Log-Linear model and Multistate Model to Assess the Rate of Fibrosis in NAFLD Patients

Iman M. Attia (✉ imanattiathesis1972@gmail.com)

Institute of Statistical Studies and Research, Cairo University <https://orcid.org/0000-0002-7333-9713>

Research Article

Keywords: log-linear model, Multistate model, Non-Alcoholic Fatty Liver Disease, Poisson regression, Continuous time Markov chains, longitudinal studies, Maximum Likelihood estimation, Mean Sojourn Time, HOMA2-IR.

Posted Date: March 14th, 2022

DOI: <https://doi.org/10.21203/rs.3.rs-1416828/v2>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Log-Linear model and Multistate Model to Assess the Rate of Fibrosis in NAFLD Patients

Iman M. Attia *, ORCID , 0000-0002-7333-9713

**Department of Mathematical Statistics, Faculty of Graduate Studies for Statistical Research, Cairo University, Egypt*

Corresponding author : Iman M. Attia (imanattiathesis1972@gmail.com)

ABSTRACT : In the present paper, the deleterious effects of obesity, type 2 diabetes and insulin resistance, systolic and diastolic hypertension on the rate of progression of fibrosis in non-alcoholic fatty liver disease (NAFLD) patients are illustrated using a new approach utilizing the Poisson regression to model the transition rate matrix. The observed counts in the transition counts matrix are used as response variables and the covariates are the risk factors for fatty liver. Then the estimated counts from running the Poisson regression are used to estimate the transition rates using the continuous time Markov chains (CTMCs) followed by exponentiation of the estimated rate matrix to obtain the transition probability matrix at specific time points. A depicted, hypothetical, observational, prospective longitudinal study of 150 participants followed up every year for a total of 28 years recording their demographic characteristics and their timeline follow up are demonstrated. The findings revealed that insulin resistance expressed by HOMA-IR 2 had the most deleterious effects among other factors for increasing the rate of forward progression of patients from state 1 to state 2 as well as from state 2 to state 3 and from state 3 to state 4. The higher the level of HOMA-IR is, the more rapid the rate of progression is. This analysis helps the health policy makers and medical insurance managers to allocate the financial and human resources for investigating and treating high risk patients for NAFLD. In addition, this analysis can be used by pharmaceutical companies to conduct longitudinal studies to assess the effectiveness of the newly emerging anti-fibrotic drugs.

Categories: Endocrinology, Hepatology, Epidemiology, Biostatistics, and Bioinformatics

Key Words: log-linear model, Multistate model, Non-Alcoholic Fatty Liver Disease, Poisson regression, Continuous time Markov chains, longitudinal studies, Maximum Likelihood estimation, Mean Sojourn Time, HOMA2-IR.

Introduction And Background

Continuous time Markov chains (CTMCs) are valuable mathematical and statistical tools. They are of great potentiality to be used for evaluation of disease progression over time. NAFLD is an increasingly worldwide epidemic, paralleling the rise in incidence of obesity and type II diabetes which are approaching a pandemic level. This emerging health problem is mainly due to sedentary life styles and western eating habits of ingesting high fat and carbohydrate diets. The pathological milestone for NAFLD is insulin resistance and hyperinsulinemia. This will eventually results in type II diabetes with all its adverse complications like vascular diseases and fatty liver disease. On the other hand, NAFLD can cause type II diabetes, as the prevalence of diabetes in NAFLD ranges between 18% and 45%. Moreover, the prevalence of NAFLD in type II diabetic patients ranges between 49% and 75%. [1].

NAFLD can be modeled using the simplest form for health, disease, and death model. It is composed of 4 states. One state for susceptible individuals with risk factors like type 2 diabetes, dyslipidemia, obesity, and hypertension. The second state is the NAFLD phenotypes. The other two competing states for death are: one for liver-related mortality as a complication of NAFLD and the other state is the death causes unrelated to liver disease [2]. This is shown in Fig. (1):

FIGURE 1: General Model Structure [2]

In addition, NAFLD can be modeled in more elaborate expanded form which includes nine states [3]. The first eight states are the states of disease progression over time and the ninth state is the death state[2], as illustrated in Fig.(2).

FIGURE 2: Disease Model Structure [2]

NAFL-NO FB = nonalcoholic fatty liver with no fibrosis (stage 1). NASH-NO FB = nonalcoholic steato-hepatitis with no fibrosis (stage 2). NASH-FB = nonalcoholic steato-hepatitis with fibrosis (stage 3). CC= compensated cirrhosis (stage 4). DCC= de-compensated cirrhosis (stage 5). LT= liver transplant (stage 6). PLT =post liver transplant (stage 7). HCC =hepato-cellular carcinoma (stage 8). EM= extra-mortality (stage 9).

Moreover, fibrogenesis is a dynamic process that goes back and forth among the early stages of the expanded model. Stages of fibrous tissue formation are early seen in NAFLD process. Fibrosis progresses if the risk factors for its formation are not eliminated. Fibrosis is an ominous sign for loss of liver functions. A subset of the early states, when the fibrous tissue develops, is used to relate these risk factors to the rates. Definition of each state is shown in Fig.(3)[4][5].

FIGURE 3: NAFLD with the evolving fibrosis stages.[3]

F0= no fibrosis (stage 0) whether hepatic steatosis is present or not. *NASH-FB-1* =nonalcoholic steatohepatitis with mild fibrosis (stage 1). *NASH -FB-2* = NASH with moderate fibrosis (stage 2). *NASH -FB-3* = NASH with advanced or severe fibrosis (stage 3). *CC*= compensated cirrhosis (stage 4) which is the more severe or advanced form of fibrosis.

Kalbfleisch & Lawless [6] related the instantaneous rate of transitions from state i to state j to covariates, by regression modeling of the Q transition rate matrix using log-linear model for the Markov rates.

The previous studies mainly included the evaluation of 2 paired biopsies, initial and second biopsies, then grouping the patients according to the findings into stable, regressors, slow progressors and rapid progressors without precise estimation of specific transition rates among states and without proper estimation of the predictive value of each variable on these specific rates. The rate of fibrosis progression was estimated by dividing the difference in fibrosis stage between biopsies by the time interval (in years) and this was performed to account for the time differences between the biopsies[7]. And either univariate or multivariate linear regression was used to relate the risk factors with the rate of progression. Some studies utilized multivariate logistic regression instead of linear regression.

This depicted study differs from the previous studies in many aspects. First, it proposes recording multiple repeated observations over time. Second, it suggests running Poisson regression to relate the transition rates among states with the risk factors. Third, it recommends using continuous time Markov chains to obtain the transition probabilities and predict the expected counts of patients in each state at specific time point in the future. The counts of each transition can be modeled as a function of some explanatory variables reflecting the characteristics of the patients. The Poisson regression model specifies that each response y_i is drawn from a Poisson population with parameter λ_i , which is related to the covariates. The primary equation (1) of the model is

$$P(Y = y_i | x_i) = \frac{\exp(-\lambda_i) \times \lambda_i^{y_i}}{y_i!}. \quad (1)$$

The most common formulation (2) for the λ_i is the log-linear model:

$$\ln \lambda_i = x_i' B. \quad (2)$$

And the expected number of events per period is given by (3) :

$$E[y_i | x_i] = \text{var}[y_i | x_i] = \lambda_i = \exp(x_i' B). \quad (3)$$

The observed counts in the transition counts matrix are used as response variables and the covariates are the risk factors for fatty liver where the participants are subjected to same periods of follow up. Then the estimated counts obtained from running the Poisson regression are used as input to estimate the transition probability matrix

using the CTMC. To expound this procedure, a hypothetical example is used in the form of an observational prospective longitudinal study.

AI [8] used the same data in previous work , but in this article the author discusses the issue of multi-collinearity, the equi-dispersion Poisson of response variables in the presence of excess zeros, more comparisons between this work and previous works, and finally the author high lightens the benefit of such analysis to pharmacoeconomic evaluation and healthcare economics.

Review

Materials and methods

Patients

One hundred fifty participants were followed up every year for 28 years, and at each visit the characteristics of the participants were recorded like: sex (0=female, 1=male), age, BMI, LDL-chol, HOMA2_IR, systolic blood pressure as well as the diastolic pressure .For each participant the recorded value is the mean of the follow up measurements. The age is the median value.

Statistical analysis

The relationship between the response variable (counts of transitions) and the predictors were nonlinear as shown by Lowess smoother. Restricted cubic spline was used to obtain a suitable functional form of the predictors to fit a Poisson model using STATA 14. The CTMCs were used to obtain transition probability matrix and transition rate matrix. *P*-value of <0.05 was considered statistically significant; all tests were two-sided tests. (See suppl. Mat., Appendix A).

Results

The distribution of the transition counts among the states was Poisson. The application of Lowess smoother showed the nonlinear relationship between the predictors and the response variables as shown in Fig (4). Table (1) & (2) show the transition counts for each state. (See also suppl. Mat., Appendix B).

FIGURE (4): Lowess smoother showing the nonlinear relationship between the transition counts from F0 to F1 and the HOMA2-IR levels.

Table (1): summary of transition counts among the states

Counts	Transition 0→1	Transition 1→2	Transition 2→3	Transition 3→4	Transition 1→0	Transition 2→1	Transition 3→2	Transition 2→0	Transition 3→1
0	63	96	121	128	121	127	130	138	139
1	58	43	23	22	24	17	17	11	9
2	25	9	4		3	5	3	1	2
3	4	2	2		2	1			

Table (2): Observed transition counts of the patients over the 28 years

	State 0	State1	State2	State3	State4	total
State0	1909	120	15	6	0	2050
State1	36	1116	67	28	0	1247
State2	13	30	703	37	0	783
State3	11	14	23	50	22	120
State4	0	0	0	0	0	0
						4200

Initial observed rates are:

$$\lambda_{01} = \frac{120}{2050} = 0.059, \quad \lambda_{12} = \frac{67}{1247} = 0.0537, \quad \lambda_{23} = \frac{37}{783} = 0.047, \quad \lambda_{34} = \frac{22}{120} = 0.183$$

$$\mu_{10} = \frac{36}{1247} = 0.0288, \mu_{21} = \frac{30}{783} = 0.0383, \mu_{32} = \frac{23}{120} = 0.191, \quad \mu_{20} = \frac{13}{783} = 0.016, \mu_{31} = \frac{14}{120} = 0.116$$

The observed counts were used as the response variables to fit the Poisson regression model. For each transition count, the model that represented the most explainable covariates with their Beta coefficients and the corresponding incidence rate ratios were illustrated in (Suppl. Mat, Appendix B). The transitions were subdivided into progressive transitions and regressive transitions. Comparison between the null model and full model was explored in supplementary materials. The main important result is that HOMA2-IR is positively correlated with all progressive transitions and is inversely related to the regressive transitions as shown in table (3) & table (4).

The continuous predictors (age, BMI, HOMA2-IR, LDL-Chol, systolic and diastolic blood pressure) were highly correlated with a correlation coefficient of 0.99, and with a condition number for data matrix ($x'x$) of 453.57. The condition number for the data matrix ($x'x$) constructed from the transformed variables used in the analysis (HOMAsp1, HOMAsp2, LDLsp2, sysPS2, diasPS2) was 54.89. These transformed variables were also highly correlated. However, the condition number did not exceed 100. Thus, this multicollinearity can be considered non-harmful and it will not affect the analysis.[9]

Progressive transitions

Table (3): Parameters for each transitions

Transition from F0 to F1						
	LDLsp2	HOMAsp1	SysSP2	LDLsp2# HOMAsp1	LDLsp2# SysSP2	HOMAsp1# SysSP2
B co.(P)	0.523 (0.032)	4.096 (0.000)	-0.628 (0.070)	-0.179 (0.011)	0.003 (0.000)	0.151 (0.122)
CI for B co	(0.046,1.000)	(3.452,4.740)	(-1.308,0.052)	(-0.317,-0.041)	(0.002 , 0.003)	(-0.040 , 0.342)
IRR	1.687	60.097	0.534	0.836	1.003	1.163
CI for IRR	(1.047 , 2.718)	(31.569,114.4)	(0.270 , 1.054)	(0.728 , 0.960)	(1.002 , 1.003)	(0.960 , 1.408)
Transition from F1 to F2						
	LDLsp2	HOMAsp1	SysSP2	LDLsp2# HOMAsp1	LDLsp2# SysSP2	HOMAsp1# SysSP2
B co.(P)	0.311 (0.432)	5.486 (0.000)	-0.314 (0.564)	-0.105 (0.367)		0.079 (0.616)
CI for B co	(-0.465 , 1.086)	(4.366, 6.606)	(-1.383 , 0.754)	(-0.332,0.123)		(-0.231 , 0.389)
IRR	1.364	241.179	0.730	0.901		1.083
CI for IRR	(0.628 , 2.962)	(78.690, 739.192)	(0.251,2.126)	(0.717,1.131)		(0.794,1.476)
Transition from F2 to F3						
	LDLsp2	HOMAsp1	SysSP2	LDLsp2# HOMAsp1	LDLsp2# SysSP2	HOMAsp1# SysSP2
B co.(P)	-1.480 (0.031)	6.174(0.046)	2.497(0.010)	0.390(0.042)	-0.001(0.687)	-0.655(0.017)
CI for B co	(-2.823,-0.137)	(0.112 , 12.237)	(0.602 , 4.391)	(0.014 , 0.766)	(-0.005 , 0.004)	(-1.191 , -0.118)
IRR	0.228	480.318	12.143	1.477	0.999	0.520
CI for IRR	(0.059, 0.872)	(1.118, 2.06e+5)	(1.826, 80.754)	(1.014 , 2.151)	(0.995 , 1.004)	(0.304 , 0.889)
Transition from F3 to F4						
	LDLsp2	HOMAsp1	SysSP2	LDLsp2# HOMAsp1	LDLsp2# SysSP2	HOMAsp1# SysSP2
B co.(P)	0.452(0.000)	10.866 (0.000)	0.073(0.141)	-0.166(0.000)		
CI for B co	(0.345, 0.559)	(8.119, 13.613)	(-0.024, 0.171)	(-0.201 , -0.131)		
IRR	1.571	52375.984	1.076	0.847		
CI for IRR	(1.412, 1.748)	(3357.9, 8.17e+5)	(0.976, 1.187)	(0.818 , 0.877)		

Regressive transitions

Table (4): Parameters for each transitions

Transition from F1 to F0						
	LDLsp2	HOMAsp2	SysSP2	LDLsp2# HOMAsp2	LDLsp2# SysSP2	HOMAsp2# SysSP2
B co.(P)	-0.454(0.063)	-4.489(0.130)	1.340(0.000)	0.290(0.002)	-0.010(0.005)	-0.286(0.048)
CI for B co	(-0.932, 0.024)	(-10.294, 1.316)	(0.729, 1.951)	(0.102, 0.478)	(-0.017, -0.003)	(-0.571, -0.002)
IRR	0.635	0.011	3.820	1.337	0.990	0.751
CI for IRR	(0.394, 1.024)	(0.000, 3.730)	(2.074, 7.034)	(1.108, 1.612)	(0.983, 0.997)	(0.565, 0.998)
Transition from F2 to F1						
	LDLsp2	HOMAsp2	SysSP2	LDLsp2# HOMAsp2	LDLsp2# SysSP2	HOMAsp2# SysSP2
B co.(P)	-0.128(0.499)	-3.288(0.242)	0.913(0.000)	0.152(0.022)	-0.010(0.003)	-0.114(0.317)
CI for B co	(-0.499, 0.243)	(-8.800, 2.224)	(0.519, 1.307)	(0.022, 0.282)	(-0.017, -0.003)	(-0.338, 0.109)
IRR	0.880	0.037	2.492	1.164	0.990	0.892
CI for IRR	(0.607, 1.275)	(0.000, 9.244)	(1.681, 3.694)	(1.022, 1.326)	(0.983, 0.997)	(0.713, 1.116)
Transition from F3 to F2						
	LDLsp2	HOMAsp2	SysSP2	LDLsp2# HOMAsp2	LDLsp2# SysSP2	HOMAsp2# SysSP2
B co.(P)	0.302 (0.154)	-5.214(0.103)	0.422(0.142)	0.002(0.984)	-0.012(0.006)	0.132(0.375)
CI for B co	(-0.113, 0.716)	(-11.478, 1.05)	(-0.142, 0.987)	(-0.198, 0.202)	(-0.02, -0.003)	(-0.16, 0.425)
IRR	1.352	0.005	1.526	1.002	0.998	1.142
CI for IRR	(0.893, 2.047)	(0.000, 2.859)	(0.868, 2.683)	(0.821, 1.223)	(0.98, 0.997)	(0.852, 1.529)
Transition from F2 to F0						
	LDLsp2	HOMAsp2	SysSP2	DiasSP2		
B co.(P)	0.076(0.335)	-2.713(0.000)	-0.123(0.010)	0.358(0.001)		
CI for B co	(-0.079, 0.231)	(-4.102, -1.324)	(-0.216, -0.030)	(0.143, 0.573)		
IRR	1.079	0.066	0.884	1.430		
CI for IRR	(0.924, 1.260)	(0.017, 0.266)	(0.806, 0.970)	(1.154, 1.773)		
Transition from F3 to F1						
	LDLsp2	HOMAsp2	SysSP2	DiasSP2		
B co.(P)	0.145(0.038)	-2.476(0.000)	-0.129(0.004)	0.276(0.003)		
CI for B co	(0.008, 0.282)	(-3.769, -1.183)	(-0.216, -0.042)	(0.093, 0.459)		
IRR	1.156	0.084	0.879	1.318		
CI for IRR	(1.008, 1.326)	(0.023, 0.306)	(0.805, 0.959)	(1.098, 1.582)		

For each of the transitions from state (i) to state (j), where λ_{ij} denotes the counts of transition from state (i) to state (j), and after running the Poisson model, the linear predictor $\ln \lambda_{ij} = x'_n B$ for each participant (n) is exponentiated, $E[y_n | x_n] = \lambda_{ij} = \exp(x'_n B)$, to obtain the expected counts of transition that had been accomplished by this participant during this 28 years. Then the result is rounded to the appropriate integer and summed to get all counts for this transition and then compare it to the observed counts accomplished by the all participants.

The n_{i+} is the total marginal transition counts out of this state, which is assumed to be constant. The estimated and obtained counts from running the Poisson will be substituted in the transition count table. Because the marginal counts are assumed to be the same and when using the initial rates calculated as $\theta_0 = \frac{n_{ijr}}{n_{i+}}$ where the n_{ijr} is the transition counts from state i to state j , the Q matrix can be estimated. (Hint: the numerators below are the estimated counts obtained from running the Poisson regression).

$$\hat{Q} = \begin{bmatrix} -0.059 & 0.059 & 0 & 0 & 0 \\ 0.029 & -0.080 & 0.051 & 0 & 0 \\ 0.015 & 0.033 & -0.093 & 0.045 & 0 \\ 0 & 0.108 & 0.158 & -0.433 & 0.167 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ where}$$

$$\lambda_{01} = \frac{120}{2050} = 0.059, \lambda_{12} = \frac{64}{1247} = 0.051, \lambda_{23} = \frac{35}{783} = 0.045, \lambda_{34} = \frac{20}{120} = 0.167$$

$$\mu_{10} = \frac{36}{1247} = 0.029, \mu_{21} = \frac{26}{783} = 0.033, \mu_{32} = \frac{19}{120} = 0.158, \mu_{20} = \frac{12}{783} = 0.015, \mu_{31} = \frac{13}{120} = 0.108$$

Probability transition matrix is obtained from exponentiation of this Q matrix after 1 year:

$$P(\Delta t = 1) = \exp(\hat{Q} \times \Delta t) = \begin{bmatrix} 0.944 & 0.055 & 0.001 & 0 & 0 \\ 0.027 & 0.925 & 0.047 & 0.001 & 0.0001 \\ 0.014 & 0.033 & 0.915 & 0.035 & 0.003 \\ 0.002 & 0.086 & 0.125 & 0.651 & 0.136 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Goodness of fit

To calculate goodness of fit for multistate model used in this example, it is like the procedure used in contingency table, and it is calculated in each interval then sum:

Step 1: $H_0 = \text{future state does not depend on the current state}$

$H_1 = \text{future state depends on the current state}$

Step 2: calculate the $p_{ij}(\Delta t = 1) = \exp(Q \times \Delta t) = \begin{bmatrix} 0.944 & 0.055 & 0.001 & 0 & 0 \\ 0.027 & 0.925 & 0.047 & 0.001 & 0.0001 \\ 0.014 & 0.033 & 0.915 & 0.035 & 0.003 \\ 0.002 & 0.086 & 0.125 & 0.651 & 0.136 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Step3: calculate the expected counts in this interval by multiplying each row in the probability matrix with the corresponding total marginal counts in the observed transition counts table in the same interval to get the expected counts as shown below:

	State 0	State 1	State 2	State 3	State 4
State 0	1934.175	112.955	2.87	0	0
State 1	34.168	1153.101	58.484	1.122	0.125
State 2	11.275	25.604	716.367	27.248	2.506
State 3	0.276	10.356	14.94	78.144	16.284
State 4	0	0	0	0	0

Step 4: apply $\sum_{i=1}^5 \sum_{j=1}^5 \frac{(O_{ij} - E_{ij})^2}{E_{ij}} = 1140.097 \sim \chi^2_{(5-1)(5-1)(.05)}$

So from the above results the null hypothesis is rejected while the alternative hypothesis is accepted and the model fits the data that is to mean the future state depends on the current state with the estimated transition rates and probability matrices as obtained.

Health Economics:

This transition probability matrix can be used to predict the count of patients in each state at specific time point using (4), for example if a cohort of 6000 patients with the following number in each state being [3000 1800 1020 180 0] so after one year the predicted counts will be [2895 1879 1044 154 28]. This can be achieved by multiplying the initial count distribution of the patients with the transition probability calculated at the required specific time point , $p_{ij}(t) = \exp(Qt)$

$$E[u_j(t)|u_j(0)] = \sum_{j=1, i=1}^5 u_j(0)P_{ij}(t) \quad i \& j = 1, \dots, 5 \quad (4)$$

Let $u(0)$ be the size of patients in a specific state at specific time $t = 0$. The initial size of patients $U(0) = u_j(0)$, as there are 4 transient states (F0 to F3) and 1 absorbing state (F4), where $u_j(0)$ is the initial size or the number of patients in state j at time $t = 0$ given that $u_5(0) = 0$ i.e initial size of patients in state 5 (absorbing state) is zero at initial time point $= 0$. As the transition or the movement of the patients among states is independent, at the end of the whole time interval $(0, t)$, there will be $u_j(t)$ patients in the transient states at time t , and there will also be $u_5(t)$ patients in state 5 (F4= liver cirrhosis) at time t .

In addition, the state probability distribution $\pi(t)$, which is the probability distribution for each state at a specific time point given the initial probability distribution $\pi(0)$, can be estimated by applying the following formula: $\pi(t) = \pi(0)P(t)$. In this example, the cohort of 6000 patients has initial probability distribution of [0.5 0.3 0.17 0.03 0] after one year, the state probability distribution will be [0.4825 0.3131 0.174 0.0257 0.0046].

Using the above information each year in addition to the costs of investigations and treatments and the quality adjusted life years for the patients, pharmaco-economic evaluation can be assessed in three different categories: The cost-benefit analysis, the cost-effectiveness analysis, and the cost utility analysis.[10] [11]

This approach differs from the one used by Rustgi *et al.* [12] who depends on calculating the cost-effectiveness analysis by following a cohort of patients, all starting at the same initial state till death. While in the approach proposed in this article, sampling the population and estimating the transition probability matrix to predict the counts in the future time, any cohort of patients can then be followed up utilizing the information gained from sampling the general population.

Discussion:

To elucidate the agreements and comparisons as regard the findings in this study with the previous studies, a brief summary of some of the previously conducted studies highlighting the effects of various factors on the rate of progression of fibrosis in NAFLD patients, is described in the following discussion.

Hui *et al.* [13] conducted a study on 17 patients who had previous liver biopsy showing evidence of steatosis with or without the presence of necroinflammation and fibrosis. Those patients underwent second liver biopsies with a median of 6 years apart (range: 3.8-8 years). Nearly, more than half of them developed progressive fibrosis as compared to the initial biopsy; this was due to the fact that these patients were initially suffering from steatohepatitis, although there was no significant correlation between the degree of steatohepatitis and the degree of fibrosis between the 2 biopsies. However the correlation was significant between the initial stage of fibrosis and the fibrosis grade in the second biopsy. Also the clinical and laboratory parameters were not statistically significant between the recorded values during the first and the second biopsy. The changes in these parameters also showed no significant correlation with changes in the scores of steatosis, necroinflammation or fibrosis. Nevertheless, there was

a negative correlation, although non-significant, between the change in the score of fibrosis and each of the changes: in the BMI, in plasma total cholesterol levels, in plasma triglyceride levels and in glycosylated hemoglobin. During the course of the follow up, 2 patients developed type II diabetes and one developed hypertension but without progression of fibrosis, their initial biopsy revealed F0 and the second one was also F0. Another one patient developed type II diabetes with evolution of the fibrosis from F0 to F2, and another 2 patients developed hypertension with advancement of fibrosis from F0 to F1.

Fassio *et al.* [14] conducted a study on 22 patients who had liver biopsy with evidence of NASH and found that 31.8% (7 patients=P group) had progression of liver fibrosis over a median follow-up of 4.7 years. The other group was 68.2% (15 patients=NP group) and did not show any progression over a median follow-up of 4.3 years. The rate of progression in the entire population was estimated as 0.059 fibrosis units per year (mean difference in fibrosis score divided by mean interval in years between the first and second biopsies $=0.32/5.34=0.059$), while; the rate of progression in the P group was $1.85/6.59=0.28$. There were no statistical difference as regards the clinical, biochemical, grade of steatosis, and grade of inflammation between the 2 groups except for presence of obesity and higher BMI (progressor were more obese with higher BMI than the nonprogressor) whether this was performed during the initial liver biopsy or the final liver biopsy. The gradients between the final and basal results were not statistically significant as regards the clinical, biochemical, grade of steatosis, and grade of inflammation between the 2 groups including the BMI.

Adam *et al.* [7] conducted a study on 103 patients who had performed 2 liver biopsies with mean follow up period of 3.2 ± 3 years (range = 0.7-21) between the first and the second biopsy. A total of 38 patients were progressors, 35 patients were stable, and 30 patients were regressors. No clinical or biochemical variables were statistically different between the progressors, stable and regressors. The rate of fibrosis change varied from -2.05 to 1.7 stages/year and calculated as stated in the introduction. Using univariate regression model, presence of diabetes, AST/ALT ratio, steatosis grades and fibrosis stage were the only significant variables. By multivariate linear regression analysis and adjusting for age and BMI, only the presence of diabetes and earlier fibrosis stage were significantly associated with a higher rate of fibrosis progression. He also found that there was no significant correlation between rate of progression and HOMA.

There are many studies performed by Ekstedt *et al.* [15], Teli *et al.* [16], Pais *et al.* [17], Argo *et al.* [18], Evans *et al.* [19], Hamaguchi *et al.* [20], and Wong *et al.* [21]. They reported the fibrosis progression rates utilizing 2 paired biopsies and the reader can refer to them.

The findings of the present study demonstrate that HOMA2-IR has a positive and a statistically significant effect on progression of fibrosis among the different states. Running multivariate Poisson regression reveals that the main key players for progression are the HOMA2-IR, LDL-chol and systolic blood pressure explaining about 35% to 60 % of variability in the rates of progression. However, HOMA2-IR has a negative effect that is not statistically significant on the rate of remission or regression from F1 to F0, from F2 to F1, and from F3 to F2; but it is statistically significant on rate of remission from F2 to F0 and from F3 to F1. Poisson regression model explained that the same factors and the interactions between them were responsible for about 60% to 70% of variability in the rates of remission among the states. Each of the high HOMA2-IR level and high LDL level significantly decreases the effect of the other on the rate of progression from F0 to F1 and from F3 to F4, and this can be a protective mechanism in attempt to slow down the rate of progression of fibrosis. It is also notable that each of the low HOMA2-IR level and low LDL level significantly increases the effect of the other on the rate of remission from F1 to F0 and from F2 to F1, and this can be a protective mechanism in attempt to accelerate the rate of regression of fibrosis. This can also be achieved with the help of rigorous control of the blood level of insulin, glucose, cholesterol and blood pressure. It is also notable that each of the high level of systolic blood pressure and low LDL level significantly decreases the effect of each other on the rate of remission from F1 to F0, from F2 to F1, and from F3 to F2. This point to the possibility that controlling the most deleterious factors like hyperinsulinemia and hypercholesterolemia, even in absence of strict control of hypertension, can still has a beneficial effect on repressing

the fibrogenesis. This in association with life style modification, in the form of physical exercise and low caloric diet, has a great impact on arresting the process of fibrogenesis.

The newly emerging anti-fibrotic drugs will also help physicians treating the fibrogenesis. In FLINT study, conducted on 283 non-cirrhotic NASH patients taking obeticholic acid (OCA), 25 mg daily; the improvement in the histology detected by NAFLD activity score (NAS) was 2 points or more with no deterioration of fibrosis and 35% of patients taking OCA had a decrement in fibrosis score by at least one stage in comparison with 19 % in the placebo arm. REGENERATE study (still in progress, with the estimated primary completion date is on September 2025 as shown in clinicaltrials.gov official site) will evaluate safety and efficacy of Obeticholic Acid (OCA) in NASH patients with fibrosis whom are randomized to a daily dose of 25 mg, 10 mg, and placebo, with end points like: amelioration of fibrosis by at least one stage and decaying of NASH with no deterioration of fibrosis. At 18 month of randomization, liver biopsy revealed statistically significant histological amelioration of fibrosis and decaying of NASH with no deterioration in fibrosis for both 10 mg and 25 mg doses. In GOLDEN study, conducted on 274 NASH patients, 120 mg Elafibranor taken daily for 52 weeks induced decaying of moderate to severe NASH in a meaningfully higher percentage of patients in comparison to placebo, moreover; these patients also showed lowering in fibrosis stage compared to non-resolving NASH patients. RESOLVE-IT trial (last update was on November 30, 2020, as shown in clinicaltrials.gov official site, but the study is still in progress according to [22]) emerged in May 2020 had shown that 19.2% of patients, on 120 mg daily Elafibranor, had NASH decay without deterioration of fibrosis compared to 14.7% in the placebo group, which was not statistically significant. Furthermore, 24.5% of patients had shown fibrosis amelioration of more than one stage compared to 22.4% in the placebo group, which was also not statistically significant. In CENTAUR trial, conducted over 289 patients taking cenicriviroc (CVC), 150 mg daily and placebo for 52 weeks, no comparative betterment in NAS between NASH group and placebo was seen, however; there was one stage or more amelioration of fibrosis with no deterioration of NASH in the group taking the CVC compared with placebo group. The AURORA trial (primary completion dates were October 2021 according to clinicaltrials.gov site and October 2028 according to [22]) will evaluate long-term safety and efficacy of 150 mg daily CVC for the treatment of fibrosis in NASH adult patients at 2 phases, the first has endpoint of at least one stage amelioration of fibrosis without deterioration of NASH at month 12, and phase 2 has end point that is cirrhosis, liver-related outcome as HCC, and all causes of mortality. In a small, open-label, randomized phase II trial including 72 biopsy-proven NASH patients (NAS $\geq$ 5 and stage 2-3 liver fibrosis) receiving 18 mg daily Selonsertib for 24 weeks, there was significant improvement in liver disease activity, fibrosis, stiffness, liver fat content, and progression to cirrhosis [23].

