This is to certify that the

thesis entitled

THE MODELING AND MEASUREMENT OF THE RELEASE, PRODUCTION, AND RETENTION OF CLOTH FIBERS IN A TOP-LOADING WASHING MACHINE

presented by

David John Fanson

has been accepted towards fulfillment of the requirements for

Master of Science degree in Mechanical Engineering

Major professor

Date 5/19/89

THE MODELING AND MEASUREMENT OF THE RELEASE, PRODUCTION, AND RETENTION OF CLOTH FIBERS IN A TOP-LOADING WASHING MACHINE

By

David John Fanson

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

1989

ABSTRACT

THE MODELING AND MEASUREMENT OF THE RELEASE, PRODUCTION, AND RETENTION OF CLOTH FIBERS IN A TOP-LOADING WASHING MACHINE

By

David John Fanson

A new lint measurement technique has been developed to measure instantaneous lint concentrations which occur in a top-loading washing machine during agitation. The technique, which is based on the transmittance of light through a water sample, provides quick and accurate measurements. Using information revealed through these measurements, a mathematical model was developed to describe the physics of the processes which cause lint to be suspended. Using this new lint measurement technique and the mathematical model an experimental study was completed. The focus of this study was to determine the effect of agitator operating conditions (stroke length and oscillation frequency) on the suspension of cloth fibers (lint). Knowledge gained from this study verified the lint model and provided the groundwork for several new relationships which pertain to the release, production and retention of lint.

To my wife Wendy and my family

TABLE OF CONTENTS

LIST OF TABLES vi

LIST OF FIGURES vii

NOMENCLATURE ix

CHAPTER 1 INTRODUCTION 1

CHAPTER 2 LINT RELEASE MODEL 4

2.1 Introduction 4

2.2 Lint release model 5

2.3 Reactivated, produced, and retained lint 11

CHAPTER 3 EXPERIMENTAL TECHNIQUES AND PROCEDURE 13

3.1 Introduction 13

3.2 Test apparatus 13

3.3 Standard fabric load 15

3.4 Lint concentration measurement technique 15

3.5 Experimental setup 16

3.6 Calibration of lint concentration measurement technique 19
3.7 Sensitivity levels of lint concentration measurements 20
3.8 Experimental procedures 21
3.9 Parameter estimation 25
3.10 Repeatability 31
3.11 Evaluation of partition coefficient 31

CHAPTER 4 RESULTS 35
4.1 Experimental operating conditions 35
4.2 Evaluation of model parameters 37
4.3 Average angular velocity 39
4.4 Low sensitivity data points 42
4.5 Partition coefficient 45
4.6 Final model 45
4.7 Dimensional analysis 49

CHAPTER 5 CONCLUSIONS 53
5.1 Overview 53
5.2 Conclusions 54
5.3 Suggested research 57

APPENDICES 58
A Program LINT 58
B Experimental data 96

LIST OF TABLES

3.1	Repeatability test without rinse	32
3.2	Repeatability test with rinse	33
4.1	Order of washing runs	38

LIST OF FIGURES

2.1	Lint release model	10
3.1	Washing machine setup	14
3.2	Light Transmittance	16
3.3	Experimental setup	17
3.4	Calibrations for sampling chambers	20
3.5	Sensitivity of lint concentration measurement	22
3.6	Typical data file	26
3.7	Plot of lint concentration versus time	27
3.8	Sensitivity coefficients of model parameters	28
3.9	Typical plot of model fit to data	30
3.10	Repeatability test without rinse	32
3.11	Repeatability test with rinse	33
4.1	Operating points for experimental matrix	36
4.2a-c	Model parameters versus stroke length	40
4.3a-c	Model parameters versus frequency	41
4.4	Operating conditions with velocities	42
4.5a-c	Model parameters versus velocity	43
4.6	Δ K versus measurement sensitivity	44
4.7a-c	Model parameters versus velocity	46
4.8	Alpha versus average angular velocity	47

4.9	Beta versus average angular velocity	47
4.10	Complete lint model	48
4.11	Comparison of lint model to actual data	49
4.12	Fanson linting constant versus velocity	51
4.13	Nondimensional lint concentration versus time	51
4.14	A nondimensional plot of all the data	52

NOMENCLATURE

Symbols	Description	Units
a	Final lint model constant	
A,B,C	Estimation parameters	
C	Lint concentration	mg/L
f	Oscillation frequency	Hz
F	Fanson lint constant	
L	Relative lint concentration	mg/L
K	Lint production rate	mg/L/min
K′	Release rate	1/min
P	Output power	mW
P1	Input power	mW
S	Sensitivity	
t	Time	min
t′	Minimum time for linear model	min
T′	Normalized transmittance	
α	Partition coefficient	
β	Lint ratio	
Δ	Difference	
γ	Volume fraction	
$\bar{\omega}$	Average angular velocity	rad/min
θ	Agitator stroke angle	deg
λ	Calibration constant	mg/L

Subscripts

1,2,3	Final lint model constants
A,B,C	Estimation parameters
d	Drain
f	Fabric
m	Measurement
r	Rinse
t	Total
w	Water
	Dimensionless

Superscripts

a	Reactivated
e	Equilibrium
f	Final
i	Initial
p	Produced

CHAPTER 1

INTRODUCTION

The cleaning of clothes using an automatic washing machine has become quite common in American households over the past three decades. To the consumer, washing clothes consists of placing laundry into the washing machine, adding detergent, selecting a particular type of wash cycle, turning on the machine, and removing the washed clothes to dry. The most important aspect of washing machine performance to the consumer is that the washer removes all visible stains and offensive odors.

Commercial manufacturers have found that removing soil from clothes in the washing machine is relatively easy. All that is needed is water, detergent, and mechanical agitation. The more intense the agitation, the faster and more thoroughly the clothes are cleaned. But also with more intense agitation greater cloth damage occurs. Fabric damage exists in two forms 1) visible tears and 2) lint.

Obviously, visible damage to the cloth must be avoided, but what about lint? To the consumer lint is only objectionable if it can be seen. Visible lint deposits occur when a very large number of lint fibers have accumulated in one location. Because lint is only a concern when very high concentrations exist, engineers have not put a high priority on thoroughly understanding the process of lint suspension. Even though, lint is always present during agitation.

Because the washing process has not undergone any major improvement in recent years, research interest has been allowed to shift from a "make it work" mode to a "why and how does it work" mode. Because of this change funding was made available by Whirlpool Corporation to study lint. Being in a university setting, a thorough and original scientific analysis was allowed to be completed. Before this study engineers have been satisfied with knowing only where lint ends up, and not knowing where, when, or how lint is produced.

The scope of the research presented in this paper is summarized by these three objectives:

1. To develop a theory to describe the suspension of lint using a mathematical model.

2. To develop and implement a technique for the quantitative measurement of the amount of lint suspended in water,

3. To study the effect of stroke length and oscillation frequency on the suspension of lint.

While completing these three objectives, many new and valuable observations have been made about lint suspension. By increasing our understanding of the processes associated with lint, it will become easier to diagnose problems related to linting.

The chapters which follow in this thesis generally follow the objectives listed above. Chapter 2 describes the development of a theoretical lint model which includes the amount of lint suspended in water as a function of time. This model is based on several assumptions which have been theorized or inferred from observations.

The lint model includes parameters describing the lint production, the lint transfer from the fabric surface to the water, and the amount of lint which is initially adhered to the fabric.

Chapter 3 describes the lint measurement technique used to produce the experimental lint concentration profiles. An explanation of the underlying physical principles of the technique (light transmittance) along with a specific example of its implementation in a laboratory environment are included. The functional relationship between amount of light transmitted through a lint-water sample and the mass of lint per unit volume of water is revealed from the calibration of the technique. The chapter also describes the experimental procedure used to obtain the experimental data. Using a parameter estimation technique raw data are reduced to a set of parameters which describe the dynamics of the linting process.

Chapter 4 outlines a sample experiment which was completed in order to explore the potential benefits and problems associated with the new technique. The analysis includes a study of the effect that changes in stroke length and oscillation frequency of the agitator have on the production, the release and the retention of lint.

Finally in Chapter 5 a summary of the most important conclusions and a list of recommendation for future research are presented.

CHAPTER 2

LINT RELEASE MODEL

2.1 INTRODUCTION

A mathematical model has been developed to help in the basic understanding of the suspension of lint during the washing of clothes. The model is used to quantitatively illustrate the origin and residence of lint fibers during the wash cycle. The theory is based on the following conceptual model of the linting process:

Suspended lint fibers occur in the washing machine in two different states: 1) lint is suspended in the water and 2) lint is adhered to the fabric. The lint suspended in the water is produced by two mechanisms, 1) the continuous breakage and subsequent release of fibers due to mechanical agitation and 2) the release of loosely embedded fibers which were either deposited on the fabric by an earlier washing or broken in some way before the washing began (i.e. in the drier or from wear). Fibers not only move from the fabric into the water under agitation, but fibers in the water may adhere again to the fabric and reside there for a certain time. Therefore, the notion of residence time, well known in surface renewal theories, can be applied to describe the dynamics of textile fibers.

2.2 LINT RELEASE MODEL

Several terms which are used in the development of the lint release model are defined:

Equilibrium conditions - conditions under which the mass of fibers leaving the fabric equals the mass of fibers deposited onto the fabric,

Lint concentration - lint mass per unit volume (fabric, water, or total),

Relative lint concentration - lint mass per unit volume pertaining to the total volume (water plus cloth),

Volume fraction - partial volume divided by the total volume,

Partition coefficient - the ratio of suspended lint concentration to attached lint concentration under equilibrium conditions,

Attached lint - lint fibers which are deposited on the fabric,

Suspended lint - lint fibers which are suspended in the water,

Total lint - attached lint plus suspended lint,

Lint production rate - the rate at which total lint is generated, and

Lint release rate - the rate at which attached lint is released from the fabric.

In addition to these definitions the following variables are defined to facilitate the development of the mathematical model:

α - partition coefficient,

β - ratio of suspended lint concentration to total lint concentration under equilibrium conditions,

γ - volume fraction,

C - lint concentration (mg/L),

L - relative lint concentration (mg/L),

K - lint production rate (mg/[L·min]),

K′ - release rate (1/min),

t - time (min).

In addition to these variables, descriptive subscripts and superscripts are implemented. The subscripts are used to describe location or type, for example,

f - fabric,

w - water, and

t - total.

Similarly, the superscripts are used to describe a time or condition, as seen in,

i - initial,

f - final, and

e - equilibrium.

Using the aforementioned definitions and variables the following governing equations emerge:

From the definition of the partition coefficient,

$$\alpha = \frac{C_w^e}{C_f^e} \tag{2.1}$$

and

$$\beta = \frac{C_w^e}{C_t} \tag{2.2}$$

Using the following relation (for all times) between C_t, C_w, and C_f,

$$C_t = \gamma_w C_w + (1 - \gamma_w) C_f \tag{2.3}$$

α and β can also be expressed as,

$$\alpha = \frac{\beta - \beta \gamma_w}{1 - \beta \gamma_w} \tag{2.4}$$

and

$$\beta = \frac{\alpha}{\alpha \gamma_w + 1 - \gamma_w} \tag{2.5}$$

If the rate at which the total concentration increases in time (production rate) is assumed to be a constant, then

$$\frac{dC_t}{dt} = K_t \qquad (2.6)$$

Similarly, it is assumed that the rate at which the lint concentration in the water increases with time (release rate) is proportional to the deficit lint concentration in the water. Thus,

$$\frac{dC_w}{dt} = K' \, (C_w^e - C_w) \qquad (2.7)$$

Note that the rate of change in C_w in Eq. 2.7 is the net result of the lint being released from and redeposited onto the fabric surface.

To solve this system of equations (Eqs. 2.1 - 2.7) and to find the concentration of lint in the water at time t, initial conditions must be given.

We assume that at t = 0 all lint is adhered to the fabric giving the following equations:

$$C_t = C_t^i \quad \text{at} \quad t = 0 \qquad (2.8)$$

and

$$C_w = 0 \quad \text{at} \quad t = 0 \qquad (2.9)$$

The total lint concentration can be found using Eqs. 2.6 and 2.8, namely

$$C_t = C_t^i + K_t \, t \qquad (2.10)$$

Using Eqs. 2.2, 2.6, and 2.10, Eq. 2.7 can be rewritten as follows:

$$\frac{dC_w}{dt} = \beta K' C_t^i + \beta K_t K' t - K' C_w \qquad (2.11)$$

Finally, the solution is

$$C_w = \beta \left(C_t^i - \frac{K_t}{K'} \right) (1 - e^{-K't}) + \beta K_t t \qquad (2.12)$$

As stated in the introduction, the suspension of lint into the water during agitation can be considered as the result of two separate processes 1) resuspension (or reactivation) of old lint and 2) the production of new lint. If we separate the suspended lint concentration from Eq. 2.12 into a reactivation (C_w^a) and production (C_w^p) concentrations, we get,

$$C_w^a = \beta C_t^i (1 - e^{-K't}) \qquad (2.13)$$

and

$$C_w^p = \beta K_t \left(\frac{t - (1 - e^{-K't})}{K'} \right) \qquad (2.14)$$

where

$$C_w = C_w^a + C_w^p \qquad (2.15)$$

From Figure 2.1 in which Eqs. 2.12, 2.13, and 2.14 are plotted using arbitrary values for β, C_t^i, K_t, and K' (0.5, 200, 0.5, and 5, respectively), we can observe qualitatively how each of these processes (reactivation and production) contribute to the suspension of lint fibers during agitation.

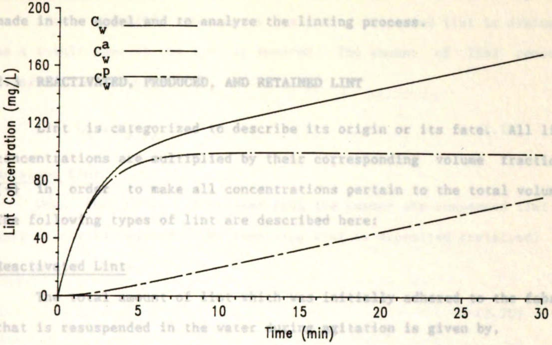


Figure 2.1 Lint release model

As seen in this plot the reactivated lint concentration increases rapidly when the agitation begins but levels out as time increases. This initially rapid increase is due to the large difference between the equilibrium and suspended lint concentrations at the beginning of agitation. As the amount of reactivated lint approaches the initial equilibrium concentration, the reactivation term becomes constant. The production term appears to be proportional to time. At early times the reactivation term dominates the production term. As time increases and the total amount of lint suspended in the water increases, the slope of the plot approaches a constant value. This result is caused by the decreasing lint deficit and eventual dominance of the lint production term.

A series of experiments was carried out to verify the assumptions made in the model and to analyze the linting process.

2.3 REACTIVATED, PRODUCED, AND RETAINED LINT

Lint is categorized to describe its origin or its fate. All lint concentrations are multiplied by their corresponding volume fractions (γ) in order to make all concentrations pertain to the total volume. The following types of lint are described here:

Reactivated Lint

The total amount of lint which was initially adhered to the fabric that is resuspended in the water during agitation is given by,

$$L_w^a = \beta\, C_t^i\, \gamma_w \tag{2.16}$$

Produced Lint

The total amount of lint which is generated during agitation and is suspended in the water is equal to,

$$L_w^p = \beta\, \gamma_w\, K_t\, (t - 1/K') \tag{2.17}$$

Final Lint

The total amount of lint which is suspended in the water at the end of agitation is

$$L_w^f = L_w^a + L_w^p$$

$$= \beta\, \gamma_w\, [C_t^i + K_t\, (t - 1/K')] \tag{2.18}$$

Drained Lint

After agitation the water containing suspended lint is drained. As a result some of the lint is removed. The amount of lint removed (drained) is shown by

$$L_d = C_d \gamma_d \qquad (2.19)$$

Retained Lint

When the water is drained from the washer the suspended lint is only partially removed. The remaining lint is deposited (retained) on the fabric. Thus,

$$L_r = L_w^f - L_d \qquad (2.20)$$

Using these values along with the coefficients from the lint release model a quantitative comparison between washing runs can be made. With this ability, different operating conditions and/or washing setups may be evaluated as to their linting potential.

CHAPTER 3

EXPERIMENTAL TECHNIQUES AND PROCEDURE

3.1 INTRODUCTION

A series of experiments were performed to find the effect of particular operating conditions on the linting process. For these experiments washing runs were completed in a modified top-loading washing machine employing a newly developed technique for the measurement of lint concentrations in water. The lint concentrations recorded during the washing run were then used to solve for the unknown parameters in the mathematical model discussed in Chapter 2. A more complete description of the entire process is found in the following sections.

3.2 TEST APPARATUS

For all experiments a basic top-loading Whirlpool washing machine was used. A number of modifications to the washer were made to provide control over some of the operating conditions. The steel outer housing of the washing machine was removed to allow greater accessibility to the drive mechanism and to the washing tub. The drive mechanism was modified so that the sweep (stroke length) of the agitator was adjustable. The standard drive motor was removed and replaced with a

1/2 horsepower variable speed DC motor. Also, the spin cycle and timer were disconnected causing the washer to run continuously in the wash cycle. In order to simplify the experiments and to concentrate on the washing cycle, all other cycles were disconnected. Since the spin cycle was not in use the washing tub suspension system was no longer needed and therefore, removed. The washer frame was then rigidly mounted to the floor. This configuration allowed experiments to be performed at maximum oscillation frequencies up to just under 3 Hertz with a stroke length range of 0 to 200+ degrees. Washing runs could be specified for any length of time. Figure 3.1 is a front view of the washing machine setup.

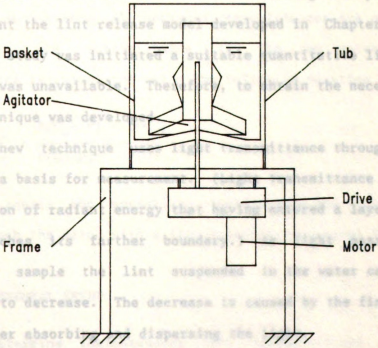


Figure 3.1 Washing machine setup

3.3 STANDARD FABRIC LOAD

The fabric load used in the experiments consisted of 30 pieces of white cotton fabric. The pieces were rectangular and measured 0.5 X 0.8 meters. Two fabric loads were used alternately to permit drying while another experiment was in progress. The sheets in both loads were selected at random from the initial fabric supply provided by Whirlpool Corporation. All of the fabric was then labeled to enable a record to be kept of the washing history of each group.

3.4 LINT CONCENTRATION MEASUREMENT TECHNIQUE

Suspended lint concentration versus washing time plots are needed to implement the lint release model developed in Chapter 2. At the time this study was initiated a suitable quantitative lint measurement technique was unavailable. Therefore, to obtain the necessary raw data a new technique was developed.

The new technique uses light transmittance through a lint-water sample as a basis for measurement. (Light transmittance is defined as the fraction of radiant energy that having entered a layer of absorbing matter reaches its farther boundary.) As light passes through a lint-water sample the lint suspended in the water causes the light intensity to decrease. The decrease is caused by the fine lint fibers in the water absorbing and dispersing the light.

To determine the transmittance a coherent beam of light (a laser) with known power P1 is directed into a volume of lint-water mixture. The power of the light beam leaving the sampling chamber is measured.

This value $P(C_w)$ is a function of the lint concentration of the water. The transmittance is then calculated by dividing $P(C_w)$ by P1. Figure 3.2 illustrates the notion of transmittance.

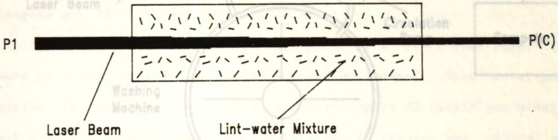


Figure 3.2 Light Transmittance

In order to eliminate the effect of fluctuations in the water supply's transmittance, all measured transmittances are normalized with respect to the transmittance of the tap water. Normalized transmittance is calculated using the following equation

$$T' = \frac{P(C_w)/P1}{P(0)/P1} = \frac{P(C_w)}{P(0)} \qquad (3.1)$$

where T' equals the normalized transmittance and P(0) is the output power through the water with a lint concentration equal to zero.

3.5 EXPERIMENTAL SETUP

To determine instantaneous lint concentrations in the washing machine during the washing cycle, water from the washing machine is sampled continuously and its transmittance measured. The experimental setup is illustrated in Figure 3.3.

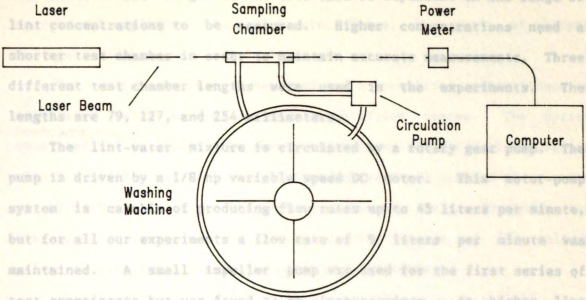


Figure 3.3 Experimental setup

The instantaneous sampling is accomplished by circulating some of the lint-water mixture from the washing machine through a transparent test chamber. The lint-water solution is continuously drawn from the area between the basket and the tub of the washing machine. This location was selected because it is close to the vicinity of the agitator while not allowing the fabric to obstruct the flow. Close to the agitator, the largest fluid velocities and the most complete mixing occur. It is assumed that this location will provide a representative approximation of the lint concentration in the washing load. After passing through the test chamber the liquid is returned to the top surface of the tub.

The test chamber is a tube 15 millimeters in diameter. The intake and outlet ports are mounted on the sides of the tube and each end is covered with a 22 X 22 X 0.2 millimeter glass microscope slide cover. The slide covers are used to minimize the dissipation of light at the boundaries. The length of the tube used is dependent on the range of lint concentrations to be measured. Higher concentrations need a shorter test chamber in order to maintain accurate measurements. Three different test chamber lengths were used in the experiments. The lengths are 79, 127, and 254 millimeters.

The lint-water mixture is circulated by a rotary gear pump. The pump is driven by a 1/8 hp variable speed DC motor. This motor-pump system is capable of producing flow rates up to 45 liters per minute, but for all our experiments a flow rate of 9 liters per minute was maintained. A small impeller pump was used for the first series of test experiments but was found to be inappropriate. At higher lint concentrations lint began accumulating on the impeller. This accumulation eventually caused a flow restriction or blockage.

