Numerical Approximator Documentation

Brandon Morgan

Last Updated: 6/4/2020

Purpose

The purpose of this document is to explain the mathematical and computer science theory behind the *Numerical Approximator* program. This program, serves as a culmination of all the knowledge, tools, and skills developed and learned throughout my computer science and mathematical career in college.

Courses Showcased

Mathematical

- Calculus 1-3
- Differential Equations
- Linear Algebra
- Numerical Analysis
- Applied Statistical Methods
- Applied Multivariate Statistical Analysis

Computer Science

- Data Structures
- Computer Organization
- Theory of Computation
- Principles of Programming Languages
- Operating Systems

Summary/Application of Program

The *Numerical Approximator* is similar to other scientific computing like software, such as MATLAB or the statistical environment R. The backend of the program is programed in C, using only native libraries, while employing Python as a friendly interactive frontend.

The *Numerical Approximator* can be used as a simple calculator, handling precedence and associativity with numerously available operations; it can be used to numerically calculate advanced single valued integrals and root solving problems; additionally, it can also be used to perform advanced matrix algebra, employed in the statistical frame of principal component analysis, factor analysis, canonical correlation analysis, and cluster analysis.

Course Contribution

In this section I will detail exactly what each of the courses listed above contribute in both theory and application to the program.

Mathematical

1. *Calculus 1-3:* These three classes serve as the basic foundation for all of the future mathematical classes.
2. *Differential Equations:* NILL
3. *Linear Algebra:* This class served as the theoretical foundation for the following mathematical classes: Numerical Analysis and Applied Multivariate Statistical Analysis.
4. *Numerical Analysis:* The basic idea behind this class is to take mathematical algorithms and apply them within a scientific computing environment with limited machine precision. We

analyzed famous numerical algorithms for approximating functions, derivates, integrals, ordinary differential equations, and matrix algebra operations; while also looking into the algorithms stability, convergence rate, and applicability.

The current numerical methods being used in the program are the following:

1) Bisection Method:
   This simple root-finding method can be applied to any continuous function. Similar to a binary search method, the Bisection Method halves the problem into smaller pieces before advancing. The advantages of this method is that it will always converge to the root; however, the disadvantages are that it is quite slow in its convergence rate (almost linear) when compared to other methods; and it requires a given interval, $[a, b]$, such that $f(a) * f(b) < 0$.

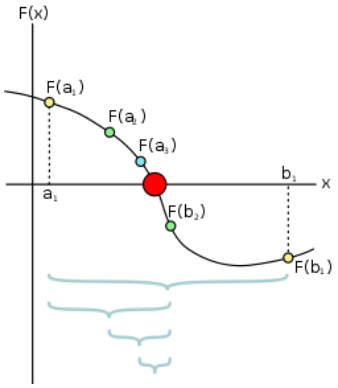


Figure 1: Bisection Method
Source: Wikipedia Bisection Page

2) Newton Raphson Method:
   This advanced root-finding method is exonerated for its extremely high convergence rate. It uses the derivative of the function to create tangent lines at the slope of a point and where this tangent line intersects the function curve is the next 'guess' point. The basic formula is the following: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$. The advantages of this method is that it is extremely fast and accurate when converging to the root; however, the disadvantages are that it requires an arbitrarily 'close' enough initial guess value that is different for all cases. In addition, Newton Raphson's method requires the knowledge of the first derivate; however, in application where this is not known, the formal definition of a derivative can be used: $f'(x) = \frac{f(x+h)-f(x)}{h}$, for h small, such as 10^{-9}. It is common practice to use the Bisection Method until a certain tolerance level as an initial guess for Newton Raphson as it will converge quicker to the root.

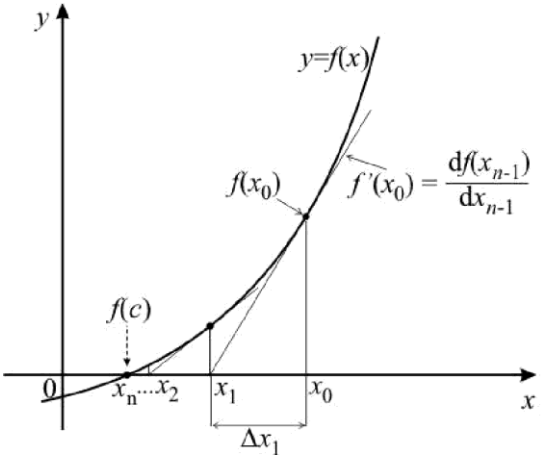


Figure 2: Newton Raphson Method
Source: ReasearchGate.net

3) Simpsons Rule:

This is a numerical integration algorithm for approximating definite integrals. It does this by subdividing the integral interval, $[a, b]$, into a preset n (must be even) number of intervals and calculates the quadrature (area of created rectangles) over the intervals. In its basic form, Simpsons Rule is calculated by:

$\int_a^b f(x)dx \approx \frac{h}{3}(f(a) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \cdots + 4 * f(x_{n-1}) + f(b))$, where $h = \frac{b-a}{n}$ and $x_i = a + i * h$.

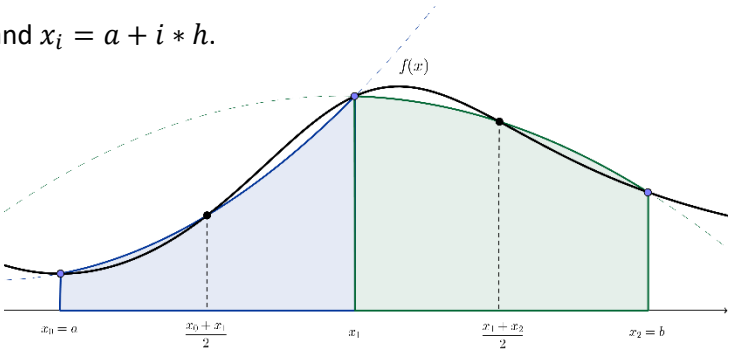


Figure 3: Simpsons Rule
(Source) Wikimedia Commons

4) Adaptive Simpsons Rule:

This method is an advanced way to estimate integrals more precisely than the standard Simpson's rule. To gain more precision in the standard Simpsons rule, one would have to increase the n number of subdivisions. However, the exact number for n is not known until the previous approximate value is approximately equal to the new value. For example, take the following integral: $\int_0^{15} x^2 e^{-\frac{x}{2}}$, which equal to 15.67589. Performing Simpsons Rule with different values of n will result in the following different approximate values:

$n = 2$	13.53984
$n = 4$	16.52778
$n = 6$	16.03077
$n = 8$	15.82175
$n = 10$	15.74342
$n = 12$	15.71072
⋮	⋮
$n = 100$	15.67590

As one can see, it took until $n = 100$ until the integral approximate converged closely enough to be accepted. In practice, this form of doubling the n value until the previous estimate minus the current estimate is less than a specified tolerance is extremely inefficient as increasing the n value vastly increases the computational cost as one is dividing the integral into n intervals and computing n number of functions. Adaptive Simpsons Rule resolves this problem by recursively sub diving the interval when the two pieces of the interval are not sufficiently close to the entire interval with a constant set n number: $|(S(a, m) + S(m, b)) - S(a, b)| < 15\epsilon$, where m is the midpoint in the interval $[a, b]$, where $S(x, y)$ is the value estimated from Simpsons rule over the interval $[x, y]$, and ϵ is a specified error tolerance. If the value given above is not met then the interval $[a, b]$ is subdivided into $[a, m]$.

5) Improper Integrals with Bounds to Infinity

Any convergent integral with infinity bounds has the following form: $\int_a^{\infty} f(x)dx$, which can be transformed to an equivalent form $\int_{1/a}^{0} -\frac{1}{x^2} f\left(\frac{1}{x}\right) dx$; therefore, one can rearrange the original function and bounds to estimate the integral using an Adaptive Simpson's rule.

6) Naive Gaussian Elimination

A main facet to linear algebra is the concept of finding the solution to system of linear equations, $Ax = b$. Take for example the following system of equations:

$$\begin{array}{lccccccccc} A_1: & x_1 & + & x_2 & & & + & 3x_4 & = 4 \\ A_2: & 2x_1 & + & x_2 & - & x_3 & + & x_4 & = 1 \\ A_3: & 3x_1 & - & x_2 & - & x_3 & + & 2x_4 & = -3 \\ A_4: & -x_1 & + & 2x_2 & + & 3x_3 & - & x_4 & = 4 \end{array}$$

This can be transformed into the following augmented matrix:

$$A_{4\times 5} = \left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 2 & 1 & -1 & 1 & 1 \\ 3 & -1 & -1 & 2 & -3 \\ -1 & 2 & 3 & -1 & 4 \end{array}\right)$$

The idea behind Naïve Gaussian Elimination is to utilize basic matrix algebra to row reduce an $n \times (n+1)$ matrix into an equivalent upper triangular matrix, which allows the solution of the system to be easily solved through Backwards Substitution. An upper triangular matrix has the following form:

$$M_{n\times(n+1)} = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & a_{1,n+1} \\ 0 & a_{22} & \dots & a_{2n} & a_{2,n+1} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & \dots & a_{nn} & a_{n,n+1} \end{array}\right)$$

In our previous example, the Naïve Gaussian Elimination reduced matrix is the following:

$$A_{4\times 5} = \left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 0 & 0 & 3 & 13 & 13 \\ 0 & 0 & 0 & -13 & -13 \end{array}\right)$$

7) Backwards/Forward Substitution

After one row reduces a matrix into an equivalent form, the solution can be calculated by use of Backwards Substitution if the row reduced matrix is an upper triangular matrix. In the example given in Naïve Gaussian Elimination, the solution of the matrix can be calculated by starting at the bottom right hand corner and dividing the right-hand side by the left-hand side, storing the value of the variable, and proceeding upwards while substituting the known values into the variable places. Take the previous example:

$$E_{4\times5} = \left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 0 & 0 & 3 & 13 & 13 \\ 0 & 0 & 0 & -13 & -13 \end{array}\right) \quad \begin{array}{l} 1x_1 + 1(2) + 0 + 3(1) = 4 \ \therefore \ x_1 = -1/1 = -1 \\ -1x_2 - 1(0) - 5(1) = -7 \ \therefore \ x_2 = -2/-1 = 2 \\ 3x_3 + 13(1) = 13 \ \therefore \ x_3 = 0/3 = 0 \\ -13x_4 = -13 \ \therefore \ x_4 = -13/-13 = 1 \end{array}$$

Therefore, the solution to the system is: $X_{4\times1} = \begin{pmatrix} -1 \\ 2 \\ 0 \\ 1 \end{pmatrix}$

Forward Substitution is the same technique but applied to a Lower Triangular matrix, which takes the following form:

$$M_{n\times(n+1)} = \left(\begin{array}{cccc|c} a_{11} & 0 & \dots & 0 & a_{1,n+1} \\ a_{21} & a_{22} & \dots & 0 & a_{2,n+1} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} & a_{n,n+1} \end{array}\right)$$

8) LU Decomposition

Lower-Upper (LU) decomposition decomposes a square $(n \times n)$ matrix into a product of a Lower (L) and Upper (U) triangular matrices: $A = LU$

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_{n1} & \dots & \dots & a_{nn} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & u_{nn} \end{pmatrix}$$

For example, the following 2×2 matrix can be decomposed into a product of its lower and upper triangular matrices:

$$\begin{pmatrix} 4 & 3 \\ 6 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1.5 & 1 \end{pmatrix} \begin{pmatrix} 4 & 3 \\ 0 & -1.5 \end{pmatrix}$$

By decomposing matrices into this form, it serves as an alternative form of solving a system of linear equations, where $A = LU \ \therefore \ LUx = b$, now let $y = Ux \ \therefore \ Ly = b$. To solve the system, one forward substitution on $Ly = b$ to obtain y, then one performs backwards substitution on $Ux = y$ to obtain the solution x. Other applications include the inverse of a matrix and calculating the determinant.

9) Inverse

As stated before, LU decomposition can be used to calculate the inverse of a matrix by the following formula: $LUA^{-1} = I$, where I is the identity matrix: $\begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix}$

Then one can perform a series of backward and forward substitutions to find the solution, A^{-1}, to the system.

10) QR Gram-Schmidt Decomposition

QR decomposition is very similar to LU decomposition in the way that a matrix A can be decomposed into two pieces, QR, where Q is an orthogonal matrix, i.e. $Q^T Q = I$, where Q^T represents the transpose of the Q matrix, and R is an upper triangular matrix. There are different ways to decompose the matrix $A = QR$, one popular method is the Gram-Schmidt process. The procedure vectorizes the A matrix into the following form: $A = [a_1 \mid \ a_2 \mid \ \dots \mid \ a_n]$

Then,

$$u_1 = a_1, \quad e_1 = \frac{u_1}{\|u_1\|}$$
$$u_2 = a_2 - (a_2 \cdot e_1)e_1, \quad e_2 = \frac{u_2}{\|u_2\|}$$
$$u_{k+1} = a_{k+1} - (a_{k+1} \cdot e_1)e_1 - \cdots - (a_{k+1} \cdot e_k)e_k, \quad e_{k+1} = \frac{u_{k+1}}{\|u_{k+1}\|}$$

Where

$$A = [a_1 \mid \ a_2 \mid \ \dots \mid \ a_n] \ = \ [e_1 \mid \ e_2 \mid \ \dots \mid \ e_n] \begin{bmatrix} a_1 \cdot e_1 & a_2 \cdot e_1 & \dots & a_n \cdot e_1 \\ 0 & a_2 \cdot e_2 & \dots & a_n \cdot e_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_n \cdot e_n \end{bmatrix}$$

Due to the method hinging on the assumption that the Q matrix is orthogonal, the algorithm is regarded as unstable, for in practice it is not guaranteed that there exists an orthogonal matrix Q for certain matrices. The QR method can be used to solve systems of equations in the form $Ax = b$, where $QRx = b$. In addition, the method is commonly used on what is known as a House-Holders matrix to calculate the eigen values and vectors of a matrix.

11) Eigen Values/Vectors

A square $n \times n$ matrix can be linearly transformed, where there exists a vector (eigenvector) that does not direction, but only in scale in a certain factor (eigenvalue). The applications of eigenvalues and eigenvectors are numerous and commonplace in advanced mathematics, physics, statistics, engineering, etc. The eigenvalues are calculated by the following form: $|A - \lambda I| = 0$, where λ is a single column vector containing the eigenvalues. Then, the associate eigenvectors are calculated by: $(\lambda I - A)e = 0$, where λ is a single eigen values and e is the associated eigen vector. Symmetric matrices, square matrices where $A^T = A$, are more computationally friendlier to calculate their eigenvalues and eigenvectors than non-symmetric matrices. Symmetric matrices can be reduced into a Householder transformation, that reflects the matrix about a plane containing the origin, $P = I - 2ww^T$, where $ww^T = 1$. A Householder transformation will have the following symmetric tri-diagonal form:

$$\begin{pmatrix} a_{11}^{k+1} & a_{12}^{k+1} & 0 & \dots & \dots & 0 \\ a_{21}^{k+1} & \ddots & \ddots & \ddots & \dots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & a_{k+1,n}^{k+1} \\ 0 & \dots & \dots & 0 & a_{n,k+1}^{k+1} & a_{nn}^{k+1} \end{pmatrix}$$

A matrix in a Householder transformation form can easily be used by a modified form of the QR Gram-Schmidt process to find the eigenvalues of the matrix. However, to find the associated eigenvectors is tricky because one is trying to find the solution to a homogenous solution $Ae = 0$; where a logical solution is the zero vector, to which normal solution finding methods discussed earlier will compute. However, by modifying the previously discussed algorithms slightly one can work around such cases to find the eigenvector.

5. *Applied Statistical Methods*: This class acted as the theoretical foundation and basic building blocks of advanced applied engineering statistics and classes such as: Applied Multivariate Statistical Analysis.

6. *Applied Multivariate Statistical Analysis:* This class embodied all of the advanced statistical and matrix theory necessary to apply to practical analysis to multivariate normal datasets. The class began with heavy matrix and statistical theory necessary to understand and derive all the applications of multivariate analysis. Such analysis covered include: Principal Component Analysis, Factor Analysis, Canonical Correlation Analysis, Discrimination and Classification, and Cluster Analysis.

1) Gamma Distribution
   The Gamma value can be computed by: $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1}e^{-x}$, which for integer values of α can be calculated by $\Gamma(\alpha) = (\alpha - 1)!$; This function is commonly used in statistical distributions, namely Gamma type distributions. The Probability Density Function for Gamma Type distributions is the following:

$$f(y) = \frac{y^{\alpha-1}e^{-y/\beta}}{\beta^{\alpha}\Gamma(\alpha)}$$

   Where α is known as the shape parameter and β is known as the scale parameter; and the mean $\mu = \alpha\beta$ and variance $\sigma^2 = \alpha\beta^2$.

2) Chi -Square
   The Chi-Square distribution is the sum of squared independent standard normal random variables. The Probability Density Function (PDF) is a type of gamma distribution where $\beta = 2$, and $\alpha = v/2$, where v is the number of degrees of freedom: the number of independent variables.

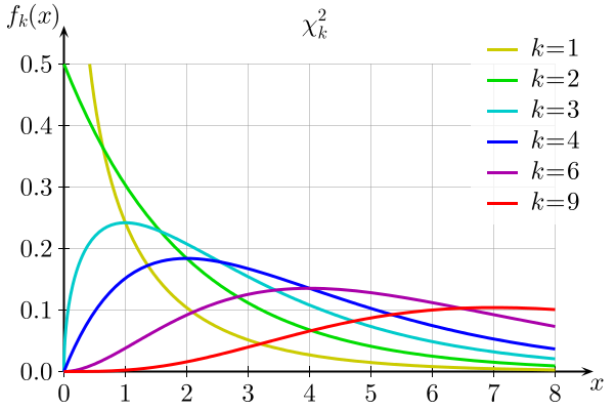
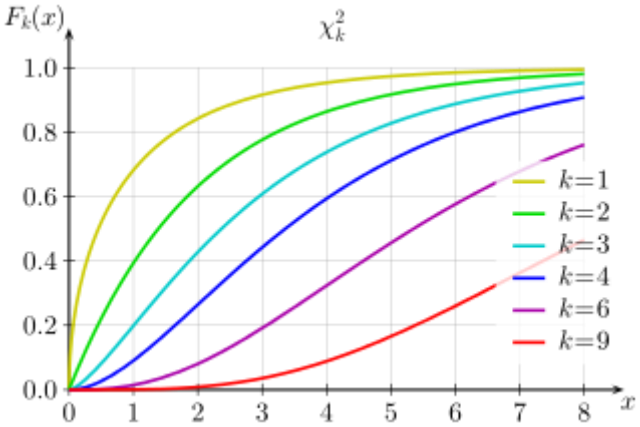


Figure 5: Chi Square PDF (Note): K represents v
(Source) Wikipedia Gamma Distribution Page

The function is used to find the probability value given a quantile. The Cumulative Density Function (CDF) is the inverse of the PDF, it finds the quantile given a probability. This is estimated in the program by creating a root-solving scenario where $0 = -f(x) + \frac{x^{\alpha-1}e^{-\frac{x}{\beta}}}{\beta^{\alpha}\Gamma(\alpha)}$, this can be solved through a modified version of the bisection method.



The Chi Square distribution is a gamma type distribution with $\beta = 2$ and $\alpha = v/2$ where v is known as the degrees of freedom (number of variables in a multivariate context). This distribution is the sum of squares of p number of independent normal random variables. In a multivariate normal context, its distribution can be modeled by a Chi Square distribution; therefore, it is the vital framework behind the basic building blocks of multivariate statistical analysis. Its Probability Density Function is the following:

$$f(y) = \frac{y^{\frac{v}{2}-1}\, e^{-y/2}}{2^{\frac{v}{2}}\, \Gamma(\frac{v}{2})}$$

In application, if a situation is modeled by a Chi Square distribution, and one wants to find the probability whether a value will fall between a range, then one can transform the probability density function into an integral:

$$p(x_1, x_2) = \int_{x_1}^{x_2} \frac{y^{\frac{v}{2}-1}\, e^{-y/2}}{2^{\frac{v}{2}}\, \Gamma(\frac{v}{2})}$$

For example, what is the probability that a given x value will fall beneath 9.48 with $v = 4$? This is calculated by:

$$p(0, 9.48) = \int_{0}^{9.48} \frac{y^{\frac{4}{2}-1}\, e^{-\frac{y}{2}}}{2^{\frac{4}{2}}\, \Gamma\left(\frac{4}{2}\right)} \approx 0.95$$

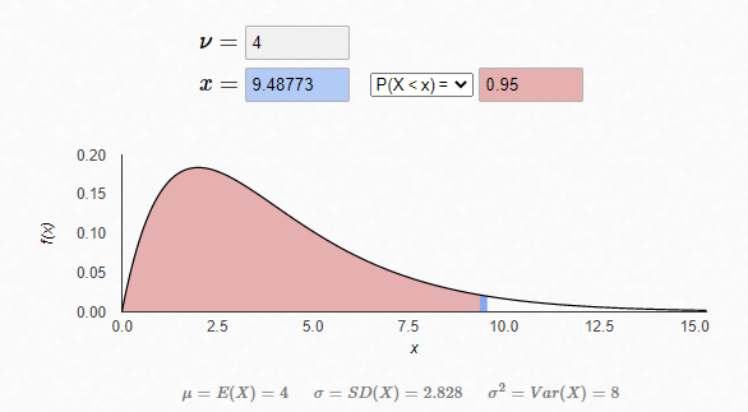


Figure 5: Chi Square PDF Example
Source: Online Calculator by Matt Bognar, Department of Statistics University of Iowa

From this value, approximately 95% of all the data will be below 9.48 in a Chi Square distribution with four degrees of freedom.

The inverse, or opposite, of the Probability Density Function is the Cumulative Density Function. Instead of giving an range of x values to calculate the probability, this function takes in a probability and calculates at what x value does it correspond to. It is modeled by this equation:

$$0 = p(0, x_2) - \int_{0}^{x_2} \frac{y^{\frac{v}{2}-1}\, e^{-y/2}}{2^{\frac{v}{2}}\, \Gamma(\frac{v}{2})}$$

As one can see, this inverse function can be calculated by being turned into a root solving problem. For example, if one wants to find at what x value does 95% of a Chi Square distribution fall beneath with $v = 4$, then it can be calculated by the following root solving problem, to which the answer would be 9.48:

$$0 = 0.95 - \int_{0}^{x} \frac{y^{\frac{4}{2}-1}\, e^{-y/2}}{2^{\frac{4}{2}}\, \Gamma(\frac{4}{2})}$$

It theory, one could calculate the x value by first using the Bisection Method until a suitable initial value could be used into Newton Raphson's method to solidify the result. However, Newton Raphson requires the knowledge of the first derivative; thankfully to the Fundamental Theorem of Calculus, $\frac{d}{dx}\int_a^x f(t)dt = f(x)$. Therefore, the first derivative can be calculated as such: $\frac{x^{\frac{v}{2}-1}\, e^{-x/2}}{2^{\frac{v}{2}}\,\Gamma(\frac{v}{2})}$. Unfortunately, in practice Newton Raphson can not be used on this function. The Newton Raphson method requires an 'arbitrarily' close initial guess that is not guaranteed to exist for all values; if the initial guess is not close enough or no initial guess is sufficient then the method will diverge to some value, usually infinity. Therefore, in this program only the Bisection Method was used to calculate the x value.

3) Covariance and Correlation matrices

In a simple linear model, i.e. (only an input and an output), there exists a variance and correlation. The variance is known as the average of the squared differences from the mean of the data, which describes the how far the average spread are the observations from the mean. Correlation coefficients, where Pearson's coefficient is most commonly used, explains how strong a relationship is between two variables. In a multivariate normal context, each variable doesn't just have a variance but also a covariance with other matrices; same with correlation coefficients. These statistical descriptors are calculated by:

Covariance: $S = s_{ik} = \frac{1}{n-1}\Sigma_{j=1}^{n}(x_{ji} - \bar{x}_i)(x_{jk} - \bar{x}_k)$
Correlation: $R = r_{ik} = \frac{s_{ik}}{\sqrt{s_{ii}}\sqrt{s_{kk}}}$
For $i = k = 1,2 \ldots p$, where p is the number of variables (i.e. columns from the data), also notice that these matrices are symmetric.

For example, take the following data matrix with three variables and four observations:

$$X = \begin{array}{c} \\ \text{Alabama} \\ \text{Alaska} \\ \text{Arizona} \\ \text{Arkansas} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Murder} & \text{Assault} & \text{Urban Population} \end{array} \\ \begin{pmatrix} 13.2 & 234 & 58 \\ 10 & 263 & 48 \\ 8.1 & 294 & 80 \\ 8.8 & 190 & 50 \end{pmatrix} \end{array}$$

The covariance of this matrix would be the following:

$$S = \begin{array}{c} \\ \text{Murder} \\ \text{Assault} \\ \text{Urban Population} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Murder} & \text{Assault} & \text{Urban Population} \end{array} \\ \begin{pmatrix} 5.095 & -18.658 & -10.767 \\ -18.658 & 1942.917 & 445 \\ -10.767 & 445 & 214.667 \end{pmatrix} \end{array}$$

The trace of the matrix (diagonal values) are the variances of the variables, where 5.095 is the variance of the Murder variable, 1942.917 the variance of the Assault variable, and so

on. The non-diagonal values are the covariance values between different variables, this is the measure of the 'joint' variable of two random variables. If the covariance value tends to zero then the variables are regarded as independent, i.e. no increase or decrease in one variable changes the values in the other variables. If the covariance value is negative then the greater values of one variable correspond the to the lesser values of the other variable; if the covariance value is positive then it is the other way around. For example, the covariance between murder and assault is a negative
value, indicated that large values of assault correspond to lower values of murder.

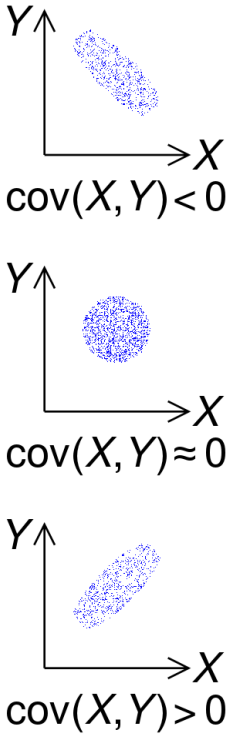


Figure 6: Covariance graphs

Source: Covariance Wikipedia Page

The correlation matrix of this dataset would be the following:

$$R = \begin{array}{c} \\ \text{Murder} \\ \text{Assault} \\ \text{Urban Population} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Murder} & \text{Assault} & \text{Urban Population} \end{array} \\ \begin{pmatrix} 1 & -0.187 & -0.326 \\ -0.187 & 1 & 0.689 \\ -0.326 & 0.689 & 1 \end{pmatrix} \end{array}$$

The correlation values in a correlation matrix will always lie between $-1 \leq r \leq 1$, where values closer to 1 represent a strong positive correlation between the variables, values closer to -1 for a strong negative correlation, and values closer to 0 for no correlation at all. The trace of a correlation matrix is always composed of only 1's because any increase in that variable directly corresponds to an increase in that same variable. From the correlation matrix example above, it seems that there is no correlation between murder and assault, and murder and urban population; however, the correlation coefficient between assault and

urban population is 0.689, which means that there is a slight positive correlation between these two variables. Theoretically then, by increasing the urban population in these four states, murder and assault will likely stay the same but assault cases will increase.

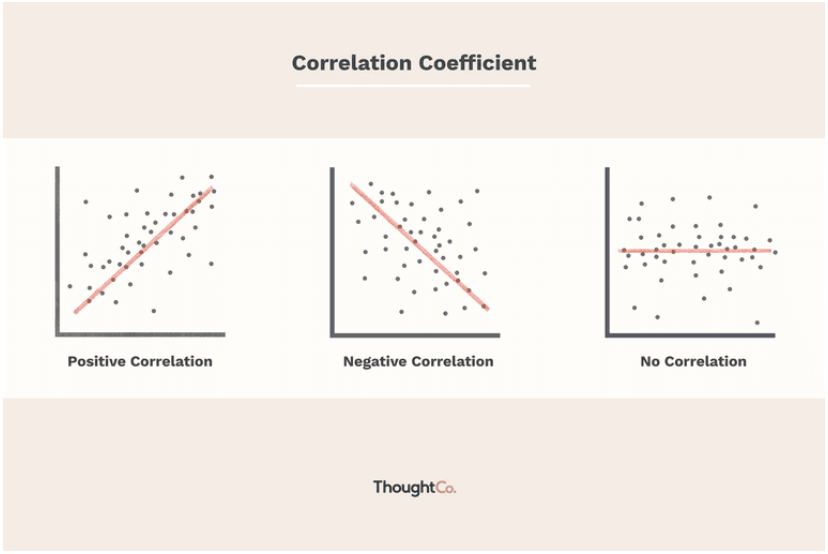


Figure 7: Correlation Graphs
Source: ThoughtCo.com

4) Normal Distribution

The Normal Distribution is a continuous probability distribution for a single random variable. The PDF of the Normal distribution is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

Where σ represents the Standard Deviation, and μ represents the Mean.

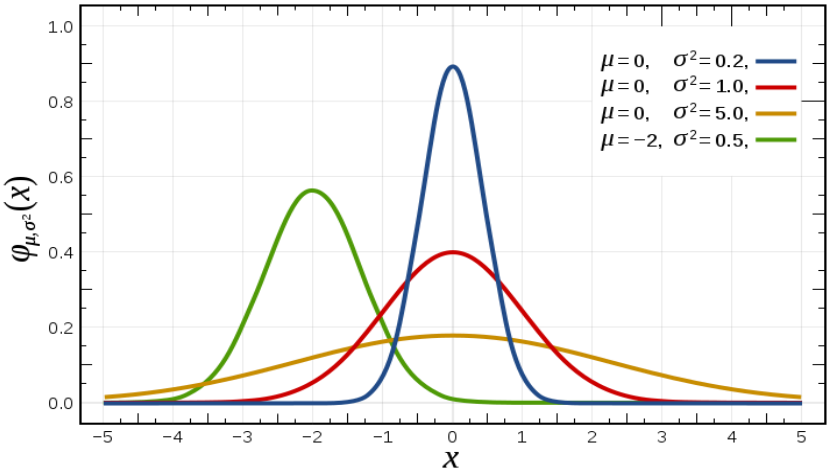


Figure 8: Normal Distribution PDF

Source: Wikipedia Normal Distribution Page

As with the Chi-Square CDF, the Normal CDF is constructed as a root-solving problem and solved for using the bisection method.

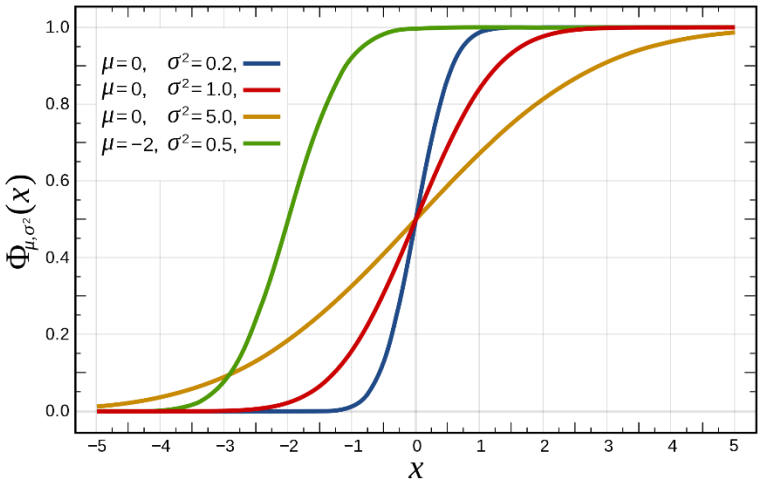


Figure 9: Normal Distribution PDF

Source: Wikipedia Normal Distribution Page

5) Outliers

In data sets, outliers are known as observations that differ extremely in distance from the average population. In a multivariate normal context, the outliers can be detected in different ways, one being by calculating the standard z-scores: $z_{jk} = \frac{(x_{jk} - \bar{x}_k)}{\sqrt{s_{kk}}}$, for $j = 1,2 \dots n$ and $k = 1,2 \dots p$; where x is the data value, $\bar{x}$ is the mean, and s is the covariance matrix. By Chebyshev's theorem, approximately 95% of the data will lie between three standard deviations away from the mean; therefore, any value with a z score higher than three is a suspected outlier. The second way is to is to calculate the generalized squared distances: $(\boldsymbol{x}_j - \overline{\boldsymbol{x}})^T \boldsymbol{S}^{-1} (\boldsymbol{x}_j - \overline{\boldsymbol{x}})$, where j is the row index of an observation. The result, a singular value, can then be compared against the Cumulative Density Function for a particular probability value, $(\boldsymbol{x}_j - \overline{\boldsymbol{x}})^T \boldsymbol{S}^{-1} (\boldsymbol{x}_j - \overline{\boldsymbol{x}}) \leq \boldsymbol{\chi}^2_{\boldsymbol{v}}(p)$. For example, by taking our previous illustrations, if a matrix of data with three columns, i.e. three variables ($v = 3$), then any observation whose $(\boldsymbol{x}_j - \overline{\boldsymbol{x}})^T \boldsymbol{S}^{-1} \left(\boldsymbol{x}_j - \overline{\boldsymbol{x}}\right) \leq \boldsymbol{\chi}^2_{\boldsymbol{3}}(0.50) = 2.3659$, value is greater than 2.3659 is an outlier. In our previous example data, the performing the following operation $(\boldsymbol{x}_j - \overline{\boldsymbol{x}})^T \boldsymbol{S}^{-1} (\boldsymbol{x}_j - \overline{\boldsymbol{x}})$**,** we get the following matrix:

$$\begin{pmatrix} 2.25 & -0.75 & -0.75 & -0.75 \\ -0.75 & 2.25 & -0.75 & -0.75 \\ -0.75 & -0.75 & 2.25 & -0.75 \\ -0.75 & -0.75 & -0.75 & 2.25 \end{pmatrix}$$

Where the diagonal values are the computations that need to be compared against 2.3659; because none of the values are above 2.3659, then there are no evident outliers.

6) Principal Component Analysis

Principal Component Analysis (PPA) explains the variance-covariance structure between a set of variables through a few linear combinations of the variables. This analysis is not regarded as independent, but rather a means to an end. If there are p variables (columns) in a multivariate normal dataset, then there are p principal components. These principal components are constructed from the eigenvectors of either the covariance or correlation matrices of the dataset. From the example covariance matrix given in section three, the following covariance matrix:

$$S = \begin{array}{c} \\ \text{Murder} \\ \text{Assault} \\ \text{Urban Population} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Murder} & \text{Assault} & \text{Urban Population} \end{array} \\ \begin{pmatrix} 5.095 & -18.658 & -10.767 \\ -18.658 & 1942.917 & 445 \\ -10.767 & 445 & 214.667 \end{pmatrix} \end{array}$$

Has the following eigen values and vectors:

$$\lambda_1 = 2050.976 \qquad e_1^T = [\,0.0101, -0.9718, -0.2356]$$
$$\lambda_1 = 101.176 \qquad e_2^T = [\,-0.0592, -0.2357, 0.9700]$$
$$\lambda_3 = 4.527 \qquad e_3^T = [\,0.9982, -0.0042, 0.0599]$$

Therefore, the dataset has the following principal components:

$$Y_1 = e_1^T X = \ 0.0101X_1 \ - 0.9718X_2 - 0.2356X_3$$
$$Y_2 = e_2^T X = \ -0.0592X_1 \ - 0.2357X_2 + 0.9700X_3$$
$$Y_3 = e_3^T X = \ 0.9982X_1 \ - 0.0042X_2 + 0.0599X_3$$

Now one can use these new linear combinations of the original dataset to represent it; however, the main aspect of PPA is to reduce the number of variables (data reduction). This can be done by examining how much variance of the original dataset is explained by each principal component. The equation for this is the following:

$$\begin{pmatrix} \text{Proportion of} \\ \text{total population variance} \\ \text{explained by the } k\text{'th} \\ \text{component} \end{pmatrix} = \lambda_k \Big/ \Sigma_i^p \lambda_i$$

Then the population variance explained by the first component is $\frac{2050.976}{2050.976+101.176+4.527} = 0.951$, therefore 95.1% of the variance of the dataset is explained by only the first component. As one can see, eigenvalues smaller in comparison to the rest or close to zero will contribute minimal to the overall picture and will be discarded. The completeness, the population variances for the rest of the components are: 4.69% for the second component, and 0.21% for the third. In practice, the second and third component would be cut off and all further analysis would be done on the first component. The correlation between the variables and a component can be calculated by: $r_{Y_i,X_k} = \frac{e_{ik}\sqrt{\lambda_i}}{\sqrt{s_{kk}}}$, where e_{ik} is the value of the eigenvector at the index, s_{kk} is the value of the covariance

matrix at the following index. In our previous example, the correlation between all three variables and the first component is the following:

$$r_{Y_1,X_1} = 0.0101\sqrt{2050.976}\Big/\sqrt{5.095} = 0.2026$$
$$r_{Y_1,X_2} = -0.9718\sqrt{2050.976}\Big/\sqrt{1942.9177} = -0.9985$$
$$r_{Y_1,X_3} = -0.2356\sqrt{2050.976}\Big/\sqrt{214.667} = -0.7282$$

As one can see, variables two and three and highly correlated with the first component, unlike the first variable.

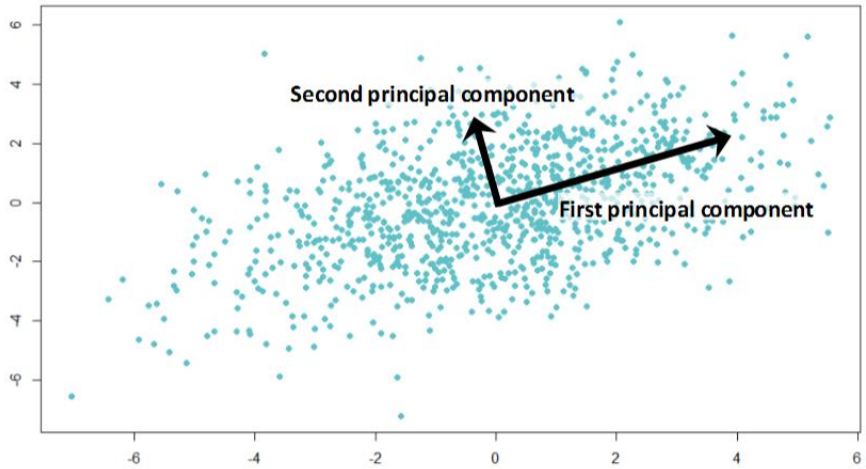


Figure 10: Example graph on an arbitrary dataset, showing how the principal components are vectors, linear combinations from the dataset
Source: Analyticsvidhya.com

Further analysis in PPA contain graphs showing the relationship between the vectors and the dataset, like Figure 8; graphs comparing the components against themselves; and graphs to determine how many components to include. As stated before, PCA is not an end game analysis, but rather a means to an end.

7) Factor Analysis

Factor Analysis seeks to explain a dataset only through a few underlying latent variables (variables that are not seen). For example, given a dataset containing test scores in language arts and physical mathematics could be explained by two underlying unmeasured variables: logic and creativity. Those who are more 'logical' will presumably score higher in the physical mathematics courses than those who are more 'creative'. Factor Analysis is not an exact science, but is commonly used in fields such as psychology. Factor Analysis (FA) is created by what is called the *Orthogonal Factor Model*: $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{LF} + \boldsymbol{\epsilon}$, where $\boldsymbol{L}$ is known as the factor loadings, and $\boldsymbol{F}$ is known as the factor scores. The covariance/correlation structure of the model is given by the following: $Cov(\boldsymbol{X}) = \boldsymbol{LL^T} + \boldsymbol{\Psi}$, where $\boldsymbol{\Psi}$ is known as the specific variances, regarded as an error matrix. One can estimate the factor loadings

through different methods such as Maximum Likelihood or Principal Component Analysis. However, in this program only PCA was use d to determine the factor loadings. To create the factor loadings, one simply lets the loadings equal to the m number of principal components times the square root of their eigen values after component reduction.

$$Cov(\boldsymbol{X}) = \boldsymbol{\Sigma} = \boldsymbol{LL^T} + \boldsymbol{\Psi}$$

$$= \begin{bmatrix} \sqrt{\lambda_1}\boldsymbol{e}_1 & \sqrt{\lambda_2}\boldsymbol{e}_2 & \cdots & \sqrt{\lambda_m}\boldsymbol{e}_m \end{bmatrix} \begin{bmatrix} \sqrt{\lambda_1}\boldsymbol{e}_1 \\ \sqrt{\lambda_2}\boldsymbol{e}_2 \\ \vdots \\ \sqrt{\lambda_m}\boldsymbol{e}_m \end{bmatrix} + \begin{pmatrix} \psi_1 & 0 & \cdots & 0 \\ 0 & \psi_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \psi_p \end{pmatrix}$$

From our previous example, the loadings would be the following:

$$L_1{}^T = \ 0.4574X_1 \ - 44.0106X_2 - 10.6697X_3$$

And the specific variances, $\boldsymbol{\Psi}$, can be calculated by $\psi_i = s_{ii} - \Sigma_j^m l_{ij}^2$. From our example, the specific variance matrix would be the following:

$$\begin{pmatrix} 4.8857 & 0 & 0 \\ 0 & 5.9841 & 0 \\ 0 & 0 & 100.8245 \end{pmatrix}$$

Then one can check how accurate the model is by calculated by the residual matrix:

$$\boldsymbol{\epsilon = \Sigma - LL^T - \Psi}$$

$$\epsilon = \begin{pmatrix} 5.095 & -18.658 & -10.767 \\ -18.658 & 1942.917 & 445 \\ -10.767 & 445 & 214.667 \end{pmatrix} - \begin{pmatrix} 0.4574 \\ -44.0106 \\ -10.6697 \end{pmatrix} \begin{pmatrix} 0.4574 & -44.0106 & -10.6697 \end{pmatrix} - \begin{pmatrix} 4.8857 & 0 & 0 \\ 0 & 5.9841 & 0 \\ 0 & 0 & 100.8245 \end{pmatrix}$$

$$= \begin{pmatrix} 0.0001 & 1.4721 & -5.8863 \\ 1.4721 & -0.0003 & -24.5799 \\ -5.8863 & -24.5799 & -0.0003 \end{pmatrix}$$

As one can see, our residual matrix is high in certain variables, therefore our model will not be extremely accurate when performing the rest of the factor analysis. To increase the reliability of the model, one can increase the number of principal components. In our example, if one were to use two principal components instead of one, the residual matrix would be the following:

$$\epsilon = \begin{pmatrix} 0.0008 & 0.0605 & -0.0771 \\ 0.0605 & -0.0003 & -1.4482 \\ -0.0771 & -1.4482 & -0.0003 \end{pmatrix}$$

Before interpreting the factor loadings, one can perform factor rotations to better amplify the values. If we let $L^* = LT$ where T orthogonal, $TT^T = T^TT = I$, then $\boldsymbol{\Sigma = LL^T + \Psi}$ becomes $\boldsymbol{\Sigma = L^*L^{*T} + \Psi}$. The matrix **T** is known as a rotation matrix, which can be one of the following for two dimensional matrices:

$$\boldsymbol{T}_{\text{clockwise}} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \text{ or } \boldsymbol{T}_{\text{counter-clockwise}} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

The basic rotation matrices for three dimensional matrices are the following:

$$\boldsymbol{T_x} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix},$$

$$\boldsymbol{T_y} = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix},$$

$$\boldsymbol{T_z} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The derivation of rotation matrices in four or more dimensions is beyond the scope of this text; in the program, only factor models with three or less factor loadings are rotated, all else is left as they were calculated. The angle chosen in these rotation matrices is chosen by maximizing the value calculated through the Varimax procedure, the angle that maximizes the following equation after rotation:

$$V = \frac{1}{p}\Sigma_j^m [\, \Sigma_i^p (\, {l_{ij}^*}/{h_i}\,)^4 - (\, \Sigma_i^p {l_{ij}^*}^2)^2 / p]$$

Where l_{ij}^* is the value of the factor loading after rotating through the angle, h is known as the communality, and p is the number of variables. For example, by running our previous loadings, using two factor loadings instead of the previous one because Varimax procedure only works with more than one dimension, through a clockwise two dimensional rotation matrix, the angle chosen that maximizes the Varimax procedure is 271°, which results in the following rotated loadings:

Original:

$${L_1}^T = 0.4574X_1 - 44.0106X_2 - 10.6697X_3$$
$${L_2}^T = -0.5954X_1 - 2.3708X_2 + 9.7569X_3$$

Rotated:

$${L^*_1}^T = -0.5873X_1 - 3.1385X_2 + 9.5692X_3$$
$${L^*_2}^T = -0.4677X_1 + 43.9625X_2 + 10.8383X_3$$

This allows for the following interpretation:

Blah insert here

The last step in factor analysis is estimating the factor scores, F, in the Orthogonal Factor Model: $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{LF} + \boldsymbol{\epsilon}$. The most common way is through the following Weighted Least Squares Method: $f_j = (L^T \Psi^{-1} L)^{-1} L^T \Psi^{-1} (x_j - \mu)$ for $j = 1,2, \ldots n$. For example, the factor scores of the following row of data original detailed in the Covariance/Correlation section, $x_1 = [13.2, 234, 58]$ as the following factor scores:

8) CCA (Not yet Implemented)

9) Discrimination (Not yet Implemented)
10) Cluster Analysis (Not yet Implemented)

Computer Science

1. Data Structures
    1) Singly Linked List
    2) Double Linked List
    3) Stack
    4) Hash Table
    5) AVL Tree
2. Computer Organization
3. Theory of Computation
4. Principles of Programming Languages
5. Operating Systems-Shared libraries, Void pointers,