FLINT, GOLDEN, and CENTAUR are phase IIb placebo-controlled RCT (randomized control trial), while REGENERATE, RESOLVE-IT, and AURORA are randomized, placebo-controlled, double-blinded, multicenter phase III trials

The strength of this study is the conduction of multiple frequent repeated observations over a long time period of follow up on a large number of at high risk participants for developing NAFLD and performing a liver biopsy in each visit. Although this may be realistically infeasible to be done in each visit, using non-invasive techniques [24][25] can substitute the invasive liver biopsy and this may be consider a weakness of the study. But the advantage of techniques like MRI and machine learning [26], to assess the liver texture and correlate these finding with the histological findings in liver biopsy, can overcome this weakness. Liver biopsy can also be reserved in situations where the results of non-invasive tests are inconclusive and these non-invasive tests decrease the number of liver biopsies each patient may encounter to do. The proposed follow up period is too long to wait for the obtained results and this can be overcome by using adaptive clinical trials.

Conclusion:

The distribution of the counts was Poisson (mean=variance); that is to mean, these counts were equidispersed. However, all the counts showed excess zeros except the transition from F0 to F1 where the zeros constituted 44% of the total count of this transition. Tlhaloganyang and Sakia [27] found that the equidispersed counts data with excessive zeros can be modeled with Poisson regression which is the best model to represent the data. Also the AIC scores obtained by them after running Poisson regression, on their tested data whether simulated or real, were less than the AIC scores after running ZIP on the same datasets. In this article, the predictors were normally distributed and applying the restricted cubic spline transformation was used to better specify the functional form of these predictors. Nevertheless, the raw predictors as well as the transformed predictors were highly correlated. But the condition number is below 100, which is not harmful for the analysis as shown in the results. Vatcheva *et al.* [28] highlighted the fact that the majority of researchers do not mention the multicollinearity diagnostics when running the regression models, discussed the causes and effects of this lack, and proposed some remedies to treat multicollinearity like: principle component analysis, partial least squares regression, and ridge regression analysis. Akram *et al.* [29] used principle component ridge type estimator for the inverse Gaussian regression model. Many investigators like: Liu [30], Kabria and Lukman [31], Lukman *et al.* [32], and had proposed different techniques to manage the multicollinearity problem between the predictors when running regression models. Some of them, who developed methods for Poisson regression are Mansson and Shukur [33], Mansson *et al.* [34], Lukman *et al.* [35], [36], and Qasim *et al.* [37]. In this paper, none of these methods were used as the Poisson model was mainly used to give preliminary vision about the effects of the high risk factors on the transition counts. It was not also used for prediction and the condition number was less than 100. Once the estimated counts were obtained, they were fed to the CTMC to estimate the transition rate matrix and transition probability matrix at any specified time point. Thus, physicians can follow a cohort of any patients in various states and can obtain their state probability distribution at different time points.

To sum up, in the present study, running Poisson regression model is used to obtain the expected counts of transition among states. These counts are used as input into the homogenous CTMC. Using this CTMC, the transition rate matrix is estimated and thus the probability of progression of participants from specific state to another one at specific time point can be estimated by exponentiation of this rate matrix. This probability matrix at any specific time point multiplied by the initial probability distribution of a cohort of patients can be used to predict the number of the participants in each state later on in different time points. This helps health policy makers and insurance managers for allocating the human and financial resources to investigate and treat the at high risk patients for developing NAFLD. The Poisson regression model is utilized to relate these high risk covariates to the rate of transmission among states. Also this approach can be used in the clinical trials to assess the effectiveness of the newly emerging anti-fibrotic drugs. The epidemiologists can utilize this methodology to estimate the effect of risk factors on the incidence rates of progression and remission among the different states of liver fibrosis due to NAFLD.

This hypothetical study is coded by stata-14 and is published in code ocean site with the following URL:
<https://codeocean.com/capsule/4752445/tree/v3> , DOI=10.24433/CO.8778229.V3

The code to estimate the Q transition rate matrix for the observed transition counts using continuous time Markov chains are published in the code Ocean site with following URL:
<https://codeocean.com/capsule/6377472/tree/v2> , DOI=10.24433/CO.2144346.V2

The dataset is present on IEEEDataPort site with the following URL:
<https://ieee-dataport.org/documents/fibrosis-nfld#files> , with the following DOI: 10.21227/dr5j-gs46

Abbreviations:

CC: compensated cirrhosis (stage 4), CTMC: continuous time Markov chains, DCC: de-compensated cirrhosis (stage 5), EM: extra-mortality (stage 9), HCC: hepato-cellular carcinoma (stage 8), LT: liver transplant(stage 6), NAFLD: non-alcoholic fatty liver disease, NAFL-NO FB: nonalcoholic fatty liver with no fibrosis (stage 1), NASH: non-alcoholic steatohepatitis, NASH-NO FB : nonalcoholic steato-hepatitis with no fibrosis (stage 2), NASH-FB: nonalcoholic steato-hepatitis with fibrosis (stage 3), PLT : post liver transplant (stage 7), T2DM: type 2 diabetes mellitus.

Declarations:**Ethics approval and consent to participate**

Not applicable.

Consent for publication

Not applicable

Availability of data and material

Not applicable. Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Competing interests

The author declares no competing interests of any type.

Funding

No funding resource. No funding roles in the design of the study and collection, analysis, and interpretation of data and in writing the manuscript are declared

Authors' contribution

AI carried the conceptualization by formulating the goals, aims of the research article, formal analysis by applying the statistical, mathematical and computational techniques to synthesize and analyze the hypothetical data, carried the methodology by creating the model, software programming and implementation, supervision, writing, drafting, editing, preparation, and creation of the presenting work.

Acknowledgement

Not applicable

Disclosures

Conflicts of interest: In compliance with the ICMJE uniform disclosure form, all authors declare the following:

Payment/services info: All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work.

Other relationships: All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

References

- [1] Z. M. Younossi, T. Gramlich, C. A. Matteoni, N. Boparai, and A. J. McCullough, "Nonalcoholic fatty liver disease in patients with type 2 diabetes," *Clinical Gastroenterology and Hepatology*, vol. 2, no. 3, pp. 262–265, 2004.
- [2] Z. M. Younossi *et al.*, "The economic and clinical burden of nonalcoholic fatty liver disease in the United States and Europe," *Hepatology*, vol. 64, no. 5, pp. 1577–1586, 2016.
- [3] I. M. Attia, "Novel Approach of Multistate Markov Chains to Evaluate Progression in the Expanded Model of Non-alcoholic Fatty Liver Disease," *Frontiers in Applied Mathematics and Statistics*, vol. 7, 2022, Accessed: Mar. 03, 2022. [Online]. Available: <https://www.frontiersin.org/article/10.3389/fams.2021.766085>
- [4] Z. M. Younossi *et al.*, "Economic and clinical burden of nonalcoholic steatohepatitis in patients with type 2 diabetes in the US," *Diabetes Care*, vol. 43, no. 2, pp. 283–289, 2020.
- [5] S. Singh, A. M. Allen, Z. Wang, L. J. Prokop, M. H. Murad, and R. Loomba, "Fibrosis progression in nonalcoholic fatty liver vs nonalcoholic steatohepatitis: a systematic review and meta-analysis of paired-biopsy studies," *Clinical gastroenterology and hepatology*, vol. 13, no. 4, pp. 643–654, 2015.
- [6] J. D. Kalbfleisch and J. F. Lawless, "The analysis of panel data under a Markov assumption," *Journal of the American Statistical Association*, vol. 80, no. 392, pp. 863–871, 1985.

- [7] L. A. Adams, P. Angulo, and K. D. Lindor, "Nonalcoholic fatty liver disease," *Cmaj*, vol. 172, no. 7, pp. 899–905, 2005.
- [8] M. A. A.-E. A. Elreesh, "Prognostic Factors for Evolution of Non Alcoholic Fatty Liver Disease Patients Utilizing Poisson Regression and Continuous Time Markov Chains," 2021.
- [9] A. Lazaridis, "A note regarding the condition number: the case of spurious and latent multicollinearity," *Quality & Quantity*, vol. 41, no. 1, pp. 123–135, 2007.
- [10] J. O'Hara *et al.*, "Cost of non-alcoholic steatohepatitis in Europe and the USA: the GAIN study," *JHEP Reports*, vol. 2, no. 5, p. 100142, 2020.
- [11] M. Nouredin *et al.*, "Screening for nonalcoholic fatty liver disease in persons with type 2 diabetes in the United States is cost-effective: a comprehensive cost-utility analysis," *Gastroenterology*, vol. 159, no. 5, pp. 1985–1987, 2020.
- [12] V. K. Rustgi, S. B. Duff, and M. I. Elsaid, "Cost-effectiveness and potential value of pharmaceutical treatment of nonalcoholic fatty liver disease," *Journal of medical economics*, no. just-accepted, pp. 1–29, 2022.
- [13] A. Y. Hui *et al.*, "Histological progression of non-alcoholic fatty liver disease in Chinese patients," *Alimentary pharmacology & therapeutics*, vol. 21, no. 4, pp. 407–413, 2005.
- [14] E. Fassio, E. Álvarez, N. Domínguez, G. Landeira, and C. Longo, "Natural history of nonalcoholic steatohapatitis: a longitudinal study of repeat liver biopsies," *Hepatology*, vol. 40, no. 4, pp. 820–826, 2004.
- [15] M. Ekstedt *et al.*, "Long-term follow-up of patients with NAFLD and elevated liver enzymes," *Hepatology*, vol. 44, no. 4, pp. 865–873, 2006.
- [16] M. R. Teli, O. F. W. James, A. D. Burt, M. K. Bennett, and C. P. Day, "The natural history of nonalcoholic fatty liver: a follow-up study," *Hepatology*, vol. 22, no. 6, pp. 1714–1719, 1995.
- [17] R. Pais *et al.*, "A systematic review of follow-up biopsies reveals disease progression in patients with non-alcoholic fatty liver," *Journal of hepatology*, vol. 59, no. 3, pp. 550–556, 2013.
- [18] C. K. Argo, P. G. Northup, A. M. S. Al-Osaimi, and S. H. Caldwell, "Systematic review of risk factors for fibrosis progression in non-alcoholic steatohepatitis," *Journal of hepatology*, vol. 51, no. 2, pp. 371–379, 2009.
- [19] C. D. J. Evans, K. A. Oien, R. N. M. MacSween, and P. R. Mills, "Non-alcoholic steatohepatitis: a common cause of progressive chronic liver injury?," *Journal of clinical pathology*, vol. 55, no. 9, pp. 689–692, 2002.
- [20] E. Hamaguchi *et al.*, "Histological course of nonalcoholic fatty liver disease in Japanese patients: tight glycemic control, rather than weight reduction, ameliorates liver fibrosis," *Diabetes care*, vol. 33, no. 2, pp. 284–286, 2010.
- [21] V. W.-S. Wong *et al.*, "Disease progression of non-alcoholic fatty liver disease: a prospective study with paired liver biopsies at 3 years," *Gut*, vol. 59, no. 7, pp. 969–974, 2010.
- [22] E. Guirguis, Y. Grace, A. Bolson, M. J. DellaVecchia, and M. Ruble, "Emerging therapies for the treatment of nonalcoholic steatohepatitis: A systematic review," *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy*, vol. 41, no. 3, pp. 315–328, 2021, doi: 10.1002/phar.2489.
- [23] N. Alkhoury, F. Poordad, and E. Lawitz, "Management of nonalcoholic fatty liver disease: Lessons learned from type 2 diabetes," *Hepatology Communications*, vol. 2, no. 7, pp. 778–785, 2018, doi: 10.1002/hep4.1195.
- [24] L. Petriclerc, G. Sebastiani, G. Gilbert, G. Cloutier, and A. Tang, "Liver fibrosis: Review of current imaging and MRI quantification techniques," *Journal of Magnetic Resonance Imaging*, vol. 45, no. 5, pp. 1276–1295, 2017.
- [25] G. Musso, R. Gambino, M. Cassader, and G. Pagano, "Meta-analysis: natural history of non-alcoholic fatty liver disease (NAFLD) and diagnostic accuracy of non-invasive tests for liver disease severity," *Annals of medicine*, vol. 43, no. 8, pp. 617–649, 2011.
- [26] K. Schawkat *et al.*, "Diagnostic accuracy of texture analysis and machine learning for quantification of liver fibrosis in MRI: correlation with MR elastography and histopathology," *European radiology*, vol. 30, no. 8, pp. 4675–4685, 2020.
- [27] B.P. Tlhaloganyang* and R.M. Sakia, "Zero inflated Poisson distribution in equidispersed data with excessive zeros," *Research Journal of Mathematical and Statistical Sciences*, vol. 8, no. 1, pp. 31–34, 2020.
- [28] K. P. Vatcheva, M. Lee, J. B. McCormick, and M. H. Rahbar, "Multicollinearity in Regression Analyses Conducted in Epidemiologic Studies," *Epidemiology (Sunnyvale)*, vol. 6, no. 2, p. 227, Apr. 2016, doi: 10.4172/2161-1165.1000227.
- [29] M. Akram, M. Amin, A. Lukman, and S. Afzal, "Principal component ridge type estimator for the inverse Gaussian regression model," *Journal of Statistical Computation and Simulation*, pp. 1–30, Jan. 2022, doi: 10.1080/00949655.2021.2020274.

- [30] K. Liu, "Using Liu-Type Estimator to Combat Collinearity," *Communications in Statistics - Theory and Methods*, vol. 32, no. 5, pp. 1009–1020, Jan. 2003, doi: 10.1081/STA-120019959.
- [31] B. M. G. Kibria and A. F. Lukman, "A New Ridge-Type Estimator for the Linear Regression Model: Simulations and Applications," *Scientifica*, vol. 2020, p. e9758378, Apr. 2020, doi: 10.1155/2020/9758378.
- [32] A. F. Lukman, K. Ayinde, S. Binuomote, and O. A. Clement, "Modified ridge-type estimator to combat multicollinearity: Application to chemical data," *Journal of Chemometrics*, vol. 33, no. 5, p. e3125, 2019, doi: 10.1002/cem.3125.
- [33] K. Månsson and G. Shukur, "A Poisson ridge regression estimator," *Economic Modelling*, vol. 28, no. 4, pp. 1475–1481, 2011.
- [34] K. Månsson, B. M. G. Kibria, P. Sjölander, and G. Shukur, "Improved Liu Estimators for the Poisson Regression Model," *International Journal of Statistics and Probability*, vol. 1, Apr. 2012, doi: 10.5539/ijsp.v1n1p2.
- [35] A. F. Lukman, E. Adewuyi, K. Månsson, and B. M. G. Kibria, "A new estimator for the multicollinear Poisson regression model: simulation and application," *Sci Rep*, vol. 11, no. 1, p. 3732, Feb. 2021, doi: 10.1038/s41598-021-82582-w.
- [36] A. F. Lukman, B. Aladeitan, K. Ayinde, and M. R. Abonazel, "Modified ridge-type for the Poisson regression model: simulation and application," *Journal of Applied Statistics*, pp. 1–13, Feb. 2021, doi: 10.1080/02664763.2021.1889998.
- [37] M. Qasim, B. M. G. Kibria, K. Månsson, and P. Sjölander, "A new Poisson Liu Regression Estimator: method and application," *Journal of Applied Statistics*, vol. 47, no. 12, pp. 2258–2271, Sep. 2020, doi: 10.1080/02664763.2019.1707485.

Appendix: “CTMC analyzing NAFLD progression (big model)”

The History of this MATLAB code on the site **CODE OCEAN** site, it is titled with “New Method Utilizing Continuous Time Markov Chains to Analyze Evolution in the Nine State Model of Non-Alcoholic Fatty Liver Disease”, and associated with article in Frontiers with DOI: 10.3389/fams.2021.766085 :

Published version 1 on 9 September 2021, URL: <https://codeocean.com/capsules/7628018/tree/v1>
DOI: 10.24433/CO.7719785.v1

Published version 2 on 10 December 2021, URL: <https://codeocean.com/capsule/7628018/tree/v2>
DOI: 10.24433/CO.7719785.v2

Published version 3 on 7 February 2022, URL: <https://codeocean.com/capsule/7628018/tree/v3>
DOI: 10.24433/CO.7719785.v3

Also the last version (v3) code is present with the associated timeline of the patients in excel format sheet at

IEEE Data Port site. The title of the code is “CTMC analyzing NAFLD progression (big model)” with the following DOI : (I uploaded on Saturday 5th of March 2022)
DOI: 10.21227/0h9c-9283

This is a documentation for calculation of statistical indices of the NAFLD process (big model or expanded model):

First: calculate the probability matrix by solving the Kolmogorov differential equations which will be the same as the exponentiation of the rate matrix .

Second: calculate the rate matrix in each time interval i.e. when $\Delta t = \{1, 2, 3\}$

Third: calculate the mean sojourn time and its variance and Life expectancy.

Forth: calculate the variance of stationary probability distribution.

MATLAB code for the calculation of the probability matrix at first year:

In this matlab code the rate matrix is the final estimated rate matrix, time is as chosen by the user :

```
clear
clc
close all
Qq = [-.397   .39    0    0    0    0    0    0    .007;
      .02   -.281   .25   0    0    0    0    0    .011;
      0    .05   -.365   .225   0    0    0    .047   .043;
      0    0    .041   -.538   .281   0    0    .109   .107;
      0    0    0    0   -.348   .19   0    .059   .099;
      0    0    0    0    0   -.934   .767   0    .167;
      0    0    0    0    0    0   -.421   0    .421;
      0    0    0    0    0    0    0   -.745   .745;
      0    0    0    0    0    0    0    0    0];
eig(Qq);
lambda12=Qq(1,2);
lambda18=Qq(1,8);
lambda19=Qq(1,9);
mu21=Qq(2,1);
lambda23=Qq(2,3);
lambda28=Qq(2,8);
lambda29=Qq(2,9);
mu32=Qq(3,2);
lambda34=Qq(3,4);
lambda38=Qq(3,8);
lambda39=Qq(3,9);
mu43=Qq(4,3);
lambda45=Qq(4,5);
lambda48=Qq(4,8);
lambda49=Qq(4,9);
lambda56=Qq(5,6);
lambda58=Qq(5,8);
lambda59=Qq(5,9);
lambda67=Qq(6,7);
lambda69=Qq(6,9);
lambda79=Qq(7,9);
lambda89=Qq(8,9);

% calculate eigenvalues of the Qq matrix
eigenValuesVector= eig(Qq);

% calculate the roots for the 4 th first polynomial in this manner
% the first 4 roots are the first 4 eigenvalues
% bur using the formula will get the roots in this order

g1=-1*Qq(1,1);
g2=-1*Qq(2,2);
g3=-1*Qq(3,3);
g4=-1*Qq(4,4);

w1= g1+g2;
w2= g1*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19);
w3=g3+g4;
```

```

w4=g3*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32);
w5=lambda23*mu32*(g1+g4);
w6=lambda23*mu32*g1*g4;

a=1;
b=w1+w3;
c=w2+w4+w1*w3-lambda23*mu32;
d=w2*w3+w1*w4-w5;
e=w2*w4-w6;

p=(8*c-3*b^2)/8;
q=(b^3-4*b*c+8*d)/8;
delta0=c^2-3*b*d+12*e;
sqrdelta0=sqrt(delta0);
delta0cubed=delta0^3;
sqrdelta0cubed=sqrt(delta0cubed);
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e;
theta=acos(delta1/(2*sqrtdelta0cubed));
m=cos(theta/3);
s=.5*sqrtdelta0cubed*(-1*p+sqrtdelta0*m);

r1=-(b/4)-s+.5*sqrtdelta0cubed*(q/s);
r2=-(b/4)-s-.5*sqrtdelta0cubed*(q/s);
r3=-(b/4)+s+.5*sqrtdelta0cubed*(q/s);
r4=-(b/4)+s-.5*sqrtdelta0cubed*(q/s);

% calculate probabilities when time t=1
t=1;

g1=-1*Qq(1,1);
g2=-1*Qq(2,2);
g3=-1*Qq(3,3);
g4=-1*Qq(4,4);
%calculate DPij*(s)-12 equations
DP11sss= 1
DP11ss=g2+g3+g4
DP11s=g2*g3 + g2*g4+ g3*g4 -lambda34*mu43 -lambda23*mu32
DP11constant=g2*g3*g4 -lambda23*mu32*g4 -lambda34*mu43*g2
DP11col=[DP11sss ;DP11ss ;DP11s ;DP11constant ]
% DP12
DP13sss= 0;
DP12ss=lambda12;
DP12s=lambda12*(g3+g4);
DP12constant=lambda12*g3*g4-lambda12*lambda34*mu43;
DP12col=[DP13sss ;DP12ss ;DP12s ;DP12constant ]
% DP13
DP13sss= 0;
DP13ss=0;
DP13s=lambda12*lambda23;
DP13constant=lambda12*lambda23*g4;
DP13col=[DP13sss ;DP13ss ;DP13s ;DP13constant ]
% DP14
DP14sss= 0;

```

```

DP14sss=0;
DP14s=0;
DP14constant=lambda12*lambda23*lambda34;
DP14col=[DP14sss ;DP14ss ;DP14s ;DP14constant ]
% DP21
DP21sss= 0;
DP21ss=mu21;
DP21s=mu21*(g3+g4);
DP21constant=mu21*g3*g4-mu21*lambda34*mu43;
DP21col=[DP21sss ;DP21ss ;DP21s ;DP21constant ]
% DP22
DP22sss= 1;
DP22ss=g1+g3+g4;
DP22s=g1*g3+g1*g4+g3*g4-lambda34*mu43;
DP22constant=g1*g3*g4-lambda34*mu43*g1;
DP22col=[DP22sss ;DP22ss ;DP22s ;DP22constant ]
% DP23
DP23sss= 0;
DP23ss=lambda23;
DP23s=lambda23*(g4+g1);
DP23constant=lambda23*g4*g1;
DP23col=[DP23sss ;DP23ss ;DP23s ;DP23constant ]
% DP24
DP24sss= 0;
DP24ss=0;
DP24s=lambda23*lambda34;
DP24constant=lambda23*lambda34*g1;
DP24col=[DP24sss ;DP24ss ;DP24s ;DP24constant ]
% DP31
DP31sss= 0;
DP31ss=0;
DP31s=mu21*mu32;
DP31constant=mu21*mu32*g4;
DP31col=[DP31sss ;DP31ss ;DP31s ;DP31constant ]
% DP32
DP32sss= 0;
DP32ss=mu32;
DP32s=mu32*(g1+g4);
DP32constant=mu32*g1*g4;
DP32col=[DP32sss ;DP32ss ;DP32s ;DP32constant ]
% DP33
DP33sss= 1;
DP33ss=g1+g2+g4;
DP33s=g1*g2+g1*g4+g2*g4-mu21*lambda12;
DP33constant=g1*g2*g4-mu21*lambda12*g4;
DP33col=[DP33sss ;DP33ss ;DP33s ;DP33constant ]
% DP34
DP34sss= 0;
DP34ss=lambda34;
DP34s=lambda34*(g1+g2);
DP34constant=lambda34*g1*g2-lambda12*lambda34*mu21;
DP34col=[DP34sss ;DP34ss ;DP34s ;DP34constant ]
% DP41
DP41sss=0;
DP41ss=0;
DP41s=0;
DP41constant=mu21*mu32*mu43;

```

```

DP41col=[DP41sss ;DP41ss ;DP41s ;DP41constant ]
% DP42
DP42sss=0;
DP42ss=0;
DP42s= mu32*mu43;
DP42constant= mu32*mu43*g1;
DP42col=[DP42sss ;DP42ss ;DP42s ;DP42constant ]
% DP43
DP43sss=0;
DP43ss=mu43;
DP43s= mu43*(g1+g2);
DP43constant= g1*g2*mu43-lambda12*mu21*mu43;
DP43col=[DP43sss ;DP43ss ;DP43s ;DP43constant ]
% DP44
DP44sss=1;
DP44ss=g1+g2+g3;
DP44s= g1*g2+g2*g3+g1*g3-lambda23*mu32-mu21*lambda12;
DP44constant= g1*g2*g3-lambda23*mu32*g1-mu21*lambda12*g3;
DP44col=[DP44sss ;DP44ss ;DP44s ;DP44constant ]
% construct K matrix
% eigvalues= [ -.4609  -.5898  -.1719  -.3584  ];
% r1=eigvalues(1,1);
% r2=eigvalues(1,2);
% r3=eigvalues(1,3);
% r4=eigvalues(1,4);

krow1=[1 1 1 1 ];
%krow2
krow2Element1=-1*(r2+r3+r4);
krow2Element2=-1*(r1+r3+r4);
krow2Element3=-1*(r1+r2+r4);
krow2Element4=-1*(r1+r2+r3);
krow2=[krow2Element1 krow2Element2 krow2Element3 krow2Element4]
%krow3
krow3Element1=(r2*r3+r3*r4+r2*r4);
krow3Element2=(r1*r3+r3*r4+r1*r4);
krow3Element3=(r1*r4+r1*r2+r2*r4);
krow3Element4=(r1*r3+r2*r1+r2*r3);
krow3=[krow3Element1 krow3Element2 krow3Element3 krow3Element4]
%krow4
krow4Element1=-1*(r2*r3*r4);
krow4Element2=-1*(r1*r3*r4);
krow4Element3=-1*(r1*r2*r4);
krow4Element4=-1*(r1*r2*r3);
krow4=[krow4Element1 krow4Element2 krow4Element3 krow4Element4]
K=[krow1;krow2;krow3;krow4];
Kin=inv(K)
%calculate Aij,Bij,Cij,Dij for each P11,P12,P13,P14
coeffP11= Kin*DP11col
coeffP12= Kin*DP12col
coeffP13=Kin*DP13col
coeffP14=Kin*DP14col
%calculate Aij,Bij,Cij,Dij for each P21,P22,P23,P24
coeffP21=Kin*DP21col
coeffP22=Kin*DP22col
coeffP23=Kin*DP23col
coeffP24=Kin*DP24col

```

```

%calculate Aij,Bij,Cij,Di for each P31,P32,P33,P34
coeffP31=Kin*DP31col
coeffP32=Kin*DP32col
coeffP33=Kin*DP33col
coeffP34=Kin*DP34col
%calculate Aij,Bij,Cij,Di for each P41,P42,P43,P44
coeffP41=Kin*DP41col
coeffP42=Kin*DP42col
coeffP43=Kin*DP43col
coeffP44=Kin*DP44col
m = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) ]
%calculate first 4 probabilities for the first row :
P11=m*coeffP11
P12=m*coeffP12
P13=m*coeffP13
P14=m*coeffP14
%calculate first 4 probabilities for the second row :
P21=m*coeffP21
P22=m*coeffP22
P23=m*coeffP23
P24=m*coeffP24
%calculate first 4 probabilities for the third row :
P31=m*coeffP31
P32=m*coeffP32
P33=m*coeffP33
P34=m*coeffP34
%calculate first 4 probabilities for the fourth row :
P41=m*coeffP41
P42=m*coeffP42
P43=m*coeffP43
P44=m*coeffP44

%calculate P15:

fifthGs=lambda45.*coeffP14;
G1=fifthGs(1,1)
G2=fifthGs(2,1)
G3=fifthGs(3,1)
G4=fifthGs(4,1)

w=lambda56+lambda58+lambda59;
F1=G1/(w+r1)
F2=G2/(w+r2)
F3=G3/(w+r3)
F4=G4/(w+r4)
F5=-1*(F1+F2+F3+F4)

m15 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) ];

F1To5vector=[F1; F2; F3; F4; F5];
P15=m15*F1To5vector
%calculate P25:

fifthHs=lambda45.*coeffP24;

```

```

H1=fifthHs(1,1);
H2=fifthHs(2,1);
H3=fifthHs(3,1);
H4=fifthHs(4,1);

w=lambda56+lambda58+lambda59;
K1=H1/(w+r1);
K2=H2/(w+r2);
K3=H3/(w+r3);
K4=H4/(w+r4);
K5=-1*(K1+K2+K3+K4);

m25 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) ];

K1To5vector=[K1; K2; K3; K4; K5];
P25=m25*K1To5vector
% calculate P35:

fifthLs=lambda45.*coeffP34;
L1=fifthLs(1,1);
L2=fifthLs(2,1);
L3=fifthLs(3,1);
L4=fifthLs(4,1);

w=lambda56+lambda58+lambda59;
M1=L1/(w+r1);
M2=L2/(w+r2);
M3=L3/(w+r3);
M4=L4/(w+r4);
M5=-1*(M1+M2+M3+M4);

m35 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) ];

M1To5vector=[M1; M2; M3; M4; M5];
P35=m35*M1To5vector
%calculate P45 :

fifthYs=lambda45.*coeffP44;
Y1=fifthYs(1,1);
Y2=fifthYs(2,1);
Y3=fifthYs(3,1);
Y4=fifthYs(4,1);

w=lambda56+lambda58+lambda59;
R1=Y1/(w+r1);
R2=Y2/(w+r2);
R3=Y3/(w+r3);
R4=Y4/(w+r4);
R5=-1*(R1+R2+R3+R4);

m45 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) ];

```

```

R1To5vector=[R1; R2; R3; R4; R5];
P45=m45*R1To5vector
%calculate P16

u=lambda67+lambda69;

F1To5vector=[F1; F2; F3; F4; F5];
sithGs=lambda56.*F1To5vector;
G5=sithGs(1,1)
G6=sithGs(2,1)
G7=sithGs(3,1)
G8=sithGs(4,1)
G9=sithGs(5,1)

F6=G5/(u+r1)
F7=G6/(u+r2)
F8=G7/(u+r3)
F9=G8/(u+r4)
F10=G9/(u-w)
F11=-1*(F6+F7+F8+F9+F10)

m16 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t)];
F6To11vector=[F6;F7;F8;F9;F10;F11];
P16=m16*F6To11vector

%calculate P26:
K1To5vector=[K1; K2; K3; K4; K5];
sithHs=lambda56.*K1To5vector;
H5=sithHs(1,1);
H6=sithHs(2,1);
H7=sithHs(3,1);
H8=sithHs(4,1);
H9=sithHs(5,1);

K6=H5/(u+r1);
K7=H6/(u+r2);
K8=H7/(u+r3);
K9=H8/(u+r4);
K10=H9/(u-w);
K11=-1*(K6+K7+K8+K9+K10);

m26 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t)];
K6To11vector=[K6;K7;K8;K9;K10;K11];
P26=m26*K6To11vector

%calculate P36:
M1To5vector=[M1; M2; M3; M4; M5];
sithLs=lambda56.*M1To5vector;
L5=sithLs(1,1);
L6=sithLs(2,1);
L7=sithLs(3,1);
L8=sithLs(4,1);
L9=sithLs(5,1);

