INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

NEURAL NETWORK APPLICATIONS IN THE CONTROL OF POWER ELECTRONIC CONVERTERS

Allan Insleay

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the degree of Master of Applied Science at

Concordia University

Montreal, Quebec, Canada

March 1997

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre reference

Our file Notre référence

0-612-26003-8

Canada

ABSTRACT

NEURAL NETWORK APPLICATIONS IN THE CONTROL OF POWER ELECTRONIC CONVERTERS

Allan Insleay

Attempts have recently been made to apply Neural Networks to control systems where they are to deal with any modeling uncertainties that may exist. This thesis proposes the Neural Network controller as a viable alternative to the conventional and widely used PI regulator for the regulation of Power Electronic converters. Neural Networks may be used to both control of and identification in a system In general, one assumes that the mapping performed by the Neural Network can adequately represent the system's behavior over the desired operating range. PI regulators being designed for a specific load or operating point, cannot compensate for any significant change in the system parameters. This thesis presents a few applications of Neural Network control to power converters. It shows its feasibility as a current control element in dc to dc buck converters. Furthermore, the operation of an on-line Neural Network controller to waveshape the input line currents and force unity power factor operation in a voltage controlled PWM rectifier is demonstrated. Finally, for a three phase current source PWM rectifier a Neural Network controller is used to waveshape the input line currents and maintain unity power factor operation. For all three applications, this thesis presents theoretical foundations of the use of Neural Network controllers and the design considerations and guidelines for the power and control circuits. Simulation results confirm the viability of the proposed Neural Network controller and demonstrate very good performance.

ACKNOWLEDGMENTS

Dr. P. D. Ziogas had a major impact in the early part of this study, although now deceased his support and guidance will always be remembered.

Special thanks are to Dr. G. Joos for his invaluable advice regarding Power Electronics and its control aspects, and his continual guidance and support during the course of this study.

Thanks to Donato Vincenti for the valued discussions regarding dynamic programming, his friendship and encouragement throughout the study.

I would like to thank all my friends and research colleagues in the Power Electronics Lab for their help.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

NN	neural network
BPN	backpropagation
PI	proportional integral
PWM	pulse width modulation
SPWM	sinusoidal pulse width modulation
CSI	current source inverter
VSI	voltage source inverter
IDF	input displacement factor
THD	total harmonic distortion
PF	power factor
M	modulation index
rms	root mean square
SW	switch
kVA	kilo volt ampere
uF	micro Farad
PC	personal computer
EPOCH	the presentation of an entire data set to the NN
LMS	least mean square error

LIST OF PRINCIPLE SYMBOLS

W_h	hidden layer weights
W_o	output layer weights
ΔW	change in weights
B_h	hidden layer bias terms
B_o	output layer bias terms
neth	combination of terms at the hidden layer node
neto	combination of terms at the output layer node
F_h	sigmoid function operator on hidden layer
F_o	sigmoid function operator on output layer
n	number of neurons in NN
θ	bias term
θ_h	bias term for the hidden layer
θ_o	bias term for the output layer
S	slope constant
V_o	output voltage
e	error term
E	global error term
ΔS	change in slope
X_i	input to hidden layer
O	output of NN at the output layer after sigmoid function
Y	output of NN at the hidden layer after sigmoid function

η	learning rate
$\Delta\eta$	change in learning rate
δ_h	error signal at the hidden layer
δ_o	error signal at the output layer
β	beta
ρ	rho
L	inductance
L_{dc}	dc side inductance
C	capacitance
C_{dc}	dc side capacitance
R_f	damping resistance
R	load resistance
Ω	measure of resistance in ohms
$V_{dc,ref}$	dc reference voltage
V_{ref}	command reference voltage
V_a	input ac phase voltage
V_{dc}	output dc voltage
V_{cdc}	dc capacitor voltage
V_{cl}	filter capacitor voltage
V_{out}	output voltage
V_{cc}	supply voltage
I	input to the output layer

I_{ll}	input line current
I_{ldc}	dc inductor current
I_{ref}	reference current
t	time constant in PI controller
k	proportional gain in PI controller
κ	term used in the adaptive learning rate algorithm
φ	term used in the adaptive learning rate algorithm

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Power Converter Control Aspects

Power electronic systems are principally concerned with controlling energy. Energy is transformed from that which is supplied from a source to that required by the load. Power converters may be as simple as a buck regulator or as complex as a controlled rectifier. Two typical power converter systems are shown in Figs. 1.1 and 1.2. No matter how complex the converter, the aim of the controller remains the same, to control a specified output or input quantity, usually by means of feedback. The controller's goal is therefore to sample a portion of the output or input, compare it with a reference and generate a control signal.

The numerous converter topologies, in order to operate at specific operating points, must have some type of controller or regulator. The function of a regulator is to ensure the converter operation is stable. The general regulator control scheme is shown in Fig. 1.3.

Traditional control methods, such as the PI controller, enjoy widespread use in the regulation of dynamical systems due to their ease of design and simplicity. For some systems however only by poor or inaccurate models are available. A regulator which includes the ability to sense its environment, process the acquired information thus reducing uncertainty, plan, generate and execute a given type of control action under normal or extreme situations defines an intelligent control system. Intelligent schemes

have to deal with system complexity, non-linearities within the system as well as uncertainties and still provide for stable operation. Learning controllers can incorporate different adaptive algorithms, reference models or performance criteria. In addition, intelligent controllers can be made to remember the optimum control parameters corresponding to the various operating points of the system

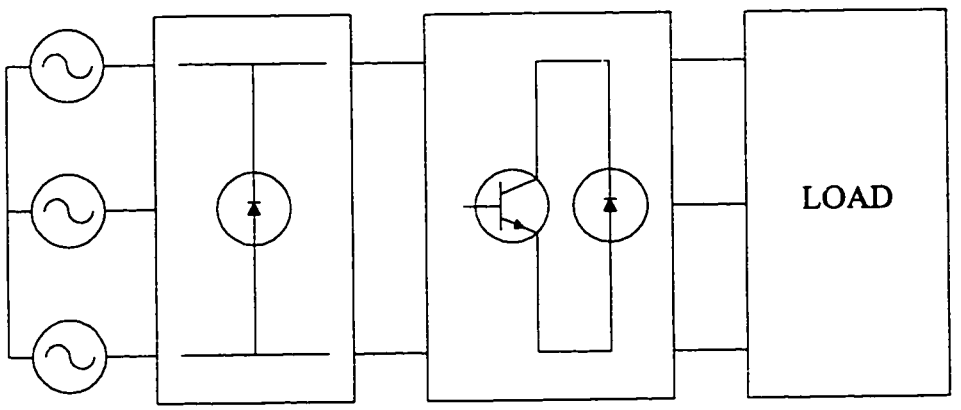


Fig. 1.1 General ac to ac power converter topology .

Control problems can be divided into two classes 1) regulation and tracking problems, in which the objective is to follow a reference trajectory and 2) optimal control problems, in which the objective is to find a function of the controlled system's behavior that is not necessarily defined in terms of a reference trajectory. Reinforcement learning is based on the common sense idea that if an action is followed by a desired reaction or an improved reaction then the tendency to produce that action is strengthened. Thus by combining methods for estimating the long term consequences of actions, reinforcement learning methods can be devised that are applicable to control problems.

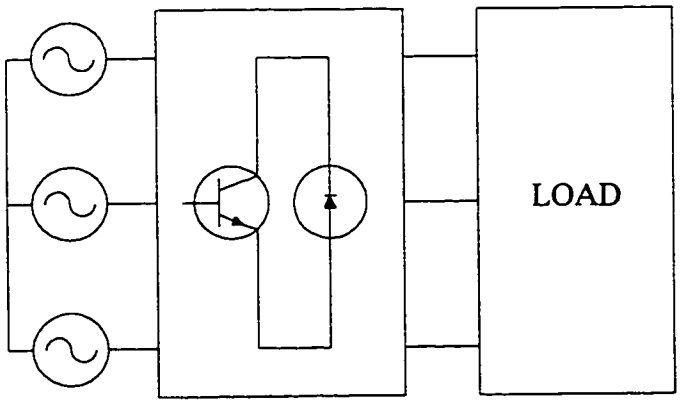


Fig. 1.2 General direct ac to dc power converter topology .

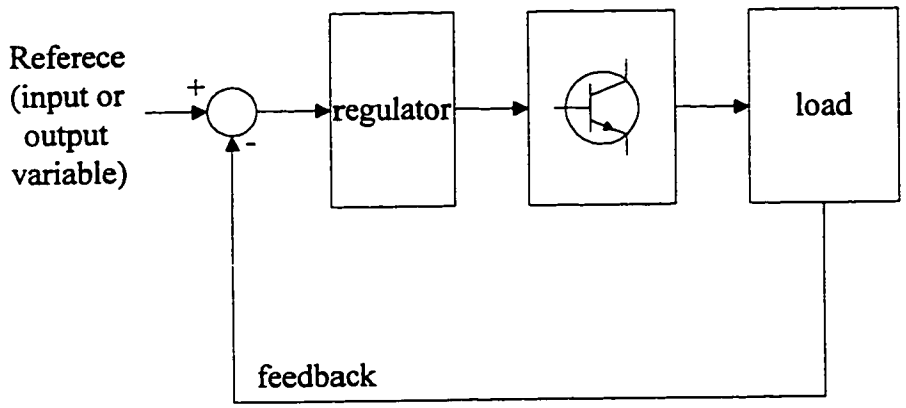


Fig. 1.3 General regulator controller topology .

The NN controller, using the Backpropagation (BPN) algorithm, uses reinforcement type learning to self tune its parameters to allow it to follow a specified trajectory.

Since there exists substantial prior information about the various subsystems of the control system in many practical control problem. In modeling these subsystems via NNs, it is desirable to incorporate this prior knowledge into a NN. Thus the NN may be trained to accurately represent a plant.

1.2 Intelligent Control Schemes

Intelligent control schemes are intended to maintain closed loop performance over a wide range of operating circumstances, taking into account the complexity of both the plant and the performance objectives, and in the presence of uncertainty. These complications may arise from non linear or time varying behavior, poorly modeled plant dynamics, imperfect measurements or other abnormal operating conditions. Each of these effects must be addressed if the system is to operate reliably in an autonomous fashion.

Intelligent control, using NNs for example can be applied to complex dynamical systems. Many attempts have been made to apply NN to the control field where they may be used to deal with non-linearities and any uncertainties that may arise within the plant (system) dynamics [6,7,8,9,10]. In [6], an on-line NN is used to control the tracking in an industrial drive. The controller consists of four units, a preprocessor, a classifier, a look-up table, and a servo drive unit. Here measurements of output and input values once classified by the NN are used to generate the appropriate signals for the proper control of the drive. A NN consisting of time delay elements and a fuzzy logic learning method was employed as a robotics motion controller in [8]. This technique boasted increased learning speed and improved convergence when dealing with non-linear dynamic

systems. Another approach uses the NN as an emulator that identifies the system's dynamics and one as a controller that tracks the dynamical process [3]. The parameters of both the emulator and the controller were determined via the inherent learning properties of the NN. A NN controller was used in [9] to control the operation of an inverter. An on-line training technique using sinusoidal currents as references forces the output current of the inverter to track these references. It was found that the NN controller exhibited improved characteristics when compared with conventional control methods. In [26] An adaptive feedforward control system, including a NN emulator, was designed for a PWM boost converter. The NN emulator was used to identify the converter parameters and characteristics.

Current source type PWM rectifiers are used as the front end ac/dc converter in power electronic systems. The load can be a CSI driven induction motor requiring regulated dc current or a load operating on regulated dc voltage. Direct interfacing with the ac mains imposes stringent specifications on the rectifier such as (a) low input current harmonics and (b) high input power factor. In standard schemes, the current source PWM rectifier is operated with off line patterns which result in slow transient response, with discontinuous control of modulation index [11,12]. Also, in order to avoid current oscillations during starting and transients, a sufficient amount of damping resistance must be provided in the input filter circuit. This reduces the overall system's efficiency and filter effectiveness [13]-[15]. These schemes control the input current oscillations by inserting damping resistors or with complicated feedback loops. As a result the stability region may be limited and the system requires a precise design of the control

components. NN technology has the ability to improve the control of power electronic systems [6,9,15,16,17].

1.3 Scope and Contributions

This thesis proposes the software implementation of a NN controller as a viable alternative to traditional control schemes particularly PI type controllers. The NN controller while demonstrating its self tuning capabilities as well as its ability to adjust to changes in system parameters, has been incorporated in the control of the following power conversion systems:

(a) A dc to dc buck regulator, where the function of the NN is to adapt to changing load conditions, where the PI controller is the traditional choice.

(b) Voltage controlled PWM rectifiers, where the function of the NN is to waveshape the input line currents, force unity power factor operation and damp the low frequency operation of the input filter,

(c) Current controlled PWM rectifiers, where the purpose of the NN is to obtain unity power factor operation, effectively damp the low frequency resonance of the input filter, and inject minimal levels of harmonic current into the ac mains.

Specifically, two types of NN controllers will be considered: (a) the off line trained and (b) the on line NN regulator.

1.4 Summary of Thesis

The contents of the thesis have been organized in the following manner. Chapter 2 provides an introduction to the concept of NNs. Also the basics of Backpropagation is discussed with emphasis on learning algorithms and weight updating techniques.

Chapter 3 describes the application of NN regulators to the control of dc to dc buck converters. Here, the purpose is to demonstrate the advantages of the NN controller compared to the traditional PI controller.

Chapter 4 details the implementation of a NN controller for voltage controlled PWM rectifiers. Here, an on line NN controller waveshapes the input line currents forcing unity power factor operation and damping the low frequency resonance of the input filter. Sensitivity to load and parameter variations, are investigated. System performance is verified through computer simulation.

In Chapter 5, the NN controller is applied to current controlled PWM rectifiers. Here the task of the NN controller is to waveshape the input line currents while attempting to achieve unity power factor operation.

Finally, the summary and conclusions of the thesis are presented.

The appendices contain information about the software tools and methodologies and a proposed practical implementation discussing the use of a digital signal processing station as the controlling tool. Block diagrams of the proposed system topology are included.

CHAPTER 2

NEURAL NETWORK TOPOLOGY AND PRINCIPLES OF OPERATION

2.1 INTRODUCTION TO NEURAL NETWORKS

This chapter presents a general overview of Neural Networks, the BPN technique and some of its variations. The present field of Neural Networks links a number of closely related areas, such as parallel processing, connectionnism and neural computing, these being joined with the common goal of attempting a self learning computing system [21]. Common to all neural network technologies is the neuron. Neurons are the processing elements of neural networks. These simple elements are connected by a variety of topological classes, trained by yet another class of learning algorithm.

2.2 BACKPROPAGATION PRINCIPLES OF OPERATION

The BPN neural network belongs to the class of feedforward networks. This implies that information flows in one direction only - from input to output. The multilayered feedforward neural network can learn a mapping of any complexity. The network's learning of a particular pattern is based on repeated presentation of the data set. This type of neural network has a propagate - adjust cycle allowing the neural network to learn an entire data set. The data set is presented to the inputs of the neural network and the information travels through the hidden layer to the output layer. This action constitutes the propagate phase. The adjust stage entails the comparison of these outputs to desired outputs, and the error information is used to modify the weights of the neural

network. This error BPN is the basis for the training algorithm used to train a multilayered feedforward network.

2.2.1 BACKPROPAGATION STRUCTURE AND COMPONENTS

The BPN is one of the many existing neural network topologies. It is composed of a number of nodes at which point computation takes place. This node or neuron is the basic building block upon which the neural network structure is built. It is intended to simulate a biological neuron.

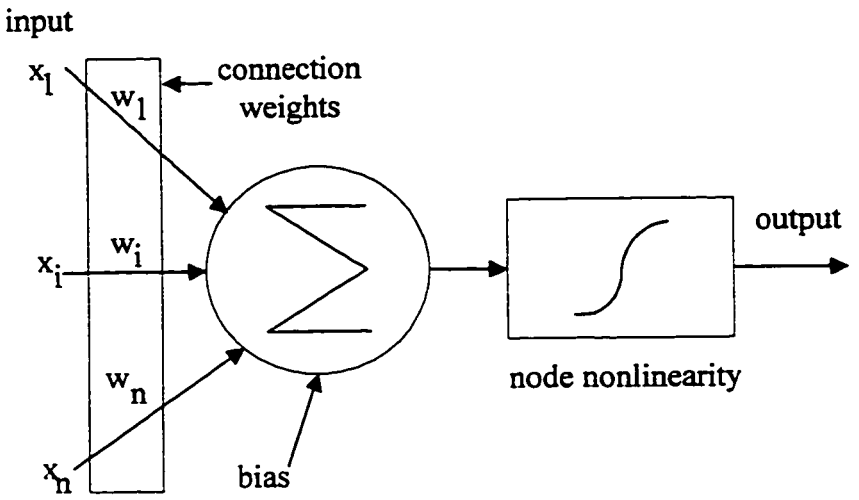


Fig. 2.1 Neuron model.

The neuron model of the BPN shown in Fig. 2.1 has multiple inputs and one output. Each input flows through a connection weight w_n . The output is a function of the input, the weights, a bias term and a squashing function (node non-linearity) [1].

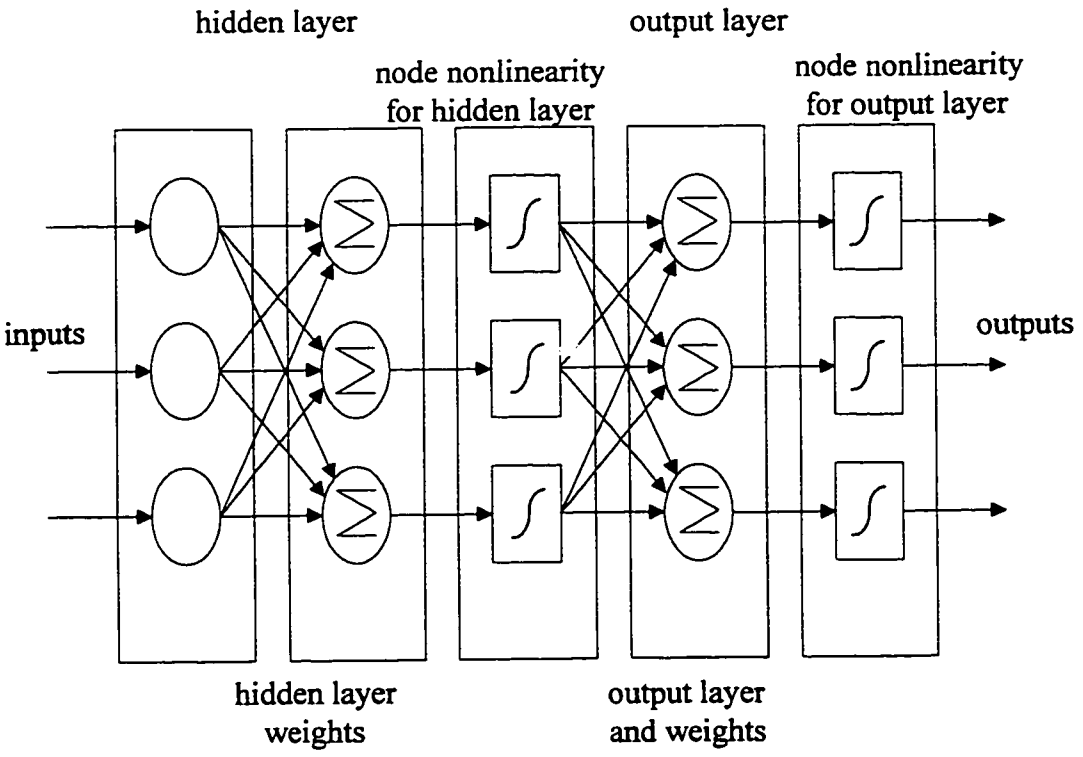


Fig. 2.2 General BPN NN structure.

The basic backpropagation structure is composed of layers of these interconnected neurons as shown in Fig. 2.2. The network is made up of three distinct layers. The input layer serves as an interface with the connecting system. It directly feeds the hidden layer via a network of connection weights. Since the hidden layer is built from the neuron model it takes the sum of all the inputs and a bias. This sum known as neth is then passed through a non linear function. This result flows through the interconnecting weights to the output layer where the summing process starts all over again. The error terms between the desired output and the network output is calculated and used to adjust the output weights and then propagated back to contribute to the process for all the hidden

layer weights. The net continues in this propagate-adjust manner until all patterns are learned.

2.2.2 BACKPROPAGATION GOVERNING EQUATIONS

The equations governing the operation of the BPN shown in Fig. 2.2 will now be presented. The input to the hidden layer is

$$neth_{pj} = \sum_{i=1}^{N} \left(wh_{ji} * x_{pi} \right) + \theta h_j \tag{2.1}$$

The weight connections between the input and the hidden layer are denoted as wh_{ij} and θh_{ji} is a bias input. The use of the bias input is optional. These are then processed by a squashing function

$$Y_{pj} = Fh_j(neth_{pj}) \tag{2.2}$$

The outputs of these hidden nodes become the inputs to the output layer. Thus

$$neto_{pk} = \sum_{j=1}^{L} \left(wo_{kj} * Y_{pj} \right) + \theta o_k \tag{2.3}$$

where the connections between the hidden layer and the output layer are denoted as wo_{kj} and θo_k is the bias input. These outputs must also be treated by a squashing function. This results in an output of

$$O_{pk} = Fo_k(neto_{pk}) \tag{2.4}$$

The next step will be to describe the delta update rule. The backpropagation algorithm performs a steepest descent minimization on a surface in weight space whose height at any point is equal to the error.

The error between the actual net output and the target is defined as

$$E_k = \left(d_k - O_k\right) \tag{2.5}$$

In order for the network to learn this error must be minimized and used in some way to update the weights in the output layer as well as those associated with the hidden layer. The following error is defined

$$E_p = \frac{1}{2} * \left[\sum_k \left(Tp_k - O_{pk}\right)^2\right] \tag{2.6}$$

and

$$\frac{\partial E_p}{\partial wo_{kj}} = -(T_{pk} - O_{pk}) * \frac{\partial Fo_k}{\partial neto_{pk}} * \frac{\partial neto_{pk}}{\partial wo_{kj}} \tag{2.7}$$

$$\frac{\partial neto_{pk}}{\partial wo_{kj}} = \frac{\partial}{\partial wo_{kj}} * \sum_j (wo_{kj} * Y_{pj} + \theta o_k) \tag{2.8}$$

$$\frac{\partial E_p}{\partial wo_{kj}} = -(T_{pk} - O_{pk}) * \frac{\partial Fo_k}{\partial neto_k} * Y_{pj} \tag{2.9}$$

The above equation allows for a weight update rule of the following form

$$wo_{kj}(t+1) = wo_{kj}(t) + \eta * (Tp_k - O_{pk}) * Fo'_k\left(neto_k\right) * Y_j \tag{2.10}$$

letting

$$\delta o_{pk} = (T_{pk} - O_{pk}) * Fo'_k\left(neto_{pk}\right) \tag{2.11}$$

we have for the output layer

$$wo_{kj}(t+1) = wo_{kj}(t) + \eta * \delta o_{pk} * Y_{pj} \tag{2.12}$$

and similarly for the hidden layer

$$\delta h_{pj} = Fh'_j(neth_{pj}) * \Sigma\left(\delta o_{pk} * wh_{kj}\right) \tag{2.13}$$

$$wh_{ji}(t+1) = wh_{ji}(t) + \eta * \delta h_{pj} * X_i \tag{2.14}$$

where the derivative of the squashing function is needed for both the hidden layer as well as the output layer. Here we can see the algorithm's dependency upon the error terms computed for the output layer as well as the hidden layers. The above described equations are a basis for the following study of some modifications on the generalized delta rule backpropagation networks [20].

2.3 BACKPROPAGATION ALGORITHMS

The backpropagation algorithm is a popular technique used to find the optimum weights of multilayer networks. The algorithm learns slowly and tends to converge to a solution slowly as well. The performance of the backpropagation algorithm is sensitive to many initial parameters. Thus it is not possible to guarantee a successful outcome. Some of the parameters that the BPN is sensitive to are the initial weights, the initial biases, the learning rate, momentum, the steepness of the slope of the activation function as well as is type etc. Here, five independent learning algorithms are discussed.

2.3.1 GENERALIZED DELTA RULE USING SIGMOID FUNCTION

The generalized delta rule as previously described is implemented using the sigmoid function as the node non linearity. The equation for the sigmoid is

$$F(x) = \frac{1}{1 + exp(-s * x)} \tag{2.15}$$

where s is the slope steepness constant [21]. The derivative of the sigmoid function is required in terms of the output of the network and is as follows

$$F'(x) = \frac{s * exp(-s * x)}{(1 + exp(-s * x))^2} \tag{2.16}$$

which can now be transformed into a more useful representation in terms of the output of the net as

$$F'(x) = s * F(x) * (1 - F(x)) \tag{2.17}$$

The weight update equations using the delta rule become

$$wo_{kj}(t+1) = wo_{kj}(t) + \sum_k \eta * \delta o_k * i_j \tag{2.18}$$

$$wh_{ji}(t+1) = wh_{ji}(t) + \sum_j \eta * \delta h_j * x_i \tag{2.19}$$

where

o is the output of a node (i.e. hidden layer node or output layer node)

and the deltas are the error signal vectors for a specific layer

$\delta ok = s(target\text{-}output)output(1\text{-}output)$

$\delta hk = sij\ (1 - ij)\Sigma(\delta ok\ wok)$

2.3.2 GENERALIZED DELTA RULE USING TANSIGMOID FUNCTION

The generalized delta rule as previously described is implemented using the tansigmoid function as the node non linearity. The equation for the sigmoid is

$$F(x) = \frac{1 - exp(-s * x)}{1 + exp(-s * x)} \tag{2.20}$$

where s is the slope steepness constant [21]. The derivative of the tansigmoid function is required in terms of the output of the network and is as follows

$$F'(x) = \frac{2 * s * exp(-s * x)}{(1 + exp(-s * x))^2} \tag{2.21}$$

which can now be transformed into a more useful representation in terms of the output of the net as

$$F'(x) = \frac{s}{2} * (1 - F(x)^2) \tag{2.22}$$

The weight update equations using the delta rule become

$$wo_{kj}(t+1) = wo_{kj}(t) + \sum_k \eta * \delta o_k * i_j \tag{2.23}$$

$$wh_{ji}(t+1) = wh_{ji}(t) + \sum_j \eta * \delta h_j * x_i \tag{2.24}$$

where the error terms will have new forms

o is the output of a node (i.e. hidden layer node or output layer node)

$\delta ok = s/2(target-output)(1-output^2)$

$\delta hk = s/2(1-i)\Sigma(\delta ok\ wok)$

2.3.3 VARIABLE SLOPES METHOD USING TANSIGMOID FUNCTION

There exists an optimal value of s (slope constant) for each node thus allowing the algorithm to converge after a short number of iterations. Since it is not possible to calculate these optimal values apriori, they must be determined adaptively. This implies

that the error should be minimized with respect to the slopes. Using the tansigmoid as a node non linearity.

$$F(x,s)=\frac{1-exp(-s*x)}{1+exp(-s*x)} \tag{2.25}$$

where s is the slope steepness constant [21]. The derivative of the tansigmoid function required in terms of the slopes is as follows

$$F_x'(x,s)=\frac{s}{2}*\left(1-F^2(x,s)\right) \tag{2.26}$$

$$F_s'(x,s)=\frac{x}{2}*\left(1-F^2(x,s)\right) \tag{2.27}$$

The gradient of the error will be minimized with respect to the slopes. Where t is the target vector and o is the output of the net

$$E=\frac{1}{2}*\Sigma(t-o)^2 \tag{2.28}$$

$$\frac{\partial(E)}{\partial(s)}=\frac{\partial(E)}{\partial(t)}*\frac{\partial(t)}{\partial(o)}*\frac{\partial(o)}{\partial(s)} \tag{2.29}$$

$$\frac{\partial(o)}{\partial(s)}=F_s'(x,s) \tag{2.30}$$

$$\frac{\partial(t)}{\partial(o)}=\frac{1}{F_x'(x,s)} \tag{2.31}$$

$$\frac{\partial(E)}{\partial(s)}=\frac{\partial(E)}{\partial(t)}*\left(\frac{F_s'(x,s)}{F_x'(x,s)}\right) \tag{2.32}$$

The weight update equations remain unchanged however the slope equations for both the hidden layer and the output layer will get updated using the delta rule as follows

$$\Delta\left(S(t+1)\right) = S(t) + \beta * \left(\Delta\left(S(t)\right)\right) + \rho * \left(S(t) - S(t-1)\right) \tag{2.33}$$

where

$$\Delta(S) = \frac{\partial(E)}{\partial(s)} = -\frac{\partial(E)}{\partial(t)} * \left(\frac{F_s'(x,s)}{F_x'(x,s)}\right) \tag{2.34}$$

if the node to be updated is an output node then

$$\Delta(S) = (t-o) * \left(\frac{F_s(x,s)}{F_x(x,s)}\right) \tag{2.35}$$

if the node to be updated is a hidden node then

$$\Delta(S) = F_S(x,s) \sum \left(\delta_{ok} * w_{okj}\right) \tag{2.36}$$

where the values of beta and rho in equation 2.33 are between 0 and 1. The slope update equations are the only steps added to the generalized delta rule [21]. The computational complexity is increased slightly since now the term Δs must be computed as well as the derivatives with respect to the slopes.