The light source is a continuous 3 Watt Argon-ion laser made by Lexel (Model 95-3). For all experiments a one watt beam was selected. This amount of power was not necessary but convenient for our equipment. A 10-15 minute warm-up time is required for the laser to reach a steady power level. A laser power meter is used to measure the power level of the incoming and outgoing beam.

The power meter is a Surface Absorbing Disc Calorimeter. The calorimeter converts the laser light to heat. A thermopile then produces a voltage proportional to the heat absorbed. A factory calibration data sheet states that 95.0 millivolts of electricity are produced per watt of laser light. The response time of the power meter is about 10 seconds. With this relatively long response time the meter averages out high frequency fluctuations.

The output voltage from the power meter is amplified by an operational amplifier with a 97.4 gain. Hence, the ratio of output voltage to light power is 9.25 volts per watt.

The output voltage is measured and a normalized transmittance is calculated using a digital data acquisition system. The system consisted of a Digital Equipment Corporation, PDP 11/73 microcomputer with D/A and A/D capabilities. Output voltages are sampled at a rate of 21.25 Hertz. Because the washing cycles are run for 30 minutes and disc space is limited, every 12 seconds the average of 255 voltage values is calculated before the processing continued.

These values are converted to lint concentrations using calibration data and stored in a data file. Details of the calibration procedure and of its results are discussed in the following section.

3.6 CALIBRATION OF LINT CONCENTRATION MEASUREMENT TECHNIQUE

The relationship between lint concentration and normalized light transmittance is obtained from a calibration experiment. For this experiment 7.5 grams of lint collected by the clothes dryer is rehydrated and suspended in the washing machine with 60 liters of water. While using the agitator to keep the lint uniformly suspended the normalized transmittance is measured. This process is repeated several times with different lint concentrations and for each of the three different test chambers. The results are plotted in Figure 3.4.

Using this calibration a general functional relationship between normalized transmittance and lint concentration is established to be

$$C_w = \ln(T') \lambda \qquad (3.2)$$

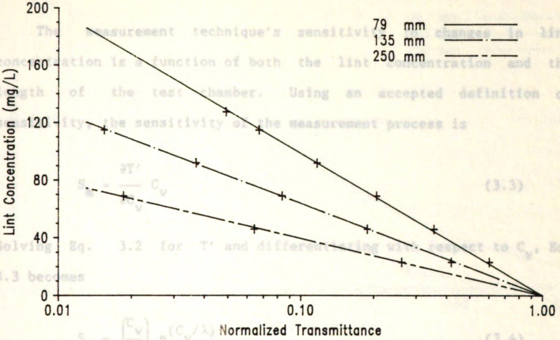


Figure 3.4 Calibrations for sampling chambers

where λ is defined as the calibration constant. The value of λ was found to be a function of tube length and fabric type, but independent of fluid speed in the test chamber and laser power. For the three tube lengths of 250, 135 and 79 millimeters values of λ for the white cotton cloth were found to be -17.1, -27.7, and -42.8 mg/L.

3.7 SENSITIVITY LEVELS OF LINT CONCENTRATION MEASUREMENTS

The measurement technique's sensitivity to changes in lint concentration is a function of both the lint concentration and the length of the test chamber. Using an accepted definition of sensitivity, the sensitivity of the measurement process is

$$S_m = \frac{\partial T'}{\partial C_w} C_w \qquad (3.3)$$

Solving Eq. 3.2 for T′ and differentiating with respect to C_w, Eq. 3.3 becomes

$$S_m = \left(\frac{C_w}{\lambda}\right) e^{(C_w/\lambda)} \qquad (3.4)$$

A plot of the measurement sensitivity for the three different lengths of test chambers is found in Figure 3.5. From these plots it is observed that each test chamber has a peak sensitivity range. In general, the long tube has a greater sensitivity at low concentrations and the short tube, at high lint concentrations.

3.8 EXPERIMENTAL PROCEDURES

In order to obtain accurate data using the lint measuring technique described above a detailed experimental procedure was established. A large portion of the measurement process is computer controlled. The FORTRAN program LINT was developed to assist the operator in performing lint experiment measurements and in following the experimental procedure. A listing of LINT is given in Appendix A.

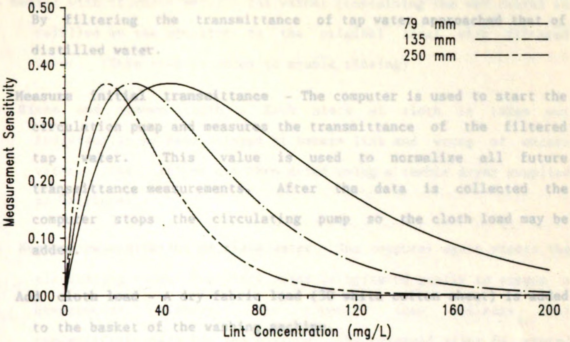


Figure 3.5 Sensitivity of lint concentration measurement

The lint measurement procedure developed and used for data gathering consists of the following sequence of steps:

- **Select and enter parameters for the run** - Using the program LINT the operator performs lint measurements for washing runs with various governing parameters. Responding to computer prompts, the operator enters his/her name, the oscillation frequency of the agitator, the stroke length of the agitator, the duration of the test, the agitator being tested, the size and composition of the cloth load and the length of test chamber used. All this information is then stored in a data file created for this run.

- **Fill tub with filtered water** - The washing machine is manually filled with 60 liters of filtered tap water. (The water filtration system consisted of two line filters installed in series, each capable of removing suspended particles larger than five microns.) By filtering the transmittance of tap water approached that of distilled water.

- **Measure initial transmittance** - The computer is used to start the circulation pump and measures the transmittance of the filtered tap water. This value is used to normalize all future transmittance measurements. After the data is collected the computer stops the circulating pump so the cloth load may be added.

- **Add cloth load** - A dry fabric load (30 white cotton sheet) is added to the basket of the washing machine.

- **Agitate and continuously measure lint concentration** - Before the experiment continues the drive linkage is manually adjusted to the correct stroke length. Then, at the operator's, command the computer starts the circulating pump, starts the drive system at the desired oscillation frequency, and begins collecting the lint concentration data at the specified sampling rate (five samples per minute). The computer maintains these conditions for the duration of the test (usually 30 minutes), then shuts off the circulating pump, stops sampling and signals the operator. All data gathered is stored in the established data file.

- **Drain and save linted water** - After the washing stops the operator must quickly drain the tank. The tank is drained by gravity (neutral drain) by opening a value connected to the bottom of the tank. The drained linted water (drain water) is transferred to a holding tank for later measurement of lint concentration.

- **Refill with filtered water** - The washer (containing the wet cloth) is refilled by the operator to the original level with filtered water. (This step is added to enable rinsing)

- **Rinse and remove cloth** - Each piece of cloth is taken out individually by hand, rinsed of excess lint and wrung of excess water. The clothes are then dried using a tumble dryer supplied by Whirlpool.

- **Measure concentration of rinse water** - The computer again starts the circulating pump. The rinse water is agitated gently to ensure a homogeneous mixture. The computer then collects lint concentration data for the mixture. The averaged value is stored as the Rinse Concentration (C_r). After the measurement has been taken the computer stops the circulating pump and the agitation.

- **Discard rinse water and refill with drain water** - The operator removes the rinse water from the tank and pumps the drain water back into the washer for lint concentration measurement.

- **Measure concentration of drain water** - The computer again starts the circulation pump. The drain water is agitated gently to ensure a homogeneous mixture. The computer measures the lint concentration of the mixture. This value is stored as the Drain Concentration (C_d). After the measurement has been taken the computer stops the circulating pump and the agitation.

- **Discard drain water** - The operator removes the drain water from the tank and prepares the washing machine for the next run.

The total time for the process is about one hour. All the data collected by the data acquisition system is placed in data files for further processing. An example of a data file is shown in Figure 3.6 and plotted in Figure 3.7.

3.9 PARAMETER ESTIMATION

Referring back to the mathematical model developed in Chapter 2 (Eq. 2.12), it is observed that the concentration of lint in the water (C_w) is a function of the variables β, C_t^i, K', K_t, and t. From a data profile obtained from the lint concentration measuring technique, values for C_w and t are known. Using parameter estimation, values for the remaining variables and/or combinations of variables can be determined from the data profile.

In order to estimate these parameters, Eq. 2.12 is simplified to the following form,

$$C_w = A\,(1 - e^{-B\,t}) + C\,t \qquad (3.5)$$

where

$$A = \beta \left(C_t^i - \frac{K_t}{K'} \right) \qquad (3.6)$$

$$B = K' \qquad (3.7)$$

$$C = \beta\, K_t \qquad (3.8)$$

Using a linear-nonlinear regression analysis, the constants A, B, and C are determined from a concentration versus time plot. The analysis technique was conducted such that the sensitivity coefficients are at maximums during evaluation.

```
; Data File - D15018A.DAT
; Test Date - 03-MAR-88
; Test Time - 19:28:00
;
; Run Description    - MATRIX
; Data Collected By - DAVE
;
; Cloth Load - A,B,C
; Agitator   - REGULAR
;
; Dwell Time -    12 sec     Oscillation Rate -  1.50 Hertz
;   Samples  -   151         Stroke Length     - 180.0 degrees
; Total Time - 30.0 min      Agitator Motion   - Symmetric
;
; Total Laser Power       -  995. mW
; Initial Chamber Power -  660. mW
;
; Rinse Concentration    -  56.6 mg/L
; Drain Concentration    -  76.7 mg/L
;
; Time (min)  Lint Concentration (mg/L)
; ----------  -------------------------
;
CO  0.0000000      0.0000000
CO  0.2000000      0.3628047
CO  0.4000000       2.963831
CO  0.6000000       7.795936
CO  0.8000000       13.49236
CO   1.000000       18.95062
CO   1.200000       24.60547
CO   1.400000       29.65394
CO   1.600000       34.01007
CO   1.800000       37.25677
CO   2.000000       41.24780
CO   2.200000       44.45755
.       .              .
.       .              .
.       .              .
CO   27.80000       109.0104
CO   28.00000       108.7201
CO   28.20000       109.0015
CO   28.40000       109.5254
CO   28.60000       110.0293
CO   28.80000       110.1405
CO   29.00000       110.0593
CO   29.20000       110.2567
CO   29.40000       110.3221
CO   29.60000       110.6039
CO   29.80000       110.7411
CO   30.00000       111.2437
```

Figure 3.6 Typical data file

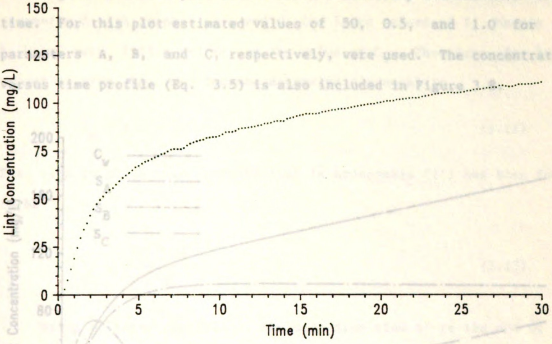


Figure 3.7 Plot of lint concentration versus time

The sensitivity coefficients are calculated with respect to each of the three parameters A, B, and C. Using Eq. 3.5, the following sensitivity coefficients are found,

$$S_A = \frac{\partial C_w}{\partial A} A = A\,(1 - e^{-B\,t}) \tag{3.9}$$

$$S_B = \frac{\partial C_w}{\partial B} B = A\,B\,t\,e^{-B\,t} \tag{3.10}$$

$$S_C = \frac{\partial C_w}{\partial C} C = C\,t \tag{3.11}$$

Figure 3.8 is a plot of the above sensitivity coefficients versus time. For this plot estimated values of 50, 0.5, and 1.0 for the parameters A, B, and C, respectively, were used. The concentration versus time profile (Eq. 3.5) is also included in Figure 3.8.

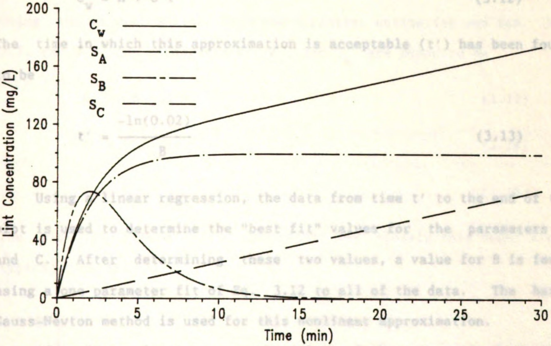


Figure 3.8 Sensitivity coefficients of model parameters

The sensitivity coefficients are important because they indicate the magnitude of change of the response of the model due to perturbations in the values of the parameters. It is observed that the sensitivity of parameter B is high at small times, but decreases to almost zero at later times.

Because of the large differences in the sensitivity of the parameter B with respect to A and C, for large values of t, changes in B have very little effect on the value of C_w. Therefore, for large values of time, Eq. 3.5 becomes approximately equal to

$$C_w = A + C\,t \tag{3.12}$$

The time in which this approximation is acceptable (t′) has been found to be

$$t' = \frac{-\ln(0.02)}{B} \tag{3.13}$$

Using a linear regression, the data from time t′ to the end of the plot is used to determine the "best fit" values for the parameters A and C. After determining these two values, a value for B is found using a one parameter fit of Eq. 3.12 to all of the data. The basic Gauss-Newton method is used for this nonlinear approximation.

Of primary interest are the values of the parameters β, K_t, K', and C_t^i. Only three of these can be determined using Eqs. 3.6 through 3.8. In order to find a solution the following new parameters are introduced:

$$C_w^{ie} = \beta\, C_t^i \tag{3.14}$$

and

$$K_w = \beta\, K_t \tag{3.15}$$

By substituting C_w^{ie} and K_w for C_t^i and K_t, the parameter β is eliminated from Eq. 2.12 leaving the equation

$$C_w = \left(C_w^{ie} - \frac{K_w}{K'}\right)(1 - e^{-K' t}) + K_w t \qquad (3.14)$$

Using the values obtained from the parameter estimation and Eqs. 3.6, 3.7, and 3.8, the parameters K', K_w, and C_w^{ie} are found to be

$$C_w^{ie} = A + C/B \qquad (3.17)$$

$$K' = B \qquad (3.18)$$

$$K_w = C \qquad (3.19)$$

The plot in Figure 3.9 illustrates how closely this model fits a typical data set.

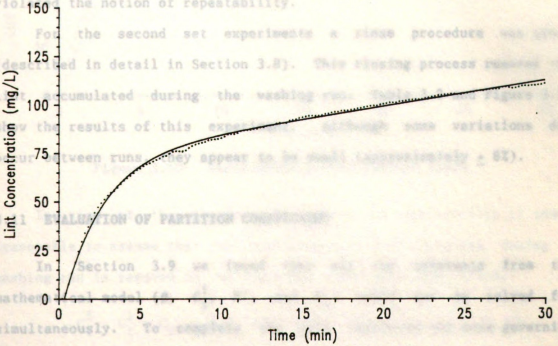


Figure 3.9 Typical plot of model fit to data

The new parameter C_w^{ie} can be interpreted as the equilibrium lint concentration in the water at time equal to zero. The meaning of the parameter K_w is the rate at which the equilibrium lint concentration in the water increases with time. By determining these three constants for a washing run, different operating conditions can be quantitatively compared with each other.

3.10 REPEATABILITY

Two sets of preliminary experiments were carried out to test the repeatability of the lint measurement technique. For the first experiment six identical runs were performed without a rinse cycle. By repeating the experiments in this manner, lint concentrations "built-up" from run to run (See Table 3.1 and Figure 3.10). This accumulation of lint over several runs was considered undesirable as it violated the notion of repeatability.

For the second set experiments a rinse procedure was added (described in detail in Section 3.8). This rinsing process removed the lint accumulated during the washing run. Table 3.2 and Figure 3.11 show the results of this experiment. Although some variations did occur between runs, they appear to be small (approximately ± 8%).

3.11 EVALUATION OF PARTITION COEFFICIENT

In Section 3.9 we found that all the constants from the mathematical model (β, C_t^i, K', and K_t) could not be solved for simultaneously. To complete the data reduction one more governing equation is needed.

Table 3.1 Repeatability test without rinse

Run	Frequency (Hertz)	Stroke Length (degrees)	C_w^{ie} (mg/L)	K′ (1/min)	K_w (mg/L/min)
1	2.00	100.00	52.63	0.4953	1.1283
2	2.00	100.00	67.73	0.3380	0.8940
3	2.00	100.00	73.50	0.3425	0.9191
4	2.00	100.00	78.88	0.2639	0.8540
5	2.00	100.00	79.63	0.2919	0.9627
6	2.00	100.00	86.63	0.2934	1.0242

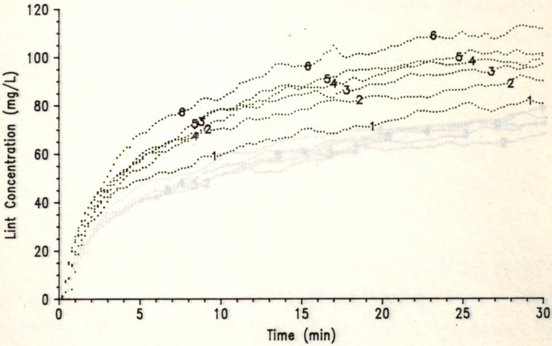


Figure 3.10 Repeatability test without rinse

In view of the previous discussion on repeatability it seems reasonable to assume that the total amount of lint generated during a washing run is removed by the drain and rinse processes. Thus,

$$C_t^f - C_t^i = C_d \gamma_d + C_r \gamma_r \tag{3.20}$$

Table 3.2 Repeatability test with rinse

Run	Frequency (Hertz)	Stroke Length (degrees)	C_w^{ie} (mg/L)	K′ (1/min)	K_w (mg/L/min)
1	2.00	100.00	56.86	0.3464	0.7818
2	2.00	100.00	46.49	0.4277	0.8095
3	2.00	100.00	46.20	0.4083	1.0112
4	2.00	100.00	54.64	0.3028	0.7746
5	2.00	100.00	50.04	0.3317	1.0140
6	2.00	100.00	52.11	0.3403	0.9602

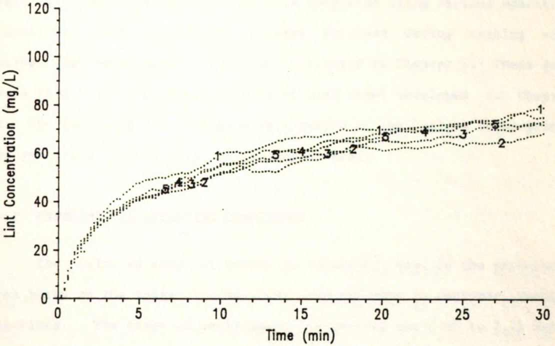


Figure 3.11 Repeatability test with rinse

Using Eqs. 2.10 and 3.15, assuming a 30 minute washing run, and solving for β, Eq. 3.20 becomes

$$\beta = \frac{30\ K_w}{C_d\ \gamma_d + C_r\ \gamma_r} \tag{3.21}$$

With a solution found for β, all of the constants from the mathematical model can be solved for and the model is complete.

CHAPTER 4

RESULTS

In this chapter a complete linting experiment is presented. The effect of changes in agitator stroke length and oscillation frequency were studied. 56 washing runs were completed using various operating conditions. Lint concentrations were recorded during washing runs using the measurement technique discussed in Chapter 3. These data were then evaluated using the lint release model developed in Chapter 2. The following sections provide a sample of the insight to be gained by measuring and modeling the linting process.

4.1 EXPERIMENTAL OPERATING CONDITIONS

The selected range of operating conditions used in the experiment was based on the range of conditions commonly used in consumer washing machines. The range of oscillation frequencies was 1.00 to 2.75 Hertz with a range of 100 to 200 degrees for the stroke length. The matrix of 28 operating points used for the experiments is shown in Figure 4.1.

Each of these operating conditions was repeated twice to help reduce systematic errors and to investigate the repeatability of the process. Because of limited supply of fabric only two fabric loads

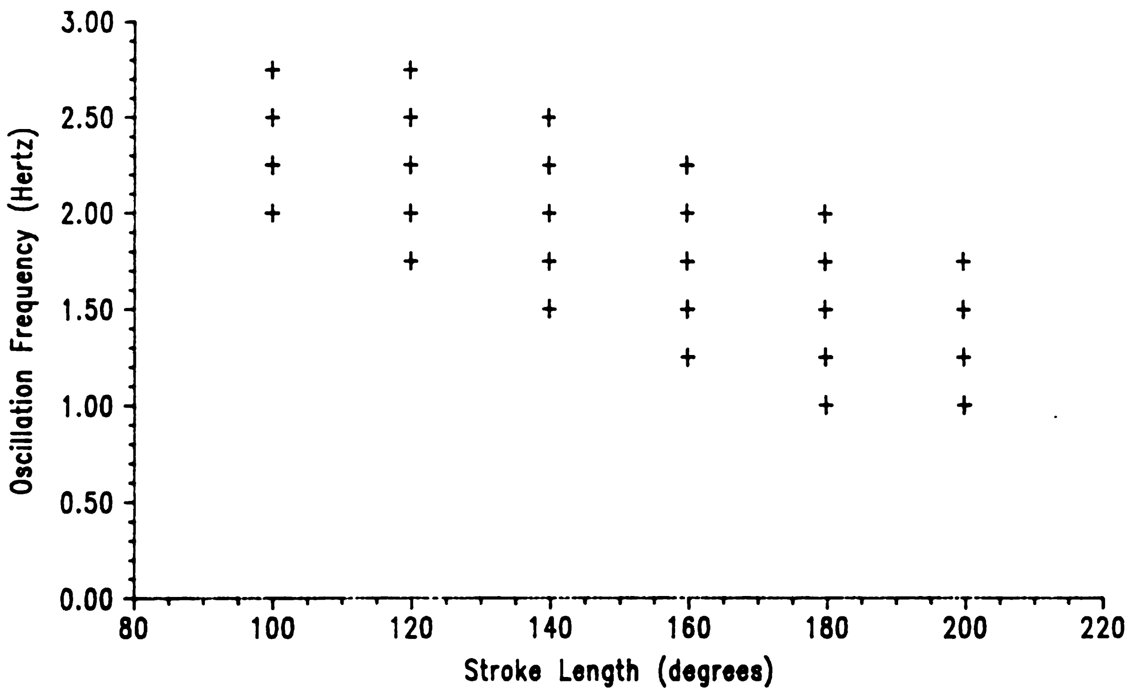


Figure 4.1 Operating points for experimental matrix

were used for the tests. When experiments were conducted consecutively, one load was used in the experiment while the other was being dried.