```



```

M6=L5/(u+r1);
M7=L6/(u+r2);
M8=L7/(u+r3);
M9=L8/(u+r4);
M10=L9/(u-w);
M11=-1*(M6+M7+M8+M9+M10);

m36 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t)];
M6To11vector=[M6;M7;M8;M9;M10;M11];
P36=m36*M6To11vector

%calculate P46:
R1To5vector=[R1; R2; R3; R4; R5];
sithYs=lambda56.*R1To5vector;
Y5=sithYs(1,1);
Y6=sithYs(2,1);
Y7=sithYs(3,1);
Y8=sithYs(4,1);
Y9=sithYs(5,1);

R6=Y5/(u+r1);
R7=Y6/(u+r2);
R8=Y7/(u+r3);
R9=Y8/(u+r4);
R10=Y9/(u-w);
R11=-1*(R6+R7+R8+R9+R10);

m46 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t)];
R6To11vector=[R6;R7;R8;R9;R10;R11];
P46=m46*R6To11vector

%calculate P17:
F6To11vector=[F6;F7;F8;F9;F10;F11];

seventhGs=lambda67.*F6To11vector;
G10=seventhGs(1,1)
G11=seventhGs(2,1)
G12=seventhGs(3,1)
G13=seventhGs(4,1)
G14=seventhGs(5,1)
G15=seventhGs(6,1)

F12=G10/(lambda79+r1)
F13=G11/(lambda79+r2)
F14=G12/(lambda79+r3)
F15=G13/(lambda79+r4)
F16=G14/(lambda79-w)
F17=G15/(lambda79-u)
F18=-1*(F12+F13+F14+F15+F16+F17)

m17 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t)];
F12To18vector=[F12;F13;F14;F15;F16;F17;F18];
P17=m17*F12To18vector

```

```

%calculate P27:
K6To11vector=[K6;K7;K8;K9;K10;K11];

seventhHs=lambda67.*K6To11vector;
H10=seventhHs(1,1);
H11=seventhHs(2,1);
H12=seventhHs(3,1);
H13=seventhHs(4,1);
H14=seventhHs(5,1);
H15=seventhHs(6,1);

K12=H10/(lambda79+r1);
K13=H11/(lambda79+r2);
K14=H12/(lambda79+r3);
K15=H13/(lambda79+r4);
K16=H14/(lambda79-w);
K17=H15/(lambda79-u);
K18=-1*(K12+K13+K14+K15+K16+K17);

m27 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) ];
K12To18vector=[K12;K13;K14;K15;K16;K17;K18];
P27=m27*K12To18vector

%calculate P37:
M6To11vector=[M6;M7;M8;M9;M10;M11];

seventhLs=lambda67.*M6To11vector;
L10=seventhLs(1,1);
L11=seventhLs(2,1);
L12=seventhLs(3,1);
L13=seventhLs(4,1);
L14=seventhLs(5,1);
L15=seventhLs(6,1);

M12=L10/(lambda79+r1);
M13=L11/(lambda79+r2);
M14=L12/(lambda79+r3);
M15=L13/(lambda79+r4);
M16=L14/(lambda79-w);
M17=L15/(lambda79-u);
M18=-1*(M12+M13+M14+M15+M16+M17);

m37 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) ];
M12To18vector=[M12;M13;M14;M15;M16;M17;M18];
P37=m37*M12To18vector

%calculate P47:
R6To11vector=[R6;R7;R8;R9;R10;R11];

seventhYs=lambda67.*R6To11vector;
Y10=seventhYs(1,1);
Y11=seventhYs(2,1);
Y12=seventhYs(3,1);

```

```

Y13=seventhYs(4,1);
Y14=seventhYs(5,1);
Y15=seventhYs(6,1);

R12=Y10/(lambda79+r1);
R13=Y11/(lambda79+r2);
R14=Y12/(lambda79+r3);
R15=Y13/(lambda79+r4);
R16=Y14/(lambda79-w);
R17=Y15/(lambda79-u);
R18=-1*(R12+R13+R14+R15+R16+R17);

m47 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) ];
R12To18vector=[R12;R13;R14;R15;R16;R17;R18];
P47=m47*R12To18vector

%calculate P18:
% define Aij,Bij,Cij,Dij for the first 4 probabilities in first 4 rows :
A11=coeffP11(1,1);
B11=coeffP11(2,1);
C11=coeffP11(3,1);
D11=coeffP11(4,1);

A12=coeffP12(1,1);
B12=coeffP12(2,1);
C12=coeffP12(3,1);
D12=coeffP12(4,1);

A13=coeffP13(1,1);
B13=coeffP13(2,1);
C13=coeffP13(3,1);
D13=coeffP13(4,1);

A14=coeffP14(1,1);
B14=coeffP14(2,1);
C14=coeffP14(3,1);
D14=coeffP14(4,1);

A21=coeffP21(1,1);
B21=coeffP21(2,1);
C21=coeffP21(3,1);
D21=coeffP21(4,1);

A22=coeffP22(1,1);
B22=coeffP22(2,1);
C22=coeffP22(3,1);
D22=coeffP22(4,1);

A23=coeffP23(1,1);
B23=coeffP23(2,1);
C23=coeffP23(3,1);
D23=coeffP23(4,1);

A24=coeffP24(1,1);

```

```

B24=coeffP24(2,1);
C24=coeffP24(3,1);
D24=coeffP24(4,1);

A31=coeffP31(1,1);
B31=coeffP31(2,1);
C31=coeffP31(3,1);
D31=coeffP31(4,1);

A32=coeffP32(1,1);
B32=coeffP32(2,1);
C32=coeffP32(3,1);
D32=coeffP32(4,1);

A33=coeffP33(1,1);
B33=coeffP33(2,1);
C33=coeffP33(3,1);
D33=coeffP33(4,1);

A34=coeffP34(1,1);
B34=coeffP34(2,1);
C34=coeffP34(3,1);
D34=coeffP34(4,1);

A41=coeffP41(1,1);
B41=coeffP41(2,1);
C41=coeffP41(3,1);
D41=coeffP41(4,1);

A42=coeffP42(1,1);
B42=coeffP42(2,1);
C42=coeffP42(3,1);
D42=coeffP42(4,1);

A43=coeffP43(1,1);
B43=coeffP43(2,1);
C43=coeffP43(3,1);
D43=coeffP43(4,1);

A44=coeffP44(1,1);
B44=coeffP44(2,1);
C44=coeffP44(3,1);
D44=coeffP44(4,1);

% F1To5vector=[F1; F2; F3; F4; F5]

lambda8vector=[lambda18; lambda28; lambda38; lambda48 ;lambda58];

g16=[A11 A12 A13 A14 F1];
g17=[B11 B12 B13 B14 F2];
g18=[C11 C12 C13 C14 F3];
g19=[D11 D12 D13 D14 F4];

G16=g16*lambda8vector

```

```

G17=g17*lambda8vector
G18=g18*lambda8vector
G19=g19*lambda8vector
G20=lambda58*F5

F19=G16/(lambda89+r1)
F20=G17/(lambda89+r2)
F21=G18/(lambda89+r3)
F22=G19/(lambda89+r4)
F23=G20/(lambda89-w)
F24=-1*(F19+F20+F21+F22+F23)

m18 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-lambda89*t)];
F19To24vector=[F19;F20;F21;F22;F23;F24];
P18=m18*F19To24vector

% calculate P28 :
% K1To5vector=[K1; K2; K3; K4; K5]

h16=[A21 A22 A23 A24 K1];
h17=[B21 B22 B23 B24 K2];
h18=[C21 C22 C23 C24 K3];
h19=[D21 D22 D23 D24 K4];

H16=h16*lambda8vector;
H17=h17*lambda8vector;
H18=h18*lambda8vector;
H19=h19*lambda8vector;
H20=lambda58*K5;

K19=H16/(lambda89+r1);
K20=H17/(lambda89+r2);
K21=H18/(lambda89+r3);
K22=H19/(lambda89+r4);
K23=H20/(lambda89-w);
K24=-1*(K19+K20+K21+K22+K23);

m28 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-lambda89*t)];
K19To24vector=[K19;K20;K21;K22;K23;K24];
P28=m28*K19To24vector

% calculate P38 :
% M1To5vector=[M1; M2; M3; M4; M5]

l16=[A31 A32 A33 A34 M1];
l17=[B31 B32 B33 B34 M2];
l18=[C31 C32 C33 C34 M3];
l19=[D31 D32 D33 D34 M4];

L16=l16*lambda8vector;
L17=l17*lambda8vector;
L18=l18*lambda8vector;
L19=l19*lambda8vector;
L20=lambda58*M5;

```

```

M19=L16/(lambda89+r1);
M20=L17/(lambda89+r2);
M21=L18/(lambda89+r3);
M22=L19/(lambda89+r4);
M23=L20/(lambda89-w);
M24=-1*(M19+M20+M21+M22+M23);

m38 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-lambda89*t)];
M19To24vector=[M19;M20;M21;M22;M23;M24];
P38=m38*M19To24vector

% calculate P48 :
% R1To5vector=[R1; R2; R3; R4; R5]

y16=[A41 A42 A43 A44 R1];
y17=[B41 B42 B43 B44 R2];
y18=[C41 C42 C43 C44 R3];
y19=[D41 D42 D43 D44 R4];

Y16=y16*lambda8vector;
Y17=y17*lambda8vector;
Y18=y18*lambda8vector;
Y19=y19*lambda8vector;
Y20=lambda58*R5;

R19=Y16/(lambda89+r1);
R20=Y17/(lambda89+r2);
R21=Y18/(lambda89+r3);
R22=Y19/(lambda89+r4);
R23=Y20/(lambda89-w);
R24=-1*(R19+R20+R21+R22+R23);

m48 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-lambda89*t)];
R19To24vector=[R19;R20;R21;R22;R23;R24];
P48=m48*R19To24vector

%calculate P19:

lambda9vector=[lambda19; lambda29; lambda39; lambda49 ;lambda59 ;lambda69;
lambda79; lambda89 ];

g21=[A11 A12 A13 A14 F1 F6 F12 F19];
g22=[B11 B12 B13 B14 F2 F7 F13 F20];
g23=[C11 C12 C13 C14 F3 F8 F14 F21];
g24=[D11 D12 D13 D14 F4 F9 F15 F22];
g25=[F5 F10 F16 F23];
g26=[F11 F17 ];
g27=F18;
g28=F24;

G21=g21*lambda9vector
G22=g22*lambda9vector
G23=g23*lambda9vector
G24=g24*lambda9vector

```

```

G25=g25*[lambda59; lambda69; lambda79; lambda89]
G26=g26*[lambda69; lambda79]
G27=g27*lambda79
G28=g28*lambda89

F25=G21/r1
F26=G22/r2
F27=G23/r3
F28=G24/r4
F29=-1*(G25/w)
F30=-1*(G26/u)
F31=-1*(G27/lambda79)
F32=-1*(G28/lambda89)
F33=-1*(F25+F26+F27+F28)-1*(F29+F30+F31+F32)

m19 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) exp(-lambda89*t) exp(0) ];
F25To33vector=[F25;F26;F27;F28;F29;F30;F31;F32;F33 ];
P19=m19*F25To33vector

% calculate P29:

h21=[A21 A22 A23 A24 K1 K6 K12 K19];
h22=[B21 B22 B23 B24 K2 K7 K13 K20];
h23=[C21 C22 C23 C24 K3 K8 K14 K21];
h24=[D21 D22 D23 D24 K4 K9 K15 K22];
h25=[K5 K10 K16 K23];
h26=[K11 K17 ];
h27=K18;
h28=K24;

H21=h21*lambda9vector;
H22=h22*lambda9vector;
H23=h23*lambda9vector;
H24=h24*lambda9vector;
H25=h25*[lambda59; lambda69; lambda79; lambda89];
H26=h26*[lambda69; lambda79];
H27=h27*lambda79;
H28=h28*lambda89;

K25=H21/r1;
K26=H22/r2;
K27=H23/r3;
K28=H24/r4;
K29=-1*(H25/w);
K30=-1*(H26/u);
K31=-1*(H27/lambda79);
K32=-1*(H28/lambda89);
K33=-1*(K25+K26+K27+K28)-1*(K29+K30+K31+K32);

m29 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) exp(-lambda89*t) exp(0) ];
K25To33vector=[K25;K26;K27;K28;K29;K30;K31;K32;K33 ];
P29=m29*K25To33vector

```

```

%calculate P39
L21=[A31 A32 A33 A34 M1 M6 M12 M19];
L22=[B31 B32 B33 B34 M2 M7 M13 M20];
L23=[C31 C32 C33 C34 M3 M8 M14 M21];
L24=[D31 D32 D33 D34 M4 M9 M15 M22];
L25=[M5 M10 M16 M23];
L26=[M11 M17 ];
L27=M18;
L28=M24;

L21=L21*lambda9vector;
L22=L22*lambda9vector;
L23=L23*lambda9vector;
L24=L24*lambda9vector;
L25=L25*[lambda59; lambda69; lambda79; lambda89];
L26=L26*[lambda69; lambda79];
L27=L27*lambda79;
L28=L28*lambda89;

M25=L21/r1;
M26=L22/r2;
M27=L23/r3;
M28=L24/r4;
M29=-1*(L25/w);
M30=-1*(L26/u);
M31=-1*(L27/lambda79);
M32=-1*(L28/lambda89);
M33=-1*(M25+M26+M27+M28)-1*(M29+M30+M31+M32);

m39 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) exp(-lambda89*t) exp(0) ];
M25To33vector=[M25;M26;M27;M28;M29;M30;M31;M32;M33 ];
P39=m39*M25To33vector

%calculate P49:
Y21=[A41 A42 A43 A44 R1 R6 R12 R19];
Y22=[B41 B42 B43 B44 R2 R7 R13 R20];
Y23=[C41 C42 C43 C44 R3 R8 R14 R21];
Y24=[D41 D42 D43 D44 R4 R9 R15 R22];
Y25=[R5 R10 R16 R23];
Y26=[R11 R17 ];
Y27=R18;
Y28=R24;

Y21=Y21*lambda9vector;
Y22=Y22*lambda9vector;
Y23=Y23*lambda9vector;
Y24=Y24*lambda9vector;
Y25=Y25*[lambda59; lambda69; lambda79; lambda89];
Y26=Y26*[lambda69; lambda79];
Y27=Y27*lambda79;
Y28=Y28*lambda89;

R25=Y21/r1;
R26=Y22/r2;
R27=Y23/r3;

```



```

R28=Y24/r4;
R29=-1*(Y25/w);
R30=-1*(Y26/u);
R31=-1*(Y27/lambda79);
R32=-1*(Y28/lambda89);
R33=-1*(R25+R26+R27+R28)-1*(R29+R30+R31+R32);

m49 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(-w*t) exp(-u*t) exp(-
lambda79*t) exp(-lambda89*t) exp(0) ];
R25To33vector=[R25;R26;R27;R28;R29;R30;R31;R32;R33 ];
P49=m49*R25To33vector

Sum1=P11+P12+P13+P14+P15+P16+P17+P18+P19
Sum2=P21+P22+P23+P24+P25+P26+P27+P28+P29
Sum3=P31+P32+P33+P34+P35+P36+P37+P38+P39
Sum4=P41+P42+P43+P44+P45+P46+P47+P48+P49

% calculate probabilities of the 5th row:
% calculate P55

P55=exp(-w*t)

% calculate P56:
V1=lambda56/(u-w)
P56=V1*(exp(-w*t)-exp(-u*t))

% calculate P57
V2=(lambda67*V1)/(lambda79-w)
V3=-1*(lambda67*V1)/(lambda79-u)
V4=-1*(V2+V3);
m57=[exp(-w*t) exp(-u*t) exp(-lambda79*t) ]
V2ToV4vector=[V2;V3;V4]
P57=m57*V2ToV4vector

% calculate P58
V5=lambda58/(lambda89-w)
P58=V5*(exp(-w*t)-exp(-lambda89*t))

% calculate P59
V6=lambda59+lambda69*V1+lambda79*V2+lambda89*V5
V7=lambda79*V3-lambda69*V1
V8=lambda79*V4
V9=-lambda89*V5
V10=-V6/w
V11=-V7/u
V12=-V8/lambda79
V13=-V9/lambda89
V14=-1*(V10+V11+V12+V13)
m59=[exp(-w*t) exp(-u*t) exp(-lambda79*t) exp(-lambda89*t) exp(0) ];
V10ToV14vector=[V10; V11; V12; V13; V14];
P59=m59*V10ToV14vector

Sum5=P55+P56+P57+P58+P59;

% calculate probability of 6th row

```

```

% calculate P66
P66=exp(-u*t)

% calculate P67:
Z1=lambda67/(lambda79-u)
P67=Z1*(exp(-u*t)-exp(-lambda79*t))

% calculate P69 :
Z2=-1*(lambda69+lambda79*Z1)/u
Z3=Z1;
m69=[exp(-u*t) exp(-lambda79*t) exp(0) exp(0)];
Z2ToZ3vector=[Z2;Z3;-Z2;-Z3];
P69=m69*Z2ToZ3vector

Sum6=P66+P67+P69;

%calculate 7 th row
% calculate P77
P77=exp(-lambda79*t);
P79=1-P77;

%calculate 8 th row
P88=exp(-lambda89*t);
P89=1-P88

Sum5=P55+P56+P57+P58+P59
Sum6=P66+P67+P69
Sum7=P77+P79
Sum8=P88+P89

% calculate P's in each row
P1vector=[P11 P12 P13 P14 P15 P16 P17 P18 P19];
P2vector=[P21 P22 P23 P24 P25 P26 P27 P28 P29];
P3vector=[P31 P32 P33 P34 P35 P36 P37 P38 P39];
P4vector=[P41 P42 P43 P44 P45 P46 P47 P48 P49];
P5vector=[0 0 0 0 P55 P56 P57 P58 P59];
P6vector=[0 0 0 0 0 P66 P67 0 P69];
P7vector=[0 0 0 0 0 0 P77 0 P79];
P8vector=[0 0 0 0 0 0 0 P88 P89];
P9vector=[0 0 0 0 0 0 0 0 1];
Pij=[P1vector;P2vector;P3vector;P4vector;P5vector;P6vector;P7vector;P8vector;
P9vector]
initialNumOfPatients=[3100 1100 405 150 140 25 35 45 0];
initialProbOfPatient= [.62 .22 .081 .03 .028 .005 .007 .009 0];
expectedNumOfPatients = initialNumOfPatients*Pij;
oneYearProbOfPatient = initialProbOfPatient*Pij;
sum(Pij');
sum(oneYearProbOfPatient');
sum(expectedNumOfPatients');

```

MATLAB code for calculation of the rate matrix at first interval:

```

clear
clc
close all

Qq1=[-.397    .39    0    0    0    0    0    0    .007;
      .02   -.28   .25    0    0    0    0    0    .01;
      0     .05  -.36   .22    0    0    0    .05   .04 ;
      0     0   .05  -.53   .28    0    0    .11   .09;
      0     0    0    0   -.33   .18    0    .06   .09;
      0     0    0    0    0   -.9     .8    0    .1 ;
      0     0    0    0    0    0   -.41    0   .41;
      0     0    0    0    0    0    0   -.75   .75 ;
      0     0    0    0    0    0    0    0    0];

lambda12=Qq1(1,2);
lambda18=Qq1(1,8);
lambda19=Qq1(1,9);
mu21=Qq1(2,1);
lambda23=Qq1(2,3);
lambda28=Qq1(2,8);
lambda29=Qq1(2,9);
mu32=Qq1(3,2);
lambda34=Qq1(3,4);
lambda38=Qq1(3,8);
lambda39=Qq1(3,9);
mu43=Qq1(4,3);
lambda45=Qq1(4,5);
lambda48=Qq1(4,8);
lambda49=Qq1(4,9);
lambda56=Qq1(5,6);
lambda58=Qq1(5,8);
lambda59=Qq1(5,9);
lambda67=Qq1(6,7);
lambda69=Qq1(6,9);
lambda79=Qq1(7,9);
lambda89=Qq1(8,9);

% calculate eigenvalues of the Qq matrix
% eigenValuesVector= eig(Qq1);

% calculate 4 th polynomial
g1=-1*Qq1(1,1);
g2=-1*Qq1(2,2);
g3=-1*Qq1(3,3);
g4=-1*Qq1(4,4);

w1= g1+g2;
w2= g1*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19);
w3=g3+g4;
w4=g3*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32);
w5=lambda23*mu32*(g1+g4);
w6=lambda23*mu32*g1*g4;

a=1;
b=w1+w3;
c=w2+w4+w1*w3-lambda23*mu32;

```

```

d=w2*w3+w1*w4-w5;
e=w2*w4-w6;

p=(8*c-3*b^2)/8;
q=(b^3-4*b*c+8*d)/8;
delta0=c^2-3*b*d+12*e;
sqrdelta0=sqrt(delta0);
delta0cubed=delta0^3;
sqrdelta0cubed=sqrt(delta0cubed);
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e;
theta=acos(delta1/(2*sqrt(delta0^3)));
m=cos(theta/3);
s=.5*sqrt(2/3*(-1*p+sqrdelta0*m));

% differentiate each b,c,d,e, with respect to each rate
%(diff [B], [C], [D], [E] w.r.t. lambda12)

diffBwrtlambda12=1;
diffCwrtlambda12=lambda23+lambda28+lambda29+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda12=(lambda23+lambda28+lambda29)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32;
diffEwrtlambda12=(lambda23+lambda28+lambda29)*(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32*(lambda45+lambda48+lambda49+mu43);

%(diff [B], [C], [D], [E] w.r.t. lambda18)

diffBwrtlambda18=1;
diffCwrtlambda18=lambda23+lambda28+lambda29+mu21+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda18=(lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32;
diffEwrtlambda18=(lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32*(lambda45+lambda48+lambda49+mu43);

%(diff [B], [C], [D], [E] w.r.t. lambda19)

diffBwrtlambda19=1;
diffCwrtlambda19=lambda23+lambda28+lambda29+mu21+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda19=(lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32;
diffEwrtlambda19=(lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-lambda23*mu32*(lambda45+lambda48+lambda49+mu43);

%(diff [B], [C], [D], [E] w.r.t. lambda23)

diffBwrtlambda23=1;

```

```

diffCwrtlambda23=lambda12+lambda18+lambda19+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43-mu32;
diffDwrtlambda23=(lambda12+lambda18+lambda19)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32)-mu32*(lambda12+lambda18+lambda19+lambda45+lambda48+lambda49+mu43);
diffEwrtlambda23=(lambda12+lambda18+lambda19)*((lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32))-mu32*(lambda12+lambda18+lambda19)*(lambda45+lambda48+lambda49+mu43);

```

```

%(diff [B], [C], [D], [E] w.r.t. lambda28)

```

```

diffBwrtlambda28=1;
diffCwrtlambda28=lambda12+lambda18+lambda19+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda28=(lambda12+lambda18+lambda19)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32);
diffEwrtlambda28=(lambda12+lambda18+lambda19)*((lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32));

```

```

%(diff [B], [C], [D], [E] w.r.t. lambda29)

```

```

diffBwrtlambda29=1;
diffCwrtlambda29=lambda12+lambda18+lambda19+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda29=(lambda12+lambda18+lambda19)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32);
diffEwrtlambda29=(lambda12+lambda18+lambda19)*((lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32));

```

```

%(diff [B], [C], [D], [E] w.r.t. mu21)

```

```

diffBwrtmu21=1;
diffCwrtmu21=lambda18+lambda19+lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43;
diffDwrtmu21=(lambda18+lambda19)*(lambda34+lambda38+lambda39+mu32+lambda45+lambda48+lambda49+mu43)+(lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32);
diffEwrtmu21=(lambda18+lambda19)*((lambda34+lambda38+lambda39+mu32)*(lambda45+lambda48+lambda49)+mu43*(lambda38+lambda39+mu32));

```

```

%(diff [B], [C], [D], [E] w.r.t. lambda34)

```

```

diffBwrtlambda34=1;
diffCwrtlambda34=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda45+lambda48+lambda49;
diffDwrtlambda34=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda45+lambda48+lambda49));

```

```
diffEwrtlambda34=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda45+lambda48+lambda49);
```

```
%(diff [B], [C], [D], [E] w.r.t. lambda38)
```

```
diffBwrtlambda38=1;
diffCwrtlambda38=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda38=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda45+lambda48+lambda49+mu43));
diffEwrtlambda38=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda45+lambda48+lambda49+mu43);
```

```
%(diff [B], [C], [D], [E] w.r.t. lambda39)
```

```
diffBwrtlambda39=1;
diffCwrtlambda39=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda45+lambda48+lambda49+mu43;
diffDwrtlambda39=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda45+lambda48+lambda49+mu43));
diffEwrtlambda39=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda45+lambda48+lambda49+mu43);
```

```
%(diff [B], [C], [D], [E] w.r.t. mu32)
```

```
diffBwrtmu32=1;
diffCwrtmu32=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda45+lambda48+lambda49+mu43-lambda23;
diffDwrtmu32=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda45+lambda48+lambda49+mu43))-lambda23*(lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21);
diffEwrtmu32=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda45+lambda48+lambda49+mu43)-lambda23*(lambda12+lambda18+lambda19)*(lambda45+lambda48+lambda49+mu43);
```

```
%(diff [B], [C], [D], [E] w.r.t. lambda45)
```

```
diffBwrtlambda45=1;
diffCwrtlambda45=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda34+lambda38+lambda39+mu32;
diffDwrtlambda45=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32))-lambda23*mu32;
diffEwrtlambda45=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda34+lambda38+lambda39+mu32)-lambda23*mu32*(lambda12+lambda18+lambda19);
```

```
%(diff [B], [C], [D], [E] w.r.t. lambda48)
```

```
diffBwrtlambda48=1;
```

```

diffCwrtlambda48=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda34+lambda38+lambda39+mu32;
diffDwrtlambda48=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32))-lambda23*mu32;
diffEwrtlambda48=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda34+lambda38+lambda39+mu32)-lambda23*mu32*(lambda12+lambda18+lambda19);

```

```

%(diff [B], [C], [D], [E] w.r.t. lambda49)

```

```

diffBwrtlambda49=1;
diffCwrtlambda49=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda34+lambda38+lambda39+mu32;
diffDwrtlambda49=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda34+lambda38+lambda39+mu32))-lambda23*mu32;
diffEwrtlambda49=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda34+lambda38+lambda39+mu32)-lambda23*mu32*(lambda12+lambda18+lambda19);

```

```

%(diff [B], [C], [D], [E] w.r.t. mu43)

```

```

diffBwrtmu43=1;
diffCwrtmu43=lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21+lambda38+lambda39+mu32;
diffDwrtmu43=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))+((lambda12+lambda18+lambda19+lambda23+lambda28+lambda29+mu21)*(lambda38+lambda39+mu32))-lambda23*mu32;
diffEwrtmu43=((lambda12+lambda18+lambda19)*(lambda23+lambda28+lambda29)+mu21*(lambda18+lambda19))*(lambda38+lambda39+mu32)-lambda23*mu32*(lambda12+lambda18+lambda19);

```

```

% diff p w.r.t. lambda 12

```

```

diffPwrtlambda12=(8*diffCwrtlambda12-6*b*diffBwrtlambda12)/8;

```

```

% diff p w.r.t. lambda 18

```

```

diffPwrtlambda18=(8*diffCwrtlambda18-6*b*diffBwrtlambda18)/8;

```

```

% diff p w.r.t. lambda 19

```

```

diffPwrtlambda19=(8*diffCwrtlambda19-6*b*diffBwrtlambda19)/8;

```

```

% diff p w.r.t. lambda 23

```

```

diffPwrtlambda23=(8*diffCwrtlambda23-6*b*diffBwrtlambda23)/8;

```

```

% diff p w.r.t. lambda 28

```

```

diffPwrtlambda28=(8*diffCwrtlambda28-6*b*diffBwrtlambda28)/8;

```

```

% diff p w.r.t. lambda 29

```

```

diffPwrtlambda29=(8*diffCwrtlambda29-6*b*diffBwrtlambda29)/8;

```

```

% diff p w.r.t. mu 21

```

```

diffPwrtmu21=(8*diffCwrtmu21-6*b*diffBwrtmu21)/8;

```

```

% diff p w.r.t. lambda 34
diffPwrtlambda34=(8*diffCwrtlambda34-6*b*diffBwrtlambda34)/8;

% diff p w.r.t. lambda 38
diffPwrtlambda38=(8*diffCwrtlambda38-6*b*diffBwrtlambda38)/8;

% diff p w.r.t. lambda 39
diffPwrtlambda39=(8*diffCwrtlambda39-6*b*diffBwrtlambda39)/8;

% diff p w.r.t. mu 32
diffPwrtmu32=(8*diffCwrtmu32-6*b*diffBwrtmu32)/8;

% diff p w.r.t. lambda 45
diffPwrtlambda45=(8*diffCwrtlambda45-6*b*diffBwrtlambda45)/8;

% diff p w.r.t. lambda 48
diffPwrtlambda48=(8*diffCwrtlambda48-6*b*diffBwrtlambda48)/8;

% diff p w.r.t. lambda 49
diffPwrtlambda49=(8*diffCwrtlambda49-6*b*diffBwrtlambda49)/8;

% diff p w.r.t. mu 43
diffPwrtmu43=(8*diffCwrtmu43-6*b*diffBwrtmu43)/8;

% diff q w.r.t. lambda 12
diffQwrtlambda12=(3*b^2-4*diffBwrtlambda12*c-
4*diffCwrtlambda12*b+8*diffDwrtlambda12)/8;

% diff q w.r.t. lambda 18
diffQwrtlambda18=(3*b^2-4*diffBwrtlambda18*c-
4*diffCwrtlambda18*b+8*diffDwrtlambda18)/8;

% diff q w.r.t. lambda 19
diffQwrtlambda19=(3*b^2-4*diffBwrtlambda19*c-
4*diffCwrtlambda19*b+8*diffDwrtlambda19)/8;

% diff q w.r.t. lambda 23
diffQwrtlambda23=(3*b^2-4*diffBwrtlambda23*c-
4*diffCwrtlambda23*b+8*diffDwrtlambda23)/8;

% diff q w.r.t. lambda 28
diffQwrtlambda28=(3*b^2-4*diffBwrtlambda28*c-
4*diffCwrtlambda28*b+8*diffDwrtlambda28)/8;

% diff q w.r.t. lambda 29
diffQwrtlambda29=(3*b^2-4*diffBwrtlambda29*c-
4*diffCwrtlambda29*b+8*diffDwrtlambda29)/8 ;

% diff q w.r.t. mu 21
diffQwrtmu21=(3*b^2-4*diffBwrtmu21*c-4*diffCwrtmu21*b+8*diffDwrtmu21)/8;

% diff q w.r.t. lambda 34