2.3.4 SATURATING LINEAR SOFT LIMITING ALGORITHM

This technique uses a different type of node non linearity. It is defined as follows

$$f(x) = \left\{1 \quad if\, x < \frac{1}{s}\right\} \tag{2.37}$$

$$f(x) = \left\{sx \quad if\, |x| < \frac{1}{s}\right\} \tag{2.38}$$

$$f(x) = \left\{-1 \quad if\, x < \frac{-1}{s}\right\} \tag{2.39}$$

The function has the form shown in Fig. 2.3.

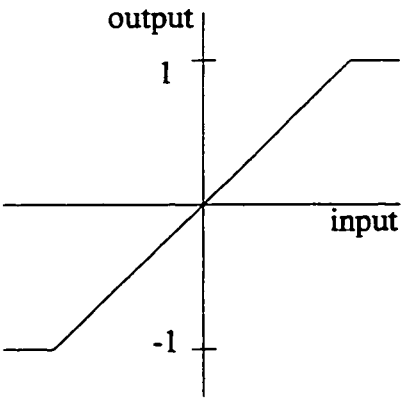


Fig. 2.3 Saturating soft limiter function.

The derivative of the saturating linear soft limiter with respect to the net output is

$$f_x'(x,s)=\{s \quad \text{if } |x|<=1/s \} \tag{2.40}$$

$$f_x'(x,s)=\{\varepsilon \quad \text{if } |x|>1/s \} \tag{2.41}$$

The derivative of the saturating linear soft limiter with respect to the slope is

$$f_S'(x,s)=\{x \quad \text{if } s<=|1/x| \} \tag{2.42}$$

$$f_S'(x,s)=\{\varepsilon \quad \text{if } s>|1/x| \} \tag{2.43}$$

This technique is similar to the variable slope method previously described. Each node will have its own slope associated with it. This allows for the slopes to approach near optimal values, since they will be adapted at each iteration. The slopes are adapted using the delta update rule.

$$\left(S(t+1)\right) = S(t) + \beta * \left(\Delta\left(S(t)\right)\right) + \rho * \left(S(t) - S(t-1)\right) \tag{2.44}$$

where the value of Δs for an output node is

$$\Delta(S) = (t-o) * \left(F_S'(x,s)\right) \tag{2.45}$$

if the node to be updated is a hidden node then

$$\Delta(S) = F'_s(x,s)\sum\left(\delta_{ok} * w_{okj}\right) \tag{2.46}$$

The weights are updated in the following manner

$$\left(w(t+1)\right) = w(t) + \beta * \left(\Delta\left(w(t)\right)\right) + \rho * \left(w(t) - w(t-1)\right) \tag{2.47}$$

if the node to be updated is an output node then

$$\Delta(w) = \left(F_x'(x,s)\right) * (t-o) \tag{2.48}$$

if the node to be updated is a hidden node then

$$\Delta(w) = F'_x(x,s)\sum\left(\delta_{ok} * w_{okj}\right) \tag{2.49}$$

where the values of beta and rho are between 0 and 1. The slope update equations are the only steps added to the generalized delta rule. The computational complexity is increased slightly since now the term Δs must be computed as well as the derivatives with respect to the slopes [2].

2.3.5 ADAPTIVE LEARNING RATE

The information used by the backpropagation algorithm is based on local gradient information. This forces the learning rate to be a small value so that the step size doesn't allow the algorithm to jump to undesirable areas of the weight space. The idea is to have the learning rate be large enough to move quickly across plateaus in the error surface and decrease as a minimum is approached. The delta bar delta algorithm does just this. It makes use of gradient information to increase or decrease the learning rate. When the gradient

$$\delta_{ij} = \frac{\partial (E)}{\partial (w_{ij})} \tag{2.50}$$

has the same sign for many iterations the rate is increased and it is decreased when δ_{ij} flips signs for several steps. We have the following rule.

$$\eta_{ij}(t+1) = \eta_{ij}(t) + \Delta \eta_{ij}(t) \tag{2.51}$$

where

$$\Delta\eta_{ij} = \{\kappa \qquad if\ \delta_{ij}^{-}(t+1) * \delta_{ij}(t) > 0\} \tag{2.52}$$

$$\Delta\eta_{ij} = \{-\phi * \eta_{ij} \qquad \delta_{ij}^{-}(t+1) * \delta_{ij}(t) < 0\} \tag{2.53}$$

$$\Delta\eta_{ij} = \{0 \qquad otherwise\} \tag{2.54}$$

and

$$\delta_{ij}^{-}(t) = (1-\theta) * \delta_{ij}(t) + \theta * \delta_{ij}^{-}(t-1) \tag{2.55}$$

The actual values for κ, ϕ, and θ are quantities specific for each application. They are in general small values between 0 and 1 [3].

2.4 ALGORITHM PERFORMANCE COMPARISON

The structure of the network for the comparison of the different algorithms was kept the same. It consists of three layers, the input layer, the hidden layer, and the output layer. The input layer and the output layer contain 25 neurons each. The hidden layer contains 15 neurons.

The performance comparison took a network used for data compression and changed the learning algorithms. Five different techniques were implemented, each resulted in a different performance.

First, the size of the network's layers affects the system performance, if the input data set is too large, and the number of hidden neurons too small, the network may not converge. For example, in the basic generalized delta rule algorithm, if the number of hidden neurons is six, the network cannot learn more than three different patterns. Here "learn" implies that the network has the ability to store and recall all the patterns shown. Now, in the network used for the performance comparison test, there are 25 input, 25 output and 15 hidden neurons, therefore the network has the ability to learn more patterns as shown in Table 2.1.

Table 2.1

The Number of Patterns Learned by Different Algorithm

ALGORITHM	# OF PATTERNS LEARNED
Generalized Delta Rule	5
Tansigmoid	9
Variable Slopes Tansigmoid	13+
Saturating Linear Soft Limiting	9
Adaptive Learning Rate	9+

From Table 2.1, for the same network size, it is seen that the Variable Slopes Tansigmoid algorithm has more learning capacity than the others. The convergence of the network is sensitive to the initial weights (for all the algorithms). The initial weights are the starting point of the network on the error surface. If the starting point is near or tends to a local minima, then the network may not converge. It could get stuck at this local minima. Therefore, the choice of the initial weights is very important. In this network (for all the algorithms) the initial weights are set randomly to small values between -0.5 and +0.5.

Here, trial by error is applied, as the network converges, the initial weights are used for the next run, thus reducing the convergence time.

The performance of a given algorithm is evaluated based on least mean square (LMS) vs epoch. An epoch is defined as a presentation of the entire data set to the network. The quantity to be minimized during training is the mean squared error over the entire data set given by

$$E_p = \sum_{p=1}^{m} E_p^2 \qquad 2.56$$

This requires that the weights be held fixed during an epoch and only one updated per epoch. The weights are minimized in the direction of minimizing equation 2.6. The weights are updated after each presentation and the error is calculated for the set. The network is said to have converged after this error is within acceptable limits. The simulations, were run using Matlab 4.0 on a Sun Sparc 1px station.

Table 2.2

The Comparison of LMS and Epoch of Different Algorithm with 9 Patterns

ALGORITHM	LMS(goal)	EPOCH
Generalized Delta Rule Tansigmoid	0.3	1400
Variable Slopes Tansigmoid	0.3	320
Saturating Linear Soft Limiting	0.3	575
Adaptive Learning Rate	0.3	75

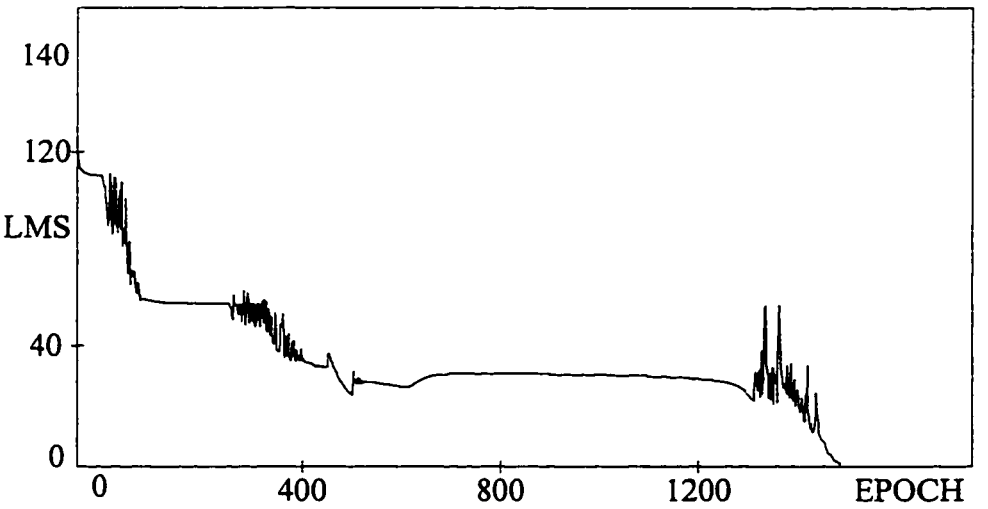


Fig. 2.4 Generalized delta rule tansigmoid learning algorithm.

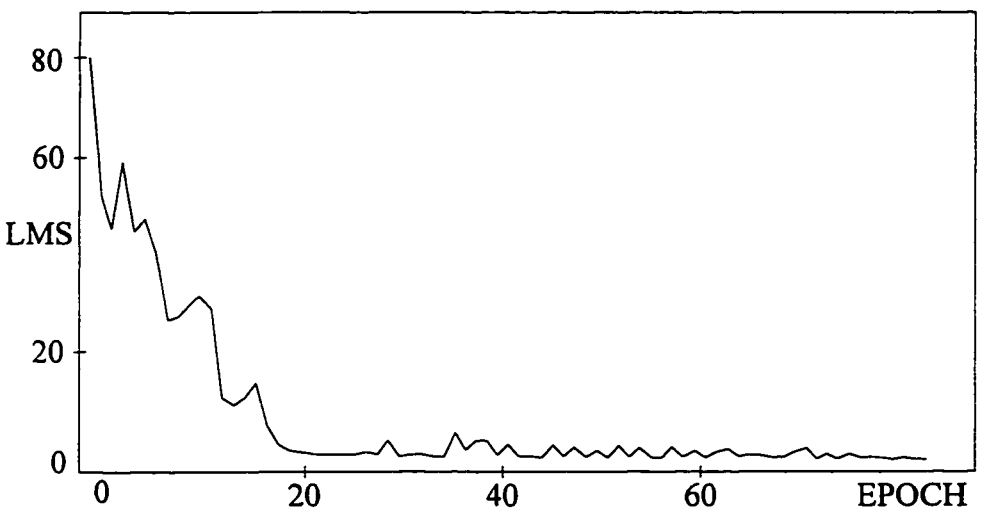


Fig. 2.5 Adaptive learning rate.

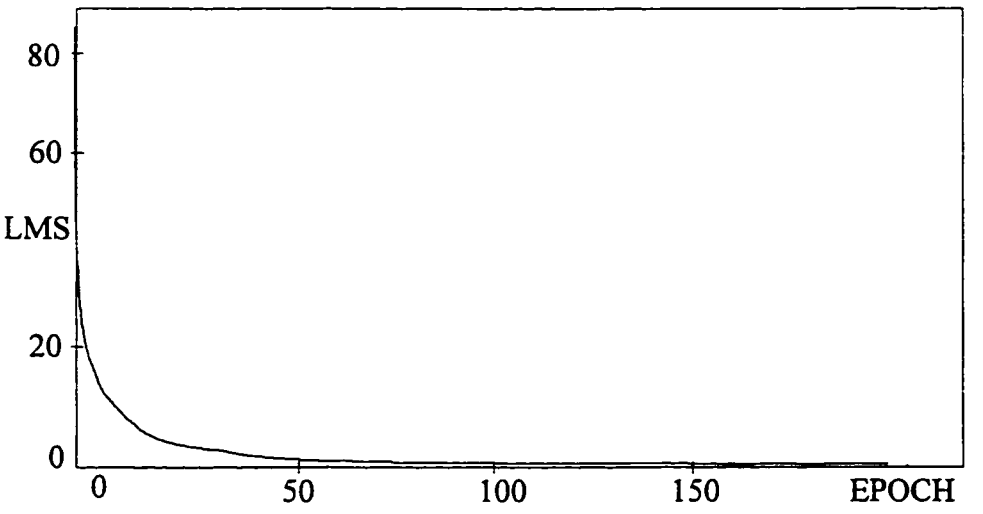


Fig. 2.6 Tansigmoid learning algorithm.

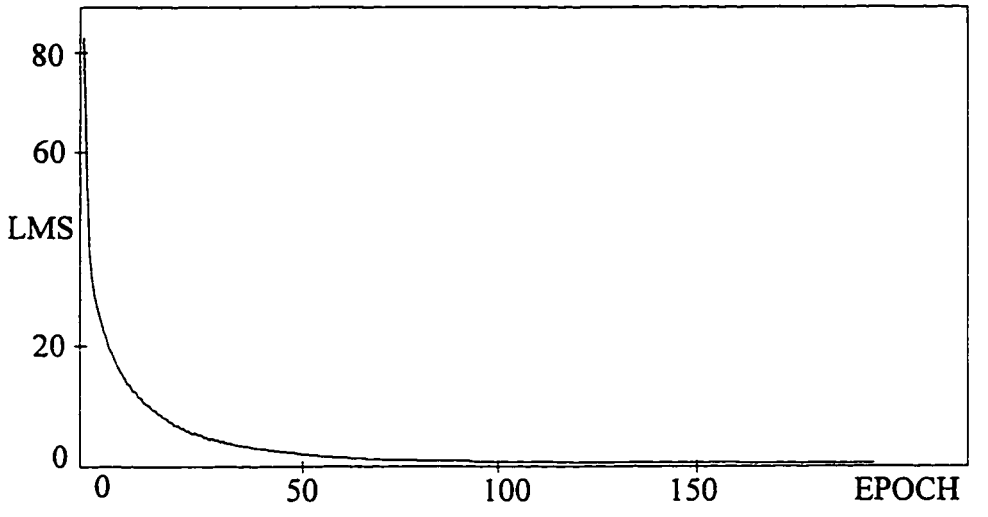


Fig. 2.7 Variable slope tansigmoid learning algorithm.

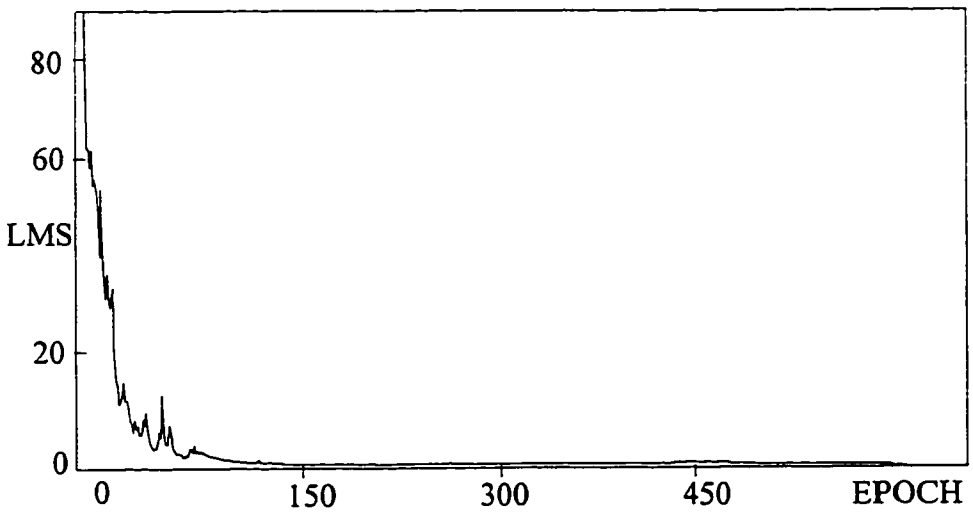


Fig. 2.8 Saturating linear soft limiter learning algorithm.

Table 2.2 shows the error and epoch values for four different algorithms with nine input patterns. It can be seen from the data that the adaptive learning rate algorithm is the fastest to converge at 75 epochs. The adaptive learning rate algorithm starts with a high value of error in the beginning and decreases rapidly Fig. 2.5.

The variable slopes tansigmoid algorithm was the next at 320 epochs Fig. 2.7. This technique has a nice smooth curve indicating minimal bouncing back and forth while searching for the minima. The generalized delta rule tansigmoid and the saturating soft limiting algorithms are the longest to converge at 575 and 1400 epochs respectively, Fig. 2.4 and 2.8. Their curves are similar however the saturating linear soft limiting is a little less stable as opposed to the generalized detla rule tansigmoid and the variable slopes tansigmoid. This may be due to the continuity of the derivative in the tansigmoid case as opposed to the saturating linear soft limiting. The variable slopes tansigmoid converges much faster than the saturating linear soft limiting. They are similar in that both have variable slopes, however we observe a much smoother curve for the tansigmoid

variation. Since for saturating linear soft limiting algorithm the derivative is either a value near zero or a function of the ramp. Whereas in the variable slopes tansigmoid algorithm there is a more gradual change from the slope to the plateau section of the tansigmoid function.

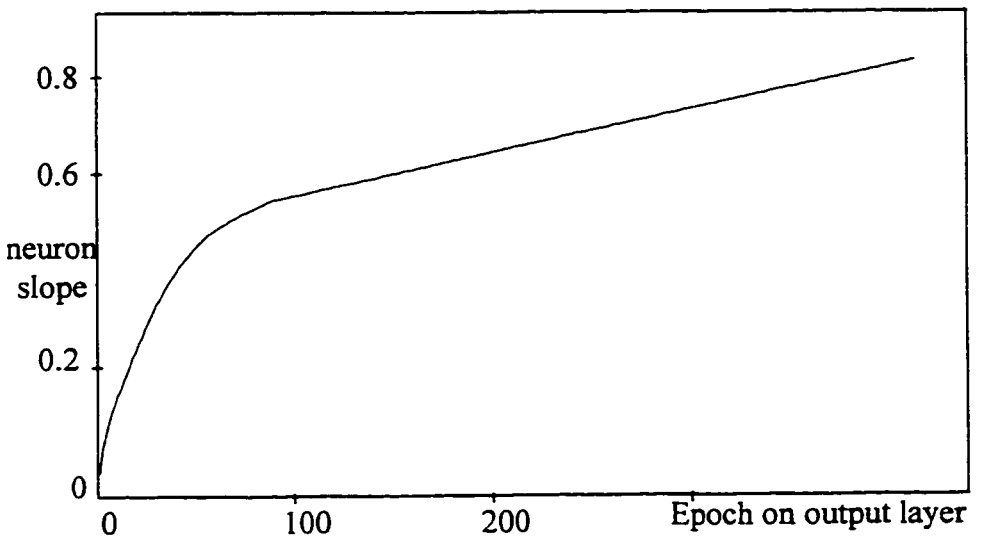


Fig. 2.9 Slope in one neuron vs. epoch on output layer.

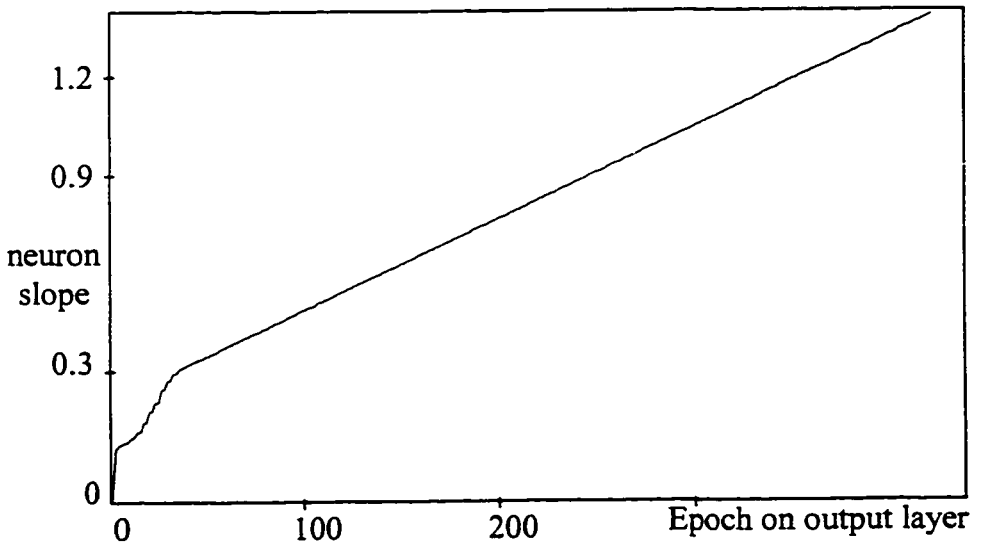


Fig. 2.10 Slope in one neuron vs. epoch on hidden layer.

The variation of the slopes, for the variable slopes tansigmoid algorithm, in the hidden layer is shown in Figs. 2.9 and 2.10. It is seen that as the error begins to decrease, the slopes start from their initial random values and increase as the network approaches it's error goal. This shows that the algorithm adapts the slopes to make the network learn faster. Since as increased slope values tends to make the tansigmoid function appear as a hard limiter, which helps in the convergence process.

2.5 CONCLUSIONS

Differences exist among the different topologies and operation of the NN, but the neurons fundamentally operate in much the same manner. The BPN training algorithm allows experimental acquisition of input / output mapping knowledge within multilayered networks. Once an input pattern has been applied to the nodes on the input layer of a three layer network, the information is processed and propagated to the hidden layer and on through to the output layer to generate an output pattern. This output pattern was compared to the desired output pattern thus generating an error signal. This error signal was then used to generate appropriately scaled error signals at each layer. The weights on the output layer were adapted using the error information. The error signal was modified and then employed to adapt the hidden layer weights. This process of comparison of output and target values continued until all patterns in the data set were learned to within the specified error. The classical algorithm known as the generalized delta rule suffers from the fact that most of the parameters governing it's learning rate, momentum, slope bias etc. are fixed and thus cannot contribute during the learning process. The simulation results presented here show that significant improvements in the backpropagation algorithm are possible. The tested modifications are worth while in that convergence time is reduced through some parameter adaptation (other than weight update) or node non-linearity.

CHAPTER 3

REGULATION OF DC TO DC BUCK CONVERTERS

3.1 INTRODUCTION

This chapter proposes the neural network controller as a viable alternative to the PI controller used in dc to dc converters of the buck type for voltage regulation. The PI controller, although robust and simple, requires a priori knowledge of the system characteristics and once designed for a specific load, its parameters remain fixed. The neural controller, in the on line mode has the ability to learn from experience, thus eliminating the need for a priori knowledge of the system dynamics. The neural network can adapt to variations in the load, and still allow the system to track a specific reference without redesign. Performance comparisons made with the standard PI regulator clearly show the neural network regulator as a viable alternative.

3.2 SYSTEM REQUIREMENTS

The main goal of this section is to show that the NN controller is a viable alternative to the PI controller. The various proposed regulator topologies are depicted in Figs. 3.1 - 3.4. The remainder of this chapter will be organized in the following manner. Section 3.2 will describe details of the implementation of the BPN algorithm. Section 3.3 provides simulation results of the proposed regulator schemes and finally, conclusions are summarized in Section 3.4.

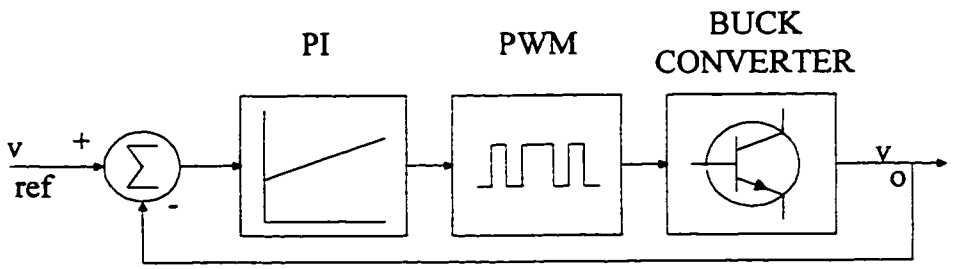


Fig. 3.1 Buck converter with PI controller.

