Population dynamics

The exercises consist of "core" exercises in normal print, and "extras" in italics. The core exercises follow on from each other, so it's best to finish them first and then move onto the extras afterwards, otherwise the code might break. I encourage you as an "extra extra" exercise to try projecting population dynamics of the matrices you created yourself.

Install the latest popdemo development package

```
install.packages("devtools")
devtools::install_github("iainmstott/popdemo/1.2-0/popdemo")
```

Load the popdemo package

```
library(popdemo)
```

Data

For deterministic population projections, we will use a matrix for the desert tortoise *Gopherus agassizzii*, with medium fecundity. The population is found in the Mojave desert, USA. There are 8 stages are based on age and size (carapace length in mm):
Yearling (age 0-1)
Juvenile 1 (<60 mm)
Juvenile 2 (90-99mm)
Immature 1 (100-139mm)
Immature 2 (140-179mm)
Subadult (180-207mm)
Adult 1 (208-239mm)
Adult 2 (>240mm).
See Doak et al. (1994) Ecol. Appl., 4, 446-460.

Load in the data

```
data(Tort); Tort
          Yr    J1    J2    I1    I2    SA    A1   A2
    Yr 0.000 0.000 0.000 0.000 0.000 1.300 1.980 2.57
    J1 0.716 0.567 0.000 0.000 0.000 0.000 0.000 0.00
    J2 0.000 0.149 0.567 0.000 0.000 0.000 0.000 0.00
    I1 0.000 0.000 0.149 0.604 0.000 0.000 0.000 0.00
    I2 0.000 0.000 0.000 0.235 0.560 0.000 0.000 0.00
    SA 0.000 0.000 0.000 0.000 0.225 0.678 0.000 0.00
    A1 0.000 0.000 0.000 0.000 0.000 0.249 0.851 0.00
    A2 0.000 0.000 0.000 0.000 0.000 0.000 0.016 0.86
```

For stochastic population projections, we will use a set of matrices for the polar bear *Ursus maritimus*. The population is found in the southern Beaufort Sea, USA and Canada. There are 6 stages based on age and reproductive status:
Stage-1: 2-year-old
Stage 2: 3-year-old
Stage 3: 4-year-old
Stage 4: adult (5+ years old), available to breed

Stage 5: adult, with cub (0-1 years old)
Stage 6: adult, with yearling (1-2 years old).
See Hunter et al. (2010) Ecology, 91, 2883-2897.

Load in the data

```
data(Pbear); Pbear
    $Y2001
           2YO    3YO    4YO     Ad    AdC    AdY
    2YO 0.0000 0.0000 0.0000 0.0000 0.0000 0.5811
    3YO 0.9858 0.0000 0.0000 0.0000 0.0000 0.0000
    4YO 0.0000 0.9858 0.0000 0.0000 0.0000 0.0000
    Ad  0.0000 0.0000 0.9858 0.5061 0.3791 0.9918
    AdC 0.0000 0.0000 0.0000 0.4857 0.0681 0.0000
    AdY 0.0000 0.0000 0.0000 0.0000 0.5433 0.0000

    $Y2002
           2YO    3YO    4YO     Ad    AdC    AdY
    2YO 0.0000 0.0000 0.0000 0.0000 0.0000 0.5800
    3YO 0.9842 0.0000 0.0000 0.0000 0.0000 0.0000
    4YO 0.0000 0.9842 0.0000 0.0000 0.0000 0.0000
    Ad  0.0000 0.0000 0.9842 0.4563 0.3654 0.9911
    AdC 0.0000 0.0000 0.0000 0.5348 0.0808 0.0000
    AdY 0.0000 0.0000 0.0000 0.0000 0.5435 0.0000

    $Y2003
           2YO    3YO    4YO     Ad    AdC    AdY
    2YO 0.0000 0.0000 0.0000 0.0000 0.0000 0.5379
    3YO 0.9415 0.0000 0.0000 0.0000 0.0000 0.0000
    4YO 0.0000 0.9415 0.0000 0.0000 0.0000 0.0000
    Ad  0.0000 0.0000 0.9415 0.2840 0.3081 0.9662
    AdC 0.0000 0.0000 0.0000 0.6822 0.1384 0.0000
    AdY 0.0000 0.0000 0.0000 0.0000 0.5160 0.0000

    $Y2004
           2YO    3YO    4YO     Ad    AdC    AdY
    2YO 0.0000 0.0000 0.0000 0.0000 0.0000 0.2773
    3YO 0.6578 0.0000 0.0000 0.0000 0.0000 0.0000
    4YO 0.0000 0.6578 0.0000 0.0000 0.0000 0.0000
    Ad  0.0000 0.0000 0.6578 0.5367 0.4243 0.7587
    AdC 0.0000 0.0000 0.0000 0.2220 0.0327 0.0000
    AdY 0.0000 0.0000 0.0000 0.0000 0.2689 0.0000

    $Y2005
           2YO    3YO    4YO     Ad    AdC    AdY
    2YO 0.0000 0.0000 0.0000 0.0000 0.0000 0.3165
    3YO 0.7034 0.0000 0.0000 0.0000 0.0000 0.0000
    4YO 0.0000 0.7034 0.0000 0.0000 0.0000 0.0000
    Ad  0.0000 0.0000 0.7034 0.7225 0.4328 0.7943
    AdC 0.0000 0.0000 0.0000 0.0718 0.0081 0.0000
    AdY 0.0000 0.0000 0.0000 0.0000 0.3254 0.0000
```