```



```

diffQwrtlambda34=(3*b^2-4*diffBwrtlambda34*c-
4*diffCwrtlambda34*b+8*diffDwrtlambda34)/8;

% diff q w.r.t. lambda 38
diffQwrtlambda38=(3*b^2-4*diffBwrtlambda38*c-
4*diffCwrtlambda38*b+8*diffDwrtlambda38)/8;

% diff q w.r.t. lambda 39
diffQwrtlambda39=(3*b^2-4*diffBwrtlambda39*c-
4*diffCwrtlambda39*b+8*diffDwrtlambda39)/8;

% diff q w.r.t. mu 32
diffQwrtmu32=(3*b^2-4*diffBwrtmu32*c-4*diffCwrtmu32*b+8*diffDwrtmu32)/8;

% diff q w.r.t. lambda 45
diffQwrtlambda45=(3*b^2-4*diffBwrtlambda45*c-
4*diffCwrtlambda45*b+8*diffDwrtlambda45)/8;

% diff q w.r.t. lambda 48
diffQwrtlambda48=(3*b^2-4*diffBwrtlambda48*c-
4*diffCwrtlambda48*b+8*diffDwrtlambda48)/8;

% diff q w.r.t. lambda 49
diffQwrtlambda49=(3*b^2-4*diffBwrtlambda49*c-
4*diffCwrtlambda49*b+8*diffDwrtlambda49)/8;

% diff q w.r.t. mu 43
diffQwrtmu43=(3*b^2-4*diffBwrtmu43*c-4*diffCwrtmu43*b+8*diffDwrtmu43)/8;

% multiply diff p by -2/3
lambda12FirstTermSDiff=-1*(2/3)*diffPwrtlambda12;
lambda18FirstTermSDiff=-1*(2/3)*diffPwrtlambda18;
lambda19FirstTermSDiff=-1*(2/3)*diffPwrtlambda19;
lambda23FirstTermSDiff=-1*(2/3)*diffPwrtlambda23;
lambda28FirstTermSDiff=-1*(2/3)*diffPwrtlambda28;
lambda29FirstTermSDiff=-1*(2/3)*diffPwrtlambda29;
mu21FirstTermSDiff=-1*(2/3)*diffPwrtmu21;
lambda34FirstTermSDiff=-1*(2/3)*diffPwrtlambda34;
lambda38FirstTermSDiff=-1*(2/3)*diffPwrtlambda38;
lambda39FirstTermSDiff=-1*(2/3)*diffPwrtlambda39;
mu32FirstTermSDiff=-1*(2/3)*diffPwrtmu32;
lambda45FirstTermSDiff=-1*(2/3)*diffPwrtlambda45;
lambda48FirstTermSDiff=-1*(2/3)*diffPwrtlambda48;
lambda49FirstTermSDiff=-1*(2/3)*diffPwrtlambda49;
mu43FirstTermSDiff=-1*(2/3)*diffPwrtmu43;

% calculate Diff of square root of delta0:
% step 1 = reciprocal of sqrdelta0
%step 2 = diff delta0 w.r.t. each rate

reciprocalOfsqrdelta0=1/sqrdelta0;

% diff sqrt delta0 w.r.t. lambda 12

```

```

diffDelta0wrtlambda12=2*c*diffCwrtlambda12-3*diffBwrtlambda12*d-
3*diffDwrtlambda12*b+12*diffEwrtlambda12;
diffSqrDelta0wrtlambda12=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda12;

% diff sqrt delta0 w.r.t. lambda 18
diffDelta0wrtlambda18=2*c*diffCwrtlambda18-3*diffBwrtlambda18*d-
3*diffDwrtlambda18*b+12*diffEwrtlambda18;
diffSqrDelta0wrtlambda18=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda18;

% diff sqrt delta0 w.r.t. lambda 19
diffDelta0wrtlambda19=2*c*diffCwrtlambda19-3*diffBwrtlambda19*d-
3*diffDwrtlambda19*b+12*diffEwrtlambda19;
diffSqrDelta0wrtlambda19=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda19;

% diff sqrt delta0 w.r.t. lambda 23
diffDelta0wrtlambda23=2*c*diffCwrtlambda23-3*diffBwrtlambda23*d-
3*diffDwrtlambda23*b+12*diffEwrtlambda23;
diffSqrDelta0wrtlambda23=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda23;

% diff sqrt delta0 w.r.t. lambda 28
diffDelta0wrtlambda28=2*c*diffCwrtlambda28-3*diffBwrtlambda28*d-
3*diffDwrtlambda28*b+12*diffEwrtlambda28;
diffSqrDelta0wrtlambda28=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda28;

% diff sqrt delta0 w.r.t. lambda 29
diffDelta0wrtlambda29=2*c*diffCwrtlambda29-3*diffBwrtlambda29*d-
3*diffDwrtlambda29*b+12*diffEwrtlambda29;
diffSqrDelta0wrtlambda29=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda29;

% diff sqrt delta0 w.r.t. mu 21
diffDelta0wrtmu21=2*c*diffCwrtmu21-3*diffBwrtmu21*d-
3*diffDwrtmu21*b+12*diffEwrtmu21;
diffSqrDelta0wrtmu21=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu21;

% diff sqrt delta0 w.r.t. lambda 34
diffDelta0wrtlambda34=2*c*diffCwrtlambda34-3*diffBwrtlambda34*d-
3*diffDwrtlambda34*b+12*diffEwrtlambda34;
diffSqrDelta0wrtlambda34=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda34;

% diff sqrt delta0 w.r.t. lambda 38
diffDelta0wrtlambda38=2*c*diffCwrtlambda38-3*diffBwrtlambda38*d-
3*diffDwrtlambda38*b+12*diffEwrtlambda38;
diffSqrDelta0wrtlambda38=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda38;

% diff sqrt delta0 w.r.t. lambda 39
diffDelta0wrtlambda39=2*c*diffCwrtlambda39-3*diffBwrtlambda39*d-
3*diffDwrtlambda39*b+12*diffEwrtlambda39;
diffSqrDelta0wrtlambda39=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda39;

% diff sqrt delta0 w.r.t. mu 32
diffDelta0wrtmu32=2*c*diffCwrtmu32-3*diffBwrtmu32*d-
3*diffDwrtmu32*b+12*diffEwrtmu32;
diffSqrDelta0wrtmu32=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu32;

% diff sqrt delta0 w.r.t. lambda 45

```

```

diffDelta0wrtlambda45=2*c*diffCwrtlambda45-3*diffBwrtlambda45*d-
3*diffDwrtlambda45*b+12*diffEwrtlambda45;
diffSqrDelta0wrtlambda45=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda45;

% diff sqrt delta0 w.r.t. lambda 48
diffDelta0wrtlambda48=2*c*diffCwrtlambda49-3*diffBwrtlambda48*d-
3*diffDwrtlambda48*b+12*diffEwrtlambda48;
diffSqrDelta0wrtlambda48=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda48;

% diff sqrt delta0 w.r.t. lambda 49
diffDelta0wrtlambda49=2*c*diffCwrtlambda49-3*diffBwrtlambda49*d-
3*diffDwrtlambda49*b+12*diffEwrtlambda49;
diffSqrDelta0wrtlambda49=.5*reciprocalOfsqrdelta0*diffDelta0wrtlambda49;

% diff sqrt delta0 w.r.t. mu 43
diffDelta0wrtmu43=2*c*diffCwrtmu43-3*diffBwrtmu43*d-
3*diffDwrtmu43*b+12*diffEwrtmu43;
diffSqrDelta0wrtmu43=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu43;

% calculate the second term in diff s

lambda12SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda12*m;
lambda18SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda18*m ;
lambda19SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda19*m;
lambda23SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda23*m;
lambda28SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda28*m;
lambda29SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda29*m;
mu21SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu21*m;
lambda34SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda34*m;
lambda38SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda38*m;
lambda39SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda39*m;
mu32SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu32*m;
lambda45SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda45*m;
lambda48SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda48*m;
lambda49SecondTermSDiff=(2/3)*diffSqrDelta0wrtlambda49*m;
mu43SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu43*m;

% calculate third term in diff s :
sintheta=-sin(theta/3);
sqrt(delta0);
num1=sintheta*sqrt(delta0);
den1=sqrt(1-(delta1/(2*sqrt(delta0^3)))^2);
k=(-1/9)*(num1/den1);

% calculate num2/den2:
% where num2= diffdelta1*sqrt of delta0cubed - delta1*diffsqrtdelta0cubed
% where den2=delta0cubed:
% calculate diff of delta1:
DiffOfdelta1wrtlambda12=6*c^2*diffCwrtlambda12-9*diffBwrtlambda12*c*d-
9*diffCwrtlambda12*b*d-
9*diffDwrtlambda12*b*c+54*b*e*diffBwrtlambda12+27*b^2*diffEwrtlambda12+54*d*d
iffDwrtlambda12-72*diffCwrtlambda12*e-72*c*diffEwrtlambda12;
DiffOfdelta1wrtlambda18=6*c^2*diffCwrtlambda18-9*diffBwrtlambda18*c*d-
9*diffCwrtlambda18*b*d-

```

```

9*diffDwrtlambda18*b*c+54*b*e*diffBwrtlambda18+27*b^2*diffEwrtlambda18+54*d*d
iffDwrtlambda18-72*diffCwrtlambda18*e-72*c*diffEwrtlambda18;
DiffOfdelta1wrtlambda19=6*c^2*diffCwrtlambda19-9*diffBwrtlambda19*c*d-
9*diffCwrtlambda19*b*d-
9*diffDwrtlambda19*b*c+54*b*e*diffBwrtlambda19+27*b^2*diffEwrtlambda19+54*d*d
iffDwrtlambda19-72*diffCwrtlambda19*e-72*c*diffEwrtlambda19;
DiffOfdelta1wrtlambda23=6*c^2*diffCwrtlambda23-9*diffBwrtlambda23*c*d-
9*diffCwrtlambda23*b*d-
9*diffDwrtlambda23*b*c+54*b*e*diffBwrtlambda23+27*b^2*diffEwrtlambda23+54*d*d
iffDwrtlambda23-72*diffCwrtlambda23*e-72*c*diffEwrtlambda23;
DiffOfdelta1wrtlambda28=6*c^2*diffCwrtlambda28-9*diffBwrtlambda28*c*d-
9*diffCwrtlambda28*b*d-
9*diffDwrtlambda28*b*c+54*b*e*diffBwrtlambda28+27*b^2*diffEwrtlambda28+54*d*d
iffDwrtlambda28-72*diffCwrtlambda28*e-72*c*diffEwrtlambda28;
DiffOfdelta1wrtlambda29=6*c^2*diffCwrtlambda29-9*diffBwrtlambda29*c*d-
9*diffCwrtlambda29*b*d-
9*diffDwrtlambda29*b*c+54*b*e*diffBwrtlambda29+27*b^2*diffEwrtlambda29+54*d*d
iffDwrtlambda29-72*diffCwrtlambda29*e-72*c*diffEwrtlambda29;
DiffOfdelta1wrtmu21=6*c^2*diffCwrtmu21-9*diffBwrtmu21*c*d-9*diffCwrtmu21*b*d-
9*diffDwrtmu21*b*c+54*b*e*diffBwrtmu21+27*b^2*diffEwrtmu21+54*d*d*diffDwrtmu21-
72*diffCwrtmu21*e-72*c*diffEwrtmu21;
DiffOfdelta1wrtlambda34=6*c^2*diffCwrtlambda34-9*diffBwrtlambda34*c*d-
9*diffCwrtlambda34*b*d-
9*diffDwrtlambda34*b*c+54*b*e*diffBwrtlambda34+27*b^2*diffEwrtlambda34+54*d*d
iffDwrtlambda34-72*diffCwrtlambda34*e-72*c*diffEwrtlambda34;
DiffOfdelta1wrtlambda38=6*c^2*diffCwrtlambda38-9*diffBwrtlambda38*c*d-
9*diffCwrtlambda38*b*d-
9*diffDwrtlambda38*b*c+54*b*e*diffBwrtlambda38+27*b^2*diffEwrtlambda38+54*d*d
iffDwrtlambda38-72*diffCwrtlambda38*e-72*c*diffEwrtlambda38;
DiffOfdelta1wrtlambda39=6*c^2*diffCwrtlambda39-9*diffBwrtlambda39*c*d-
9*diffCwrtlambda39*b*d-
9*diffDwrtlambda39*b*c+54*b*e*diffBwrtlambda39+27*b^2*diffEwrtlambda39+54*d*d
iffDwrtlambda39-72*diffCwrtlambda39*e-72*c*diffEwrtlambda39;
DiffOfdelta1wrtmu32=6*c^2*diffCwrtmu32-9*diffBwrtmu32*c*d-9*diffCwrtmu32*b*d-
9*diffDwrtmu32*b*c+54*b*e*diffBwrtmu32+27*b^2*diffEwrtmu32+54*d*d*diffDwrtmu32-
72*diffCwrtmu32*e-72*c*diffEwrtmu32;
DiffOfdelta1wrtlambda45=6*c^2*diffCwrtlambda45-9*diffBwrtlambda45*c*d-
9*diffCwrtlambda45*b*d-
9*diffDwrtlambda45*b*c+54*b*e*diffBwrtlambda45+27*b^2*diffEwrtlambda45+54*d*d
iffDwrtlambda45-72*diffCwrtlambda45*e-72*c*diffEwrtlambda45;
DiffOfdelta1wrtlambda48=6*c^2*diffCwrtlambda48-9*diffBwrtlambda48*c*d-
9*diffCwrtlambda48*b*d-
9*diffDwrtlambda48*b*c+54*b*e*diffBwrtlambda48+27*b^2*diffEwrtlambda48+54*d*d
iffDwrtlambda48-72*diffCwrtlambda48*e-72*c*diffEwrtlambda48;
DiffOfdelta1wrtlambda49=6*c^2*diffCwrtlambda49-9*diffBwrtlambda49*c*d-
9*diffCwrtlambda49*b*d-
9*diffDwrtlambda49*b*c+54*b*e*diffBwrtlambda49+27*b^2*diffEwrtlambda49+54*d*d
iffDwrtlambda49-72*diffCwrtlambda49*e-72*c*diffEwrtlambda49;
DiffOfdelta1wrtmu43=6*c^2*diffCwrtmu43-9*diffBwrtmu43*c*d-9*diffCwrtmu43*b*d-
9*diffDwrtmu43*b*c+54*b*e*diffBwrtmu43+27*b^2*diffEwrtmu43+54*d*d*diffDwrtmu43-
72*diffCwrtmu43*e-72*c*diffEwrtmu43;

% calculate diff of sqrt of delta0cubed :
sqrdelta0=sqrt(delta0);
DiffsqrtOfdelta0cubedwrtlambda12=(3/2)*sqrdelta0*(2*c*diffCwrtlambda12-
3*diffBwrtlambda12*d-3*diffDwrtlambda12*b+12*diffEwrtlambda12);

```

```

DiffsqrtOfdelta0cubedwrtlambda18=(3/2)*sqrdelta0*(2*c*diffCwrtlambda18-
3*diffBwrtlambda18*d-3*diffDwrtlambda18*b+12*diffEwrtlambda18);
DiffsqrtOfdelta0cubedwrtlambda19=(3/2)*sqrdelta0*(2*c*diffCwrtlambda19-
3*diffBwrtlambda19*d-3*diffDwrtlambda19*b+12*diffEwrtlambda19);
DiffsqrtOfdelta0cubedwrtlambda23=(3/2)*sqrdelta0*(2*c*diffCwrtlambda23-
3*diffBwrtlambda23*d-3*diffDwrtlambda23*b+12*diffEwrtlambda23);
DiffsqrtOfdelta0cubedwrtlambda28=(3/2)*sqrdelta0*(2*c*diffCwrtlambda28-
3*diffBwrtlambda28*d-3*diffDwrtlambda28*b+12*diffEwrtlambda28);
DiffsqrtOfdelta0cubedwrtlambda29=(3/2)*sqrdelta0*(2*c*diffCwrtlambda29-
3*diffBwrtlambda29*d-3*diffDwrtlambda29*b+12*diffEwrtlambda29);
DiffsqrtOfdelta0cubedwrtmu21=(3/2)*sqrdelta0*(2*c*diffCwrtmu21-
3*diffBwrtmu21*d-3*diffDwrtmu21*b+12*diffEwrtmu21);
DiffsqrtOfdelta0cubedwrtlambda34=(3/2)*sqrdelta0*(2*c*diffCwrtlambda34-
3*diffBwrtlambda34*d-3*diffDwrtlambda34*b+12*diffEwrtlambda34);
DiffsqrtOfdelta0cubedwrtlambda38=(3/2)*sqrdelta0*(2*c*diffCwrtlambda38-
3*diffBwrtlambda38*d-3*diffDwrtlambda38*b+12*diffEwrtlambda38);
DiffsqrtOfdelta0cubedwrtlambda39=(3/2)*sqrdelta0*(2*c*diffCwrtlambda39-
3*diffBwrtlambda39*d-3*diffDwrtlambda39*b+12*diffEwrtlambda39);
DiffsqrtOfdelta0cubedwrtmu32=(3/2)*sqrdelta0*(2*c*diffCwrtmu32-
3*diffBwrtmu32*d-3*diffDwrtmu32*b+12*diffEwrtmu32);
DiffsqrtOfdelta0cubedwrtlambda45=(3/2)*sqrdelta0*(2*c*diffCwrtlambda45-
3*diffBwrtlambda45*d-3*diffDwrtlambda45*b+12*diffEwrtlambda45);
DiffsqrtOfdelta0cubedwrtlambda48=(3/2)*sqrdelta0*(2*c*diffCwrtlambda48-
3*diffBwrtlambda48*d-3*diffDwrtlambda48*b+12*diffEwrtlambda48);
DiffsqrtOfdelta0cubedwrtlambda49=(3/2)*sqrdelta0*(2*c*diffCwrtlambda49-
3*diffBwrtlambda49*d-3*diffDwrtlambda49*b+12*diffEwrtlambda49);
DiffsqrtOfdelta0cubedwrtmu43=(3/2)*sqrdelta0*(2*c*diffCwrtmu43-
3*diffBwrtmu43*d-3*diffDwrtmu43*b+12*diffEwrtmu43);

```

```

%calculate num2 wrt each rate :
delta0cubed=delta0^3;
sqrdelta0cubed=sqrt(delta0cubed);
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e;
k;
lambda12LastTermSDiff=k*(DiffOfdelta1wrtlambda12*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda12)/delta0cubed;
lambda18LastTermSDiff=k*(DiffOfdelta1wrtlambda18*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda18)/delta0cubed;
lambda19LastTermSDiff=k*(DiffOfdelta1wrtlambda19*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda19)/delta0cubed;
lambda23LastTermSDiff=k*(DiffOfdelta1wrtlambda23*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda23)/delta0cubed;
lambda28LastTermSDiff=k*(DiffOfdelta1wrtlambda28*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda28)/delta0cubed;
lambda29LastTermSDiff=k*(DiffOfdelta1wrtlambda29*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda29)/delta0cubed;
mu21LastTermSDiff=k*(DiffOfdelta1wrtmu21*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu21)/delta0cubed;
lambda34LastTermSDiff=k*(DiffOfdelta1wrtlambda34*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda34)/delta0cubed;
lambda38LastTermSDiff=k*(DiffOfdelta1wrtlambda38*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda38)/delta0cubed;
lambda39LastTermSDiff=k*(DiffOfdelta1wrtlambda39*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda39)/delta0cubed;
mu32LastTermSDiff=k*(DiffOfdelta1wrtmu32*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu32)/delta0cubed;

```

```

lambda45LastTermSDiff=k*(DiffOfdelta1wrtlambda45*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda45)/delta0cubed;
lambda48LastTermSDiff=k*(DiffOfdelta1wrtlambda48*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda48)/delta0cubed;
lambda49LastTermSDiff=k*(DiffOfdelta1wrtlambda49*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlambda49)/delta0cubed;
mu43LastTermSDiff=k*(DiffOfdelta1wrtmu43*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu43)/delta0cubed;

% calculate diff of s w.r.t. each rate :
n=sqrt(2/3*(-1*p+sqrdelta0*m));
h=1/n;
diffSwrtlambda12=.25*h*(lambda12FirstTermSDiff+lambda12SecondTermSDiff+lambda
12LastTermSDiff);
diffSwrtlambda18=.25*h*(lambda18FirstTermSDiff+lambda18SecondTermSDiff+lambda
18LastTermSDiff);
diffSwrtlambda19=.25*h*(lambda19FirstTermSDiff+lambda19SecondTermSDiff+lambda
19LastTermSDiff);
diffSwrtlambda23=.25*h*(lambda23FirstTermSDiff+lambda23SecondTermSDiff+lambda
23LastTermSDiff);
diffSwrtlambda28=.25*h*(lambda28FirstTermSDiff+lambda28SecondTermSDiff+lambda
28LastTermSDiff);
diffSwrtlambda29=.25*h*(lambda29FirstTermSDiff+lambda29SecondTermSDiff+lambda
29LastTermSDiff);
diffSwrtmu21=.25*h*(mu21FirstTermSDiff+mu21SecondTermSDiff+mu21LastTermSDiff)
;
diffSwrtlambda34=.25*h*(lambda34FirstTermSDiff+lambda34SecondTermSDiff+lambda
34LastTermSDiff);
diffSwrtlambda38=.25*h*(lambda38FirstTermSDiff+lambda38SecondTermSDiff+lambda
38LastTermSDiff);
diffSwrtlambda39=.25*h*(lambda39FirstTermSDiff+lambda39SecondTermSDiff+lambda
39LastTermSDiff);
diffSwrtmu32=.25*h*(mu32FirstTermSDiff+mu32SecondTermSDiff+mu32LastTermSDiff)
;
diffSwrtlambda45=.25*h*(lambda45FirstTermSDiff+lambda45SecondTermSDiff+lambda
45LastTermSDiff);
diffSwrtlambda48=.25*h*(lambda48FirstTermSDiff+lambda48SecondTermSDiff+lambda
48LastTermSDiff);
diffSwrtlambda49=.25*h*(lambda49FirstTermSDiff+lambda49SecondTermSDiff+lambda
49LastTermSDiff);
diffSwrtmu43=.25*h*(mu43FirstTermSDiff+mu43SecondTermSDiff+mu43LastTermSDiff)
;

% differentiate roots( first 4 roots):

p=(8*c-3*b^2)/8;
q=(b^3-4*b*c+8*d)/8;
delta0=c^2-3*b*d+12*e;
sqrdelta0=sqrt(delta0);
delta0cubed=delta0^3;
sqrdelta0cubed=sqrt(delta0cubed);
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e;
theta=acos(delta1/(2*sqrt(delta0^3)));
m=cos(theta/3);
s=.5*sqrt(2/3*(-1*p+sqrdelta0*m));

```

```

% for root 1,2
% diff root1= -b/4 - s'+ .25 (-4s^2-2p+q/s)^-.5 * (-8ss'-2p'+ (q'*s-s'*q)/s^2)
% diff root2= -b/4 - s'- .25 (-4s^2-2p+q/s)^-.5 * (-8ss'-2p'+ (q'*s-s'*q)/s^2)
% diff root3= -b/4 + s'+ .25 (-4s^2-2p-q/s)^-.5 * (-8ss'-2p'- (q'*s-s'*q)/s^2)
% diff root4= -b/4 + s'- .25 (-4s^2-2p-q/s)^-.5 * (-8ss'-2p'- (q'*s-s'*q)/s^2)

% root 1
f=1/sqrt(-4*s^2-2*p+(q/s));
fNeg=1/sqrt(-4*s^2-2*p-(q/s));

diffroot1wrtlambda12=-(diffBwrtlambda12/4)-diffSwrtlambda12+.25*f*(-
8*s*diffSwrtlambda12-2*diffPwrtlambda12+((diffQwrtlambda12*s-
diffSwrtlambda12*q)/s^2));
diffroot1wrtlambda18=-(diffBwrtlambda18/4)-diffSwrtlambda18+.25*f*(-
8*s*diffSwrtlambda18-2*diffPwrtlambda18+((diffQwrtlambda18*s-
diffSwrtlambda18*q)/s^2));
diffroot1wrtlambda19=-(diffBwrtlambda19/4)-diffSwrtlambda19+.25*f*(-
8*s*diffSwrtlambda19-2*diffPwrtlambda19+((diffQwrtlambda19*s-
diffSwrtlambda19*q)/s^2));
diffroot1wrtlambda23=-(diffBwrtlambda23/4)-diffSwrtlambda23+.25*f*(-
8*s*diffSwrtlambda23-2*diffPwrtlambda23+((diffQwrtlambda23*s-
diffSwrtlambda23*q)/s^2));
diffroot1wrtlambda28=-(diffBwrtlambda28/4)-diffSwrtlambda28+.25*f*(-
8*s*diffSwrtlambda28-2*diffPwrtlambda28+((diffQwrtlambda28*s-
diffSwrtlambda28*q)/s^2));
diffroot1wrtlambda29=-(diffBwrtlambda29/4)-diffSwrtlambda29+.25*f*(-
8*s*diffSwrtlambda29-2*diffPwrtlambda29+((diffQwrtlambda29*s-
diffSwrtlambda29*q)/s^2));
diffroot1wrtmu21=-(diffBwrtmu21/4)-diffSwrtmu21+.25*f*(-8*s*diffSwrtmu21-
2*diffPwrtmu21+((diffQwrtmu21*s-diffSwrtmu21*q)/s^2));
diffroot1wrtlambda34=-(diffBwrtlambda34/4)-diffSwrtlambda34+.25*f*(-
8*s*diffSwrtlambda34-2*diffPwrtlambda34+((diffQwrtlambda34*s-
diffSwrtlambda34*q)/s^2));
diffroot1wrtlambda38=-(diffBwrtlambda38/4)-diffSwrtlambda38+.25*f*(-
8*s*diffSwrtlambda38-2*diffPwrtlambda38+((diffQwrtlambda38*s-
diffSwrtlambda38*q)/s^2));
diffroot1wrtlambda39=-(diffBwrtlambda39/4)-diffSwrtlambda39+.25*f*(-
8*s*diffSwrtlambda39-2*diffPwrtlambda39+((diffQwrtlambda39*s-
diffSwrtlambda39*q)/s^2));
diffroot1wrtmu32=-(diffBwrtmu32/4)-diffSwrtmu32+.25*f*(-8*s*diffSwrtmu32-
2*diffPwrtmu32+((diffQwrtmu32*s-diffSwrtmu32*q)/s^2));
diffroot1wrtlambda45=-(diffBwrtlambda45/4)-diffSwrtlambda45+.25*f*(-
8*s*diffSwrtlambda45-2*diffPwrtlambda45+((diffQwrtlambda45*s-
diffSwrtlambda45*q)/s^2));
diffroot1wrtlambda48=-(diffBwrtlambda48/4)-diffSwrtlambda48+.25*f*(-
8*s*diffSwrtlambda48-2*diffPwrtlambda48+((diffQwrtlambda48*s-
diffSwrtlambda48*q)/s^2));
diffroot1wrtlambda49=-(diffBwrtlambda49/4)-diffSwrtlambda49+.25*f*(-
8*s*diffSwrtlambda49-2*diffPwrtlambda49+((diffQwrtlambda49*s-
diffSwrtlambda49*q)/s^2));
diffroot1wrtmu43=-(diffBwrtmu43/4)-diffSwrtmu43+.25*f*(-8*s*diffSwrtmu43-
2*diffPwrtmu43+((diffQwrtmu43*s-diffSwrtmu43*q)/s^2));

% root2

```

```

diffroot2wrtlambda12=- (diffBwrtlambda12/4)-diffSwrtlambda12-.25*f*(-
8*s*diffSwrtlambda12-2*diffPwrtlambda12+((diffQwrtlambda12*s-
diffSwrtlambda12*q)/s^2));
diffroot2wrtlambda18=- (diffBwrtlambda18/4)-diffSwrtlambda18-.25*f*(-
8*s*diffSwrtlambda18-2*diffPwrtlambda18+((diffQwrtlambda18*s-
diffSwrtlambda18*q)/s^2));
diffroot2wrtlambda19=- (diffBwrtlambda19/4)-diffSwrtlambda19-.25*f*(-
8*s*diffSwrtlambda19-2*diffPwrtlambda19+((diffQwrtlambda19*s-
diffSwrtlambda19*q)/s^2));
diffroot2wrtlambda23=- (diffBwrtlambda23/4)-diffSwrtlambda23-.25*f*(-
8*s*diffSwrtlambda23-2*diffPwrtlambda23+((diffQwrtlambda23*s-
diffSwrtlambda23*q)/s^2));
diffroot2wrtlambda28=- (diffBwrtlambda28/4)-diffSwrtlambda28-.25*f*(-
8*s*diffSwrtlambda28-2*diffPwrtlambda28+((diffQwrtlambda28*s-
diffSwrtlambda28*q)/s^2));
diffroot2wrtlambda29=- (diffBwrtlambda29/4)-diffSwrtlambda29-.25*f*(-
8*s*diffSwrtlambda29-2*diffPwrtlambda29+((diffQwrtlambda29*s-
diffSwrtlambda29*q)/s^2));
diffroot2wrtmu21=- (diffBwrtmu21/4)-diffSwrtmu21-.25*f*(-8*s*diffSwrtmu21-
2*diffPwrtmu21+((diffQwrtmu21*s-diffSwrtmu21*q)/s^2));
diffroot2wrtlambda34=- (diffBwrtlambda34/4)-diffSwrtlambda34-.25*f*(-
8*s*diffSwrtlambda34-2*diffPwrtlambda34+((diffQwrtlambda34*s-
diffSwrtlambda34*q)/s^2));
diffroot2wrtlambda38=- (diffBwrtlambda38/4)-diffSwrtlambda38-.25*f*(-
8*s*diffSwrtlambda38-2*diffPwrtlambda38+((diffQwrtlambda38*s-
diffSwrtlambda38*q)/s^2));
diffroot2wrtlambda39=- (diffBwrtlambda39/4)-diffSwrtlambda39-.25*f*(-
8*s*diffSwrtlambda39-2*diffPwrtlambda39+((diffQwrtlambda39*s-
diffSwrtlambda39*q)/s^2));
diffroot2wrtmu32=- (diffBwrtmu32/4)-diffSwrtmu32-.25*f*(-8*s*diffSwrtmu32-
2*diffPwrtmu32+((diffQwrtmu32*s-diffSwrtmu32*q)/s^2));
diffroot2wrtlambda45=- (diffBwrtlambda45/4)-diffSwrtlambda45-.25*f*(-
8*s*diffSwrtlambda45-2*diffPwrtlambda45+((diffQwrtlambda45*s-
diffSwrtlambda45*q)/s^2));
diffroot2wrtlambda48=- (diffBwrtlambda48/4)-diffSwrtlambda48-.25*f*(-
8*s*diffSwrtlambda48-2*diffPwrtlambda48+((diffQwrtlambda48*s-
diffSwrtlambda48*q)/s^2));
diffroot2wrtlambda49=- (diffBwrtlambda49/4)-diffSwrtlambda49-.25*f*(-
8*s*diffSwrtlambda49-2*diffPwrtlambda49+((diffQwrtlambda49*s-
diffSwrtlambda49*q)/s^2));
diffroot2wrtmu43=- (diffBwrtmu43/4)-diffSwrtmu43-.25*f*(-8*s*diffSwrtmu43-
2*diffPwrtmu43+((diffQwrtmu43*s-diffSwrtmu43*q)/s^2));