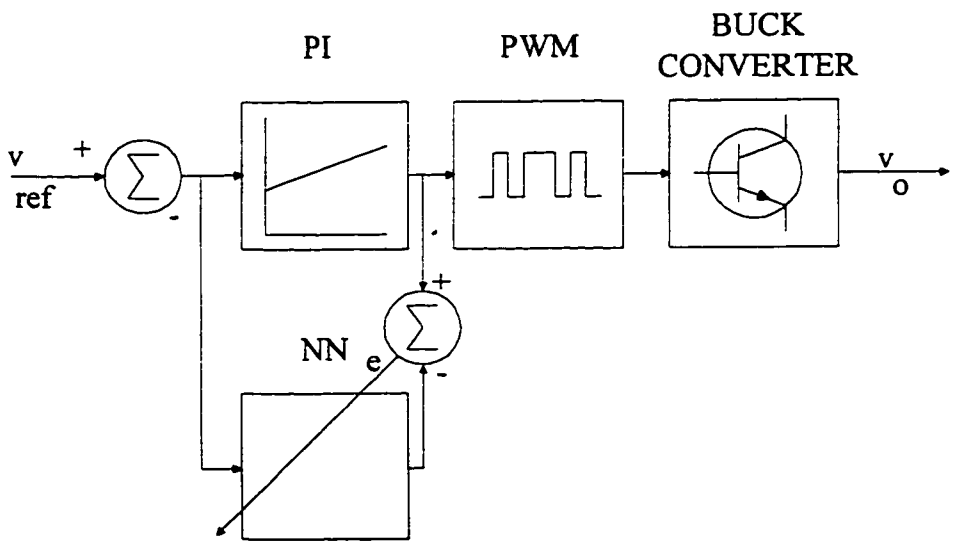


Fig. 3.2 Off line training method.

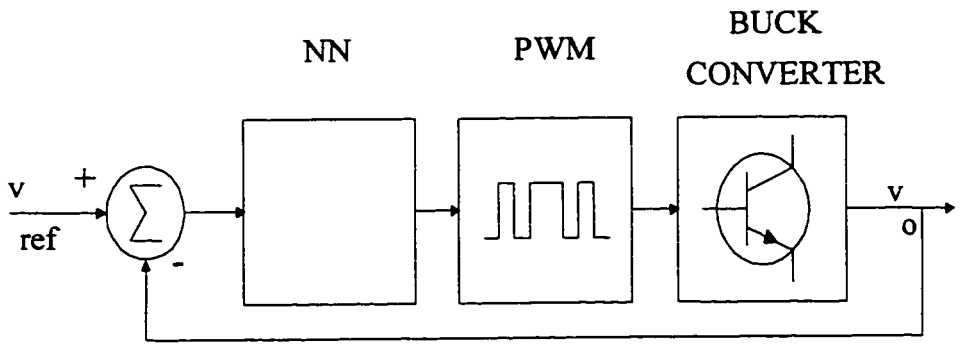


Fig. 3.3 Off line NN in recall mode.

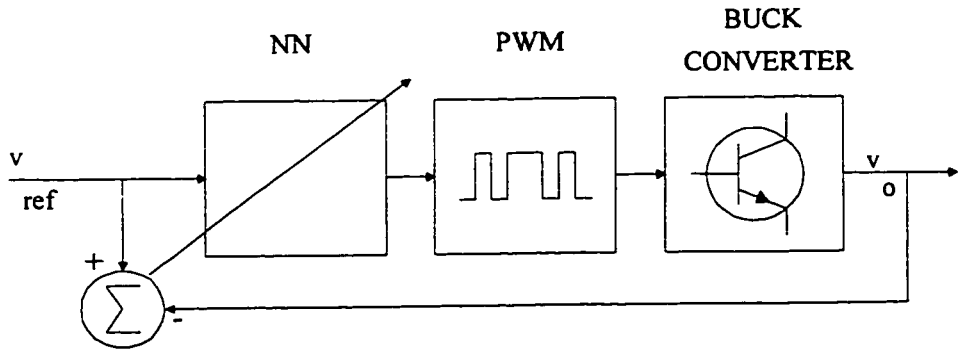


Fig. 3.4 On line NN regulator.

3.3 IMPLEMENTATION

The BPN shown in Fig. 3.2 will perform the least square minimization based on equation (3.1). This performance index will generate the update rules for the weights. The steepest descent gradient search will be applied to (3.1) to compute an update rule for the output layer weights.

$$E = \frac{1}{2} \cdot \left(v_{ref} - v_{out} \right)^2 \tag{3.1}$$

$$\frac{\partial E}{\partial w_o} = -\left(v_{ref} - v_{out} \right) \cdot \frac{\partial v_{out}}{\partial w_o} \tag{3.2}$$

$$\frac{\partial v_{out}}{\partial w_o} = \frac{\partial v_{out}}{\partial O} \cdot \frac{\partial O}{\partial net} \cdot \frac{\partial net}{\partial w_o} \tag{3.3}$$

$$\frac{\partial E}{\partial w} = \left(v_{ref} - v_o \right) \cdot \frac{Es}{v_{carrier}} \cdot F'(net) \cdot i_j \tag{3.4}$$

$$w_{o_j}(t+1) = w_{o_j}(t) + \frac{\partial E}{\partial w_{o_j}} \tag{3.5}$$

$$E = \frac{1}{2} \cdot \left(v_{ref} - O \cdot \frac{Es}{v_{carrrier}} \right)^2 \tag{3.6}$$

$$\frac{\partial O}{\partial w_h} = \frac{\partial O}{\partial neto} \cdot \frac{\partial neto}{\partial i_j} \cdot \frac{\partial i_j}{\partial neth} \cdot \frac{\partial neth}{\partial w_h} \tag{3.7}$$

$$\frac{\partial E}{\partial w_h} = F'(net) \cdot x_j \cdot \sum_j \partial o_j \cdot w_{o_j} \tag{3.8}$$

$$w_{h_j}(t+1) = w_{h_j}(t) + \frac{\partial E}{\partial w_{h_j}} \tag{3.9}$$

These equations show that some knowledge of the system is needed in order to completely define the weight update rules.[5]

3.4 RESULTS

A typical DC to DC buck topology was designed to operate at a frequency of 2 kHz a DC bus of (V_{dc}) 380 V and rated power of 8 kW into a 0.8 Ω load Fig. 3.5. The converter was controlled in turn by each of the three previously discussed regulators. The three controllers were subjected to step changes in the reference voltage shown in Fig. 3.6. It can be observed that the PI controller always exhibits similar overshoot response for each step change in the reference. The off-line trained NN has similar characteristics, whereas the on-line NN controller has minimal overshoot and fast dynamic response. Table 3.1 illustrates the performance characteristics of each of the three regulators.

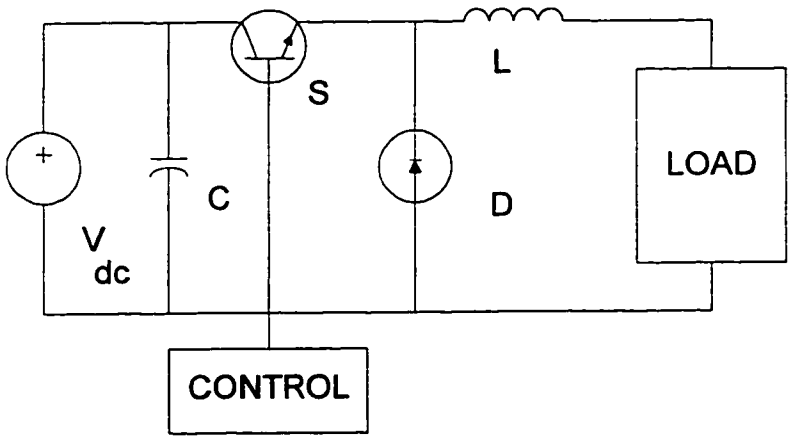


Fig. 3.5 Dc to Dc buck converter system.

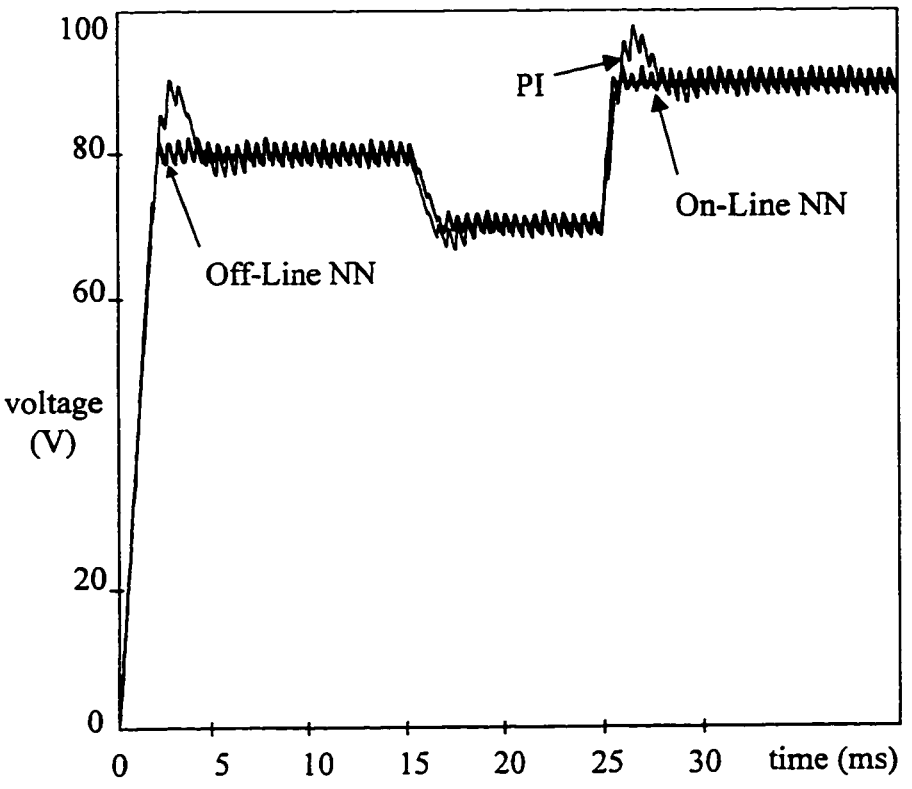


Fig. 3.6 Response to step changes in the reference voltage

TABLE 3.1

Performance Indices Evaluation

CONTROLLER TYPE	% overshoot start up	% overshoot at 70V	% overshoot at 90V
PI	5	9.71	4.79
off-line NN	2	8.9	2.35
on-line NN	1.68	0.7	1.6

Each regulator performed to the best of its ability however, their physical limitations were reached when the load was dropped from its initial value of 0.8Ω to a value of 0.1Ω and then increased to double its original value. Their responses are shown in Fig. 3.7.

The system with the on-line NN in place was tested with the natural frequency held constant while the damping ratio was varied. The training time is defined as the settling time of the output response. It may be observed from Fig. 3.8 and Table 3.2 that once the on-line NN has acquired a base value for its weights during start up, its response to subsequent variations in the reference is rapid and dependent of the load connected.

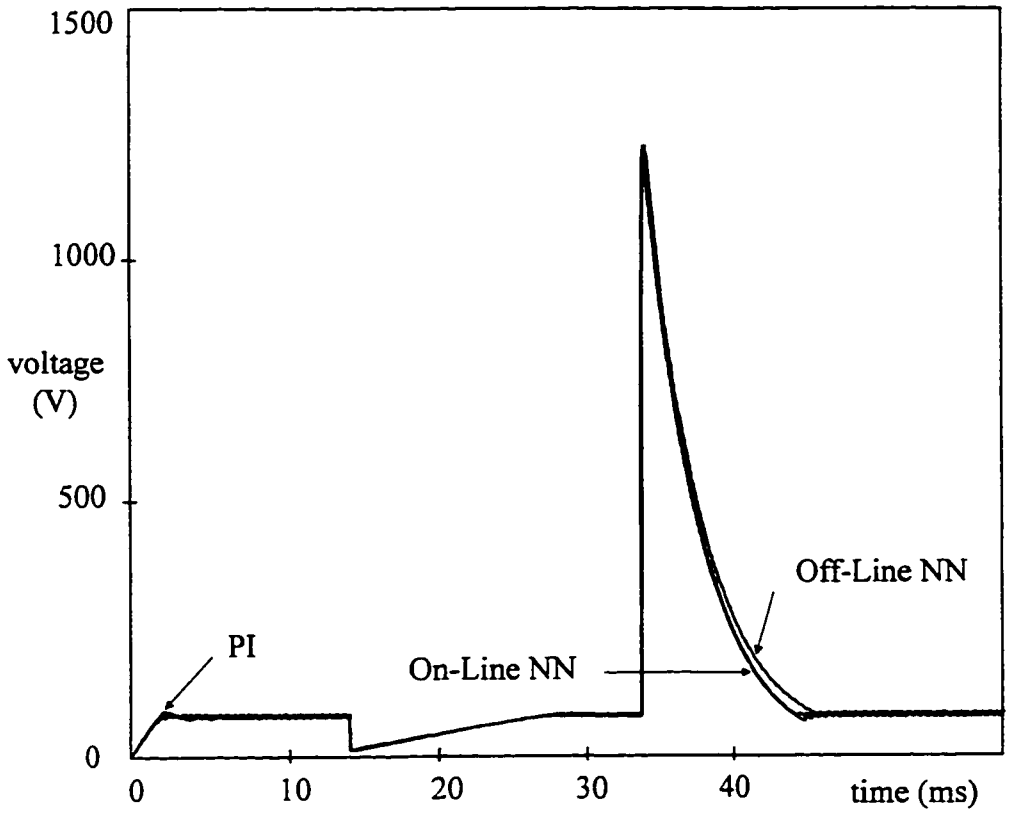


Fig. 3.7 Response to step changes in load resistance, PI, Off-line and On-line NN controllers.

TABLE 3.2

Training Time Evaluation For on-Line NN

	load r = 0.4Ω	load r = 0.8Ω	load r = 1.6Ω
start up	3.2mS	2.3mS	1.8mS
11% step down	3.4mS	2.4mS	2.3mS
25% step up	1.4mS	1.3mS	1mS

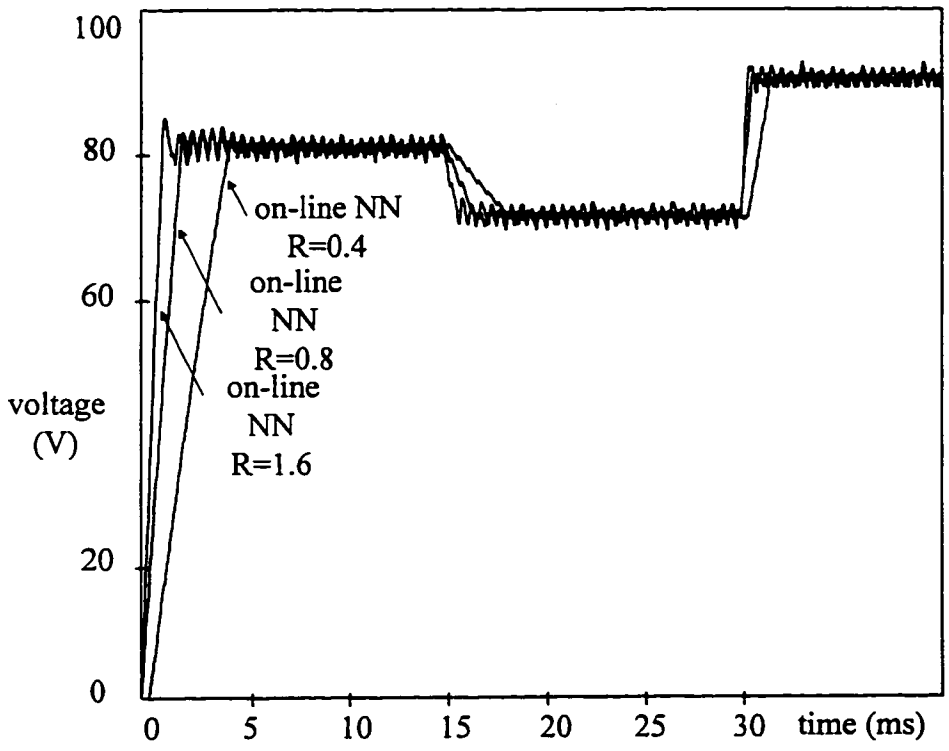


Fig. 3.8 Training times for On-line NN controller.

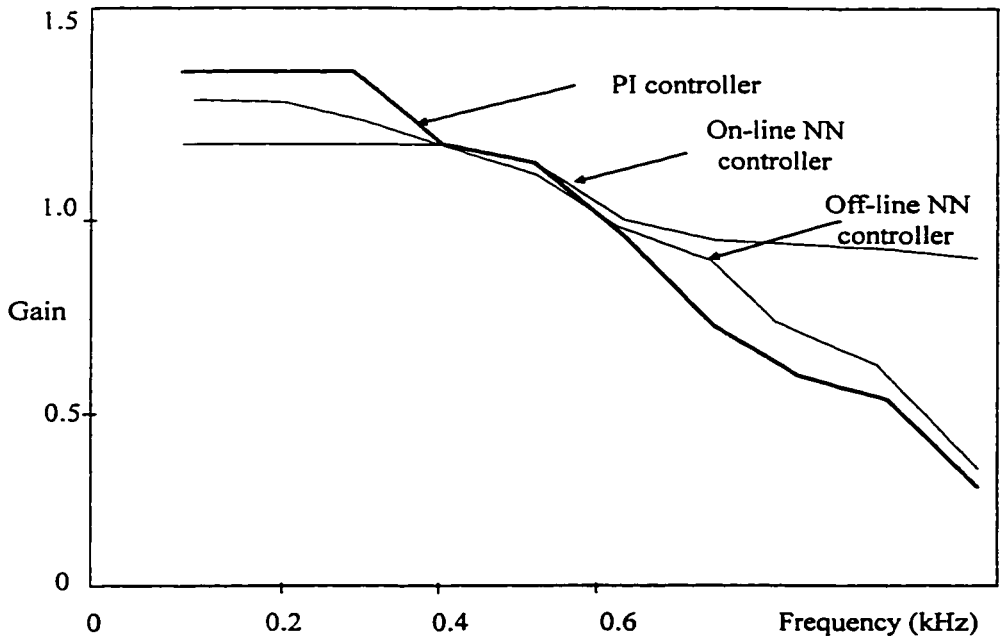


Fig. 3.9 Frequency response R=0.8Ω a) PI controller, b) Off line NN controller c) On line NN controller

The bandwidth of each system was tested and their response is shown in Fig. 3.9. They were fed with a reference upon which a sinusoidal signal was overlaid. The frequency of the sinusoid was varied from 100Hz to 1kHz. The bandwidth of the PI controller in general is restrained by the cut-off frequency of the design which is dependent on the resonant frequency of the system. The off-line trained NN has a bandwidth similar to that of the PI controller. Whereas the gain of the on-line NN is near unity throughout all the test frequencies. Furthermore, these bandwidth figures change substantially with load for the PI and the off-line NN but not for the on-line NN.

3.5 CONCLUSIONS

The simulation results show that the NN controllers offer stable response and good output regulation. The buck system is linearized about a single operating point to complete a PI controller design. The training set for the of the off-line NN was based on this design. Its training was lengthy and dependent upon the size and the quality of the data set. However good results were obtained. The on-line NN required the least effort in terms of design time and provided the most accurate and uniform results under changing load conditions. Finally, the NN offers fast dynamic response and enhances the performance of DC to DC buck converters over the conventional PI regulated converters.

CHAPTER 4

VOLTAGE CONTROLLED PWM RECTIFIER

4.1 INTRODUCTION

This chapter proposes a NN implementation for a voltage controlled PWM current source rectifier, used as a general purpose dc voltage source. The simplest method of operating three-phase PWM rectifiers is based on the use of off-line PWM patterns. However, in this scheme the input power factor may be less than unity since it varies with the rectifier operating point due to the presence of the input LC filter. Furthermore, the response to transient conditions is slow and large current oscillations may occur due to the resonance of the filter. Here an on-line Neural Network controller, is proposed to waveshape the input line currents, force unity power factor operation and damp the low frequency resonance of the input filter. The proposed controller is insensitive to load/parameter variations thus resulting in a robust system. The performance of the proposed NN controller is verified through simulation.

4.2 POWER CONVERTER AND CONTROL REQUIREMENTS

Current-source type PWM rectifiers are used as the front-end ac/dc converter in power electronic systems (Fig. 4.1). The load in Fig. 4.1 can be a CSI driven induction motor requiring regulated dc current or a load operating on regulated dc voltage. Direct interfacing with ac mains often imposes stringent specifications on the rectifier such as: (a) low input current harmonics and (b) high input power factor. In standard schemes, the

current-source PWM rectifier is operated with off-line patterns which result in slow transient response, with discontinuous control of modulation index [11],[12]. Also, in order to avoid current oscillations during starting and transients, a sufficient amount of damping resistance must be provided in the input filter circuit. This reduces the overall system's efficiency and filter effectiveness. Recently, on-line pattern generators have been proposed [13]-[15]. In these schemes the control of input current oscillations is achieved by inserting damping resistors or using complicated feedback loops. As a result the stability region may be limited and the system requires a precise design of the control loop components. Preliminary investigations have shown that NN technology has the potential to improve the control of the power electronic systems [5,6,9,15,16,17]. A NN controller is proposed as a means to solve the problems introduced by non-linearities in the power converter topology, and to obtain a rugged controller. NNs have self adapting capabilities which makes them well suited to handle non-linearities, uncertainties and parameter variations which may occur in a controlled plant.

In this chapter, the task of the NN controller is to waveshape the input line currents of the PWM rectifier. The input line currents of the rectifier are controlled in a closed loop fashion. The purpose is to obtain unity power factor operation and effectively damp low frequency resonance of the input LC filter. Also very low levels of harmonic current are injected into the ac mains and fast response to transient conditions are obtained. The overall system is robust since, the NN has self organizing capability and is insensitive to load/parameter variations.

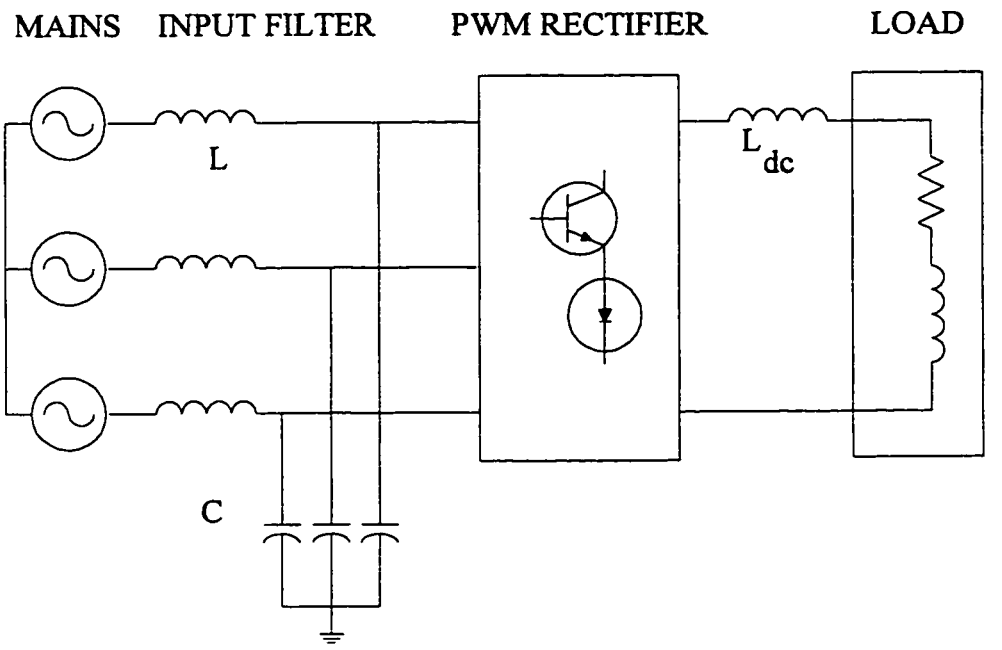


Fig. 4.1 PWM rectifier with input filter.

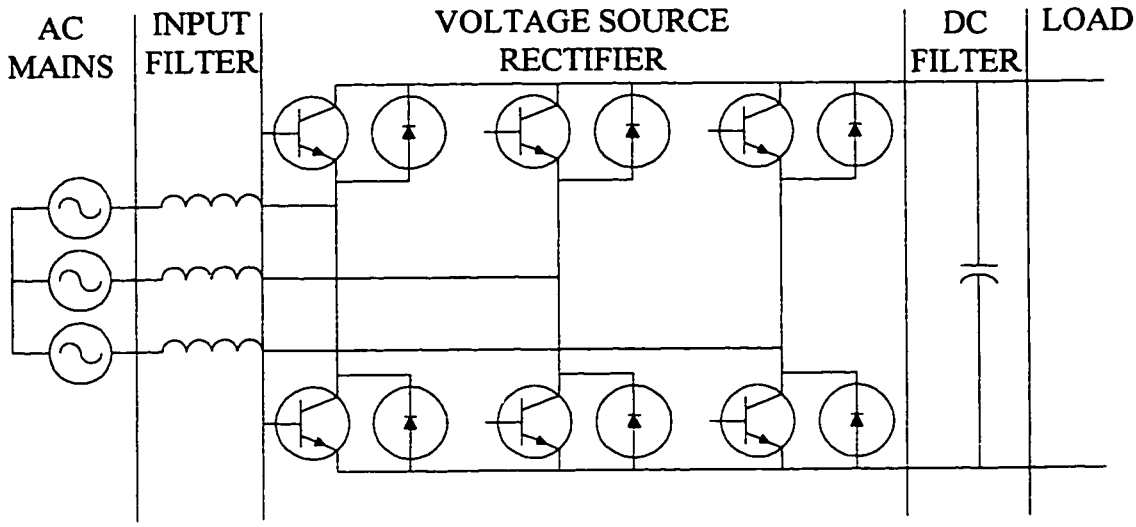


Fig. 4.2 PWM voltage source rectifier, power section.

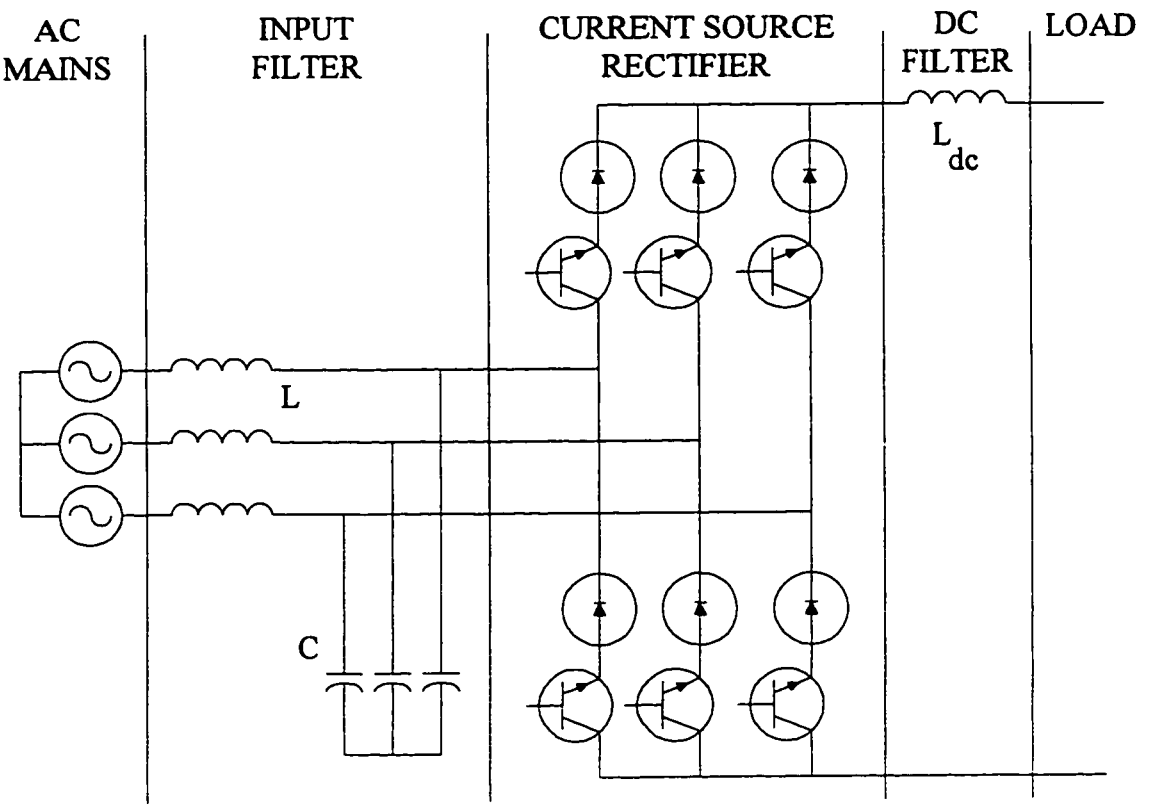


Fig. 4.3 PWM current source rectifier, power section.