In order to minimize the bias due to differences in cloth loads, a specific ordering for performing runs was established. When possible, the two runs at a particular operating condition were performed using different cloth groups. Each of these runs at a particular operating condition uses cloth groups with opposing histories (i.e. one run is performed with a cloth group from a wash with a low agitation frequency and the other run with a group from a high frequency).

Because of the high lint concentrations values encountered in some of the higher frequency runs, a shorter sampling tube was needed to make accurate measurements. As shown in Chapter 3, the shorter sampling tube generally has a greater sensitivity at the higher lint concentrations than a longer tube.

Table 4.1 shows the chronological sequence of the washing runs. The table also includes, the stroke length, the oscillation frequency, the cloth load and the sampling chamber length used for each run. All runs are consecutive unless otherwise noted (for a few runs data were not collected due to equipment malfunctions). Lint concentration profiles for all runs are graphically displayed in Appendix B.

A complete description of the operating conditions for each run in the matrix is given in Table 4.1. No other washing variables were varied during the testing. Any changes in the measured parameters, therefore, are a result of the differences in the operating conditions and/or the variability of the washing process itself.

4.2 EVALUATION OF MODEL PARAMETERS

Using the parameter estimation technique from Chapter 3, the experimental data from all 56 runs were analyzed. Values were found for the model parameters C_w^{ie}, K′, and K_w. Figures 4.2a-c and 4.3a-c are plots of these model parameters versus stroke length and oscillation frequency. As seen from these plots the parameters do not correlate well with either stoke length or oscillation frequency.

Table 4.1 Order of washing runs

RUN #	STROKE (degrees)	FREQUENCY (Hz)	CLOTH LOAD	Sampling Tube Length (mm)
1	100	2.00	2	135
2	100	2.25	1	135
3	100	2.50	2	135
4	100	2.75	1	135
5*	100	2.75	2	135
6	100	2.50	1	135
7	100	2.25	2	135
8	100	2.00	1	135
9*	140	1.50	2	135
10	140	1.75	1	135
11*	140	2.00	1	135
12	140	2.25	2	135
13	140	2.25	1	135
14	140	2.00	2	135
15	140	1.75	1	135
16	140	1.50	2	135
17	180	1.00	1	135
18	180	1.25	2	135
19	180	1.50	1	135
20*	180	1.50	2	135
21	180	1.25	1	135
22	180	1.00	2	135
23	120	1.75	1	135
24	120	2.00	2	135
25	120	2.25	1	135
26*	120	2.25	2	135
27	120	2.00	1	135
28	120	1.75	2	135
29	160	1.25	1	135
30	160	1.50	2	135
31	160	1.75	1	135
32	160	2.00	2	135
33	160	2.00	1	135
34	160	1.75	2	135
35	160	1.50	1	135
36	160	2.25	1	79
37	160	1.25	2	79
38*	160	2.25	1	79
39	200	1.00	1	79
40	200	1.25	2	79
41	200	1.50	1	79
42	200	1.75	2	79
43	200	1.75	1	79
44	200	1.50	2	79
45	200	1.25	1	79
46	200	1.00	2	79

Table 4.1 continued

RUN #	STROKE (degrees)	FREQUENCY (Hz)	CLOTH LOAD	Sampling Tube Length (mm)
47	180	1.75	1	79
48	180	1.75	2	79
49	180	2.00	1	79
50	180	2.00	1	79
51	140	2.50	2	79
52	140	2.50	1	79
53	120	2.50	2	79
54	120	2.75	1	79
55	120	2.75	2	79
56	120	2.50	1	79

* Non-sequential washing runs

4.3 AVERAGE ANGULAR VELOCITY

Because no meaningful correlations were obtained using stroke length or oscillation frequency as independent variables, a new variable, average angular velocity ($\bar{\omega}$), was introduced. Average angular velocity combines both the stroke length (θ) and the oscillation frequency (f) into a new independent variable. More specifically,

$$\bar{\omega} = (2\ \theta)\ \frac{2\ \pi}{360}\ (60\ f) \tag{4.1}$$

simplifying

$$\bar{\omega} = 2.0944\ \theta\ f \tag{4.2}$$

The units for $\bar{\omega}$ are radians per minute. Figure 4.4 shows the relationship of average angular velocities to the operating conditions tested in the experimental matrix.

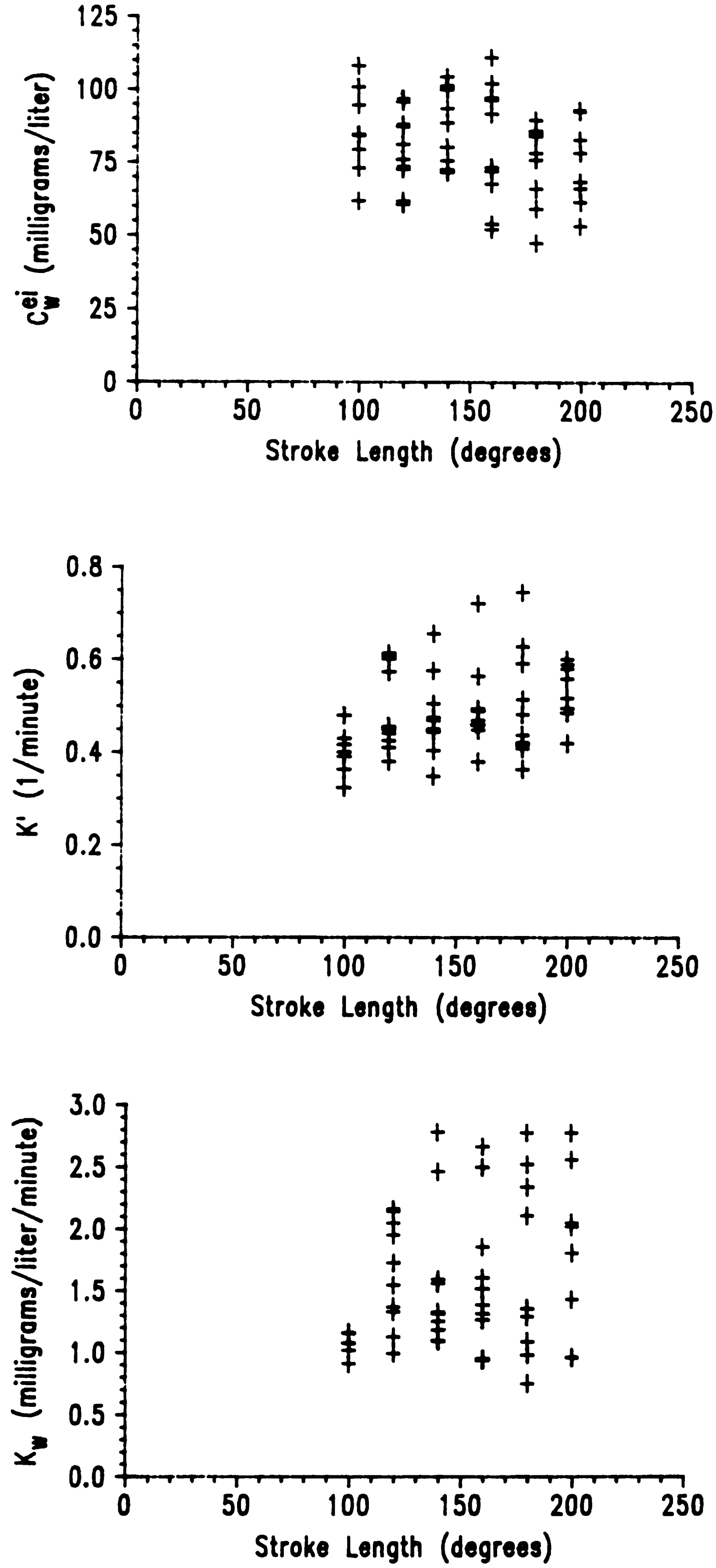


Figure 4.2a-c Model parameters versus stroke length

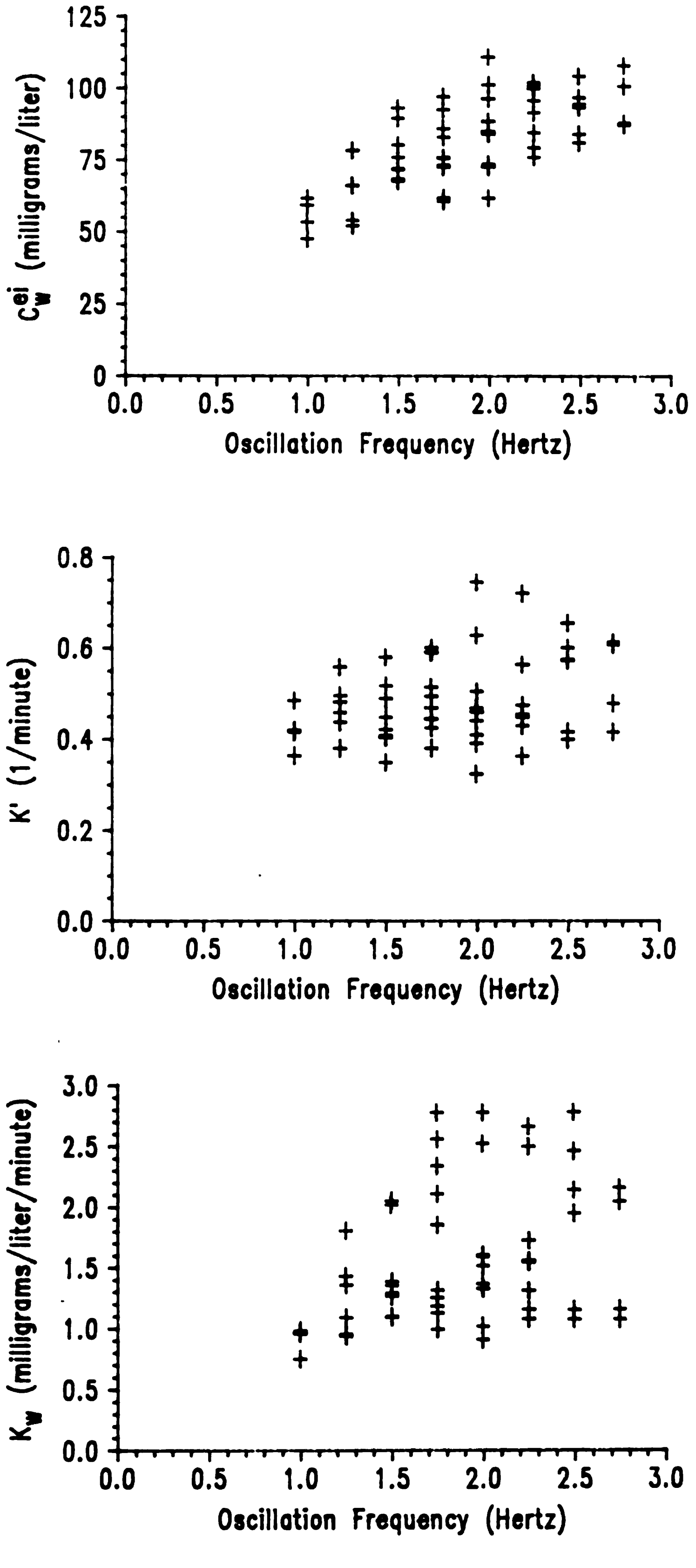


Figure 4.3a-c Model parameters versus frequency

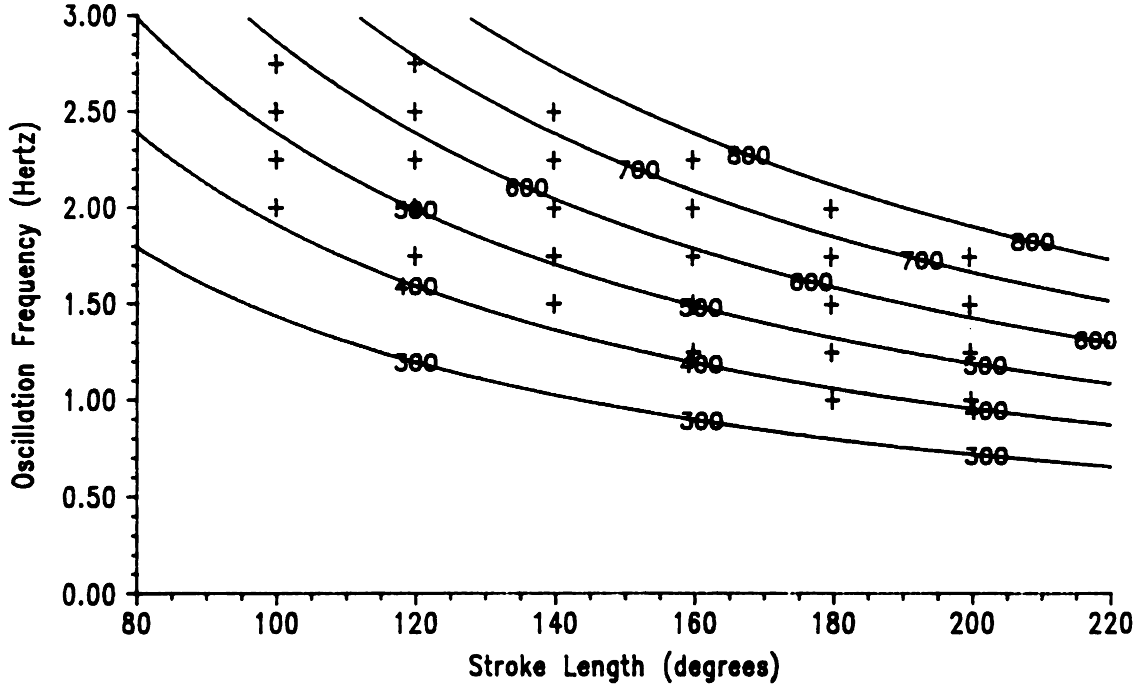


Figure 4.4 Operating conditions with velocities

Now, using this new variable as the independent variable, we again look at plots of the lint concentration model parameters (Figures 4.5a-c). As seen from these plots, definite correlations exist. These relationships will be discussed in Section 4.4.

4.4 LOW SENSITIVITY DATA POINTS

As mentioned earlier in Chapter 3, the sensitivity of the measurement procedure, in general, decreases with increased lint concentration. This raises the question of when does low measurement sensitivity affect accuracy? To give some indication a plot of the variation in the parameter K versus measurement sensitivity is shown in Figure 4.6. ΔK is defined as the deviation of K from a power equation (calculated from a least-squares regression).

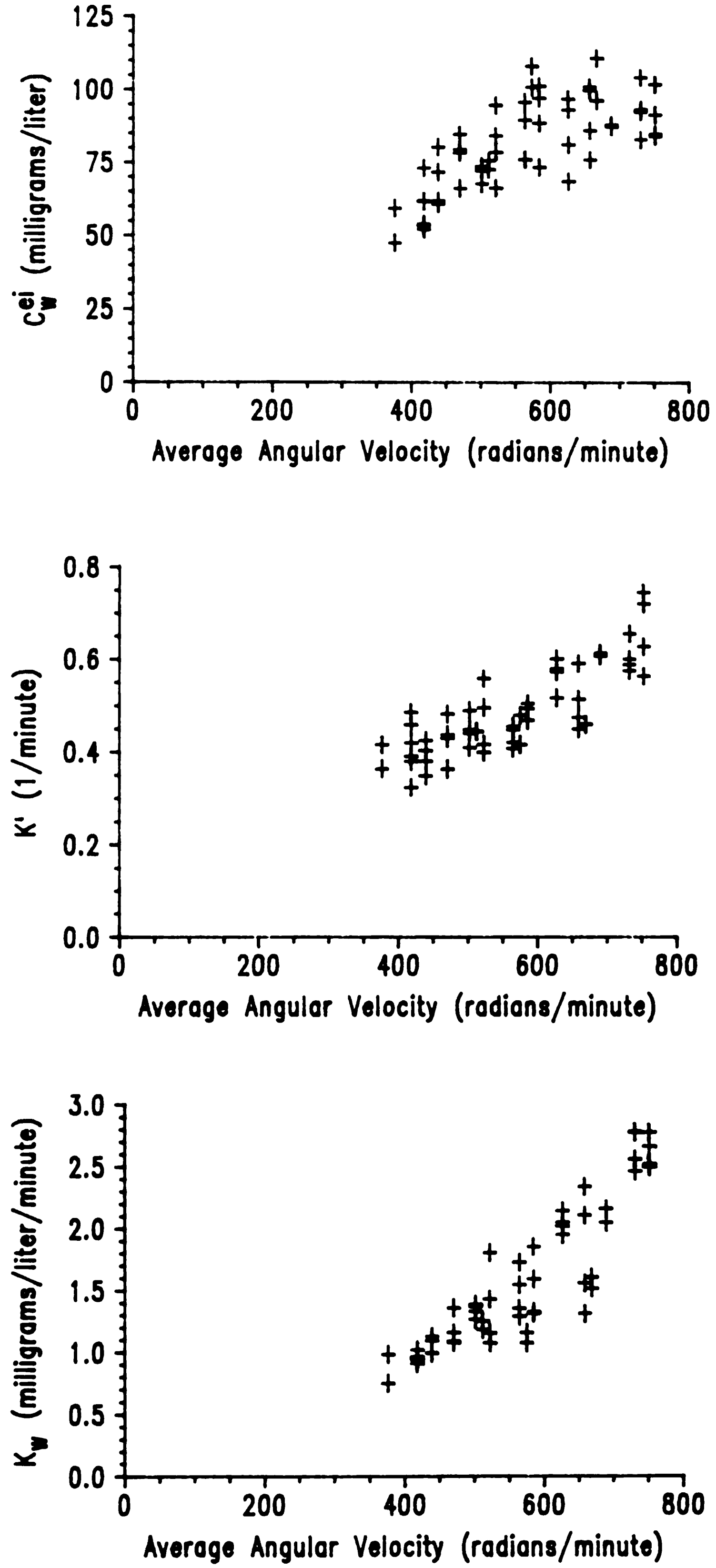


Figure 4.5a-c Model parameters versus velocity

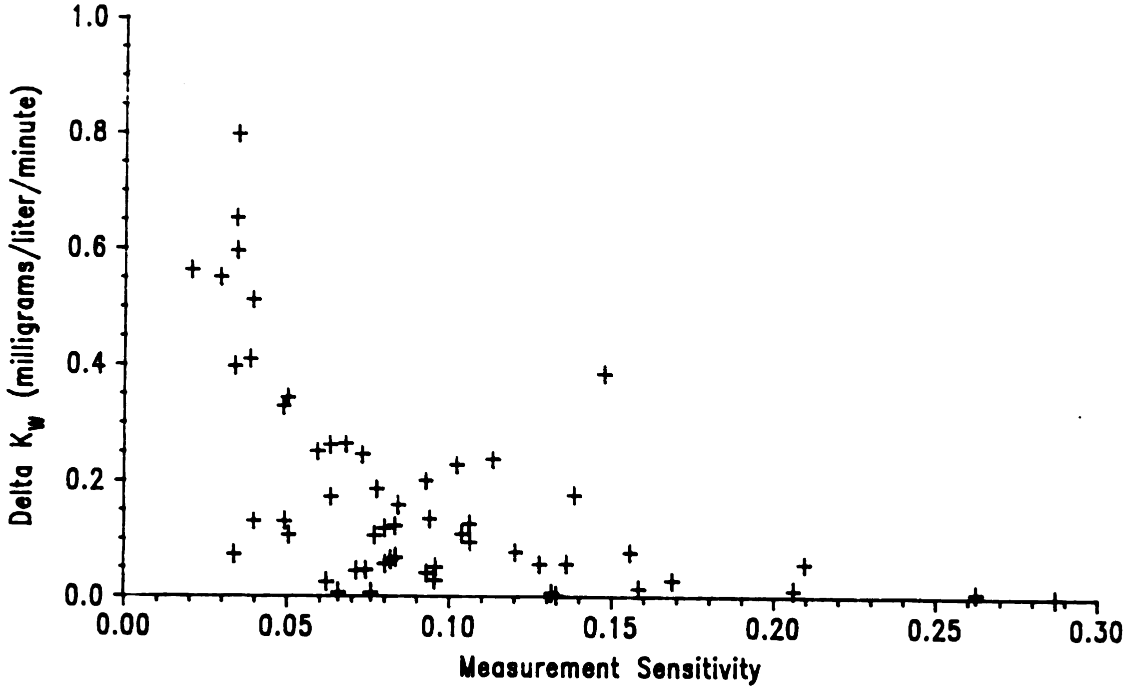


Figure 4.6 Δ K versus measurement sensitivity

From this plot we see that the highest values of Δ K occur at low measurement sensitivities. Data collected at the low measurement sensitivities are the most likely to be influenced by measurement noise. By eliminating the data collected at low measurement sensitivities, the overall spread of the parameters is greatly reduced (from ± 0.8 to ± 0.25 mg/L/min). The minimum acceptable measurement sensitivity is found to be,

$$S_m \approx 0.06 \quad (4.3)$$

With the low sensitivity data points removed, a regression using the power equation reveals the following relationships between the model parameters and average angular velocity:

$$C_w^{ie} = 0.669\ \bar{\omega}^{\,0.75} \qquad (4.4)$$

$$K' = 0.00884\ \bar{\omega} \qquad (4.5)$$

$$K_w = 0.000248\ \bar{\omega}^{\,1.75} \qquad (4.6)$$

Figure 4.7a-c illustrates how well these functions fit the data.

Functions other than the power equation may provide better "fits" or may even be closer to the actual process, but because of the limited range of $\bar{\omega}$'s tested we were unable to determine how the parameters behaved outside the test range. Therefore, a simple power law function (which passes through the origin) was used.