```

% root 3 :

```

diffroot3wrtlambda12=- (diffBwrtlambda12/4)+diffSwrtlambda12+.25*fNeg*(-
8*s*diffSwrtlambda12-2*diffPwrtlambda12-((diffQwrtlambda12*s-
diffSwrtlambda12*q)/s^2));
diffroot3wrtlambda18=- (diffBwrtlambda18/4)+diffSwrtlambda18+.25*fNeg*(-
8*s*diffSwrtlambda18-2*diffPwrtlambda18-((diffQwrtlambda18*s-
diffSwrtlambda18*q)/s^2));
diffroot3wrtlambda19=- (diffBwrtlambda19/4)+diffSwrtlambda19+.25*fNeg*(-
8*s*diffSwrtlambda19-2*diffPwrtlambda19-((diffQwrtlambda19*s-
diffSwrtlambda19*q)/s^2));

```



```

diffroot3wrtlambda23=-(diffBwrtlambda23/4)+diffSwrtlambda23+.25*fNeg*(-
8*s*diffSwrtlambda23-2*diffPwrtlambda23-((diffQwrtlambda23*s-
diffSwrtlambda23*q)/s^2));
diffroot3wrtlambda28=-(diffBwrtlambda28/4)+diffSwrtlambda28+.25*fNeg*(-
8*s*diffSwrtlambda28-2*diffPwrtlambda28-((diffQwrtlambda28*s-
diffSwrtlambda28*q)/s^2));
diffroot3wrtlambda29=-(diffBwrtlambda29/4)+diffSwrtlambda29+.25*fNeg*(-
8*s*diffSwrtlambda29-2*diffPwrtlambda29-((diffQwrtlambda29*s-
diffSwrtlambda29*q)/s^2));
diffroot3wrtmu21=-(diffBwrtmu21/4)+diffSwrtmu21+.25*fNeg*(-8*s*diffSwrtmu21-
2*diffPwrtmu21-((diffQwrtmu21*s-diffSwrtmu21*q)/s^2));
diffroot3wrtlambda34=-(diffBwrtlambda34/4)+diffSwrtlambda34+.25*fNeg*(-
8*s*diffSwrtlambda34-2*diffPwrtlambda34-((diffQwrtlambda34*s-
diffSwrtlambda34*q)/s^2));
diffroot3wrtlambda38=-(diffBwrtlambda38/4)+diffSwrtlambda38+.25*fNeg*(-
8*s*diffSwrtlambda38-2*diffPwrtlambda38-((diffQwrtlambda38*s-
diffSwrtlambda38*q)/s^2));
diffroot3wrtlambda39=-(diffBwrtlambda39/4)+diffSwrtlambda39+.25*fNeg*(-
8*s*diffSwrtlambda39-2*diffPwrtlambda39-((diffQwrtlambda39*s-
diffSwrtlambda39*q)/s^2));
diffroot3wrtmu32=-(diffBwrtmu32/4)+diffSwrtmu32+.25*fNeg*(-8*s*diffSwrtmu32-
2*diffPwrtmu32-((diffQwrtmu32*s-diffSwrtmu32*q)/s^2));
diffroot3wrtlambda45=-(diffBwrtlambda45/4)+diffSwrtlambda45+.25*fNeg*(-
8*s*diffSwrtlambda45-2*diffPwrtlambda45-((diffQwrtlambda45*s-
diffSwrtlambda45*q)/s^2));
diffroot3wrtlambda48=-(diffBwrtlambda48/4)+diffSwrtlambda48+.25*fNeg*(-
8*s*diffSwrtlambda48-2*diffPwrtlambda48-((diffQwrtlambda48*s-
diffSwrtlambda48*q)/s^2));
diffroot3wrtlambda49=-(diffBwrtlambda49/4)+diffSwrtlambda49+.25*fNeg*(-
8*s*diffSwrtlambda49-2*diffPwrtlambda49-((diffQwrtlambda49*s-
diffSwrtlambda49*q)/s^2));
diffroot3wrtmu43=-(diffBwrtmu43/4)+diffSwrtmu43+.25*fNeg*(-8*s*diffSwrtmu43-
2*diffPwrtmu43-((diffQwrtmu43*s-diffSwrtmu43*q)/s^2));

```

```

% root 4 :

```

```

diffroot4wrtlambda12=-(diffBwrtlambda12/4)+diffSwrtlambda12-.25*fNeg*(-
8*s*diffSwrtlambda12-2*diffPwrtlambda12-((diffQwrtlambda12*s-
diffSwrtlambda12*q)/s^2));
diffroot4wrtlambda18=-(diffBwrtlambda18/4)+diffSwrtlambda18-.25*fNeg*(-
8*s*diffSwrtlambda18-2*diffPwrtlambda18-((diffQwrtlambda18*s-
diffSwrtlambda18*q)/s^2));
diffroot4wrtlambda19=-(diffBwrtlambda19/4)+diffSwrtlambda19-.25*fNeg*(-
8*s*diffSwrtlambda19-2*diffPwrtlambda19-((diffQwrtlambda19*s-
diffSwrtlambda19*q)/s^2));
diffroot4wrtlambda23=-(diffBwrtlambda23/4)+diffSwrtlambda23-.25*fNeg*(-
8*s*diffSwrtlambda23-2*diffPwrtlambda23-((diffQwrtlambda23*s-
diffSwrtlambda23*q)/s^2));
diffroot4wrtlambda28=-(diffBwrtlambda28/4)+diffSwrtlambda28-.25*fNeg*(-
8*s*diffSwrtlambda28-2*diffPwrtlambda28-((diffQwrtlambda28*s-
diffSwrtlambda28*q)/s^2));
diffroot4wrtlambda29=-(diffBwrtlambda29/4)+diffSwrtlambda29-.25*fNeg*(-
8*s*diffSwrtlambda29-2*diffPwrtlambda29-((diffQwrtlambda29*s-
diffSwrtlambda29*q)/s^2));
diffroot4wrtmu21=-(diffBwrtmu21/4)+diffSwrtmu21-.25*fNeg*(-8*s*diffSwrtmu21-
2*diffPwrtmu21-((diffQwrtmu21*s-diffSwrtmu21*q)/s^2));

```

```

diffroot4wrtlambda34=-(diffBwrtlambda34/4)+diffSwrtlambda34-.25*fNeg*(-
8*s*diffSwrtlambda34-2*diffPwrtlambda34-((diffQwrtlambda34*s-
diffSwrtlambda34*q)/s^2));
diffroot4wrtlambda38=-(diffBwrtlambda38/4)+diffSwrtlambda38-.25*fNeg*(-
8*s*diffSwrtlambda38-2*diffPwrtlambda38-((diffQwrtlambda38*s-
diffSwrtlambda38*q)/s^2));
diffroot4wrtlambda39=-(diffBwrtlambda39/4)+diffSwrtlambda39-.25*fNeg*(-
8*s*diffSwrtlambda39-2*diffPwrtlambda39-((diffQwrtlambda39*s-
diffSwrtlambda39*q)/s^2));
diffroot4wrtmu32=-(diffBwrtmu32/4)+diffSwrtmu32-.25*fNeg*(-8*s*diffSwrtmu32-
2*diffPwrtmu32-((diffQwrtmu32*s-diffSwrtmu32*q)/s^2));
diffroot4wrtlambda45=-(diffBwrtlambda45/4)+diffSwrtlambda45-.25*fNeg*(-
8*s*diffSwrtlambda45-2*diffPwrtlambda45-((diffQwrtlambda45*s-
diffSwrtlambda45*q)/s^2));
diffroot4wrtlambda48=-(diffBwrtlambda48/4)+diffSwrtlambda48-.25*fNeg*(-
8*s*diffSwrtlambda48-2*diffPwrtlambda48-((diffQwrtlambda48*s-
diffSwrtlambda48*q)/s^2));
diffroot4wrtlambda49=-(diffBwrtlambda49/4)+diffSwrtlambda49-.25*fNeg*(-
8*s*diffSwrtlambda49-2*diffPwrtlambda49-((diffQwrtlambda49*s-
diffSwrtlambda49*q)/s^2));
diffroot4wrtmu43=-(diffBwrtmu43/4)+diffSwrtmu43-.25*fNeg*(-8*s*diffSwrtmu43-
2*diffPwrtmu43-((diffQwrtmu43*s-diffSwrtmu43*q)/s^2));

```

```

% calculate t*exp(rooti*t)*diffOfRoot with respect to each root let's call
% rootlagain

```

```

t=1;
eigenValuesVector= eig(Qq1);
r1=-(b/4)-s+.5*sqrt(-4*s^2-2*p+(q/s));
r2=-(b/4)-s-.5*sqrt(-4*s^2-2*p+(q/s));
r3=-(b/4)+s+.5*sqrt(-4*s^2-2*p-(q/s));
r4=-(b/4)+s-.5*sqrt(-4*s^2-2*p-(q/s));

```

```

% rootlagain

```

```

rootlagainwrtlambda12=t*exp(r1*t)*diffroot1wrtlambda12;
rootlagainwrtlambda18=t*exp(r1*t)*diffroot1wrtlambda18;
rootlagainwrtlambda19=t*exp(r1*t)*diffroot1wrtlambda19;
rootlagainwrtlambda23=t*exp(r1*t)*diffroot1wrtlambda23;
rootlagainwrtlambda28=t*exp(r1*t)*diffroot1wrtlambda28;
rootlagainwrtlambda29=t*exp(r1*t)*diffroot1wrtlambda29;
rootlagainwrtmu21=t*exp(r1*t)*diffroot1wrtmu21;
rootlagainwrtlambda34=t*exp(r1*t)*diffroot1wrtlambda34;
rootlagainwrtlambda38=t*exp(r1*t)*diffroot1wrtlambda38;
rootlagainwrtlambda39=t*exp(r1*t)*diffroot1wrtlambda39;
rootlagainwrtmu32=t*exp(r1*t)*diffroot1wrtmu32;
rootlagainwrtlambda45=t*exp(r1*t)*diffroot1wrtlambda45;
rootlagainwrtlambda48=t*exp(r1*t)*diffroot1wrtlambda48;
rootlagainwrtlambda49=t*exp(r1*t)*diffroot1wrtlambda49;
rootlagainwrtmu43=t*exp(r1*t)*diffroot1wrtmu43;

```

```

% root2again

```

```

root2againwrtlambda12=t*exp(r2*t)*diffroot2wrtlambda12;
root2againwrtlambda18=t*exp(r2*t)*diffroot2wrtlambda18;
root2againwrtlambda19=t*exp(r2*t)*diffroot2wrtlambda19;
root2againwrtlambda23=t*exp(r2*t)*diffroot2wrtlambda23;
root2againwrtlambda28=t*exp(r2*t)*diffroot2wrtlambda28;
root2againwrtlambda29=t*exp(r2*t)*diffroot2wrtlambda29;

```

```

root2againwrtmu21=t*exp(r2*t)*diffroot2wrtmu21;
root2againwrtlambda34=t*exp(r2*t)*diffroot2wrtlambda34;
root2againwrtlambda38=t*exp(r2*t)*diffroot2wrtlambda38;
root2againwrtlambda39=t*exp(r2*t)*diffroot2wrtlambda39;
root2againwrtmu32=t*exp(r2*t)*diffroot2wrtmu32;
root2againwrtlambda45=t*exp(r2*t)*diffroot2wrtlambda45;
root2againwrtlambda48=t*exp(r2*t)*diffroot2wrtlambda48;
root2againwrtlambda49=t*exp(r2*t)*diffroot2wrtlambda49;
root2againwrtmu43=t*exp(r2*t)*diffroot2wrtmu43;

```

```

% root3again

```

```

root3againwrtlambda12=t*exp(r3*t)*diffroot3wrtlambda12;
root3againwrtlambda18=t*exp(r3*t)*diffroot3wrtlambda18;
root3againwrtlambda19=t*exp(r3*t)*diffroot3wrtlambda19;
root3againwrtlambda23=t*exp(r3*t)*diffroot3wrtlambda23;
root3againwrtlambda28=t*exp(r3*t)*diffroot3wrtlambda28;
root3againwrtlambda29=t*exp(r3*t)*diffroot3wrtlambda29;
root3againwrtmu21=t*exp(r3*t)*diffroot3wrtmu21;
root3againwrtlambda34=t*exp(r3*t)*diffroot3wrtlambda34;
root3againwrtlambda38=t*exp(r3*t)*diffroot3wrtlambda38;
root3againwrtlambda39=t*exp(r3*t)*diffroot3wrtlambda39;
root3againwrtmu32=t*exp(r3*t)*diffroot3wrtmu32;
root3againwrtlambda45=t*exp(r3*t)*diffroot3wrtlambda45;
root3againwrtlambda48=t*exp(r3*t)*diffroot3wrtlambda48;
root3againwrtlambda49=t*exp(r3*t)*diffroot3wrtlambda49;
root3againwrtmu43=t*exp(r3*t)*diffroot3wrtmu43;

```

```

% root4again

```

```

root4againwrtlambda12=t*exp(r4*t)*diffroot4wrtlambda12;
root4againwrtlambda18=t*exp(r4*t)*diffroot4wrtlambda18;
root4againwrtlambda19=t*exp(r4*t)*diffroot4wrtlambda19;
root4againwrtlambda23=t*exp(r4*t)*diffroot4wrtlambda23;
root4againwrtlambda28=t*exp(r4*t)*diffroot4wrtlambda28;
root4againwrtlambda29=t*exp(r4*t)*diffroot4wrtlambda29;
root4againwrtmu21=t*exp(r4*t)*diffroot4wrtmu21;
root4againwrtlambda34=t*exp(r4*t)*diffroot4wrtlambda34;
root4againwrtlambda38=t*exp(r4*t)*diffroot4wrtlambda38;
root4againwrtlambda39=t*exp(r4*t)*diffroot4wrtlambda39;
root4againwrtmu32=t*exp(r4*t)*diffroot4wrtmu32;
root4againwrtlambda45=t*exp(r4*t)*diffroot4wrtlambda45;
root4againwrtlambda48=t*exp(r4*t)*diffroot4wrtlambda48;
root4againwrtlambda49=t*exp(r4*t)*diffroot4wrtlambda49;
root4againwrtmu43=t*exp(r4*t)*diffroot4wrtmu43;

```

```

% sum the 4 row entries of the score function :

```

```

sumSCORE1=root1againwrtlambda12+root2againwrtlambda12+root3againwrtlambda12+r
oot4againwrtlambda12;
sumSCORE2=root1againwrtlambda18+root2againwrtlambda18+root3againwrtlambda18+r
oot4againwrtlambda18;
sumSCORE3=root1againwrtlambda19+root2againwrtlambda19+root3againwrtlambda19+r
oot4againwrtlambda19;
sumSCORE4=root1againwrtlambda23+root2againwrtlambda23+root3againwrtlambda23+r
oot4againwrtlambda23;
sumSCORE5=root1againwrtlambda28+root2againwrtlambda28+root3againwrtlambda28+r
oot4againwrtlambda28;

```

```

sumSCORE6=root1againwrtlambda29+root2againwrtlambda29+root3againwrtlambda29+root4againwrtlambda29;
sumSCORE7=root1againwrtmu21+root2againwrtmu21+root3againwrtmu21+root4againwrtmu21;
sumSCORE8=root1againwrtlambda34+root2againwrtlambda34+root3againwrtlambda34+root4againwrtlambda34;
sumSCORE9=root1againwrtlambda38+root2againwrtlambda38+root3againwrtlambda38+root4againwrtlambda38;
sumSCORE10=root1againwrtlambda39+root2againwrtlambda39+root3againwrtlambda39+root4againwrtlambda39;
sumSCORE11=root1againwrtmu32+root2againwrtmu32+root3againwrtmu32+root4againwrtmu32;
sumSCORE12=root1againwrtlambda45+root2againwrtlambda45+root3againwrtlambda45+root4againwrtlambda45;
sumSCORE13=root1againwrtlambda48+root2againwrtlambda48+root3againwrtlambda48+root4againwrtlambda48;
sumSCORE14=root1againwrtlambda49+root2againwrtlambda49+root3againwrtlambda49+root4againwrtlambda49;
sumSCORE15=root1againwrtmu43+root2againwrtmu43+root3againwrtmu43+root4againwrtmu43;

% calculate the root5,root6,root7,root8
g5=-1*Qq1(5,5);
g6=-1*Qq1(6,6);
g7=-1*Qq1(7,7);
g8=-1*Qq1(8,8);

sqroot5=sqrt((g5+g6)^2-4*g5*g6);
sqroot6=-1*sqrt((g5+g6)^2-4*g5*g6);
sqroot7=sqrt((g7+g8)^2-4*g7*g8);
sqroot8=-1*sqrt((g7+g8)^2-4*g7*g8);

r5=.5*(-1*(g5+g6)+sqroot5);
r6=.5*(-1*(g5+g6)+sqroot6);
r7=.5*(-1*(g7+g8)+sqroot7);
r8=.5*(-1*(g7+g8)+sqroot8);

diffroot5wrtlambda56=-.5+.5*1/sqroot5*(g5-g6);
diffroot5wrtlambda58=-.5+.5*1/sqroot5*(g5-g6);
diffroot5wrtlambda59=-.5+.5*1/sqroot5*(g5-g6);
diffroot5wrtlambda67=-.5+.5*1/sqroot5*(g6-g5);
diffroot5wrtlambda69=-.5+.5*1/sqroot5*(g6-g5);

diffroot6wrtlambda56=-.5-.5*1/sqroot5*(g5-g6);
diffroot6wrtlambda58=-.5-.5*1/sqroot5*(g5-g6);
diffroot6wrtlambda59=-.5-.5*1/sqroot5*(g5-g6);
diffroot6wrtlambda67=-.5-.5*1/sqroot5*(g6-g5);
diffroot6wrtlambda69=-.5-.5*1/sqroot5*(g6-g5);

% calculate 16th to 20th enteries in the score function
root5againwrtlambda56=t*exp(r5*t)*diffroot5wrtlambda56;
root5againwrtlambda58=t*exp(r5*t)*diffroot5wrtlambda58;
root5againwrtlambda59=t*exp(r5*t)*diffroot5wrtlambda59;
root5againwrtlambda67=t*exp(r5*t)*diffroot5wrtlambda67;

```

```

root5againwrtlambda69=t*exp(r5*t)*diffroot5wrtlambda69;

root6againwrtlambda56=t*exp(r6*t)*diffroot6wrtlambda56;
root6againwrtlambda58=t*exp(r6*t)*diffroot6wrtlambda58;
root6againwrtlambda59=t*exp(r6*t)*diffroot6wrtlambda59;
root6againwrtlambda67=t*exp(r6*t)*diffroot6wrtlambda67;
root6againwrtlambda69=t*exp(r6*t)*diffroot6wrtlambda69;

sumSCORE16=root5againwrtlambda56+root6againwrtlambda56;
sumSCORE17=root5againwrtlambda58+root6againwrtlambda58;
sumSCORE18=root5againwrtlambda59+root6againwrtlambda59;
sumSCORE19=root5againwrtlambda67+root6againwrtlambda67;
sumSCORE20=root5againwrtlambda69+root6againwrtlambda69;

% differentiate root 7 , root 8

diffroot7wrtlambda79=-.5+.5*1/sqroot7*(g7-g8);
diffroot7wrtlambda89=-.5+.5*1/sqroot7*(g8-g7);
diffroot8wrtlambda79=-.5-.5*1/sqroot7*(g7-g8);
diffroot8wrtlambda89=-.5-.5*1/sqroot7*(g8-g7);

% calculate 16th to 20th enteries in the score function

root7againwrtlambda79=t*exp(r7*t)*diffroot7wrtlambda79;
root7againwrtlambda89=t*exp(r7*t)*diffroot7wrtlambda89;
root8againwrtlambda79=t*exp(r8*t)*diffroot8wrtlambda79;
root8againwrtlambda89=t*exp(r8*t)*diffroot8wrtlambda89;

sumSCORE21=root7againwrtlambda79+root8againwrtlambda79;
sumSCORE22=root7againwrtlambda89+root8againwrtlambda89;

% construct score function or score vector(column vector )
scoreFunction =[sumSCORE1;sumSCORE2;sumSCORE3;sumSCORE4;
                sumSCORE5;sumSCORE6;sumSCORE7;sumSCORE8;
                sumSCORE9;sumSCORE10;sumSCORE11;sumSCORE12;
                sumSCORE13;sumSCORE14;sumSCORE15;sumSCORE16;
                sumSCORE17;sumSCORE18;sumSCORE19;sumSCORE20;
                sumSCORE21;sumSCORE22 ];
reArrangedScoreFunction =[ sumSCORE1;sumSCORE7;sumSCORE4;sumSCORE11;
                           sumSCORE8;sumSCORE15;sumSCORE21;sumSCORE22;
                           sumSCORE2;sumSCORE3;sumSCORE5;sumSCORE6;
                           sumSCORE9;sumSCORE10;sumSCORE12;sumSCORE13;
                           sumSCORE14;sumSCORE16;sumSCORE17;sumSCORE18;
                           sumSCORE19;sumSCORE20 ];
% scale the score function by 14204
VectorScalar=14204;
scaledScoreFunction=14204*reArrangedScoreFunction;

% construct the M matrix (hessian matrix), scale this by 606523.3 then invert
% the result
MatrixScalar=606523.3;

```

```

M_matrix=scaledScoreFunction*scaledScoreFunction' ;
scaledM_matrix=MatrixScalar*M_matrix;

invertible_Mmatrix=scaledM_matrix(1:8,1:8);
% invertible_Mmatrix is singular so calculate pseudoinverse
invertM_matrix=pinv(invertible_Mmatrix)

% construct inverted scaled hessian matrix :
M1=invertM_matrix;
M2=zeros(8,14);
M3=zeros(14,8);
M4=zeros(14,14);

invertedScaledHessianMatrix1=[ M1 M2 ;M3 M4 ]

% apply quasiNewton formula :
initialRateVector=[
lambda12;mu21;lambda23;mu32;lambda34;mu43;lambda79;lambda89;lambda18

lambda19;lambda28;lambda29;lambda38;lambda39;lambda45;lambda48;lambda49;
lambda56;lambda58;lambda59;lambda67;lambda69 ];
invertedScaledHessianMatrix1*scaledScoreFunction

desiredVector1= initialRateVector +
(invertedScaledHessianMatrix1*scaledScoreFunction);

% Q1 is the matrix of transition counts among the states,to calculate the
% scalar for the score function it is the total row count for each cell where
the count is nonzero
% scalar for the hessian matrix is the square of the total row divided by
% the cell count where the counts are nonzeros .

Q1=[784      573      74      21      20      0      0      0      10;
9      333      130      21      19      0      0      0      6;
4      11      103      44      19      0      0      10      9 ;
0      0      4      33      21      0      0      8      7;
0      0      0      0      35      12      10      4      6 ;
0      0      0      0      0      1      8      0      1;
0      0      0      0      0      0      10      0      7 ;
0      0      0      0      0      0      0      5      15;
0      0      0      0      0      0      0      0      0];

% calculate the total for each row totsum is a column vector that contains
in each row
% the total of each row

totSum=sum(Q1')% calculate the sum of each row to get the total sum
n1=totSum(1,1) % extract the total of the first row
n2=totSum(1,2) % extract the total of the second row
n3=totSum(1,3) % extract the total of the third row
n4=totSum(1,4) % extract the total of the fourth row
n5=totSum(1,5) % extract the total of the fifth row
n6=totSum(1,6) % extract the total of the sixth row
n7=totSum(1,7) % extract the total of the seventh row

```

```

n8=totSum(1,8) % extract the total of the eighth row

s1=sum(Q1(1,:)>0) % count the total number of cells with nonzero
counts in row1
s2=sum(Q1(2,:)>0) % count the total number of cells with nonzero
counts in row2
s3=sum(Q1(3,:)>0) % count the total number of cells with nonzero
counts in row3
s4=sum(Q1(4,:)>0) % count the total number of cells with nonzero
counts in row4
s5=sum(Q1(5,:)>0) % count the total number of cells with nonzero
counts in row5
s6=sum(Q1(6,:)>0) % count the total number of cells with nonzero
counts in row6
s7=sum(Q1(7,:)>0) % count the total number of cells with nonzero
counts in row7
s8=sum(Q1(8,:)>0) % count the total number of cells with nonzero
counts in row8

% calculate the scoreScalar value

nVector=[n1 n2 n3 n4 n5 n6 n7 n8];% each total row
sVector=[s1 s2 s3 s4 s5 s6 s7 s8]% number of cells in each row with
nonzero counts
scoreScalar=n1*s1+n2*s2+n3*s3+n4*s4+n5*s5+n6*s6+n7*s7+n8*s8

% extract each row from the matrix of the transition counts

v1=Q1(1,1:9)
v2=Q1(2,1:9)
v3=Q1(3,1:9)
v4=Q1(4,1:9)
v5=Q1(5,1:9)
v6=Q1(6,1:9)
v7=Q1(7,1:9)
v8=Q1(8,1:9)

% extract cells with nonzero counts in each row

V1=v1(v1>0)
V2=v2(v2>0)
V3=v3(v3>0)
V4=v4(v4>0)
V5=v5(v5>0)
V6=v6(v6>0)
V7=v7(v7>0)
V8=v8(v8>0)

hessianMatrixscalar=sum(n1^2./V1) + sum(n2^2./V2) + sum(n3^2./V3) +
sum(n4^2./V4) +sum(n5^2./V5) +sum(n6^2./V6) +sum(n7^2./V7) +sum(n8^2./V8)

```

This is the initial Q rate matrix in the second interval :

```
Qq2=[-.397    .39      0      0      0      0      0      0      .007;
      .02    -.28     .25     0      0      0      0      0      .01;
      0      .05    -.38     .24     0      0      0      .04     .05 ;
      0      0      .03    -.55     .28     0      0      .1     .14;
      0      0      0      0     -.37     .19     0      .07     .11;
      0      0      0      0      0     -1      .75     0      .25 ;
      0      0      0      0      0      0     -.43     0      .43;
      0      0      0      0      0      0      0     -1      .75 ;
      0      0      0      0      0      0      0      0      0];
```

Then implement the same code for rate matrix estimation (the code in the first interval) .

This is the initial Q rate matrix in the third interval :

```
Qq3=[-.397    .39      0      0      0      0      0      0      .007;
      .02    -.29     .25     0      0      0      0      0      .02;
      0      .05    -.36     .21     0      0      0      .05     .05 ;
      0      0      0     -.57     .29     0      0      .14     .14;
      0      0      0      0     -.43     .29     0      0      .14;
      0      0      0      0      0     -1      .5      0      .5 ;
      0      0      0      0      0      0     -.5      0      .5;
      0      0      0      0      0      0      0     -1      .67 ;
      0      0      0      0      0      0      0      0      0];
```

Then implement the same code for rate matrix estimation (the code in the first interval)

Now calculate the final rate vector and its variance :

```
clear
clc
close all

% calculate weighted rate vector ( final vector):
load desiredVector1
load desiredVector2
load desiredVector3
load invertedScaledHessianMatrix1
load invertedScaledHessianMatrix2
load invertedScaledHessianMatrix3

% calculate the weighted sum of rate vectors to obtain the final rate vector
```



```

finalRatevectorNEW
=(2/3)*desiredVector1+(4/15)*desiredVector2+(1/15)*desiredVector3

% calculate the weighted sum of the inverted scaled hessian matrix :
finalMofThetaMatrixNEW =(2/3)*invertedScaledHessianMatrix1+(4/15)*
invertedScaledHessianMatrix2 +(1/15)* invertedScaledHessianMatrix3

```

Calculate the Estimated Mean Sojourn Time and Life Expectancy :

```

clear
clc
close all

% calculate mean sojourn time for each state
Qq = [-.397   .39    0    0    0    0    0    0    .007;
       .02   -.281   .25   0    0    0    0    0    .011;
       0     .05   -.365  .225  0    0    0    .047  .043;
       0     0    .041  -.538  .281  0    0    .109  .107;
       0     0     0     0   -.348  .19   0    .059  .099;
       0     0     0     0     0   -.934  .767   0    .167;
       0     0     0     0     0     0   -.421   0    .421;
       0     0     0     0     0     0     0   -.745  .745;
       0     0     0     0     0     0     0     0    0];

g1=Qq(1,1);
g2=Qq(2,2);
g3=Qq(3,3);
g4=Qq(4,4);
g5=Qq(5,5);
g6=Qq(6,6);
g7=Qq(7,7);
g8=Qq(8,8);

% calculate mean sojourn time , it is the reciprocal of the squared values
% on the diagonal of the Qq matrix :
S1=1/(-g1);
S2=1/(-g2);
S3=1/(-g3);
S4=1/(-g4);
S5=1/(-g5);
S6=1/-g6;
S7=1/-g7;
S8=1/-g8;

Svector=[S1;S2;S3;S4;S5;S6;S7;S8]

% calculate variance of the sojourn time:
s1=1/(g1^4);
s2=1/(g2^4);
s3=1/(g3^4);
s4=1/(g4^4);
s5=1/(g5^4);
s6=1/g6^4;

```

```

s7=1/g7^4;
s8=1/g8^4;

svector=[s1;s2;s3;s4;s5;s6;s7;s8]

% create row vector of -1 ones sized 22 by 1 :
Ones= ones(22,1);
NegativeOnes= -1.*Ones;
load finalMofThetaMatrixNEW
v=NegativeOnes'*finalMofThetaMatrixNEW*NegativeOnes
vars1=s1*v
vars2=s2*v
vars3=s3*v
vars4=s4*v
vars5=s5*v
vars6=s6*v
vars7=s7*v
vars8=s8*v

% calculate life expectancy:
% step 1:
B = Qq(1:8,1:8);
% step 2:
Binverse=inv(B);
% step 3:
NegativeBinverse=-1*Binverse;
% step 4:
L=ones(8,1);
LifeExpectancy=NegativeBinverse*L

```

Calculate The Variance Of The Stationary Distribution:

```

clear
clc
close all
% step 1:calculate stationary distribution using pi*Q=0
%where the pi is the stationary distribution and Q is the rate matrix
Qq = [-.397   .39    0    0    0    0    0    0    .007;
      .02   -.281   .25    0    0    0    0    0    .011;
      0    .05   -.365   .225    0    0    0    .047   .043;
      0    0    .041   -.538   .281    0    0    .109   .107;
      0    0    0    0   -.348   .19    0    .059   .099;
      0    0    0    0    0   -.934   .767    0   .167;
      0    0    0    0    0    0   -.421    0   .421;
      0    0    0    0    0    0    0   -.745   .745;
      0    0    0    0    0    0    0    0    0];

% in matrix notation the system of equation is X*pi=y
QqTranspose= Qq'
LowerRow = ones(1,9)
X= [ QqTranspose ; LowerRow ]
Y=[0;0;0;0;0;0;0;0;0;1];

```

```

%calculate Stationary Distribution using  $\pi=(X'X)^{-1}X'Y$ 
inv(X'*X)*X'*Y;
X';
X'*X ;
inv(X'*X);
StatDistrib=inv(X'*X)*X'*Y

% step 2:differentiate the rate matrix Q with respect to each rate
% this will give row vector of ones sized 22 by 1

diffQmatrix =ones(22,1);
transposedDiffMatrix =diffQmatrix';

% step 3: calculate C of theta ; C(theta); it is (9 by 22 matrix )
C_Theta=StatDistrib*transposedDiffMatrix;

% step 4: let  $R=Qq'$  , calculate pseudoinverse for this R matrix
R=Qq';
R_inverse=pinv(R);

%step 5 : multiply this R_inverse by -1
R_inverseNegation =-1*R_inverse;

%step 6 : calculate A of theta ; A(theta): it is (9 by 22 matrix )
A_Theta =R_inverseNegation*C_Theta

% step 7 : multiply the A_Theta *finalMofThetaMatrixNEW*A_Theta'
% finalMofThetaMatrixNEW is the weighted sum of the inverted scaled hessian
matrix

load finalMofThetaMatrixNEW.mat

var_statDistrib=A_Theta*finalMofThetaMatrixNEW*A_Theta'

Hint : if you please at step 7 : multiply the A Theta , in code ocean there is (inverse )after this Theta , please remove it , it is only A
Theta .