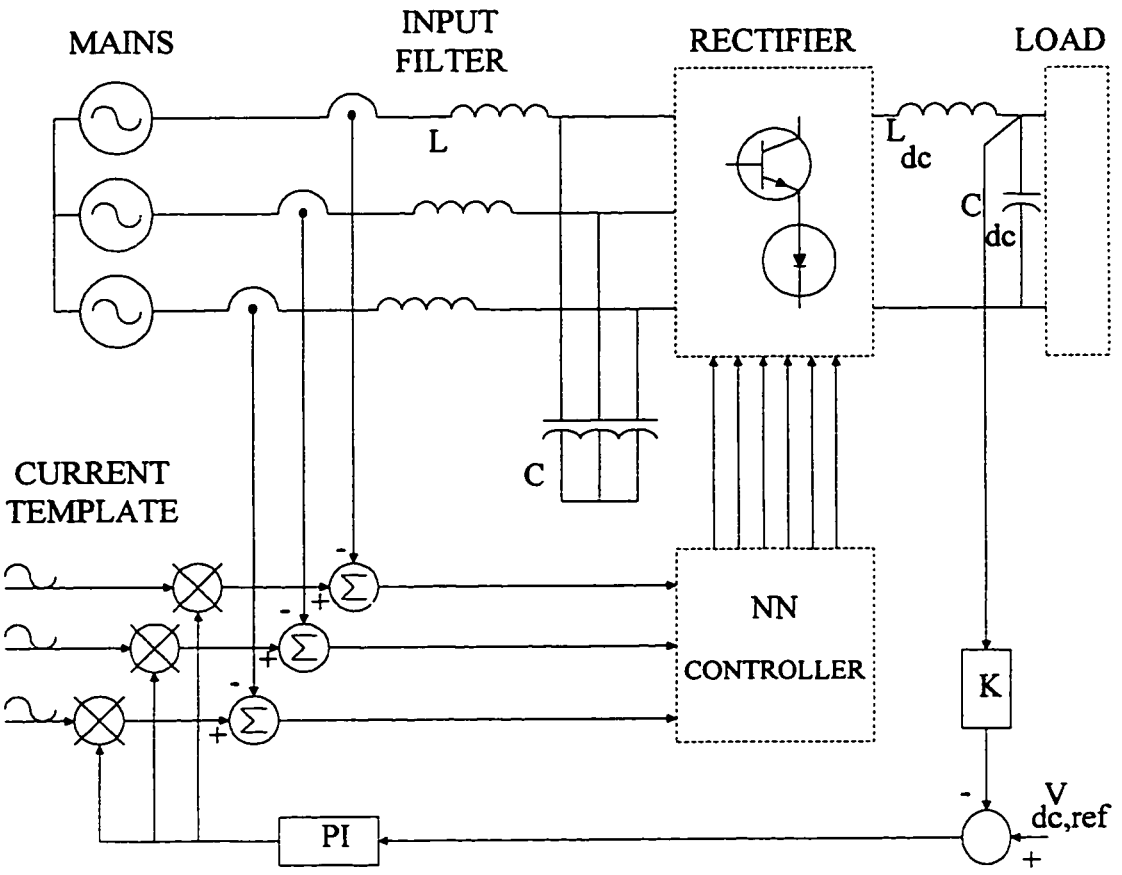


Fig. 4.4 PWM rectifier with the proposed NN controller.

4.3 PWM RECTIFIER SYSTEM EQUATIONS AND OPERATION

The voltage source and the current source PWM rectifiers including the power section is shown in Fig. 4.2 and Fig. 4.3. The rectifier system shown in Fig. 4.4, comprises the system upon which the control is based. The rectifier is described by the following differential equation:

$$\frac{dX}{dt} = A \cdot X + B \cdot U \tag{4.1}$$

where;

$$X = \begin{bmatrix} I_{l_1} & I_{l_2} & I_{l_3} & V_{c_1} & V_{c_2} & V_{c_3} & I_{l_{dc}} & V_{c\,dc} \end{bmatrix}^T \tag{4.2}$$

$$U = \begin{bmatrix} V_a & V_b & V_c & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.3}$$

$$B = \begin{bmatrix} [I_{3\times3}] & [0] \\ [0] & [0] \end{bmatrix}_{8\times8} \tag{4.4}$$

where I_{l1} = line currents, V_{c1}= filter capacitor voltages, I_{ldc} = dc inductor current and V_{cdc}=dc capacitor voltage. As can be seen from the Appendix C, matrix A includes time varying parameters (S_1 , S_2 , S_3). These are the line-to-line switching functions that dictate the shape of the input rectifier current. The gating signals of each switch must satisfy the typical requirements of a current-source rectifier. The gating generator is not shown in Fig. 4.4 and is discussed in [15]. Also, it is seen that the state variables of the system include inductor currents and capacitor voltages. Since, the three capacitor voltages are not controlled directly, the input currents may oscillate. However, a tight control over the inductor currents with minimal damping resistance in the filter circuit

can effectively damp these oscillations. The NN controller provides this necessary control. The control circuit consists of (a) the NN current controller for generating the current pattern and (b) a standard dc loop for output voltage regulation. Alternatively, output current control can be implemented for the outer loop.

4.4 THE NN CONTROLLER

The NN structure is shown in Fig. 4.5. Each neuron in the back propagation neural network (BPN) has multiple inputs and one output. Each neuron's output is a function of the input, the weights associated with that layer, a bias term and a node activation function. The NN consists of three distinct layers. The input layer, which is used to establish connection points to transfer the input signal to the nodes in the hidden layer. A hidden layer which begins the learning process and an output layer continues the learning process and provides outputs.

Once the signal e_j flows through the connection weights associated with the hidden layer (Wh) it is combined with a bias term (B_h) and fed through the node non-linearity. This signal becomes the input to the output layer, (I_j), flowing through connection weights (W_O), to be combined with bias term (B_O) and then processed by the output layer to provide the gating signals to the rectifier.

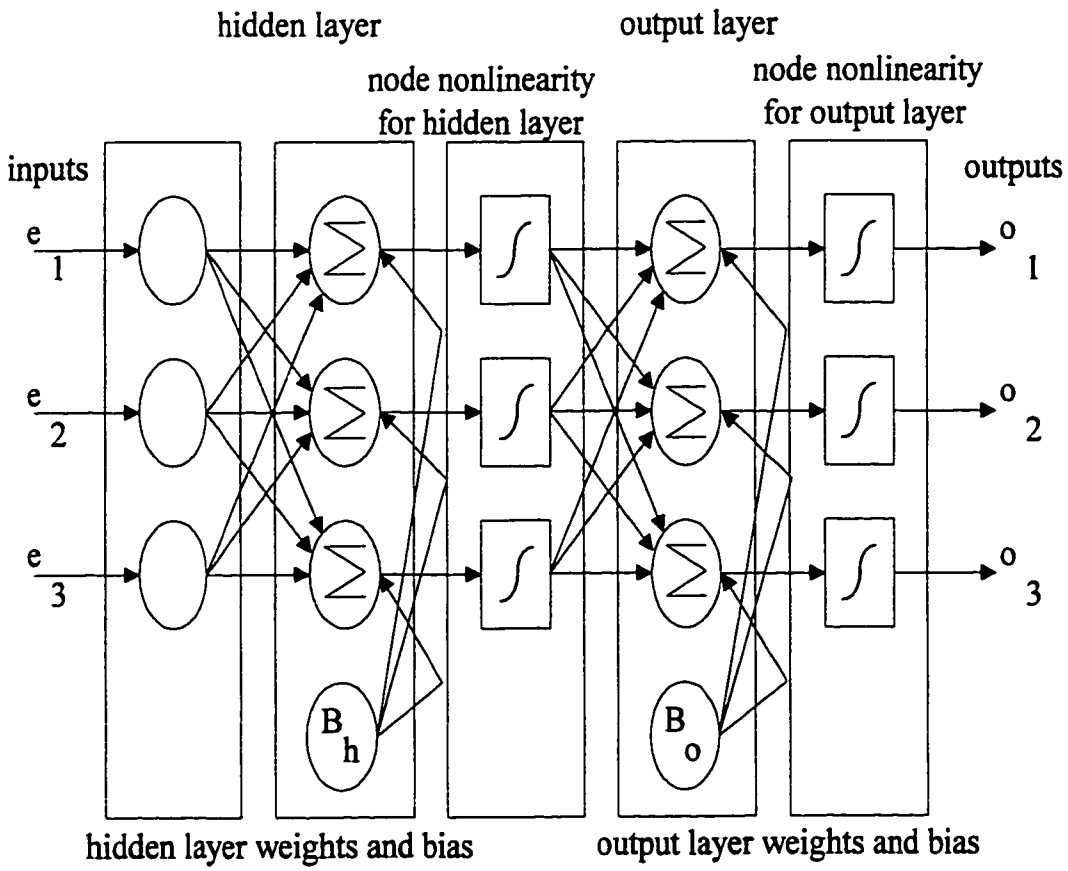


Fig. 4.5 NN structure.

The equations associated with the signals flowing from each layer to the next are:

$$I_j = f\left(W^{\mathrm{T}}{}_{h_j} \cdot e^{\mathrm{T}} + B^{\mathrm{T}}{}_h\right) \tag{4.5}$$

$$O_j = f\left(W^{\mathrm{T}}{}_{o_j} \cdot I^{\mathrm{T}} + B^{\mathrm{T}}{}_o\right) \tag{4.6}$$

where, I_j = the input to the output layer, O_j = the output of the NN, f = the sigmoid non-linearity, e_j = the input to the hidden layer and B_j = bias term for each layer.

Gradient descent minimization is used to optimize the weights on each layer such that the output of the NN will approach the desired trajectory. Thus the error between the target and the net output will be minimized. The weights associated with each layer will be

updated using the following equations:

$$w_{h_{ij}} = w_{h_{ij}} + \eta \cdot \partial_{h_j} \cdot e_j \tag{4.7}$$

$$w_{o_{ij}} = w_{o_{ij}} + \eta \cdot \partial_{o_k} \cdot I_j \tag{4.8}$$

where, w_h = weight on the input layer, e = error between the reference current and the measured current (i_{ref} - $i_{measured}$), η = learning rate, δ_{hj} = error signal term associated with the hidden layer, W_O = output layer weights, I_j =output of the hidden layer (i.e. input to the output layer) and, δ_{oj} = error signal term associated with the output layer of the net and the desired target. The control objective of the NN structure will be to provide the required gating to force the PWM rectifier to have a constant output dc voltage while maintaining unity power factor on the input side.

4.5 THE PATTERN GENERATION SCHEME

The target pattern generation scheme is based on the ramp comparison technique and is described by:

$$\begin{aligned} &if\ e \rangle carrier \quad Target = 1 \\ &else \quad Target = 0 \end{aligned} \tag{4.9}$$

The output of the NN is continuous and follows the targets defined above. It must be digitized to obtain the necessary gating signals for the rectifier, resulting in near constant switching frequency. The digitized gating pattern is the output of the NN passed through the hard limiter given by:

$$\begin{aligned} &if\ O \rangle 0.5 \quad O = 1 \\ &else \quad O = 0 \end{aligned} \tag{4.10}$$

4.6 THE DC VOLTAGE REGULATION LOOP

A standard voltage loop with a PI regulator is used initially to vary the output voltage according to the load demands. The characteristics of the PI regulator and the limitation of the performance of the proposed scheme are discussed in the next section. An alternative design will also be presented.

4.7 DESIGN CRITERIA

In this section, the effect of different control parameters both inside and outside the NN controller on the performance characteristics of the PWM rectifier is studied. The dq transformation is used as a tool to transform the ac values to dc values in the rotating (dqo) frame tied to the ac mains. This enables one to observe and study the step response of the system more clearly. In each case the output dc voltage and the q-axis current are shown. The d-axis current (not shown) is zero, since unity power factor is achieved.

4.7.1 NUMBER OF NEURONS IN THE HIDDEN LAYER

The number of neurons in the hidden layer was chosen considering the convergence time. The NN for the control scheme shown in Fig. 4.4 required a minimum of two for neurons in the hidden layer, while that shown in Fig. 4.11 needed a minimum of 18 to achieve good results. Fig. 4.6 indicates that increasing the number of neurons does not necessarily improve the convergence time. The effect of number of neurons on the step response of the system design with the PI regulator is depicted in Fig. 4.7. It can be seen that the transient response of the system is not significantly affected by an increase in neurons.

be seen that the transient response of the system is not significantly affected by an increase in neurons.

4.7.2 OUTPUT PI REGULATOR

The dc voltage loop regulates the output voltage according to the load demands. The voltage loop uses a PI to vary the amplitudes of the current references. These currents are then compared with the actual line currents and the three current errors are the input to the NN controller.

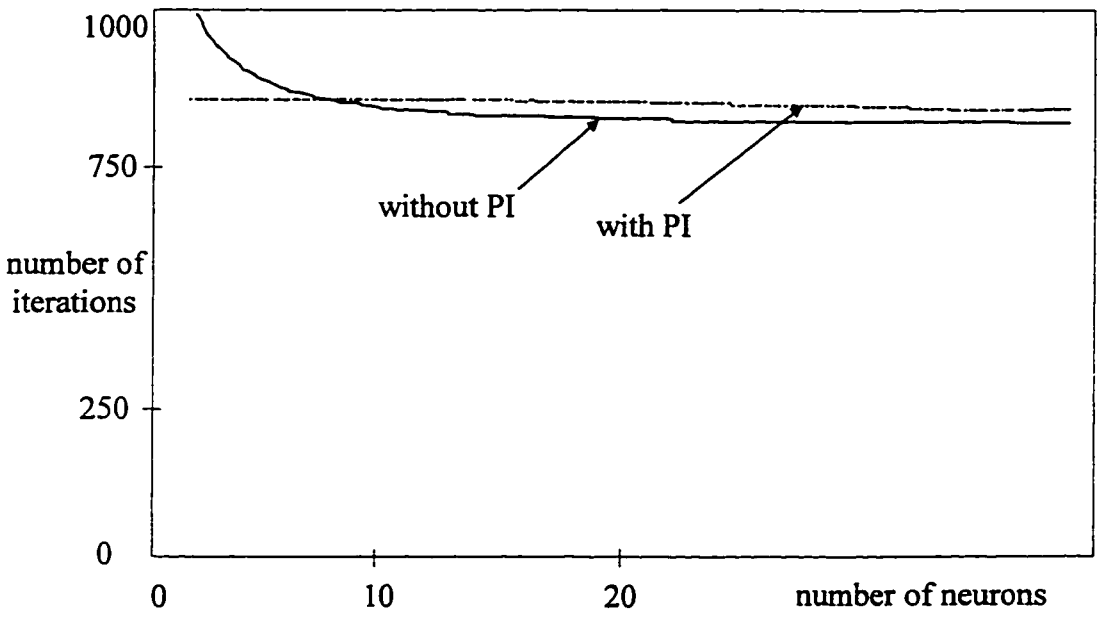


Fig. 4.6 Effect of number of neurons on the convergence time, ac current control.

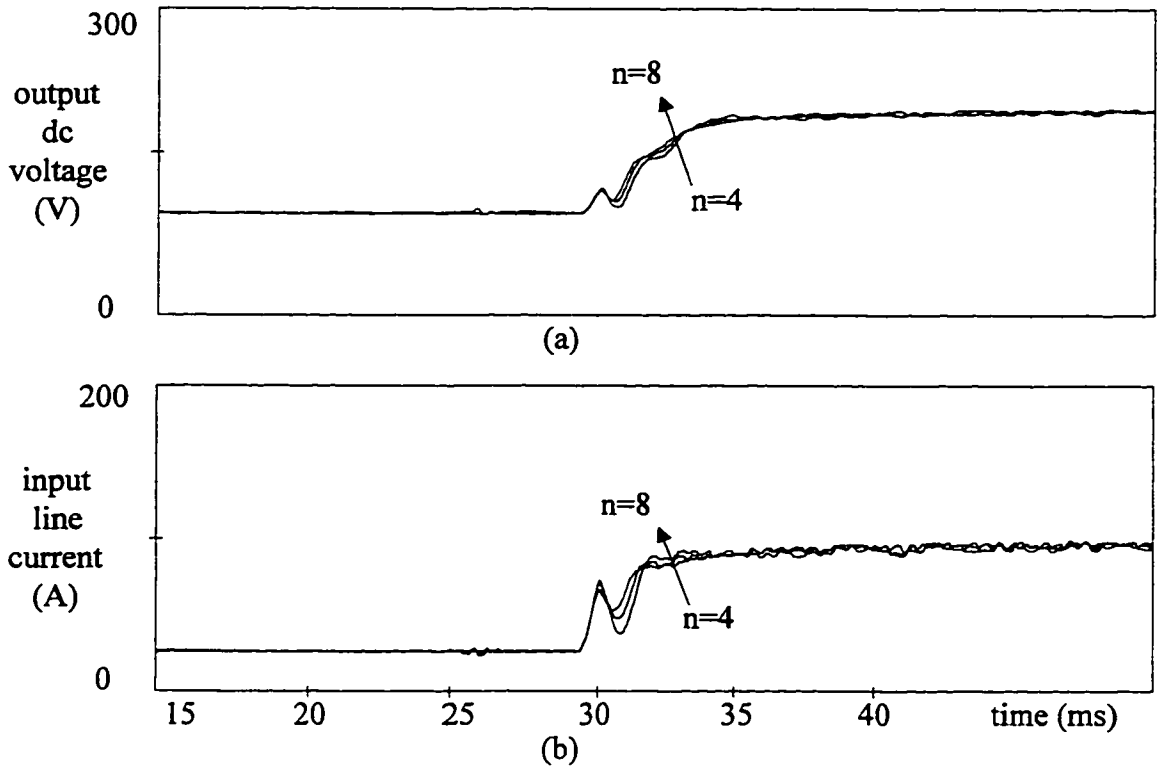


Fig. 4.7 Effect of number of neurons on current loop response, (a) output dc voltage, (b) input line currents transformed to dq frame, I_q.

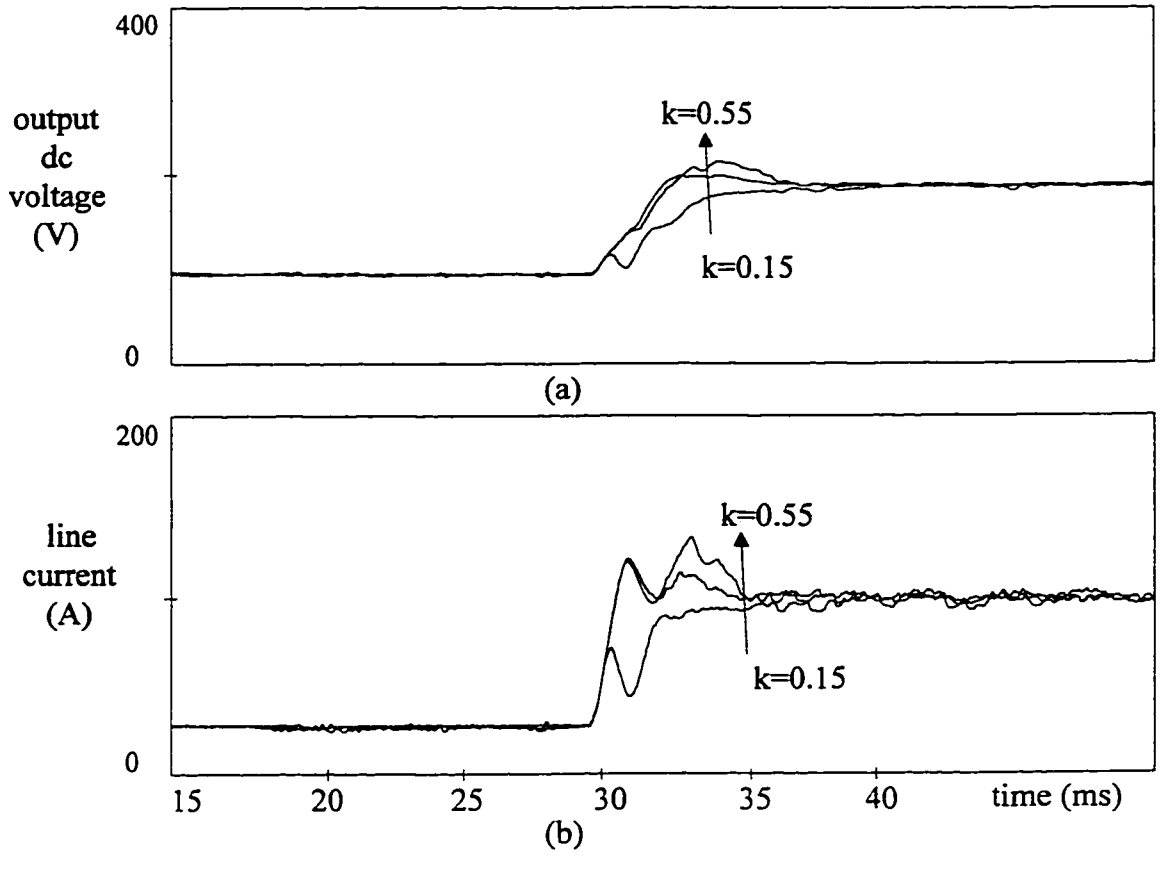


Fig. 4.8 Effect of proportional gain PI voltage controller, (a) output dc voltage, (b) line current transformed to dq- frame, I_q.

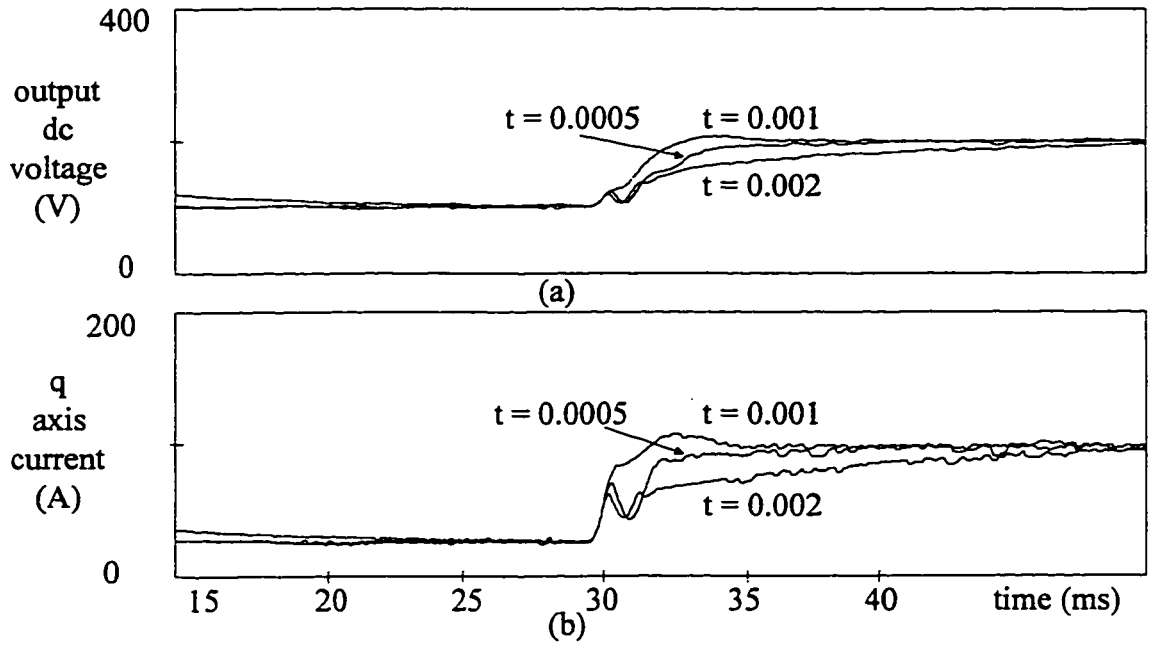


Fig. 4.9 Effect of PI voltage controller time constant, (a) output dc voltage, (b) q-axis current.

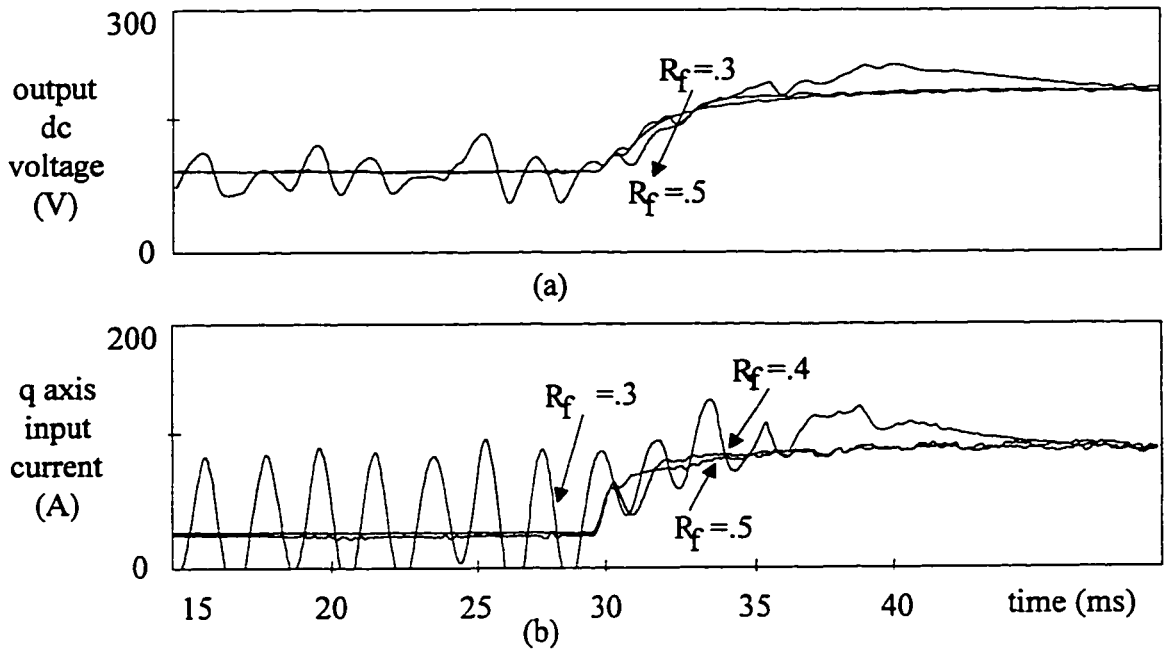


Fig. 4.10 Effect of damping resistance, (a) output dc voltage, (b) q-axis component of the input current.

