Before any math is involved I will load the MATH280 package for the necessary functions

```lisp
(%i1) load("C:/Users/Rj/Documents/Differential Equations/MATH280.mac")S
```

Figure 1:

A network model for the short-term prediction of the evolution of cocaine consumption in Spain

Francisco-José Castaño³, Ivoa-C. Lloboda#, María Hulie³, Emilio Sánchez³, Javier Villanueva³

*Departamento de Biostatística e Investigación Operativa, Universidad de Valladolid
#Instituto de Matemáticas, Universidad Autónoma de Madrid
#Centro de Estudio Aplicación, SIR, Avenida Universidad de Valladolid

Abstract

Cocaine consumption is a social problem with acute consequences and its dependency can be regarded as a health concern of major importance. This fact leads us to develop the idea that its transmission dynamics can be studied using mathematical models. Under this point of view, in this paper we propose a network model to study the short-term evolution of the cocaine consumer population. The model parameters are obtained from data source and from an analogous continuous model. Sensitivity of the model parameters is studied. The results are associated with prevention and treatment policies and the sensitivity study gives us information about which parameters have more incidence on the future evolution of cocaine. Results and discussions are also presented.

This network model was constructed using the following non-linear system of equations

Figure 2:

\[
\begin{align*}
N(t) &= \mu P(t) - dN(t) - \rho N(t)(C_0(t) + C_r(t) + C_h(t)) + \gamma C_0(t), \\
C_0(t) &= \rho N(t)C_0(t) + C_r(t) + C_h(t) - dC_0(t) - \gamma C_0(t), \\
C_r(t) &= \gamma C_0(t) - dC_r(t) - \sigma C_r(t), \\
C_h(t) &= \sigma C_r(t) - dC_h(t) - \epsilon C_h(t), \\
P(t) &= N(t) + C_0(t) + C_r(t) + C_h(t).
\end{align*}
\]

Where \(N(t)\) represents the function of the non-consumer population in Spain with respect to time. 
\(C_0\) represents the time function of the occasional consumer population: who are defined to have consumed cocaine sometimes. 
\(C_r\) represents the time function of the regular consumer population: who are defined to have consumed cocaine in the last year. 
\(C_h\) represents the time function of the habitual consumer population: who are defined to have consumed cocaine in the last month.

Furthermore, most of the constants used to model the dynamic of cocaine were derived from research (cited in the paper) related to SIR, despite the network model approach.
Figure 3: Defined Coefficients

- $\mu = 0.01 \text{ years}^{-1}$, is the average Spanish birth rate between years 1995-2007 [18].
- $d = 0.008388 \text{ years}^{-1}$ is the average Spanish death rate between years 1995-2007 [18].
- $d_1 = 0.01636 \text{ years}^{-1}$ is the augmented death rate due to drug consumption. In Spain, approximately 6.8% of mortality is due to drug consumption [17].

Figure 4:

- $\epsilon = 0.0000456 \text{ years}^{-1}$. From official data [17] 4.25% of habitual consumers begin a therapy program every year. Furthermore, using data from Table 1 corresponding to National Drug Observatory Reports [17], the average proportion of population with habitual consumption is 0.937. Moreover, the conclusion specified in [17, 19, 20] that a habitual consumer takes about nine years before going to therapy. Therefore, the percentage of habitual consumers in therapy per year is 0.00439%. To be precise, $0.0093 \times 0.0425 \times 1/9 = 0.0000439$. Additionally, [20, 21, 22, 23, 24, 25, 26] they conclude that around 52% of the individuals on therapy recover with an average of six months. Then, we obtained $\epsilon = 0.0000439 \times 0.52 \times 1/0.5 = 0.0000456$, i.e.,

$$
\epsilon = \epsilon_1 \times \epsilon_2 \times \epsilon_3 \times \epsilon_4 \times \epsilon_5 = 0.0093 \times 0.0425 \times 1/9 \times 0.52 \times 1/0.5 = 0.0000456. \quad (6)
$$

- $\beta = 0.00614$, transmission rate due to social pressure to consume cocaine.
- $\gamma = 0.0596$, rate at which an occasional consumer transits to the regular consumption subpopulation.
- $\sigma = 0.0579$, rate at which a regular consumer transits to the habitual consumption subpopulation.
- The initial conditions of the model are taken for year 1995 ($t = 0$), $N(t = 0) = 0.944$, $C_2(t = 0) = 0.034$, $C_4(t = 0) = 0.018$ and $C_5(t = 0) = 0.004$.