```

This is the MATLAB code of how to construct the transition count table in each interval according to the timeline of the patients in the excel sheet :

```

clear
clc
close all

% // this is a code to calculate the observed transition counts among states
in various
% different time interval

```

```
% first load excel sheet as a matrix ( remove any vertical or horizontal  
% headers ) , then extract each row as a vector to count the transitions in  
% different interval . In the time point where the patient did not show up  
% put zeros
```

```
data = xlsread('BigModel1.xlsx','Sheet1');  
class(data)
```

```
[row,col]=size(data)  
x=1:row  
y=1:col
```

```
count=0  
count11=0  
count12=0  
count13=0  
count14=0  
count15=0  
count16=0  
count17=0  
count18=0  
count19=0  
count21=0  
count22=0  
count23=0  
count24=0  
count25=0  
count26=0  
count27=0  
count28=0  
count29=0  
count31=0  
count32=0  
count33=0  
count34=0  
count35=0  
count36=0  
count37=0  
count38=0  
count39=0  
count41=0  
count42=0  
count43=0  
count44=0  
count45=0  
count46=0  
count47=0  
count48=0  
count49=0  
count51=0  
count52=0  
count53=0  
count54=0  
count55=0  
count56=0
```

```

count57=0
count58=0
count59=0
count61=0
count62=0
count63=0
count64=0
count65=0
count66=0
count67=0
count68=0
count69=0
count71=0
count72=0
count73=0
count74=0
count75=0
count76=0
count77=0
count78=0
count79=0
count81=0
count82=0
count83=0
count84=0
count85=0
count86=0
count87=0
count88=0
count89=0

```

```

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/////
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
///
% Now calcualte the transitions among states in the whole matrix during the
3 years period :
% % to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below

for x=1:row
    count=count+1
    a1=data(x,:)
    i=1:length(a1)
    a11=[a1 -1 -1 -1 ]

for i=1:length(a11)-3
    if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count11=count11+1

```

```

        end
    end
    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
            count12=count12+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
            count13=count13+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
            count14=count14+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
            count15=count15+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
            count16=count16+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
            count17=count17+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
            count18=count18+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==1 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
            count19=count19+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
            count21=count21+1
        end
    end

```

```

        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
            count22=count22+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
            count23=count23+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
            count24=count24+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
            count25=count25+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
            count26=count26+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
            count27=count27+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
            count28=count28+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==2 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
            count29=count29+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
            count31=count31+1
        end
    end

```

```

end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
        count32=count32+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
        count33=count33+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
        count34=count34+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
        count35=count35+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
        count36=count36+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
        count37=count37+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
        count38=count38+1
    end
end

for i=1:length(a11)-3
    if a11(i)==3 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
        count39=count39+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count41=count41+1
    end
end

```



```

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
        count42=count42+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
        count43=count43+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
        count44=count44+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
        count45=count45+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
        count46=count46+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
        count47=count47+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
        count48=count48+1
    end
end

for i=1:length(a11)-3
    if a11(i)==4 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
        count49=count49+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count51=count51+1
    end
end

```

```

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
        count52=count52+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
        count53=count53+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
        count54=count54+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
        count55=count55+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
        count56=count56+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
        count57=count57+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
        count58=count58+1
    end
end

for i=1:length(a11)-3
    if a11(i)==5 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
        count59=count59+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count61=count61+1
    end
end

for i=1:length(a11)-3

```

```

        if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
            count62=count62+1
        end
    end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
        count63=count63+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
        count64=count64+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
        count65=count65+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
        count66=count66+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
        count67=count67+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
        count68=count68+1
    end
end

for i=1:length(a11)-3
    if a11(i)==6 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
        count69=count69+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count71=count71+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2

```

```

        count72=count72+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
        count73=count73+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
        count74=count74+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
        count75=count75+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
        count76=count76+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
        count77=count77+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
        count78=count78+1
    end
end

for i=1:length(a11)-3
    if a11(i)==7 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
        count79=count79+1
    end
end

for i=1:length(a11)-3
    if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==1
        count81=count81+1
    end
end

for i=1:length(a11)-3
    if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==2
        count82=count82+1
    end
end

```

```

        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==3
            count83=count83+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==4
            count84=count84+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==5
            count85=count85+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==6
            count86=count86+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==7
            count87=count87+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==8
            count88=count88+1
        end
    end

    for i=1:length(a11)-3
        if a11(i)==8 && a11(i+1)==0 && a11(i+2)==0 && a11(i+3)==9
            count89=count89+1
        end
    end

    Ct3=[count11 count12 count13 count14 count15 count16 count17 count18
count19;
        count21 count22 count23 count24 count25 count26 count27 count28
count29;
        count31 count32 count33 count34 count35 count36 count37 count38
count39;
        count41 count42 count43 count44 count45 count46 count47 count48
count49;
        count51 count52 count53 count54 count55 count56 count57 count58
count59;

```

```

        count61 count62 count63 count64  count65 count66 count67 count68
count69;
        count71 count72 count73 count74  count75 count76 count77 count78
count79;
        count81 count82 count83 count84  count85 count86 count87 count88
count89;
    ]

end

% % % % % % % % % % % % % % % % % % ////////////////

%
////////////////////
//
%
////////////////////
/////
%
////////////////////
///
% Now calculate the transitions among states in the whole matrix during the
2 years period :
% % to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below

Count11=0
Count12=0
Count13=0
Count14=0
Count15=0
Count16=0
Count17=0
Count18=0
Count19=0
Count21=0
Count22=0
Count23=0
Count24=0
Count25=0
Count26=0
Count27=0
Count28=0
Count29=0
Count31=0
Count32=0
Count33=0
Count34=0
Count35=0
Count36=0
Count37=0
Count38=0
Count39=0
Count41=0
Count42=0

```

```

Count43=0
Count44=0
Count45=0
Count46=0
Count47=0
Count48=0
Count49=0
Count51=0
Count52=0
Count53=0
Count54=0
Count55=0
Count56=0
Count57=0
Count58=0
Count59=0
Count61=0
Count62=0
Count63=0
Count64=0
Count65=0
Count66=0
Count67=0
Count68=0
Count69=0
Count71=0
Count72=0
Count73=0
Count74=0
Count75=0
Count76=0
Count77=0
Count78=0
Count79=0
Count81=0
Count82=0
Count83=0
Count84=0
Count85=0
Count86=0
Count87=0
Count88=0
Count89=0

```

```

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/////
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
///
% Now calcualte the transitions among states in the whole matrix during the
2 years period :

```

```
% % to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below
```

```
for x=1:row
    count=count+1
    a1=data(x,:)
    j=1:length(a1)
    a11=[a1  -1  -1]

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==1
        Count11=Count11+1
    end
end
for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==2
        Count12=Count12+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==3
        Count13=Count13+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==4
        Count14=Count14+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==5
        Count15=Count15+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==6
        Count16=Count16+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==7
        Count17=Count17+1
    end
end

for j=1:length(a11)-2
    if a11(j)==1 && a11(j+1)==0 && a11(j+2)==8
        Count18=Count18+1
    end
end
```



```

        end
    end

    for j=1:length(a11)-2
        if a11(j)==1 && a11(j+1)==0 && a11(j+2)==9
            Count19=Count19+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==1
            Count21=Count21+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==2
            Count22=Count22+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==3
            Count23=Count23+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==4
            Count24=Count24+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==5
            Count25=Count25+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==6
            Count26=Count26+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==7
            Count27=Count27+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==8
            Count28=Count28+1
        end
    end

```

```

        end
    end

    for j=1:length(a11)-2
        if a11(j)==2 && a11(j+1)==0 && a11(j+2)==9
            Count29=Count29+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==1
            Count31=Count31+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==2
            Count32=Count32+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==3
            Count33=Count33+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==4
            Count34=Count34+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==5
            Count35=Count35+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==6
            Count36=Count36+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==7
            Count37=Count37+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==3 && a11(j+1)==0 && a11(j+2)==8
            Count38=Count38+1
        end
    end

```

```

end

for j=1:length(a11)-2
    if a11(j)==3 && a11(j+1)==0 && a11(j+2)==9
        Count39=Count39+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==1
        Count41=Count41+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==2
        Count42=Count42+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==3
        Count43=Count43+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==4
        Count44=Count44+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==5
        Count45=Count45+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==6
        Count46=Count46+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==7
        Count47=Count47+1
    end
end

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==8
        Count48=Count48+1
    end
end

```

```

for j=1:length(a11)-2
    if a11(j)==4 && a11(j+1)==0 && a11(j+2)==9
        Count49=Count49+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==1
        Count51=Count51+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==2
        Count52=Count52+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==3
        Count53=Count53+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==4
        Count54=Count54+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==5
        Count55=Count55+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==6
        Count56=Count56+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==7
        Count57=Count57+1
    end
end

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==8
        Count58=Count58+1
    end
end

```

```

for j=1:length(a11)-2
    if a11(j)==5 && a11(j+1)==0 && a11(j+2)==9
        Count59=Count59+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==1
        Count61=Count61+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==2
        Count62=Count62+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==3
        Count63=Count63+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==4
        Count64=Count64+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==5
        Count65=Count65+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==6
        Count66=Count66+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==7
        Count67=Count67+1
    end
end

for j=1:length(a11)-2
    if a11(j)==6 && a11(j+1)==0 && a11(j+2)==8
        Count68=Count68+1
    end
end

for j=1:length(a11)-2

```

```

        if a11(j)==6 && a11(j+1)==0 && a11(j+2)==9
            Count69=Count69+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==1
            Count71=Count71+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==2
            Count72=Count72+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==3
            Count73=Count73+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==4
            Count74=Count74+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==5
            Count75=Count75+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==6
            Count76=Count76+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==7
            Count77=Count77+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==8
            Count78=Count78+1
        end
    end

    for j=1:length(a11)-2
        if a11(j)==7 && a11(j+1)==0 && a11(j+2)==9

```

```

        Count79=Count79+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==1
        Count81=Count81+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==2
        Count82=Count82+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==3
        Count83=Count83+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==4
        Count84=Count84+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==5
        Count85=Count85+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==6
        Count86=Count86+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==7
        Count87=Count87+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==8
        Count88=Count88+1
    end
end

for j=1:length(a11)-2
    if a11(j)==8 && a11(j+1)==0 && a11(j+2)==9
        Count89=Count89+1
    end
end

```

```

        end
    end

    Ct2=[Count11 Count12 Count13 Count14 Count15 Count16 Count17 Count18
Count19;
        Count21 Count22 Count23 Count24 Count25 Count26 Count27 Count28
Count29;
        Count31 Count32 Count33 Count34 Count35 Count36 Count37 Count38
Count39;
        Count41 Count42 Count43 Count44 Count45 Count46 Count47 Count48
Count49;
        Count51 Count52 Count53 Count54 Count55 Count56 Count57 Count58
Count59;
        Count61 Count62 Count63 Count64 Count65 Count66 Count67 Count68
Count69;
        Count71 Count72 Count73 Count74 Count75 Count76 Count77 Count78
Count79;
        Count81 Count82 Count83 Count84 Count85 Count86 Count87 Count88
Count89;
    ]

end

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/
% Now calculate the transitions among states in the whole matrix in delta t =
one year interval
% % to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below

C11=0
C12=0
C13=0
C14=0
C15=0
C16=0
C17=0
C18=0
C19=0
C21=0
C22=0
C23=0
C24=0
C25=0
C26=0
C27=0
C28=0
C29=0
C31=0
C32=0

```


C33=0
C34=0
C35=0
C36=0
C37=0
C38=0
C39=0
C41=0
C42=0
C43=0
C44=0
C45=0
C46=0
C47=0
C48=0
C49=0
C51=0
C52=0
C53=0
C54=0
C55=0
C56=0
C57=0
C58=0
C59=0
C61=0
C62=0
C63=0
C64=0
C65=0
C66=0
C67=0
C68=0
C69=0
C71=0
C72=0
C73=0
C74=0
C75=0
C76=0
C77=0
C78=0
C79=0
C81=0
C82=0
C83=0
C84=0
C85=0
C86=0
C87=0
C88=0
C89=0

```

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/////
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
///
% Now calculate the transitions among states in the whole matrix during the
1 years period :
% % to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below

for x=1:row
    count=count+1
    a1=data(x,:)
    k=1:length(a1)
    a11=[a1  -1 ]

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==1
        C11=C11+1
    end
end
for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==2
        C12=C12+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==3
        C13=C13+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==4
        C14=C14+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==5
        C15=C15+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==6
        C16=C16+1
    end
end

```

```

end

for k=1:length(a11)-1
    if a11(k)==1    && a11(k+1)==7
        C17=C17+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1    && a11(k+1)==8
        C18=C18+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1    && a11(k+1)==9
        C19=C19+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==1
        C21=C21+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==2
        C22=C22+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==3
        C23=C23+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==4
        C24=C24+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==5
        C25=C25+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    && a11(k+1)==6
        C26=C26+1
    end
end

```

```

end

for k=1:length(a11)-1
    if a11(k)==2    &&  a11(k+1)==7
        C27=C27+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    &&  a11(k+1)==8
        C28=C28+1
    end
end

for k=1:length(a11)-1
    if a11(k)==2    &&  a11(k+1)==9
        C29=C29+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==1
        C31=C31+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==2
        C32=C32+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==3
        C33=C33+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==4
        C34=C34+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==5
        C35=C35+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    &&  a11(k+1)==6
        C36=C36+1
    end
end

```

```

for k=1:length(a11)-1
    if a11(k)==3    && a11(k+1)==7
        C37=C37+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    && a11(k+1)==8
        C38=C38+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3    && a11(k+1)==9
        C39=C39+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==1
        C41=C41+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==2
        C42=C42+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==3
        C43=C43+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==4
        C44=C44+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==5
        C45=C45+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==6
        C46=C46+1
    end
end

```

```

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==7
        C47=C47+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==8
        C48=C48+1
    end
end

for k=1:length(a11)-1
    if a11(k)==4    && a11(k+1)==9
        C49=C49+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==1
        C51=C51+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==2
        C52=C52+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==3
        C53=C53+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==4
        C54=C54+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==5
        C55=C55+1
    end
end

for k=1:length(a11)-1
    if a11(k)==5    && a11(k+1)==6
        C56=C56+1
    end
end

for k=1:length(a11)-1

```

```

        if a11(k)==5    && a11(k+1)==7
            C57=C57+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==5    && a11(k+1)==8
            C58=C58+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==5    && a11(k+1)==9
            C59=C59+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==1
            C61=C61+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==2
            C62=C62+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==3
            C63=C63+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==4
            C64=C64+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==5
            C65=C65+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==6
            C66=C66+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==6    && a11(k+1)==7

```

```

        C67=C67+1
    end
end

for k=1:length(a11)-1
    if a11(k)==6    && a11(k+1)==8
        C68=C68+1
    end
end

for k=1:length(a11)-1
    if a11(k)==6    && a11(k+1)==9
        C69=C69+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==1
        C71=C71+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==2
        C72=C72+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==3
        C73=C73+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==4
        C74=C74+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==5
        C75=C75+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==6
        C76=C76+1
    end
end

for k=1:length(a11)-1
    if a11(k)==7    && a11(k+1)==7
        C77=C77+1
    end
end

```



```

        end
    end

    for k=1:length(a11)-1
        if a11(k)==7    && a11(k+1)==8
            C78=C78+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==7    && a11(k+1)==9
            C79=C79+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==1
            C81=C81+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==2
            C82=C82+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==3
            C83=C83+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==4
            C84=C84+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+2)==5
            C85=C85+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==6
            C86=C86+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==8    && a11(k+1)==7
            C87=C87+1
        end
    end

```

```

end

for k=1:length(a11)-1
    if a11(k)==8    && a11(k+1)==8
        C88=C88+1
    end
end

for k=1:length(a11)-1
    if a11(k)==8    && a11(k+1)==9
        C89=C89+1
    end
end

Ct1=[C11 C12 C13 C14 C15 C16 C17 C18 C19;
      C21 C22 C23 C24 C25 C26 C27 C28 C29;
      C31 C32 C33 C34 C35 C36 C37 C38 C39;
      C41 C42 C43 C44 C45 C46 C47 C48 C49;
      C51 C52 C53 C54 C55 C56 C57 C58 C59;
      C61 C62 C63 C64 C65 C66 C67 C68 C69;
      C71 C72 C73 C74 C75 C76 C77 C78 C79;
      C81 C82 C83 C84 C85 C86 C87 C88 C89;
    ]

end

Ctot=Ct1+Ct2+Ct3
sum(Ct1')
sum(Ct2')
sum(Ct3')

```

MATLAB code:

codeocean.com/capsule/7628018/tree/v3

Published New Method Utilizing Continuous Time Markov Chains to analyze Evolution in the Nine State Model of N... (iman attia)

Capsule File Help

Files

Core Files

▼ metadata 797 B ✓

▼ environment 263 B ✓

▼ code 295.69 KB ✓

- BigModel1.xlsx 64.32 KB ✓
- desiredVector1.mat 320 B ✓
- desiredVector2.mat 321 B ✓
- desiredVector3.mat 310 B ✓
- finalMofThetaMatrixNEW... 627 B ✓
- finalRateVectorNEW.mat 358 B ✓
- imanTimeEqualsFIFTYye... 19.91 KB ✓
- imanTimeEqualsONEYea... 19.82 KB ✓
- invertedScaledHessianMat... 629 B ✓
- invertedScaledHessianMat... 631 B ✓
- invertedScaledHessianMat... 628 B ✓
- LICENSE 1.03 KB ✓
- maenSojournTimeAndVa... 1.56 KB ✓
- rateEstimationTequalON... 48.35 KB ✓
- rateEstimationTequalTH... 48.31 KB ✓
- rateEstimationTequalTW... 47.53 KB ✓
- rootsOfFinalQmatrix.m 1.96 KB ✓
- run 2.16 KB ✓
- TRANSITIONcountsBIGm... 34.79 KB ✓
- VarianceOfStationaryDist... 1.56 KB ✓
- weightedRateAndHessian... 617 B ✓

▼ data Manage Datasets 18.56 KB

▼ LICENSE 18.71 KB

Metadata

Bioinformatics

New Method Utilizing Continuous Time Markov Chains to analyze Evolution in the Nine State Model of Non-alcoholic Fatty Liver Disease

iman attia

it contains the code to calculate the transition rate matrix , the transition probability matrix , mean sojourn time in each state , variance-covariance matrix for stationary distribution in the nine states model describing the non-alcoholic fatty liver disease process.

Biostatistics applied mathematical statistics non-alcoholic fatty liver disease continuous time Markov chains Panel data analysis missing data

Capsule

DOI [10.24433/CO.7719785.v3](https://doi.org/10.24433/CO.7719785.v3)

Citation General

iman attia (2022) New Method Utilizing Continuous Time Markov Chains to analyze Evolution in the Nine State Model of Non-alcoholic Fatty Liver Disease [Source Code]. <https://doi.org/10.24433/CO.7719785.v3>

Licenses Code MIT license Data Attribution (CC BY)

Authors iman attia faculty of graduate studies for statistical research

Datasets

Standard Dataset

CTMC ANALYZING NAFLD PROGRESSION (BIG MODEL)

View

Edit

☆☆☆☆☆ 0 ratings - click the stars to submit your

Citation Author(s): Iman Mohammed Attia Abd El-Khalik Abo-Elreesh
(Institute of Statistical Studies and Research, Cairo University)

Submitted by: Iman Attia

Last updated: Sat, 03/05/2022 - 19:49

DOI: 10.21227/0h9c-9283

Data Format: .zip (included file .mat) and matlab file

License: Creative Commons Attribution

13 Views

Categories: Biomedical and Health Sciences

Keywords: continuous time Markov chains, non-alcoholic fatty liver disease, mean sojourn time, stationary probability distribution.

[ACCESS DATASET](#)[CITE](#)[SHARE/EMBED](#)

“CTMC analyzing NAFLD progression (with covariates)”

The History of this **STATA code** of the Poisson regression model on the site **CODE OCEAN** site, with the following versions titled with **“Prognostic Factors for Evolution of Non-Alcoholic Fatty Liver Disease Patients Utilizing Poisson Regression and Continuous Time Markov Chains”**:

Published version 1 on 22 September 2021, URL: <https://codeocean.com/capsules/4752445/tree/v1>
DOI: 10.24433/CO.8778229.v1

Published version 2 on 4 October 2021, URL: <https://codeocean.com/capsule/4752445/tree/v2>
DOI: 10.24433/CO. 8778229.v2

Published version 3 on 8 November 2021, URL: <https://codeocean.com/capsule/4752445/tree/v3>
DOI: 10.24433/CO. 8778229.v3

The data are present at IEEE Data Port site. The title of the code is **“Fibrosis in NAFLD”** with the following DOI: (I uploaded on Sunday 13th of Feb 2022).
DOI: 10.21227/dr5j-gs46

The History of this **MATLAB code** on the site **CODE OCEAN** site, titled **“Using CTMC to Estimate the Rates of Fibrosis Progression in a Subset of States of NAFLD model”**:

Published version 1 on 6 December 2021, URL: <https://codeocean.com/capsules/6377472/tree/v1>
DOI: 10.24433/CO.2144346.v1

Published version 2 on 7 February 2022, URL: <https://codeocean.com/capsule/6377472/tree/v2>
DOI: 10.24433/CO. 2144346.v2

This is a documentation for calculation of statistical indices of the NAFLD process (big model or expanded model):

First: calculate the probability matrix by solving the Kolmogorov differential equations which will be the same as the exponentiation of the rate matrix .

Second: calculate the rate matrix in each time interval i.e. when $\Delta t = \{1\}$

MATLAB code for the calculation of the probability matrix at first year:

In this MATLAB code, the rate matrix is the final estimated rate matrix, time is as chosen by the user :

```
clear
clc
close all
Q1= [ -.059    .059    0    0    0    ,
      .029    -.08    .051    0    0    ,
      .015    .033    -.093    .045    0    ,
      0    .108    .158    -.433    .167    ,
      0    0    0    0    0    ]
lam01=Q1(1,2)
lam12=Q1(2,3)
lam23=Q1(3,4)
lam34=Q1(4,5)
mu10=Q1(2,1)
mu21=Q1(3,2)
mu32=Q1(4,3)
mu20=Q1(3,1)
mu31=Q1(4,2)
g0=-1*Q1(1,1)
g1=-1*Q1(2,2)
g2=-1*Q1(3,3)
g3=-1*Q1(4,4)

b=lam01+lam12+lam23+lam34+mu10+mu21+mu32+mu20+mu31
c=lam01*(lam12+g2+g3)+lam12*(lam23+mu20+g3)+mu10*(g2
+g3)+(lam34+mu31)*(g2)+mu32*(mu21+mu20)
d=lam12*(mu20+lam01)*g3+(lam01+mu10)*(lam34+mu31)*g2+mu32*(lam01+mu10)*(mu21+
mu20)+lam12*lam23*(lam01+lam34)
e=lam01*lam12*lam23*lam34

p=[ 1  b  c  d  e ]
r=roots(p)
r1=r(1,1)
r2=r(2,1)
r3=r(3,1)
r4=r(4,1)

% % eig(Q1)

% calculate probabilities when time t=1
t=1;

% % %calculate DPij*(s)-12 equations
DP00sss= 1
DP00ss=g1+g2+g3
```

```

DP00s=g1*g2 + g2*g3+ g1*g3 -lam23*mu32 -lam12*mu21
DP00constant=g1*g2*g3 -lam23*mu32*g1 -lam12*mu21*g3-lam12*lam23*mu31
DP00col=[DP00sss ;DP00ss ;DP00s ;DP00constant ]
% % % DP01
DP01sss= 0;
DP01ss=lam01;
DP01s=lam01*(g2+g3);
DP01constant=lam01*g2*g3-lam01*lam23*mu32;
DP01col=[DP01sss ;DP01ss ;DP01s ;DP01constant ]
% % % DP02
DP02sss= 0;
DP02ss=0;
DP02s=lam01*lam12;
DP02constant=lam01*lam12*g3;
DP02col=[DP02sss ;DP02ss ;DP02s ;DP02constant ]
% % % DP03
DP03sss= 0;
DP03ss=0;
DP03s=0;
DP03constant=lam01*lam12*lam23;
DP03col=[DP03sss ;DP03ss ;DP03s ;DP03constant ]
% % % DP10
DP10sss= 0;
DP10ss=mu10;
DP10s=mu10*(g3+g2)+mu20*lam12;
DP10constant=mu10*g3*g2+mu20*lam12*g3-mu10*lam23*mu32;
DP10col=[DP10sss ;DP10ss ;DP10s ;DP10constant ]
% % % DP11
DP11sss= 1;
DP11ss=g0+g2+g3;
DP11s=g2*g3+g0*g2+g0*g3-lam23*mu32;
DP11constant=g0*g2*g3-lam23*mu32*g0;
DP11col=[DP11sss ;DP11ss ;DP11s ;DP11constant ]
% % % DP12
DP12sss= 0;
DP12ss=lam12;
DP12s=lam12*(g0+g3);
DP12constant=lam12*g0*g3;
DP12col=[DP12sss ;DP12ss ;DP12s ;DP12constant ]
% % % DP13
DP13sss= 0;
DP13ss=0;
DP13s=lam12*lam23;
DP13constant=lam12*lam23*g0;
DP13col=[DP13sss ;DP13ss ;DP13s ;DP13constant ]
% % % DP20
DP20sss= 0;
DP20ss=mu20;
DP20s=g3*mu20+g1*mu20+mu21*mu10;
DP20constant=mu20*g1*g3+mu21*mu10*g3+mu31*mu10*lam23;
DP20col=[DP20sss ;DP20ss ;DP20s ;DP20constant ]
% % % DP21
DP21sss= 0;
DP21ss=mu21;
DP21s=mu21*(g0+g3)+lam23*mu31+lam01*mu20;
DP21constant=mu21*g0*g3+lam23*mu31*g0+lam01*mu20*g3;
DP21col=[DP21sss ;DP21ss ;DP21s ;DP21constant ]

```

```

% % % DP22
DP22sss= 1;
DP22ss=g0+g1+g3;
DP22s=g0*g1+g0*g3+g1*g3-mu10*lam01;
DP22constant=g0*g1*g3-mu10*lam01*g3;
DP22col=[DP22sss ;DP22ss ;DP22s ;DP22constant ]
% % % DP23
DP23sss= 0;
DP23ss=lam23;
DP23s=lam23*(g0+g1);
DP23constant=lam23*g0*g1-lam23*lam01*mu10;
DP23col=[DP23sss ;DP23ss ;DP23s ;DP23constant ]
% % % DP30
DP30sss=0;
DP30ss=0;
DP30s=mu10*mu31+ mu20*mu32;
DP30constant=mu10*mu21*mu32 +mu10*mu31*g2 +mu20*mu32*g1+mu20*mu31*lam12 ;
DP30col=[DP30sss ;DP30ss ;DP30s ;DP30constant ]
% % % DP31
DP31sss=0;
DP31ss=mu31;
DP31s= mu31*(g0+g2)+mu32*mu21;
DP31constant= mu31*g0*g2 +mu32*mu21*g0+mu20*mu32*lam01;
DP31col=[DP31sss ;DP31ss ;DP31s ;DP31constant ]
% % % DP32
DP32sss=0;
DP32ss=mu32;
DP32s= mu32*(g1+g0)+lam12*mu31;
DP32constant= lam12*mu31*g0+g0*g1*mu32-lam01*mu10*mu32;
DP32col=[DP32sss ;DP32ss ;DP32s ;DP32constant ]
% % % DP33
DP33sss=1;
DP33ss=g0+g1+g2;
DP33s= g0*g1+g0*g2+g1*g2-lam12*mu21-mu10*lam01;
DP33constant= g0*g1*g2-lam01*mu10*g2-mu21*lam12*g0-lam12*lam01*mu20;
DP33col=[DP33sss ;DP33ss ;DP33s ;DP33constant ]

```

```

% % % construct K matrix
% % % then get the inverse of K

```

```

krow1=[1 1 1 1 ];
%krow2
krow2Element1=-1*(r2+r3+r4);
krow2Element2=-1*(r1+r3+r4);
krow2Element3=-1*(r1+r2+r4);
krow2Element4=-1*(r1+r2+r3);
krow2=[krow2Element1 krow2Element2 krow2Element3 krow2Element4]
%krow3
krow3Element1=(r2*r3+r3*r4+r2*r4);
krow3Element2=(r1*r3+r3*r4+r1*r4);
krow3Element3=(r1*r4+r1*r2+r2*r4);
krow3Element4=(r1*r3+r2*r1+r2*r3);
krow3=[krow3Element1 krow3Element2 krow3Element3 krow3Element4]
%krow4
krow4Element1=-1*(r2*r3*r4);