4.5 PARTITION COEFFICIENT

The partition coefficient α and the related parameter β also correlate well when plotted versus $\bar{\omega}$, as illustrated in Figures 4.8 and 4.9. Applying the power equation regression to the data the following relationships are found,

$$\alpha = 0.0000125\ \bar{\omega}^{\,1.5} \qquad (4.7)$$

$$\beta = 0.000775\ \bar{\omega} \qquad (4.8)$$

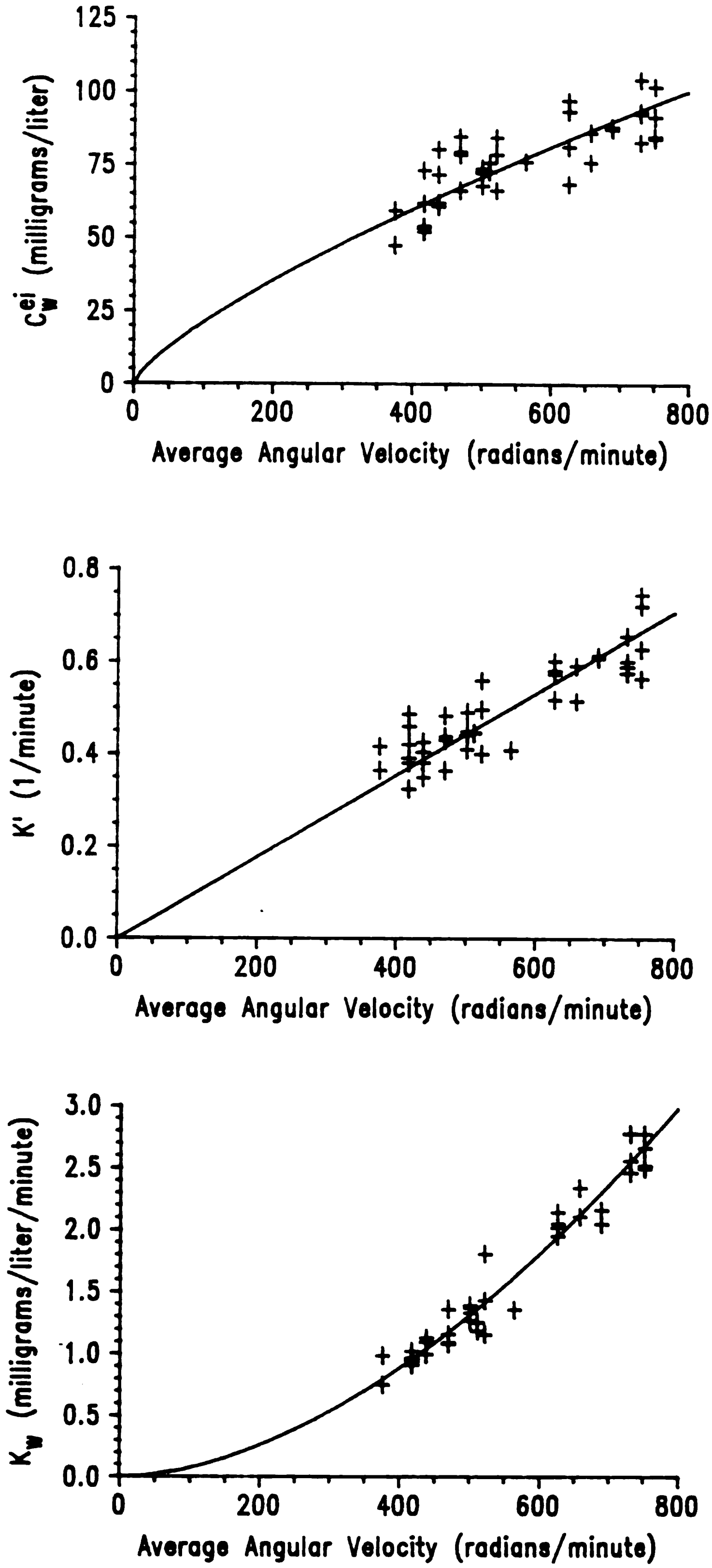


Figure 4.7a-c Model parameters versus velocity

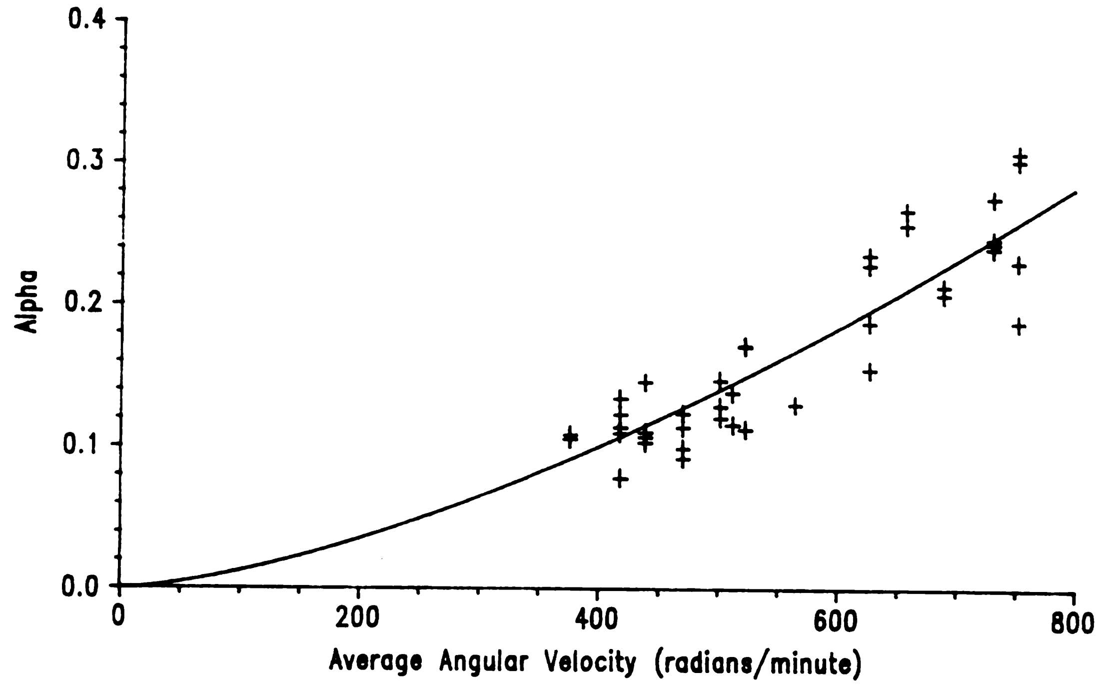


Figure 4.8 Alpha versus average angular velocity

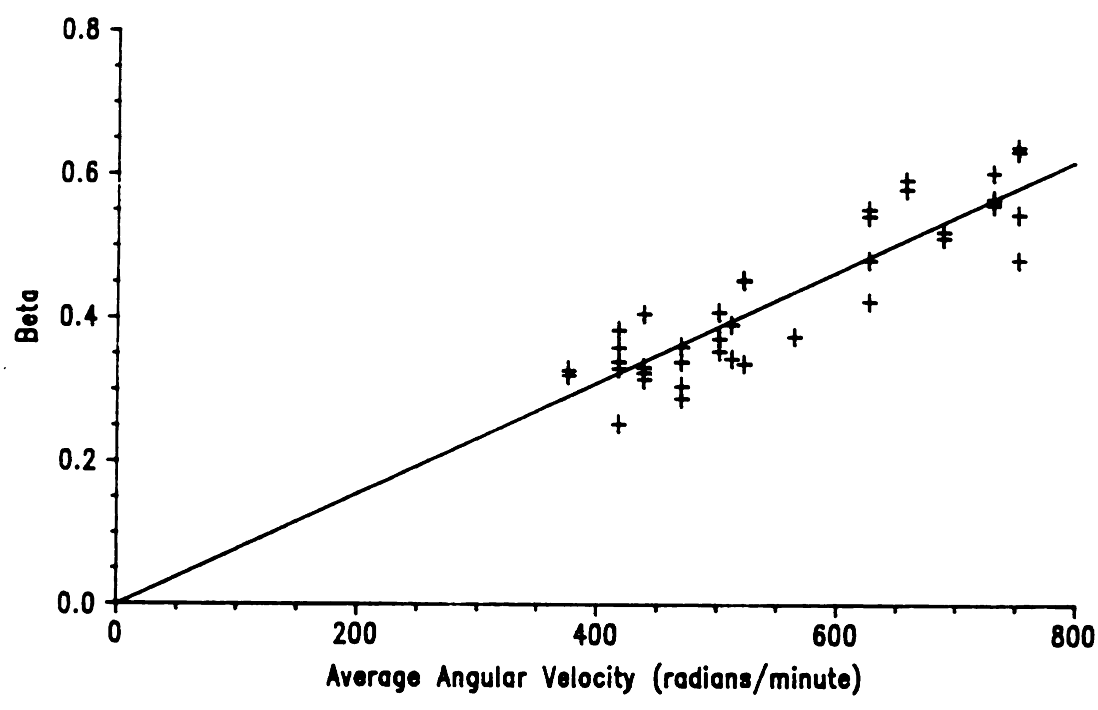


Figure 4.9 Beta versus average angular velocity

4.6 FINAL MODEL

Substituting Eqs. 4.4, 4.5, and 4.6 into Eq. 3.14 we find that,

$$C_w = a_1\,\bar{\omega}^{\,0.75}\,(1 - e^{-a_2\,\bar{\omega}\,t}) + a_3\,\bar{\omega}^{\,1.75}\,t \quad (4.9)$$

where

$a_1 = 0.641$

$a_2 = 0.000884$

$a_3 = 0.0000248$

(Eq. 4.9 is dimensionally homogeneous as long as $\bar{\omega}$ is in rad/sec.). The concentration of lint in the water appears to be only a function of average angular velocity and time. Figure 4.10 shows C_w versus time for several different $\bar{\omega}$'s.

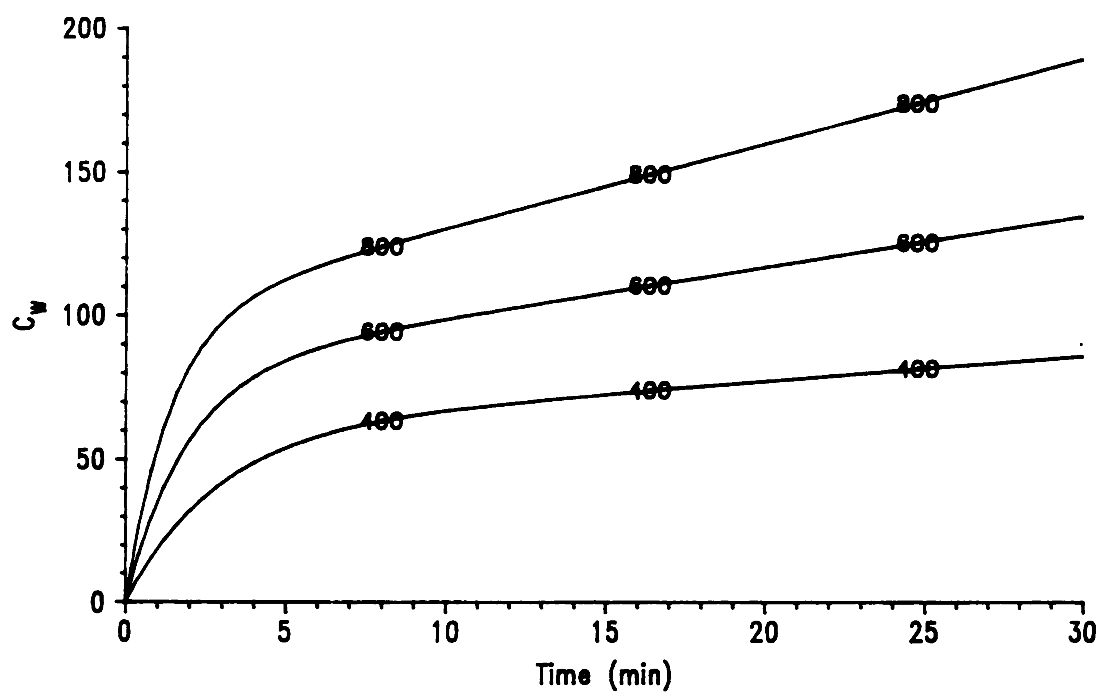


Figure 4.10 Complete lint model

Using Eq. 4.9, the data collected during the test runs can be compared to a generalized theory. By calculating the percent deviation of the actual data points from the theoretical C_w values (calculated using Eq. 4.9), a ΔC_w is defined. By plotting this value (for all data points) versus time (Figure 4.11), an approximation of the equation's accuracy ($\pm$ 20%) is demonstrated.

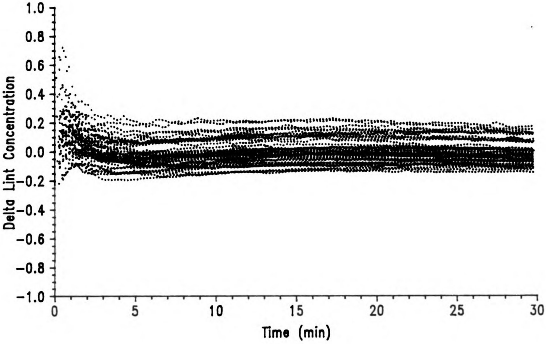


Figure 4.11 Comparison of lint model to actual data

4.7 DIMENSIONAL ANALYSIS

As seen previously, C_w is a function of the model parameters β, C_w^{ie}, K', and K_w, and t. Using these variables, their corresponding units and employing dimensional analysis, the following dimensionless values emerge:

$$C^* = \frac{C_w}{C_w^{ie}} \tag{4.10}$$

$$t^* = K' \; t \tag{4.11}$$

$$F = \frac{K_w}{K' \; C_w^{ie}} \tag{4.12}$$

Using Eqs. 4.4, 4.5, 4.6, and 4.12, F (Fanson linting constant) is found to be a constant equal to 0.0417. Figure 4.12 shows F plotted versus average angular velocity.

By substituting Eqs. 4.10, 4.11, and 4.12 into Eq. 3.14, the following dimensionless lint equation develops,

$$C^* = (1 - F)\,(1 - e^{-t^*}) + F \, t^* \tag{4.13}$$

Figure 4.13 is a plot of this relationship.

By nondimensionalizing the experimental lint profiles using Eqs. 4.10 and 4.11, all of the data collapses down to the single curve formed by Eq. 4.13. Figure 4.14 is a plot of C^* versus t^* for all data points.

The constant F is hypothesized to be a characteristic of the washing load. Because only one cloth type was used during this experiment, no concrete conclusions can be drawn. However, one could

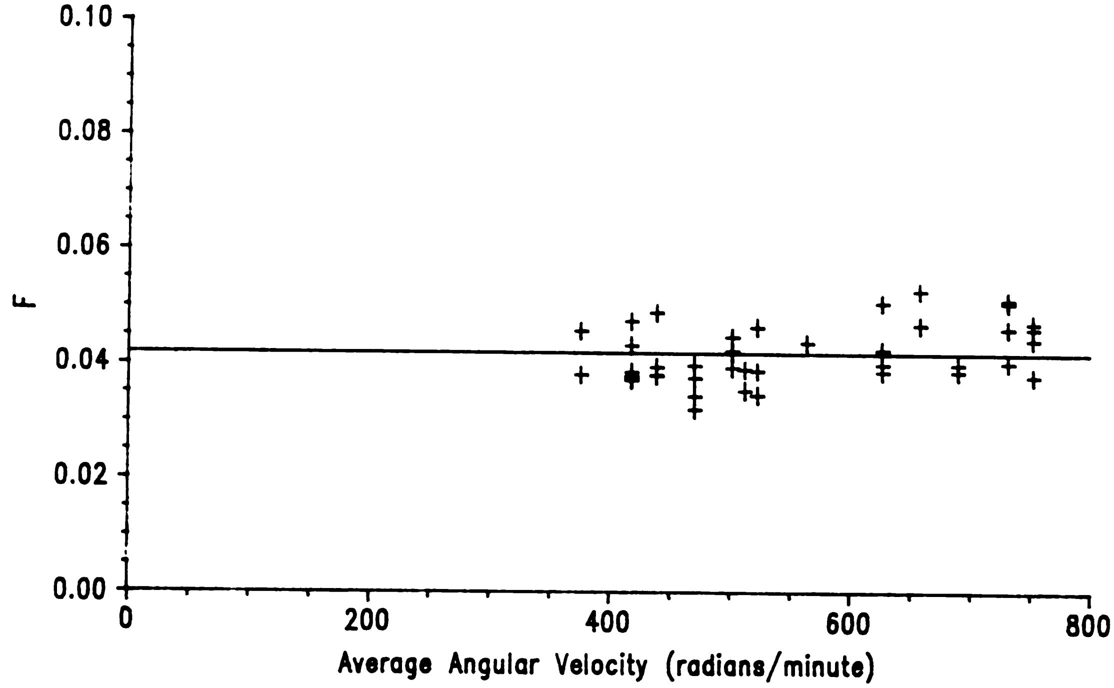


Figure 4.12 Fanson linting constant versus velocity

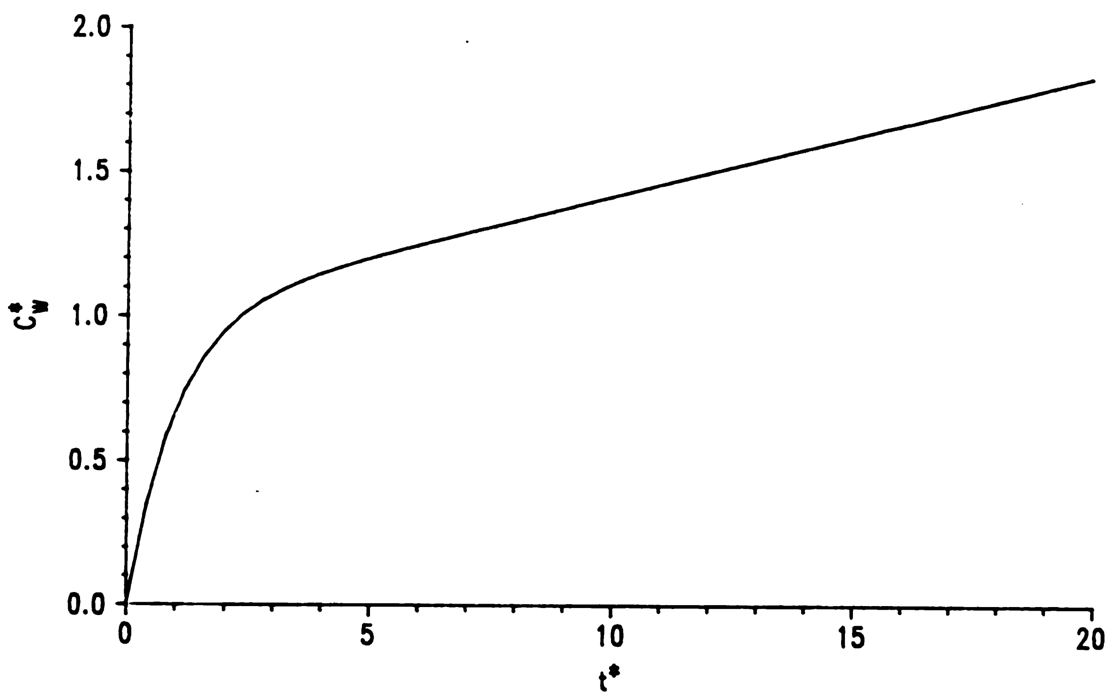


Figure 4.13 Nondimensional lint concentration versus time

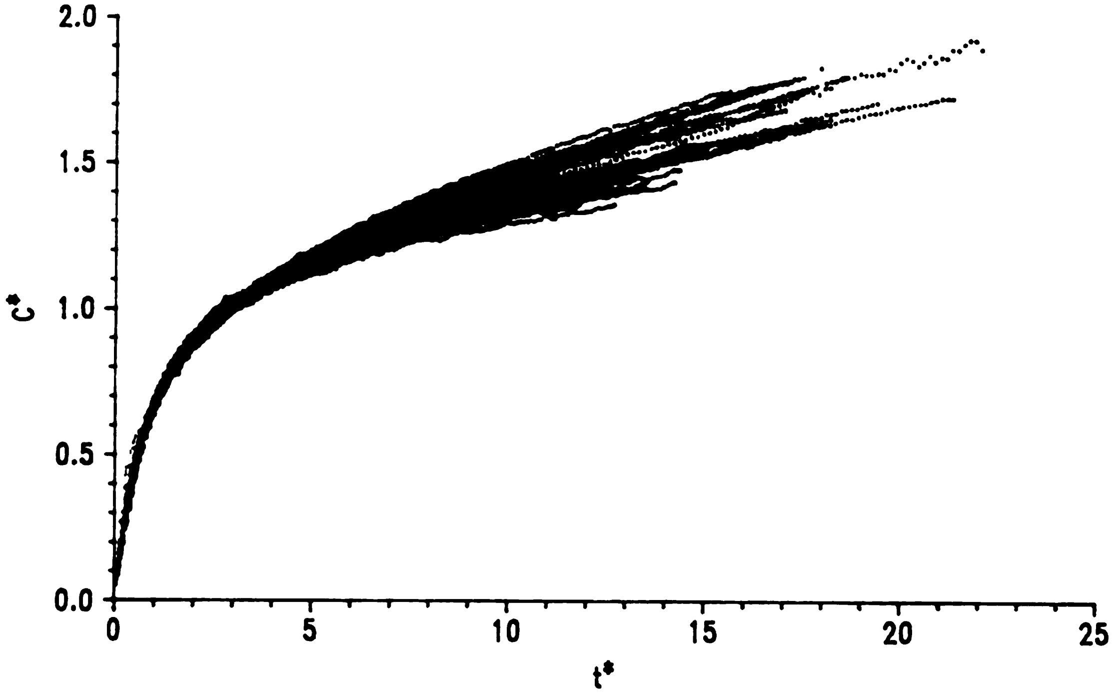


Figure 4.14 A nondimensional plot of all the data

assume that different fabric types would have different F constants. Controlled experiments, like this one, could be performed for different cloth types in order to determine the effect on the linting constant.

