V10
## 33 0.0 0.07 5.00 65.30 189.10 330.70 679.70 1142.80
## 34 0.1 0.00 0.00  0.00   0.00   0.00   0.00    0.00
## 35 0.0 0.70 0.66  0.04   0.00   0.00   0.00    0.00
## 36 0.0 0.00 0.25  0.50   0.00   0.00   0.00    0.00
## 37 0.0 0.00 0.02  0.33   0.52   0.00   0.00    0.00
```

```
## 38 0.0 0.00 0.00  0.08   0.32   0.76   0.17    0.00
## 39 0.0 0.00 0.00  0.00   0.03   0.17   0.50    0.17
## 40 0.0 0.00 0.00  0.00   0.02   0.03   0.33    0.83
##
## [[6]]
##     V3  V4   V5    V6    V7     V8     V9    V10
## 41 0.0 1.8 2.30 12.60 73.40 153.40 568.90 1431.7
## 42 0.1 0.0 0.00  0.00  0.00   0.00   0.00    0.0
## 43 0.0 0.7 0.76  0.05  0.04   0.02   0.00    0.0
## 44 0.0 0.0 0.09  0.66  0.06   0.00   0.00    0.0
## 45 0.0 0.0 0.00  0.29  0.56   0.04   0.07    0.0
## 46 0.0 0.0 0.00  0.00  0.29   0.67   0.00    0.0
## 47 0.0 0.0 0.00  0.00  0.04   0.27   0.60    0.0
## 48 0.0 0.0 0.00  0.00  0.00   0.00   0.33    1.0
##
## [[7]]
##     V3  V4   V5   V6   V7     V8     V9 V10
## 49 0.0 0.0 0.00 0.00 0.90 2.3000 2.6000 0.8
## 50 0.1 0.0 0.00 0.00 0.00 0.0000 0.0000 0.0
## 51 0.0 0.7 0.95 0.17 0.00 0.0000 0.0000 0.0
## 52 0.0 0.0 0.01 0.66 0.00 0.0000 0.0000 0.0
## 53 0.0 0.0 0.00 0.17 0.96 0.0000 0.0000 0.0
## 54 0.0 0.0 0.00 0.00 0.04 0.8800 0.0400 0.0
## 55 0.0 0.0 0.00 0.00 0.00 0.0001 0.9599 0.0
## 56 0.0 0.0 0.00 0.00 0.00 0.0000 0.0001 1.0
```

In this dataset, there are 3 transition matrices for the seven environmental states. Set the number of environments:

```
nr.env <- length(mats)
# and assign environments numbers 1,2,..., up
# to the dimensions of the transition matrix
# for the environment
my.envs <- 1:nr.env
my.envs
```

```
## [1] 1 2 3 4 5 6 7
```

Each environmental transition matrix correspond to a hurricane frequency 0.81, 0.081, 0.0081, where the (i, j) entry is the probability of environment i at time $t + 1$ given environment j at time t, i.e. matrices are column stochastic. Extract the transition matrices between environmental states, store them in a list and report them:

```
env.mat <- read.delim("020306appxb.txt",
                      header = FALSE, skip = 8)
env.mat <- env.mat[-c(8,9,17,18),-c(1,2)]
env.mat <- lapply(1:3, function(x) {
  m <- env.mat[((nr.env*(x-1))+1):(nr.env*x),]
  m <- as.matrix(m)
})
env.mat
```

```
## [[1]]
##       V3     V4     V5     V6     V7     V8     V9
## 1 0.9525 0.8100 0.6202 0.4303 0.2936 0.1569 0.0785
## 2 0.0475 0.1425 0.1898 0.1898 0.1367 0.1367 0.0785
## 3 0.0000 0.0475 0.0950 0.1898 0.1898 0.1367 0.1367
```