The PI regulator was designed based on the overshoot in the step response of the system. The three phase ac currents are transferred to the rotating frame to show the effect of the PI components more clearly. It is seen that increasing the proportional gain or reducing the time constant of the PI may result in a oscillatory system (Figs. 4.8, 4.9). The PI is designed to obtain satisfactory response time and limited overshoot during transient conditions.

4.7.3 DAMPING RESISTORS

The system under consideration always needs a minimum damping especially for low modulation indices. This fact is obvious from Fig. 4.10. Where the damping resistors are decreased, current oscillations appear for low modulation index operation. The damping required is the same as that in reference [15].

4.7.4 ALTERNATIVE DESIGN

An alternative design is to replace the PI regulator with another input to the NN controller. The overall system is shown in Fig. 4.11. The new NN block has four inputs and four outputs and is shown in Fig. 4.12. The fourth output of the net (O_d) obtains the input current amplitude reference required to achieve the demanded dc voltage. This design requires more neurons in the hidden layer for the algorithm to converge (in this case 18). The effect of number of neurons on the number of iterations was shown in Fig. 4.6.

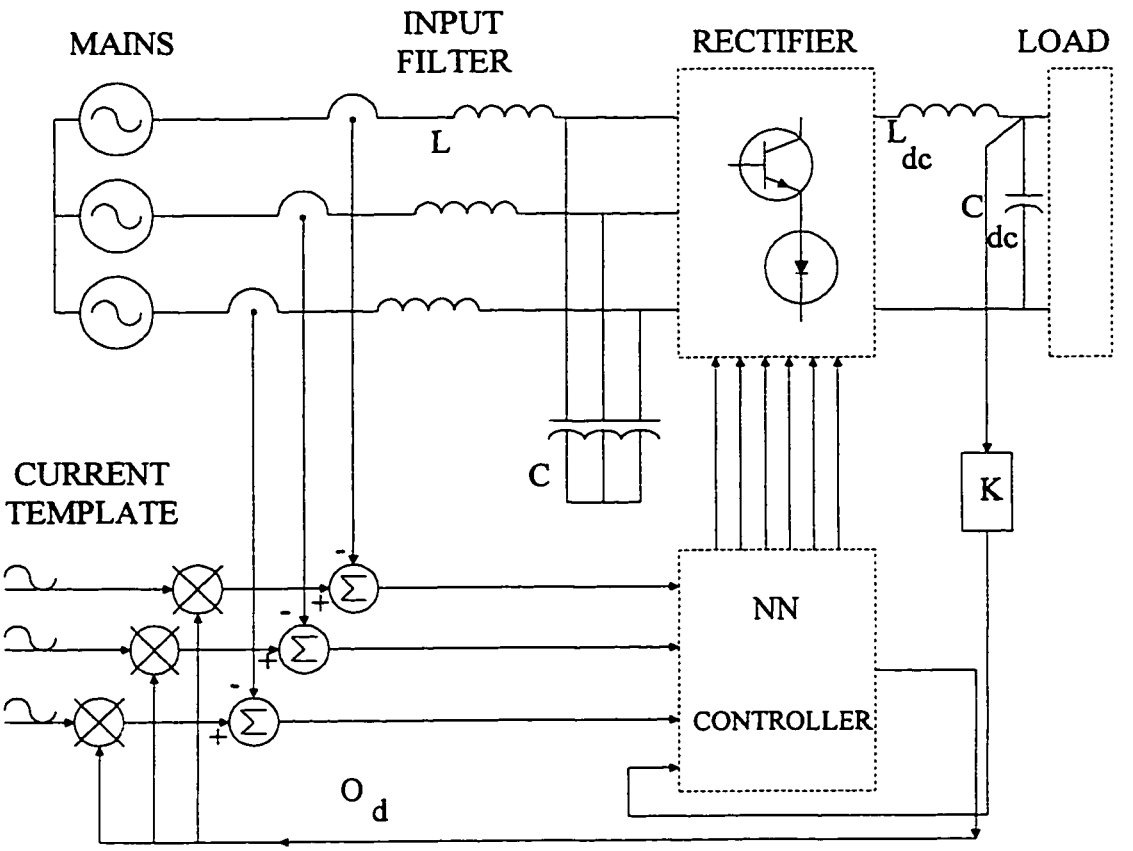


Fig. 4.11 Alternative design for the dc voltage loop

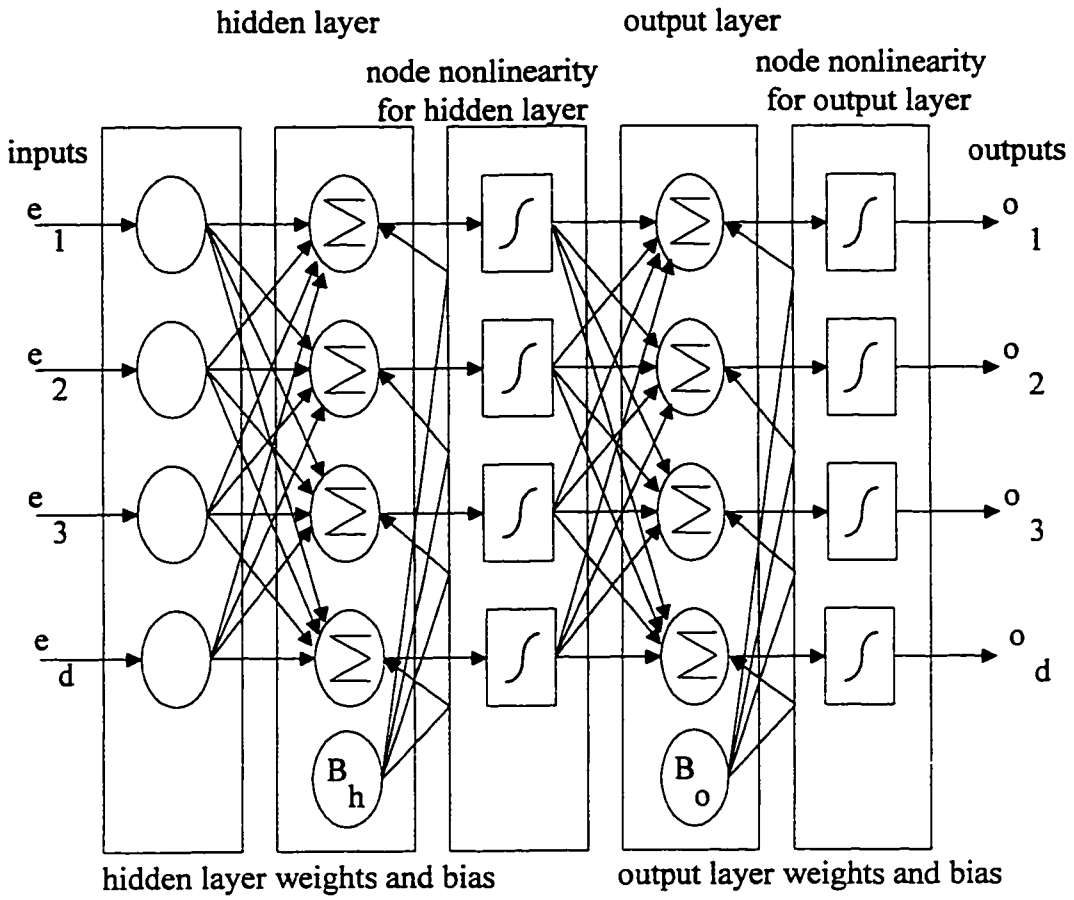


Fig. 4.12 Incorporation of the PI voltage into the NN.

4.8 RESULTS

4.8.1 STEADY STATE

The starting and steady state performance of the PWM rectifier with NN controller is shown in Fig. 4.13. The rectifier has a better starting and a wider range of operation as compared to [15]. The controller provides good tracking of the reference waveforms and unity power operation is achieved. The input displacement factor as a function of rectifier operating point is shown in Fig. 4.14. For the purpose of comparison the case of a rectifier with off-line SPWM pattern is also included. This figure clearly shows that the proposed NN controller provides unity displacement factor over its entire operating region. The harmonic distortion of the input line current is very low. This means that the overall input power factor remains near unity. The frequency spectra of the line current and switching function (S_i) are depicted in Fig. 4.15. It is seen that the dominant harmonic is around the switching frequency (2160Hz).

4.8.2 DC BUS TRANSIENTS

The response of the system is studied under various transient conditions. The reference dc voltage is changed from 100V to 200V while in another step the load resistance is doubled. The results for the two designs shown in Figs. 4.4 & 4.11 are shown in Fig. 4.16. The NN controller successfully replaces the PI. The proposed rectifier exhibits excellent transient response. There are no low frequency oscillations in the currents due to the closed loop control. Power factor remains near unity.

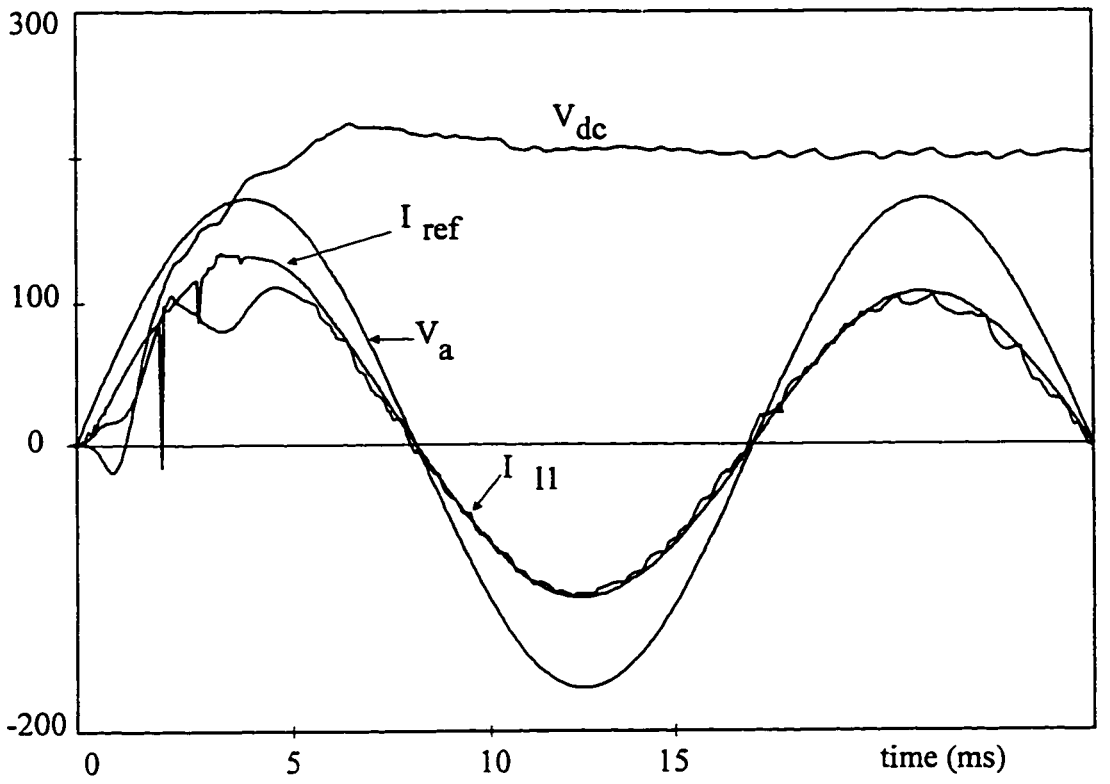


Fig. 4.13 Start up dc transient conditions output dc voltage V_{dc} (V), current reference I_{ref} (A), input line current I_{l1} (A), and phase voltage V_a (V).

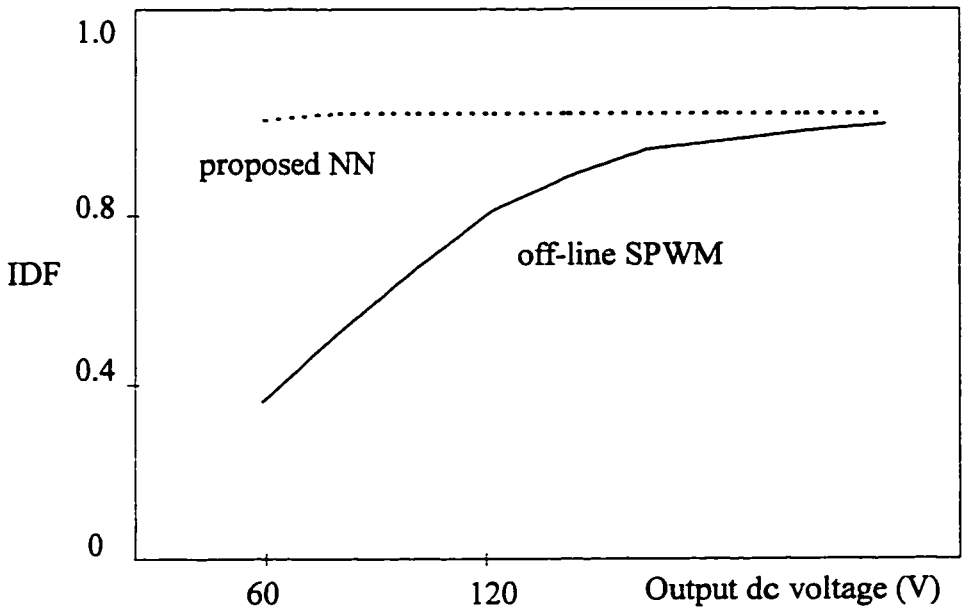


Fig. 4.14 Effect of operating point (output dc voltage) on the input displacement factor (IDF).

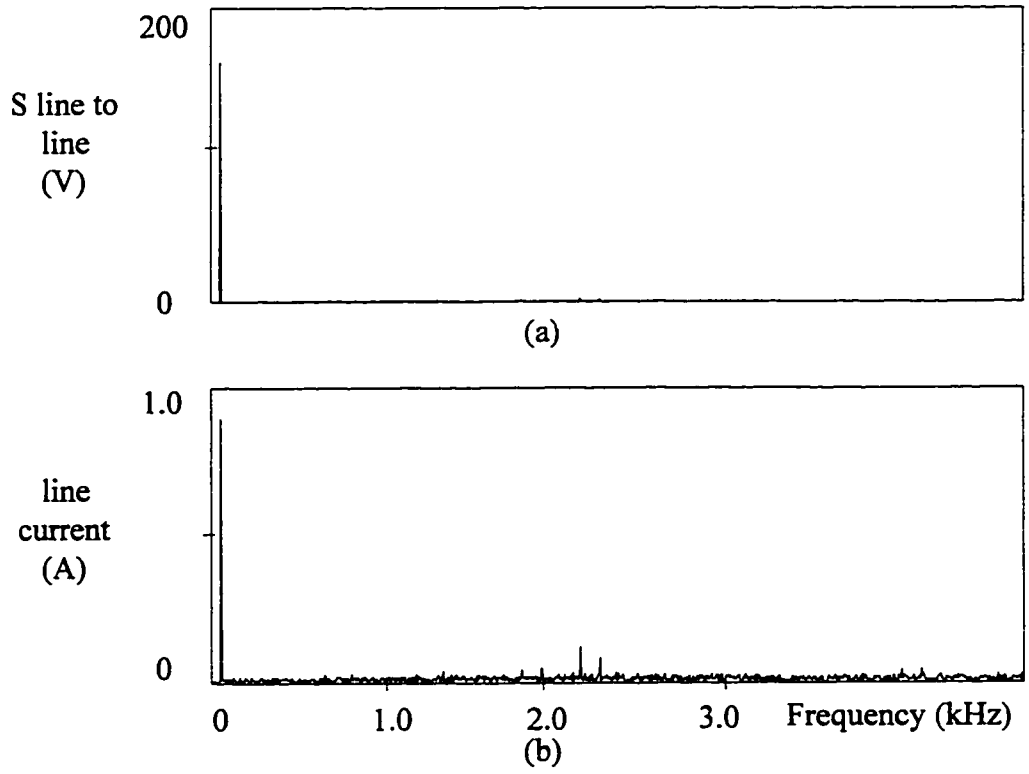


Fig. 4.15 Frequency spectra, (a) line to line switching function S and (b) line current (switching frequency = 1.98 kHz).

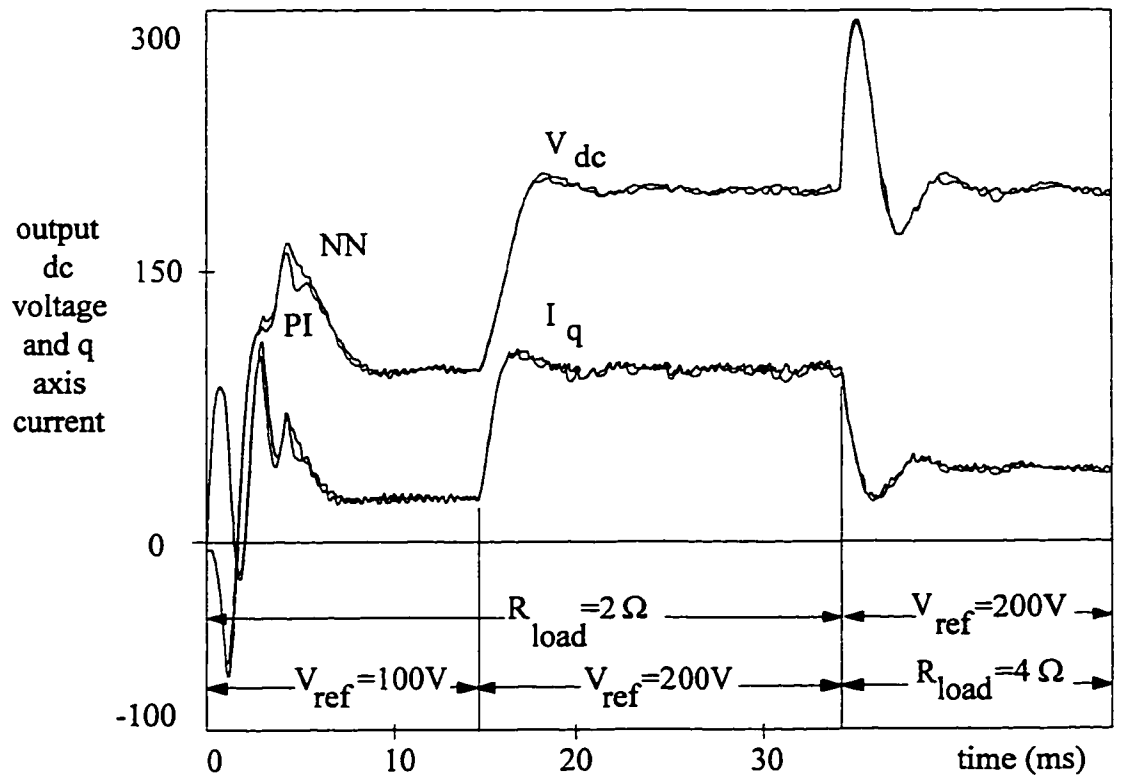


Fig. 4.16 Transient response at start up of the two proposed designs. Output dc voltage V_{dc} (V) and q-axis current, I_q (A).

4.9 CONCLUSIONS

An on-line Neural Network is proposed to waveshape the input line currents of a PWM rectifier and provide unity power factor operation. The Neural Network obtains a robust system since it is insensitive to load/parameter variations. Low frequency resonance of the input filter is effectively damped through closed loop control. Excellent steady state waveforms and a fast response to transient conditions are obtained. Theoretical considerations were verified through simulation.

CHAPTER 5

THREE PHASE CURRENT SOURCE PWM RECTIFIER

5.1 INTRODUCTION

This chapter presents a NN implementation of a current controlled current source, to be used as a front end for a current source inverter in applications such as adjustable speed ac drives. Three-phase current source PWM rectifiers are generally operated with off-line PWM patterns. This results in non-unity input displacement factor and possible large inrush current due to the presence of the input LC filter. Moreover, the input displacement factor is dependent upon the operating point and response to transient conditions is slow.

The overall system is robust since, the NN has self organizing capability and is insensitive to load / parameter variations. Therefore the task of the NN controller is to waveshape the input line currents of the PWM rectifier. The input line currents of the rectifier are controlled in a closed loop fashion. The purpose is to obtain unity power factor operation and effectively damp low frequency resonance of the input LC filter without the insertion of damping resistors.

5.2 POWER CONVERTERS AND CONSTRAINTS

Current-source type PWM rectifiers are used as the front-end ac/dc converter in power electronic systems (Fig. 5.1). The generalized three phase current source PWM rectifier system is shown in Fig. 5.2, and the three phase PWM voltage source rectifier is shown in Fig. 5.3. Direct interfacing with ac mains often imposes stringent specifications on the rectifier such as: (a) low input current harmonics and (b) high input power factor. In standard schemes, the current-source PWM rectifier is operated with off-line patterns which result in slow transient response, with discontinuous control of modulation index [11]. Also, in order to avoid current oscillations during starting and transients, a sufficient amount of damping resistance must be provided in the input filter circuit. This reduces the overall system's efficiency and filter effectiveness. Recently, on-line pattern generators have been proposed [13]-[15]. In these schemes the control of input current oscillations is achieved by inserting damping resistors or using complicated feedback loops. As a result the stability region is limited and the system requires a precise design of the control loop components. Preliminary investigations have shown that Neural Network (NN) technology has the potential to improve the control of the power electronic systems [6,7,17,18,19]. A NN controller is proposed in this paper as a means to solve the problems introduced by non-linearities in the power converter topology, and to obtain a rugged controller. NNs have self adapting capabilities which makes them well suited to handle non-linearities, uncertainties and parameter variations which may occur in a controlled plant.

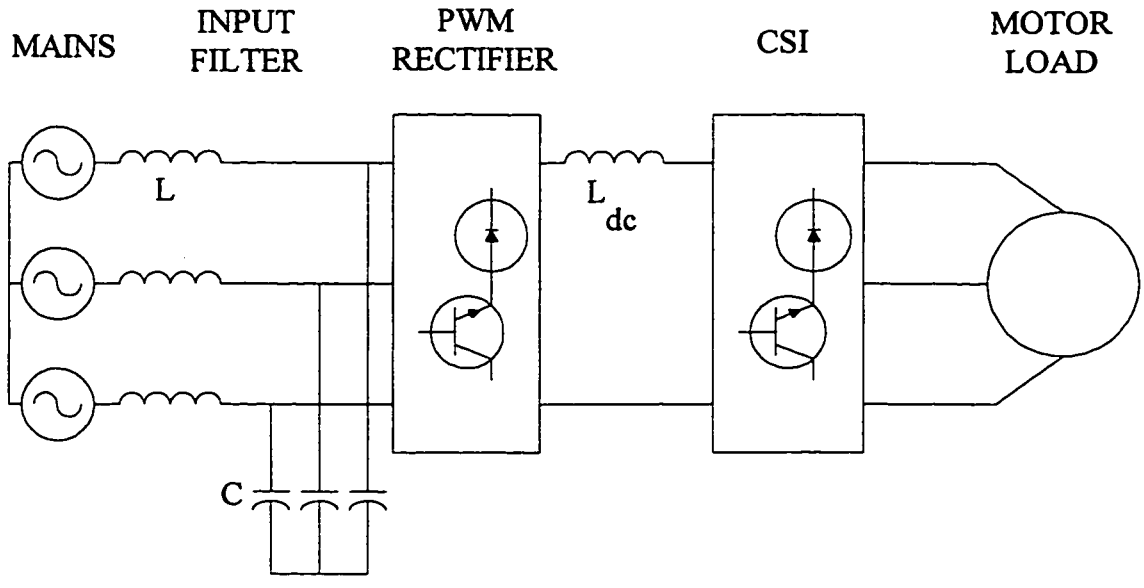


Fig. 5.1 PWM current source rectifier with input filter.

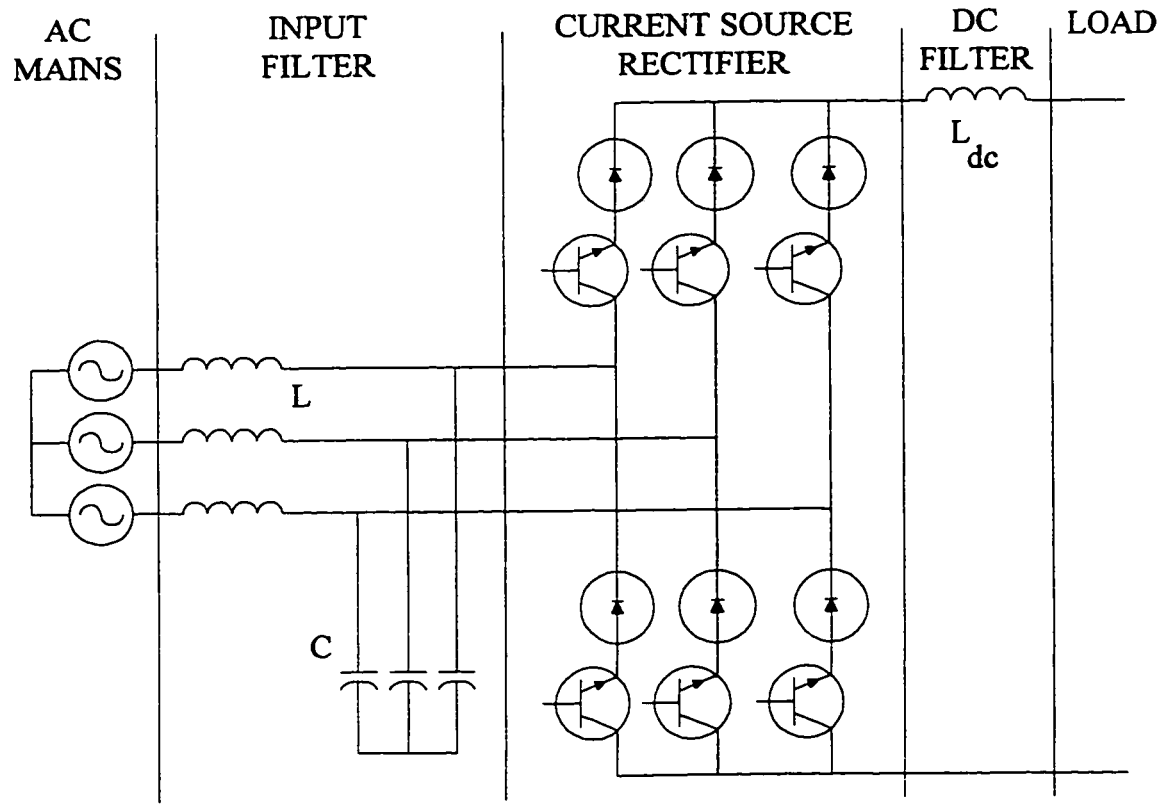


Fig. 5.2 Three phase PWM current source rectifier, power section.