The study also gives the population vector.

```
Pbearvec <- c(0.106, 0.068, 0.106, 0.461, 0.151, 0.108)
```

The 'project' function

The core function for understanding population dynamics is the 'project' function. It can be used to understand a number of different types of population dynamics, including deterministic models, stochastic models, long-term dynamics and short-term dynamics.

We'll start with the desert tortoise data. The desert tortoise model is a 'deterministic' model with no density dependence: the vital rates of the model (matrix entries) don't change from timestep to timestep. In the long term, population growth settles to a stable rate and the population structure also becomes stable.

Population dynamics come from multiplying the matrix and the poulation vector. We have the matrix; let's choose a vector at random.

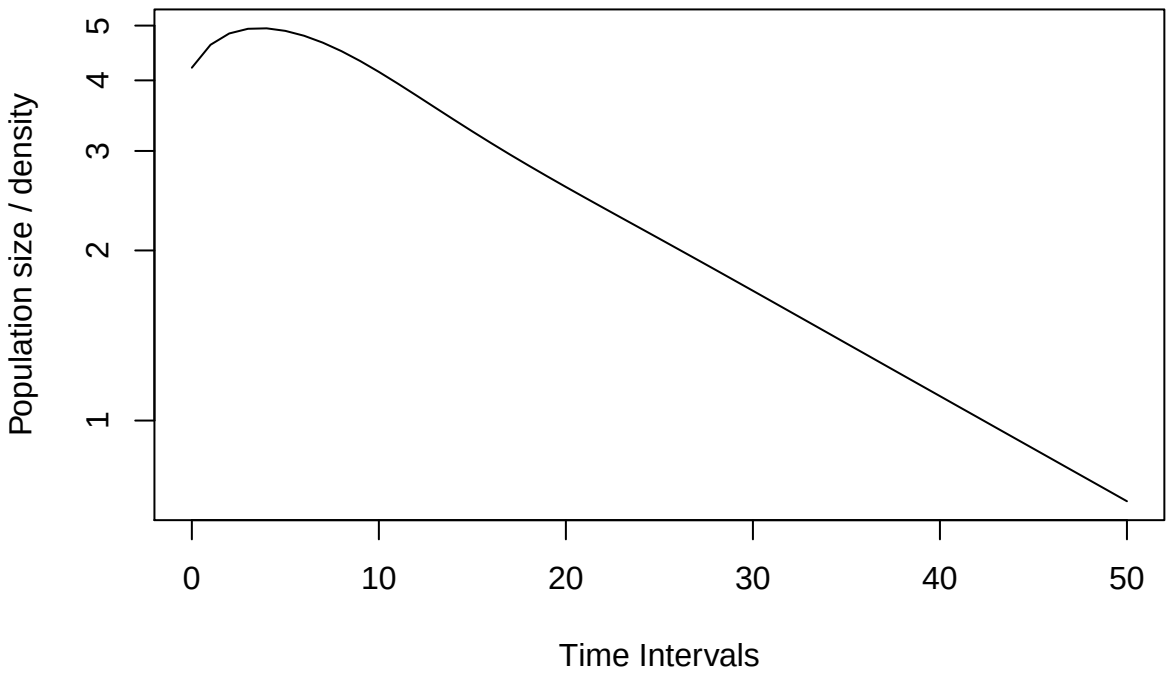
```
Tortvec <- runif(8)
```

Now, we'll project this vector.

```
( Tortp <- project(Tort, Tortvec, time = 50) )
     [1] 4.2116779 4.6246529 4.8426497 4.9352787 4.9444034 4.8938678 4.7974272
     [8] 4.6650176 4.5056631 4.3281328 4.1406141 3.9501893 3.7624708 3.5814829
    [15] 3.4097520 3.2485330 3.0980977 2.9580299 2.8274915 2.7054396 2.5907892
    [22] 2.4825229 2.3797544 2.2817574 2.1879662 2.0979617 2.0114459 1.9282146
    [29] 1.8481303 1.7710981 1.6970469 1.6259150 1.5576406 1.4921564 1.4293870
    [36] 1.3692488 1.3116512 1.2564983 1.2036915 1.1531316 1.1047205 1.0583626
    [43] 1.0139659 0.9714427 0.9307098 0.8916883 0.8543039 0.8184861 0.7841682
    [50] 0.7512871 0.7197825
```

It's easy to plot this projection:

```
plot(Tortp, log = "y")
```



Extras: Try different parameters in the 'project' function: change the amount of time the model is projected, change the population vector, explore some of the extra arguments of the function (e.g. standardising for asymptotic growth). The plot function takes any of the usual graphical parameters: try changing the lines to points, changing their colour, getting rid of the box around the plot, or something similar.

Asymptotic dynamics

Note that in the long term, the population shows a stable rate of growth. This is equal to the dominant eigenvalue of the matrix:

```
eigs(Tort, "lambda")
    [1] 0.9580592
```

The 'eigs' function returns the dominant eigendata (or "eigenstuff") of the matrix. The growth rate is commonly referred to as 'lambda'. In this case, it's slightly below 1 which means the population declines. Equal to 1 means the population neither grows nor declines, and above 1 means the population grows. The rest of the eigenstuff describes other aspects of stable dynamics:

```
eigs(Tort, c("ss", "rv"))
    $ss
    [1] 0.22166176 0.40584601 0.15463401 0.06507518 0.03841807 0.03086514
    [7] 0.07178663 0.01171319

    $rv
    [1] 0.1954968 0.2615887 0.6865549 1.8019036 2.7148109 4.8029132 4.3813423
    [8] 5.1237087
```

"ss" refers to the stable structure: this is the ratio of numbers of individuals in each stage once the population reaches stable growth. "rv" refers to the revproductive value: this is the contribution that each stage makes to stable growth (through survival, growth and reproduction).

Extras: the 'project' function will allow simultaneous projection of multiple population vectors. Create a matrix with several vectors (each column is a vector, so for the desert tortoise model, the matrix of vectors will have 8 rows and as many columns as you like). Project this and plot the projection in the same way as above, replacing the single vector with your multiple vectors. See how each vector does its own thing for the first 20-30 time intervals, after which each settles to stable growth.

