Calculus 3 for Computer Science Project

Neil Vohra, Amol Patel & Bhavani Jaladanki

Professor: Dr. Luz Vela-Arevalo

March 31, 2015

Part 1

Discussion:

Why is it justified to use the LU or QR-factorizations as opposed of calculating an inverse matrix?

Calculating the inverse of a matrix without either of these methods is a tedious task that, when using Gauss – Jordan elimination/row reduction, requires many calculations, so using either LU or QR decomposition greatly speeds up calculation time by requiring fewer operations to be done. For LU decomposition (after L and U are found), finding the inverse is a simple matter of forward and backward substitution "n" times with the identity matrix (requires $O(n^2)$ calculations). For QR decomposition (once Q and R have been calculated), the inverse is calculated via $R^{-1} * Q^t$, which is a simple calculation since R is an upper triangular matrix (also requires $O(n^2)$ calculations). Something else to note for QR decomposition is that, on average, QR decomposition via Givens rotations requires twice as many calculations as Householder reflections. So this may be something to note for calculations on a much larger scale (Linear Algebra: Numerical Methods, p. 43).

In addition to this speed up in computation time, error/conditioning is also a huge factor in why LU and QR decompositions are preferred over the more traditional way of calculating the inverse of a matrix, and this factor is discussed below.

What is the benefit of using LU or QR-factorizations in this way?

Finding the inverse of a matrix using Gauss – Jordan elimination/row reduction can be unstable because this method can greatly amplify any error already present in the problem from the first step; even a well-conditioned problem can turn into an ill conditioned problem. One can use attempt to reduce the error that results from this method by having the largest value of each column at or below the diagonal be a pivot in the matrix (since this way, the row operations have to use some multiplier less than one when row reducing, therefore error is amplified to a lesser extent), but LU and QR decompositions provide a much better alternative.

On LU decomposition, the page 10 of Linear Algebra: Numerical Methods mentions the following equation: "$\text{cond}(A) = \text{cond}(LU) \leq \text{cond}(L)\text{cond}(U)$" What this equation means is that the amount of error in the original matrix is, at most, equal to the condition of L and U multiplied together. Therefore, we can observe that LU decomposition does not introduce any error into our computations, so we only have to worry about the error that is inherent in our original problem.

On QR decomposition, this method gives a huge benefit to calculating the inverse in terms error amplification. Given a matrix A whose inverse is being calculated, the Webnotes (Linear Algebra: Numerical Methods, p. 1) supplies the following equations:

$$A = Q * R$$

$$A^{-1} = R^{-1} * Q^{-1}$$

$$||A^{-1}|| = ||R^{-1}|| * ||Q^{-1}||$$

Because Q is an orthogonal matrix, its inverse is equal to Q^t. Therefore, the condition number is calculated by the following:

$$\text{cond}(Q) = \sigma_{\max} / \sigma_{\min} = \sqrt{(\lambda_{\max}(Q^t * Q))} / \sqrt{(\lambda_{\min}(Q^t * Q))} = 1$$

Therefore, there is no error added to our calculations involving Q. So in terms of the condition number, cond(A) = cond(R). . Minimal error amplification arises in computations involving R (namely Rx = y). And to be clear, the computation of QR, when using Householder reflections or Givens rotations, doesn't add any error as well since $Q_n * ... * Q_1 * A = R$, and we just established that the condition number of Q_n is equal to 1. This method of calculating an inverse is considered to be very stable since it uses orthogonal transformations.