```



```

krow4Element2=-1*(r1*r3*r4);
krow4Element3=-1*(r1*r2*r4);
krow4Element4=-1*(r1*r2*r3);
krow4=[krow4Element1 krow4Element2 krow4Element3 krow4Element4]
K=[krow1;krow2;krow3;krow4];
% get the inverse of K
Kin=inv(K)

% % %calculate Aij,Bij,Cij,Dij for each P00,P01,P02,P03
coeffP00= Kin*DP00col
coeffP01= Kin*DP01col
coeffP02=Kin*DP02col
coeffP03=Kin*DP03col
% % %calculate Aij,Bij,Cij,Dij for each P10,P11,P12,P13
coeffP10=Kin*DP10col
coeffP11=Kin*DP11col
coeffP12=Kin*DP12col
coeffP13=Kin*DP13col
% % %calculate Aij,Bij,Cij,Dij for each P20,P21,P22,P23
coeffP20=Kin*DP20col
coeffP21=Kin*DP21col
coeffP22=Kin*DP22col
coeffP23=Kin*DP23col
% % %calculate Aij,Bij,Cij,Dij for each P30,P31,P32,P33
coeffP30=Kin*DP30col
coeffP31=Kin*DP31col
coeffP32=Kin*DP32col
coeffP33=Kin*DP33col
m = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) ]
% % %calculate first 4 probabilities for the first row :
P00=m*coeffP00
P01=m*coeffP01
P02=m*coeffP02
P03=m*coeffP03
% % %calculate first 4 probabilities for the second row :
P10=m*coeffP10
P11=m*coeffP11
P12=m*coeffP12
P13=m*coeffP13
% % %calculate first 4 probabilities for the third row :
P20=m*coeffP20
P21=m*coeffP21
P22=m*coeffP22
P23=m*coeffP23
% % %calculate first 4 probabilities for the fourth row :
P30=m*coeffP30
P31=m*coeffP31
P32=m*coeffP32
P33=m*coeffP33

% % %calculate P04:

Hs=lam34.*coeffP03;
H1=Hs (1,1)
H2=Hs (2,1)
H3=Hs (3,1)

```

```

H4=Hs(4,1)

h1=H1/r1
h2=H2/r2
h3=H3/r3
h4=H4/r4
h5=-1*(h1+h2+h3+h4)

m04 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(0) ];

h1To5vector=[h1; h2; h3; h4; h5];
P04=m04*h1To5vector

% % %calculate P14
Ms=lam34.*coeffP13;
M1=Ms(1,1)
M2=Ms(2,1)
M3=Ms(3,1)
M4=Ms(4,1)

m1=M1/r1
m2=M2/r2
m3=M3/r3
m4=M4/r4
m5=-1*(m1+m2+m3+m4)

m14 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(0) ];

m1To5vector=[m1; m2; m3; m4; m5];
P14=m14*m1To5vector

% % %calculate P24
Ls=lam34.*coeffP23;
L1=Ls(1,1)
L2=Ls(2,1)
L3=Ls(3,1)
L4=Ls(4,1)

l1=L1/r1
l2=L2/r2
l3=L3/r3
l4=L4/r4
l5=-1*(l1+l2+l3+l4)

m24 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(0) ];

l1To5vector=[l1; l2; l3; l4; l5];
P24=m24*l1To5vector

% % %calculate P34
Zs=lam34.*coeffP33;
Z1=Zs(1,1)
Z2=Zs(2,1)

```

```

Z3=Zs(3,1)
Z4=Zs(4,1)

z1=Z1/r1
z2=Z2/r2
z3=Z3/r3
z4=Z4/r4
z5=-1*(l1+l2+l3+l4)

m34 = [ exp(r1*t) exp(r2*t) exp(r3*t) exp(r4*t) exp(0) ];

z1To5vector=[z1; z2; z3; z4; z5];
P34=m34*z1To5vector

Pfibrosis=[P00 P01 P02 P03 P04;
            P10 P11 P12 P13 P14 ;
            P20 P21 P22 P23 P24;
            P30 P31 P32 P33 P34;
            0 0 0 0 1 ]

```

MATLAB code for calculation of the rate matrix at first interval:

```

clear
clc
close all
Q1= [ -.059      .059      0      0      0;
      .0288     -.0825     .0537     0      0;
      .016      .0383     -.1013     .047     0;
      0         .116      .191      -.49     .183;
      0         0         0         0      0 ]

lam01=Q1(1,2)
lam12=Q1(2,3)
lam23=Q1(3,4)
lam34=Q1(4,5)
mu10=Q1(2,1)
mu21=Q1(3,2)
mu32=Q1(4,3)
mu20=Q1(3,1)
mu31=Q1(4,2)
gamma1=-1*Q1(2,2)
gamma2=-1*Q1(3,3)
gamma3=-1*Q1(4,4)

b=lam01+lam12+lam23+lam34+mu10+mu21+mu32+mu20+mu31
c=lam01*(lam12+gamma2+gamma3)+lam12*(lam23+mu20+gamma3)+mu10*(gamma2
+gamma3)+(lam34+mu31)*(gamma2)+mu32*(mu21+mu20)
d=lam12*(mu20+lam01)*gamma3+(lam01+mu10)*(lam34+mu31)*gamma2+mu32*(lam01+mu10
)*(mu21+mu20)+lam12*lam23*(lam01+lam34)
e=lam01*lam12*lam23*lam34

p=[ 1 b c d e ]
r=roots(p)

```

```

eig(Q1)

% calculate the roots of the 4 th polynomila :
% we need to calculate p,q, ( $\Delta_0$ ), ( $\Delta_1$ ),  $\cos(\varnothing/3)$ , s as defined in the following
routines:

p=(8*c-3*b^2)/8;
q=(b^3-4*b*c+8*d)/8;
delta0=c^2-3*b*d+12*e;
sqrdelta0=sqrt(delta0);
delta0cubed=delta0^3;
sqrdelta0cubed=sqrt(delta0cubed);
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e;
theta=acos(delta1/(2*sqrt(delta0^3)));
m=cos(theta/3);
s=.5*sqrt(2/3*(-1*p+sqrdelta0*m));

r1=-(b/4)-s+.5*sqrt(-4*s^2-2*p+(q/s))
r2=-(b/4)-s-.5*sqrt(-4*s^2-2*p+(q/s))
r3=-(b/4)+s+.5*sqrt(-4*s^2-2*p-(q/s))
r4=-(b/4)+s-.5*sqrt(-4*s^2-2*p-(q/s))

rootscalcu=[ r1  r2  r3  r4]

% % differentiate the coefficient(b,c,d,e)with respect to each theta

% differentiate [B] , [C] , [D] , [E] w.r.t. lambda01
diffBwrtlam01=1
diffCwrtlam01=lam12+lam23+mu21+mu20+lam34+mu32+mu31
diffDwrtlam01=lam12*(lam34+mu32+mu31)+(lam34+mu31)*(lam23+mu21+mu20)+mu32*(mu
21+mu20)+lam12*lam23
diffEwrtlam01=lam12*lam23*lam34
diffLAM01= [ diffBwrtlam01  diffCwrtlam01  diffDwrtlam01  diffEwrtlam01]

% differentiate [B] , [C] , [D] , [E] w.r.t. lambda12
diffBwrtlam12=1
diffCwrtlam12=lam01+lam23+mu20+lam34+mu32+mu31
diffDwrtlam12=(lam01+mu20)*(lam34+mu32+mu31)+lam23*(lam01+lam34)
diffEwrtlam12=lam01*lam23*lam34
diffLAM12= [ diffBwrtlam12  diffCwrtlam12  diffDwrtlam12  diffEwrtlam12 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. mu10
diffBwrtmu10=1
diffCwrtmu10=lam23+mu21+mu20+lam34+mu32+mu31
diffDwrtmu10=(lam34+mu31)*(lam23+mu21+mu20)+mu32*(mu21+mu20)
diffEwrtmu10=0
diffMU10= [ diffBwrtmu10  diffCwrtmu10  diffDwrtmu10  diffEwrtmu10 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. lambda23
diffBwrtlam23=1
diffCwrtlam23=lam01+lam12+mu10+lam34+mu31
diffDwrtlam23=(lam01+mu10)*(lam34+mu31)+lam12*(lam01+lam34)
diffEwrtlam23=lam01*lam12*lam34
diffLAM23= [ diffBwrtlam23  diffCwrtlam23  diffDwrtlam23  diffEwrtlam23 ]

```

```

% differentiate [B] , [C] , [D] , [E] w.r.t. mu21
diffBwrtmu21=1
diffCwrtmu21=lam01+mu10+lam34+mu31+mu32
diffDwrtmu21=(lam01+mu10)*(lam34+mu31)+mu32*(lam01+mu10)
diffEwrtmu21=0
diffMU21= [ diffBwrtmu21    diffCwrtmu21    diffDwrtmu21    diffEwrtmu21 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. mu20
diffBwrtmu20=1
diffCwrtmu20=lam01+lam12+mu10+lam34+mu31+mu32
diffDwrtmu20=lam12*(lam34+mu32+mu31)+(lam01+mu10)*(lam34+mu31)+mu32*(lam01+mu10)
diffEwrtmu20=0
diffMU20= [ diffBwrtmu20    diffCwrtmu20    diffDwrtmu20    diffEwrtmu20 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. lambda34
diffBwrtlam34=1
diffCwrtlam34=lam01+lam12+mu10+lam23+mu21+mu20
diffDwrtlam34=lam12*(lam01+mu20)+(lam01+mu10)*(lam23+mu21+mu20)+lam12*lam23
diffEwrtlam34=lam01*lam12*lam23
diffLAM34= [ diffBwrtlam34    diffCwrtlam34    diffDwrtlam34    diffEwrtlam34 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. mu32
diffBwrtmu32=1
diffCwrtmu32=lam01+lam12+mu10+mu21+mu20
diffDwrtmu32=lam12*(lam01+mu20)+(lam01+mu10)*(mu21+mu20)
diffEwrtmu32=0
diffMU32= [ diffBwrtmu32    diffCwrtmu32    diffDwrtmu32    diffEwrtmu32 ]

% differentiate [B] , [C] , [D] , [E] w.r.t. mu31
diffBwrtmu31=1
diffCwrtmu31=lam01+lam12+mu10+lam23+mu21+mu20
diffDwrtmu31=lam12*(lam01+mu20)+(lam01+mu10)*(lam23+mu21+mu20)
diffEwrtmu31=0
diffMU31= [ diffBwrtmu31    diffCwrtmu31    diffDwrtmu31    diffEwrtmu31 ]

% differentiaiton of the coefficient of polynomial ( aggregated or in
summary)
diffLAM01=[ diffBwrtlam01    diffCwrtlam01    diffDwrtlam01    diffEwrtlam01]
diffLAM12=[ diffBwrtlam12    diffCwrtlam12    diffDwrtlam12    diffEwrtlam12 ]
diffMU10=[diffBwrtmu10    diffCwrtmu10    diffDwrtmu10    diffEwrtmu10 ]
diffLAM23=[ diffBwrtlam23    diffCwrtlam23    diffDwrtlam23    diffEwrtlam23 ]
diffMU21=[ diffBwrtmu21    diffCwrtmu21    diffDwrtmu21    diffEwrtmu21 ]
diffMU20=[ diffBwrtmu20    diffCwrtmu20    diffDwrtmu20    diffEwrtmu20 ]
diffLAM34=[ diffBwrtlam34    diffCwrtlam34    diffDwrtlam34    diffEwrtlam34 ]
diffMU32=[ diffBwrtmu32    diffCwrtmu32    diffDwrtmu32    diffEwrtmu32 ]
diffMU31=[ diffBwrtmu31    diffCwrtmu31    diffDwrtmu31    diffEwrtmu31 ]

% now we need to differentiate the p , q and S with respect to each rate .
% differentiating the p and the q are straightforward
% in the next few lines, there is an explanation of differentiating the s
%  $s=1/2*\sqrt{(-2p)/3 + 2/(3a)*\sqrt{\Delta}*\cos \varnothing/3}$  and as  $a=1 \rightarrow$ 
%  $s=1/2*\sqrt{(-2p)/3 + 2/(3a)*\sqrt{\Delta}*\cos \varnothing/3}$ 
%  $S'=1/4*\{\text{reciprocal of } \sqrt{(-2p)/3 + 2/3*\sqrt{\Delta}*\cos (\varnothing/3)}\}*(-2p')/3+2/3*[\sqrt{\Delta}]*'\cos(\varnothing/3) + 2/3*\sqrt{\Delta}*\cos (\varnothing/3) ]'$ 
% where  $\{\text{reciprocal of } \sqrt{(-2p)/3 + 2/3*\sqrt{\Delta}*\cos (\varnothing/3)}\}=h$ 

```

```

% sqrt[2/3*(-1*p)+sqrt(Δ0)*cos (Ø/3)]=sqrt{2/3*[(-1*p)+ sqrt(Δ0)*m]}= n where
% cos (Ø/3)=m , and thus h=1/n
% (-2p')/3 is called first term in the differentiation of S
% 2/3*[sqrt(Δ0)]*'cos (Ø/3) is called the second term in differentiation of S
% 2/3*sqrt(Δ0)*[cos (Ø/3)]' is called third term of differentiation of S
% for the second term , we need to differentiate sqrt(Δ0)thus:
% sqrt(Δ0)=sqrt(c^2-3bd+12e)?[sqrt(Δ0)]'=1/2*(c^2-3bd+12e)^(-.5)*(2cc'-3b'd-
3bd'+12e')
% second term in differentiation of S is :
% 2/3*[sqrt(Δ0)]*'cos(Ø/3)=1/3*(c^2-3bd+12e)^(-.5)*(2cc'-3b'd-
3bd'+12e')*cos(Ø/3)
% for the third term we need to differentiate [cos (Ø/3)] thus
% [cos (Ø/3)]'= (-sin (Ø/3))*{(-1)/sqrt(1-
[Δ1/(2*sqrt(Δ0^3)) ]^2)}*{1/2*[(Δ1)'*sqrt(Δ0^3)-Δ1*(sqrt(Δ0^3))']/(Δ0^3)}
% where:
% (Δ1)'=6c^2*c'-9b'*cd-9bc'*d-9bcd'+54bb'*e+27b^2*e'+54dd'-72c'*e-72ce'
% [Δ0^(3/2)]'=3/2*(c^2-3bd+12e)^.5 *(2cc'-3b'd-3bd'+12e')
% differentiation of the third term is
% 2/3*sqrt(Δ0)*[cos(Ø/3)]'
% 2/9*sqrt(Δ0)*[(-sin(Ø/3))*{(-1)/sqrt(1-
[Δ1/(2*sqrt(Δ0^3)) ]^2)}*{1/2*[(Δ1)'*sqrt(Δ0^3)-Δ1*(sqrt(Δ0^3))']/(Δ0^3)}
% third term in the differentiation of S =(-1)/9*{(sqrt(Δ0)*[-
sin(Ø/3)])/sqrt(1-[Δ1/(2*sqrt(Δ0^3)) ]^2)}*{[(Δ1)'*sqrt(Δ0^3)-
Δ1*(sqrt(Δ0^3))']/(Δ0^3)}
% num1=(sqrt(Δ0)*[-sin(Ø/3)])
% den1=sqrt(1-[Δ1/(2*sqrt(Δ0^3)) ]^2)
% k=(-1)/9*(num1/den1)
% num2=(Δ1)'*sqrt(Δ0^3)-Δ1*(sqrt(Δ0^3))'
% den2=Δ0^3
% third term in the differentiation of S=k*(num2/den2)
% S'=(1/4)*h*(differentiation of first term+differentiation of second term
% + differentiation of third term)

% ///////////NOW DIFFERENTATE P WITH RESPECT TO EACH RATE

% differentiate p w.r.t.  lambda 01:
diffPwrtlam01=(8*diffCwrtlam01-6*b*diffBwrtlam01)/8

% differentiate p w.r.t.  lambda 12:
diffPwrtlam12=(8*diffCwrtlam12-6*b*diffBwrtlam12)/8

% differentiate p w.r.t.  mu 01:
diffPwrtmu10=(8*diffCwrtmu10-6*b*diffBwrtmu10)/8

% differentiate p w.r.t.  lambda 23:
diffPwrtlam23=(8*diffCwrtlam23-6*b*diffBwrtlam23)/8

% differentiate p w.r.t.  mu 21:
diffPwrtmu21=(8*diffCwrtmu21-6*b*diffBwrtmu21)/8

% differentiate p w.r.t.  mu 20:
diffPwrtmu20=(8*diffCwrtmu20-6*b*diffBwrtmu20)/8

% differentiate p w.r.t.  lambda 34:
diffPwrtlam34=(8*diffCwrtlam34-6*b*diffBwrtlam34)/8

```

```

% differentiate p w.r.t. mu 32:
diffPwrtmu32=(8*diffCwrtmu32-6*b*diffBwrtmu32)/8

% differentiate p w.r.t. mu31:
diffPwrtmu31=(8*diffCwrtmu31-6*b*diffBwrtmu31)/8

% ////////// NOW DIFFERENTIATE THE Q WITH RESPECT TO EACH RATE

% diff q w.r.t. lambda 01
diffQwrtlam01=(3*b^2-4*diffBwrtlam01*c-4*diffCwrtlam01*b+8*diffDwrtlam01)/8

% diff q w.r.t. lambda 12
diffQwrtlam12=(3*b^2-4*diffBwrtlam12*c-4*diffCwrtlam12*b+8*diffDwrtlam12)/8

% diff q w.r.t. mu 10
diffQwrtmu10=(3*b^2-4*diffBwrtmu10*c-4*diffCwrtmu10*b+8*diffDwrtmu10)/8

% diff q w.r.t. lambda 23
diffQwrtlam23=(3*b^2-4*diffBwrtlam23*c-4*diffCwrtlam23*b+8*diffDwrtlam23)/8

% diff q w.r.t. mu 21
diffQwrtmu21=(3*b^2-4*diffBwrtmu21*c-4*diffCwrtmu21*b+8*diffDwrtmu21)/8

% diff q w.r.t. mu 20
diffQwrtmu20=(3*b^2-4*diffBwrtmu20*c-4*diffCwrtmu20*b+8*diffDwrtmu20)/8

% diff q w.r.t. lambda 34
diffQwrtlam34=(3*b^2-4*diffBwrtlam34*c-4*diffCwrtlam34*b+8*diffDwrtlam34)/8

% diff q w.r.t. mu 32
diffQwrtmu32=(3*b^2-4*diffBwrtmu32*c-4*diffCwrtmu32*b+8*diffDwrtmu32)/8

% diff q w.r.t. mu 31
diffQwrtmu31=(3*b^2-4*diffBwrtmu31*c-4*diffCwrtmu31*b+8*diffDwrtmu31)/8

% ///////////NOW DIFFERENTIATING THE S
% THE FIRST TERM IN DIFFERENTIATION OF S is (-2*p')/3 so multiplying p'
% with -2/3

lam01FirstTermSDiff=(-2/3)*diffPwrtlam01
lam12FirstTermSDiff=(-2/3)*diffPwrtlam12
mu10FirstTermSDiff=(-2/3)*diffPwrtmu10
lam23FirstTermSDiff=(-2/3)*diffPwrtlam23
mu21FirstTermSDiff=(-2/3)*diffPwrtmu21
mu20FirstTermSDiff=(-2/3)*diffPwrtmu20
lam34FirstTermSDiff=(-2/3)*diffPwrtlam34
mu32FirstTermSDiff=(-2/3)*diffPwrtmu32
mu31FirstTermSDiff=(-2/3)*diffPwrtmu31

% NOW the second term in differentiating the S
% 2/3*[sqrt(Δ0)]'*cos(Ø/3) is called the second term in differentiation of S
% for the second term , we need to differentiate sqrt(Δ0)thus:

```

```

% sqrt( $\Delta 0$ )=sqrt( $c^2-3bd+12e$ )→[sqrt( $\Delta 0$ )]'=1/2*( $c^2-3bd+12e$ )(-.5)*(2cc'-3b'd-3bd'+12e')
% second term in differentiation of S is :
% 2/3*[sqrt( $\Delta 0$ )]'*cos( $\emptyset/3$ )=1/3*( $c^2-3bd+12e$ )(-.5)*(2cc'-3b'd-3bd'+12e')*cos( $\emptyset/3$ )
% this is done in two steps
% step 1 =( $c^2-3bd+12e$ )(-.5)
% step 2 = differentiate the  $\Delta 0$  w.r.t. each rate
% [sqrt( $\Delta 0$ )]'=1/2*( $c^2-3bd+12e$ )(-.5)*(2cc'-3b'd-3bd'+12e')

reciprocalOfsqrdelta0=1/sqrdelta0

% diff sqrt delta0( $\Delta 0$ ) w.r.t. lambda 01
diffDelta0wrtlam01=2*c*diffCwrtlam01-3*diffBwrtlam01*d-
3*diffDwrtlam01*b+12*diffEwrtlam01
diffSqrDelta0wrtlam01=.5*reciprocalOfsqrdelta0*diffDelta0wrtlam01

% diff sqrt delta0( $\Delta 0$ ) w.r.t. lambda 12
diffDelta0wrtlam12=2*c*diffCwrtlam12-3*diffBwrtlam12*d-
3*diffDwrtlam12*b+12*diffEwrtlam12
diffSqrDelta0wrtlam12=.5*reciprocalOfsqrdelta0*diffDelta0wrtlam12

% diff sqrt delta0( $\Delta 0$ ) w.r.t. mu 10
diffDelta0wrtmu10=2*c*diffCwrtmu10-3*diffBwrtmu10*d-
3*diffDwrtmu10*b+12*diffEwrtmu10
diffSqrDelta0wrtmu10=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu10

% diff sqrt delta0( $\Delta 0$ ) w.r.t. lambda 23
diffDelta0wrtlam23=2*c*diffCwrtlam23-3*diffBwrtlam23*d-
3*diffDwrtlam23*b+12*diffEwrtlam23
diffSqrDelta0wrtlam23=.5*reciprocalOfsqrdelta0*diffDelta0wrtlam23

% diff sqrt delta0( $\Delta 0$ ) w.r.t. mu 21
diffDelta0wrtmu21=2*c*diffCwrtmu21-3*diffBwrtmu21*d-
3*diffDwrtmu21*b+12*diffEwrtmu21
diffSqrDelta0wrtmu21=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu21

% diff sqrt delta0( $\Delta 0$ ) w.r.t. mu 20
diffDelta0wrtmu20=2*c*diffCwrtmu20-3*diffBwrtmu20*d-
3*diffDwrtmu20*b+12*diffEwrtmu20
diffSqrDelta0wrtmu20=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu20

% diff sqrt delta0( $\Delta 0$ ) w.r.t. lambda 34
diffDelta0wrtlam34=2*c*diffCwrtlam34-3*diffBwrtlam34*d-
3*diffDwrtlam34*b+12*diffEwrtlam34
diffSqrDelta0wrtlam34=.5*reciprocalOfsqrdelta0*diffDelta0wrtlam34

% diff sqrt delta0( $\Delta 0$ ) w.r.t. mu 32
diffDelta0wrtmu32=2*c*diffCwrtmu32-3*diffBwrtmu32*d-
3*diffDwrtmu32*b+12*diffEwrtmu32
diffSqrDelta0wrtmu32=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu32

% diff sqrt delta0( $\Delta 0$ ) w.r.t. mu 31
diffDelta0wrtmu31=2*c*diffCwrtmu31-3*diffBwrtmu31*d-
3*diffDwrtmu31*b+12*diffEwrtmu31

```



```

diffSqrDelta0wrtmu31=.5*reciprocalOfsqrdelta0*diffDelta0wrtmu31

% NOW calculate the second term in differentiating the S
% 2/3*[sqrt(Δ0)]'*cos(Ø/3) where cos(Ø/3)= m

lam01SecondTermSDiff=(2/3)*diffSqrDelta0wrtlam01*m
lam12SecondTermSDiff=(2/3)*diffSqrDelta0wrtlam12*m
mu10SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu10*m
lam23SecondTermSDiff=(2/3)*diffSqrDelta0wrtlam23*m
mu21SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu21*m
mu20SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu20*m
lam34SecondTermSDiff=(2/3)*diffSqrDelta0wrtlam34*m
mu32SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu32*m
mu31SecondTermSDiff=(2/3)*diffSqrDelta0wrtmu31*m

% NOW the third term in differentiating the S :

% 2/3*sqrt(Δ0)*[cos (Ø/3)]' is called third term in differentiation of S

% for the third term we need to differentiate [cos (Ø/3)] thus
% [cos (Ø/3)]' = (-sin (Ø/3))*{(-1)/sqrt(1-
[Δ1/(2*sqrt(Δ0^3))]^2)}*{1/2*[(Δ1)'*sqrt(Δ0^3)-?1*(sqrt(Δ0^3))']/ (Δ0^3)}
% where:
% (Δ1)' = 6c^2*c' - 9b'*cd - 9bc'*d - 9bcd' + 54bb'*e + 27b^2*e' + 54dd' - 72c'*e - 72ce'
% [Δ0^(3/2)]' = 3/2*(c^2 - 3bd + 12e)^.5 *(2cc' - 3b'd - 3bd' + 12e')
% differentiation of the third term is
% 2/3*sqrt(Δ0)*[cos(Ø/3)]'
% 2/9*sqrt(Δ0)*[(-sin(Ø/3))*{(-1)/sqrt(1-
[Δ1/(2*sqrt(Δ0^3))]^2)}*{1/2*[(Δ1)'*sqrt(Δ0^3)-?1*(sqrt(Δ0^3))']/ (Δ0^3)}
% third term in the differentiation of S = (-1)/9*{(sqrt(Δ0))*[-
sin(Ø/3)]/sqrt(1-[Δ1/(2*sqrt(Δ0^3))]^2)}*{[(Δ1)'*sqrt(Δ0^3)-
Δ1*(sqrt(Δ0^3))']/(Δ0^3)}

% ///////////NOW calculate the k as previously illustrated and as in the
% following steps :
% num1=(sqrt(Δ0)*[-sin(Ø/3)])
% den1=sqrt(1-[Δ1/(2*sqrt(Δ0^3))]^2)
% k=(-1)/9*(num1/den1)

sintheta=-sin(theta/3)
sqrt(delta0)
num1=sintheta*sqrt(delta0)
den1=sqrt(1-(delta1/(2*sqrt(delta0^3)))^2)
k=(-1/9)*(num1/den1)

% NOW//// to obtain the third term in differentiation of S
% calculate (Δ1)' & (sqrt(Δ0^3))'
% num2=(Δ1)'*sqrt(Δ0^3)-Δ1*(sqrt(Δ0^3))'
% den2=Δ0^3
% third term in the differentiation of S=k*(num2/den2)
% S'=(1/4)*h*(differentiation of first term+differentiation of second term
% + differentiation of third term)

delta0cubed=delta0^3
sqrdelta0cubed=sqrt(delta0cubed)

```

```
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e
```

```
% calculate ( $\Delta_1$ )' with respect to each theta :
```

```
DiffOfdelta1wrtlam01=6*c^2*diffCwrtlam01-9*diffBwrtlam01*c*d-
9*diffCwrtlam01*b*d-
9*diffDwrtlam01*b*c+54*b*e*diffBwrtlam01+27*b^2*diffEwrtlam01+54*d*diffDwrtla
m01-72*diffCwrtlam01*e-72*c*diffEwrtlam01
DiffOfdelta1wrtlam12=6*c^2*diffCwrtlam12-9*diffBwrtlam12*c*d-
9*diffCwrtlam12*b*d-
9*diffDwrtlam12*b*c+54*b*e*diffBwrtlam12+27*b^2*diffEwrtlam12+54*d*diffDwrtla
m12-72*diffCwrtlam12*e-72*c*diffEwrtlam12
DiffOfdelta1wrtmu10=6*c^2*diffCwrtmu10-9*diffBwrtmu10*c*d-9*diffCwrtmu10*b*d-
9*diffDwrtmu10*b*c+54*b*e*diffBwrtmu10+27*b^2*diffEwrtmu10+54*d*diffDwrtmu10-
72*diffCwrtmu10*e-72*c*diffEwrtmu10
DiffOfdelta1wrtlam23=6*c^2*diffCwrtlam23-9*diffBwrtlam23*c*d-
9*diffCwrtlam23*b*d-
9*diffDwrtlam23*b*c+54*b*e*diffBwrtlam23+27*b^2*diffEwrtlam23+54*d*diffDwrtla
m23-72*diffCwrtlam23*e-72*c*diffEwrtlam23
DiffOfdelta1wrtmu21=6*c^2*diffCwrtmu21-9*diffBwrtmu21*c*d-9*diffCwrtmu21*b*d-
9*diffDwrtmu21*b*c+54*b*e*diffBwrtmu21+27*b^2*diffEwrtmu21+54*d*diffDwrtmu21-
72*diffCwrtmu21*e-72*c*diffEwrtmu21
DiffOfdelta1wrtmu20=6*c^2*diffCwrtmu20-9*diffBwrtmu20*c*d-9*diffCwrtmu20*b*d-
9*diffDwrtmu20*b*c+54*b*e*diffBwrtmu20+27*b^2*diffEwrtmu20+54*d*diffDwrtmu20-
72*diffCwrtmu20*e-72*c*diffEwrtmu20
DiffOfdelta1wrtlam34=6*c^2*diffCwrtlam34-9*diffBwrtlam34*c*d-
9*diffCwrtlam34*b*d-
9*diffDwrtlam34*b*c+54*b*e*diffBwrtlam34+27*b^2*diffEwrtlam34+54*d*diffDwrtla
m34-72*diffCwrtlam34*e-72*c*diffEwrtlam34
DiffOfdelta1wrtmu32=6*c^2*diffCwrtmu32-9*diffBwrtmu32*c*d-9*diffCwrtmu32*b*d-
9*diffDwrtmu32*b*c+54*b*e*diffBwrtmu32+27*b^2*diffEwrtmu32+54*d*diffDwrtmu32-
72*diffCwrtmu32*e-72*c*diffEwrtmu32
DiffOfdelta1wrtmu31=6*c^2*diffCwrtmu31-9*diffBwrtmu31*c*d-9*diffCwrtmu31*b*d-
9*diffDwrtmu31*b*c+54*b*e*diffBwrtmu31+27*b^2*diffEwrtmu31+54*d*diffDwrtmu31-
72*diffCwrtmu31*e-72*c*diffEwrtmu31
```

```
% calculate ( $\sqrt{\Delta_0^3}$ )' with respect to each theta :
```

```
sqrdelta0=sqrt(delta0);
DiffsqrtOfdelta0cubedwrtlam01=(3/2)*sqrdelta0*(2*c*diffCwrtlam01-
3*diffBwrtlam01*d-3*diffDwrtlam01*b+12*diffEwrtlam01)
DiffsqrtOfdelta0cubedwrtlam12=(3/2)*sqrdelta0*(2*c*diffCwrtlam12-
3*diffBwrtlam12*d-3*diffDwrtlam12*b+12*diffEwrtlam12)
DiffsqrtOfdelta0cubedwrtmu10=(3/2)*sqrdelta0*(2*c*diffCwrtmu10-
3*diffBwrtmu10*d-3*diffDwrtmu10*b+12*diffEwrtmu10)
DiffsqrtOfdelta0cubedwrtlam23=(3/2)*sqrdelta0*(2*c*diffCwrtlam23-
3*diffBwrtlam23*d-3*diffDwrtlam23*b+12*diffEwrtlam23)
DiffsqrtOfdelta0cubedwrtmu21=(3/2)*sqrdelta0*(2*c*diffCwrtmu21-
3*diffBwrtmu21*d-3*diffDwrtmu21*b+12*diffEwrtmu21)
DiffsqrtOfdelta0cubedwrtmu20=(3/2)*sqrdelta0*(2*c*diffCwrtmu20-
3*diffBwrtmu20*d-3*diffDwrtmu20*b+12*diffEwrtmu20)
```

```

DiffsqrtOfdelta0cubedwrtlam34=(3/2)*sqrdelta0*(2*c*diffCwrtlam34-
3*diffBwrtlam34*d-3*diffDwrtlam34*b+12*diffEwrtlam34)
DiffsqrtOfdelta0cubedwrtmu32=(3/2)*sqrdelta0*(2*c*diffCwrtmu32-
3*diffBwrtlam01*d-3*diffDwrtmu32*b+12*diffEwrtmu32)
DiffsqrtOfdelta0cubedwrtmu31=(3/2)*sqrdelta0*(2*c*diffCwrtmu31-
3*diffBwrtmu31*d-3*diffDwrtmu31*b+12*diffEwrtmu31)

% NOW calculate the third term as in the following steps
% num2=(Δ1)'*sqrt(Δ0^3)-Δ1*(sqrt(Δ0^3))'
% den2=Δ0^3
% third term or the last term in the differentiation of S=k*(num2/den2)

delta0cubed=delta0^3
sqrdelta0cubed=sqrt(delta0cubed)
delta1=2*c^3-9*b*c*d+27*b^2*e+27*d^2-72*c*e

lam01LastTermSDiff=k*(DiffOfdelta1wrtlam01*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlam01)/delta0cubed
lam12LastTermSDiff=k*(DiffOfdelta1wrtlam12*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlam12)/delta0cubed
mu10LastTermSDiff=k*(DiffOfdelta1wrtmu10*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu10)/delta0cubed
lam23LastTermSDiff=k*(DiffOfdelta1wrtlam23*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlam23)/delta0cubed
mu21LastTermSDiff=k*(DiffOfdelta1wrtmu21*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu21)/delta0cubed
mu20LastTermSDiff=k*(DiffOfdelta1wrtmu20*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu20)/delta0cubed
lam34LastTermSDiff=k*(DiffOfdelta1wrtlam34*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtlam34)/delta0cubed
mu32LastTermSDiff=k*(DiffOfdelta1wrtmu32*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu32)/delta0cubed
mu31LastTermSDiff=k*(DiffOfdelta1wrtmu31*sqrdelta0cubed-
delta1*DiffsqrtOfdelta0cubedwrtmu31)/delta0cubed

% /// NOW CALCULATE DIFFERENTIATION OF S WITH W.R.T. TO EACH RATE:
% S'=(1/4)*h*(differentiation of first term + differentiation of second term
% + differentiation of third term)
% S'=1/4*{reciprocal of sqrt[(-2p)/3 + 2/3*sqrt(Δ0)*cos (Ø/3)]}*{(-2p')/3+
2/3*[sqrt(Δ0)]'*cos (Ø/3) + 2/3*sqrt(Δ0)*[cos (Ø/3)]'}
% where {reciprocal of sqrt[(-2p)/3 + 2/3*sqrt(Δ0)*cos (Ø/3)]}=h
% sqrt[2/3*(-1*p)+sqrt(Δ0)*cos (Ø/3)]=sqrt{2/3*[(-1*p)+ sqrt(Δ0)*m]}= n where
% cos (Ø/3)=m , and thus h=1/n

n=sqrt(2/3*(-1*p+sqrdelta0*m))
h=1/n
diffSwrtlam01=.25*h*(lam01FirstTermSDiff+lam01SecondTermSDiff+lam01LastTermSD
iff)
diffSwrtlam12=.25*h*(lam12FirstTermSDiff+lam12SecondTermSDiff+lam12LastTermSD
iff)
diffSwrtmu10=.25*h*(mu10FirstTermSDiff+mu10SecondTermSDiff+mu10LastTermSDiff)
diffSwrtlam23=.25*h*(lam23FirstTermSDiff+lam23SecondTermSDiff+lam23LastTermSD
iff)
diffSwrtmu21=.25*h*(mu21FirstTermSDiff+mu21SecondTermSDiff+mu21LastTermSDiff)
diffSwrtmu20=.25*h*(mu20FirstTermSDiff+mu20SecondTermSDiff+mu20LastTermSDiff)