Transient dynamics

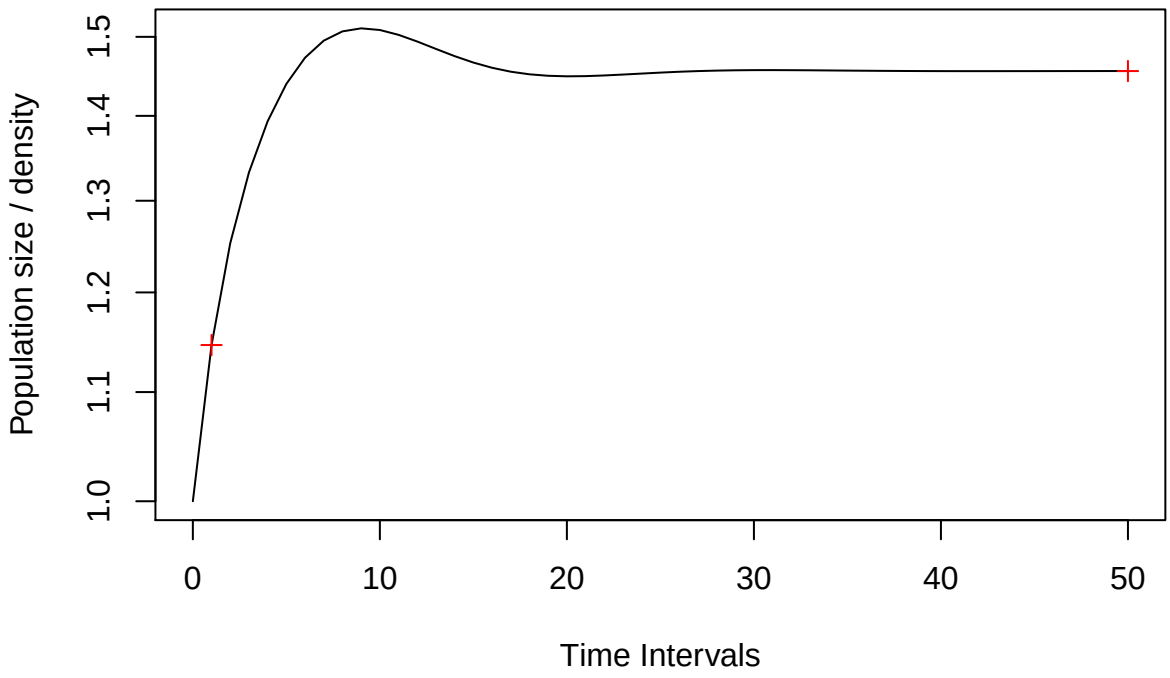
Before settling to stable growth rate, a population will grow, decline or fluctuate in growth at rates faster and/or slower than asymptotic growth. We can see this in our plot of population dynamics of the desert tortoise. These population dynamics are called "transient dynamics" and are a little harder to characterise than asymptotic dynamics as they're so variable and depend on the population structure. For comparative analyses, we often standardise for overall population size (because sizes of different populations in our dataset may vary), and measure transient dynamics relative to asymptotic growth (because different populations, with different matrices, have different growth rates). We can see this in a "standardised" population projection:

```
Tortps <- project(Tort, Tortvec, time = 50,