```
## 4 0.0000 0.0000 0.0950 0.0475 0.1898 0.1898 0.1367
## 5 0.0000 0.0000 0.0000 0.1425 0.0475 0.1898 0.1898
## 6 0.0000 0.0000 0.0000 0.0000 0.1425 0.0475 0.1898
## 7 0.0000 0.0000 0.0000 0.0000 0.0000 0.1425 0.1900
##
## [[2]]
##        V3     V4     V5     V6     V7     V8     V9
## 10 0.7703 0.0810 0.0620 0.0430 0.0294 0.0157 0.0078
## 11 0.2298 0.6893 0.0190 0.0190 0.0137 0.0137 0.0078
## 12 0.0000 0.2298 0.4595 0.0190 0.0190 0.0137 0.0137
## 13 0.0000 0.0000 0.4595 0.2298 0.0190 0.0190 0.0137
## 14 0.0000 0.0000 0.0000 0.6893 0.2298 0.0190 0.0190
## 15 0.0000 0.0000 0.0000 0.0000 0.6893 0.2298 0.0190
## 16 0.0000 0.0000 0.0000 0.0000 0.0000 0.6893 0.9190
##
## [[3]]
##       V3     V4     V5     V6     V7     V8     V9
## 19 0.752 0.0081 0.0062 0.0043 0.0029 0.0016 0.0008
## 20 0.248 0.7439 0.0019 0.0019 0.0014 0.0014 0.0008
## 21 0.000 0.2480 0.4960 0.0019 0.0019 0.0014 0.0014
## 22 0.000 0.0000 0.4960 0.2480 0.0019 0.0019 0.0014
## 23 0.000 0.0000 0.0000 0.7439 0.2480 0.0019 0.0019
## 24 0.000 0.0000 0.0000 0.0000 0.7439 0.2480 0.0019
## 25 0.000 0.0000 0.0000 0.0000 0.0000 0.7439 0.9919
```

Check the column sums of these matrices:

```
sapply(env.mat, function(x) colSums(x), simplify = TRUE)
```

```
##      [,1]   [,2]   [,3]
## V3 1.0000 1.0001 1.0000
## V4 1.0000 1.0001 1.0000
## V5 1.0000 1.0000 1.0001
## V6 0.9999 1.0001 1.0000
## V7 0.9999 1.0002 1.0000
## V8 0.9999 1.0002 1.0001
## V9 1.0000 1.0000 1.0001
```

As not all environmental transition matrices are exactly column stochastic, their columns are normalized to add up to 1:

```
env.mat <- lapply(env.mat,
                  function(x) sweep(x, 2, colSums(x), "/"))
```

The column sums are checked again and the normalized matrices reported:

```
sapply(env.mat,
       function(x) colSums(x), simplify = TRUE)
```

```
##    [,1] [,2] [,3]
## V3    1    1    1
## V4    1    1    1
## V5    1    1    1
## V6    1    1    1
## V7    1    1    1
## V8    1    1    1
## V9    1    1    1
```

```
env.mat
```

```
## [[1]]
##       V3     V4     V5         V6         V7         V8     V9
## 1 0.9525 0.8100 0.6202 0.43034303 0.29362936 0.15691569 0.0785
## 2 0.0475 0.1425 0.1898 0.18981898 0.13671367 0.13671367 0.0785
## 3 0.0000 0.0475 0.0950 0.18981898 0.18981898 0.13671367 0.1367
## 4 0.0000 0.0000 0.0950 0.04750475 0.18981898 0.18981898 0.1367
## 5 0.0000 0.0000 0.0000 0.14251425 0.04750475 0.18981898 0.1898
## 6 0.0000 0.0000 0.0000 0.00000000 0.14251425 0.04750475 0.1898
## 7 0.0000 0.0000 0.0000 0.00000000 0.00000000 0.14251425 0.1900
##
## [[2]]
##          V3        V4     V5        V6         V7         V8     V9
## 10 0.770223 0.0809919 0.0620 0.0429957 0.02939412 0.01569686 0.0078
## 11 0.229777 0.6892311 0.0190 0.0189981 0.01369726 0.01369726 0.0078
## 12 0.000000 0.2297770 0.4595 0.0189981 0.01899620 0.01369726 0.0137
## 13 0.000000 0.0000000 0.4595 0.2297770 0.01899620 0.01899620 0.0137
## 14 0.000000 0.0000000 0.0000 0.6892311 0.22975405 0.01899620 0.0190
## 15 0.000000 0.0000000 0.0000 0.0000000 0.68916217 0.22975405 0.0190
## 16 0.000000 0.0000000 0.0000 0.0000000 0.00000000 0.68916217 0.9190
##
## [[3]]
##       V3     V4         V5     V6     V7         V8         V9
## 19 0.752 0.0081 0.00619938 0.0043 0.0029 0.00159984 0.00079992
## 20 0.248 0.7439 0.00189981 0.0019 0.0014 0.00139986 0.00079992
## 21 0.000 0.2480 0.49595040 0.0019 0.0019 0.00139986 0.00139986
## 22 0.000 0.0000 0.49595040 0.2480 0.0019 0.00189981 0.00139986
## 23 0.000 0.0000 0.00000000 0.7439 0.2480 0.00189981 0.00189981
## 24 0.000 0.0000 0.00000000 0.0000 0.7439 0.24797520 0.00189981
## 25 0.000 0.0000 0.00000000 0.0000 0.0000 0.74382562 0.99180082
```

Function to extract the dominant eigenvalue (stable growth rate) λ of population projection matrix:

```
lambda <- function(ppm) {(
  eig <- abs(eigen(ppm, only.values = TRUE)$values))
  eig <- eig[which.max(eig)]
  return(eig)
}
```

Stores the transition matrix for hurricane frequency 0.81:

```
MCenv1 <- env.mat[[1]]
```

Retrieves the stationary distribution of the transition matrix as the right eigenvector corresponding to the eigenvalue 1 scaled so that the components of this vector add up to unity:

```
stationary.env1 <- Re(eigen(MCenv1)$vectors)[,1]
stationary.env1 <- stationary.env1/sum(stationary.env1)
stationary.env1
```

```
## [1] 9.438098e-01 5.298664e-02 2.853904e-03 2.953791e-04 4.586169e-05
## [6] 7.111236e-06 1.251176e-06
```

Stores the transition matrix for hurricane frequency 0.081:

```
MCenv2 <- env.mat[[2]]
```

Retrieves the stationary distribution of the transition matrix as the right eigenvector corresponding to the eigenvalue 1 scaled so that the components of this vector add up to unity:

```
stationary.env2 <- Re(eigen(MCenv2)$vectors)[,1]
stationary.env2 <- stationary.env2/sum(stationary.env2)
stationary.env2
```

```
## [1] 0.09041992 0.09370269 0.06006343 0.04928024 0.06004383 0.06799306 0.57849683
```

Stores the transition matrix for hurricane frequency 0.0081

```
MCenv3 <- env.mat[[3]]
```

Retrieves the stationary distribution of the transition matrix as the right eigenvector corresponding to the eigenvalue 1 scaled so that the components of this vector add up to unity:

```
stationary.env3 <- Re(eigen(MCenv3)$vectors)[,1]
stationary.env3 <- stationary.env3/sum(stationary.env3)
stationary.env3
```

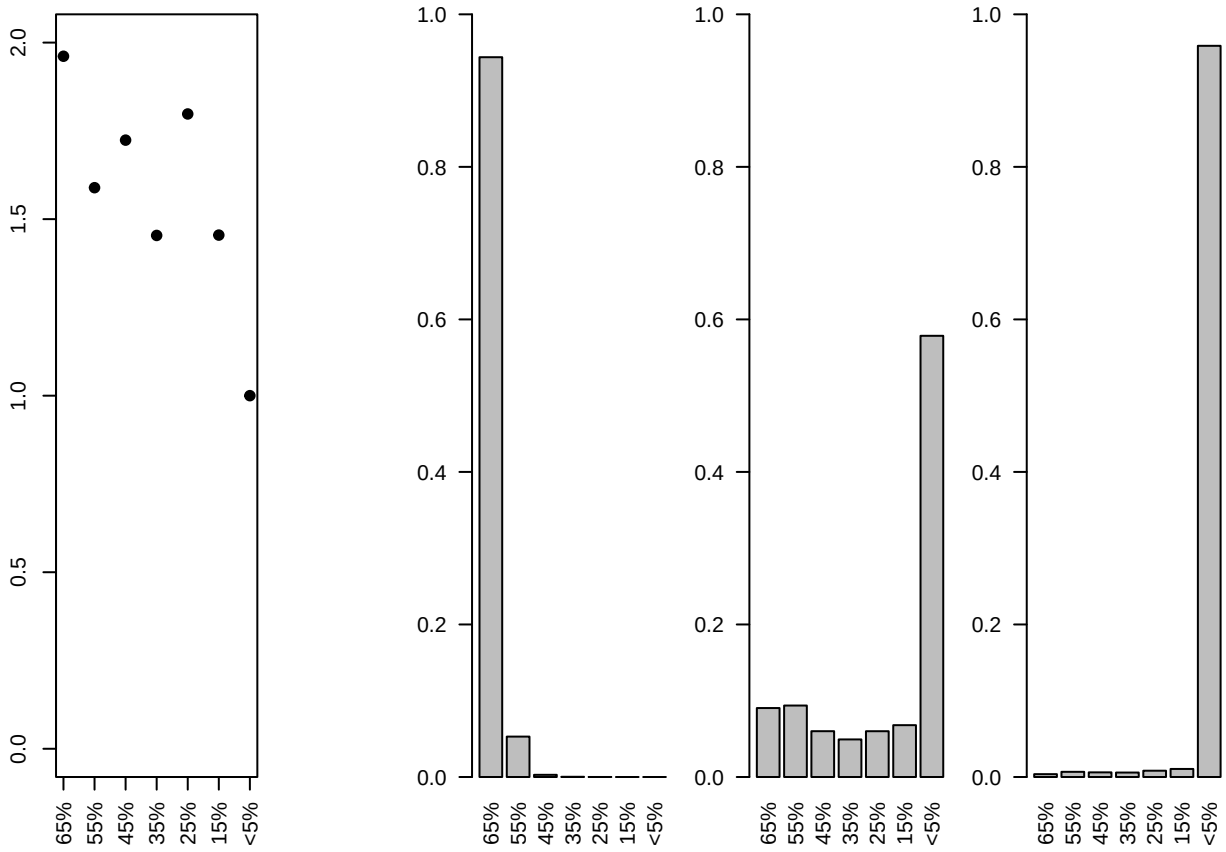
```
## [1] 0.003732607 0.006800455 0.006090960 0.005849224 0.008234984 0.010568000
## [7] 0.958723770
```

Computes constant growth in each environmental state:

```
env.specgrowth <- sapply(mats, function(x) lambda(x))
```

Plots Figure 2

```
# plots stationary environmental distribution for each hurricane frequency
layout(matrix(c(1,0,2,3,4), nrow=1, ncol = 5), widths = c(1,0.5,1,1,1))
par(mar=c(3,2,1,1))
xlabels <- c("65%", "55%", "45%", "35%", "25%",
             "15%", paste("<5%", sep = ""))
plot(env.specgrowth, ylim = c(0,2),
     col = "black", xaxt = "n", pch= 19)
axis(1, at = 1:7, labels = xlabels, las = 2)
barplot(stationary.env1, ylim = c(0,1),
        names.arg = xlabels, las = 2)
barplot(stationary.env2, ylim = c(0,1),
        names.arg = xlabels, las = 2)
barplot(stationary.env3, ylim = c(0,1),
        names.arg = xlabels, las = 2)
```


0.0
0.5
1.0
1.5
2.0
65%
55%
45%
35%
25%
15%
<5%
0.0
0.2
0.4
0.6
0.8
1.0
65%
55%
45%
35%
25%
15%
<5%
0.0
0.2
0.4
0.6
0.8
1.0
65%
55%
45%
35%
25%
15%
<5%
0.0
0.2
0.4
0.6
0.8
1.0
65%
55%
45%
35%
25%
15%
<5%