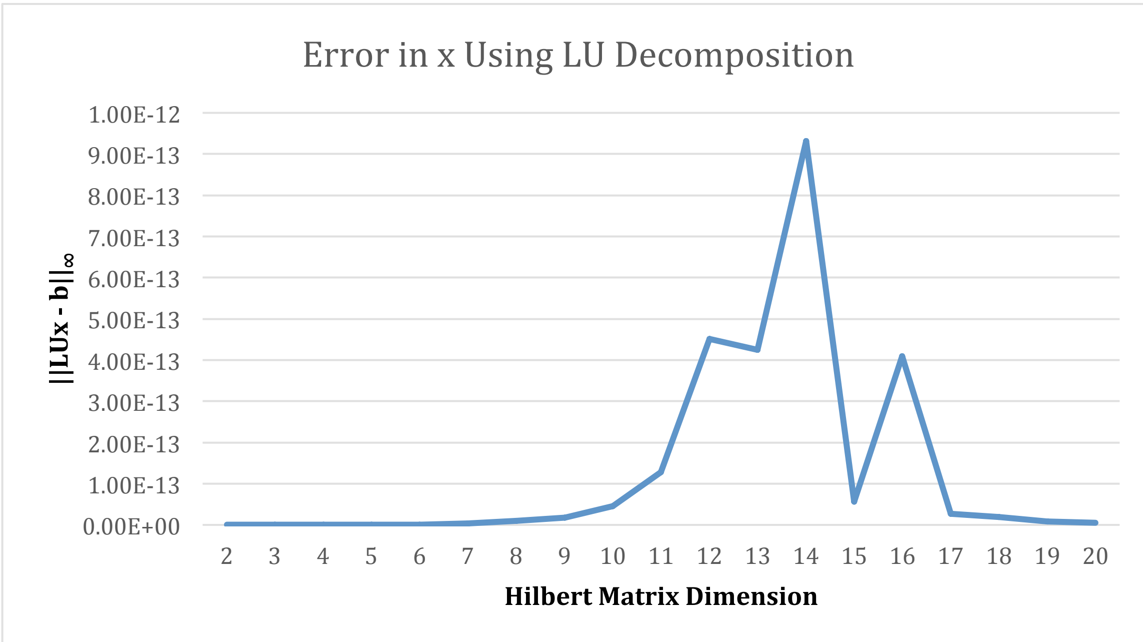
To compare the two methods of QR decomposition, it is worth noting the following observations taken from Linear Algebra: Numerical Methods (p. 43):

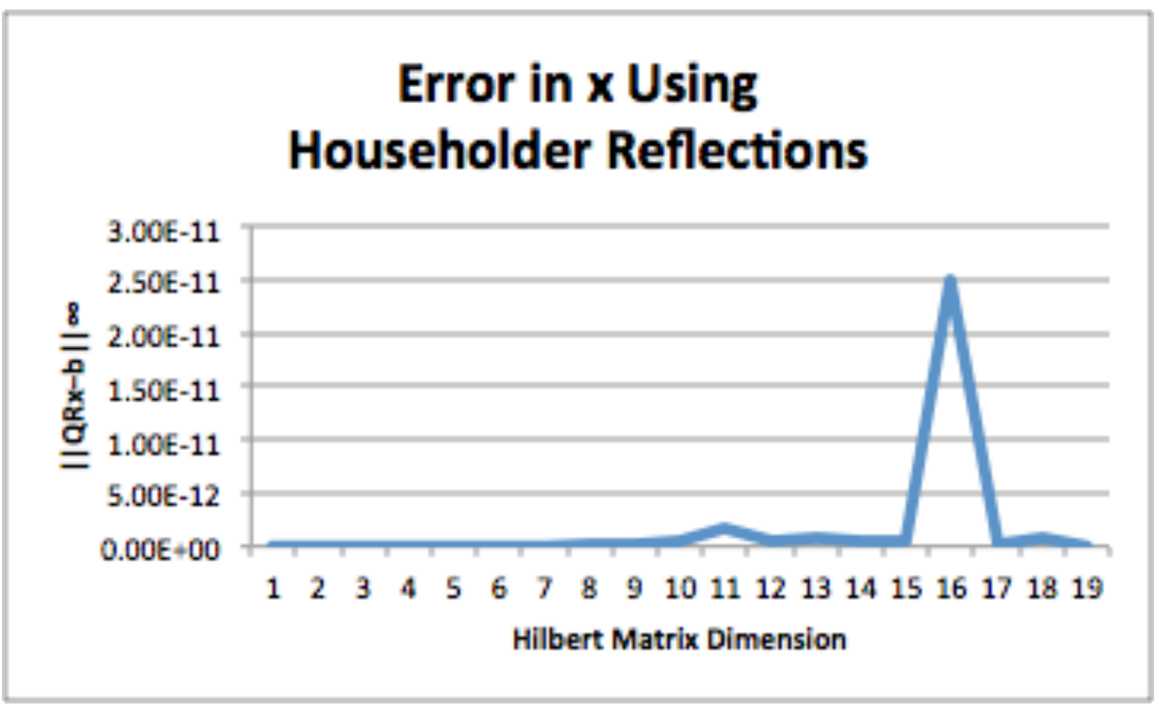
(i) Householder is faster, especially for larger matrices, but

