HW6 - Do Not Post

1) $n_1(t) = 0.95 n_0(t)$

$n_2(t) = 0.97 n_1(t)$

$n_3(t) = 0.9 n_2(t)$

$n_1(t-1) = 0.95 n_0(t-2)$

$n_2(t-2)\ 0.97 \cdot 0.95 n_0(t-3)$

$n_3(t-3) = 0.9 \cdot 0.97 \cdot 0.95 n_0(t-4)$

Fertility

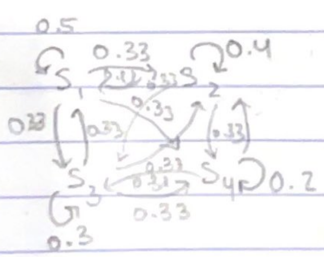
$n_0(t) = 0.1 n_0(t-1) + 1.2 n_1(t) + 0.9 n_2(t) + 0.1 n_3(t)$

$n_0(t) = 0.1 n_0(t-1) + 1.2(0.95) n_0(t-2) + 0.9\ (0.95)(0.97) n_0(t-3)$
$+ 0.1(0.9)(0.95)(0.97) n_0(t-4)$

$n_0(t) = 0.1 n_0(t-1) + 1.14 n_0(t-2) + 0.82935 n_0(t-3) + 0.082935 n_0(t-4)$

REC := [0.1, 1.14, 0.82935, 0.089235]

3) A



	S_1	S_2	S_3	S_4
S_1	0.5	0.33	0.33	0.33
S_2	0.33	0.4	0.33	0.33
S_3	0.33	0.33	0.3	0.33
S_4	0.33	0.33	0.33	0.2

B The page ranks are as follows: $S_1 > S_3 > S_2 > S_4$

most popular ... least popular

```
> #Do not Post
  #Nikita John, September 27th, 2021, Assigment 6
> #Maple code for Lecture 5

  #DMB:="Dynamical Models in Biology" by S.P. Ellner and J. Gukenheimer

  with(LinearAlgebra) :

  Help5 :=proc( ) : print( `RecToSeq(INI,REC,N), GrowthC(INI,REC,K) , GrowthCe(REC)`) :
  print( ` LeslieMod(SUR,FER): e.g. LeslieMod([9/10,9/10],[0,1,1]);`) :
  print( ` LeslieMat(SUR,FER); e.g. LeslieMat([9/10,9/10],[0,1,1]);`) :
  end:




      #RecToSeq(INI,REC,N): Inputs two lists of numbers, INI and REC (of the same length, let's
      call it k) and a positive integer N larger than their length

      #outputs the list of the first N members of the sequence satisfying the linear recurrence with
      constants coefficients or order k
  #f(n)=REC[1]*f(n-1)+... +REC[k]*f(n-k)

  RecToSeq :=proc(INI, REC, N ) local i, k, L, newguy :
  if not (type(INI, list) and type(REC, list) and nops(INI) = nops(REC) and type(N, integer)
      and N ≥ nops(INI) ) then
   print( `bad innput`) :
   RETURN(FAIL) :
  fi:

  k := nops(INI) :

  L := INI :

  while nops(L) < N do
   newguy := add(REC[i] * L[ -i], i = 1 ..k) :
   L := [op(L), newguy] :
  od:
  L :
  end:




      #GrowthC(INI,REC,K): The estimate of the growth constant of the sequence given by the
      inital conditions and recurrence pair [INI,REC] using K terms
  GrowthC :=proc(INI, REC, K) local L, a, b :
  L := RecToSeq(INI, REC, K) :

  a := L[ -1 ]/L[ -2 ] :
  b := L[ -2 ]/L[ -3 ] :
```

```
if abs(a-b) < 1/10^(Digits + 3) then
RETURN(evalf(a)) :
else
print(` make `, K, `bigger `) :
 RETURN(FAIL) :
fi:
end:




     #GrowthCe(REC): The  EXACT  growth constant of the recurrence REC using the
     characteristic equation
GrowthCe :=proc(REC) local x, i :
evalf([solve(1-add(REC[i]/x^i, i=1 ..nops(REC)))])[1] :

end:


#LeslieMod(SUR,FER):In a population with A age-groups and survival vector
#(following the notation in the book DMB, p.33

     #SUR=[p[0], ...p[A-1]] where p[i] is the probability of somebody of age i will still be alive
     the next year and

     #FER=[f[0],..., f[A]] where f[i] is the fertility factor (of course f[0]=0 in real life, and also in
     real life the
#younger ages can't have babies, so SUR has A components and FER has A+1 components
#Outputs the recurrence vector REC that enables the computation of the population growth. TRY:
#LelieMod([9/10,9/10],[0,1/2,1]]);
LeslieMod :=proc(SUR, FER) local i, L, A :

 if not (type(SUR, list) and type(FER, list) and nops(SUR) + 1 = nops(FER)) then
  print(`bad input`) :
 RETURN(FAIL) :
 fi:

A := nops(SUR) :

L[0] := 1 :

for i from 1 to A do
 L[i] := L[i-1]*SUR[i] :
od:

[seq(FER[i + 1]*L[i], i=0 ..A)] :

end:
```

```
#LeslieMat(SUR,FER):In a population with A age-groups and survival vector
#(following the notation in the book DMB, p.33

    #SUR=[p[0], ...p[A-1]] where p[i] is the probability of somebody of age i will still be alive
    the next year and

    #FER=[f[0],..., f[A]] where f[i] is the fertility factor (of course f[0]=0 in real life, and also in
    real life the
#younger ages can't have babies, so SUR has A components and FER has A+1 components
#Outputs the Leslie A+1 by A+1 Leslie Matrix (DMB, p. 36, Eq. (2.17))
#LelieMat([9/10,9/10],[0,1/2,1]]);
LeslieMat := proc(SUR, FER) local i, A :

 if not (type(SUR, list) and type(FER, list) and nops(SUR) + 1 = nops(FER)) then
  print(`bad input`) :
 RETURN(FAIL) :
 fi:

A := nops(SUR) :

matrix([ FER, seq([0$(i-1), SUR[i], 0$(A + 1 - i)], i = 1 ..A)]) :
end:
```

```
> #1: Calculating Growth Constant
GrowthCe([0.1, 1.14, 0.82935, 0.089235]);
```