CHAPTER 5

CONCLUSIONS

5.1 OVERVIEW

In the previous chapters of this thesis, a study of the suspension of cloth fibers in a top-loading washing machine was presented. The focus of the study was based on the following objectives:

1. To develop a general theory for the physical processes associated with the suspension of lint,

2. To develop and implement a technique for the quantitative measurement of the amount of lint suspended in water,

3. To study the effect of agitator stroke length and oscillation frequency on the suspension and retention of lint.

Some of the main developments which occurred during the completion of these objectives were:

- The development of a mathematical model which incorporated parameters that characterize the release and production of suspended lint,

- The development, calibration, and implementation of a suspended lint concentration (mass per unit volume) measurement technique which was based on light transmittance,

- The development of software for digital data acquisition and parameter estimation procedures,

- The utilization of the lint model, lint concentration measurement technique, and associated software to gather lint data from a set of experiments in which the stroke length and oscillation frequency of the washing machine agitator were varied,

- The systematic analysis of the experimental data to determine the effect of changes in agitator stroke length and oscillation frequency on the release, production, and retention of lint.

5.2 CONCLUSIONS

Before the conclusions of this research are summarized, it should be noted that the quantitative results obtained from the experimental study partially depend on the particular experimental setup. The main area of concern is the lint measurement sampling location. Water was continually drawn from beneath the agitator, and it is not known if this location is truly a representative of the entire washing machine.

Keeping this in mind, the following observations were made and conclusions were drawn from the results of this research. The comments have been sorted in to three groups, physics, methodology, and measurements.

Physics

- Lint exists in two states, adhered to fabric and suspended in water.
- The transfer rate of adhered lint into suspension is dependent on the amount of lint currently in suspension.
- The rapid increase of suspended lint in the early stages of a wash cycle is primarily due to the suspension of adhered lint.
- Increases in the later part of a wash cycle are due to the generation of new lint.

Methodology

- Quick and accurate quantitative lint concentration measurements are obtained using the light transmittance technique.
- The calibration constant for and sensitivity of this measurement technique are functions of sampling tube length and cloth type.
- The build-up of adhered lint is greatly reduced by using a rinse cycle.

- Agitator average angular velocity effectively characterizes stroke length and oscillation frequency.

Measurements

- The rate at which new lint is produced increases with angular velocity.

- The transfer rate of adhered lint into the water increases proportionally with angular velocity.

- Using the mathematical lint model and parameter estimation good ($\pm$ 2 %) correlations are found between theoretical and experimental lint concentration profiles.

- A general lint equation which is only a function of angular velocity and time, satisfactorily models the amount of suspended fibers at any time for any of the tested agitator average angular velocities.

- Lint concentration profiles can be nondimensionalized using the dimensionless parameters, t^*, C^*, and F. The lint constant (F) may offer comparisons between other untested conditions.

Continued research and industrial implementation will refine the lint model and lint measurement technique into valuable developmental and evaluation tools.

5.3 SUGGESTED RESEARCH

In the experimental study only the agitator's stroke length and oscillation frequency were varied. Variations in any of the other operating conditions may produce different results. For example, changes in any of the following items will most likely have an effect on the release, production, and/or retention of lint:

- Type of cloth load (i.e. cotton, polyester, terry cloth)
- Size of cloth load (total mass of cloth)
- Size of cloth pieces
- Cloth to water ratio
- Water sample intake location
- Agitator
- Cloth's age
- Length of washing cycle
- Rinse cycle
- Method of drying cloth

Since no studies (using these new methods) have been completed in which these conditions were varied, their effect on linting can not be evaluated at this time. Each of these conditions should be studied in order to fully understand the process of lint suspension.