Figure 5:

![Diagram of the model](image-url)
Figure 6: In order to simulate new individuals entering the population, the authors include:

The evolution rules are as follows:

- The new 15-years-old individuals enter in the system as non-consumers following a Poisson distribution with mean \( \lambda = \mu N / 12 = 0.04 N \) people/month. Poisson distribution has been chosen because it is the natural discretization of the underlying exponential distribution in the continuous model which mean is \( \mu \). Thus, for every time step, we compute a pseudorandom number \( 0 \leq s \leq 1 \) and we find the minimum natural number \( i \) such that

\[
\sum_{k=0}^{i} e^{-\lambda} \frac{\lambda^k}{k!} > s.
\]

Based on this "evolution rule",

\[
\text{ss:}\sum \left( e^{-\mu N} (\mu N)^k / k! \right)
\]

\[
3.8330934847373110^{-4} \sum_{k=0}^{\infty} \frac{7.866666666666667^k}{k!}
\]

Unfortunately, in maxima it is difficult to solve for "i" in this situation. However, there are other ways to model this even if they are crude.

Here, I ran the sum for various values of \( i \) until the sum \( ss=1 \).

At this point, \( ss > s \) must be true and we can add the new individuals to population \( N \)

\[
(\text{sstest0}) \text{ss:}\sum \left( e^{-\mu N} (\mu N)^k / k! \right)
\]

\[
0.003398681422313375
\]

\[
0.01525914260634632
\]

\[
0.04635990748892159
\]

\[
0.107524745913196
\]

\[
0.2037574229190925
\]

\[
0.3299291560710615
\]

\[
0.471721514227981
\]

\[
0.611151930185339
\]

\[
0.733023884955599
\]

\[
0.828896483748004
\]

\[
0.8974599276867421
\]

\[
0.9424070705801261
\]

\[
0.9696058544848405
\]
Using these, I estimated that the original authors had new individuals enter the system at an average rate of $0.000075^\times t$ so in rhsN (the derivative of N) this looks like $+0.000075$

\[
\text{Let's solve using the numerical solver rkf45. Then graph the populations vs time.}
\]

\[
\text{Figure 7: Data Table 1}
\]

<table>
<thead>
<tr>
<th>Percentages/years</th>
<th>1995</th>
<th>1997</th>
<th>1999</th>
<th>2001</th>
<th>2003</th>
<th>2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>% Non-consumer</td>
<td>0.944</td>
<td>0.948</td>
<td>0.948</td>
<td>0.911</td>
<td>0.903</td>
<td>0.884</td>
</tr>
<tr>
<td>% Occasional consumers</td>
<td>0.034</td>
<td>0.032</td>
<td>0.031</td>
<td>0.028</td>
<td>0.029</td>
<td>0.017</td>
</tr>
<tr>
<td>% Regular consumers</td>
<td>0.015</td>
<td>0.015</td>
<td>0.015</td>
<td>0.015</td>
<td>0.015</td>
<td>0.015</td>
</tr>
<tr>
<td>% Habitual consumers</td>
<td>0.004</td>
<td>0.005</td>
<td>0.006</td>
<td>0.014</td>
<td>0.011</td>
<td>0.018</td>
</tr>
</tbody>
</table>

\[
\text{Lastly, the initial conditions are found in data table 1.}
\]

\[
t=0 \text{ corresponds to December 1995 with the given values.}
\]

\[
\text{Let's solve using the numerical solver rkf45. Then graph the populations vs time.}
\]

\[
\text{ (%i928) \hspace{1em} kill(N) S}
\]

\[
\text{ (%i929) \hspace{1em} sol:=rkf45(["rhsN","rhsCo","rhsCr","rhsCb"]\{N,Co,Cr,Cb\},}
\]