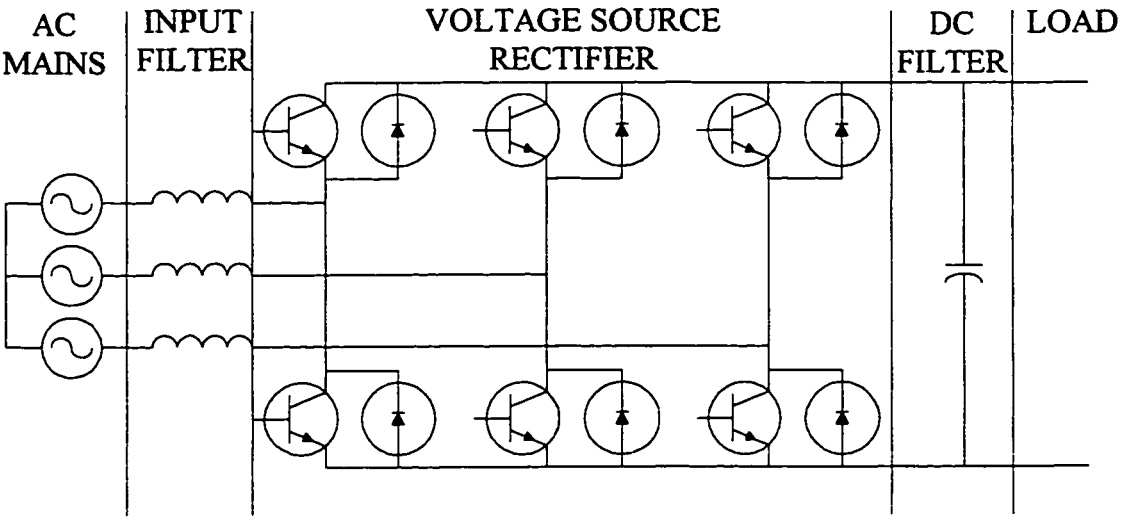


Fig. 5.3 Three phase PWM voltage source rectifier, power section.

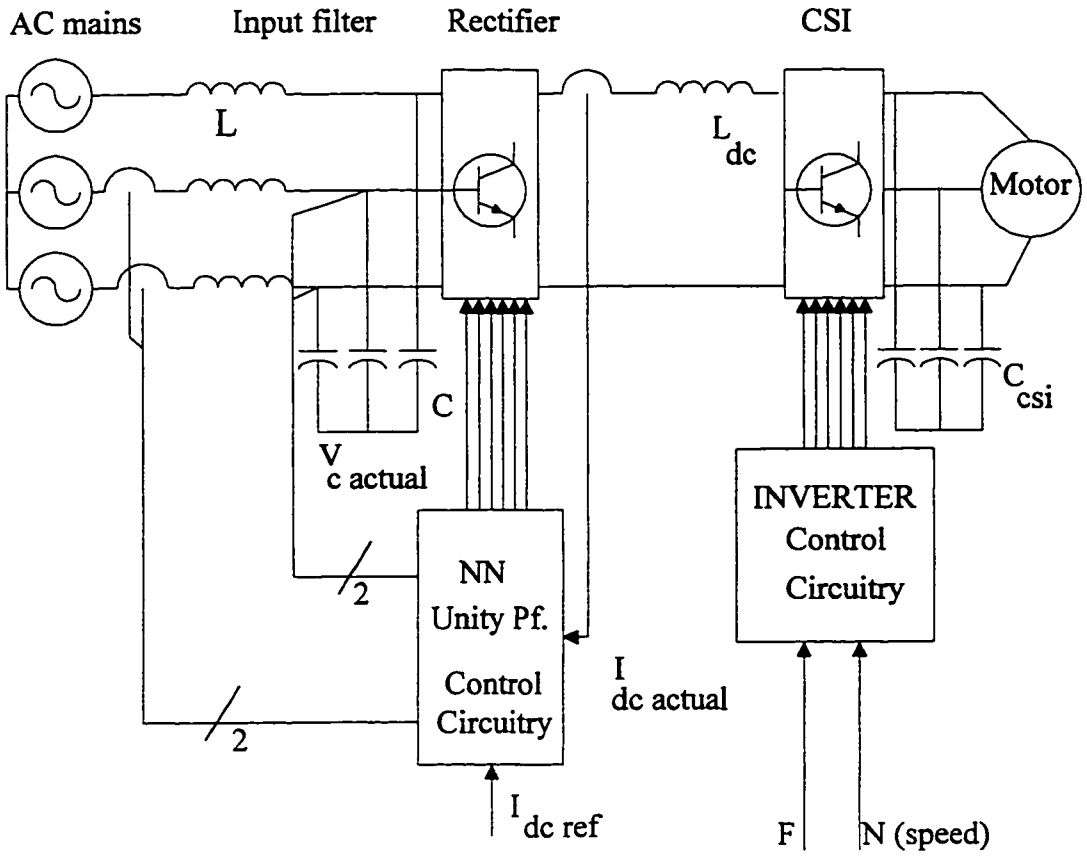


Fig. 5.4 Proposed control structure.

5.3 DESCRIPTION OF THE PROPOSED PWM RECTIFIER

5.3.1 SYSTEM EQUATIONS AND OPERATION

The complete rectifier system is shown in Fig. 5.4. The control circuit consists of the NN current controller and a NN current loop for output current regulation. The NN current loop is used to vary the amplitudes of the current references. The current template is obtained from the capacitor voltages. This results in effective damping of the input currents during starting and transients. Thus the damping resistors (R_f) in the filter capacitor branch are considerably reduced as shown in Table 5.1. Hence the overall efficiency is improved. However, since there is a phase shift between the capacitor voltages and the ac mains Fig. 5.5, the power factor at the source will be less than unity. The power factor will decrease as the current increases, since the angle between the capacitor voltage and the ac mains increases. Therefore, a phase shift is necessary to improve the power factor, especially for high load currents. The required phase shift depends on the rectifier operating point and the filter components and is given by the following equation:

$$\tan\phi = \frac{\left(\omega \cdot L_f \cdot I_m\right)}{\left(V_m - R_l \cdot I_m\right)} \tag{5.1}$$

where V_m = phase voltage amplitude, R = line resistance and L = line inductance and I_m= line current amplitude.

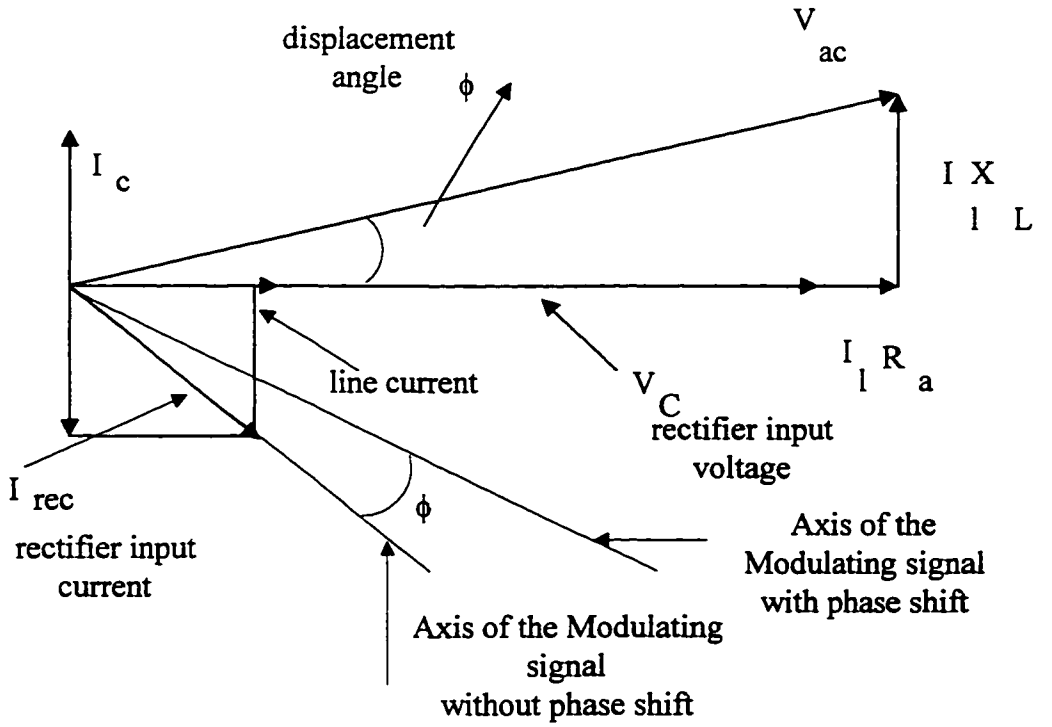


Fig. 5.5 Phasor diagram and reference axis of the modulating signal.

5.3.2 CONTROL OF THE LOAD CSI

To simplify the analysis the motor CSI operates with an off-line SPWM gating pattern and a fixed modulation index (0.91 in the design example).

5.3.3 THE NN CONTROLLER

As shown in Figs. 5.4 and 5.6 there are six inputs to the NN Unity Pf. Control Circuitry block. Three of the four inputs to the NN are current errors while the fourth is the actual dc link current. The control objective of the NN structure will be to provide the required gating to force the PWM rectifier to have a constant output dc current while maintaining unity power factor on the input side. The derivation of the current template

is shown in Fig. 5.6. Note in particular the implementation of the modulating signal phase shifter.

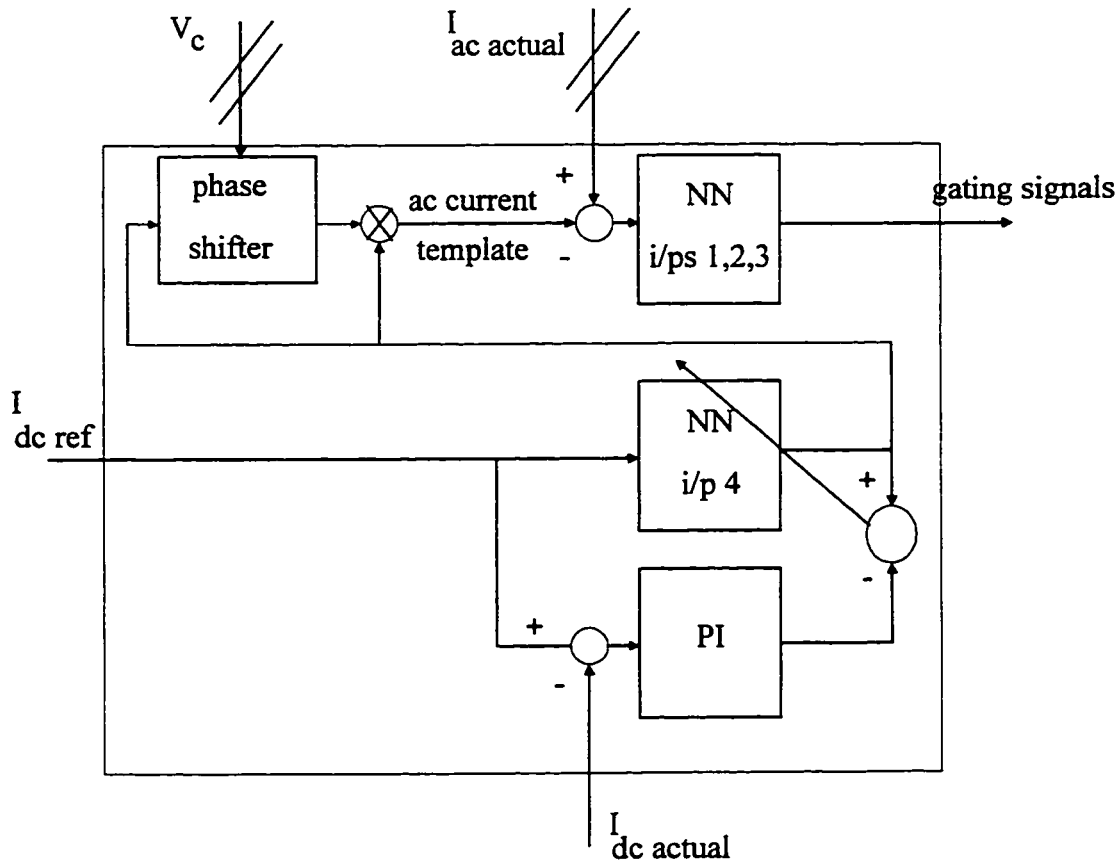


Fig. 5.6 NN unity power factor control circuitry, software implementation.

The NN structure is shown in Fig. 5.7. Each neuron in the back propagation neural network (BPN) has multiple inputs and one output. Each neuron's output is a function of the input, the weights associated with that layer, a bias term and a node activation function. The NN consists of three distinct layers. The input layer, which is used to establish connection points to transfer the input signal to the nodes in the hidden layer. A

hidden layer which begins the learning process and an output layer to continue the process and provide outputs.

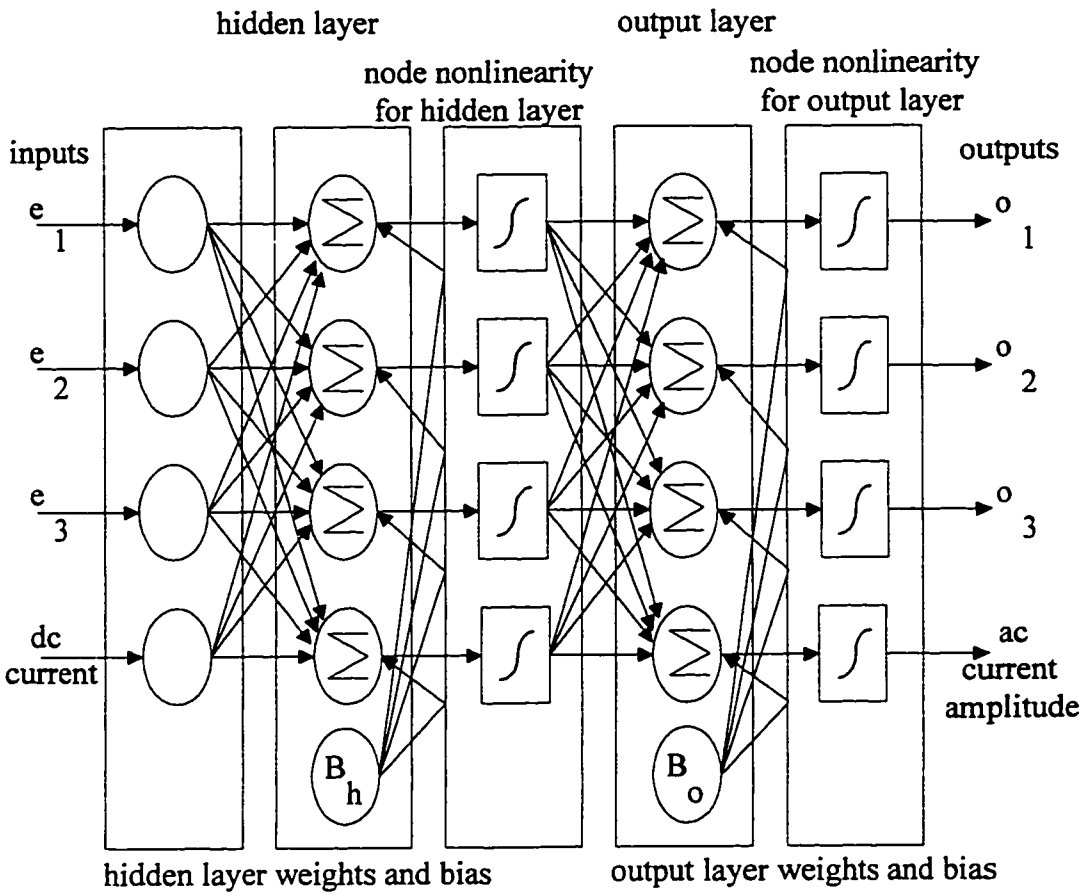


Fig. 5.7 NN structure, input and output current control.

Once the signals from the inputs flow through the connection weights associated with the hidden layer (W_h) they are combined with a bias term (B_h) and fed through the node non-linearities. These become the inputs to the output layer, I_j, flowing through connection weights (W_O), to be combined with bias term (B_O) and then processed by the

output layer to provide the gating signals to the rectifier. The equations associated with the signals flowing from each layer to the next are:

$$I_j = f\left(W^T{}_{h_j} \cdot e^T + B^T{}_h\right) \tag{5.2}$$

$$O_j = f\left(W^T{}_{o_j} \cdot I^T + B^T{}_o\right) \tag{5.3}$$

where, I_j = the input to the output layer, O_j = the output of the NN, f = the sigmoid non-linearity, e_j = the input to the hidden layer and B_j = bias term for each layer.

Gradient descent minimization is used to optimize the weights on each layer such that the output of the NN will approach the desired trajectory. Thus the error between the target and the net output will be minimized. The weights associated with each layer will be updated using the following equations:

$$w_{h_{ij}} = w_{h_{ij}} + \eta \cdot \partial_{h_j} \cdot e_j \tag{5.4}$$

$$w_{o_{ij}} = w_{o_{ij}} + \eta \cdot \partial_{o_k} \cdot I_j \tag{5.5}$$

where, w_h = weight on the input layer, e = error between the reference current and the measured current $(i_{ref} - i_{measured})$, η = learning rate, δ_{hj} = error signal term associated with the hidden layer, w_O = output layer weights, I_j = output of the hidden layer (i.e. input to the output layer) and , δ_{oj} = error signal term associated with the output layer of the net and the desired target.

5.3.4 THE PATTERN GENERATION SCHEME

The target pattern generation scheme is based on the ramp comparison technique and is described by:

$$\begin{aligned} &\text{if } e > \text{carrier } Target = 1 \\ &\text{else } Target = 0 \end{aligned} \tag{5.6}$$

The output of the NN is continuous and follows the targets defined above. It must be digitized to obtain the necessary gating signals for the rectifier, resulting in near constant switching frequency. The digitized gating pattern is the output of the NN passed through the hard limiter given by:

$$\begin{aligned} &\text{if } O > 0.5 \; O = 1 \\ &\text{else } O = 0 \end{aligned} \tag{5.7}$$

These NN outputs (shown in Fig. 5.7) are the phase switching functions. The actual switch gating signals must be obtained generating the line to line pattern and adding the short circuit pulses to ensure a path for the dc link.

5.4 DESIGN EXAMPLE AND RESULTS

5.4.1 DESIGN CONSIDERATIONS

The system is designed for a 15 kVA motor drive operating on a 3 phase, 208 V, 60 Hz ac mains. A common 2 kHz switching frequency is used, suitable for IGBT switch technology. The rectifier input filter is designed to yield a line current THD of less than 5% in a conventional design with a break frequency of 347 Hz. The dc bus filter inductor yields a dc bus voltage THD of less than 5% [4]. The NN parameters are the result of a compromise between computational time, convergence and accuracy [6]. The design characteristics are summarized in Table 5.1.

5.4.2 STEADY STATE

The steady state performance of the PWM rectifier with NN controller is shown in Fig. 5.8, with rectifier input current, motor current and input / output terminal voltages. The controller provides good tracking of the reference waveforms and unity power operation is achieved, the output inverter operates with a fixed modulation index of 0.91. The frequency spectra of the input / output waveforms are depicted in Fig. 5.9 and medium switching frequency operation is achieved (1980 Hz) for both input and output converters. The effects of dc link current regulation on the input current total harmonic distortion and on the input displacement factor (IDF) are shown in Fig. 5.10. The cases of a rectifier with off - line SPWM pattern and an on - line controlled PWM rectifier without the phase shifter are included. It is seen that the IDF is always near unity for the proposed scheme but is very low for the off - line pattern (compare curves i and iii in Fig. 5.10). The effect of the phase shift introduced in the modulating signal is not significant for low dc link currents, it results in a deviation from unity at higher current demands (curve ii in Fig. 5.10). The THD of the input line currents for these three schemes are also shown in Fig. 5.11. The proposed system always obtains unity power factor within the limitations of the power circuit, which is dictated by the input filter.

5.4.3 DYNAMIC BEHAVIOR

The response of the system is studied under various transient conditions. The output dc current is increased by 50%. The system responds, Fig. 5.12, within a quarter of a cycle (4 ms). The proposed rectifier therefore exhibits excellent transient response. This is confirmed by the 200 rad/s bandwidth, Fig. 5.13 corresponding to a response time

of about 5 ms. This compares favorably with a high performance analog implementation. There are no low frequency oscillations in the currents due to the closed loop control. Power factor remains near unity, even under transient conditions.

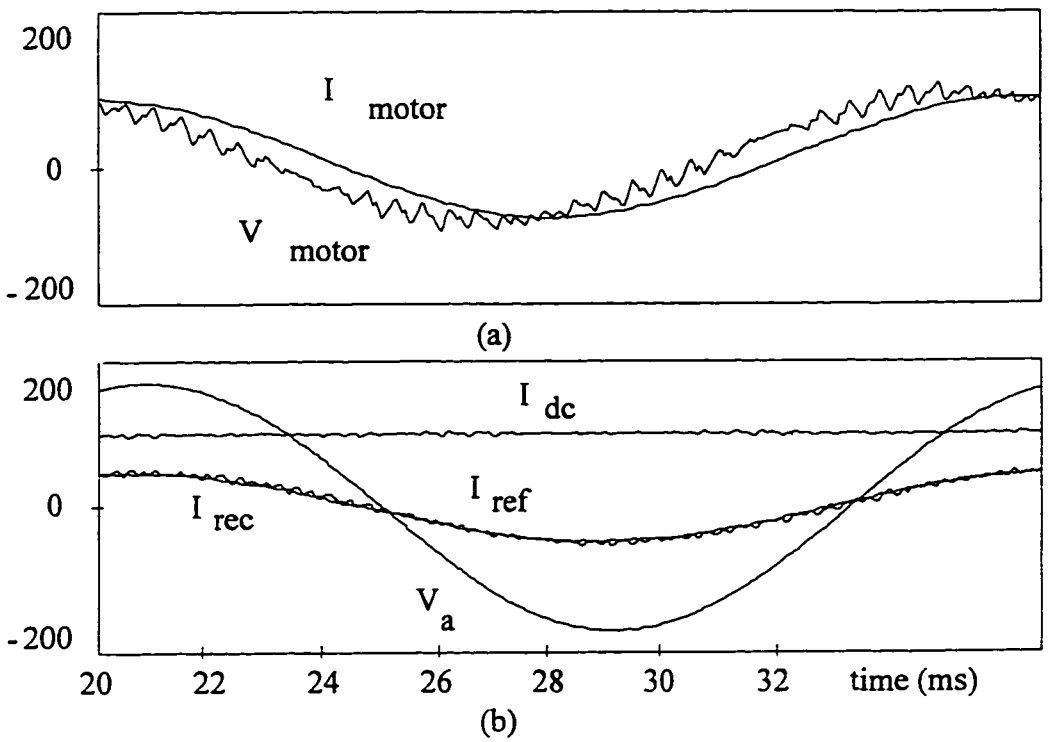


Fig. 5.8 Steady state performance, (a) Motor current, I_{motor} (A), motor voltage V_{motor} (V) , (b) DC link current I_{dc} (A), input line voltage V_a (V), reference current I_{ref} (A) and rectifier input current I_{rec} (A).

TABLE 5.1

DESIGN PARAMETERS

Rated Inputs	Line voltage, V_l	208 V
	Line current, I_l	69.4 A
RECTIFIER	dc inductance, L_{dc}	5 mH
	Input filter capacitance, C_f	300 uF
	Input filter inductance, L_f	0.7 mH
	Input filter damping resistance, R_f	0.1 Ω
	Input line resistance, R_l	0.05 Ω
	Switching frequency, F_{sw}	1980 Hz
INVERTER MOTOR	Inductance, L	1.2 mH
	Capacitance, C	200 uF
	Resistance, R	0.95 Ω
	Switching frequency, F_{sw}	1980 Hz
CONTROLLER	PI gain, k	0.25
	PI time constant, τ	0.001
	number of hidden layers	1
	number of neurons in hidden layer	18

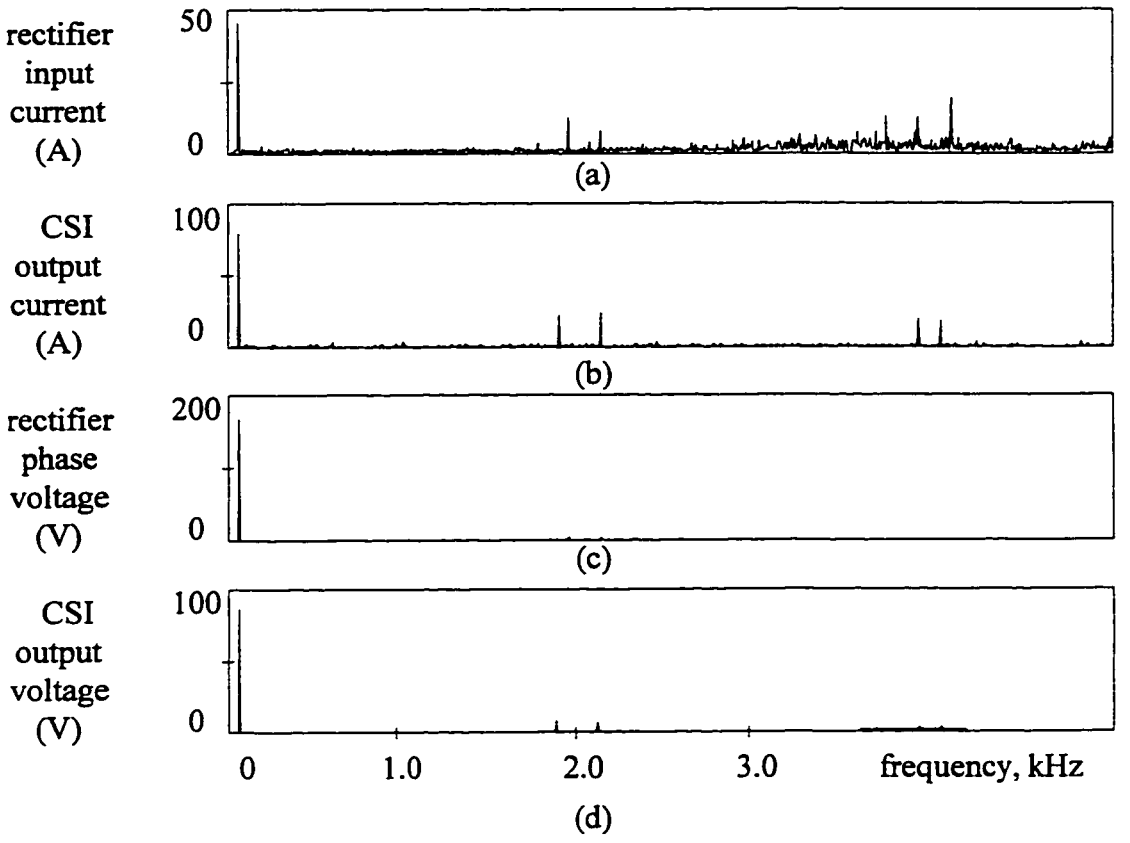


Fig. 5.9 Frequency spectra, a) Rectifier input current, b) CSI output current, c) Input rectifier phase voltage, d) CSI output phase voltage.