$$1.386767237 \tag{1}$$

```
> #2: The largest eigenvalue does match the growth constant found using GrowthCe!!
 with(LinearAlgebra) :
SUR := [0.95, 0.97, 0.9];
FER := [0.1, 1.2, 0.9, 0.1];
 A := LeslieMat(SUR, FER);
```

$$SUR := [0.95, 0.97, 0.9]$$

$$FER := [0.1, 1.2, 0.9, 0.1]$$

$$A := \begin{bmatrix} 0.1 & 1.2 & 0.9 & 0.1 \\ 0.95 & 0 & 0 & 0 \\ 0 & 0.97 & 0 & 0 \\ 0 & 0 & 0.9 & 0 \end{bmatrix} \tag{2}$$

```
>  Eigenvalues(%);
```

$$\begin{bmatrix} 1.38573262885364 + 0.\ I \\ -0.583351516086360 + 0.403125877485025\ I \\ -0.583351516086360 - 0.403125877485025\ I \\ -0.119029596680917 + 0.\ I \end{bmatrix} \tag{3}$$

> #3: Create transition matrix and evaluate P^{1000}

```
 P := Matrix([[0.5, 0.33, 0.33, 0.33], [0.33, 0.4, 0.33, 0.33], [0.33, 0.33, 0.3, 0.33], [0.33, 0.33,
```

0.33, 0.2]]);

$$P := \begin{bmatrix} 0.5 & 0.33 & 0.33 & 0.33 \\ 0.33 & 0.4 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.3 & 0.33 \\ 0.33 & 0.33 & 0.33 & 0.2 \end{bmatrix} \quad \textbf{(4)}$$

> $evalm(P^{1000});$

[[$4.44537264177654\ 10^{129}$, $4.09792732994178\ 10^{129}$, $3.80085666028387\ 10^{129}$, $3.54394572802698\ 10^{129}$], **(5)**
[$4.09792732994178\ 10^{129}$, $3.77763795180344\ 10^{129}$, $3.50378599062595\ 10^{129}$, $3.26695492706958\ 10^{129}$],
[$3.80085666028387\ 10^{129}$, $3.50378599062595\ 10^{129}$, $3.24978635452501\ 10^{129}$, $3.03012386351322\ 10^{129}$],
[$3.54394572802698\ 10^{129}$, $3.26695492706958\ 10^{129}$, $3.03012386351322\ 10^{129}$, $2.82530899775849\ 10^{129}$]]

>