\[
\text{[.9440,0.0340,0.0180,0.0004],[0,216],}
\]

\[
\text{report=true,}
\]

\[
\text{absolute_tolerance=1e-9,}
\]

\[
\text{max_iterations=20000,}
\]

\[
\text{S}
\]

\[
\text{Info : rkf45 :}
\]

\[
\text{Integration points selected : 10}
\]

\[
\text{Total number of iterations : 9}
\]

\[
\text{Bad steps corrected : 0}
\]

\[
\text{Minimum estimated error : 1.31031291201644410^{-14}}
\]

\[
\text{Maximum estimated error : 5.3193467018856210^{-10}}
\]

\[
\text{Minimum integration step taken : 2.16}
\]

\[
\text{Maximum integration step taken : 30.62385568522025}
\]

\[
\text{ (%i924) \hspace{1em} wxdraw2d(point_type=-1,points_joined=true,}
\]

\[
\text{xlabel="Time (months)", ylabel="Percentage of Total Population", color=blue,}
\]

\[
\text{key="N", points(makelist([1],p[1],[p,sol]),color=green,}
\]

\[
\text{key="Co",points(makelist([1],p[2],[p,sol]),color=red,}
\]

\[
\text{key="Cr",points(makelist([1],p[3],[p,sol]),color=blue,}
\]

\[
\text{key="Cb",points(makelist([1],p[4],[p,sol])));}
\]

\[
\]
Our results match those published in the paper.

Figure 8:

In addition to solving the system, the authors introduce a sensitivity analysis to simulate the randomness present in human populations. They do this by varying the given parameters using a Latin Hypercube Sampling algorithm.

Figure 9:

In order to perform the sensitivity analysis, let us use the technique called Latin Hypercube Sampling (LHS) to vary parameter values in the proposed model. Latin Hypercube Sampling, a type of stratified Monte Carlo sampling, is a sophisticated and efficient method for achieving equitable sampling of all input parameters simultaneously [28, 29]. Each parameter of the model can be defined as having an appropriate probability density function associated with it. It is usual to use the uniform distribution centered at deterministic parameters estimators in absence of data to inform on the distribution for a given parameter [30, 31]. Then, the model can be simulated by sampling a single value from each parameter distribution. Many samples should be taken and many simulations should be run, producing varying output values.

To vary $c_2$, $c_3$, $c_4$, $c_5$ and $\beta$, we assume that all of them follow a uniform probability density with support on the intervals $[0,0.0882]$, $[0.06,0.2]$, $[0.32,0.72]$, $[0.4]$ and $[0,0.19]$ respectively. The intervals for $c_2$ and $\beta$ are chosen.

Here are the initial parameter values along with the previously defined $\beta=0.09614$.

Figure 10:

$$
\epsilon = \epsilon_1 \times \epsilon_2 \times \epsilon_3 \times \epsilon_4 \times \epsilon_5 = 0.0093 \times 0.0425 \times 1/9 \times 0.52 \times 1/0.5 = 0.0000456.
$$

Here, I have attempted to create this algorithm to select unique variants of the parameters. The code is identical to the famous computer science "N queens" problem; The objective of "N queens" problem is to create a recursive algorithm which will determine where to place N queens on an NxN chessboard so that no two queens will threaten each other- that is no queens share a row, column, or diagonal. This translates to the LHS (Latin Hypercube Sampling) in the sense that the random numbers generated will not share any coordinate values thus generating a unique set. This will yield a better sample than random numbers generated in the same bounds.
In my explanations, I will continually refer to the N queens problem since it is simpler to conceptualize. I will average the outputs to give me a new E2,... E5. Then I will evaluate the equation in figure 10. I will create 5 sets of these new variables so they can be introduced at different time intervals on the graph to simulate the authors' "realizations".