                  standard.A = TRUE, standard.vec = TRUE)
```

Long-term dynamics of a standardised projection are the same as if lambda equals 1 (because growth at each timestep is effectively divided by lambda).

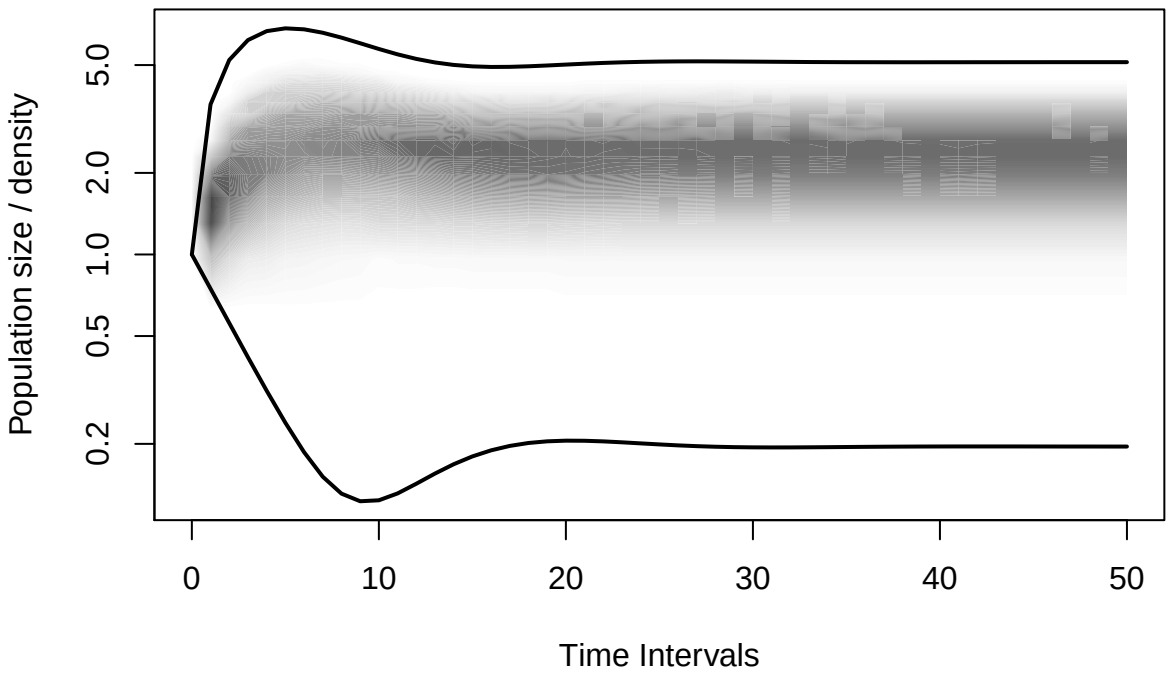
We can measure transient density at any time along the projection (we say 'density' because a standardised dynamic is no longer directly equivalent to size; not the same thing as spatial density!). But in comparative analysis, to make things comparable between populations, we should use comparable timepoints. Two possibilities are at t = 1 and at $t \to \infty$. These indices are called 'reactivity' and 'inertia' respectively:

```
( r1 <- reac(Tort, Tortvec) )
    [1] 1.146124
( ri <- inertia(Tort, Tortvec) )
    [1] 1.455868
plot(Tortps, log = "y")
points(c(1, 50), c(r1, ri), pch = 3, col = "red")
```