```

```

diffSwrtlam34=.25*h*(lam34FirstTermSDiff+lam34SecondTermSDiff+lam34LastTermSD
iff)
diffSwrtmu32=.25*h*(mu32FirstTermSDiff+mu32SecondTermSDiff+mu32LastTermSDiff)
diffSwrtmu31=.25*h*(mu31FirstTermSDiff+mu31SecondTermSDiff+mu31LastTermSDiff)

% NOW differentiate roots:
% diff root1= -b/4 - s'+ .25 (-4s^2-2p+q/s)^-.5 *(-8ss'-2p'+ (q'*s-s'*q)/s^2)
% diff root2= -b/4 - s'- .25 (-4s^2-2p+q/s)^-.5 *(-8ss'-2p'+ (q'*s-s'*q)/s^2)
% diff root3= -b/4 + s'+ .25 (-4s^2-2p-q/s)^-.5 *(-8ss'-2p'- (q'*s-s'*q)/s^2)
% diff root4= -b/4 + s'- .25 (-4s^2-2p-q/s)^-.5 *(-8ss'-2p'- (q'*s-s'*q)/s^2)

% root 1
f=1/sqrt(-4*s^2-2*p+(q/s));
fNeg=1/sqrt(-4*s^2-2*p-(q/s));

diffroot1wrtlam01=-(diffBwrtlam01/4)-diffSwrtlam01+.25*f*(-8*s*diffSwrtlam01-
2*diffPwrtlam01+((diffQwrtlam01*s-diffSwrtlam01*q)/s^2))
diffroot1wrtlam12=-(diffBwrtlam12/4)-diffSwrtlam12+.25*f*(-8*s*diffSwrtlam12-
2*diffPwrtlam12+((diffQwrtlam12*s-diffSwrtlam12*q)/s^2))
diffroot1wrtmu10=-(diffBwrtmu10/4)-diffSwrtmu10+.25*f*(-8*s*diffSwrtmu10-
2*diffPwrtmu10+((diffQwrtmu10*s-diffSwrtmu10*q)/s^2))
diffroot1wrtlam23=-(diffBwrtlam23/4)-diffSwrtlam23+.25*f*(-8*s*diffSwrtlam23-
2*diffPwrtlam23+((diffQwrtlam23*s-diffSwrtlam23*q)/s^2))
diffroot1wrtmu21=-(diffBwrtmu21/4)-diffSwrtmu21+.25*f*(-8*s*diffSwrtmu21-
2*diffPwrtmu21+((diffQwrtmu21*s-diffSwrtmu21*q)/s^2))
diffroot1wrtmu20=-(diffBwrtmu20/4)-diffSwrtmu20+.25*f*(-8*s*diffSwrtmu20-
2*diffPwrtmu20+((diffQwrtmu20*s-diffSwrtmu20*q)/s^2))
diffroot1wrtlam34=-(diffBwrtlam34/4)-diffSwrtlam34+.25*f*(-8*s*diffSwrtlam34-
2*diffPwrtlam34+((diffQwrtlam34*s-diffSwrtlam34*q)/s^2))
diffroot1wrtmu32=-(diffBwrtmu32/4)-diffSwrtmu32+.25*f*(-8*s*diffSwrtmu32-
2*diffPwrtmu32+((diffQwrtmu32*s-diffSwrtmu32*q)/s^2))
diffroot1wrtmu31=-(diffBwrtmu31/4)-diffSwrtmu31+.25*f*(-8*s*diffSwrtmu31-
2*diffPwrtmu31+((diffQwrtmu31*s-diffSwrtmu31*q)/s^2))

% root2
diffroot2wrtlam01=-(diffBwrtlam01/4)-diffSwrtlam01-.25*f*(-8*s*diffSwrtlam01-
2*diffPwrtlam01+((diffQwrtlam01*s-diffSwrtlam01*q)/s^2))
diffroot2wrtlam12=-(diffBwrtlam12/4)-diffSwrtlam12-.25*f*(-8*s*diffSwrtlam12-
2*diffPwrtlam12+((diffQwrtlam12*s-diffSwrtlam12*q)/s^2))
diffroot2wrtmu10=-(diffBwrtmu10/4)-diffSwrtmu10-.25*f*(-8*s*diffSwrtmu10-
2*diffPwrtmu10+((diffQwrtmu10*s-diffSwrtmu10*q)/s^2))
diffroot2wrtlam23=-(diffBwrtlam23/4)-diffSwrtlam23-.25*f*(-8*s*diffSwrtlam23-
2*diffPwrtlam23+((diffQwrtlam23*s-diffSwrtlam23*q)/s^2))
diffroot2wrtmu21=-(diffBwrtmu21/4)-diffSwrtmu21-.25*f*(-8*s*diffSwrtmu21-
2*diffPwrtmu21+((diffQwrtmu21*s-diffSwrtmu21*q)/s^2))
diffroot2wrtmu20=-(diffBwrtmu20/4)-diffSwrtmu20-.25*f*(-8*s*diffSwrtmu20-
2*diffPwrtmu20+((diffQwrtmu20*s-diffSwrtmu20*q)/s^2))
diffroot2wrtlam34=-(diffBwrtlam34/4)-diffSwrtlam34-.25*f*(-8*s*diffSwrtlam34-
2*diffPwrtlam34+((diffQwrtlam34*s-diffSwrtlam34*q)/s^2))
diffroot2wrtmu32=-(diffBwrtmu32/4)-diffSwrtmu32-.25*f*(-8*s*diffSwrtmu32-
2*diffPwrtmu32+((diffQwrtmu32*s-diffSwrtmu32*q)/s^2))
diffroot2wrtmu31=-(diffBwrtmu31/4)-diffSwrtmu31-.25*f*(-8*s*diffSwrtmu31-
2*diffPwrtmu31+((diffQwrtmu31*s-diffSwrtmu31*q)/s^2))

% root 3 :

```

```

diffroot3wrtlam01=-(diffBwrtlam01/4)+diffSwrtlam01+.25*fNeg*(-
8*s*diffSwrtlam01-2*diffPwrtlam01-((diffQwrtlam01*s-diffSwrtlam01*q)/s^2))
diffroot3wrtlam12=-(diffBwrtlam12/4)+diffSwrtlam12+.25*fNeg*(-
8*s*diffSwrtlam12-2*diffPwrtlam12-((diffQwrtlam12*s-diffSwrtlam12*q)/s^2))
diffroot3wrtmu10=-(diffBwrtmu10/4)+diffSwrtmu10+.25*fNeg*(-8*s*diffSwrtmu10-
2*diffPwrtmu10-((diffQwrtmu10*s-diffSwrtmu10*q)/s^2))
diffroot3wrtlam23=-(diffBwrtlam23/4)+diffSwrtlam23+.25*fNeg*(-
8*s*diffSwrtlam23-2*diffPwrtlam23-((diffQwrtlam23*s-diffSwrtlam23*q)/s^2))
diffroot3wrtmu21=-(diffBwrtmu21/4)+diffSwrtmu21+.25*fNeg*(-8*s*diffSwrtmu21-
2*diffPwrtmu21-((diffQwrtmu21*s-diffSwrtmu21*q)/s^2))
diffroot3wrtmu20=-(diffBwrtmu20/4)+diffSwrtmu20+.25*fNeg*(-8*s*diffSwrtmu20-
2*diffPwrtmu20-((diffQwrtmu20*s-diffSwrtmu20*q)/s^2))
diffroot3wrtlam34=-(diffBwrtlam34/4)+diffSwrtlam34+.25*fNeg*(-
8*s*diffSwrtlam34-2*diffPwrtlam34-((diffQwrtlam34*s-diffSwrtlam34*q)/s^2))
diffroot3wrtmu32=-(diffBwrtmu32/4)+diffSwrtmu32+.25*fNeg*(-8*s*diffSwrtmu32-
2*diffPwrtmu32-((diffQwrtmu32*s-diffSwrtmu32*q)/s^2))
diffroot3wrtmu31=-(diffBwrtmu31/4)+diffSwrtmu31+.25*fNeg*(-8*s*diffSwrtmu31-
2*diffPwrtmu31-((diffQwrtmu31*s-diffSwrtmu31*q)/s^2))

```

```
% root 4 :
```

```

diffroot4wrtlam01=-(diffBwrtlam01/4)+diffSwrtlam01-.25*fNeg*(-
8*s*diffSwrtlam01-2*diffPwrtlam01-((diffQwrtlam01*s-diffSwrtlam01*q)/s^2))
diffroot4wrtlam12=-(diffBwrtlam12/4)+diffSwrtlam12-.25*fNeg*(-
8*s*diffSwrtlam12-2*diffPwrtlam12-((diffQwrtlam12*s-diffSwrtlam12*q)/s^2))
diffroot4wrtmu10=-(diffBwrtmu10/4)+diffSwrtmu10-.25*fNeg*(-8*s*diffSwrtmu10-
2*diffPwrtmu10-((diffQwrtmu10*s-diffSwrtmu10*q)/s^2))
diffroot4wrtlam23=-(diffBwrtlam23/4)+diffSwrtlam23-.25*fNeg*(-
8*s*diffSwrtlam23-2*diffPwrtlam23-((diffQwrtlam23*s-diffSwrtlam23*q)/s^2))
diffroot4wrtmu21=-(diffBwrtmu21/4)+diffSwrtmu21-.25*fNeg*(-8*s*diffSwrtmu21-
2*diffPwrtmu21-((diffQwrtmu21*s-diffSwrtmu21*q)/s^2))
diffroot4wrtmu20=-(diffBwrtmu20/4)+diffSwrtmu20-.25*fNeg*(-8*s*diffSwrtmu20-
2*diffPwrtmu20-((diffQwrtmu20*s-diffSwrtmu20*q)/s^2))
diffroot4wrtlam34=-(diffBwrtlam34/4)+diffSwrtlam34-.25*fNeg*(-
8*s*diffSwrtlam34-2*diffPwrtlam34-((diffQwrtlam34*s-diffSwrtlam34*q)/s^2))
diffroot4wrtmu32=-(diffBwrtmu32/4)+diffSwrtmu32-.25*fNeg*(-8*s*diffSwrtmu32-
2*diffPwrtmu32-((diffQwrtmu32*s-diffSwrtmu32*q)/s^2))
diffroot4wrtmu31=-(diffBwrtmu31/4)+diffSwrtmu31-.25*fNeg*(-8*s*diffSwrtmu31-
2*diffPwrtmu31-((diffQwrtmu31*s-diffSwrtmu31*q)/s^2))

```

```
% calculate t*exp(rooti*t)*diffOfRoot with respect to each theta let's call
% rootiagain
```

```

t=1;
eigenValuesVector= eig(Q1);
r1=-(b/4)-s+.5*sqrt(-4*s^2-2*p+(q/s));
r2=-(b/4)-s-.5*sqrt(-4*s^2-2*p+(q/s));
r3=-(b/4)+s+.5*sqrt(-4*s^2-2*p-(q/s));
r4=-(b/4)+s-.5*sqrt(-4*s^2-2*p-(q/s));

```

```
% rootlagain
```

```

rootlagainwrtlam01=t*exp(r1*t)*diffroot1wrtlam01
rootlagainwrtlam12=t*exp(r1*t)*diffroot1wrtlam12
rootlagainwrtmu10=t*exp(r1*t)*diffroot1wrtmu10
rootlagainwrtlam23=t*exp(r1*t)*diffroot1wrtlam23
rootlagainwrtmu21=t*exp(r1*t)*diffroot1wrtmu21
rootlagainwrtmu20=t*exp(r1*t)*diffroot1wrtmu20

```

```

rootlagainwrtlam34=t*exp(r1*t)*diffroot1wrtlam34
rootlagainwrtmu32=t*exp(r1*t)*diffroot1wrtmu32
rootlagainwrtmu31=t*exp(r1*t)*diffroot1wrtmu31

% root2again
root2againwrtlam01=t*exp(r2*t)*diffroot2wrtlam01
root2againwrtlam12=t*exp(r2*t)*diffroot2wrtlam12
root2againwrtmu10=t*exp(r2*t)*diffroot2wrtmu10
root2againwrtlam23=t*exp(r2*t)*diffroot2wrtlam23
root2againwrtmu21=t*exp(r2*t)*diffroot2wrtmu21
root2againwrtmu20=t*exp(r2*t)*diffroot2wrtmu20
root2againwrtlam34=t*exp(r2*t)*diffroot2wrtlam34
root2againwrtmu32=t*exp(r2*t)*diffroot2wrtmu32
root2againwrtmu31=t*exp(r2*t)*diffroot2wrtmu31

% root3again
root3againwrtlam01=t*exp(r3*t)*diffroot3wrtlam01
root3againwrtlam12=t*exp(r3*t)*diffroot3wrtlam12
root3againwrtmu10=t*exp(r3*t)*diffroot3wrtmu10
root3againwrtlam23=t*exp(r3*t)*diffroot3wrtlam23
root3againwrtmu21=t*exp(r3*t)*diffroot3wrtmu21
root3againwrtmu20=t*exp(r3*t)*diffroot3wrtmu20
root3againwrtlam34=t*exp(r3*t)*diffroot3wrtlam34
root3againwrtmu32=t*exp(r3*t)*diffroot3wrtmu32
root3againwrtmu31=t*exp(r3*t)*diffroot3wrtmu31

% root4again
root4againwrtlam01=t*exp(r4*t)*diffroot4wrtlam01
root4againwrtlam12=t*exp(r4*t)*diffroot4wrtlam12
root4againwrtmu10=t*exp(r4*t)*diffroot4wrtmu10
root4againwrtlam23=t*exp(r4*t)*diffroot4wrtlam23
root4againwrtmu21=t*exp(r4*t)*diffroot4wrtmu21
root4againwrtmu20=t*exp(r4*t)*diffroot4wrtmu20
root4againwrtlam34=t*exp(r4*t)*diffroot4wrtlam34
root4againwrtmu32=t*exp(r4*t)*diffroot4wrtmu32
root4againwrtmu31=t*exp(r4*t)*diffroot4wrtmu31

% sum the 4 row entries of the score function :
sumSCORE1=rootlagainwrtlam01+root2againwrtlam01+root3againwrtlam01+root4again
wrtlam01
sumSCORE2=rootlagainwrtlam12+root2againwrtlam12+root3againwrtlam12+root4again
wrtlam12
sumSCORE3=rootlagainwrtmu10+root2againwrtmu10+root3againwrtmu10+root4againwrt
mu10
sumSCORE4=rootlagainwrtlam23+root2againwrtlam23+root3againwrtlam23+root4again
wrtlam23
sumSCORE5=rootlagainwrtmu21+root2againwrtmu21+root3againwrtmu21+root4againwrt
mu21
sumSCORE6=rootlagainwrtmu20+root2againwrtmu20+root3againwrtmu20+root4againwrt
mu20
sumSCORE7=rootlagainwrtlam34+root2againwrtlam34+root3againwrtlam34+root4again
wrtlam34
sumSCORE8=rootlagainwrtmu32+root2againwrtmu32+root3againwrtmu32+root4againwrt
mu32
sumSCORE9=rootlagainwrtmu31+root2againwrtmu31+root3againwrtmu31+root4againwrt
mu31

```

```

% construct score function or score vector(column vector )
scoreFunction =[sumSCORE1;sumSCORE2;sumSCORE3;sumSCORE4;
                sumSCORE5;sumSCORE6;sumSCORE7;sumSCORE8;
                sumSCORE9]

% scale the score function by 16920
VectorScalar=16920
scaledScoreFunction=16920*scoreFunction

% construct the M matrix (hessian matrix), scale this by 1230084 then invert
% the result
MatrixScalar=1230084
M_matrix=scaledScoreFunction*scaledScoreFunction'
scaledM_matrix=MatrixScalar*M_matrix

% //////////////////////////////////////
% HINT at the end of the code , there are the matrix of transition counts and
the
% functional routine to obtain the scalar for the score function and hessian
matrix
% //////////////////////////////////////

% scaledM_matrix is singular or near singular so calculate pseudoinverse
invertM_matrix=pinv(scaledM_matrix)

% apply quasiNewton formula :
initialRateVector=[lam01;lam12;mu10;lam23;mu21;mu20;lam34;mu32;mu31]
desiredVector1= initialRateVector+(invertM_matrix*scaledScoreFunction)

% //////////////////////////////////////
% In the following few lines , a code to calculate the scalar for score
% function and hessian matrix:
% Q3 is the matrix of transition counts among the states,to calculate the
% scalar for the score function it is the total row count for each cell where
the count is nonzero
% scalar for the hessian matrix is the square of the total row divided by
% the cell count where the counts are nonzeros .

Q3=[1909      120      15      6      0 ;
     36      1116     67      28      0 ;
     13       30     703      37      0 ;
     11       14      23      50     22 ;
      0        0       0       0      0]

% calculate the total for each row let's call it totsum which is a column
vector that contains in each row
% the total counts of transitions from this state,
% for example 2050 is the total number of transitions from state 0 ,
% 1247 is the total number of transitions from state 1 , 783 is the total
% number of transitions from state 2 and 120 is the total number of
% transitions from state 3

totSum=sum(Q3') % calculate the sum of each row to get the total sum

```

```

n1=totSum(1,1) % extract the total of the first row
n2=totSum(1,2) % extract the total of the second row
n3=totSum(1,3) % extract the total of the third row
n4=totSum(1,4) % extract the total of the fourth row
n5=totSum(1,5) % extract the total of the fifth row

% NOW calculate the number of cells with nonzero counts in each row

s1=sum(Q3(1,:)>0) % count the total number of cells with nonzero counts in
row1
s2=sum(Q3(2,:)>0) % count the total number of cells with nonzero counts in
row2
s3=sum(Q3(3,:)>0) % count the total number of cells with nonzero counts in
row3
s4=sum(Q3(4,:)>0) % count the total number of cells with nonzero counts in
row4
s5=sum(Q3(5,:)>0) % count the total number of cells with nonzero counts in
row5

% calculate the scoreScalar value

nVector=[n1 n2 n3 n4 n5] % total marginal for each row sum over j for nij
in each row
sVector=[s1 s2 s3 s4 s5] % number of cells in each row with nonzero
counts
scoreScalar=n1*s1+n2*s2+n3*s3+n4*s4+n5*s5

% extract each row from the matrix of the transition counts

v1=Q3(1,1:5)
v2=Q3(2,1:5)
v3=Q3(3,1:5)
v4=Q3(4,1:5)
v5=Q3(5,1:5)

% extract cells with nonzero counts in each row

V1=v1(v1>0)
V2=v2(v2>0)
V3=v3(v3>0)
V4=v4(v4>0)
V5=v5(v5>0)

% calculate the scalar for hessian matrix ; by squaring the total marginal
% for each row and divide this total marginal by each cell count in this
% row and then sum up the results. repeat this procedure in each row and
% sum up the final results in all rows together.
hessianMatrixscalar=sum(n1^2./V1)+sum(n2^2./V2)+sum(n3^2./V3
)+sum(n4^2./V4)+sum(n5^2./V5)

```

This is the MATLAB code of how to construct the transition count table in each interval according to the timeline of the patients in the excel sheet :

```
clear
clc
close all

% // this is a code to calculate the observed transition counts among
% states

% first load excel sheet as a matrix ( remove any vertical or horizontal
% headers ) , then extract each row as a vector to count the transitions in
% the interval ( here we have interval is one year ).

data = xlsread('CovariateModel.xlsx','Sheet1');
class(data)

[row,col]=size(data)
x=1:row
y=1:col

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/
% Now calculate the transitions among states in the whole matrix in delta t =
one year interval
% to overcome the problem of index out of bound , add multiple of -1 and
subtract this multiple as
% shown below

count=0
C00=0
C01=0
C02=0
C03=0
C04=0
C10=0
C11=0
C12=0
C13=0
C14=0
C20=0
C21=0
C22=0
C23=0
C24=0
C30=0
```

```

C31=0
C32=0
C33=0
C34=0

for x=1:row
    count=count+1
    a1=data(x,:)
    k=1:length(a1)
    a11=[a1  -1 ]

for k=1:length(a11)-1
    if a11(k)==0 && a11(k+1)==0
        C00=C00+1
    end
end

for k=1:length(a11)-1
    if a11(k)==0 && a11(k+1)==1
        C01=C01+1
    end
end

for k=1:length(a11)-1
    if a11(k)==0 && a11(k+1)==2
        C02=C02+1
    end
end

for k=1:length(a11)-1
    if a11(k)==0 && a11(k+1)==3
        C03=C03+1
    end
end

for k=1:length(a11)-1
    if a11(k)==0 && a11(k+1)==4
        C04=C04+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==0
        C10=C10+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==1
        C11=C11+1
    end
end

for k=1:length(a11)-1
    if a11(k)==1 && a11(k+1)==2

```

```

        C12=C12+1
    end
end
    for k=1:length(a11)-1
        if a11(k)==1 && a11(k+1)==3
            C13=C13+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==1 && a11(k+1)==4
            C14=C14+1
        end
    end
    for k=1:length(a11)-1
        if a11(k)==2 && a11(k+1)==0
            C20=C20+1
        end
    end
    for k=1:length(a11)-1
        if a11(k)==2 && a11(k+1)==1
            C21=C21+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==2 && a11(k+1)==2
            C22=C22+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==2 && a11(k+1)==3
            C23=C23+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==2 && a11(k+1)==4
            C24=C24+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==3 && a11(k+1)==0
            C30=C30+1
        end
    end

    for k=1:length(a11)-1
        if a11(k)==3 && a11(k+1)==1
            C31=C31+1
        end
    end

```

```

end

for k=1:length(a11)-1
    if a11(k)==3 && a11(k+1)==2
        C32=C32+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3 && a11(k+1)==3
        C33=C33+1
    end
end

for k=1:length(a11)-1
    if a11(k)==3 && a11(k+1)==4
        C34=C34+1
    end
end

Ct1=[ C00 C01 C02 C03 C04      ; C10 C11 C12 C13 C14 ; C20 C21 C22 C23 C24  ;
      C30 C31 C32 C33 C34 ]

end

Ctot=Ct1
sum(Ct1')

```

STATA CODE :

codeocean.com/capsule/4752445/tree/v3

Published prognostic factors for evolution of non alcoholic fatty liver disease patients utilizing Poisson regression a... (iman attia)

Capsule File Help

Files

Core Files

▶ metadata 1.14 KB ✓

▶ environment 219 B ✓

▼ code 17.73 KB ✓

LICENSE 1.03 KB ✓

poisson2.do 16.52 KB ✓

run 177 B ✓

▼ data Manage Datasets 41.77 KB

LICENSE 18.21 KB

patients' data.dta 23.55 KB

.gitignore 7 B ✓

Results

▶ results 83.05 KB

Other Files

Prognostic Factors Fatty Liver ... 1.17 MB ✓

results after running poisson... 954.87 KB ✓

supplementary materials for pr... 2.69 MB ✓

Metadata

Bioinformatics

prognostic factors for evolution of non alcoholic fatty liver disease patients utilizing Poisson regression and continuous time Markov chains

iman attia

This capsule uses Poisson regression to relate the transition among states of fatty liver disease (early reversible stages) to the covariates or risk factors of the disease like : type 2 diabetes, hypercholesterolemia, and hypertension . It contains hypothetical data of 150 patients followed up every year for 28 years . the demographic data are shown in stata file (.dta) and the time line of the stages of the disease are shown in associated pdf file . Also pdf files are used to illustrate the idea and the mathematical and statistical concepts are provided. In addition , there is PDF file that contains the stata output results of all the routines or code statements in the do file .

biostatistics

continuous time Markov chain

non-alcoholic fatty liver disease

insulin resistance

Poisson regression

Panel data analysis

hypercholesterolemia

hypertension

Capsule

DOI [10.24433/CO.8778229.v3](https://doi.org/10.24433/CO.8778229.v3)

Citation

General

Iman attia (2021) prognostic factors for evolution of non alcoholic fatty liver disease patients utilizing Poisson regression and continuous time Markov chains [Source Code].
<https://doi.org/10.24433/CO.8778229.v3>

Licenses

Code MIT license

Data Attribution (CC BY)

MATLAB code :

codeocean.com/capsule/6377472/tree/v2

Using CTMC to estimate the rates of fibrosis progression in a subset of states of NAFLD model (Iman Mohammed Attia Abd-Elkhalik)

Capsule File Help

Files

Tabs

Core Files

metadata

1.12 KB

✓

environment

222 B

✓

code

54.63 KB

✓

CovariateModel.xlsx

19.44 KB

✓

fibrosis1.m

30.44 KB

✓

LICENSE

1.03 KB

✓

run

222 B

✓

TRANSITIONcountsCOVARIA...

3.48 KB

✓

data

Manage Datasets

0 B

✓

Results

225.36 KB

Other Files

Prognostic Factors Fatty Liver ...

1.17 MB

✓

rates.pdf

734.7 KB

✓

supplementary materials for pr...

2.69 MB

✓

Metadata

Bioinformatics

Using CTMC to estimate the rates of fibrosis progression in a subset of states of NAFLD model

Iman Mohammed Attia Abd-Elkhalik

A subset of states that represents the rate of fibrosis progression in non-alcoholic fatty liver disease patients with high risk factors such as type 2 diabetes, obesity, hypercholesterolemia and hypertension is analyzed using continuous time homogenous Markov chains to estimate the rate of transition between states. the states are F0, F1, F2, F3, F4 which is liver cirrhosis (advanced form of fibrosis). In this capsule code, it is shown how the estimated rates using CTMC are approximately equal to the observed rates and subsequently these rates can be used in Poisson regression model to relate the risk factors with these rates. The observed rates can be used directly in Poisson regression with no need to estimate the rates with CTMC.

bistatistics

bioinformatics

non-alcoholic fatty liver disease

applied mathematical statistics

hepatology

Capsule

DOI

[10.24433/CO.2144346.v2](https://doi.org/10.24433/CO.2144346.v2)

Citation

General

Iman Mohammed Attia Abd-Elkhalik (2022) Using CTMC to estimate the rates of fibrosis progression in a subset of states of NAFLD model [Source Code]. <https://doi.org/10.24433/CO.2144346.v2>

Licenses

Code MIT license

Datasets

Standard Datasets

FIBROSIS IN NAFLD

View

Edit

☆☆☆☆☆ 0 ratings - click the stars to submit your rating

Citation Author(s): Iman Attia Abd-Elkhalik Abo-Elreesh (Institute of Statistical Studies and Research, Cairo University)

Submitted by: Iman Attia

Last updated: Sun, 02/13/2022 - 11:38

DOI: 10.21227/dr5j-gs46

Data Format: dta

License: Creative Commons Attribution

50 Views

Categories: Biomedical and Health Sciences

Keywords: log linear model, continuous time Markov chains, Poisson regression

ACCESS DATASET

CITE

SHARE/EMBED

ABSTRACT

one hundred fifty patients were followed up every year for 28 years, and at each visit the characteristics of the patients were recorded (these are the predictors) like: sex(0= female, 1= male) , age, BMI, LDL-cho, HOMA2-IR, systolic blood pressure and diastolic blood pressure. For each participant, the recorded value is the mean of the follow up measurements. The age is

DATASET FILES

fibrosis in NAFLD (7.16 KB)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- [SupplementarymaterialsAppendixA.docx](#)
- [SupplementarymaterialsAppendixB.docx](#)
- [SupplementarymaterialsAppendixC.docx](#)