In my explanations, I will continually refer to the N queens problem since it is simpler to conceptualize.

```scheme
(%i1433) /* the 5x5 array represents a 5x5 chessboard*/
   array(A,5,5)$
   block(
      for i:1 thru 5 do(
         for j:1 thru 5 do(
            /* 0.085 here will represent the variance of the E1 parameter*/
            A[i,j]:random(0.085),
            A[2,i,j]:random(0.085),
            A[3,i,j]:random(0.085),
            A[4,i,j]:random(0.085),
            A[5,i,j]:random(0.085)
          )));
/* next the code will set each position on the chessboard to false, since it is not yet determined to be unique for which it will mark the position as true (essentially placing a queen there)*/
solved:false$
   c:makeList(false,i,1,5,1)$
   r:makeList(false,i,1,5,1)$
   d1:makeList(false,i,1,9,1)$
   d2:makeList(false,i,1,9,1)$
   t:makeList(0,i,1,5,1)$
   f:makeList(0,i,1,5,1)$
   place(x,A):=block(
      for i:1 thru 5 step 1 do(
         if not c[i] and not d1[(x-i)+5] and not d2[x+i-1] and not r[x] then(
            /* here the code evaluates whether the ith column, ith row, and the ith diagonal are free or non threatening
            if so, the algorithm will "place a queen" at that value by marking it true, essentially verifying that the value is unique*/
            c[i]:true,
            d1[(x-i)+5]:true,
            d2[x+i-1]:true,
            r[x]:true,
            t[x]:A[x,i],
            if x=5 and not solved then (/* since solved was previously set to false, the code will read not solved as true,
            then in the next few lines will output the unique set of values (positions of queens) */
               for j:1 thru 5 step 1 do(
                  final[j]:t[j],
                  solved:true)
            /* the recursive property comes into play next, since if the algorithm gets to the 5th column and
            it is unable to find a coordinate value in the row for which the value is unique (the queen is not threatened),
            it will return to the previous row and reevaluate for a new free coordinate */
               else if not x=5 then
               place(x+1,A),
               c[i]:false,
               d1[(x-i)+5]:false,
               d2[x+i-1]:false,
               r[x]:false,
               t[x]:false)$
   place(1,A),
   for i:1 thru 5 step 1 do(
      f[i]:final[i],
```

```
done.03178229806689240.07247913619910840.028213020939530210.0033178200088482980.08475821742753331 (%}\{1419)
```
Again for E3

```lisp
(%i1518) array(A,5,5)$
block(
  for i:1 thru 5 do(
    for j:1 thru 5 do
      A[i,j]:random(0.2),
    A[3,j]:random(0.2),
    A[4,j]:random(0.2),
    A[5,j]:random(0.2)
  ),
  solved:false$
$c:makelist(false,i,1,5,1)$
$r:makelist(false,i,1,5,1)$
$d1:makelist(false,i,1,9,1)$
$d2:makelist(false,i,1,9,1)$
$t:makelist(0,i,1,5,1)$
$f:makelist(0,i,1,5,1)$
place(x,A):=block(
  for i:1 thru 5 step 1 do(
    if not c[i] and not d1[(x-i)+5] and not d2[(x+i)-1] and not r[x] then(
      c[i]:true,
      d1[(x-i)+5]:true,
      d2[(x+i)-1]:true,
      r[x]:true,
      t[x]:A[x,i],
    )
    if x=5 and not solved then (  
      for j:1 thru 5 step 1 do(
        final[j]:t[j],
        solved:true
      )
    )
  else if not x=5 then  
    place(x+1,A),
    c[i]:false,
    d1[(x-i)+5]:false,
    d2[(x+i)-1]:false,
    r[x]:false,
    t[x]:false$
  )$
  place(1,A),
  for i:1 thru 5 step 1 do(
    t[i]:final[i],
    print(t[i])$
  )$

done$0.16061597462184570.10880191663566970.039660185133479730.053617736113923710.168598844165845$

(\%o1504)

done$\%o1513$

(\%i1451) E21avg:mean([E21,E22,E23,E24,E25])$
(\%i1468) E22avg:mean([E21,E22,E23,E24,E25])$
(\%i1485) E23avg:mean([E21,E22,E23,E24,E25])$
(\%i1502) E24avg:mean([E21,E22,E23,E24,E25])$
(\%i1519) E25avg:mean([E21,E22,E23,E24,E25])$

E4

(%i1603) array(A,5,5)$
block(
  for i:1 thru 5 do(
    for j:1 thru 5 do
      A[i,j]:random(0.65),
    A[3,j]:random(0.65),
    A[4,j]:random(0.65),
    A[5,j]:random(0.65)
  )
)`
(\%{}o1589)

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%

\%
Next, I will evaluate the equation in figure 10 for each of these determined values. These will become the new epsilon parameters for each realization.

Again for beta