Extras: in the previous section, we saw the diversity of different dynamics that can be shown by a model. We can see an overview by drawing thousands of different vectors at random from a dirichlet distribution, and plotting a shade map of the pobability of a certain population size at a certain point in time:

```
Tortpd <- project(Tort, "diri", time = 50,
                  standard.A = TRUE)
plot(Tortpd, plottype = "shady", bounds = T, log = "y")
```



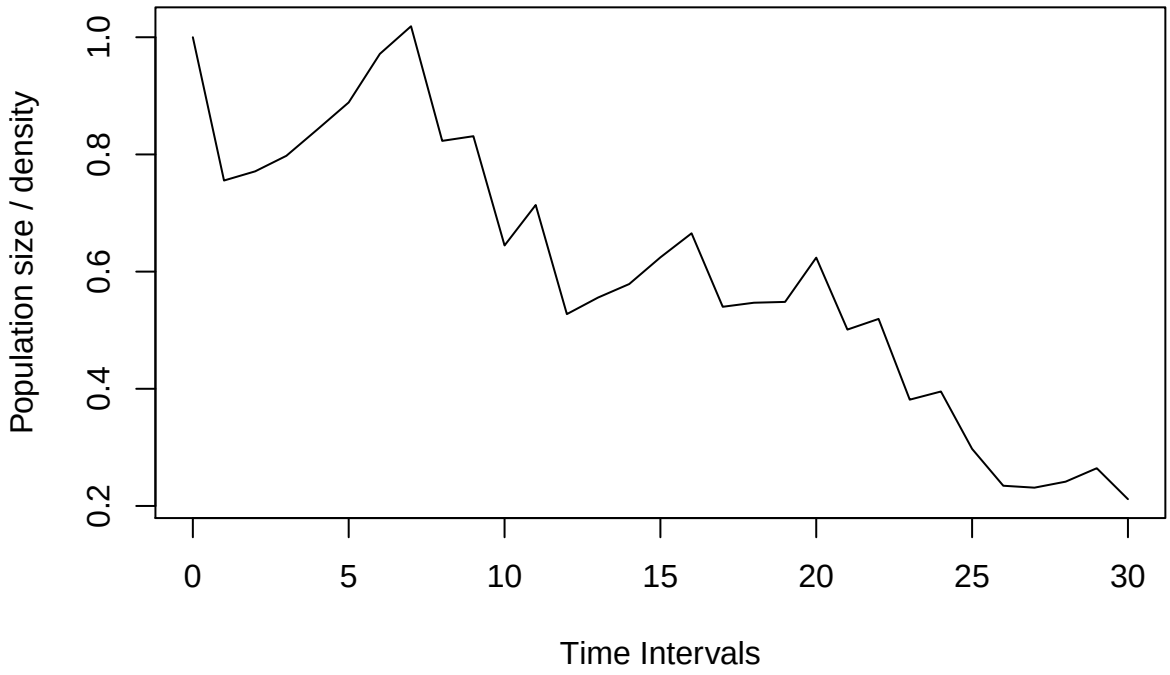
The thick lines show the "bounds" on population density, which are the region within which all population dynamics must stay.

Stochastic dynamics

It's a big assumption that over decades, the vital rates of a population won't change. One reason that they may change is because of environmental stochasticity, where environmental conditions, which change from year to year, impact survival, reproduction and growth.