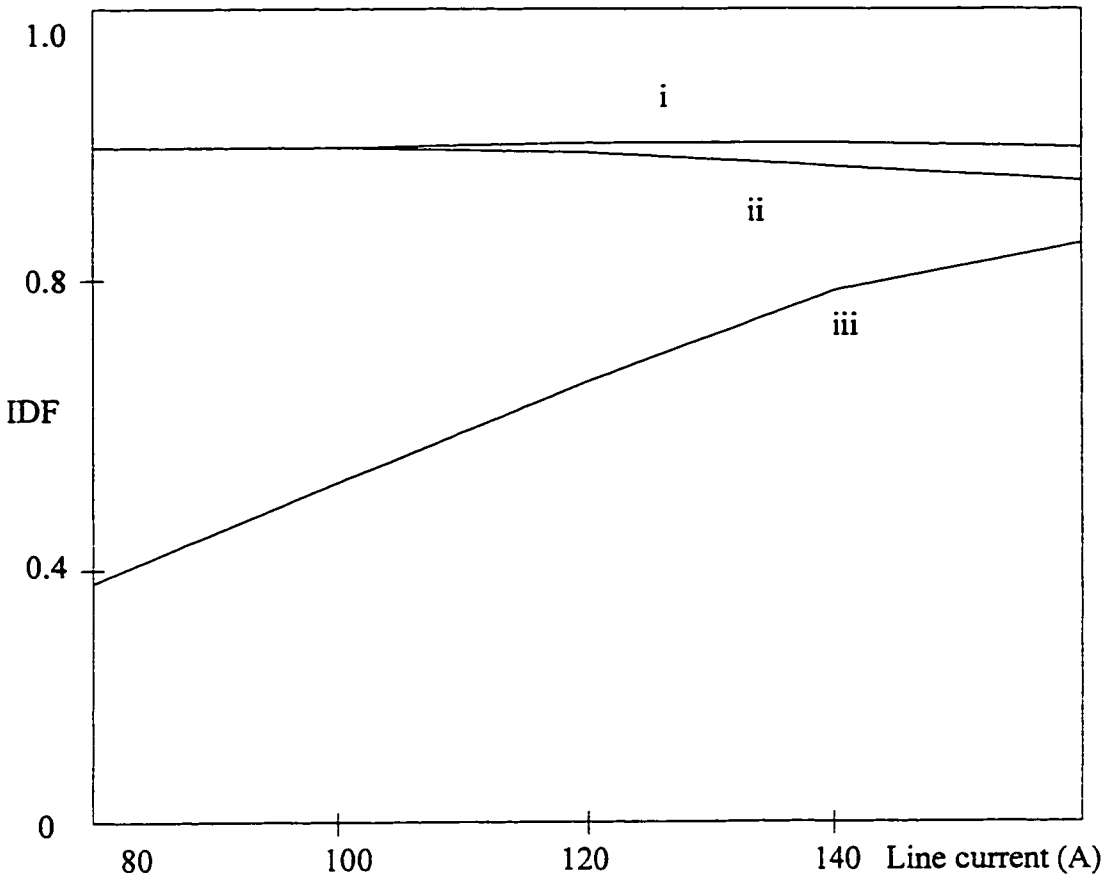


Fig. 5.10 Input displacement factor, i) NN with phase shift, ii) NN without phase shift, iii) PWM rectifier with off line pattern.

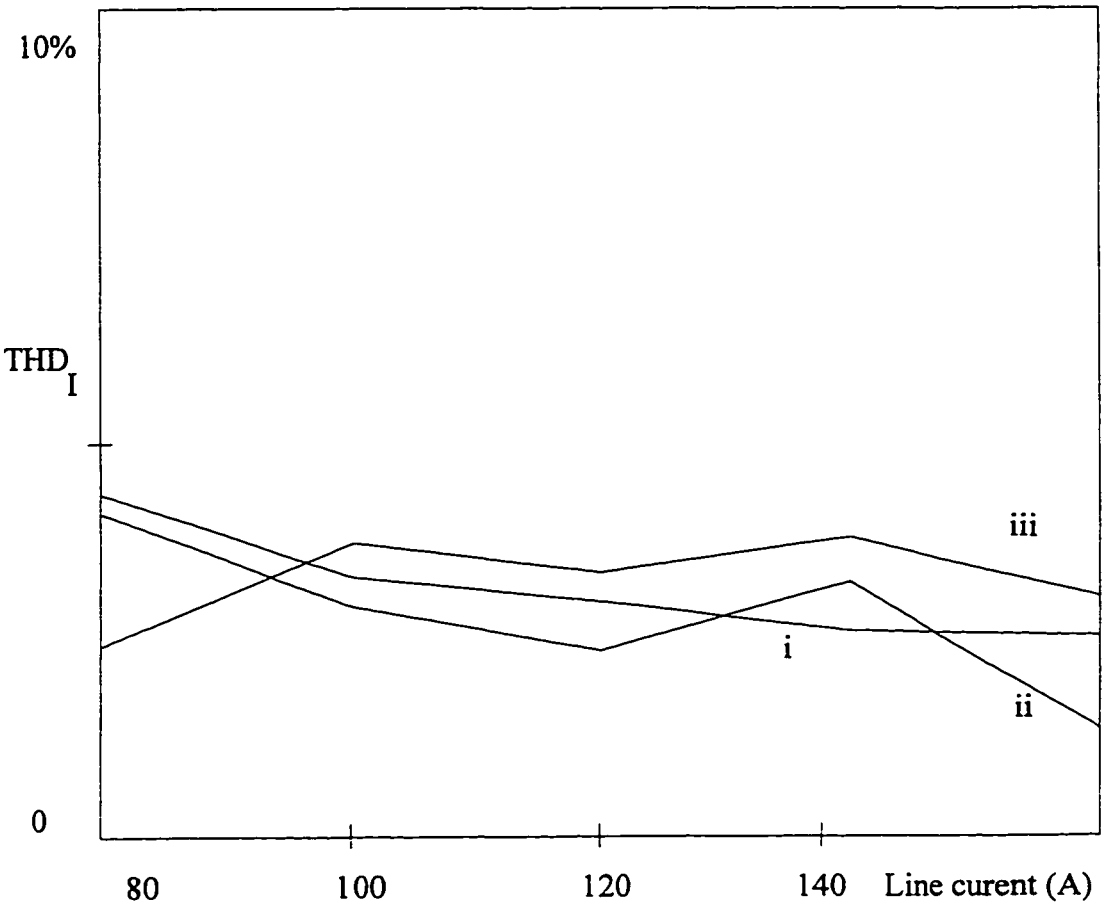


Fig. 5.11 Total harmonic distortion of input line current, i) NN with phase shift, ii) NN without phase shift, iii) PWM rectifier with off line pattern.

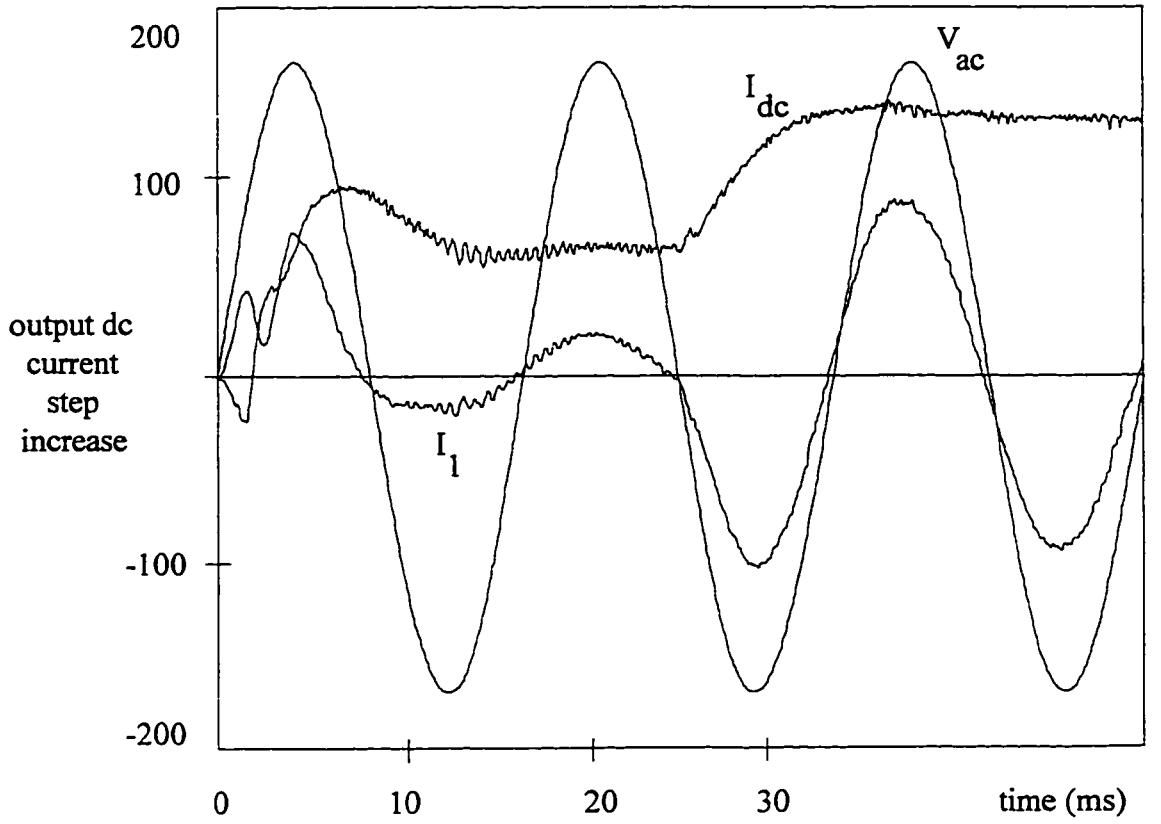


Fig. 5.12 Transient response of the system to a step increase of output dc current of 50 %, dc bus current I_{dc}(A), input phase voltage V_{ac}(V), line current I_l (A).

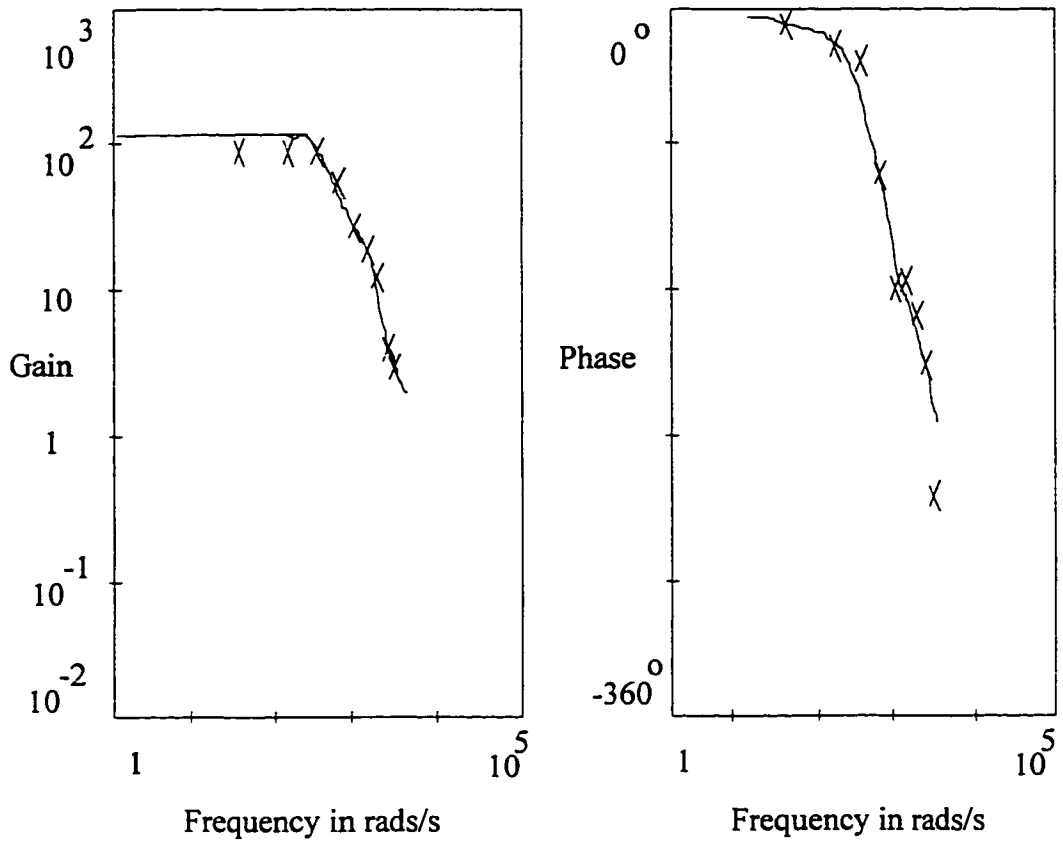


Fig. 5.13 Closed loop frequency and phase response of the rectifier dc current loop system.

5.5 CONCLUSIONS

An on-line Neural Network was proposed to waveshape the input line currents of a PWM rectifier and provide unity power factor operation. Theoretical considerations were verified through computer implementation. Low frequency resonance of the input filter is effectively damped through closed loop control. Excellent steady state waveforms and a fast response to transient conditions are obtained, while low levels of harmonic current are injected into the ac mains. The Neural Network results in a robust system insensitive to load / parameter variations.

CHAPTER 6

SUMMARY AND CONCLUSIONS

6.1 SUMMARY OF THE THESIS

Power converter control aspects and intelligent regulator schemes were reviewed. Chapter 2 outlined the overall concept of Neural Networks. The basics of Backpropagation was discussed with emphasis on learning algorithms and weight updating techniques. Performance evaluation of the various training algorithms were discussed. Chapter 3 described the application of NN regulators to the control of dc to dc buck converters. Here, the purpose is to show that the NN controller is a viable alternative to the traditional PI controller. Chapter 4 demonstrated the implementation of a NN controller for voltage controlled PWM rectifiers. The on line NN controller waveshaped the input line currents forcing unity power factor operation and introducing damping of the low frequency resonance of the input filter. The proposed controller insensitivity to load and parameter variations was investigated. In Chapter 5, the NN controller was applied to current controlled PWM rectifiers. The task of the NN controller was to waveshape the input line currents while attempting to achieve unity factor operation. Appendix A discusses a possible experimental setup to test the behavior of the NN regulator. The setup requires a TMS320C30 DSP module a PC and acquisition and interface hardware. The hardware system and the software flow chart are outlined.

6.2 CONCLUSIONS

A control system that treats every distinct operating point or situation as a novel one has limited performance capabilities. Whereas a system that correlates past experiences with past situations, and that can recall and exploit those past experiences is capable of learning. Learning control schemes operate by optimizing over a relatively large set of parameters to construct a mapping that captures the plant's peculiarities throughout the desired operating range.

The NN controllers used are composed of three layers an input, a hidden and an output layer. Each layer has its own specific purpose. The input layer receives its information from the normalized input vector, the hidden layer sees these weighted inputs and processes them based on an activation function. These weighted outputs are then sent to the output layer to pass through a similar activation function to become the output of the NN.

The supervised learning models which are introduced in Chapter 2 have the capability to adapt to system changes and create their own knowledge base about system dynamics. Supervised control is employed to ensure that the NN is trained on a data base that contains the correct control signals. When the NN controller is to be trained to accurately represent or replace a system, the input data content and supervision of its training are essential. The target output vector informs the NN controller that it is on track or that it is not. The target vector is chosen to provide system stability. The NN uses this vector to make the necessary changes to its parameters so that it may accurately track. The NN controller attains its degree of knowledge primarily due to the amount of

supervision during training. This implies that a richer data set will permit a more complete training.

In this thesis, the NN controller is applied to various power converter topologies and found to be a viable alternative to the traditional controllers particularly of the PI type. The input training vector is based on an entire range of operating points of the Buck converter. The NN controller is trained from this input vector to reproduce the performance of a PI controller. It is found that the number of neurons necessary in the hidden layer, for the NN controller varies from 2 to 18, as the system complexity increases. As system complexity increases, the amount of information required in the input vector to accurately represent the system also increases. The NN controller was found to provide excellent steady state dynamics as well as fast response to transients to varying loads.

It is demonstrated in Chapter 4 and in Chapter 5 that the NN controller has the ability to generate appropriate control signals for the converter. Low frequency resonance of the input filter is effectively damped. It was able to maintain near unity power factor and provide excellent transient response.

Although the NN controller seems, at first glance to be a versatile and possibly ideal controller for most applications, its practical implementation is quite involved. It requires complex algorithms that must be resolved quickly. This requires the use of heavy computational engines such as DSPs. The expense of a DSP system and the amount of programming necessary to produce a complete system may not be warranted for relatively simple systems. However the combination of the desirable properties of traditional control techniques and NN show great promise as to the direction of intelligent control.

6.3 SUGGESTIONS FOR FUTURE WORK

The use of the proposed NN regulator topology could be implemented using the proposed practical implementation discussed in appendix A. If more than one variable has to be controlled, the implementation of a parallel control system may be examined.

REFERENCES

[1] FW-Chuang Chen, "Back Propagation Neural Networks for Nonlinear Self Training Adaptive Control", *Control Systems Magazine.*, April 1990.

[2] Ali Rezgui and Nazil Tepedentiogln, "The Effect of the Slope Activation Function on the Back-propagation Algorithm", *IJCNN*, Washington D.C., 1990.

[3] Ali A. Minai and Ronald D. Williams, "Acceleration of Back-Propagation Through Learning and Momentum Adaptation", *IJCNN*, Washington D.C., 1990.

[4] Youngjik Lee, Sang-Hoon On, and Myung Won Kim, "The Effect of the Initial Weights on Premature Saturation in Back-Propagation Learning", *IJCNN*, Seattle, 1991.

[5] V. K. Sood, N. Kandil, R. V. Patel, K. Khorasani, "Comparative Evaluation of Neural Network Based and PI Current Controllers for HVDC Transmission", *PESC Conf. Rec.*, pp.553-560, 1992.

[6] Heng-Ming Tai, Junli Wang and Kaveh Ashenayi, "A Neural Network -Based Tracking Control System," *IEEE Trans. Ind. Elect.*, vol.39, no. 6, pp. 504-510, Dec. 1992.

[7] Bor-Ren Lin and Richard G. Hoft, "Power Electronics Inverter Control with Neural Networks", *APEC Conf. Rec.*, pp. 128-134, 1993.

[8] Toshio Fukuda, Takanori Shibata, Masatoshi Tokita, and Toyokazu Mitsuoka, "Neuromorphic Control: Adaptation and Learning", *IEEE Trans. Ind. Elect.*, vol.39,no. 6, pp. 497-503, Dec. 1992.

[9] H. Nguyen and B.Widrow, "Neural Networks for Self-Learning Control Systems",*Control Systems Magazine*, pp.18-23, April 1990.

[10] Julio Tanomaru and Sigeru Omatu, "Process Control by On-Line Trained Neural Controllers", *IEEE Trans. Ind. Elect.*, vol.39,no. 6, pp. 511-521, Dec. 1992.

[11] L. Malesani and P. Tenti, "Three-phase ac/dc PWM converter with sinusoidal ac currents and minimum filter requirements," *IEEE Trans. on Ind. App.*, Vol. IA-23, No. 1, pp. 71-77, Jan./Feb. 1987.

[12] P. Enjeti, P. D. Ziogas and J. F. Lindsay, "A current-source PWM inverter with instantaneous current control capability," *IEEE Trans. on Ind. App.*, Vol. 27, No. 3, pp. 582-588,May/June 1991.

[13] X. Wang and B. T. Ooi, "Real-time multi-DSP control of three phase current source unity power factor PWM rectifier," *Conf. Rec. IEEE-PESC*, pp. 1376-1383, 1992.

[14] Y. Sato and T. Kataoka, "State feedback control of current type PWM ac to dc converters," *Conf. Rec. IEEE-IAS*, pp. 840-846, 1991.

[15] N. R. Zargari and G. Joós, "A current-controlled current source type unity power factor PWM rectifier," *Conf. Rec. IEEE-IAS*, pp. 793-798, 1993.

[16] Bor-Ren Lin and Richard G. Hoft, "Power Electronics Converter Control based on Neural Network and Fuzzy Logic Methods," *PESC Conf. Rec.*, pp. 900-906, 1993.

[17] John Bates, Malik E. Elbuluk, Donald S. Springer, "Neural Networks Control of a Chopper-Fed DC Motor", *PESC Conf. Rec.*, pp.893-899, 1993.

[18] Allan Insleay and G. Joos, "Neural Network based approach to the regulation of dc/dc buck converters", *Conf. Rec. CEEE*, 1993.

[19] Allan Insleay, N. R. Zargari and G. Joos, "A Neural Network controlled unity power factor three phase PWM rectifier", *Conf. Rec. IEEE-PESC*, pp. 577-582, 1994.

[20] James A. Freeman and David M. Skapura, "Neural Networks Algorithms, Applications, and Programming Techniques", Addison-Wesley Publishing Company,1991.

[21] Branks Soucek and IRIS Group, "Neural and Intelligent Systems Integration : Fifth and Sixth Generation Integrated Reasoning Information Systems", John Wiley and Sons,1991.

[22] Allan Insleay, N. R. Zargari and G. Joos, "A Neural Network controlled unity power factor three phase current source PWM front end rectifier for adjustable speed drives", *Conf. Rec. IEE-PEVD*, pp. 371-376, 1994.

[23] K. S. Narendra, K. Parthasarathy, "Identification and control of dynamic system using neural networks", *EEE-Transactions on Neural Networks* Vol. 1, No. 1, pp. 4-27, March 1990.

[24] K. S. Narendra, S. Mukhopadhyay, "Adaptive control oc nonlinear multivariable systems using neural networks", *Neural Networks* Vol. 7, No. 5, pp. 737-752.

[25] M. Chow, R. N. Sharpe and J. C. Hung, "On the application and design of artificial neural networks for motor fault detection - part I and II", *IEEE-Transactions on Ind. Elec. Vol. 40, No.2*, pp. 181-196, April 1993.

[26] Hau-Chuen Chan, K. T. Chau, and C. C. Chan, "A Neural Network Controller for Switching Power Converters", *PESC*, 1993.

[27] M. Boost and P. D. Ziogas, "State of the art PWM techniques: A critical evaluation", *IEEE-Transactions on Ind. App.* Vol. 24, No.2, pp. 271-280, March/April 1988.

[28] P. D. Ziogas, Y. G. Kang and Stefanovic, "PWM control techniques for rectifier filter minimization", *IEEE-Transactions on Ind. App.* Vol. IA-21, No.5, pp. 1206-1213, Sept/Oct 1985.

[29] Panos J. Antsaklis, "Neural Networks in Control Systems", *IEEE-Control Systems Magazine*. Vol. 12, No.2, pp. 8-11, April 1992

[30] Richard S. Sutton, Andrew G. Barto and Ronald J. Williams, "Reinforcement Learning is Direct Adaptive Optimal Control", *IEEE-Control Systems Magazine*. Vol. 12, No.2, pp. 11-19, April 1992

[31] Michael A. Sartori and Panos J. Antsaklis, "Implementations of Learning Control Systems Using Neural Networks", *IEEE-Control Systems Magazine*. Vol. 12, No.2, pp. 49-57, April 1992

[32] S. Reynold Chu, Rahmat Shoureshi and Manoel Tenorio, "Neural Networks for System Identification", *IEEE-Control Systems Magazine*. Vol. 10, No.3, pp. 31-36, April 1990

[33] Derrick Nguyen and Bernard Widrow, "Neural Networks for Self Learning Control Systems", *IEEE-Control Systems Magazine*. Vol. 10, No.3, pp. 18-23, April 1990

[34] K. H. Gurubasavaraj. "Implementation of a Self Tuning Controller Using Digital Signal Processor Chips", *IEEE-Control Systems Magazine.* Vol. 9, No.4, pp. 38-43, June 1989

[35] M. Saad, P. Bigras, L. A. Dessaint, K. Al-Haddad, "Adaptive Robot Control using Neural Networks", *IEEE-Transactions on Industrial Electronics.* Vol. 41, No.2, pp. 173-81, April 1994

[36] M. Saad, P. Bigras, L. A. Dessaint, K. Al-Haddad, "Adaptive versus Neural Adaptive control: Application to Robotics", *International Journal of Adaptive Control and Signal Processing.* Vol. 8, No.3, pp. 223-36, May-June 1994

[37] Matlab reference guide, Math Works Inc. 1992.

[38] Matlab Neural toolbox, Math Works Inc. 1992.

[39] Microsoft Quick Basic reference guide, Microsoft Inc. 1990.

[40] Microsoft Quick Basic user's guide, Microsoft Inc. 1990.

APPENDIX A

PROPOSED PRACTICAL IMPLEMENTATION

A.2 IMPLEMENTATION DESCRIPTION

The Texas Instruments TMS320C30 DSP evaluation module was selected to implement the Neural Network controller shown in Fig. A.1. The TMS320C30 is a 32-bit digital signal processor having four levels of pipe lining and capable of performing floating point, integer and logical operations. The TMS320C30 has 2K words (32-bit) of on-chip memory (4K words of ROM in the microcomputer mode) and a total addressable range of 16 million words (32-bits) of memory containing program, data, and input/output space. Separate program, data, and direct memory access (dma) busses enable the TMS320C30 to perform concurrent read and write, plrogram fetches, and DMA operations. The TMS320C30 has a 60-nS instruction cycle time, with most instructions requiring only a single cycle. Hence it can execute 16.6 million instructions per second. Furthermore, because many of the instructions can be performed in parallel, such as load with store and multiply with add, the TMS320C30 effectively can execute up to 33.3 million instructions per second.

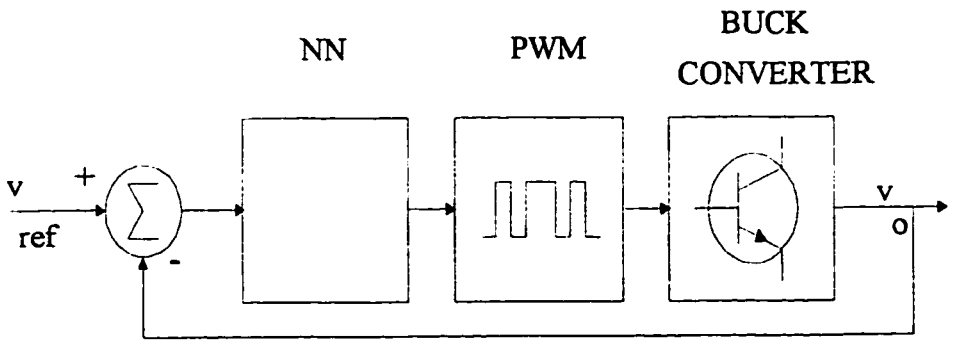


Fig. A.1 System block diagram.

The basic elements of the block diagram of Fig. A.1 are the NN controller and the dc to dc converter. The experimental setup requires many components. Figure A.2 shows

that in order to effectively regulate the dc to dc buck converter with a NN controller it is necessary to have a pc, a DSP board, an analog interface board and the power system itself. The purpose of the pc is to provide the end user with and interface from which control information can be entered and displayed, as well as a host for the TMS320C30. The analog to digital interface board is used to sample the data and to provide the correct signal levels for the TMS320C30.