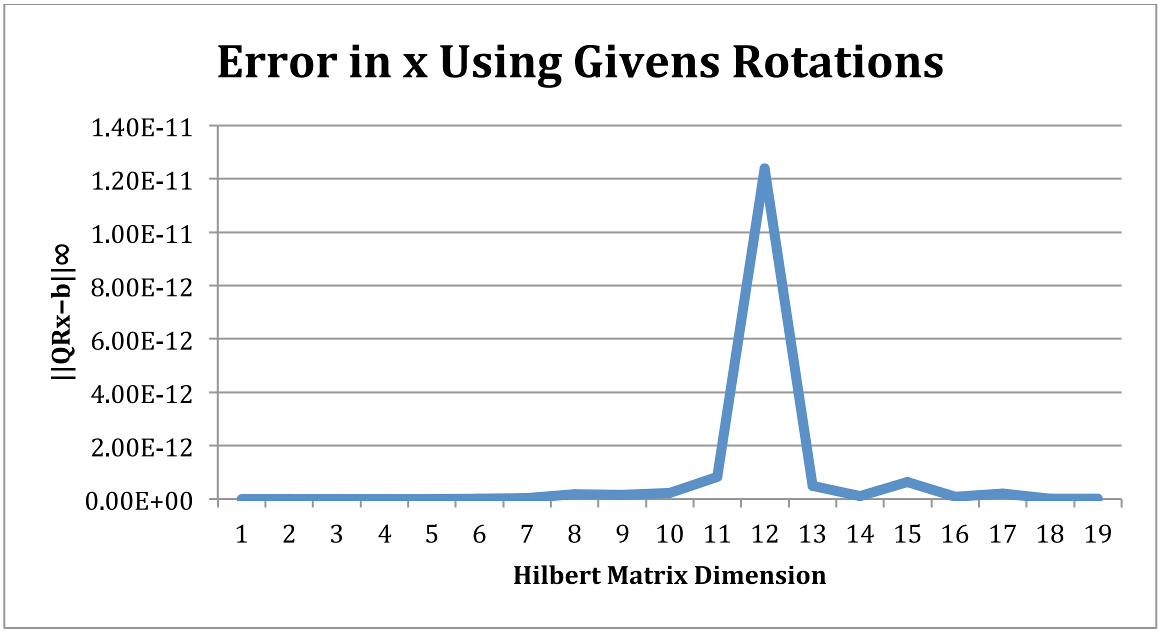
(ii) Givens is slightly more accurate.

The reason is that Householder is a "greedier" algorithm: it tries to zero more elements at the same time. Hence it is faster, but "lousier". Givens is a slow but more accurate algorithm. However, the error is in fact almost negligible in both cases.