In the polar bear data, there are 5 matrices from 2001-2005. We can project dynamics using these matrices in the same way as before, just passing the list of matrices rather than a single matrix. The function knows how to do the rest:

```
Pbearp <- project(Pbear, Pbearvec, time = 30)
plot(Pbearp)
```



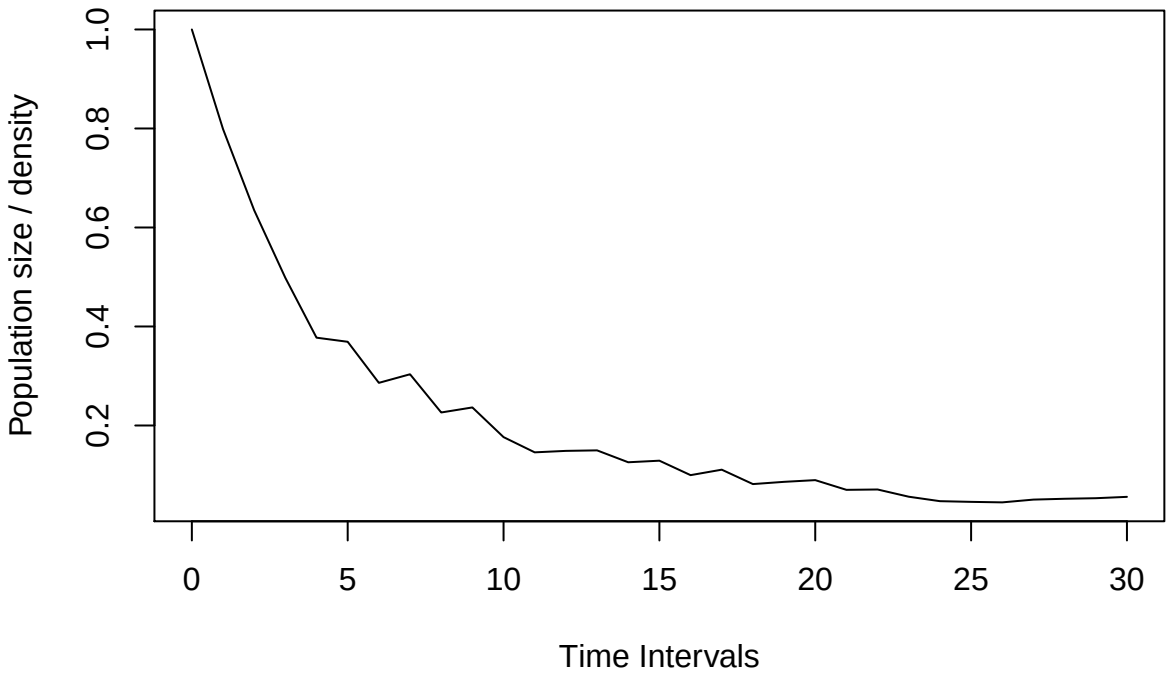
By default, the projection selects each matrix with an equal probability at each time interval. However, there are situations for some systems where we might want to vary the probability that a matrix is chosen (e.g. cyclical or nonrandom patterns in environmental conditions, behaviour, food availability). In the polar bear data, years '04 nad '05 were "bad" years. If bad years occur with probability q, then we can construct a transition matrix (similar to a population matrix) which describes transitions between years:

```
q <- 0.5
( PbearM <- matrix(rep(c((1-q)/3, (1-q)/3, (1-q)/3, q/2, q/2), 5), 5, 5) )
             [,1]      [,2]      [,3]      [,4]      [,5]
[1,] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
[2,] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
[3,] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
```

```
     [4,] 0.2500000 0.2500000 0.2500000 0.2500000 0.2500000
     [5,] 0.2500000 0.2500000 0.2500000 0.2500000 0.2500000
```

We can pass this to the "Aseq" argument of the function to get a new estimate of population dynamics:

```
Pbearp2 <- project(Pbear, Pbearvec, Aseq = PbearM, time = 30)
plot(Pbearp2)
```



We can calculate long-term growth of a stochastic model by averaging per-timestep growth rates over a long projection time. Because stochastic models also have random per-timestep growth, we can calculate its variance too:

```
stoch(Pbear, c("lambda", "var"), Aseq = PbearM)
```

```
        lambda        var
   1 0.9387041 0.01642502
```

Extras: the sequence of matrices can affect what the growth of a stochastic model is. What happens if you increase the likelihood of a bad year in the example above? What about directionality of transition probability: it's possible, for example, to make moving into a bad year cycle from a good year cycle more likely than moving into a good year cycle from a bad year cycle.