WARNING: To obtain these results took numerous simulations.

parameters for the time interval 216 months and graphing.

Now solving the system of equations with the new variant
parameters for the time interval 216 months and graphing.

Since my LHS used far less simulations and randomizations than the author's the results are not perfect.
Fortunately, the reproduced graphs do resemble the graphs in the paper.

%newbeta1:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta2:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta3:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta4:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta5:mean([beta1,beta2,beta3,beta4,beta5])/128

%newbeta1:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta2:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta3:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta4:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta5:mean([beta1,beta2,beta3,beta4,beta5])/128

Now solving the system of equations with the new variant
parameters for the time interval 216 months and graphing.

Since my LHS used far less simulations and randomizations than the author's the results are not perfect.
Fortunately, the reproduced graphs do resemble the graphs in the paper.

%newbeta1:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta2:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta3:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta4:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta5:mean([beta1,beta2,beta3,beta4,beta5])/128

Now solving the system of equations with the new variant
parameters for the time interval 216 months and graphing.

Since my LHS used far less simulations and randomizations than the author's the results are not perfect.
Fortunately, the reproduced graphs do resemble the graphs in the paper.

%newbeta1:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta2:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta3:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta4:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta5:mean([beta1,beta2,beta3,beta4,beta5])/128

Now solving the system of equations with the new variant
parameters for the time interval 216 months and graphing.

Since my LHS used far less simulations and randomizations than the author's the results are not perfect.
Fortunately, the reproduced graphs do resemble the graphs in the paper.

%newbeta1:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta2:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta3:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta4:mean([beta1,beta2,beta3,beta4,beta5])/128
%newbeta5:mean([beta1,beta2,beta3,beta4,beta5])/128

Now solving the system of equations with the new variant
parameters for the time interval 216 months and graphing.

Since my LHS used far less simulations and randomizations than the author's the results are not perfect.
Fortunately, the reproduced graphs do resemble the graphs in the paper.
% Given Equations with newvar4

\[
\begin{align*}
\text{rhsN4:} & \quad \mu(N+C_r+C_b) - N \cdot \text{newbeta}4 \cdot (N \cdot (C_r+C_b)) (N+C_r+C_b) + \text{newepsilon}4 \cdot C_b + 0.000075S \\
\text{rhsC04:} & \quad \text{newbeta}4 \cdot (N \cdot (C_r+C_b)) (N+C_r+C_b) - d \cdot C_r \cdot \text{gamma} \\
\text{rhsCr4:} & \quad \text{gamma} \cdot C_r - d \cdot C_r \cdot \text{sigma} \cdot C_b \\
\text{rhsCb4:} & \quad \text{sigma} \cdot C_r - d \cdot C_b \cdot \text{newepsilon}4 \cdot C_b \\
\end{align*}
\]

\text{sol4:} \text{rk45}([[\text{rhsN4, rhsC04, rhsCr4, rhsCb4, N, Cr, Cb}], \\
\text{rest(last(sol3)), i=126, 168}, \\
\text{report=false,} \\
\text{absolute_tolerance=1e-9,} \\
\text{max_iterations=20000}] )$

\text{sol5:} \text{rk45}([[\text{rhsN5, rhsC05, rhsCr5, rhsCb5, N, Cr, Cb}], \\
\text{rest(last(sol4)), i=1, 216}, \\
\text{report=false,} \\
\text{absolute_tolerance=1e-9,} \\
\text{max_iterations=20000}]] )$

\text{done}
Clearly, this graph resembles the provided graphs. Unfortunately, without the proper number of realizations and computations of the LHS the data does suffer. However, the code written above does prove that it is a reproduction of the original that could be improved with more of the same computations.

Figure 11: