GENETIC ALGORITHM BASED OPTIMAL DESIGN OF SINGLE PHASE INDUCTION MOTOR

A PROJECT REPORT

Submitted by

S.NANDA KUMAR
Reg no : 71204415006

in partial fulfillment for the award of the degree
of

MASTER OF ENGINEERING

in

POWER ELECTRONICS AND DRIVES

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE-641006

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report entitled **"Genetic algorithm based optimal design of single phase induction motor"** is the bonafide work of

Mr. S.NANDA KUMAR - Register No. 71204415006

Who carried out the project work under my supervision.

KSubramanian

Signature of the Head of the Department

Prof. K.Regupathy Subramanian

Signature of the Supervisor

Mr.R.K. Pongiannan

KSubramanian

Internal Examiner

V. Chandrasekaran

External Examiner

25/6/06

M.TECH ELECTRICALS
9 / 53, Nehru nagar
Civil aerodrome post
Kallapatti road,
Coimbatore – 641 014.

Date : 28.04.2006

CERTIFICATE

This is to certify that Mr. S.NANDA KUMAR ,final year M.E., (Power electronics and drives) student of KUMARAGURU COLLEGE OF TECHNOLOGY,COIMBATORE, has sucessfully completed his project work entitled " GENETIC ALGORITHM BASED OPTIMUM DESIGN OF SINGLE PHASE INDUCTION MOTOR ", as a part of his course, in our company during project period 06.12.2005 to 28.04.2006.

He has evinced been interest in absorbing the nature, concept and functions of our organisation and his conduct and character were good during the period.

For M.TECH ELECTRICALS,

DIRECTOR

National Conference on

ization Techniques in Engineering Sciences and Technologies

TÜV CERT
EN ISO 9001
Certificate : 0410021309
RWTÜV

OPTEST - 2006

held at

BANNARI AMMAN INSTITUTE OF TECHNOLOGY
Sathyamangalam-638 401 during
April 11-12, 2006

Certificate

Ir./Ms./~~Mrs~~ ....S. NANDAKUMAR..

ed a paper entitled .....GENETIC ... ALGORITHM .... BASED .... OPTIMAL .... DESIGN.....

..PHASE .... INDUCTION .... MOTOR..

erence on "**Optimization Techniques in Engineering Sciences and Technologies**

ng **11-12 April 2006**, organized by the **Department of Mechanical Engineering,**

e of Technology, Sathyamangalam.

KUMAR
ies

Dr K THIRUNAVUKKARASU
Convener

Dr A SHANMUGAM
Chairman

ABSTRACT

Single-phase induction motors are widely used in many domestic appliances and low power industrial applications. About a few thousands of single-phase induction motors are manufactured every month in this country. A small savings in manufacturing cost as well as running cost for single motors would have multiple beneficial effects to the industry. It is desirable to produce the motors with economical cost and at the same time not compromising on its performance. Hence, many attempts have been made to optimize design for optimum performance the single-phase induction motors .A complete energy conservation solution of a drive will be achieved by implementing an optimized design as well as the suitable control scheme. The energy conservation is quite essential in induction motors by various efficient control schemes.

The traditional search algorithms used for optimization include Runge Kutta method, Newton Raphson method and Gauss Seidel method. The non-traditional search algorithms commonly used for electrical machine design optimization are Genetic Algorithm, Simulated Annealing, Ant colony and Neural nets. The Selection of the method depends on its flexibility in computing complex equations and computation time.

In this project , a Genetic algorithm is adopted for the optimal design of single phase induction motor. The optimal design of stack length, turns ratio, cross sectional area of main winding, current density of main winding, Length of mean turn, air gap length and the flux density of stator teeth are described in this report. The simulation part has been implemented using programming language VC++. The simulation results are verified with the experimental results.

ஆய்வுச்சுருக்கம்

பொதுவாக விவசாய, அலுவலக மற்றும் வீட்டு பயன்பாடுகளுக்கு ஒரு படி உயத்துணரவைப்பு விசைப்பொறி ஆனது பயன்படுத்தப்படுகிறது. இவ்வகை விசைப்பொறியின் வடிவமைப்பானது தேசிய மற்றும் உலகத் தர அடிப்படையில் அமைந்துள்ளது. சில குறிப்பிட்ட பயன்பாடுகளுக்கு திறன் அதிகமாக தேவைப்படுவதால் விசைப்பொறியின் விலையை மிகைப்படுத்தாமல் திறனை அதிகப்படுத்த வேண்டும். இவ்வாறாக தேவைப்படும் விசைப்பொறியின் திறன் மற்றும் மின்சக்தி விகிதத்தின் துணை சார்ந்த மாறிலிகளின் சிறந்த அளவுகளை கண்டுப்பிடிக்க கணிப்பொறியின் மென்பொருள் சார்ந்த மரபு நிலை விதி அமைப்பு பயன்படுகிறது.

இந்த திட்ட அறிக்கையில் மரபு நிலை விதி அமைப்பைக் கொண்டு விசைப்பொறியின் நிலைப்பகுதியின் நீள அளவு, பிரதான மற்றும் துவக்கக் கம்பிசுருளின் குறுக்குவெட்டுப் பரப்பின் விகிதம், பிரதான மற்றும் துவக்கக் கம்பிசுருளின் எண்ணிக்கையின் விகிதம், மின்னோட்டத்தின் செரிவு விசைப்பொறியின் சுழற்சிப் பகுதியின் பரப்பு ஆகிய மாறிலிகளின் மிகச்சிறந்த மதிப்பு கண்டுப்பிடிக்கப்படுகிறது. மரபு நிலை விதிஅமைப்பின் இறுதியில் கிடைக்கும் மாறிலியின் மதிப்பைக் கொண்டு வடிவமைக்கப்படும் விசைப்பொறியானது அதிக திறன் உள்ளதாக அமைகிறது. இந்த மரபு நிலை விதி அமைப்பு கணிப்பொறியின் பயன்பாட்டு மொழியான விசி++ கொண்டு உருவாக்கப்பட்டுள்ளது.

ACKNOWLEGEMENT

I express our heartfelt gratitude to our principal **Dr.K.K.Padmanabhan,** for providing the necessary facilities for successful completion of the project.

I am highly grateful to **Prof. K.Regupathy Subramanian,** Head of the Department, Electrical and Electronics Engineering, for his enthusiasm and encouragement which has been instrumental in the success of the project.

I wish to express our deep sense of gratitude and indebtedness to my project guide **Mr.R.K.Pongiannan**, Department of Electrical and Electronics Engineering, for his invaluable guidance and constructive suggestions, the keen and constant inspiration and interest evinced by him at all stages right from the inception to the completion. I would also like to thank Mr.G.Mohan.
M/s M.TECH ELECTRICALS for his valuable guidance.

I am thankful to our parents, all the staff in Electrical and Electronics Engineering Department and other student friends for their encouragement and help during the course of the project.

TABLE OF CONTENTS

APPENDIX I

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS:

A_m - area of running winding conductor

L_m - length of mean turn of main winding

L_b - length of each bar.

A_b - area of each bar

S_r - number of rotor slots.

Rs - starting winding resistance

X_{lm} - total leakage reactance in terms of main winding.

X_{la} - total leakage reactance in terms of aux. winding

R_m or R_{sm} or R_{rm}- main winding resistance.

R_{rm}' - rotor resistance referred to running winding.

Xs - stator reactance.

D - core diameter

P - number of poles.

L_g - length of airgap.

L - length of stator core

τ - pole pitch.

μ_o - constant

K_{wm} - winding factor for the main winding

R_{rm}' - resistance of rotor referred to main winding

X_{lm} - leakage reactance of stator main winding plus rotor in terms of running winding

T_m - number of turns in main winding.

W_t - width of rotor teeth

d - depth of rotor core

R_m - resistance of main winding.

X_m - reactance of main winding

Z_m - impedance of main winding

R_a - resistance of main winding

X_a - reactance of auxiliary winding

I_a - current in auxiliary winding

I_m - current in main winding

θ_m - phase angle of main winding current

θ_a - phase angle of auxiliary winding current

F_t - The instantaneous value of mmf at the coil axis and is proportional to instantaneous stator current.

Φ_m - flux per pole

CHAPTER 1

INTRODUCTION

Single-phase induction motors are widely used in many domestic appliances and light-duty industrial applications where three-phase power is not readily available. The most common single phase induction motor types are split-phase, capacitor start, and capacitor start and run motors. The split-phase motors have the problem of poor starting torque that is solved by using a capacitor-start in series with the auxiliary winding. The start capacitor may be disconnected after starting by means of a centrifugal switch. In capacitor start and run single phase induction motors, one more capacitor is permanently connected to the auxiliary winding for improving motor performance. The function of the capacitor is to generate a leading phase current in the auxiliary winding so that the motor can produce a sufficiently high starting torque and operate as a balanced two-phase machine.

A complete energy conservation of a drive will be achieved by implementing an optimized design as well as control [1] – [8]. The energy conservation is possible single phase induction motor by various efficient control schemes such as triac phase control, PWM inverter control and multilevel inverter control. A review in the relevant technical literature shows that there are several research efforts concerning efficiency and performance improvement of single phase induction motors[1] – [6]. The major areas in the control part of single phase induction motor are the optimal placement of the run capacitor, minimum cost design and improved performance control scheme with two triacs, design considerations for improving triac controlled single phase induction motor performance, a triac controlled strategy that adjusts the supplied voltage to the motor

The traditional search algorithms used for optimization include Runge kutta method, Newton Raphson method, Gauss seidel method. The non-traditional search algorithms commonly used for electrical machine design optimization are Genetic Algorithm, Simulated Annealing, Ant colony and Neural nets .The Selection of the method depends on its flexibility in computing, complex equations and computation time[9],[10].

In this project, a Genetic algorithm is adopted for the optimal design of single phase induction motor [1]. The optimal design of stack length, turns ratio, cross sectional area of main winding, current density of main winding and the flux density of stator teeth are described in this report. The simulation part has been implemented using programming language VC++. The simulation results are verified with the experimental results.

1.1 OBJECTIVE OF THE PROJECT

- To understand Genetic algorithm and to study relevant programming languages for implementation of the engineering problem.
- To adopt Genetic algorithm for optimal design of single-phase capacitor start induction motor parameters.
- To simulate Genetic algorithm coding to satisfying constraints.
- To analyse the simulation results in order to get the improved efficiency and power factor.
- To fabricate the motor based on the simulation results.

1.2 ORGANISATION OF THE PROJECT

Chapter 1:

Chapter-1 deals with the objective of the project and organization of the project.

Chapter 2:

This chapter explains about the construction, operating principle of a single-phase induction motor, design of stator ,design of rotor parameter details and discuss about Genetic algorithm.

Chapter 3:

This chapter explains about the software and hardware implementation of single phase induction motor design.

Chapter 4:

This chapter contains the results obtained from the simulation and hardware details.

Chapter 5:

This chapter provides conclusion, advantages of the project and future enhancement of the project.

CHAPTER 2

SINGLE PHASE INDUCTION MOTOR

The construction of a single-phase induction motor is similar to polyphase induction motor. The stator is provided with a single phase winding and a centrifugal switch is used in some types of motors, in order to cut out a winding, used only for starting purposes. It has a distributed stator and a squirrel cage rotor. When fed from a single phase supply, stator winding produces a field or flux which is only alternative one which alternates along one space axis only. It is not a synchronously rotating flux as in the case of two or three phase stator winding, fed from a two or three phase supply. Now an alternating or pulsating flux acting on a stationary squirrel cage rotor cannot produce rotation. That is why a single-phase induction motor is not self starting. However, if the rotor of such a machine is given an initial start by hand or otherwise in either direction, then immediately a torque arises and the motor accelerates to its final speed [5]-[6].

Double-field Revolving Theory:

This theory makes use of the idea that an alternating uni-axial quantity can be represented by two oppositely – rotating vectors of half magnitude. Accordingly, an alternating sinusoidal flux can be represented by two revolving fluxes,each equal to half the value of the alternating flux and each rotating synchronously in opposite direction.

As shown in Fig 2.1 (a), let the alternating flux have a maximum value of Φ_m. Its component fluxes A and B will each be equal to $\Phi_m / 2$ revolving in anticlock wise and clock wise direction respectively. After some time ,when A and B would have rotated

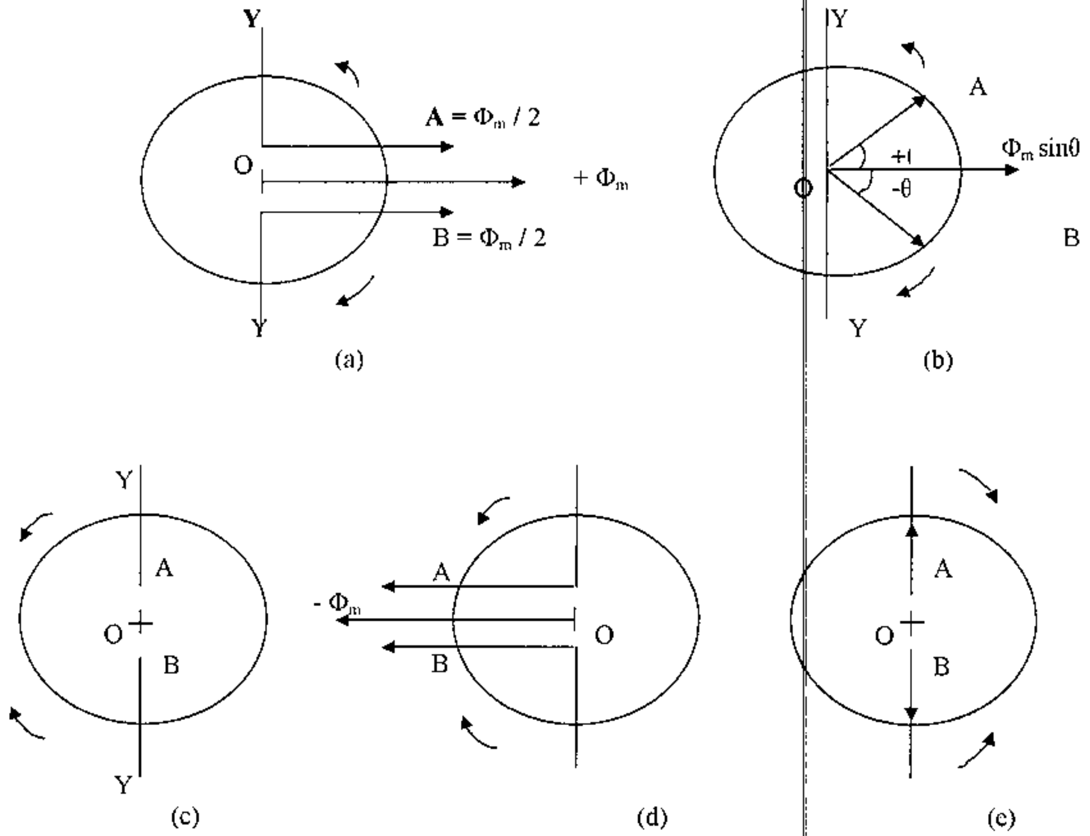
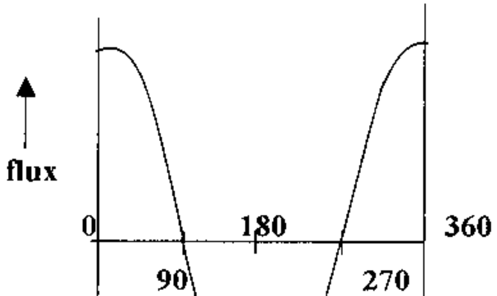


Fig 2.1 Double revolving field



After half a cycle, fluxes A anb B will have a resultant of $-2 \times (\Phi_m / 2) = -\Phi_m$ .After three quarters of a cycle, again the resultant is zero,as shown in Fig. 2.1 (e) and so on . If we plot the values of resultant flux against θ between limits $\theta = 0^\circ$ to $\theta = 360^\circ$, then a curve similar to the one shown in Fig 2.2 is obtained. That is why an alternating flux can be looked upon as composed of two revolving fluxes, each of half the value and revolving synchronously in opposite directions.

It may be noted that if the slip of the rotor is s with respect to the forward rotating flux then its slip with respect to the backward rotating flux is (2-s).

Each of the two component fluxes,.while revolving round the stator ,cuts the rotor, induces an e.m.f and this produces its own torque. Obviously, the two torques (called forward and backward torques) are oppositely – directed, so that the net or resultant torques is equal to theirdifference as shown in Fig 2.3

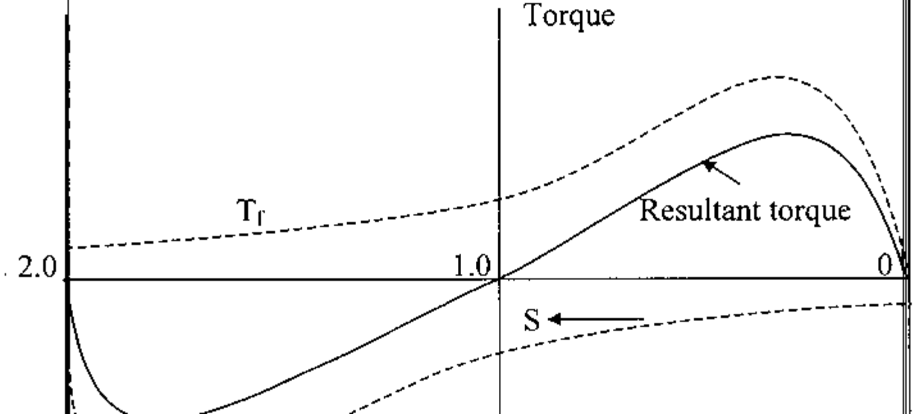
Now power developed by a rotor is $P_g = [(1-s)/s]\, I_2^2 R_2$ ----- (2.1)

If N is the rotor r.p.s., then torque is given by

$$T_g = (1/2\pi N)\, [(1-s)/s]\, I_2^2 R_2 \qquad \text{----- (2.2)}$$

$$T_g = (1/2\pi N_s)\, I_2^2 R_2 / s \qquad \text{----- (2.3)}$$

$$T_g = k \,.\, (I_2^2 R_2 / s) \qquad \text{----- (2.4)}$$



Hence, the forward and backward torques are given by

$$T_f = k . (I_2^2 R_2 /s) \qquad \text{----- (2.5)}$$

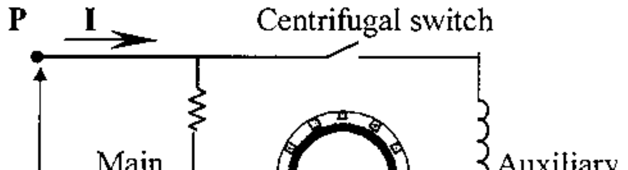
$$T_b = - k . (I_2^2 R_2 / (2-s)) \qquad \text{----- (2.6)}$$

$$\text{Total torque} \quad T = T_f + T_b \qquad \text{----- (2.7)}$$

Fig 2.3 shows both torques and the resultant torque for slips between zero and two. At standstill, $s = 1$ and $(2-s) = 1$.Hence,T_f and T_b are numerically equal but ,being oppositely directed, produce no resultant torque. That explains why there is no starting torque in a single phase induction motor. However , if the rotor is started somehow,say in the clockwise direction, the clockwise torque starts increasing and, at the same time ,the anticlockwise torque starts decreasing .Hence ,there is a certain amount of net torque in the clockwise direction which accelerates the motor to full speed.To overcome this drawback and make the motor self-starting ,it is temporarily converted into a two phase motor during starting period. For this purpose, the stator of a single phase induction motor is provided with an extra winding, known as starting or auxiliary winding,in addition to the main or running winding . The two windings are spaced 90° electrically apart and they are connected in parallel across the single-phase supply.

It is so arranged that the phase–difference between the currents in the two stator windings is very large. Hence, the motor behaves like a two phase motor. These two currents produce a revolving flux and hence make the motor self-starting.

Single-phase Capacitor Start Induction Motor:



In single-phase capacitor start induction motor, the necessary phase difference between I_a and I_m is produced by connecting a capacitor in series with the starting winding. When the motor reaches about 75% of full-load speed, the centrifugal switch S, opens and cuts out both the starting winding and the capacitor from the supply, thus leaving only the running winding across the lines [6].

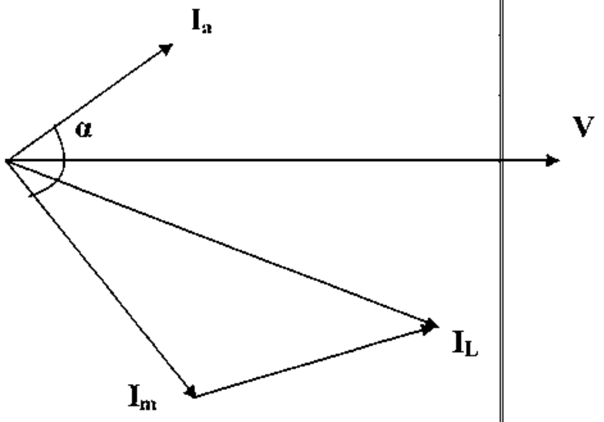


Fig 2.5 Phasor diagram of a single-phase capacitor start induction motor

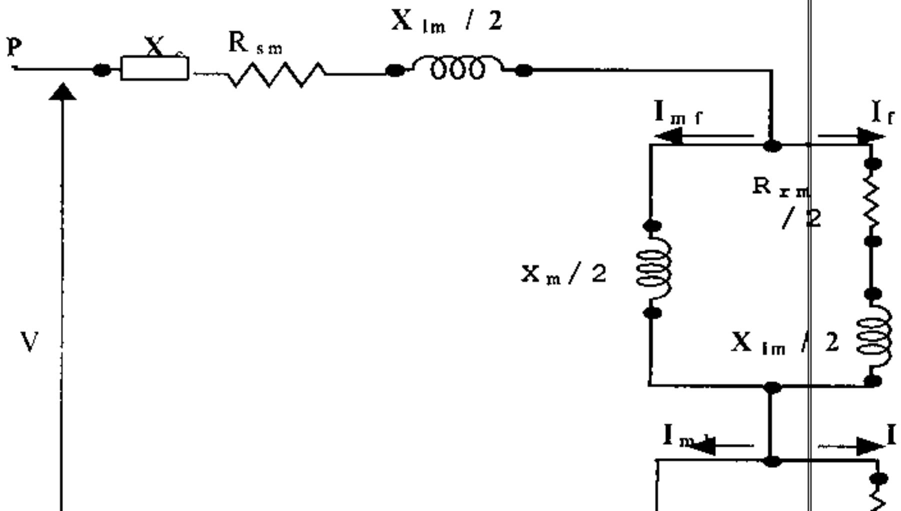
The current I_m drawn by the main winding lags the supply voltage V by a large angle where as the current drawn by the starting winding leads the supply voltage V by a certain angle. The two currents are out of phase by about 80^0 as compared to 30^0 for split phase motor. The resultant current I_L is small and it is almost in phase with the supply voltage

Capacitor start/induction run motors have several advantages over split-phase motors. Since the capacitor is in series with the start circuit, it creates more starting torque, typically 200 to 400% of rated load. And the starting current, usually 450 to 575% of rated current and it is much lower than the split-phase due to the larger wire in the start

The capacitor-start induction motor is more expensive than a comparable split phase design because of the additional cost of the start capacitor. But the application range is much wider because of higher starting torque and lower starting current. Use them on a wide range of belt-drive applications like small conveyors, large blowers and pumps, as well as many direct-drive or geared applications. These are the "workhorses" of general-purpose single-phase industrial motors. A capacitor can store reletive to the voltage applied [5].

Equivalent circuit parameters:

A single- phase motor may be looked upon as consisting of two motors, having a common stator winding, but with their respective rotors revolving in opposite directions the equivalent circuit of such a motor based on double field revolving theory. Here, the single-phase motor has been imagined to be made up of (i) one stator winding and (ii) two imaginary rotors Since iron loss has been neglected, the exciting branch is shown consisting of exciting reactance only. Each rotor has been assigned half the magnetizing reactance[5].



For any slip s, the value of I_f and I_b can be calculated,

Forward torque = $I_f^2 (R_{rm}' / 2s)$ syn.watt ----- (2.8)

Backward torque = $I_b^2 (R_{rm}' / 2(2-s))$ syn.watt ----- (2.9)

Gross motor torque = $(R_{rm}'/ 2) [(I_f^2/s) - (I_b^2/(2-s))]$ syn.watt ----- (2.10)

Net motor torque = gross motor torque – iron, friction and windage losses.

Net output = Net motor torque (1-s) watt ----- (2.11)

= Net motor torque (1-s) / 746 h.p ----- (2.12)

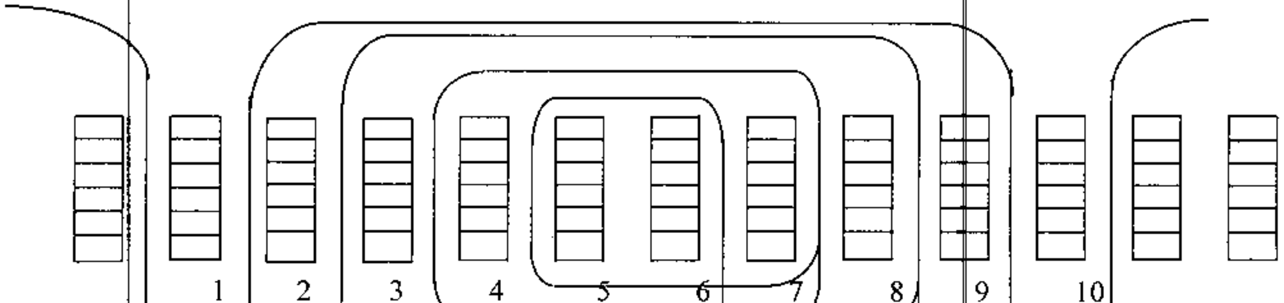
Under standstill conditions, $V_f = V_b$, but under running conditions Vf is almost 90 to 95% the applied voltage.

The total torque is $T = T_f - T_b$. ----- (2.13)

2.1 DESIGN OF STATOR

Main winding:

The stator windings of single-phase induction motors are concentric winding type. There are usually 3 or more coils per pole each having same or different number of turns. The arrangement for winding is governed largely by the necessity of minimizing harmonic fluxes which may otherwise give rise to noise and uneven accelerating torque.



Such harmonics are produced owing to non-sinusoidal shape of mmf wave. This mmf wave harmonics can be reduced by utilizing about 70 percent of the total slots for the running winding as this arrangement gives minimum low order harmonics. The remaining slots (about 30 percent of total slots) are used for accommodating the starting winding. In a small single-phase induction motor may be desirable to reduce the harmonics still further by grading the winding i.e. by having different number of conductors in each slot thereby giving an mmf wave, which nearly approaches a sine wave [5].

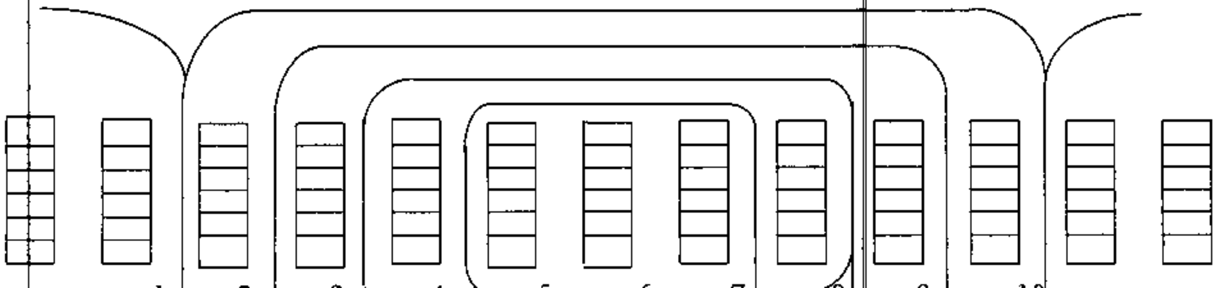
The winding arrangement shown in Fig 2.7 is for a stator with 9 slots per pole. There are four coils per pole.

Coil (1-9) --- it spans 8 slots
Coil (2-8) --- it spans 6 slots
Coil (3-7) --- it spans 4 slots
Coil (4-6) --- it spans 2 slots

The coils can be re-arranged as shown in Fig 2.8

There are again 4 coils per pole.

Coil (1-10) --- it spans 9 slots
Coil (2-9) --- it spans 7 slots
Coil (3-8) --- it spans 5 slots
Coil (4-7) --- it spans 3 slots



With this arrangement the number of turns in the outside coil (1-10) must be one half of the conductors in slot 1, the other half conductors belong to the outside coil of the adjacent pole. Suppose, for instance, that a motor with 9 slots per pole is to be wound as shown in Fig 2.7, It is desired to have a sinusoidal distribution.

The turns required in each coil are found as follows:

Coil (4-6)-sin ½ coil span = sin(2/9) x 90°= 0.342
Coil (3-7)-sin ½ coil span = sin(4/9) x 90°= 0.643
Coil (2-8)-sin ½ coil span = sin(6/9) x 90°= 0.866
Coil (1-9)-sin ½ coil span = sin(8/9) x 90°= 0.985

= 2.836

Percent turns per pole in coil (4-6) = (0.342/2.836) x 100 = 12.10
Percent turns per pole in coil (3-7) = (0.342/2.836) x 100 = 22.70
Percent turns per pole in coil (2-8) = (0.342/2.836) x 100 = 30.60
Percent turns per pole in coil (1-9) = (0.342/2.836) x 100 = 34.60

The same procedure can be used for arrangement of Fig. 2.8 except in determining the base for percent turn calculations, one half the sine of 90° must be used for coil (1-10).

Coil (4-7)-sin ½ coil span = sin (3/9) x 90° = 0.500
Coil (3-8)-sin ½ coil span = sin (5/9) x 90° = 0.766
Coil (2-9)-sin ½ coil span = sin (7/9) x 90° = 0.940
Coil (1-10)- ½ sin ½ coil span = ½ sin(9/9) x 90°= 0.500

= 2.7

Winding distribution factor:

The winding distribution factor for concentric type single layer winding is a weighted mean pitch factor and it is calculated by multiplying the pitch factor of each coil per pole group by the turns in the coil and dividing the sum of these produced by the total number of turns. For the winding shown in Fig 2.7.

Pitch factor for coil (4-6) = $\sin (2/9) \times 90^\circ = 0.342$

Pitch factor for coil (3-7) = $\sin (4/9) \times 90^\circ = 0.643$

Pitch factor for coil (2-8) = $\sin (6/9) \times 90^\circ = 0.866$

Pitch factor for coil (1-9) = $\sin (8/9) \times 90^\circ = 0.985$

Suppose turns in coil (4-6), coil (3-7), coil (2-8) and coil (1-9) are respectively $T_{4\text{-}6}$, $T_{3\text{-}7}$, $T_{2\text{-}8}$, and $T_{1\text{-}9}$.

Winding factor:

$$Kw = (0.342\ T_{4\text{-}6} + 0.643\ T_{3\text{-}7} + 0.866\ T_{2\text{-}8} + 0.985\ T_{1\text{-}9}) / (T_{4\text{-}6} + T_{3\text{-}7} + T_{2\text{-}8} + T_{1\text{-}9})$$

$$= (0.342 \times 12.1 + 0.643 \times 22.7 + 0.866 \times 30.6 + 0.985 \times 34.6) / 100 \approx 0.8$$

For usual winding distribution, the value of Kw will be between 0.75 to 0.85.

Number of turns in running winding:

The number of turns in the running winding can be calculated as below:

Stator induced voltage $E = 4.44\ F\ T_m\ K_{wm}\ \Phi_m$ ----- (2.14)

The value of stator induced voltage E is approximately equal to 95 percent of supply voltage V. The winding factor for the running winding can be assumed between 0.75 to 0.85.

The number of turns in series per pole for the main winding.

$$T_{pm} = T_m / p. \qquad \text{----- (2.17)}$$

Running winding conductors:

Current carried by each running winding conductor

$$I = (\text{h.p} \times 746) / (V \eta \cos \Phi) \qquad \text{----- (2.18)}$$

Area of main winding conductor $a_m = I_m / \delta_m$ ----- (2.19)

δm is the current density for main winding in A/mm².

For open type motors split phase, capacitor and repulsion start ,the current density can usually be 3 to 5 A/mm². For enclosed motors much lower values should be used.

Size of stator slots:

All the stator slots do not have the same number of conductors, and some contain both running winding and starting winding conductors. The starting winding conductor has a small cross sectional area, and its effect upon the size of the slot is small. Generally the running winding coil with the largest number of turns will determine the size of the slot. For semi – enclosed slots, the insulation between core and coils is placed in the slot, as slot lining .The slot linear is usually 0.3 to 0.4 mm thick.

Area required for insulated conductors = $Z_s \times (\pi/4)\, d_1^2$. ----- (2.20)

Minimum slot area required = $(1/0.5)\, Z_s \times (\pi/4)\, d_1^2$. ----- (2.21)

The slot area provided in the stamping is calculated by multiplying the mean width by the depth of the slot.

The average slot width $W_{s\,(av)} = [\,\pi\,(D + d_{ss})/S_s\,] - W_{ts}$ ----- (2.22)

Where

d_{ss} = depth of stator slot .

W_{ts} = width of stator tooth .

S_s = number of stator slots .

Area of each slot = $W_{s\,(av)} \times d_{ss}$. ----- (2.23)

Stator teeth:

The stator tooth density B_{ts} can generally be from 1.4 to 1.7 Wb/m ^{2}.

If the lower losses and noise are important or if the motor is totally enclosed, then lower densities should be used . For general purpose machines a flux density of 1.45 Wb/m ^{2} is taken while for high torque machines it may go up to 1.8 Wb/m ^{2} . A stacking factor of 0.95 is taken.

Net iron length $L_i = 0.95\, L$. ----- (2.24)

Flux density in the stator teeth $B_{ts} = \Phi_m / ((S_s/p) \times L_i \times W_{ts})$ ----- (2.25)

Stator core:

The flux density in the stator core should not exceed 1.5 Wb/m^2. Generally it lies

The flux density should also be checked in rotor teeth and rotor core. The values of flux density in rotor should not exceed the permissible values which are the same as corresponding values for stator . A little higher values are permissible in rotor.

Length of mean turn:

The length of mean turn for each of the coils per pole of a concentric winding

$$L_{ml} = [8.4\ (D+d_{ss})/\ S_s\ ] \times \text{slot spanned} + 2L. \qquad \text{----- (2.28)}$$

Airgap length:

The considerations for taking a particular air gap length are same as for three phase induction motors. The following empirical relation gives satisfactory values.

Gap length $l_g = (\ 0.007 \times \text{rotor diameter}\ ) / \sqrt{p}$ ----- (2.29)

2.2 DESIGN OF ROTOR

Number of rotor slots:

The number of rotor slots is so chosen that there is no noise producing combinations. It has been found that a quiet –running motor will result if there are no harmonic fields, of stator and rotor slots, with number of poles differing by less than 4.

Harmonic poles due to slots = $2\ (S \pm p/2)$ ----- (2.30)

Where

S is the number of slots

The number of stator slots is usually fixed by winding arrangement, number of poles

For motors with more than 2 poles, quiet operation can generally be expected, when the numbers of rotor slots are divisible by the number of pairs of poles and when the number of rotor slots differs from the number of stator slots by more than the number of poles.

Another useful rule states that the number of rotor slots should be equal to the number of stator slots plus twice the number of poles. There are many other combinations which might prove satisfactory, although the number of rotor slots are rarely more than 1.5 times the number of stator slots. The number of rotor slots must be selected from the point of view of magnetic locking [5]-[6].

Area of rotor bars:

The cage rotor winding may be either of copper bars or cast aluminium. Also technical advantages lie with copper but manufacture is cheaper with cast aluminium. Also with cast rotors, the joints between bars and end rings are eliminated.

Total stator copper section for main winding $A_m = 2\ T_m\ a_m\ \text{mm}^2$ ----- (2.31)

A high rotor resistance is desirable from the standpoint of starting torque and current but leads to high slip and poor efficiency. The total rotor copper section is generally 0.5 to 0.8 of total stator copper section.

Total cross section of rotor bars $A_r = S_r \times a_b$. ----- (2.32)

Where

Rotor resistance:

Choice of rotor resistance is very important factor in the design of single phase motors. It should be as low as possible to keep down rotor copper loss and to maintain high efficiency , high full load speed and minimum temperature rise. In the case of single phase motors there is an added advantage that the rotor resistance effects the maximum torque for given flux and therefore a higher value of pull out torque is obtained with large value of rotor resistance.

A significant parameter is the ratio R_{rm}' / X_{lm} . Experience has shown that for normal commercial fractional kilowatt machines, the value of R_{rm}' / X_{lm} is approximately:

For split phase motors	--- 0.45 to 0.55
For capacitor motors	--- 0.45 to 0.8

Where

R_{rm}' = Resistance of rotor referred to main winding

X_{lm} = Leakage reactance of stator main winding plus rotor in terms of running winding.

The value of R_{rm}' is usually lower for the larger horse power ratings. These values can be used as a guide in designing the ring or R_{rm}' choose the best ring from a number of available designs.

current would result. This will give rise to poor power factor and therefore these densities are taken only slightly higher than the corresponding stator densities[6].

Flux density in rotor teeth:

$$B_{ts} = \Phi_m / [(S_r / p) \times L_i \times W_{tr}] \qquad \text{----- (2.35)}$$

Where

W_{tr} = width of rotor teeth

Rotor core:

Flux density in rotor core

$$B_{cr} = \Phi_m / (2 \times L_i \times d_{cr}) \qquad \text{----- (2.36)}$$

Where

d_{cr} = Depth of rotor core

Auxiliary winding:

Design of auxiliary winding for starting the single-phase induction motor.

After a satisfactory main winding has been designed, the next step is to design a suitable starting or auxiliary winding .In order that the starting winding can produce a revolving field the flux set up by it must be out of phase with flux set up by the main winding. The number of turns of the main winding must satisfy the requirements of the core and the size of conductor requirements of the load. This means that the reactance of main winding is high and its resistance is low . The starting winding must have parameters just the reverse of those of main winding, It is essential to design the best possible starting winding for the main winding and rotor already designed. For any given

With single phase induction motor, the required resistance is usually obtained by using small section wire i.e. about 25% of that of main winding . The current density at starting may be as high as 100 A/mm². This is permissible as the winding is in service for about 2 seconds. The phase angle between the starting winding current and the line voltage should be about 0.4 of that for the main winding [5].

Operating characteristics:

Mmf for air gap:

The flux produced by stator mmf passes through the following parts:

(i) air gap (ii) stator teeth (iii) stator core (iv) rotor teeth and (v) rotor core due to saturation in teeth , the flux density distribution curve is flat topped. The calculation of mmf should be based upon the value of flux density at 60° from interpolar axis is 1.57 times B_{av} for single-phase machines.

$$B_{60} = 1.57\, B_{av} \qquad \text{----- (2.37)}$$

$$\text{Mmf required for airgap } At_{g60} = 800000\, B_{g60}\, K_g\, L_g \qquad \text{----- (2.38)}$$

Saturation factor:

The saturation factor which is the ratio of the total mmf required for the magnetic circuit to the mmf required for air gap is difficult to pre determine from the d.c magnetization curves. The shape of magnetizing current wave is not sinusoidal because of the non-linear nature of the magnetization curve of the core material.

Saturation factor F_s = Total mmf required for magnetic circuit / mmf required for airgap

= 1.1 to 1.35 for single-phase induction motor.

When the flux densities in teeth and core are low, the lower values should be used, and when high higher values should be used.

Iron loss:

The iron losses in stator teeth and core are found by calculating their flux densities and weights. The total iron loss for induction motors is 1.5 to 2.5 times the sum of stator tooth and core loss due to fundamental frequency flux. The multiplying factor should be obtained from tests of motors of similar design. When test data is not available, a value of 1.75 to 2.2 may be used.

Friction and windage loss:

The bearing friction and windage loss will depend upon the type of bearing to be used. whether ball bearing or sleeve bearing . For sleeve bearings and a speed 1500 rpm, it is usually from 4.0 to 8.0 % of the watt output. The high values apply to small motors below 180 W [5].

Running winding resistance:

$R_{rm} = 0.021 \; [\, T_m \, L_{mtm} / a_m]$ at 75 ° c (hot) ----- (2.39)

$R_{rm} = 0.017 \; [\, T_m \, L_{mtm} / a_m]$ at 20 ° c (cold) ----- (2.40)

Rotor resistance:

We can calculate the value of rotor resistance referred to stator by putting $m_s = 2$ for single-phase machines.

Rotor resistance referred to running winding

$$R_{rm}' = 4\, m_s\, T_s^{2}\, K_{ws}^{2}\, \rho\, [\, (\, L_b / S_r\, a_b\,) + (\, 2/\pi\, (\, D_s / p^2\, a_s\,)\,)\, K_{ring}\,] \qquad \text{----- (2.41)}$$

$$R_{rm}' = 8\, T_s^{2}\, K_{ws}^{2}\, \rho\, [\, (\, L_b / S_r\, a_b\,) + (\, 2/\pi\, (\, D_s / p^2\, a_s\,)\,)\, K_{ring}\,] \qquad \text{----- (2.42)}$$

Leakage reactance calculations of single-phase induction motor:

Slot leakage reactance:

The windings of induction motors are concentrictype with different number of conductors in each slot. Suppose the number of conductors in different slot are Z_1, Z_2, Z_3 ……etc. Therefore , there are 2p groups of conductors of Z_1, Z_2, Z_3 ……etc. conductors per slot.

Hence the total stator slot leakage reactance

$$X_{ss} = 2\,\pi\, f\, T^{2}\, L\, \lambda\, . \qquad \text{----- (2.43)}$$

$$= 2\,\pi\, f\; L\, [Z_1^2 + Z_2^2 + Z_3^2 + \ldots\ldots.]\, L\, \lambda_{ss} \times 2p.$$

$$= 4\,\pi\, f\; L\, [Z_1^2 + Z_2^2 + Z_3^2 + \ldots\ldots.]\, p\, L\, \lambda_{ss}. \qquad \text{----- (2.44)}$$

$$= 4\,\pi\, f\; (Z\, K_{wm})^{2} \times (L / S_s)\; \lambda_{ss}\, C_{\alpha}. \qquad \text{----- (2.45)}$$

$$= 16\pi f\, (T_m\, K_{wm})^{2} \times (L / S_s\,)\; \lambda_{ss}\; C_{\alpha}. \qquad \text{----- (2.46)}$$

$$\lambda_{ss} = \mu_o [(f + (d/e)+2e/(e+a1)] \qquad \text{----- (2.48)}$$

$$\lambda_{sr} = \mu_o [(h / 3W_s)+(h_d / W_o)] \qquad \text{----- (2.49)}$$

Rotor slot leakage reactance:

The slot leakage reactance is proportional to the specific slot permeance divided by the number of slots. Thus the rotor slot leakage reactance, in terms of the stator, may be obtained from this equation by substituting rotor specific permeance λsr for stator slot specific permeance $\lambda_{ss}\, C_\alpha$ and S_r for S_s.

Thus rotor slot leakage reactance in terms of stator

$$X_{sr}' = 16\pi f (T_m K_{wm})^2 (L / S_s)\ \lambda_{sr}. \qquad \text{----- (2.50)}$$

Total slot leakage reactance in terms of stator $X_s = X_{ss} + X_{sr}'$ ----- (2.51)

$$X_s = 16\pi f (T_m K_{wm})^2 (L / S_s) [\lambda_{ss}\ C_\alpha + (S_s / S_r) \lambda_{ss}]. \qquad \text{----- (2.52)}$$

Zig –Zag leakage reactance:

$$X_z = 16\pi f (T_m K_{wm})^2 (L / S_s) \lambda_s \qquad \text{----- (2.53)}$$

Where

$$\lambda_s = [\mu o\ W_{ts} W_{tr} (W_{ts}^2 + W_{tr}^2)] / [12\, l_g\, Y_{ss}^2 Y_{sr}]. \qquad \text{----- (2.54)}$$

$$Y_{ss} = \pi * D / S_s., \ Y_{sr} = \pi\ D / S_r . \qquad \text{----- (2.55)}$$

Over hang leakage reactance:

$X_o = 16\pi f (T_m K_{wm})^2 ((L\, \mu_o) / 6.4S_s\, p) [\pi (D+d_{ss})$ x average coil span in slots]) ----- (2.56)

Where,

θ_s = rotor bar skew angle expressed in radian ;

= $[\pi / (S_r / P)]$ x (rotor slot pitches through which bars are skewed)

K_l - stator slot leakage factor ≈ 0.95

X_m - magnetizing reactance.

Magnetizing reactance:

$$X_m = 16\pi f \{T_m K_{wm})^2 \ ((\mu_o L \tau) /(10 L_g K_g p F_s) \quad \text{----- (2.58)}$$

Where,

F_s = saturation factor.(1.1 to 1.85)

$K_g = K_{gs} \ K_{gr}$ ----- (2.59)

$K_{gs} = Y_{ss} / [Y_{ss}$- (ratio)slot opening]. ----- (2.60)

$K_{gr} = Y_{sr} / [Y_{sr}$ – (ratio)slot opening]. ----- (2.61)

Capacitive reactance:

$$X_c = X_a + (R_a R_m)/(X_m). \quad \text{----- (2.62)}$$

Where

C - capacitance value

$X_c = 1/(2 \pi f C)$ ----- (2.63)

Total leakage reactance:

Total leakage reactance referred to main winding plus rotor in terms of running

$$X_{om} = X_m + X_{lm} / 2 \quad ----- (2.65)$$

The leakage flux factors

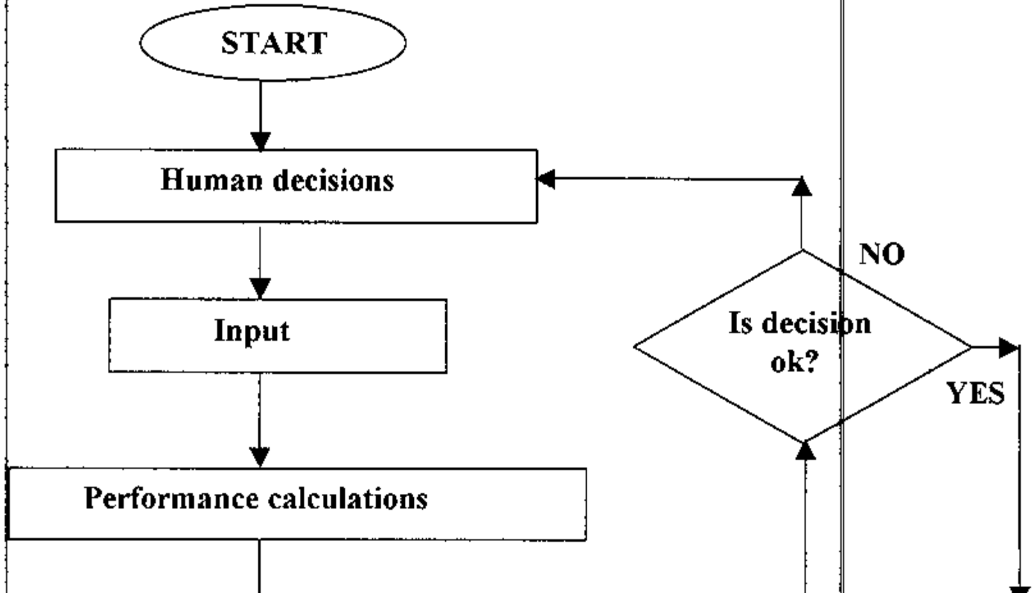
$$K_r = (X_{lm} - X_{om}) / X_{om.} \quad ----- (2.66)$$

$$K_s = \sqrt{} ((X_{om} - X_{lm}) / X_{om}) \quad ----- (2.67)$$

2.3 OPTIMAL DESIGN CONCEPTS

Design may be defined as a creative physical realization of theoretical concepts. Design is a process to accomplish a specified task optimally, under certain solution constraints. Design has been defined as both art and science. It is a science because it follows established and universally accepted physical and mathematical principles which have been verified by the experimental methods and an art because of uncertain factors and decisions involving judgment or experience. analysis and synthesis are two methods used in designing[4].

Design analysis:



The design analysis may be defined as the determination of the performance of a system from knowledge of the relationships among the variables with the system parameters defined initially remaining unchanged. The design analysis is an excellent starting point for one beginning with use of application of digital computers for the design of electrical machines. The flow chart for common design analysis is shown in Fig 2.9.

Basic summary:

In order to use optimization algorithms in engg design activities the first task is to formulate the optimization problem. The formulation process begins with identifying the important design variables that can be changed in design. The other design parameters are usually kept fixed. There after, constraints associated with the design are formulated. The constraints may arise due to resource limitations such as deflection limitations, strength limitations, frequency limitations and others. Constraints may also arise due to code restrictions that govern the design. The next task is to formulate the objective function, which the designer is interested in min (or) max. The final task of the formulation phase is to identify some bounding limits for the design variables. The formulation of an optimizing problem can be more difficult than solving the optimization problem; unfortunately every optimization problem requires different considerations for formulating objectives, constraints and variables.

Optimal design:

Optimal design of electrical machines is an iterative process wherein assumed data may have to be varied many times to arrine at the desired design. The associated with the design are formulated. The constraints may arise due to resource limitations such as deflection limitations, strength limitations, frequency limitations and others. Constraints may also arise due to code restrictions that govern the design. The next task is to formulate the objective function, which the designer is interested in min (or) max. The final task of the formulation phase is to identify some bounding limits for the design variables. The formulation of an optimizing problem can be more difficult than solving the optimization problem; unfortunately every optimization problem requires different considerations for formulating objectives, constraints and variables.

Classification of optimization methods:

The methods of optimization are classified in several ways as below

i) Based on the existence of constraints

Depending on whether constraints exist or not optimization can be classified into two types:

(a) constrained optimization

(b) un constrained optimization.

ii) Based on physical structure of the problem

(a) optimal problems

(b) Non – optimal problems

iv) Based on permissible values of design variables

(a) Integer valued

(b) Real – valued programming methods.

v) Based on the nature of design variables

(a) The problem is to find values to a set of design parameters which make some prescribed function of these parameters minimum subject to certain constraints.

(b) The objective is to find a set of design parameters consisting of all continuous functions of some other parameter that optimizes the objective function subject to the prescribed constraints.

Objective function:

The common engineering objectives involve minimization of overall cost of manufacturing (or) minimization of overall weight of an overall weight of a component or maximization of net profit earned (or) maximization of product. Although most of the above objectives can be quantified, there are some objectives that may not be quantified easily.

The objective function can be of two types either the objective function is to be maximized or it has to be minimized. The optimization algorithms are usually written either for minimization problems or for maximization problems. The duality principle helps by allowing the algorithm to be used for minimization or maximization with a minor change in the objective function instead of a change in entire algorithm. If the algorithm is developed for solving minimization problem, it can also be used to solve a

Mathematical formulation of design problems:

The typical form of a non linear programming problem in machine design is expressed as,

Optimize F(x) by determined x $(x_1, x_2, x_3 \ldots\ldots x_n)$

Subject to constraints ,

$G_i(x) \leq 0$ (for $j = 1,2,3 \ldots\ldots m$)

and $X_{min} < X_i < X_{maxi}$ (for $i = 1,2,\ldots.n$)

Where F(x) is the objective function ,

$x(x_1, x_2, x_3 \ldots\ldots x_n)$ is the set of independent design variables with the minimum and maximum limits as X_{min} and X_{maxi}.

2.4 DESIGN OPTIMIZATION OF SINGLE PHASE INDUCTION MOTOR

The optimal design of single-phase induction motor is usually formulated as a general nonlinear programming problem [4].

Optimize f (x)

Such that X exists within the n- dimensional feasible region D.

$X \varepsilon D$, where $D = \{x \mid x >= 0 , g_i(x) < = 0, h_i(x) = 0, i=1 \text{ to } n\}$

In the above equations, f(x) g(x) are real valued scalar functions and vector x comprises the n principal variables for which the optimization is to be performed. The function f(x) is called the objective function, for which the optimal values of x result in the maximum value for f(x), and these optimal values satisfy the given constraints. The

Optimal problem formulation:

In many industrial design activities a native optimal design is achieved by comparing a few alternative design solutions created by using a problem knowledge. In such an activity, the feasibility of each design solution is first investigated. Thereafter an estimate of the underlying objective (cost, profit) of each design solution is computed and the best solution is adopted. This method is often followed because of the time and resource limitations. But in many cases this method is followed simply because of the lack of knowledge of the existing optimization procedures [4]-[5].

We begin our discussion with the formulation procedure by mentioning that it is almost impossible to apply a single formulation procedure for all engg design problems. Since the objective in a design problem and the associated design parameters vary from product to product, different techniques need to be used in different problems. The purpose of the formulation procedure is to create a mathematically model of an optimal design problem, which then can be solved using an optimization algorithm. Since an optimization algorithm accepts an optimization problem in a particular format, every optimal design problem must be formulated in that format.

Design variables:

The formulation of an optimization problem begins with identifying the underlying design variables, which are primarily varied during the optimization process. A design problem usually involves many design parameters of which some are highly sensitive to the proper working of the design. These parameters are called design variables in the of optimization procedures. Other (not so important) design parameters usually remain fixed or vary in relation to the design variables. There is no rigid guideline variables. The efficiency of the optimization process can be increased. The first thumb rule of the formulation of an optimization problem is to choose as few design variables as possible. The outcome of that optimization procedure may indicate whether to include more design variables in a revised formulation or to replace some previously considered design variables [4].

- Internal diameter of stator
- External diameter of stator.
- Winding turns ratio.
- Number of stator slots
- Depth of stator slots
- Width of stator slots
- Capacitance value of capacitor
- Length of stator stack.
- Cross sectional area of stator conductors
- Number of rotor slots
- Length of mean turn of main winding
- Number of parallel path
- Number of conductors in parallel
- Length of air gap.
- Number of main winding turns.
- Number of auxiliary winding turns.
- Height of stator stator slot.
- Height of rotor slot

- Current density of main winding ≤ 3 to 5 A/mm²
- Full load power factor ≥ 0.8.
- Efficiency > 0.75
- Length of airgap ≤ 0.32 to 0.7 mm
- Area of rotor bars ≤ 17 to 20 mm^2
- Stator core flux density ≤ 0.8 to 1.5 wb / m^2
- L / T ratio ≤ 0.5 to 1.2
- Length of core ≤ 0.07 to 0.1 mm

2.4 GENETIC ALGORITHM

In order to use optimization algorithms in engineering design activities, the first task is to formulate the optimization problem. The formulation process begins with identifying the important design variables that can be changed in a design. The other design parameters are usually kept fixed. Thereafter, constraints associated with the design are formulated. The constraints may arise due to resource limitations such as deflection limitations, strength limitations, frequency limitations, and others. Constraints may also arise due to codal restrictions that govern the design. The next task is to formulate the objective function, which the designer is interested in minimizing or maximizing. The final task of the formulation phase is to identify some bounding limits for the design variables[9].The formulation of an optimization problem can be more difficult than solving the optimization problem. Unfortunately, every optimization problem requires different considerations for formulating objectives, constraints, and variable bounds. Thus, it is not possible to describe all considerations in a single book. However, many of these considerations require some knowledge about the mechanism is

A roulette-wheels for five individuals having different fitness values. Since the third individuals has a higher fitness value

Point	Fitness
1	25.0
2	5.0
3	40.0
4	10.0
5	20.0

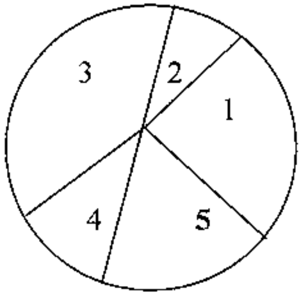


Fig 2.10 Roulette wheel marked for five individuals accoring to their fitness values.

Than any other, it is expected that the roulette-wheel selection will choose the third individual more than any other individual. This roulette-wheel selection scheme can be simulated easily. Using the fitness value f_i of all strings, the probability of selecting a string P_i can be calculated. Thereafter, the cumulative probability (P_i) of each string being copied can be calculated by adding the individual probabilities from the top of the list. Thus, the bottom-most string in the population should have a cumulative probability (P_n) equal to 1. The roulette-wheel concept can be simulated by realizing that the i-th string in the population represents the cumulative probability values from P_{i-1} to P_i. The first string represents the cumulative values from zero to P_1. Thus, the cumulative probability any string lies between 0 to 1. In order to choose n strings, n random numbers between zeros to one are created at random. Thus, a string that represents the chosen random number in the cumulative probability range for the string is copied to the mating pool. This way the string with a higher fitness value will represent a larger range in the cumulative probability values and therefore has a higher probability of being copied into

In reproduction, good strings in a population are probabilistically assigned a larger number of copies and a mating pool is formed. It is important to note that no new strings are formed in the reproduction phase. In the crossover operator, new strings are created by exchanging information among strings of the mating pool. Many crossover operators exist in the GA literature. In most crossover operators, two strings are picked from the mating pool at random and some portions of the strings are exchanged between the strings. A single-point crossover operator is performed by randomly choosing a crossing site along the string and by exchanging all bits on the right side of the crossing site as shown:

$$\begin{array}{cc|c} 0\,0 & 0\,0\,0 \\ 1\,1 & 1\,1\,1 \end{array} \Rightarrow \begin{array}{cc|c} 0\,0 & 1\,1\,1 \\ 1\,1 & 0\,0\,0 \end{array}$$

The two strings participating in the crossover operation are known as parent strings and the resulting strings are known as children strings. It is intuitive from this construction that good sub strings from parent strings can be combined to form a better child string, if an appropriate site is chosen. Since the knowledge of an appropriate site is usually not known beforehand, a random site is often chosen. With a random site, the children strings produced may or may not have a combination of good substrings from parent strings, depending on whether or not the crossing site falls in the appropriate place. But we do not worry about this too much, because if good strings are created by crossover, there will be more copies of them in the next mating pool generated by the reproduction operator. But if good strings are not created by crossover, they will not survive too long, because reproduction will select against those strings in subsequent generations.

in the crossover operation and 100 (1- P_c) percent of the population to a fixed mapping rule. Usually, the following linear mapping rule is used:

$$X_i = X_i^L + [(X_i^U - X_i^L) / (2^L - 1)] \text{ x decoded value } (s_i). \qquad \text{---- (2.69)}$$

In the above equation, the variable x_i is coded in a sub string si of length l_i. The decoded value of a binary sub string s_i is calculated as $\sum^{L-1}_{i=0} 2^i s_i$ where S_i ε (0,1) and the string is represented as ($s_{i-1} s_{i-2} \ldots s_2 s_1 s_0$)).

Fitness function:

As pointed out earlier, GAs mimic the survival of the fittest principle of nature to make a search process. Therefore, GAs are naturally suitable for solving maximization problems. Minimization problems are usually transformed into maximization problems by some suitable transformation. In general a fitness function F *(x)* is first derived from the objective function and used in successive genetic operations. Certain genetic operators require that the fitness function be non negative, although certain operators do not have this requirement. For maximization problems, the fitness function can be considered to be the same as the objective function of F (x) = f (x). For minimization problems, the fitness function is an equivalent maximization problem chosen such that the optimum point remains unchanged. A number of such transformations are possible. The following fitness function is often used:

$$F(x) = 1 / (1 + f(x)) \qquad \text{---- (2.70)}$$

This transformation does not alter the location of the minimum, but converts a

The population is then operated by three main operators – reproduction, crossover, and mutation – to create a new population of points. The new population is further evaluated and tested for termination. If the termination criterion is not met, the population is iteratively operated by the above three operators and evaluated. This procedure is continued until the termination criterion is met. One cycle of these operations and the subsequent evaluation procedure is known as a generation in GA's terminology [9].

GA operators:

Reproduction is usually the first operator applied on a population. Reproduction selects good strings in a population and forms a mating pool. That is why the reproduction operator is sometimes known as the selection operator. There exist a number of reproduction operators in GA literature, but the essential idea in all of them is that the above-average strings are picked from the current population and their multiple copies are inserted in the mating pool in a probabilistic manner. The commonly- used reproduction operator is the proportionate reproduction operator where a string is selected for the mating pool with a probability proportional to its fitness.Thus,the i-th string in the population is selected with a probability proportional to f_i . Since the population size is usually kept fixed in a simple GA, the sum of the probability of each string being selected for the mating pool must be one. There fore, the probability of selecting the i-th string is

$$P_i = f_i / (\Sigma^{n}_{j-1} \quad f_i) \qquad \text{-----} \ (2.71)$$

Where n is the population size. One way to implement the selection scheme is to imagine a roulette- wheel with its circumference marked for each string proportionate to the string's fitness. The roulette-wheel is spun n times, each time selecting an instance of

The most striking difference between GAs and many traditional optimization methods are that GAs work with a population of points instead of a single point. Because there are more than one string being process simultaneously, it is very likely that the expected GA solution may be a global solution. Even though some traditional algorithms are population -based, like Box's evolutionary optimization and complex search methods do not use previously obtained information efficiently. In GA, previously found good information is emphasized using reproduction operator and propagated adaptively through crossover and mutation operators. Another advantage with a population -based search algorithm is that multiple optimal solutions can be captured in the population easily, thereby reducing the effort to use the same algorithm many times.

In discussing GA operators or their working principles in the previous section, nothing has been mentioned about the gradient or any other auxiliary problem information. In fact, GAs do not require any auxiliary information except the objective function values. Although the direct search methods used in traditional optimization methods do not explicitly require the gradient information, some of those methods use search directions that are similar in concept to the gradient of the function. Moreover, some direct search methods work under the assumption that the function to be optimized is unimodal and continuous. In GAs, no such assumption is necessary[10].

One other difference in the operation of GAs is the value of probabilities in their operators. None of the genetic operators work deterministically. In the reproduction operator, even though is expected to have f_i / f copies in the mating pool, a simulation of the roulette-wheel selection scheme is used to assign the true number of copies. In the crossover operator, even though good strings are crossed, strings to be crossed are created at random and cross-sites are created at random. In the mutation operator, a random bit is to solve a multimodal problem with many local optimum points, search procedures may easily get trapped in one of the local optimum points. The objective function has one local minimum and one global minimum. If the initial point is chosen to be a point remains as they are in the current population.

A crossover operator is mainly responsible for the search of new strings, even though a mutation operator is also used for this purpose sparingly. The mutation operators changes 1 to 0 and vice versa with a small mutation probability, $\mathbf{P_m}$. The bit-wise mutation is performed bit by bit by flipping a coin with a probability $\mathbf{P_m}$ If at any bit the out come is true then the bit is altered; otherwise the bit is kept unchanged. The need for mutation is to create a point in the neighbourhood of the current point, thereby achieving a local search around the current solution. The mutation is also used to maintain diversity in the population. For example, consider the following population having four eight-bit strings:

0110 1011
0011 1101
0001 0110
0111 1100

Notice that all four strings have a 0 in the left-most bit position. If the true optimum solution requires 1 in that position, then neither reproduction nor crossover operator described above will be able to create 1 in that position. The inclusion of mutation introduces some probability (NP_m) of turning 0 into 1.

These three operators are simple and straightforward. The reproduction operator created, they will be eliminated by the reproduction operator in the next generation and if good strings are created, they will be increasingly emphasized[9].

The parameters used in genetic algorithm are as follows:

Scaling factor - 0.001
Population size - 20
Maximum generation number - 500

CHAPTER 3

IMPLEMENTATION

A design software for single-phase induction motor has been developed using a programming language VC++. The design of 230 Volts, 0.5 h.p, 4 poles, 50 Hz single-phase induction motor is implemented and hardware is fabricated.

3.1 SOFTWARE IMPLEMENTATION

The optimal design of the single-phase induction motor has been implemented using genetic algorithm (GA). The GA is implemented using programming language VC++.

The implementation of GA with the help of VC++ has following advantages:

- The computation time is much lesser when compared to other software tools such as Matlab- Genetic algorithm tool box (GAOT).
- The object oriented programming concept helps to build the algorithm more effectively thereby producing accurate results.

In this the optimal design of single-phase capacitor start induction motor, efficiency and power factor are taken as the objective function and the constraints are the turns ratio, length of stator stack, current density of main winding conductor, air gap length and the cross sectional area of main winding conductor. Then the coding is simulated and the results are used in fabrication of the single-phase capacitor start induction motor.

Algorithm:

Step 1: Choose a coding to represent problem parameters, a selection operator, a

Based on the simulation results, the motor is fabricated with the specifications detailed below.

Horse power : 0.5 H.P
Rated Voltage : 230 V
Number of Poles : 4
Frequency : 50 Hz
Rated current : 2.5A
Capacitor : 100μF

Table 3.2 : Optimal design design parameters of 0.5 hp ,4 pole, 230v, 50 Hz single phase induction motor

Parameters	**Main winding**	**Auxiliary winding**
Wire size (SWG)	18	22
Number of turns	308	652
Pitch	(1-9) (2-8) (3-7) (4-6)	(1-10) (2-9) (3-8) (4-7)
Length of mean turn (m)	0.203	0.34
Resistance (Ω)	1.76	9.65
Current density (A/mm^2)	3.301	22.5

Rotor design:

Length of air gap (mm) = 0.321
Rotor diameter (m) = 0.0932
Area of rotor (mm^2) = 19.57

CHAPTER 4

RESULTS

Genetic algorithm has been applied to solve the single-phase induction motor design problem and it was demonstrated with a test motor. The genetic algorithm has been used to optimize induction motor design problem with objective functions. With efficiency as objective function, the motor parameters are changed significantly thereby resulting the improvement of efficiency. The number of turns in main winding has suitably changed according to the input data. By optimizing the motor design with respect to efficiency results in improved operating performance. The simulation results are presented in table 4.1.

4.1 SIMULATION RESULTS

```
SUPER BEST  OPTIMAL EFFICIENCY:  0.806521
POWERFACTOR:  0.752918
Diameter of core  D: 0.109524
Length of core    L: 0.0971429
Flux density of stator teeth Bts:  1.32857
Current density  Isd:  3.03175
Length of airgap lg:0.32
Area of rotor bars ab: 19.5714
L/polepitch ratio  LTOWR :  1.12988
Stator core flux density Bcs  1.19571
Area or rotor/area of main winding Aratio  1.80813

FLUX PER POLE :   0.00398449
DEPTH TO WIDTH OF STATOR SLOT RATIO   :2.8473
STATOR CORE FLUX DENSITY:  1.15286
Turns in series per pole : Tpm  77
turns in each coil  :
TOTAL TURNS IN SERIES FOR MAIN WINDING:   308
LENGTH OF MEAN TURN: Lmtm0.203513
total area of conductors in main winding: Am459.333
```

```
MAGNETISING REACTANCES      Xm  42.1935
SKEW LEAKAGE REACTANCE      xsk  0.409189
TOTAL LEAKAGE REACTANCE     Xlm  2.3686
```

The computation time required for genetic algorithm to converge to an optimal solution is listed in table 4.1

Table 4.1 : Simulation results for optimal design compared with conventional design.

S.No	Parameters	Conventional method	Proposed Genetic algorithm
1	Stator bore diameter (m)	0.109	0.1095
2	Stator stack of length (m)	0.075	0.097
3	Current density in main winding (A/mm^2)	4.8	3.031
4	Stator core flux density (Wb/m^2)	1.12	1.195
5	Area of rotor bars (mm^2)	20	19.57
6	Length of air gap(mm)	0.32	0.321
7	Stator tooth flux density (Wb/m^2)	1.1	1.328
8	Turns in series per pole	114	77
9	Resistance of main winding (Ω)	2.5	1.765
10	Resistance of rotor referred to main winding (Ω)	2.51	1.435

4.2 HARDWARE RESULTS

The design of a single-phase capacitor srart induction motor specifications 0.5 h.p, 230 volts , 4 pole, 50 Hz induction motor has been optimized and fabricated.

The auxiliary winding is not changed.

Main winding:

Standard wire gauge (SWG) = 18
Turns in series per pole = 77
Length of mean turn (m) = 0.203
Resistance of main winding (Ω) = 1.76
Current density (A/mm^2) = 3.301

Rotor design:

Length of air gap (mm) = 0.321
Rotor diameter (m) = 0.0932
Area of rotor (mm^2) = 19.57

Load test:

Table 4.2: Load test

S.No	Voltage (volts)	Current (amps)	Power (watts)	Speed (rpm)	Frequency (hz)	S1 (kg)	S2 (kg)
1	230	2.5	500	1390	49	3.7	1.0

Blocked rotor test:

Table 4.3: Blocked rotor test

Performance:

Efficiency = (Output power / Input power) x 100.

= (370 / 500) x 100.

= 74 %

Power factor = power / (v x i)

= 500 / (230 x 2.5)

= 0.869

From the results, the efficiency of motor is increased by 9% and the power factor is improved from 0.7 to 0.869 .

CHAPTER 5

CONCLUSION

In this project, a Genetic algorithm is adopted for the optimum design of single-phase capacitor start induction motor. This procedure employs genetic algorithm to search for optimal values of independent variables. The optimum design of a single-phase capacitor start induction motor has been achieved for maximizing efficiency and power factor with the constraints such as main dimensions, current density in main winding, length of air gap, flux density and area of rotor bars. The simulation results are verified with experimental results. The energy conservation is achieved by implementing the optimized design in single-phase induction motors. Also the developed GA based software can be used for all single-phase induction motors with minor modification, which is much useful for design engineers.

Advantages and applications of the project:

- The developed software can be used to optimize the design parameters for all Fractional Horse Power (FHP) induction motors.
- The overall efficiency of the motor has been increased.
- The developed hardware can be used for compressor, pump applications
- Optimization of running performances such as losses and power factor

Future enhancement of the project:

- Design of controller for efficient control using power electronic devices.

REFERENCES

[1] S.Subramanian and R.Bhuvaneswari,"Optimization of single phase induction motor design using evolutionary programming," *Journal of Systems and Engineering, Paritantra,* vol 12 ,pp 30-37 ,2005.

[2] E.F. Fuchs, J.C.White and J.Holi. "Design analysis of capacitor run single phase induction motor," *IEEE Transactions on Energy Conversion*. Vol.5, no.2 , 1990.

[3] P.Pillay, V.Levin, P.Otaduy and J.Rueck, " Induction motor efficiency determination using genetic algorithm, " *IEEE Transactions on Energy Conversion.* Vol 13, no.5, pp 326-333, 1998

[4] M.Ramamoorty, *Computer Aided Optimal Design For Electrical Machines,* New Delhi: East-west Press, 1997.

[5] Dr.A.K.Sawhney, *Electrical machine design,* New Delhi: Dhanpat rai & co, 2001.

[6] Cyril .G. Veinott , *Fractional and Sub-Fractional Horse Power Electric Motors,* fourth edition, McGraw-Hill International Edition ,1992.

[7] H.Huang, E.F.Fuchs and J.C. White, "Optimization of single phase induction motor design – part I formulation of the optimization technique," *IEEE Transactions on Energy Conversion* , Vol 3, no 2, pp 349-356, 1998.

[8] H.Huang, E.F.Fuchs and J.C. White ,"Optimization of single phase induction motor design – part II maximum efficiency and minimum cost of optimal design," *IEEE Transactions on Energy Conversion,* Vol 3, no 2, pp 357- 363,1998.

[9] Kalyanmoy Deb. *Optimization for Engineering Design, Algorithms and Examples*. New Delhi: Prentice-hall of India Private Limited, 2001.

APPENDIX I

SOFTWARE CODE

```
#include<iostream.h>
#include<cmath>
#include<stdlib.h>
#include<fstream.h>
#include<time.h>
ifstream fin("run.dat");
ofstream fout("run.out");
#define SLN 6
#define POP 20
#define  NX 6
#define noit 500

void main()
{
        int P,V,freq,pole,Sslot,Rslot;
        int bestcnt;
        double
c1,Ltowr[POP+1],FLE,PF,AN,SN,ns,coecos,DsL,polepitch[POP+1],L[noit+1][POP+1],
D[noit+1][POP+1];
        double
Li[POP+1],Sslotpitch,Wts,Bts[noit+1][POP+1],fluxperpole[POP+1],Do[POP+1],dss[PO
P+1],dsc[POP+1],Ws[POP+1],Bcs[POP+1],ae[POP+1];
```

```
1],ab[noit+1][POP+1],Ar[POP+1],Aratio[POP+1],rsm[POP+1],Lb,wb,c2[POP+1],rrm[P
OP+1];
        double
c3[POP+1],xs[POP+1],ysr[POP+1],yss[POP+1],C4[POP+1],xz[POP+1],xo[POP+1],Xm[
POP+1],xsk[POP+1],Xlm[POP+1],Xom[POP+1],Kl[POP+1],Cl,FWl,Tl,Im[POP+1],rrm
0[POP+1];
        double
F1[POP+1],F2[POP+1],F3[POP+1],F4[POP+1],F5[POP+1],F6[POP+1],F7[POP+1],F8[
POP+1],F9[POP+1];
        double
H1[POP+1],H2[POP+1],H3[POP+1],H4[POP+1],H5[POP+1],H6[POP+1],H7[POP+1],H
8[POP+1],H9[POP+1],H10[POP+1],H11[POP+1],H12[POP+1],H13[POP+1],H14[POP+
1],H15[POP+1],H16[POP+1],H17[POP+1],H18[POP+1],H19[POP+1],H20[POP+1],H2
1[POP+1],H22[POP+1],H23[POP+1],H24[POP+1],H25[POP+1],H26[POP+1],H27[POP
+1],H28[POP+1],H29[POP+1],H30[POP+1],H31[POP+1],H32[POP+1],H33[POP+1],H
34[POP+1],H35[POP+1],H36[POP+1],H37[POP+1],H38[POP+1];
        double eff[POP+1],bpf[noit+1];
        double powf[POP+1],bltowr[noit+1],bbcs[noit+1],bAratio[noit+1]
        int i,j,k;
        double vbest[noit][NX]={0}; //Declarations were done here
        double best[noit+1],minval[POP+1][NX+1],bestk[noit+1];
        double
vmin[NX+1],vmax[NX+1],x[POP+1][NX+1],f[POP+1],ff[POP+1],y[POP+1][NX+1];
   unsigned int a;
   int xyz[POP+1][NX+1][SLN+1]={0},val[POP+1][NX+1]
={0},pqr[POP+1][NX+1][SLN+1]={0};
```

```
        //cout<<ns<<"\n";
//      coecos=16.5;
//DsL=(370)/(coecos*ns)/1000;
        //cout<<DsL<<"\n";
        //c1=(3.14/pole)*Ltowr;
        for( i=1;i<=NX;i++)
    {
     fin>>vmin[i]>>vmax[i]; // Min Max values of the generations were read here
    fout<<vmin[i]<<vmax[i]; // Min Max values of the generations were given out
    }
fout<<endl;
```

```
     for( j=1;j<=NX;j++)

       int   sum =0,num=0,c,n;
       for( k=0,n=SLN;k<SLN;k++,n--)
                 {
            c = pow(2,k);
             num = xyz[i][j][n] * c ;
                                  sum +=num;
       }
      val[i][j] = sum;

    }
  }

//*************************************
fout<<endl;
fout<<"\n BITS AND CORRESPONDING VALUES \n";
 for( i=1;i<=POP;i++)
  {

    for( j=1;j<=NX;j++)
       {

       for( k=1;k<=SLN;k++)
       {
```

```
            fout<<endl;
   }
   int   dr = pow(2,SLN) -1;
  fout<<"VALUES BETWEEN MIN AND MAX \n  ";

    for( i=1;i<=POP;i++)
     {   fout<<endl;

     for( j=1;j<=NX;j++)
          {

       x[i][j] = vmin[j] + ( (vmax[j] - vmin[j]) /dr * val[i][j]  );
       minval[i][j] = x[i][j];
         fout<< "\t "<<val[i][j]<<"\t"<<x[i][j]<<"\t";
          }
   }




//          D=0.107;//between 0.09  to  0.11optimise
//          L=0.0784;//L=DsL/(D*D);// between 0.07  to  0.1
            polepitch[i]=3.14*D[nt][i]/pole;
            Ltowr[i]=L[nt][i]/polepitch[i];
//          cout<<Ltowr[i]<<"\n";//between 0.5 to 1.2constraints if not change d and L
            Li[i]=0.95*L[nt][i];
            yss[i]=3.14*D[nt][i]/Sslot;
```

```
    fout<<"FLUX PER POLE :   "<<fluxperpole[i]<<"\n";
    //CHECK FLUX DENSITY IN CORE
    Do[i]=D[nt][i]+0.072;
    dsc[i]=20;//min17 max 26
    dss[i]=((Do[i]-D[nt][i])*1000-(2*dsc[i]))/2;
    //optimise
    Ws[i]=(3.14*(D[nt][i]*1000+dss[i])/Sslot)-Wts;
    fout<<"DEPTH TO WIDTH OF STATOR SLOT RATIO :"<<dss[i]/Ws[i]<<"\n";//this min 2.5  to  4.5 according to select Wts
    Bcs[i]=fluxperpole[i]/(2*Li[i]*dsc[i]/1000);
    fout<<"STATOR CORE FLUX DENSITY:  "<<Bcs[i]<<"\n";//thid min0.8  max 1,5 according to select dsc
//STATOR WINDING
    Kwm=0.8;
    E=0.95*V;
    //Tm=450;
    Tm[i]=E/(4.44*Kwm*freq*fluxperpole[i]);//Calculate flux per pole and Bcs if not withinlimit change Tm
    Tpm[i]=abs(Tm[i]/4);
    fout<<"Turns in series per pole : Tpm  "<<Tpm[i]<<"\n";
    fout<<"turns in each coil  : \n";
    c46[i]=0.121*Tpm[i];c37[i]=0.227*Tpm[i];c28[i]=0.306*Tpm[i];c19[i]=0.346*Tpm[i];
    Tm[i]=Tpm[i]*4;
    fout<<"TOTAL TURNS IN SERIES FOR MAIN WINDING:   "<<Tm[i]<<"\n";
    //LENGTH OF MEAN TURN
```

```
        Lmtm[i]=(c46[i]*Lml46[i]+c37[i]*Lml37[i]+c28[i]*Lml28[i]+c19[i]*Lml19[i])/
Tpm[i];
        fout<<" LENGTH OF MEAN TURN: Lmtm"<<Lmtm[i]<<"\n";
        //conductor size
        I=P/(V*FLE*PF);
        //Isd[i]=5;// this value between 3 to 5;
        am[i]=I/Isd[nt][i];
        Am[i]=2*Tm[i]*am[i];
        fout<<" total area of conductors in main winding: Am"<<Am[i]<<"\n";
        //ROTOR DESIGN
//      lg[i]=0.32;//optimise length of airgap between 0.32  to 0.7
        fout<<"LENGTH OF AIRGAP lg   "<<lg[nt][i]<<"\n";
        Dr[i]=D[nt][i]-(2*lg[nt][i])/1000;
        fout<<"ROTOR DIAMETER:   "<<Dr[i]<<"\n";
        Rslot=44;
        Btr=1.3;//this value higher than Bts
        Lb=0.0756;
//      ab[i]=17;//lies between 17 to 25
        wb=ab[nt][i]/Lb;
        Ar[i]=Rslot*ab[nt][i];
        Aratio[i]=Ar[i]/Am[i];//this lies between 1 to 2.4 else change ab
        fout<<"RATIO OF ROTOR BAR AREA /MAIN WINDING CONDUCTOR
AREA  Aratio"<<Aratio[i]<<"\n";
        //END RING CALCULATION
        ae[i]=0.32*Ar[i]/pole;
        //RESISTANCE OF MAIN WINDING
```

```
    fout<<" RESISTANCE OF ROTOR REFERED TO AMIN  rrm
"<<rrm[i]<<"\n";
    //REACTANCES
    c3[i]=(1.875+(Sslot*1.78/Rslot));
    xs[i]=16*3.14*freq*Tm[i]*Tm[i]*0.8*0.8*L[nt][i]*4*3.14*0.96*c3[i]/(36*10000
*1000);
    fout<<"SLOT LEAKAGE REACTANCE   xs  "<<xs[i]<<"\n";
    //ZIGZAG LEAKAGE REACTANCE
    Wtr=Wts-1;
    ysr[i]=3.14*Dr[i]/Rslot;//check
    C4[i]=Wts*Wtr*(pow((Wts),2)+pow((Wtr),2))/(12*lg[nt][i]*pow((yss[i]*1000),2
)*(ysr[i]*1000));
    xz[i]=16*3.14*freq*pow(Tm[i],2)*0.8*0.8*L[nt][i]*4*3.14*C4[i]/(36*10000*10
00);
    fout<<"ZIGZAG LEAKAGE REACTANCE :xz   "<<xz[i]<<"\n";
    //OVERHANG LEAKAGE REACTANCE
        xo[i]=16*3.14*freq*pow(Tm[i],2)*0.8*0.8*4*3.14*(
(3.14*(D[nt][i]+(dss[i]/1000))*5)  )/(10000*1000*6.4*Sslot*pole);
        fout<<"OVERHANG LEAKAGE REACTANCE   xo"<<xo[i]<<"\n";
    //magnetising reactance
    Xm[i]=16*3.14*freq*pow(Tm[i],2)*0.8*0.8*4*3.14*L[nt][i]*polepitch[i]/(10*(lg
[nt][i]/1000)*1.28*pole*1.15*10000*1000);
fout<<"MAGNETISING REACTANCES   Xm"<<Xm[i]<<"\n";
    //skew leakge reactance
    xsk[i]=Xm[i]*pow(0.35,2)*0.95/12;
    fout<<" SKEW LEAKAGE REACTANCE   xsk  "<<xsk[i]<<"\n";
```

```
Kl[i]=sqrt( (Xom[i]-Xlm[i])/Xom[i] );
fout<<" LEAKAGE FACTOR:  Kl  "<<Kl[i]<<"\n";
Cl=0.15*P;
FWl=0.04*P;
Tl=Cl+FWl;
fout<<" TOTAL LOSSES:  Tl  "<<Tl<<"\n";
//PERFORMANCE CALCULATION
rrm0[i]=0.021*Tm[i]*Lmtm[i]/ab[nt][i];
Im[i]=V/Xom[i];
F1[i]=(2-pow(Kl[i],2))*rrm[i];
F2[i]=(2*rsm[i]+rrm[i])*rrm[i]/Xom[i];
F3[i]=Im[i]*rrm0[i]*rrm[i]/Xlm[i];
F4[i]=2*Im[i]*rrm[i];
F5[i]=Im[i]*rrm[i]*Kl[i];
F6[i]=pow( (Im[i]*rrm[i]*Kl[i]),2)*rrm[i];
F7[i]=V*Kl[i];
F8[i]=pow( (V*Kl[i]),2)*rrm[i];
F9[i]=Cl/(2*V);
fout<<rrm[i]/Xlm[i]<<"\n";
H1[i]=0.984;
H2[i]=pow(H1[i],2);
H3[i]=1-H2[i];
H4[i]=H3[i]*rrm[i];
H5[i]=F1[i];
H6[i]=H4[i]+H5[i];//U
H7[i]=H3[i]*Xlm[i];
```

```
H13[i]=H11[i]-H12[i];
H14[i]=F9[i]*H6[i];
double N[POP+1];
H15[i]=H13[i]+H14[i];
N[i]=H15[i];
H16[i]=sqrt( pow(N[i],2)+pow(F4[i],2)  );
double I1[POP+1];
H17[i]=H16[i]/H10[i];
I1[i]=H17[i];
H18[i]=H3[i]*F7[i];
H19[i]=sqrt( pow(N[i],2)+pow(F4[i],2)  );
double I2[POP+1];
H20[i]=H19[i]/H10[i];
I2[i]=H20[i];
H21[i]=H1[i]*F5[i];
H22[i]=H21[i]/H10[i];
double I3[POP+1];
I3[i]=H22[i];
H23[i]=H3[i]*F8[i];
H24[i]=F6[i];
H25[i]=H23[i]-H24[i];
H26[i]=I1[i]*I1[i]*rsm[i];
double pmyculoss[POP+1];
pmyculoss[i]=H26[i];
H27[i]=I2[i]*I2[i]*rrm[i];
double secculossm[POP+1];
```

```
        Cl=H29[i];
        H30[i]=H25[i]*H2[i]/(H10[i]*H10[i]);
        H31[i]=H26[i]+H27[i]+H28[i]+H29[i]+H30[i];
        H32[i]=H29[i]+FWl;
        H33[i]=H30[i]-H32[i];
        H34[i]=H1[i]*SN;
        H35[i]=(60/(2*3.14))*(H33[i]/H34[i]);
        H36[i]=H33[i]/H31[i];
        eff[i]=H36[i];
        H37[i]=H31[i]/(E*I1[i]);
        powf[i]=H37[i];
        //cout<<eff[i]<<"  "<<powf[i]<<"  "<<H35[i]<<"\n";
        fout<<eff[i]<<"PF  "<<powf[i]<<"  "<<H35[i]<<"\n";
        //cout<<I3<<"  "<<rrm<<"  "<<H28;

}

for( j=1;j<=NX;j++)
 {
   fout<<"\t"<<x[cnt][j];
        {
                if(j==1)
                D[nt][j]=x[cnt][j];
                if(j==2)
                        L[nt][j]=x[cnt][j];
                if(j==3)
                        Bts[nt][j]=x[cnt][j];
```

```
                    ab[nt][j]=x[cnt][j];
                }

  fout<<endl;
 }
    fout<<"best"<<bestk[nt];

}
for( i=1;i<=noit;i++)
{
    fout<<noit<<"\t";
    fout<<endl<<"ITER BEST["<<i<<"]   "<<bestk[i]<<"\n";
//for( j=1;j<=NX;j++)
    {
        fout<<D[i][1]<<" "<<L[i][2]<<"  "<<Bts[i][3]<<"  "<<lsd[i][4]<<"
"<<lg[i][5]<<"  "<<ab[i][6]<<"\n";
    }
 //  fout<<"\t"<<PGs[i][j];
}
  bestcnt=1;
for( i=1,j=i+1;j<noit;j++)
     {
      if(bestk[i]<bestk[j])
        {
        bestk[i] = bestk[j];
         bestcnt =j;
```

```
            fout<<"D: "<<D[bestcnt][1]<<"\n"<<"L: "<<L[bestcnt][2]<<"\n"<<" Bts:
"<<Bts[bestcnt][3]<<"\n"<<"Isd:
"<<Isd[bestcnt][4]<<"\n"<<"lg:"<<lg[bestcnt][5]<<"\n"<<"ab: "<<ab[bestcnt][6]<<"\n";
             fout<<"LTOWR :  "<<bltowr[bestcnt]<<"\n";
             fout<<"Bcs  "<<bbcs[bestcnt]<<"\n";
             fout<<"Aratio  "<<bAratio[bestcnt]<<"\n";
            cout<<"\n\n\n  SUPER BEST  "<<"OPTIMAL EFFICIENCY:
"<<bestk[bestcnt]<<"\n  "<<"POWERFACTOR:  "<<bpf[bestcnt]<<"\n";
            cout<<"D: "<<D[bestcnt][1]<<"\n"<<"L: "<<L[bestcnt][2]<<"\n"<<" Bts:
"<<Bts[bestcnt][3]<<"\n"<<"Isd:
"<<Isd[bestcnt][4]<<"\n"<<"lg:"<<lg[bestcnt][5]<<"\n"<<"ab: "<<ab[bestcnt][6]<<"\n";
cout<<"LTOWR :  "<<bltowr[bestcnt]<<"\n";
 cout<<"Bcs  "<<bbcs[bestcnt]<<"\n";
 cout<<"Aratio  "<<bAratio[bestcnt]<<"\n";
}
```