APPENDICES

APPENDIX A

PROGRAM LINT

```
C-----------------------------------------------------------------------
C
        PROGRAM LINT
C
C-----------------------------------------------------------------------
C
C          LINT is the main driving program for preforming lint
C       measurement. The measurement technique is based on light
C       transmittance.
C
C-----------------------------------------------------------------------
C
C       Linking Procedure:
C
C       Using the taskbuilder TKB type the following at the TKB> prompt:
C       TKB> LINT  = LINT, KSAM, TNKFRQ, NFILE, LNTUTL, LNTTXT, PDLDAT
C       TKB> @[1,54]LNK2KLAB
C       TKB> @[1,54]KCOM
C
C-----------------------------------------------------------------------
C
C       Written by:
C               David J. Fanson
C               Michigan State University
C
C       Last Modification: April 1, 1988
C
C-----------------------------------------------------------------------
C
C       Variable List:
C
C               DWELL   Sampling dwell
C
C               FILEN   Name of output data file
C
C               FREQ    Frequency of tank
C
C               N       Number of samples
C
C               NFILE   Subroutine to make plotting files
C
```

```
C               POWER   Power of laser beam in milliwatts
C
C               RATIO   POWER/TNKPWR (Power Ratio)
C
C               SAMPLE  Subroutine to sample voltages from A/D board
C
C               TNKFRQ  Subroutine to set frequency of oscillating tank
C
C               TNKPWR  Laser power through tank
C
C               TOTPWR  Total laser power
C
C               VAL     An array of all points taken
C
C               VOLT    Average voltage from thermopile
C
C----------------------------------------------------------------------
C
        REAL VAL(256),CONC(1000),TIME(1000),B(5)
        INTEGER DWELL,MODE,N,I,J,NCHAN,SCHAN,MOTOR,TUBE
        REAL RATE
        CHARACTER*15 FILEN
        CHARACTER*40 XTIT,YTIT
        CHARACTER ANS
        BYTE ESC
C                                               Create temporary
C                                                storage file of
C                                                measured values (in
C                                                case program crashes)
        CALL ASSIGN(3,'LINT.TMP')

        ESC = "033
C                                               Set both D to A
C                                                channals to zero
        CALL DTOA(0,0.0)
        CALL DTOA(1,0.0)
C                                               Set sampling
C                                                parameters
        J = 0
        N = 256
        RATE = 50.
        SCHAN = 0
        NCHAN = 1
        MODE = 1
C                                               Set plotting labals
        XTIT = 'TIME (min)'
        YTIT = 'LINT CONCENTRATION (mG per L)'
C                                               Set motor calibration
        MOTOR = 2

C       WRITE(5,95)
C 95    FORMAT(/' Using (1) Linear Tank or (2) Washing Machine : '$)
C       READ(5,*) MOTOR
C                                               Ask which sampling tube
```

```
C                                                       used
      WRITE(5,96)
 96   FORMAT(/' Which Sampling Tube Length '
     +       /'        (1) 25.4 cm, '
     +       /'        (2) 12.7 cm, or '
     +       /'        (3)  7.9 cm  : '$)
      READ(5,*) TUBE

      IF (TUBE.EQ.3) THEN
C                                                      Constant for 7.9 cm
C                                                       sampling tube
        CONST = -98.6
      ELSE IF (TUBE.EQ.2) THEN
C                                                      Constant for 12.7 cm
C                                                       sampling tube
        CONST = -63.7
      ELSE IF (TUBE.EQ.1) THEN
C                                                      Constant for 25.4 cm
C                                                       sampling tube
        CONST = -39.4
      END IF
C                                                      Open output file for
C                                                       results
      TYPE *
      CALL OFILE(FILEN,XTIT,YTIT)
C                                                      Enter run text
      CALL TXT(DWELL,FREQ,NUM)
C                                                      Start run
      TYPE 122
 122  FORMAT(' Hit "RETURN" to Start Sampling..... '$)
      READ(5,*)
C                                                      Turn on circulating pump
      CALL DTOA(1,3.1)
C                                                      Sample voltage produced
C                                                       by laser
 5    CONTINUE
      TYPE *
      TYPE *
      TYPE *,'Measure laser beam power in air : '
      TYPE *
      CALL FSAM(VAL,VOLT,SCHAN,NCHAN,RATE,N,MODE)
C
C                                                      Convert voltage to
C                                                       milliwatts of power
C                                                       using known constants
      CALL CALPOW(TOTPWR,VOLT)
C                                                      Print measured value
      TYPE *
      TYPE *
      WRITE(5,98) TOTPWR
 98   FORMAT(' ; Total power of laser = ',F5.0,' mW')

C                                                      Check power level to
C                                                       see if in range and
```

```
C                                                          give warning if not in
C                                                          range
      IF (TOTPWR.LT.10.0) THEN
        TYPE *
        TYPE *,'******** CHECK CONECTIONS!! ********'
        CALL BEL(3)
      ELSE IF (TOTPWR.LT.900.0) THEN
        TYPE *
        TYPE *,'******** POWER TOO LOW!! ********'
        CALL BEL(3)
      ELSE IF (TOTPWR.GT.1017.0) THEN
        TYPE *
        TYPE *,'******** POWER TOO HIGH!! ********'
        CALL BEL(3)
      END IF
C                                                         Sample voltage of
C                                                          laser through sample
C
      TYPE *
      TYPE *,'Measure laser beam power through test chamber : '
      TYPE *
      CALL FSAM(VAL,VOLT,SCHAN,NCHAN,RATE,N,MODE)
C
C                                                         Convert voltage to
C                                                          milliwatts of power
C                                                          using known constants
      CALL CALPOW(TNKPWR ,VOLT)
 7    CONTINUE
C                                                         Write results on screen
      TYPE *
      WRITE(5,100) TOTPWR,TNKPWR
 100  FORMAT (' ;',/,
    +        ' ; Total Laser Power     - ',F5.0,' mW',/,
    +        ' ; Initial Chamber Power - ',F5.0,' mW',/,' ;')

      TYPE 99
 99   FORMAT(' Do you want to change the power (Y,N) ? '$)
      CALL WTCHAR(ANS)
      IF (ANS.EQ.'Y'.OR.ANS.EQ.'y') GOTO 5
      IF (ANS.NE.'N'.AND.ANS.NE.'n') GOTO 7
C                                                         Shut off circulating
C                                                          pump
      CALL DTOA(1,0.0)
C                                                         Print sampling screen
      CALL LSCRL(5,8)

      WRITE(5,105) ESC
 105  FORMAT('+',A1,'[12;1H')

      WRITE(5,100) TOTPWR,TNKPWR
      WRITE(1,100) TOTPWR,TNKPWR

      TYPE *,'      Time(min) Voltage(volts) Power(mW)',
    +' Concentration(mG/L)'
```

```
      TYPE *,'----- --------- ------------- ---------',
     +' -------------------'
      TYPE *
C                                                Initialize sampling
      TIME(1) = 0.
      CONC(1) = 0.
      POWER = TNKPWR
      MODE = 0
      RATE = N/DWELL
      RATE = RATE*1.1
      SAVE1 = 0.0
      SAVE2 = 0.0
      I = 1

      TYPE 123
      READ(5,*)
C                                                Start drive motor
      CALL TNKFRQ(FREQ,MOTOR)
C                                                Start Circulating Pump
      CALL DTOA(1,3.1)

      CALL LSCRL(5,8)

      WRITE(5,110)I,TIME(I),VOLT,POWER,CONC(I)
      WRITE(3,110)I,TIME(I),VOLT,POWER,CONC(I)
 110  FORMAT(' ',I4,2X,F7.2,7X,F6.3,6X,F7.2,8X,F7.3)

      CALL STRTIM

      DO 10 I=2,NUM

        CALL MARK(45,DWELL,2,IDS)
C                                                Sample from channel 0

        CALL FSAM(VAL,VOLT,SCHAN,NCHAN,RATE,N,MODE)

C                                                Check for errors and
C                                                 correct
        IF (VOLT.EQ.SAVE1) THEN
          VOLT = 2.0*SAVE1-SAVE2
        END IF

        SAVE2 = SAVE1
        SAVE1 = VOLT
C                                                Up date time
        TIME(I) = (I-1)*DWELL/60.
C                                                Convert voltage to
C                                                 millivatts of power
C                                                 using known constants
        CALL CALPOW(POWER,VOLT)
C                                                Calculate Power Ratio
        RATIO = POWER/TNKPWR
C                                                Calculate Concentration
        IF (RATIO.GT.0.0) THEN
```

```
            CONC(I) = LOG10(RATIO)*(CONST)
          ELSE
            CONC(I) = 0.0
          END IF
C                                                    Print results on the
C                                                     screen and in
C                                                     temporary storage file

          WRITE(3,110)I,TIME(I),VOLT,POWER,CONC(I)
          WRITE(5,110)I,TIME(I),VOLT,POWER,CONC(I)

          CALL WAITFR(45,IDS)

 10     CONTINUE
C                                                    Stop Circulating Pump
        CALL DTOA(1,0.0)

        CALL BEL(3)

        WRITE(5,112) ESC
 112    FORMAT('+',A1,'[1;24r')

        WRITE(5,113) ESC
 113    FORMAT('+',A1,'[24;1H')

        CALL STPTIM('SAMPLE')
        CALL STRTIM

        CALL CLOSE(3)
C                                                    Stop tank
        CALL TNKFRQ(0.,MOTOR)

        TYPE *
        TYPE *,'***** CALCULATING CONSTANTS FOR THEORETICAL FIT *****'
        TYPE *
C                                                    Calculate parameters for
C                                                     theoretical fit
        CALL FIT(NUM,TIME,CONC,B)

        BGNCNC = B(1)+B(4)
        FNLCNC = B(1)+B(3)*TIME(NUM)+B(4)

        WRITE(1,117) BGNCNC,B(2),B(3),FNLCNC
        WRITE(5,117) BGNCNC,B(2),B(3),FNLCNC
 117    FORMAT(' ; Initial Concentration - ',F5.1,' mG/L',/,
     +         ' ; Transfer Rate         - ',F5.3,' 1/min',/,
     +         ' ; Production Rate       - ',F5.3,' mG/(L*min)',/,
     +         ' ; Final Concentration   - ',F5.1,' mG/L',/,' ;')

        CALL STPTIM(' FIT  ')

        RATE = 2.
        MODE = 1
        RNSCNC = 0.0
```

```
 15     CONTINUE

        TYPE *
        TYPE *,'Are you going to rinse the cloth (Y,N)? '

        CALL WTCHAR(ANS)
C                                                  Start Circulating Pump
        CALL DTOA(1,3.1)

        IF (ANS.EQ.'Y'.OR.ANS.EQ.'y') THEN

 16       CONTINUE
C                                                  Sample voltage of
C                                                   laser through tank
C                                                   of rinse water
          TYPE *
          TYPE*,'Measure laser beam power through tank and rinse water:'
          TYPE *
          CALL FSAM(VAL,VOLT,SCHAN,NCHAN,RATE,N,MODE)

C                                                  Convert voltage to
C                                                   millivatts of power
C                                                   using known constants
          CALL CALPOW(POWER,VOLT)
          RATIO = POWER/TNKPWR
C                                                  Calculate concentration
C                                                   of lint in rinse water
          IF (RATIO.GT.0.0) THEN
            RNSCNC = LOG10(RATIO)*(CONST)
          ELSE
            RNSCNC = 0.0
          END IF

 17       CONTINUE
C                                                  Print Results
          TYPE *
          WRITE(5,118) RNSCNC
 118      FORMAT (' ; Rinse Concentration   - ',F5.1,' mG/L')

          TYPE 130
 130      FORMAT(/' Acceptable (Y/N) ? '$)
          CALL WTCHAR(ANS)
          IF (ANS.EQ.'N'.OR.ANS.EQ.'n') GOTO 16
          IF (ANS.NE.'Y'.AND.ANS.NE.'y') GOTO 17

        ELSE IF (ANS.NE.'N'.AND.ANS.NE.'n') THEN
          GOTO 15
        END IF

 20     CONTINUE
C                                                  Sample voltage of
C                                                   laser through tank
C                                                   of drain water
```

```
      TYPE *
      TYPE *,'Measure laser beam power through tank and drain water:'
      TYPE *
      CALL FSAM(VAL,VOLT,SCHAN,NCHAN,RATE,N,MODE)

C                                                 Convert voltage to
C                                                  millivatts of power
C                                                  using known constants
      CALL CALPOW(POWER,VOLT)
      RATIO = POWER/TNKPWR
C                                                 Calculate concentration
C                                                  of lint in drain water
      IF (RATIO.GT.0.0) THEN
        DRNCNC = LOG10(RATIO)*(CONST)
      ELSE
        DRNCNC = 0.0
      END IF
C                                                 Print Results
      TYPE *
      WRITE(5,119) RNSCNC,DRNCNC
 119  FORMAT (' ; Rinse Concentration   - ',F5.1,' mG/L',/,
     +        ' ; Drain Concentration   - ',F5.1,' mG/L',/,
     +        ' ;')

 25   CONTINUE

      TYPE 130
      CALL WTCHAR(ANS)
      IF (ANS.EQ.'N'.OR.ANS.EQ.'n') GOTO 20
      IF (ANS.NE.'Y'.AND.ANS.NE.'y') GOTO 25
C                                                 Stop Circulating Pump
      CALL DTOA(1,0.0)

      TYPE *
      WRITE(1,119) RNSCNC,DRNCNC
      WRITE(1,107)
 107  FORMAT(' ; Time (min)  Lint Concentration (mG/L)',/,
     +       ' ; ----------  ------------------------',/,' ;')

      DO 30 I=1,NUM
        WRITE(1,120)J,TIME(I),CONC(I)
 120    FORMAT('C',I1,2G15.7)
 30   CONTINUE

      J = J+1

      CALL MODPLT(NUM,TIME,CONC,B)

      DO 31 I=1,NUM
        WRITE(1,121)J,TIME(I),CONC(I)
 121    FORMAT('C',I1,2G15.7)
 31   CONTINUE

      XMAX = TIME(NUM)
```

```
      CALL CFILE(FILEN,XMAX,0)

      CALL BEL(1)

      TYPE 123
123   FORMAT(' Hit "RETURN" to Continue ..... '$)
      READ(5,*)

      CALL EXIT
      END


      SUBROUTINE CALPOW(POWER,V2)

C--- Calibration for power meter -----

      C1 = 0.95E-04
      C2 = 0.011
      C3 = 97.1
      C4 = -0.770

      IF (V2.GT.0.0) THEN
        V1 = (V2-C4)/C3
        POWER = (V1-C2)/C1
      ELSE
        POWER = 0.0
      END IF

      RETURN
      END


C-----------------------------------------------------------------------
C
      SUBROUTINE STRTIM
C
C-----------------------------------------------------------------------
C
C     STRTIM  STPTIM are subroutines to keep track of elapsed time
C      between events. STRTIM is called to start timing and STPTIM
C      is called to stop timing and to print elapsed time.
C
C     David J. Fanson                 29-MAY-87
C
C-----------------------------------------------------------------------

C--------- Variables in Common TIM ---------

      INTEGER ITIME1(8)
      COMMON /TIM/ ITIME1
C                                         Get first time
      CALL GETTIM(ITIME1)
```

```
      RETURN
      END


C-----------------------------------------------------------------------
C
      SUBROUTINE STPTIM(NAME)
C
C-----------------------------------------------------------------------

C--------- Variables in Common TIM ---------

      INTEGER ITIME1(8)
      COMMON /TIM/ ITIME1

C--------- Local  Other Variables ---------

      CHARACTER*6 NAME
      INTEGER MIN,DIF(8),ITIME2(8)
      REAL SEC
C                                               Get second time
      CALL GETTIM(ITIME2)
C                                               Find difference between
C                                                first and second time
C                                                calls
      DO 10 I=1,7
        DIF(I) = ITIME2(I)-ITIME1(I)
 10   CONTINUE
C                                               Calculate time
C                                                difference in minutes
C                                                and seconds

      SEC = DIF(6)*1.+DIF(7)/60.
      MIN = DIF(4)*60+DIF(5)

      IF (SEC.LT.0.0) THEN
        SEC = SEC+60.0
        MIN = MIN-1
      END IF
C                                               Write routine name and
C                                                time
      WRITE(5,100) NAME,MIN,SEC
 100  FORMAT(/' Total CPU time used in ',A6,' routine = ',I3,
     + ' minutes ',F6.3,' seconds')

      RETURN
      END


C-----------------------------------------------------------------------
C
      SUBROUTINE LSCRL(IU,N)
C
C-----------------------------------------------------------------------
```

```
C
C       LSCRL causes the unit number device (IU) to scroll "N"
C        lines.
C
C-----------------------------------------------------------------------
C
C       Written by:
C                       David J. Fanson
C                       Michigan State University
C
C       Last Modification:
C                       July 16, 1987
C
C-----------------------------------------------------------------------

        INTEGER IU,N
C                                                     Write "N" blank lines
C                                                      to unit "IU"
        DO 10 I=1,N
          WRITE(IU,*)
 10     CONTINUE

        RETURN
        END



C-----------------------------------------------------------------------
C
        SUBROUTINE SCROLL(TOP,BOTTOM)
C
C-----------------------------------------------------------------------
C
C       SCROLL sets the scrolling region on the terminal.
C
C-----------------------------------------------------------------------
C
C       Written by:
C                       David J. Fanson
C                       Michigan State University
C
C       Last Modification:
C                       July 16, 1987
C
C-----------------------------------------------------------------------

C--------- local  other variables ---------

        BYTE ESC
        INTEGER TOP,BOTTOM

        ESC = "033
C                                                     Set scrolling region
        WRITE(5,100) ESC,TOP,BOTTOM
```

```
 100    FORMAT('+',A1,'[',I2,';',I2,'r')

        RETURN
        END

C-----------------------------------------------------------------------
C
        SUBROUTINE CSCRN
C
C-----------------------------------------------------------------------
C
C       CSCRN clears the whole srceen.
C
C-----------------------------------------------------------------------
C
C       Written by:
C                       David J. Fanson
C                       Michigan State University
C
C       Last Modification:
C                       July 16, 1987
C
C-----------------------------------------------------------------------

C--------- local  other variables ---------

        BYTE ESC

        ESC = "033
C                                                  Clear screen
        WRITE(5,110) ESC
 110    FORMAT('+',A1,'[2J')
        RETURN
        END


C-----------------------------------------------------------------------
C
        SUBROUTINE BEL(N)
C
C-----------------------------------------------------------------------
C
C       BEL will immediately beep the bell of the calling terminal.
C
C       Peter J. McKinney               7-aug-1986
C
C-----------------------------------------------------------------------

        BYTE BELL         ! define BEL as type BYTE
        BELL='7'O         ! octal code for the bell character (BEL)

C                               Ring bell N times
        DO 10 I=1,N
C                               Write to user's terminal
```

```
C                                 (logical unit 5 by default)
C                                 as a character
         WRITE(5,100)BELL
 100     FORMAT('+',A1)
 10     CONTINUE

        END


C-----------------------------------------------------------------------
C
        SUBROUTINE WTCHAR(CHAR)
C
C-----------------------------------------------------------------------
C
C       WTCHAR returns the the first character in the keyboard buffer
C       in BYTE form.  If no character is available program execution
C       is suspended until a character is entered.
C
C       Written by Peter J. McKinney  --  Michigan State University
C
C       Reference:  Engineering Computer Facility  --  M.S.U.
C
C       Last Modification: August 11, 1986
C
C-----------------------------------------------------------------------

        INTEGER DSW                 ! directive status word
        CHARACTER*1 CHAR            ! character input
        LOGICAL*1 ISB(4)            ! I/O status block as byte array
        INTEGER IOST(2)                     ! I/O status block as integer
array
        INTEGER PRAMIN(6)           ! device dependant paramater array
        BYTE CHARS(2)               ! buffer count call  result

        EQUIVALENCE (IOST(1),ISB(1))

                                    ! integer  byte versions of I/O status

        CALL GETADR(PRAMIN(1),CHAR)

                                    ! get address of WTCHAR
        PRAMIN(2) = 1               ! set to read one character

        CALL WTQIO("1010.OR."20,5,1,,IOST,PRAMIN,DSW)

                                    ! read all bits with no echo

        IF((DSW.NE.1).OR.(IOST(1).NE.1)) CALL ERRORS(ISB,DSW)

100     RETURN
        END
```

```
C------------------------------------------------------------------
C
      SUBROUTINE ERRORS(ISB,DSW)
C
C------------------------------------------------------------------
C
C     ERRORS returns the values of the Device Status Word and the
C     I/O Status Block if an error occurs durring an I/O operation.
C
C------------------------------------------------------------------

      INTEGER DSW
      LOGICAL*1 ISB(4)        ! I/O status block as a byte array

      WRITE(5,100)ISB(1),DSW
100   FORMAT(/,' ISB = ',I8,/,' DSW =   ',I6)

      CALL EXIT
      END
C
C------------------------------------------------------------------
C
C     Subroutine IPOKE and Function IPEEK
C
C     PEEK  POKE routines written for the DEC PDP11/73 running under
C      RSX-11M-PLUS.  These functions  subroutines appear the same
C      to FORTRAN as the RT-11 calls of the same name.  KCOM is a
C      common area set up by RSX that resides directly over the
C      device registers (addresses 160000 to 177777 octal). The
C      following routines set up arrays (IREG  IREGB) that coincide
C      with this comman area. By reading from and writing to this
C      array, the the program is actually PEEKing and POKEing the
C      coincident device registers.
C
C
C     Routines written by Peter J. Mckinney  --
C                                     Michigan State University
C
C     Adapted from routines by Dr. C. Radcliffe  --
C                                     Michigan State University
C
C     Last Modification:  July 10, 1986
C
C------------------------------------------------------------------

      SUBROUTINE IPOKE(ICSR,IVAL)

C------------------------------------------------------------------
C
C     This subroutine puts IVAL into a device register at ICSR
C
C------------------------------------------------------------------

C                                     Define word array residing
```

```
C                                       over device registers by
C                                       declaring common to KCOM

      COMMON /KCOM/ IREG('70000'0:'77776'0)

C                                      POKE by placing value in array
C                                       (divide ICSR by 2 since we are
C                                       using a word array and
C                                       addresses increment by bytes)
      IREG(ICSR/2) = IVAL
      RETURN
      END

C-----------------------------------------------------------------------

      FUNCTION IPEEK(ICSR)

C-----------------------------------------------------------------------

C     This function returns the contents of a device register at ICSR

C-----------------------------------------------------------------------

C                                      Define word array residing
C                                       over device registers by
C                                       declaring common to KCOM

      COMMON /KCOM/ IREG('70000'0:'77776'0)

C                                      PEEK by reading value from
C                                       array. (divide ICSR by 2 since
C                                       we are using a word array and
C                                       addresses increment by bytes)
      IPEEK = IREG(ICSR/2)
      RETURN
      END


C**********************************************************************
C
      SUBROUTINE TNKFRQ(FREQ,MOTOR)
C
C**********************************************************************
C
C     TNKFRQ sets the speed of the DC motor that opperates the linear
C      tank. The speed is entered to the computer as a frequency then
C      converted to a voltage using channel 0 of the DEC AXV11-C D/A
C      converter.
C
C**********************************************************************
C
C     Program written by:
C             David J. Fanson
C             Michigan State University
```

```
C
C       Last modification: August 14, 1987
C
C***********************************************************************
C
C       Variable List:
C
C               FREQ    Frequency
C               MOTOR   Motor number
C                               1 = Linear Tank
C                               2 = Washing Machine
C               RPM     Desired RPM of drive system
C               VOLTS   Voltage produced by D/A board
C
C***********************************************************************
C
        RPM = FREQ*60.
        VOLTS = 0.0
C                                       Determine if desired speed is
C                                        within the motor's range
C
        IF (MOTOR.EQ.1) THEN
          IF (RPM.LE.190.0.AND.RPM.GE.0.0) THEN
C
C                                       Calculate output voltage using
C                                        constants found from a 3rd
C                                        order polynomial fit of a set
C                                        of calibration points
            CALL CAL1(VOLTS,RPM)
          ELSE
            TYPE *,' '
            TYPE *,'Speed out of range !'
          ENDIF

        ELSE IF (MOTOR.EQ.2) THEN
C                                       Determine if desired speed is
C                                        within the motor's range
C
          IF (RPM.LE.360.0.AND.RPM.GE.0.0) THEN
C
C                                       Calculate output voltage using
C                                        constants found from a 3rd order
C                                        polynomial fit of a set of
C                                        calibration points
C           RPM = RPM*8.056
            RPM = RPM*12.66
            CALL CAL2(VOLTS,RPM)
          ELSE
            TYPE *
            TYPE *,'Speed out of range !'
          ENDIF

        ELSE
          TYPE *
```

```
          TYPE *,'Invalid Motor  '
        ENDIF
C                                           Set channel 0 of the D/A board
C                                            to disired voltage
        CALL DTOA(0,VOLTS)

        RETURN
        END


C***********************************************************************
C
        SUBROUTINE CAL1(Y,X)
C
C***********************************************************************
C
C       Subroutine CAL1 is 3rd order polynomial fit to a set of
C        calibration test points
C
C***********************************************************************
C
C       Written by:
C               David J. Fanson
C
C       Last Modification: September 1986
C
C***********************************************************************
C
C       Variables:
C
C       C1-C4   Constants for fit
C       X       Independent variable
C       Y       Dependent variable
C
C***********************************************************************
C
C                                           Set constant to values found in
C                                            polynomial fit
        IF (X.EQ.0.0) THEN
         Y = 0.0
        ELSE
         C1 = 0.168121677E-07
         C2 =-0.279793403E-05
         C3 = 0.482184552E-01
         C4 = 0.195198685E+00
C                                           Solve for the value of the
C                                            dependent varible using the
C                                            current value of the
C                                            independent variable

         Y = X**3*C1 + X**2*C2 + X*C3 + C4
        ENDIF
        RETURN
        END
```

```
C
C***********************************************************************
C
      SUBROUTINE CAL2(Y,X)
C
C***********************************************************************
C
C     Subroutine CAL2 is 3rd order polynomial fit to a set of
C      calibration test points
C
C     Written by:
C             David J. Fanson
C
C     Last Modification: April 1987
C
C***********************************************************************
C
C     Variables:
C
C     C1-C4   Constants for fit
C     X       Independent variable
C     Y       Dependent variable
C
C***********************************************************************
C
C                                Set constant to values found in
C                                 polynomial fit
      IF (X.EQ.0.0) THEN
        Y = 0.0
      ELSE
        C1 = 0.0
        C2 = 0.243787728E-07
        C3 = 0.455234572E-02
        C4 = 0.198532641E+00
C                                Solve for the value of the
C                                 dependent varible using the
C                                 current value of the
C                                 independent variable
C
        Y = X**3*C1 + X**2*C2 + X*C3 + C4
      ENDIF

      RETURN
      END


C-----------------------------------------------------------------------
C
      SUBROUTINE DTOA(CHNL,VOLT)
C
C-----------------------------------------------------------------------
C
```

```
C       DTOA is a generic subroutine that sets channels 0 or 1 of the
C        DEC AXV11-C D/A convertor to a selected voltage.
C
C-----------------------------------------------------------------------
C
C       Written by:
C               David J. Fanson
C               Michigan State University
C
C       Last Modification: August 14, 1987
C
C-----------------------------------------------------------------------
C
C       Variable List:
C
C               ADRESS  Octal address of register for D/A channels
C               CHNL    D/A channel to be altered
C               VALUE   Integer value to be POKEd into the address
C               VOLT    Desired voltage of output
C
C-----------------------------------------------------------------------
C
        INTEGER ADRESS,CHNL,VALUE
C                                                       Check for errors in
C                                                        input and exit if
C                                                        found
        IF (CHNL.NE.0.AND.CHNL.NE.1) THEN
          TYPE *,'******* Illegal channel   *******'
          GOTO 15
        ELSEIF (VOLT.GT.10.0.AND.VOLT.LT.0.0) THEN
          TYPE *,'******* Voltage out of bounds *******'
          GOTO 15
        ELSE
C                                                       Set adress and POKE
C                                                        value from input for
C                                                        channels 0 or 1
          IF (CHNL.EQ.0) THEN
C                                                       Address of channel 0
            ADRESS = "170404
          ELSE
C                                                       Address of channel 1
            ADRESS = "170406
          ENDIF
C                                                       Calculate the POKE
C                                                        value from the desired
C                                                        voltage using a linear
C                                                        calibration
          VALUE = VOLT*409.6+0.5
C                                                       Make sure VALUE is
C                                                        within range
          IF (VALUE.GT.4095) VALUE = 4095
          IF (VALUE.LT.0) VALUE = 0
C                                                       POKE the calculated
C                                                        value into the
```

```
C                                                     selected address
          CALL IPOKE(ADRESS,VALUE)
        ENDIF

 15     CONTINUE
        RETURN
        END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE FSAM(VOLT,VLTAVE,SCHAN,NCHAN,RATE,NSAMPL,MODE)
C
C       This subroutine is a basic sampling routine. The starting
C        channel, the number of channels to be sampled, the sampling
C        rate, and the number of samples to be taken are passed in.
C        An array of integer values in the range of 0-4095
C        corresponding to 0.0 - 10.0 volts DC.
C
C       Arguments:
C
C       Input-  SCHAN   Starting channel of A/D board
C               NCHAN   Number of channels sampled
C               RATE    Sampling rate
C               NSAMPL  Number of samples taken
C               MODE    Sampling starting mode
C                       (0 = instant start,1 = wait)
C
C       Output-         VOLT    Array of voltages from sampling
C               VLTAVE  Average of voltages
C
C***********************************************************************
C
C       Linking procedure:
C
C        Ribbit $ TKB
C        TKB> filename = filename, FSAM, LNTUTL
C        TKB> @[1,54]KCOM
C
C***********************************************************************
C
C       Written by:
C        David J. Fanson
C
C       Michigan State University
C
C       Last modified on August 26, 1987
C
C************************************************************************
C
C       Variable list:
C
C       ADSWP     Routine to do A/D sweep
C
C       BUF       Buffer array that hold sampled data
C
```

```
C       IND        Error code
C
C***********************************************************************
C
C -----  VARIABLE DECLARATIONS ----------------------------------------
C
        INTEGER BUF(256),SCHAN
        REAL VOLT(NSAMPL),RATE
        CHARACTER ANS
C
C  ----------------------  Start A/D sweep ----------------------------
C
        SUM = 0.

        IF (MODE.EQ.1) THEN
 20       TYPE 100
 100      FORMAT(' Hit "S" to start sampling : '$)
          CALL VTCHAR(ANS)
          IF(ANS.NE.'S'.AND.ANS.NE.'s') GOTO 20
          CALL BEL(1)
          TYPE *,'************** SAMPLING ************'
        ELSE IF (MODE.NE.0) THEN
          TYPE *,'***** Illegal mode ******'
          GOTO 50
        END IF

        CALL ADSWP(BUF,RATE,SCHAN,NCHAN,NSAMPL,IND)

        IF (IND.NE.0) THEN
          TYPE 110,IND
 110      FORMAT(' ***** ERROR ',I1,' IN SAMPLING ROUTINE *****')
          CALL BEL(2)
        ELSE

          DO 40 I=1,NSAMPL
            CALL ADCAL(VOLT(I),BUF(I))
            SUM = VOLT(I)+SUM
 40       CONTINUE

          IF (SUM.EQ.0.) THEN
            VLTAVE = 0.
          ELSE
            VLTAVE = SUM/NSAMPL
          END IF

        END IF

 50     CONTINUE

        RETURN
        END

C
C----------------------------------------------------------------------
```

```
C
C       ADCAL is a subroutine to convert integer value into a voltage
C
C       Written by: David J. Fanson
C
C-----------------------------------------------------------------------
C
        SUBROUTINE ADCAL(Y,X)
        INTEGER X
        A = 1./409.6
        Y = X*A
        RETURN
        END

C-----------------------------------------------------------------------
C
        SUBROUTINE ADSWP(BUF,FREQ,SCHAN,NCHAN,N,ERR)
C
C-----------------------------------------------------------------------
C
C       ADSWP is FORTRAN subroutine that uses the A/D converter
C        (AXV11-C) and Programable Real-Time Clock (KWV11-C) to collect
C        a buffer of integer sample data. The subroutine is written to
C        allow the first sample at each time step to be started by the
C        Programable Clock and sub subsequent samples to be taken
C        individually by the A/D converter.
C
C-----------------------------------------------------------------------
C
C       Written by:
C               David J. Fanson
C               Michigan State University
C
C       Last Modification: February 22, 1987
C
C-----------------------------------------------------------------------
C
C       Variable List:
C
C               BUF     Array of integers from sampling
C
C               DBR     ADDRESS of Data Buffer Register
C
C               CSR     ADDRESS of Control/Status Register
C
C               ERR     Error flag
C
C               FREQ    Desired sampling frequency
C
C               N       Number of samples per channel
C
C               NCHAN   Number of channels to sample
C
C               PERIOD  Sampling period
```

```
C
C              SCHAN   Starting channel
C
C              TOTAL           Total number of samples
C
C              VALUE   POKE value for CSR
C
C-----------------------------------------------------------------------
C
C                                         Declare variables
C
      INTEGER CSR,DBR,VALUE,SCHAN,NCHAN,N,ERR,SAMPLE,TOTAL,SET
      INTEGER BUF(256)
      K = 0
      ERR = 0
      TOTAL = N*NCHAN
C                                         Set addresses of registers
      CSR = "170400
      DBR = "170402
C                                         Find value of starting channel
C                                          and move it over 8 places
      SET = (SCHAN*2**8).OR."40
C                                         Enter starting channel
      CALL IPOKE(CSR,SET)
C                                         Start clock
      CALL CSTART(FREQ)
C                                         Top of sampling loop
 10   CONTINUE
      IF (ERR.EQ.0.AND.K.LE.TOTAL) THEN
        K = K+1
 20     CONTINUE
C                                         Check to see if sample is
C                                          complete
        VALUE = IPEEK(CSR)
        IF ((VALUE.AND."200).EQ."200) THEN
C
C                                         If complete then set BUF
C                                          equal to the value of the
C                                          DBR
          BUF(K) = IPEEK(DBR)
C                                         Gather rest of channels using
C                                          function SAMPLE
C
          DO 30 I=SCHAN+1,SCHAN+NCHAN-1
            K = K+1
            BUF(K) = SAMPLE(I)
 30       CONTINUE
C                                         Set A/D to starting channel
          CALL IPOKE(CSR,SET)
C                                         Wait for sampling intervule
          CALL CWAIT(ERR)
        ELSE
          GOTO 20
        END IF
```

```
        GOTO 10
      END IF
C                                       Shut off clock
      CALL CSTOP
      RETURN
      END
C-----------------------------------------------------
C
      SUBROUTINE CSTART(FREQ)
C
C-----------------------------------------------------
C
C     CSTART and CSTOP are FORTRAN subroutines that
C      start and stop the KWV11-C Programmable Real-Time
C      Clock. Subroutine CSTART sets up the Clock to
C      generate a DONE bit at a user specified rate.
C
C-----------------------------------------------------
C
C     Written By :
C             David J. Fanson
C             Michigan State University
C
C     Last Modification : January 29, 1987
C
C-----------------------------------------------------
C
C     Variable List :
C
C             BPR     ADDRESS of Buffer/Preset Register
C
C             CSR     ADDRESS of Control/Status Register
C
C             FREQ    Overflow Frequncy
C
C             RATE    Clock Rate
C
C             SET     Clock Rate Bit Settings
C
C             STORE   Real Clock Count
C
C             VALUE   Clock Count
C
C-----------------------------------------------------
C
C                                       Declare variables
C
      INTEGER BPR,CSR,SET,VALUE
      REAL FREQ,RATE,STORE
C                                       Set constants
      CSR = "170420
      BPR = "170422
C                                       Values for maximium
C                                        Clock Rate
```

```
      RATE = 1.E+06
      SET = "12
C                                      Calculate Clock Count
C                                       using maximium Clock Rate
      STORE = RATE/FREQ
 10   CONTINUE
C                                      IF Clock Count value is
C                                       greater than 77777 OCTAL
C                                       THAN lower Clock Rate
C                                       until it is
      IF (STORE.GT."77777) THEN
        STORE = STORE/10.0
        RATE = RATE/10.0
        SET = SET+"10
        GOTO 10
      END IF
C                                      Calculate Integer Clock Count
      VALUE = -1*(STORE+0.5)
C                                      Calculate actual Overflow
C                                       frequency
      FREQ = -1.*RATE/VALUE
C                                      POKE the Clock Count into BPR
      CALL IPOKE(BPR,VALUE)
C                                      POKE Clock Setting into CSR
      CALL IPOKE(CSR,SET)
C                                      Flip the GO Bit to start the
C                                       Clock
      CALL IPOKE(CSR,(SET.OR."1))
      RETURN
      END
C
C-------------------------------------------------------
C
      SUBROUTINE CSTOP
C
C-------------------------------------------------------
C
C     CSTOP is a subroutine to stop the KWV11-C
C      Programmable Real-Time Clock.
C
C-------------------------------------------------------
C
      INTEGER CSR
      CSR = "170420
C                                      Set CSR to zero
      CALL IPOKE(CSR,0)
      RETURN
      END
C
C-------------------------------------------------------
C
      SUBROUTINE CWAIT(ERR)
C
C-------------------------------------------------------
```

```
C
C       CWAIT is a FORTRAN subroutine that waits for the
C        DONE bit on the KWV11-C Programmable Real-Time
C        Clock to be set.
C
C-------------------------------------------------------
C
C       Written By :
C               David J. Fanson
C               Michigan State University
C
C       Last Modification : January 29, 1987
C
C-------------------------------------------------------
C
C       Variable List :
C
C               BIT1    Check for bit 1
C
C               BIT7    Check for bit 7
C
C               BIT12   Check for bit 12
C
C               CSR     ADDRESS of Control/Status Register
C
C               ERR     Error Flag
C
C               VALUE   Test value
C
C-------------------------------------------------------
C
C                                       Declare variables
C
        INTEGER BIT1,BIT7,BIT12,CSR,ERR,VALUE
C
C                                       Set constants
        ERR = 0
        CSR = "170420
C                                       PEEK value of CSR
        VALUE = IPEEK(CSR)
C                                       Find status of Bits
C                                        1 and 12
        BIT1 = VALUE.AND."1
        BIT12 = VALUE.AND."10000
C                                       IF Bit 1 is not set
C                                        Error Flag equals 1
C                                        (Clock not started)
        IF (BIT1.NE."1) THEN
          ERR = 1
C                                       IF Bit 12 is set
C                                        Error Flag equals 2
C                                        (Flag Overrun)
        ELSE IF (BIT12.EQ."10000) THEN
          ERR = 2
```

```
      ELSE
 10     CONTINUE
C                                       Find status of Bit 7
        BIT7 = IPEEK(CSR).AND."200
C                                       IF Bit 7 is set
        IF (BIT7.EQ."200) THEN
C                                         THEN
C                                               Clear Bit 7
C                                               and exit
          VALUE = IPEEK(CSR).AND."177577
          CALL IPOKE(CSR,VALUE)
C                                       ELSE
C                                               Continue Loop until
C                                               Bit 7 is set
        ELSE
          GOTO 10
        END IF
      END IF
      RETURN
      END


C-------------------------------------------------------
C
      INTEGER FUNCTION SAMPLE(CHNL)
C
C-------------------------------------------------------
C
C     SAMPLE is a FORTRAN function that uses the A/D
C      converter (AXV11-C) to collect a single sample
C      from a specified A/D channel.
C
C-------------------------------------------------------
C
C     Written By :
C             David J. Fanson
C             Michigan State University
C
C     Last Modification : February 6, 1987
C
C-------------------------------------------------------
C
C     Variable List :
C
C             DBR     ADDRESS of Data Buffer Register
C
C             CSR     ADDRESS of Control/Status Register
C
C             CHNL    A/D channel to be sampled
C
C             VALUE   POKE value for CSR
C
C-------------------------------------------------------
C
```

```
C                                               Declare variables
       INTEGER CHNL,CSR,DBR,VALUE
C                                               Set addresses of registers
       CSR = "170400
       DBR = "170402
C                                               Find value of channel and
C                                                move it over 8 places
       VALUE = CHNL.AND."17
       VALUE = VALUE*2**8
C                                               POKE channel   into CSR
       CALL IPOKE(CSR,VALUE)
C                                               Start Sample
       VALUE = VALUE.OR."1
       CALL IPOKE(CSR,VALUE)
 10    CONTINUE
C                                               Check to see if sample is
C                                                complete
       VALUE = IPEEK(CSR)
       IF ((VALUE.AND."200).EQ."200) THEN
C
C                                               If complete then set SAMPLE
C                                                equal to the value of the
C                                                DBR
         SAMPLE = IPEEK(DBR)
       ELSE
         GOTO 10
       END IF
C                                               Clear the CSR
       CALL IPOKE(CSR,0)
       RETURN
       END


C-----------------------------------------------------------------------

       SUBROUTINE FIT(N,X,Y,A,R)

C-----------------------------------------------------------------------

       REAL X(N),Y(N),A(4)
       REAL E,DX,R
       INTEGER I,N,M,MM,MAX

       E = 2.0
       MAX = 10
       DX = X(N)-X(N-1)

       CALL OFFSET(N,X,Y,A)

C      TYPE *,N,A

C      DO 10 E=0.5,5.0,0.1

       A2 = 0.3
```

```
      A(2) = A2
      I = 0

 30   CONTINUE

        I = I+1
        MM = M

        M = (-LOG(E/100)/(A(2)*DX))+0.5

        IF (M.GT.N) THEN
          M = (-LOG(E/100)/(A2*DX))+0.5
          MM = M
        ENDIF

C       TYPE *,M,X(M)

        CALL LINREG(M,N,X,Y,A,R)

        CALL NLREG(N,X,Y,A,R)

C       TYPE *,R,A

      IF (M.NE.MM.AND.I.LT.MAX) GOTO 30

C     TYPE *,E,R,A(1),A(2),A(3)

 10   CONTINUE

      RETURN
      END


C-----------------------------------------------------------------

      SUBROUTINE OFFSET(N,X,Y,A)

C-----------------------------------------------------------------

      REAL X(N),Y(N),A(4)
      REAL MAXSLP,SLP
      INTEGER I,N,MAX

      MAXSLP = 0.0
      DX = X(N)-X(N-1)
C                                              Find maximum slope of
C                                               data and it's location
      DO 10 I=2,N-1
        SLP = (Y(I+1)-Y(I))/DX

        IF (SLP.GT.MAXSLP) THEN
          MAX = I
          MAXSLP = SLP
```

```
          END IF

 10     CONTINUE
C                                                  Calculate offset
        A(4) = X(MAX)-Y(MAX)/MAXSLP
C                                                  Correct data with
C                                                   respect to offset
        DO 20 I=MAX,N
          X(I-MAX+2) = X(I)-A(4)
          Y(I-MAX+2) = Y(I)
 20     CONTINUE
C                                                  Update number of data
C                                                   points because of
C                                                   data correction
        N = N-MAX+2

        RETURN
        END


C-----------------------------------------------------------------------

        SUBROUTINE LINREG(M,N,X,Y,A,R)

C-----------------------------------------------------------------------

        REAL X(N),Y(N),A(4)
        REAL SUMX,SUMY,SUMX2,SUMXY,SR,ST,R
        INTEGER I,M,N,NN

        SUMX = 0.0
        SUMY = 0.0
        SUMX2 = 0.0
        SUMXY = 0.0

        DO 10 I=M,N
          SUMX = SUMX+X(I)
          SUMY = SUMY+Y(I)
          SUMX2 = SUMX2+X(I)**2
          SUMXY = SUMXY+X(I)*Y(I)
 10     CONTINUE

        NN = N-M+1
        A(3) = (NN*SUMXY-SUMX*SUMY)/(NN*SUMX2-SUMX**2)
        A(1) = SUMY/NN-A(3)*SUMX/NN

        SR = 0.0
        ST = 0.0

        DO 20 I=M,N
          SR = SR+(Y(I)-A(1)-A(3)*X(I))**2
          ST = ST+(Y(I)-SUMY/NN)**2
 20     CONTINUE
```

```
      R = SQRT((ST-SR)/ST)

      RETURN
      END




C-----------------------------------------------------------------------

      SUBROUTINE NLREG(M,XI,YI,A,R)

C-----------------------------------------------------------------------

      REAL XI(M),YI(M),A(4)
      REAL D,E,F,Z,SUMZ2,SUMZD,SUMY,DA,YY,ST,SR,R
      INTEGER I,M,MAX

      F(I) = A(1)*(1-EXP(-A(2)*XI(I)))+A(3)*XI(I)

      E = 0.01
      MAX = 100
      I = 0
C                                               Perform nonlinear
C                                                regression using
C                                                Gauss-Newton Method to
C                                                solve for A(2) in the
C                                                function F
 10   CONTINUE

        I = I+1

        SUMZ2 = 0.0
        SUMZD = 0.0

        DO 20 I=1,M
          Z = A(1)*XI(I)*EXP(-A(2)*XI(I))
          D = YI(I)-F(I)
          SUMZ2 = SUMZ2+Z**2
          SUMZD = SUMZD+Z*D
 20     CONTINUE

        DA = SUMZD/SUMZ2
        A(2) = A(2)+DA
C                                               Stop iterating after DA
C                                                becomes less than E
C                                                percent of A(2) or MAX
C                                                iterations have been
C                                                completed

      IF (ABS(DA*100.0/A(2)).GE.E.AND.I.LE.MAX) GOTO 10

C                                               Calculate correlation
```

```
C                                                       coefficient R
      SUMY = 0.0

      DO 30 I=1,M
        SUMY = SUMY+YI(I)
 30   CONTINUE

      YY = SUMY/M
      ST = 0.0
      SR = 0.0

      DO 40 I=1,M
        ST = ST+(YI(I)-YY)**2
        SR = SR+(YI(I)-F(I))**2
 40   CONTINUE

      R = SQRT(abs(1-SR/ST))
C                                                      If subroutine stoped
C                                                       because of reaching
C                                                       maximum number of
C                                                       iterations, create a
C                                                       value for R that is
C                                                       impossible and can be
C                                                       used as a flag.
      IF (I.EQ.MAX) R = 1.111111

      RETURN
      END



C----------------------------------------------------------------------
C
      SUBROUTINE TXT(DWELL,FREQ,NUM)
C
C----------------------------------------------------------------------
C
C     Written by:
C                     David J. Fanson
C                     Michigan State University
C
C     Last Modification:
C                     August 24, 1987
C
C----------------------------------------------------------------------

      INTEGER DWELL,NUM
      REAL TOTAL,FREQ,LENGTH
      CHARACTER ANS
      CHARACTER*10 MOTION
      CHARACTER*40 TEXT,LOAD,BLADE,OPER,TITLE

      TITLE = '       LINT SAMPLING AND ANALYSIS        '
```

```
 10     CONTINUE
        TYPE *
        TYPE *,'Enter Run Discription for Data File (<40 char) : '
        ACCEPT 210,TEXT
 210    FORMAT(A40)
        TYPE *
        TYPE *,'Enter Name of Test Operator (<40 char) : '
        ACCEPT 210,OPER
 20     CONTINUE
        TYPE *
        WRITE(5,120) TEXT,OPER
 120    FORMAT(' ;',/,
     +         ' ; Run Description   - ',A40,/,
     +         ' ; Data Collected By - ',A40)
        TYPE 130
 130    FORMAT(/' Acceptable (Y/N) ? '$)
        CALL WTCHAR(ANS)
        IF (ANS.EQ.'N'.OR.ANS.EQ.'n') GOTO 10
        IF (ANS.NE.'Y'.AND.ANS.NE.'y') GOTO 20
        WRITE(1,120) TEXT,OPER
C                                                   Enter sampling
C                                                    conditions

 40     CONTINUE
        TYPE *
        TYPE *,'Enter Load Discription (<40 char) : '
        ACCEPT 210,LOAD
        TYPE *,'Enter Agitator Discription (<40 char) : '
        ACCEPT 210,BLADE
 50     CONTINUE
        TYPE *
        WRITE(5,140) LOAD,BLADE
 140    FORMAT(' ;',/,
     +         ' ; Cloth Load - ',A40,/,
     +         ' ; Agitator   - ',A40)
        TYPE 130
        CALL WTCHAR(ANS)
        IF (ANS.EQ.'N'.OR.ANS.EQ.'n') GOTO 40
        IF (ANS.NE.'Y'.AND.ANS.NE.'y') GOTO 50
        WRITE(1,140) LOAD,BLADE
C                                                   Set tank speed
 60     CONTINUE
        TYPE *
        TYPE 150
 150    FORMAT(' Enter Tank Frequency (<3 Hz) : '$)
        READ(5,*) FREQ
C                                                   Check speed

        IF (FREQ.GT.3.0) THEN
          TYPE *
          TYPE *,'****** FREQUENCY TOO HIGH!! ******'
          CALL BEL(3)
          GOTO 60
```

```
        END IF

        TYPE *
        TYPE 160
 160    FORMAT(' Enter Stroke Length (degrees) : '$)
        READ(5,*) LENGTH
C       LENGTH = 10.0
        MOTION = 'Symmetric '

C       TYPE 180
C 180   FORMAT(/' Enter D/A sampling dwell time (sec) : '$)
C       READ(5,*) DWELL
        DWELL = 12
        TYPE 190
 190    FORMAT(/' Enter total time to Sample (min) : '$)
        READ(5,*) TOTAL
        NUM = (TOTAL*60./DWELL)+1.5
        TOTAL = (NUM-1)*DWELL/60.
 90     CONTINUE

        TYPE *
        WRITE(5,170) DWELL,FREQ,NUM,LENGTH,TOTAL,MOTION
 170    FORMAT (' ;',/,
     +  ' ; Dwell Time - ',I4,' sec      ',
     +  'Oscillation Rate - ',F5.2,' Hz',/,
     +  ' ;   Samples  - ',I4,'           ',
     +  'Stroke Length     - ',F5.1,' degrees',/,
     +  ' ; Total Time - ',F4.1,' min      ',
     +  'Agitator Motion  - ',A10)

        TYPE 130
        CALL WTCHAR(ANS)
        IF (ANS.EQ.'N'.OR.ANS.EQ.'n') GOTO 60
        IF (ANS.NE.'Y'.AND.ANS.NE.'y') GOTO 90
        WRITE(1,170) DWELL,FREQ,NUM,LENGTH,TOTAL,MOTION

        CALL CSCRN
        CALL SCROLL(19,24)

        CALL WTITLE(IU,TITLE)

        WRITE(5,120) TEXT,OPER
        WRITE(5,140) LOAD,BLADE
        WRITE(5,170) DWELL,FREQ,NUM,LENGTH,TOTAL,MOTION

        CALL LSCRL(5,8)

        RETURN
        END


C-----------------------------------------------------------------------
C
        SUBROUTINE WTITLE(IU,TITLE)
```

```
C
C-----------------------------------------------------------------------
C
C       VTITLE writes the program title in large print to unit "IU".
C
C-----------------------------------------------------------------------
C
C       Written by:
C                       David J. Fanson
C                       Michigan State University
C
C       Last Modification:
C                       July 16, 1987
C
C-----------------------------------------------------------------------

C--------- local  other variables ---------

        CHARACTER*40 TITLE
        INTEGER IU
        BYTE ESC

        ESC = "033
C                                                   Write title in large
C                                                    print
        WRITE(5,100) TITLE
 100    FORMAT('+',A40)
        WRITE(5,110) ESC
 110    FORMAT('+',A1,' 3')
        WRITE(5,120) TITLE
 120    FORMAT(' ',A40)
        WRITE(5,130) ESC
 130    FORMAT('+',A1,' 4')
        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE OFILE(FNAME,TIX,TIY)
C
C       This subroutine setsup data files to put the collected data
C       into, and supplies pdl file with axies and title information.
C       Last modified on 3/10/86 by Dan Budny
C
C       BLANK   Used to setup file names
C       CB      Used to check for upper case letters in FNAME
C       CH      Used to check for upper case letters in FNAME
C       DAT     Date
C       FNAME   Name of file
C       PDLNAM  Name of PDL file
C       POS     Used to check length of file name
C       TIM     Time of day
C  ---------------------------------------------------------------------
        CHARACTER*15 FNAME,PDLNAM,BLANK
```

```
      CHARACTER*40 TIX,TIY
      CHARACTER DAT(9),TIM(8)
      CHARACTER*1 CH
      INTEGER POS
      BYTE CB
      EQUIVALENCE (CB,CH)
C ---------------------------- Put DAT after filename ----------------
101   TYPE 107
107   FORMAT(1X,'Enter a filename for the data : ',$)
      READ(5,'(A)',ERR=101) FNAME
      BLANK = '                 '
      IPOS = INDEX(FNAME,'.')
      ILEN = INDEX(FNAME,' ') - 1
      IMAX=LEN(FNAME)
      IF (ILEN .EQ. 0) GOTO 101
      IF (IPOS .EQ. 0) THEN
       IF (ILEN .GE. (IMAX-4)) THEN
        FNAME((IMAX-3):IMAX) = '.DAT'
       ELSE
        FNAME(ILEN+1:ILEN+4) = '.DAT'
        FNAME(ILEN+5:IMAX) = BLANK(ILEN+5:IMAX)
       ENDIF
      ELSE
       IF (IPOS .GE. (IMAX-3)) THEN
        FNAME((IMAX-3):IMAX) = '.DAT'
       ELSE
        FNAME(IPOS:IPOS+3) = '.DAT'
        FNAME(IPOS+4:IMAX) = BLANK(IPOS+4:IMAX)
       ENDIF
      ENDIF
C ------------- Make sure all characters in FNAME are UPPER CASE ------
      ILEN=INDEX(FNAME,'.')-1
      DO 110  I=1,ILEN
      CH=FNAME(I:I)
      IF ((CB .GE. 97) .AND. (CB .LE. 122)) THEN
         CB=CB .AND. 95
      ENDIF
      FNAME(I:I)=CH
110   CONTINUE
      TYPE*
      TYPE*,'Filename= ',FNAME
C----------  Generate a PDL file with the same name as the data file --
      POS=INDEX(FNAME,'.')
      PDLNAM(1:POS)=FNAME(1:POS)
      PDLNAM(POS+1:POS+3)='PDL'
      OPEN(UNIT=2,NAME=PDLNAM,FORM='FORMATTED',TYPE='NEW')
      WRITE(2,220) FNAME
220   FORMAT(1X,';  Automatic PDL file for ',A15)
C -----        Setup axes and title information
-----------------------------
C     Get time and date
      CALL DATE(DAT)
      CALL TIME(TIM)
      WRITE(2,*) 'DTO'
```