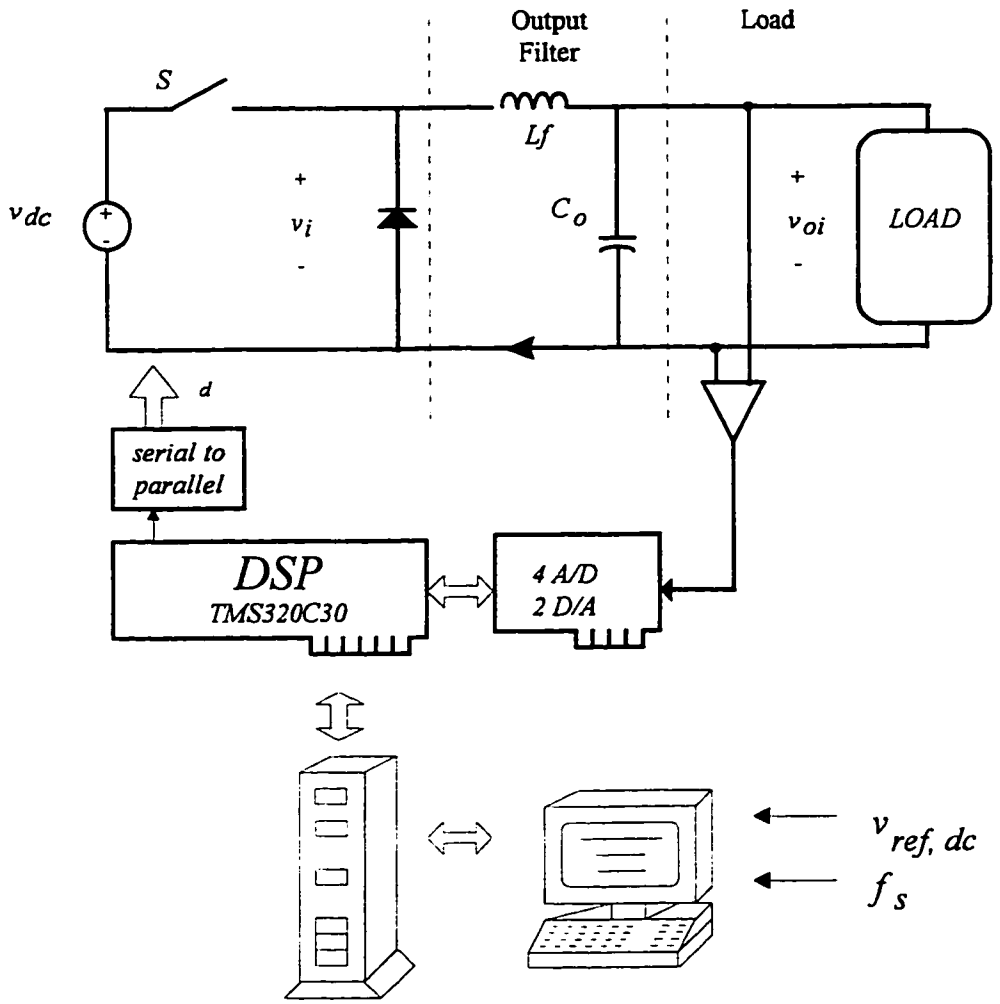


Fig. A.2 The complete system including all necessary hardware and DSP interface.

The output voltage is sampled, and fed to the analog to digital interface board which is then used to regulate the output. Thus the power section is controlled by the duty cycle parameter *d*, which is generated within the DSP. The flow chart of Fig. A.3 details the user interface. The user inputs the reference voltage and the operating

frequency through the pc interface to the DSP common memory locations. At this point the system is setup and ready to go. The user will observe the parameters changing on the computer display. The flow chart of Fig. A.4 is the watch dog of the over sofware system. It ensures that the proper variables have been initialized and enables the timer systems. It then checks to see if any control parameters have been changed and responds accordingly.

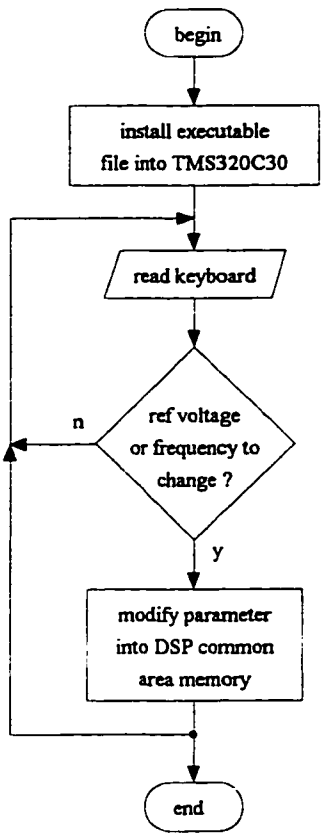


Fig. A.3 User interface flow chart

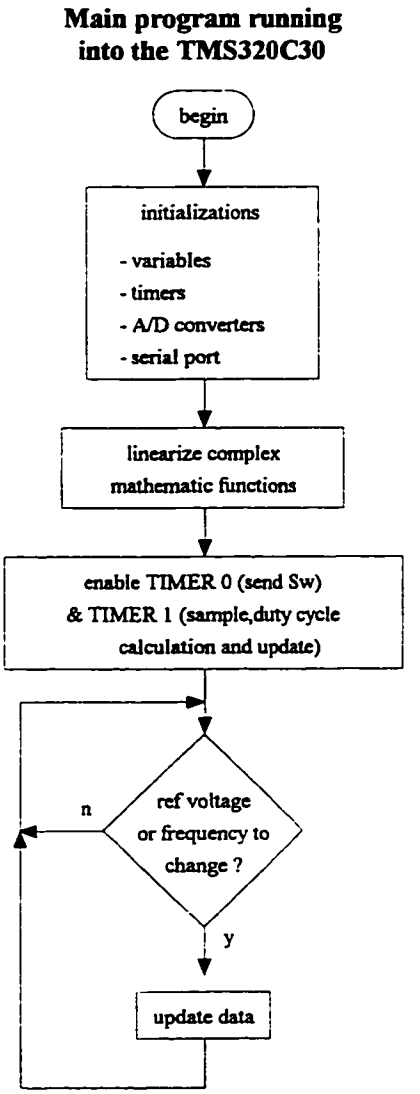


Fig. A.4 DSP initialization and user interface protocol flow chart

The heart of the software control lies in the timer systems. They are set up in such a way that timers 0 and 1 operate independently to control the operation of the dc to dc

converter. Timer 0 has the relatively simple task of controlling the output of the duty cycle d, to the serial port while timer 1 takes the rest. Timer 1 must sample the output dc voltage, compute the error signal between it and the reference voltage, and compute the duty cycle d and store it in a memory location. The error signal is fed into the NN controller. It is here that the bulk of the computational time is spent. The NN generates the duty cycle d, updates the weights and tests to ensure that the overall error is within the desired boundary.

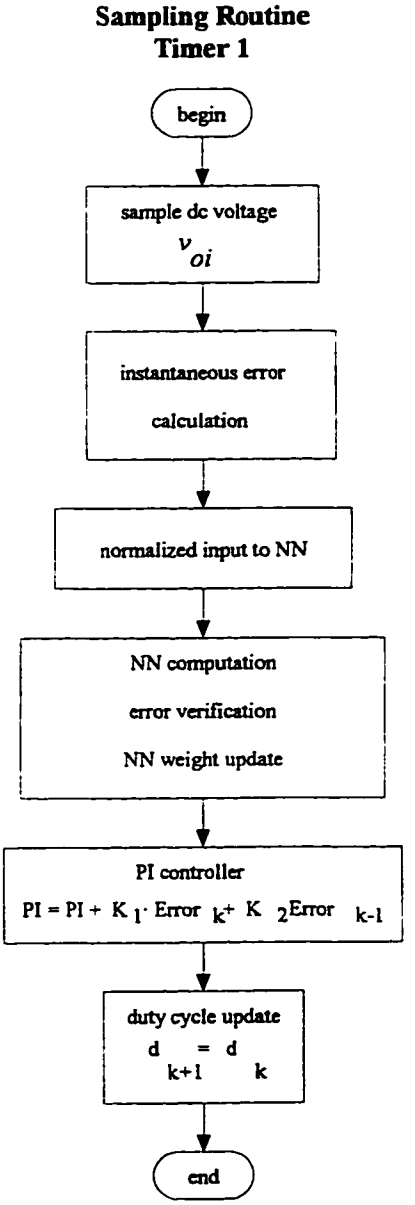


Fig. A.5 Sample and NN routine flow chart

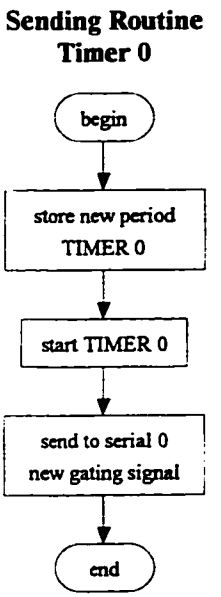


Fig. A.6 Gating signal sending flow chart routine

APPENDIX B

MICROSOFT QUICK BASIC IMPLEMENTATION

B.1 Software Implementation of NN Buck Regulator

The following appendices contain example source code for the implementation of the NN Buck Regulator. Section B.2 shows the system while it is in the learning stage. Section B.3 shows the system in recall mode.

B.2 Microsoft Quick Basic Implementation Training Mode

```
DECLARE SUB tri (t!, y!)
DECLARE SUB PARAMETERS ()

NN PARAMETERS; NUMNRONS = 15

DIM THETAH(NUMNRONS), WO(NUMNRONS), WH(NUMNRONS), SLOPEH(NUMNRONS)
DIM DDELTAH(NUMNRONS), NETH(NUMNRONS), DSH(NUMNRONS), DELTAH(NUMNRONS)
DIM DELTASLOPEH(NUMNRONS), ZIP(NUMNRONS), SLOPEHOLD(NUMNRONS)

//BIAS AND WEIGHT DEFINITIONS

FOR J = 1 TO NUMNRONS;  THETAH(J) = (2 * RND - 1) * .5; WH(J) = (2 * RND - 1) * .5;
 WO(J) = (2 * RND - 1) * .5;  SLOPEH(J) = RND * .1;
NEXT J

THETAO = (2 * RND - 1) * .5; SLOPEO = RND * .1; NETO = 0; ETA = .02; BETA = .02; RHO = .02;
Probe$ = "Y";
X(1) = 0; X(2) = 0; DX(1) = 0; DX(2) = 0

OPEN "BASIC.DAT" FOR OUTPUT AS #1

PRINT #1, 121228; PRINT #1, "TITLE"; PRINT #1, "IL", "VC", "tri", "D", "E", "NETO"; PRINT #1;

//SET UP THE INITIAL CONDTIONS FOR EACH PARAMETER
KP = 1; KI = 888; ES = 100; R = 10; LF = 24 * .001; C = 10.55 * .000001; FS = 1000
VC = 0; VREF = 40; UOLD = 0; EOLD = 0; stp = .00001

//SUB TO INITIALIZE THE RUNGE KUTTA FUNCTION
CALL PARAMETERS
FOR L = 1 TO 3000; t = stp * L; TIME = t; E = VREF - VC;
GOSUB PI; CALL tri(t, y);
IF U > y THEN; D = 1; ELSE; D = 0; END IF;
GOSUB RKQ; IL = X(2); VC = X(1);
PRINT #1, TIME; "  "; IL; "  "; VC; "  "; y; "  "; D; "  "; E; "  "; NETO
EOLD = E;
NEXT L;
```

```
FOR J = 1 TO 10; LMS = 1; WHILE LMS > .001#; L = L + 1;
t = stp * L; TIME = t; E = VREF - VC;
GOSUB PI; CALL tri(t, y);
IF U > y THEN; D = 1; ELSE; D = 0; END IF;
GOSUB RKQ;
IL = X(2); VC = X(1);
PRINT #1, TIME; "  "; IL; "  "; VC; "  "; y; "  "; D; "  "; E; "  "; NETO
EOLD = E; ENN = 1;
WHILE ENN > .001#;
GOSUB NN; ENN = (VC / ES) - NETO;
GOSUB ERRCALC; GOSUB WEIGHTADJUST; GOSUB SLOPEADJUST; WEND;
LMS = ENN * ENN / 2; WEND; NEXT J;
CLOSE #1;
IF Probe$ = "Y" THEN;
    CLS : LOCATE 10, 10: PRINT "Executing Filter ..."
    SHELL "PSPICEDV.EXE BASIC.DAT"
    CLS : LOCATE 10, 10: PRINT "Calling Probe...   "
    SHELL "PROBE BASIC.TXT"
END IF; END;
```

B.3 Microsoft Quick Basic Implementation Recall Mode

```
//RECALL THE WEIGHTS AND TEST THE PERFORMANCE OF THE ON LINE NEURAL NET
DECLARE SUB tri (t!, y!); DECLARE SUB PARAMETERS ();
OPEN "BASIC.DAT" FOR OUTPUT AS #1;
OPEN "c:\qb45\dat\W1C.DAT" FOR INPUT AS #2;
OPEN "c:\qb45\dat\W2C.DAT" FOR INPUT AS #3;
OPEN "c:\qb45\dat\B1C.DAT" FOR INPUT AS #4;
OPEN "c:\qb45\dat\B2C.DAT" FOR INPUT AS #5;
PRINT #1, "121228"; PRINT #1, "TITLE"; PRINT #1, "IL", "VC", "TRI", "E"; PRINT #1,;
//SET UP THE NN PARAMETERS
NUMNRONS = 30;
DIM W1(NUMNRONS); W2(NUMNRONS); B1(NUMNRONS); B2(NUMNRONS);
DIM NETH(NUMNRONS); DSH(NUMNRONS);
DIM DELTASLOPEH(NUMNRONS); ZIP(NUMNRONS); SLOPEHOLD(NUMNRONS);
J = 1; DO UNTIL EOF(2); INPUT #2, W1(J); J = J + 1; LOOP;
J = 1; DO UNTIL EOF(3); INPUT #3, W2(J); J = J + 1; LOOP;
J = 1; DO UNTIL EOF(4); INPUT #4, B1(J); J = J + 1; LOOP;
INPUT #5, B2; CLOSE #2; CLOSE #3; CLOSE #4; CLOSE #5;
//SET UP THE INITIAL CONDITIONS
Probe$ = "Y"
//PI CONTROLLER CONDITIONS
KP = 1; KI = 888;
//CIRCUIT PARAMETERS
ES = 100; VREF = 40; VC = 0; VCN = 0; X(1) = 0; X(2) = 0; DX(1) = 0; DX(2) = 0;
R = 10; LF = 24 * .001; C = 10.55 * .000001; FS = 1000; stp = .00001;
t = stp * L; TIME = t; E = VREF - VC;
GOSUB PI; CALL tri(t, y);
IF U > y THEN; D = 1; ELSE; D = 0; END IF;
GOSUB RKQ; IL = X(2); VC = X(1); EOLD = E;
//THIS PART WILL USE THE NET INSTEAD OF THE PI
IF L = 1 THEN; X(1) = 0; X(2) = 0; DX(1) = 0; DX(2) = 0; END IF;
E = (VREF - VC) / ES; EN = E;
```

```
GOSUB NN; U = NETO * ES; CALL tri(t, y);
IF U > y THEN; D = 1; ELSE; D = 0; END IF;
GOSUB RKQ; IL = X(2); VC = X(1); NEXT L;
//ADD A SHORT AS THE LOAD
R = 1; FOR L = 2001 TO 4000; t = stp * L; TIME = t; E = (VREF - VC) / ES; EN = E;
GOSUB NN; U = NETO * ES; CALL tri(t, y);
IF U > y THEN;   D = 1; ELSE; D = 0; END IF;
GOSUB RKQ; IL = X(2); VC = X(1); NEXT L;
//ADD A STEP CHANGE IN THE REF VOLTAGE
R = 10; FOR L = 4001 TO 6000; t = stp * L; TIME = t; E = (VREF - VC) / ES; EN = E;
GOSUB NN; U = NETO * ES; CALL tri(t, y);
IF U > y THEN;   D = 1; ELSE;   D = 0; END IF;
GOSUB RKQ; IL = X(2); VC = X(1);
NEXT L; CLOSE #1; IF Probe$ = "Y" THEN;
     CLS : LOCATE 10, 10: PRINT "Executing Filter ..."
     SHELL "PSPICEDV.EXE BASIC.DAT"
     CLS : LOCATE 10, 10: PRINT "Calling Probe...   "
     SHELL "ps BASIC"
END IF; END;
//SUBROUTINES COMMON TO BOTH PROGRAMS
RKQ:
        FOR Ind% = 1 TO 10; WSC(Ind%) = 0; NEXT Ind%; FOR Jnd% = 1 TO 4
                GOSUB DERIV; FOR Ind% = 1 TO 2; ZZ = A(Jnd%) * (DX(Ind%) - B(Jnd%) *
                WSC(Ind%)); X(Ind%) = X(Ind%) + stp * ZZ; WSC(Ind%) = WSC(Ind%) + 3! * ZZ -
                C(Jnd%) * DX(Ind%); NEXT Ind%; NEXT Jnd%; RETURN;
DERIV:
     DX(1) = X(2) / C - X(1) / (R * C); DX(2) = (ES / LF) * D - X(1) / LF; RETURN;
PI:
    U = UOLD + KP * (E - EOLD) + (KI / 2) * (E + EOLD) * stp; UOLD = U; RETURN;
NN:
        FOR I = 1 TO NUMNRONS; NETH(I) = WH(I) * D + THETAH(I);
        //GET THE LIMITED VALUE FOR THE HIDDEN NODE
        DSH(I) = SLOPEH(I) * NETH(I); NETH(I) = (1 - EXP(-DSH(I))) / (1 + EXP(-DSH(I)));
        //GET THE OUTPUT NODE VALUES
        NETO = NETO + WO(I) * NETH(I); NEXT I; NETO = NETO + THETAO; DSO = SLOPEO *
        NETO;  NETO = (1 - EXP(-DSO)) / (1 + EXP(-DSO)); RETURN;
ERRCALC:
        //CALCULATE THE ERROR TERMS
        DELTAO = ENN * (1 - NETO * NETO) * SLOPEO / 2;
                FOR I = 1 TO NUMNRONS;
                        DDELTAH(I) = .5 * SLOPEH(I) * (1 - NETH(I) * NETH(I));
                        DELTAH(I) = DDELTAH(I) * WO(I) * ENN; NEXT I; RETURN;
WEIGHTADJUST:
        //UPDATE THE HIDDEN AND THE OUTPUT LAYER WEIGHTS
                FOR I = 1 TO NUMNRONS; WO(I) = WO(I) + ETA * DELTAO * NETH(I);
                        WH(I) = WH(I) + ETA * DELTAH(I) * D; NEXT I; RETURN;
SLOPEADJUST:
        DELTASLOPEO = (SLOPEO / 2) * ENN * (1 - NETO * NETO); FOR I = 1 TO NUMNRONS;
                        ZIP(I) = ENN * WO(I); DELTASLOPEH(I) = ZIP(I) * (SLOPEH(I) / 2) * (1 -
        NETH(I) * NETH(I));  NEXT I; IF DELTASLOPEO <= .02 THEN;
        DELTASLOPEO = .002; END IF;  FOR I = 1 TO NUMNRONS; IF DELTASLOPEH(I) <= .02
        THEN; DELTASLOPEH(I) = .002;END IF; NEXT I; SLOPEO = SLOPEO + BETA *
        DELTASLOPEO + RHO * (SLOPEO - SLOPEOLD); SLOPEOLD = SLOPEO; FOR I = 1 TO
        NUMNRONS;
```

```
        SLOPEH(I) = SLOPEH(I)+BETA* DELTASLOPEH(I)+RHO* (SLOPEH(I) - SLOPEHOLD(I));
        SLOPEHOLD(I) = SLOPEH(I); NEXT I; RETURN;
SUB PARAMETERS
        SHARED A(), B(), C(), stp; A(1) = .5: A(2) = .29289322#: A(3) = 1.70710678#: A(4) =
        .1666667#; B(1) = 2!: B(2) = 1!: B(3) = 1!: B(4) = 2!; C(1) = .5: C(2) = .29289322#: C(3) =
        1.7071678#: C(4) = .5; stp = .00001; END SUB;
SUB tri (t, y)
        SHARED FS; TS = 1 / FS; DIV1 = INT(t / TS); tt = t - (TS * DIV1); SELECT CASE tt;
                CASE 0 TO 1 / (2 * FS); y = (20 * 2 * FS * tt) - 10; CASE 1 / (2 * FS) TO 1 / FS;
                        y = -(20 * 2 * FS * tt) + 30; END SELECT; END SUB;
```

APPENDIX C

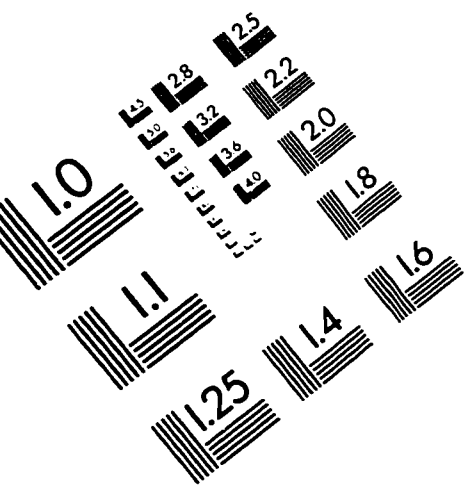
CONTROLLED RECTIFIER SYSTEM MATRIX

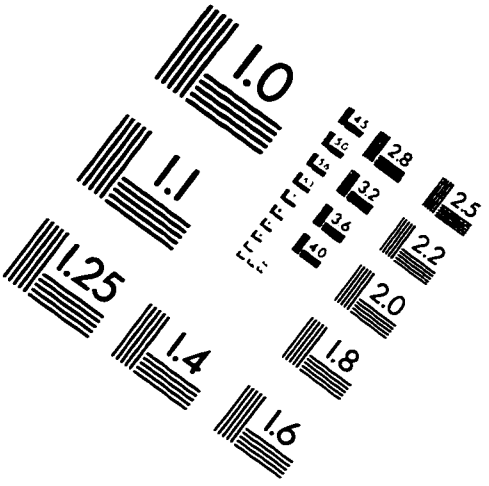
The matrix A in the differential equation describing the controlled rectifier system shown in Fig. 4.2. is given by:

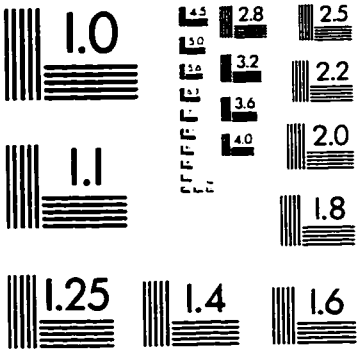
$$A = \begin{bmatrix} -\frac{(R_l + R_f)}{L} & 0 & 0 & 1 & 0 & 0 & \frac{(R_f \cdot S_1)}{L} & 0 \\ 0 & -\frac{(R_l + R_f)}{L} & 0 & 0 & 1 & 0 & \frac{(R_f \cdot S_2)}{L} & 0 \\ 0 & 0 & -\frac{(R_l + R_f)}{L} & 0 & 0 & 1 & \frac{(R_f \cdot S_3)}{L} & 0 \\ \frac{1}{C} & 0 & 0 & 0 & 0 & 0 & -\frac{S_1}{C} & 0 \\ 0 & \frac{1}{C} & 0 & 0 & 0 & 0 & -\frac{S_2}{C} & 0 \\ 0 & 0 & \frac{1}{C} & 0 & 0 & 0 & -\frac{S_3}{C} & 0 \\ \frac{(S_1 \cdot R_f)}{L_{dc}} & \frac{(S_2 \cdot R_f)}{L_{dc}} & \frac{(S_2 \cdot R_f)}{L_{dc}} & \frac{S_1}{L_{dc}} & \frac{S_2}{L_{dc}} & \frac{S_3}{L_{dc}} & \frac{-R_f}{L_{dc}} \cdot (S^2_1 + S^2_2 + S^2_3) & -\frac{1}{L_{dc}} \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{C_{dc}} & -\frac{1}{R_{dc} C_{dc}} \end{bmatrix}$$

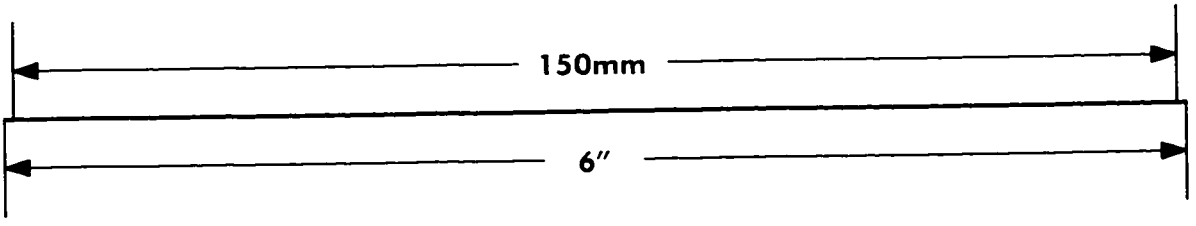
where: R_l = line resistance
R_f = filter resistance in the capacitor branch
L,C = ac side components
L_{dc} ,C_{dc} = dc side components
S_i = line-to-line switching function

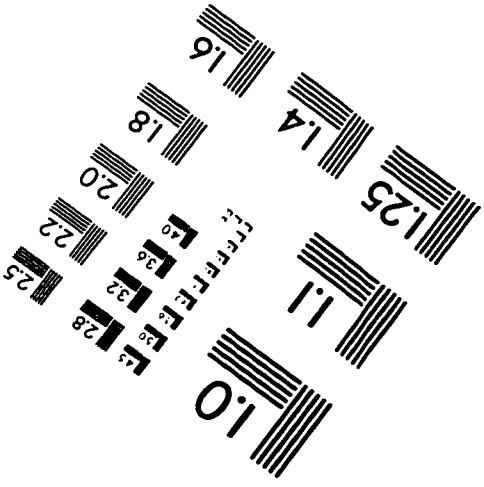
IMAGE EVALUATION
TEST TARGET (QA-3)

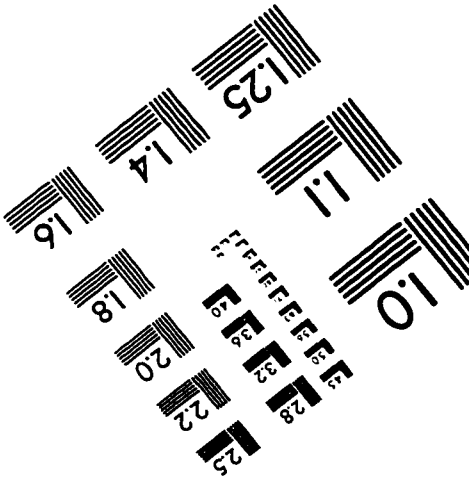












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989