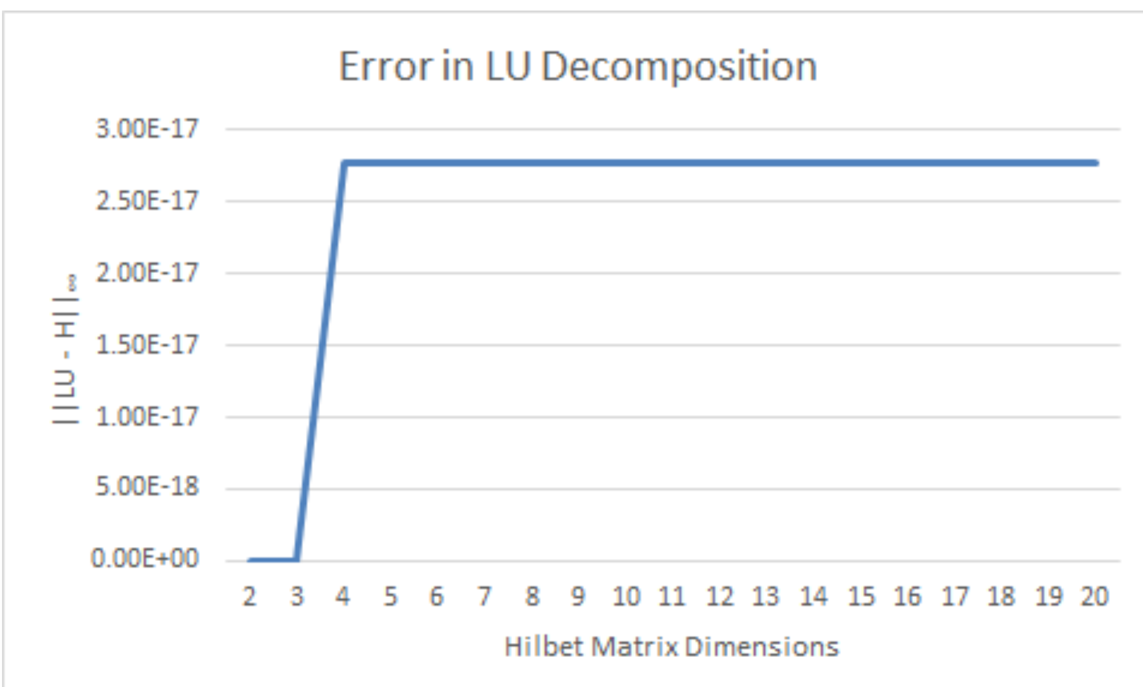
Below are the plots requested for part 1 of the project.