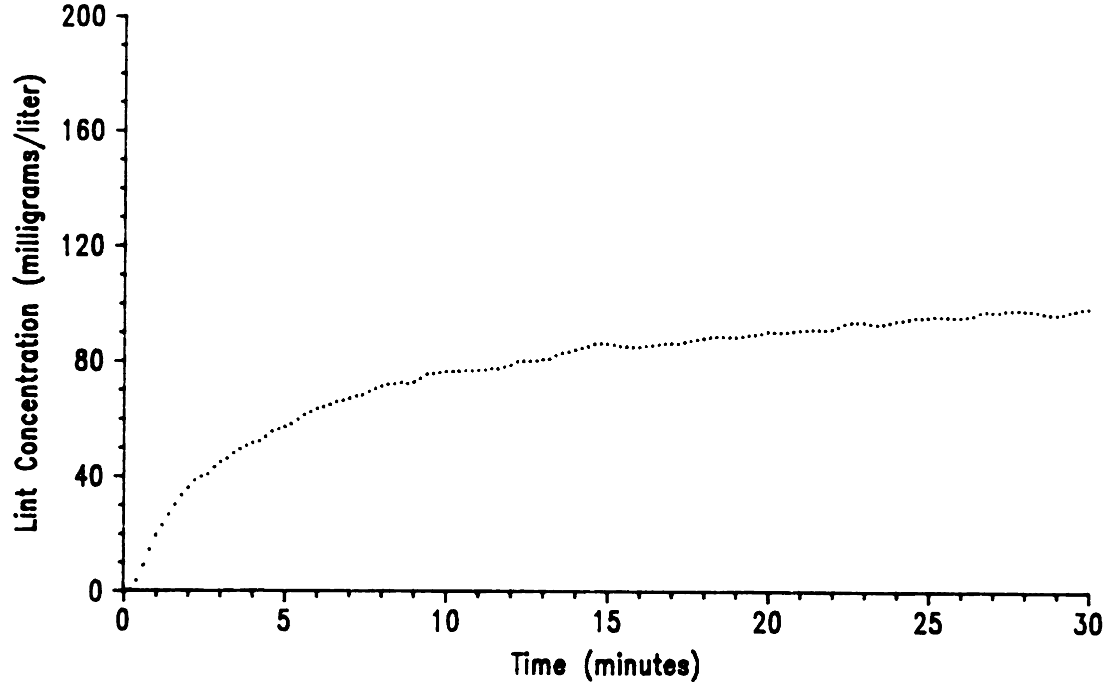
```
      WRITE(2,*) 'YA0.4,3.5,0.5'
      WRITE(2,*) 'XA0.5,5.0,0.5'
      WRITE(2,*) 'CPT10.12,0.125,0.05,0.05'
      WRITE(2,*) 'CPT2.12,.125,.05,.05'
      WRITE(2,*) 'CPTL.1,.1,.05,.01'
      WRITE(2,*) 'CPAT.1,.1,.05,.05'
      WRITE(2,*) 'FMXL(F7.0)'
      WRITE(2,*) 'FMYL(F8.1)'
      WRITE(2,*) 'FMXU(I3)'
      WRITE(2,*) 'NA2,2,2,2'
      WRITE(2,*) 'CNT12,0,0'
      WRITE(2,*) 'CNT22,0,0'
      WRITE(2,*) 'CNAT2,0,0'
      WRITE(2,*) 'CNTL2,0,0'
222   WRITE(2,112) TIX
112   FORMAT(1X,'TIXL',A40)
      WRITE(2,113) TIY
113   FORMAT(1X,'TIYL',A40)
      WRITE(2,225) FNAME,(DAT(I),I=1,9)
225   FORMAT(1X,'TIT1 ',A15,2X,9A1)
C
C  Open the data file and note the date and time
C
      OPEN(UNIT=1,NAME=FNAME,FORM='FORMATTED',TYPE='NEW')
      WRITE(1,115) FNAME,(DAT(I),I=1,9),(TIM(J),J=1,8)
115   FORMAT (' ;'/,
     +         ' ; Data File - ',A15,/,
     +         ' ; Test Date - ',9A1,/,
     +         ' ; Test Time - ',8A1)
      RETURN
      END

      SUBROUTINE CFILE(FILEN,XMAX,J)
C     SUBROUTINE CFILE(FILEN,XMAX,YMAX,J)
C
C-------------------------Create plotting files------------------------
C
      CHARACTER FILEN*15
      WRITE(2,130) FILEN                          ! set F phase in MULPLT
130   FORMAT(1X,'FN',A15)
      WRITE(2,134) J                              ! J is data tag
134   FORMAT(1X,'TGC',I1)
      WRITE(2,*) 'FT1,150'
      WRITE(2,135)
135   FORMAT(1X,'SS')
      J = J+1
      WRITE(2,140) FILEN                          ! set F phase in MULPLT
140   FORMAT(1X,'FN',A15)
      WRITE(2,144) J                              ! J is data tag
144   FORMAT(1X,'TGC',I1)
      WRITE(2,*) 'FT2,150'
      WRITE(2,145)
145   FORMAT(1X,'SS')
C ----------------- Set pdl information -----------------------------
```

```
        WRITE(2,153)
153     FORMAT(1X,'GP')                      ! Set G phase in MULPLT
        WRITE(2,155)                         ! Set Y axis limits
155     FORMAT(1X,'MMY10.0,150.')
C155    FORMAT(1X,'MMY10.0,',F7.2)
C       YTIC =
        WRITE(2,159)                         ! Set tick marks on YL
159     FORMAT(1X,'TMYL30.0,5.0,1,0.0')
        WRITE(2,160)XMAX                     ! Set X axis limits
160     FORMAT(1X,'MMX10.0,',F7.3)
        WRITE(2,165)                         ! Set tick marks on XL
165     FORMAT(1X,'TMXL5.0,1.0,1,0.0')
        WRITE(2,*) 'G01'                     ! Include the command to
                                             !  plot the graph on the
                                             !  screen
        CLOSE(UNIT=2)                        ! Close data files
        CLOSE(UNIT=1)
        RETURN
        END
```

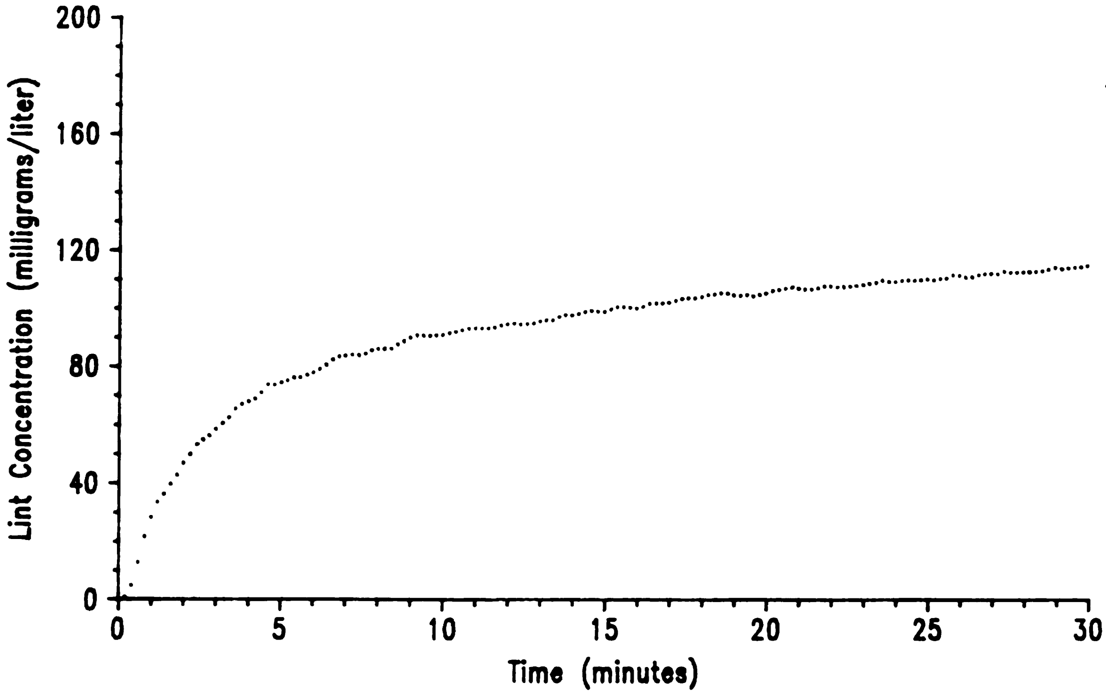
APPENDIX B

EXPERIMENTAL DATA

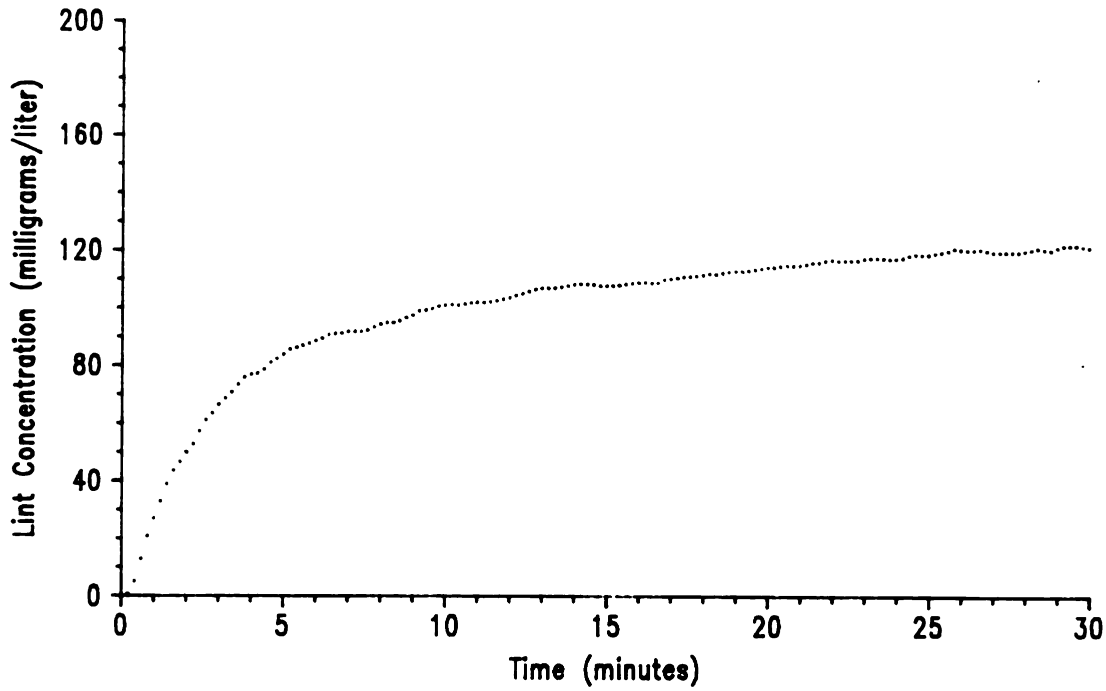
RUN - 1 Frequency - 2.00 Stroke Length - 100