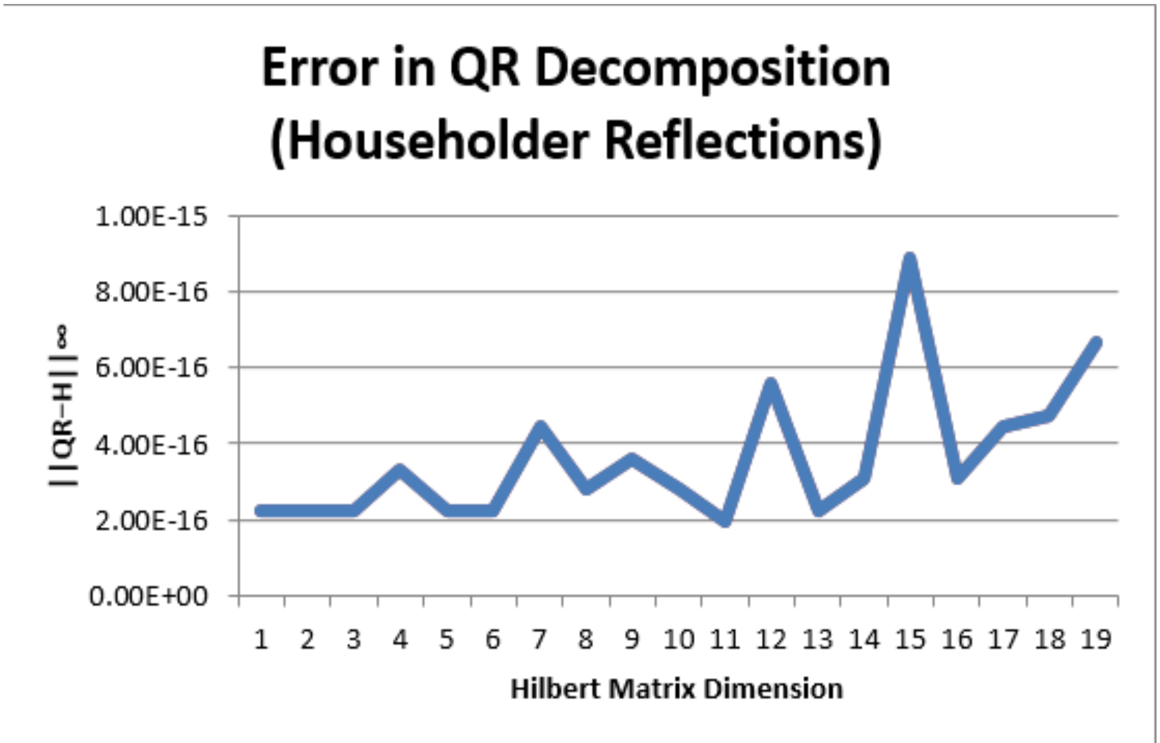


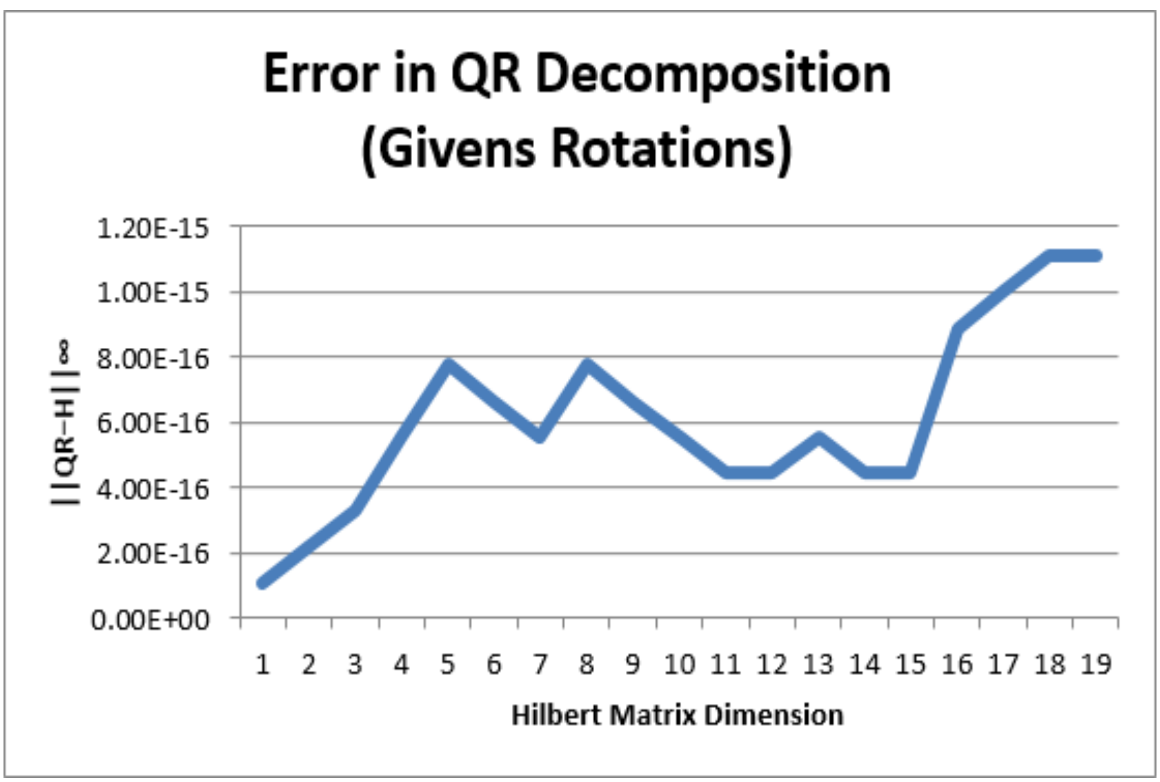




All three methods have a big increase in error some Hilbert matrix size of 12 or higher, and this is most likely because Hilbert matrices are ill conditioned. Therefore, this error could not have been avoided. But the error for the other values of the Hilbert matrix dimensions is incredibly low and near zero.




Error in QR Decomposition
(Householder Reflections)
1.00E-15
8.00E-16
6.00E-16
4.00E-16
2.00E-16
0.00E+00
||QR−H||∞
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
Hilbert Matrix Dimension


Error in QR Decomposition
(Givens Rotations)
1.20E-15
1.00E-15
8.00E-16
6.00E-16
4.00E-16
2.00E-16
0.00E+00
||QR−H||∞
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
Hilbert Matrix Dimension

As a conclusion to this set of graphs, LU decomposition appears to be the most stable way of decomposing Hilbert matrices while using Givens rotations appears to be just slightly more stable than using Householder reflections (for reasons that are addressed in the two questions at the beginning). Although there is this small difference in the error between Givens rotations and Householder reflections, the error is negligible. Therefore, these results are consistent with the expected results.

Part 2

Discussion:

Gauss-Seidal is a method used to iteratively solve linear systems in form Ax=b. For binary matrices, the error is quite small and often converges quicker than the Jacobi method. For normal matrices, error is a factor and has an effect on the number of iterations to obtain the desired precision. However, Gauss-Seidal still converges with a lesser number of iterations than Jacobi.

When testing the test case provided in the project description, take Y0 for example:

$$Y0 = (1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0)$$

with length n = 17 to produce:

$$X = (1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0)$$

Gauss-Seidel converges only after 1 iterations while Jacobi converges after 9 iterations to achieve the error tolerance 10E^-9.

If we shorten the output stream to

$$Y0 = (1\ 0\ 0\ 0\ 0\ 0\ 1\ 0)$$

with length n = 8 to produce:

$$X = (1\ 0\ 1\ 1\ 1\ 0\ 1\ 1)$$

Gauss-Seidel converges only after 1 iteration while Jacobi converges after 4 iterations to achieve the error tolerance 10E^-9.

The length of the initial stream n is definitely important as it can reduce error and lower the number of required iterations to achieve error tolerance 10E^-9. If it is longer, it may take more iterations for either of the methods to converge to achieve the error tolerance.

For normal matrices, consider the following test cases:

$$A = \begin{bmatrix} 5 & 2 & -1 \\ 3 & 7 & 3 \\ 1 & -4 & 6 \end{bmatrix} \qquad B = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

Below are the results after running Gauss-Seidel and Jacobi with tolerance of 10E^-6:

GAUSS-SEIDEL	**JACOBI**
Tolerance is: 1.0E-4	Tolerance is: 1.0E-5
ERROR:0.5203566	ERROR:0.45627397
ERROR:0.10501395	ERROR:0.30557927
ERROR:0.011356253	ERROR:0.19093087
ERROR:0.0029144932	ERROR:0.06837635
ERROR:5.6696375E-4	ERROR:0.046537325
ERROR:6.371627E-5	ERROR:0.0169065
Method converges after 5 iteration(s).	ERROR:0.011488597
0.4980723755553548	ERROR:0.0056036557
-0.30501152949813365	ERROR:0.0021189265
-0.11968641559131488	ERROR:0.0016702825
	ERROR:5.6606304E-4
	ERROR:3.6560697E-4
	ERROR:1.9997744E-4
	ERROR:6.926809E-5
	ERROR:5.5759214E-5
	ERROR:2.0187706E-5
	ERROR:1.1874491E-5
	ERROR:6.938538E-6
	Method converges after 17 iteration(s).
	0.4980712334146915
	-0.3050215130658671
	-0.11969207002806721

As seen above, Jacobi takes about 3-4 times the number of iterations to obtain the precision of tolerance when performed on a normal matrix.

Part 3

Discussion:

Given Leslie Matirx A:

$$A = \begin{pmatrix} 0 & 1.2 & 1.1 & .9 & .1 & 0 & 0 & 0 & 0 \\ .7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .85 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .9 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .9 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .88 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & .8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .77 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & .40 & 0 \end{pmatrix}$$

Interpret the data in the matrix, and discuss the social factors that influence those numbers.

As the age group increases, the s_x (the fraction of individual that survives from age class x to age class x+1 (basically from one age class to the next age class)), decreases. This is because as age grows, people are more likely to pass away, making the survival rate much lower at age groups like 70-80, and 0 at the group 80-90. Likewise, f_x, (the fecundity, or the per capita average number of female offspring reaching n_1 ($2.1x10^5$ people) born from mother of age class x) also decreases. Clearly, in the age group 0-9, the fecundity (f_x), is 0 since it is pretty much impossible for girls that age to even give birth. At age group 10-19 it is the highest because around the age of 18 and 19, women can easily give birth. Now as age increases, the fecundity decreases since older women cannot give birth that much. In the age group 20-29 the fecundity is very close to the fecundity of the age group before (10-19), since most women give birth in that age group before they reach their late twenties, but in the next two age groups (30-39 and 40-49), the fecundity drops to 0.9 and then sharply to 0.1 since almost no women give birth in their forties. As age increases, the harder it is for womens' bodies to give birth, which is why the fecundity keeps decreasing and finally hits zero in the age group 50-60. The fraction of individuals that survive in a certain age group (s_x) decreases as the age group increases, and fecundity (per capita average number of female offsprings reaching n_1 ($2.1 x 10^5$ people) born from mother of age class x) also decreases; therefore, s_x and fecundity (f_x) are directly proportional to each other.

What will the population distribution be in 2010? 2020? 2030? 2040? 2050?

Calculate also the total population in those years, and by what fraction the total population changed each year.

<u>2000</u>

Population Distribution:

$$\begin{bmatrix} 2.1 \\ 1.9 \\ 1.8 \\ 2.1 \\ 2.0 \\ 1.7 \\ 1.2 \\ 0.9 \\ 0.5 \end{bmatrix} \quad x\ 10^5$$

Total Population: $14.2 \text{ x } 10^5$ people

$\underline{2010}$

Population Distribution:

$$\begin{bmatrix} 6.35 \\ 1.47 \\ 1.615 \\ 1.62 \\ 1.89 \\ 1.76 \\ 1.36 \\ 0.924 \\ 0.36 \end{bmatrix} \text{ x } 10^5$$

Total Population: 17.349 x 10^5 people

Fraction the total population changed by last year: $\frac{2010-2000}{2000}$

$$= \frac{17.349 \text{ x } 10^\wedge 5 \text{ people} - 14.2 \text{ x } 10^\wedge 5 \text{ people}}{14.2 \text{ x } 10^\wedge 5 \text{ people}} = 22.1761\%$$

$\underline{2020}$

Population Distribution:

$$\begin{bmatrix} 5.188 \\ 4.445 \\ 1.250 \\ 1.454 \\ 1.458 \\ 1.663 \\ 1.408 \\ 1.047 \\ 0.370 \end{bmatrix} \text{ x } 10^5$$

Total Population: 18.283 x 10^5 people

Fraction the total population changed by last year: $\frac{2020-2010}{2010}$

$$= \frac{18.283 \text{ x } 10^\wedge 5 \text{ people} - 17.349 \text{ x } 10^\wedge 5 \text{ people}}{17.349 \text{ x } 10^\wedge 5 \text{ people}} = 5.3836\%$$

2030

Population Distribution:

$$\begin{bmatrix} 8.16240 \\ 3.63125 \\ 3.77825 \\ 1.12455 \\ 1.30815 \\ 1.28304 \\ 1.33056 \\ 1.08416 \\ 0.41888 \end{bmatrix} \text{ x } 10^5$$

Total Population: 22.1212 x 10^5 people

Fraction the total population changed by last year: $\frac{2030-2020}{2020}$

$$= \frac{22.1212 \text{ x } 10\text{^}5 \text{ people} - 18.283 \text{ x } 10\text{^}5 \text{ people}}{18.283 \text{ x } 10\text{^}5 \text{ people}} = 20.9933\%$$

2040

Population Distribution:

$$\begin{bmatrix} 9.6565 \\ 5.7137 \\ 3.0866 \\ 3.4004 \\ 1.0121 \\ 1.1512 \\ 1.0264 \\ 1.0245 \\ 0.4337 \end{bmatrix} \text{ x } 10^5$$

Total Population: 26.5051 x 10^5

Fraction the total population changed by last year: $\frac{2040-2030}{2030}$

$$= \frac{26.5051 \text{ x } 10\text{^}5 \text{ people} - 22.1212 \text{ x } 10\text{^}5 \text{ people}}{22.1212 \text{ x } 10\text{^}5 \text{ people}} = 19.8176\%$$

2050

Population Distribution:

$$\begin{bmatrix} 13.4134 \\ 6.7596 \\ 4.8567 \\ 2.7779 \\ 3.0604 \\ 0.8906 \\ 0.9209 \\ 0.7904 \\ 0.4098 \end{bmatrix} \text{ x } 10^5$$

Total Population: 33.8797 x 10^5

Fraction the total population changed by last year: $\frac{2050-2040}{2040}$

$$= \frac{33.8797 \text{ x } 10^{\wedge}5 \text{ people} - 26.5051 \text{ x } 10^{\wedge}5 \text{ people}}{26.5051 \text{ x } 10^{\wedge}5 \text{ people}} = 27.8233\%$$

Use the power method to calculate the largest eigenvalue of the Leslie matrix A. The iteration of the power method should stop when you get 8 digits of accuracy. What does this tell you? Will the population go to zero, become stable, or be unstable in the long run? Discuss carefully and provide the mathematical arguments for your conclusion. You might want to investigate the convergence of $||A^k||$.

max eigenvalue: 1.2886562376686987

Since the maximum eigenvalue is greater than one, the population will become stable and slowly increase.

2030 new result: A'u_2020, where A': A with decreased birth rate of second age group by half and u_2020: Previous vector of results for 2020 from question #2

$$\begin{bmatrix} 5.496 \\ 3.632 \\ 3.778 \\ 1.125 \\ 1.309 \\ 1.283 \\ 1.330 \\ 1.084 \\ 0.419 \end{bmatrix} \text{ x } 10^5$$

Total Population: 19.456 x 10^5

2040 new result: **A'u_2030**, where A': A with decreased birth rate of second age group by half and u_2030: New vector of results for 2030 from question #4 (this question), which was calculated just above.

$$\begin{bmatrix} 7.478 \\ 3.847 \\ 3.087 \\ 3.400 \\ 1.013 \\ 1.152 \\ 1.026 \\ 1.024 \\ 0.434 \end{bmatrix} \text{ x } 10^5$$

Total Population: 22.461 x 10^5

2050 new result: A'u_2040, where A': A with decreased birth rate of second age group by half and u_2040: New vector of results for 2040 from question #4 (this question), which was calculated just above.

$$\begin{bmatrix} 8.865 \\ 5.235 \\ 3.270 \\ 2.778 \\ 3.060 \\ 0.891 \\ 0.922 \\ 0.790 \\ 0.410 \end{bmatrix} \text{ x } 10^5$$

Total Population: 26.221 x 10^5

References

For pages 19 through 29 of Linear Algebra: Numerical Methods,

http://www-old.math.gatech.edu/academic/courses/core/math2601/Web-notes/2num.pdf

For pages 30 through 44 of Linear Algebra: Numerical Methods,

http://www-old.math.gatech.edu/academic/courses/core/math2601/Web-notes/3num.pdf