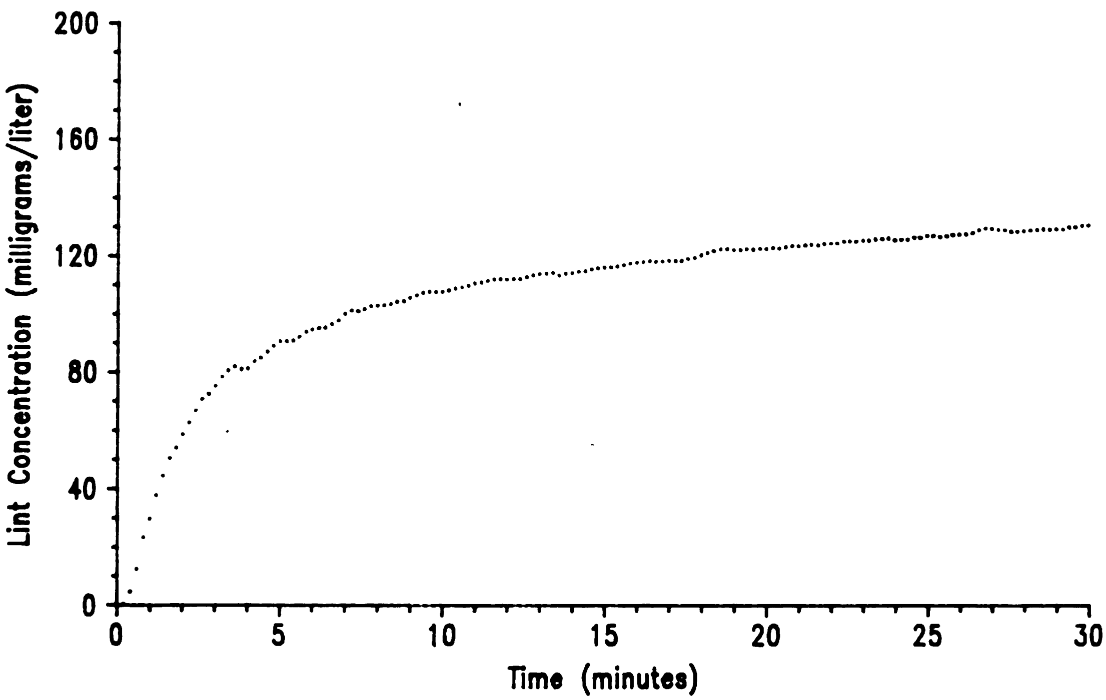
RUN - 2 Frequency - 2.25 Stroke Length - 100



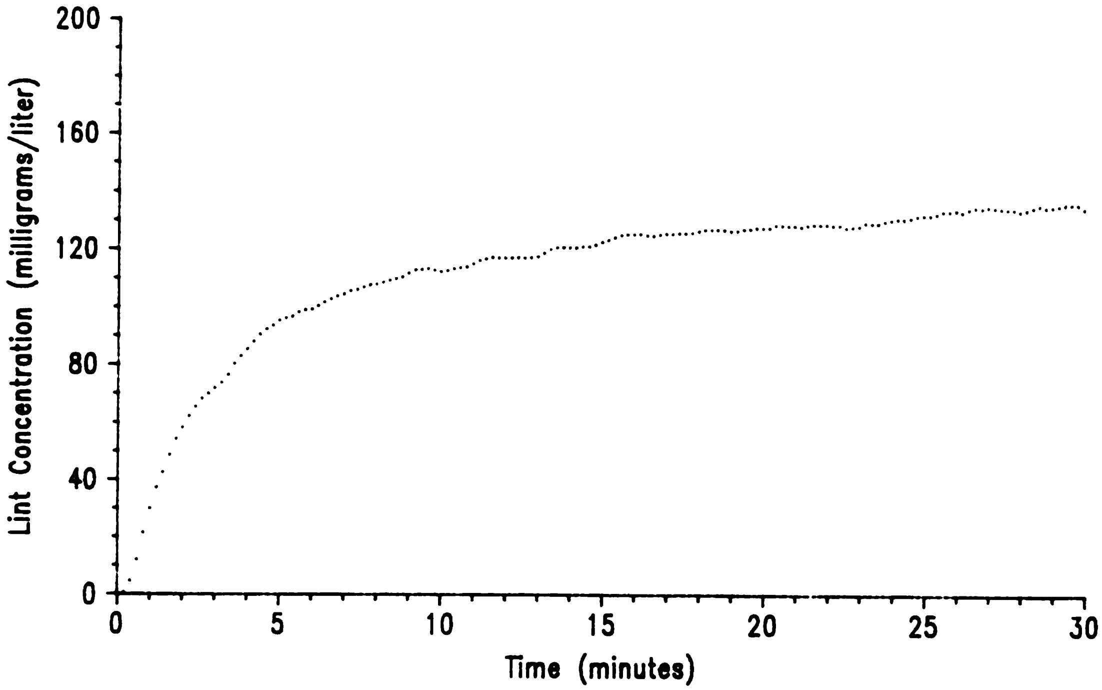
RUN - 3 Frequency - 2.50 Stroke Length - 100



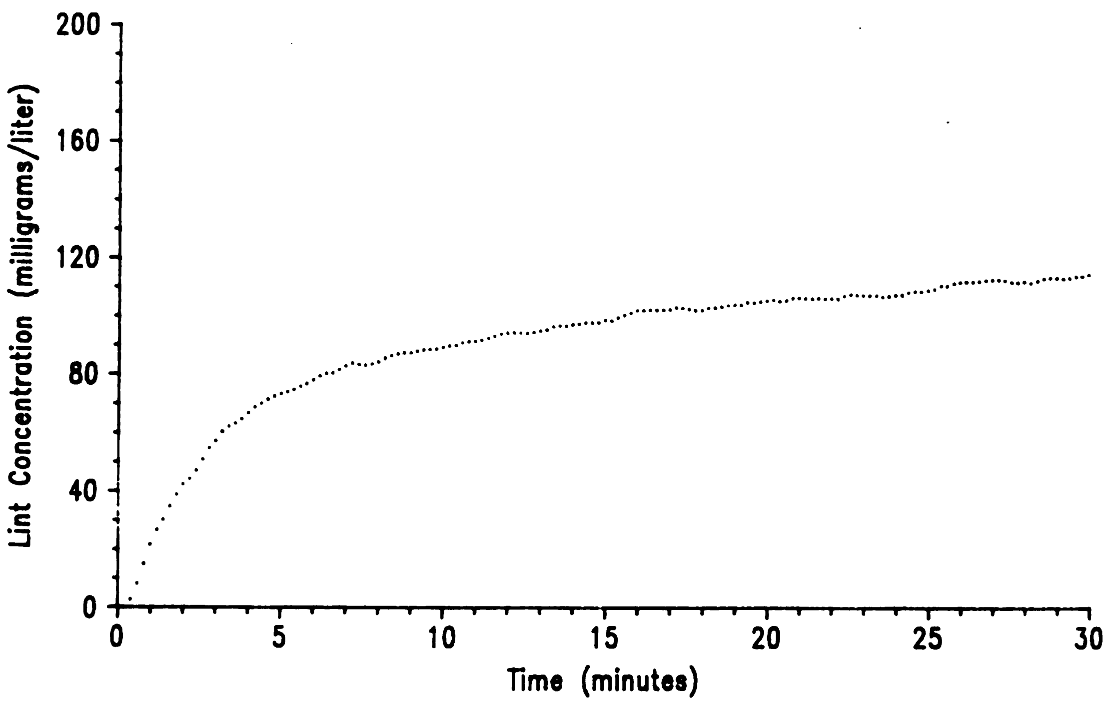
RUN - 4 Frequency - 2.75 Stroke Length - 100



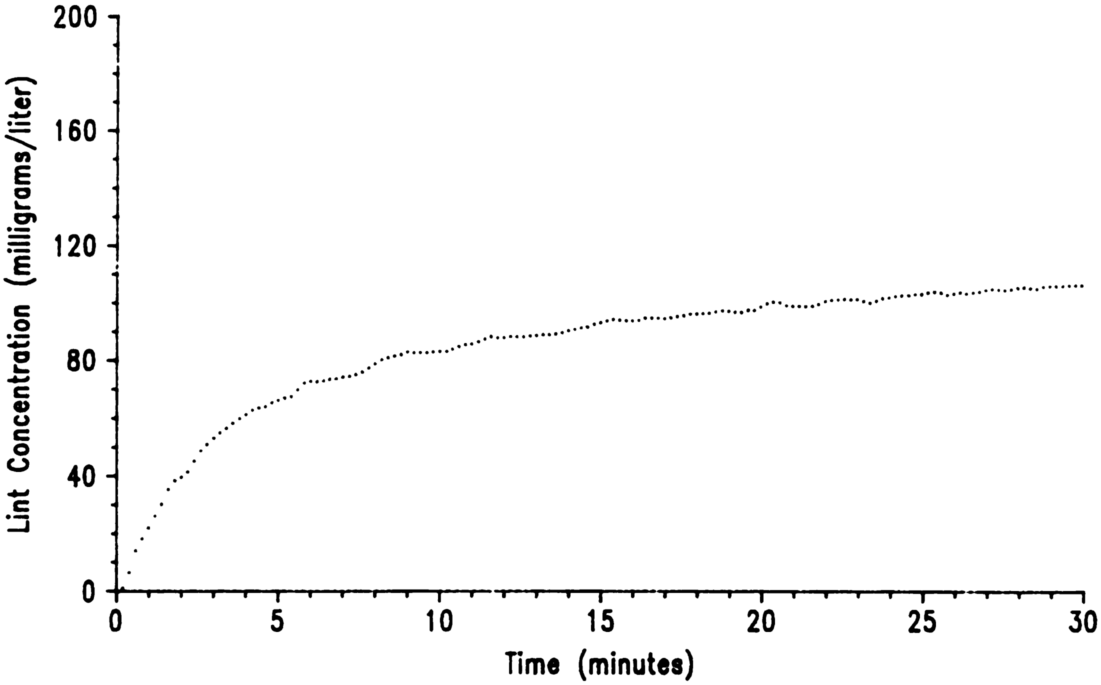
RUN - 5 Frequency - 2.75 Stroke Length - 100



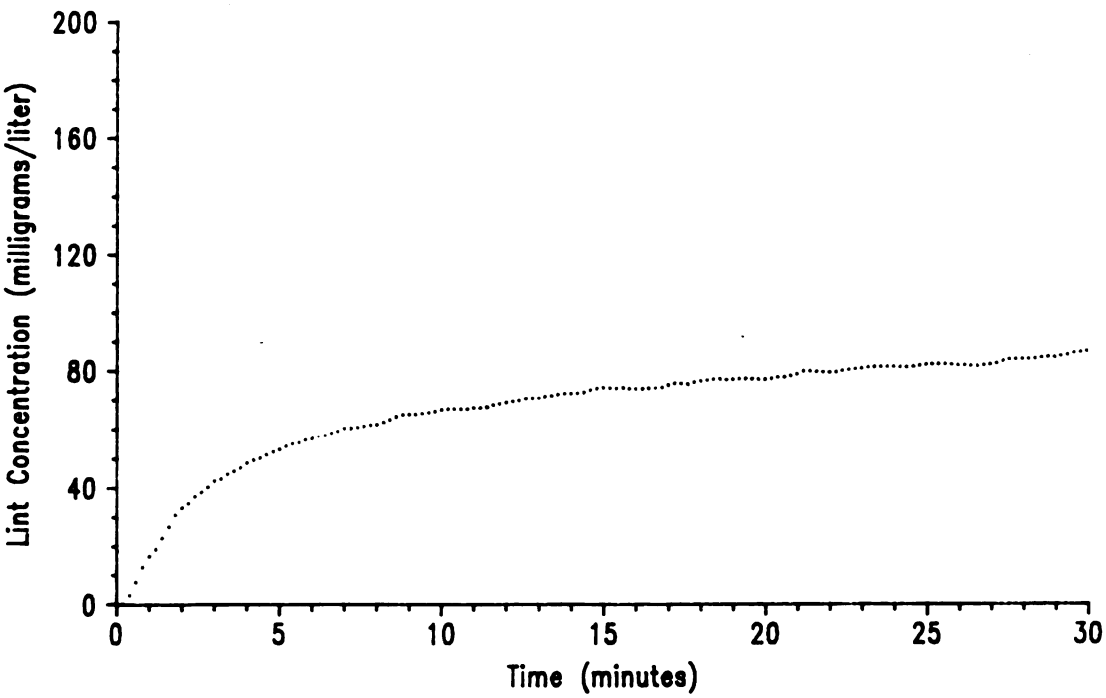
RUN - 6 Frequency - 2.50 Stroke Length - 100



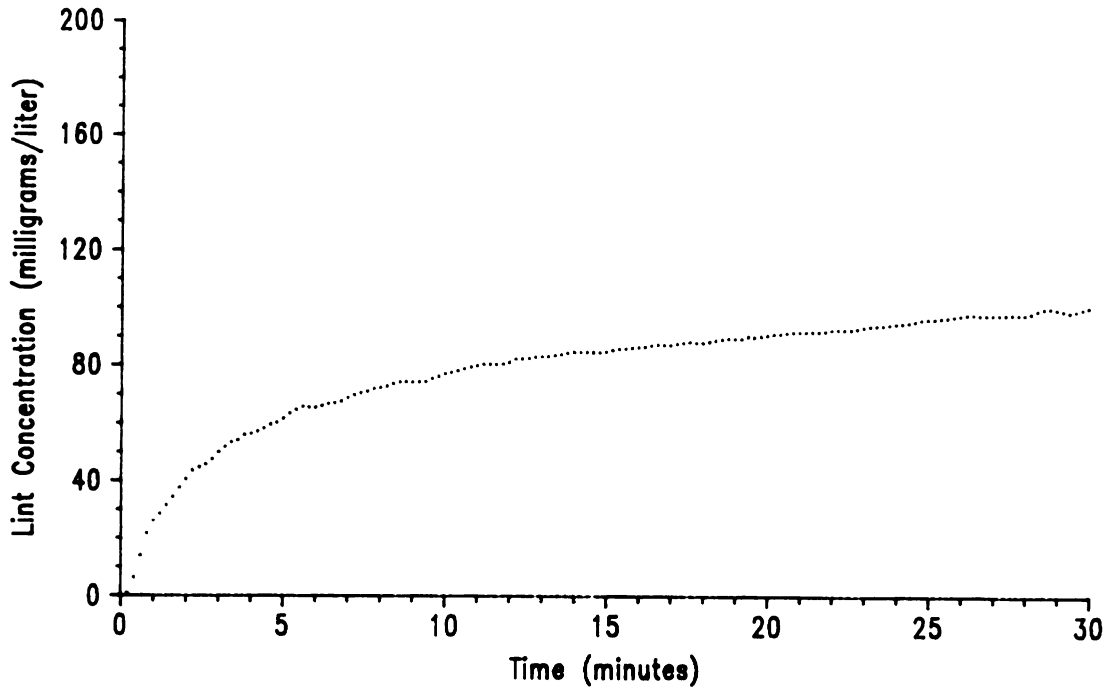
RUN - 7 Frequency - 2.25 Stroke Length - 100



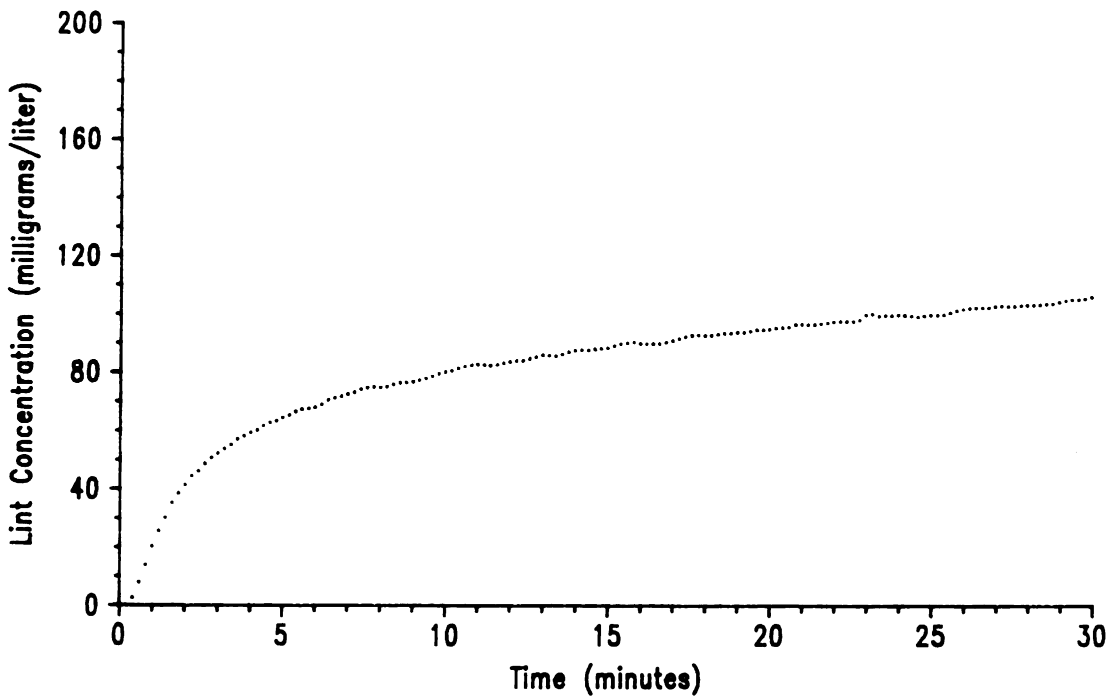
RUN - 8 Frequency - 2.00 Stroke Length - 100



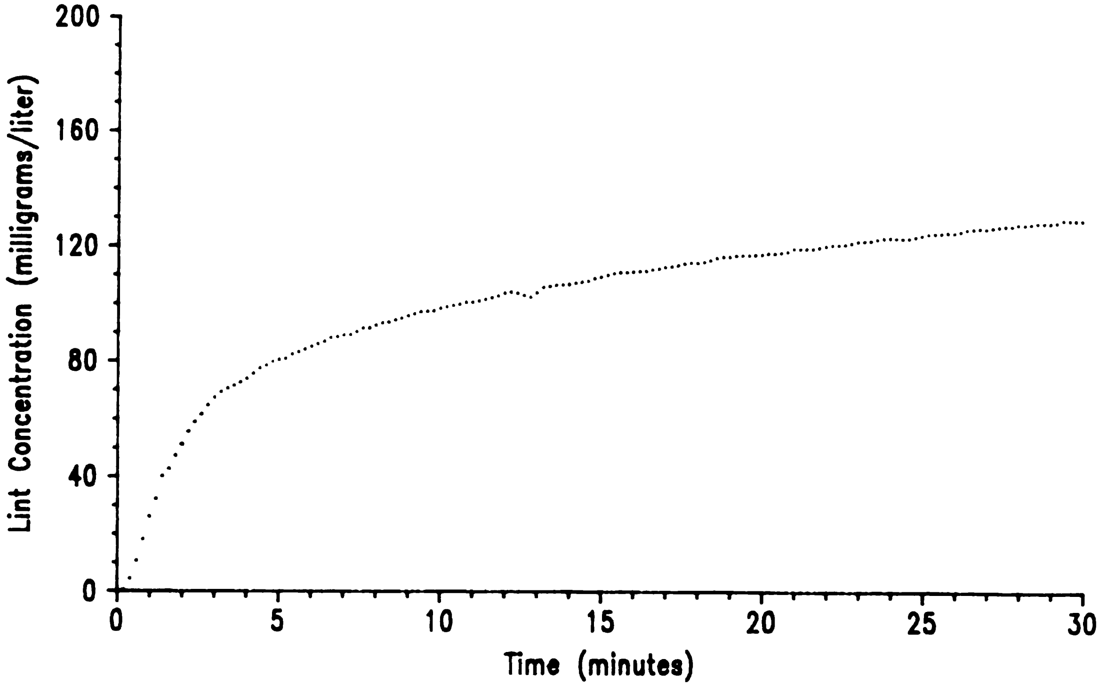
RUN - 9 Frequency - 1.50 Stroke Length - 140



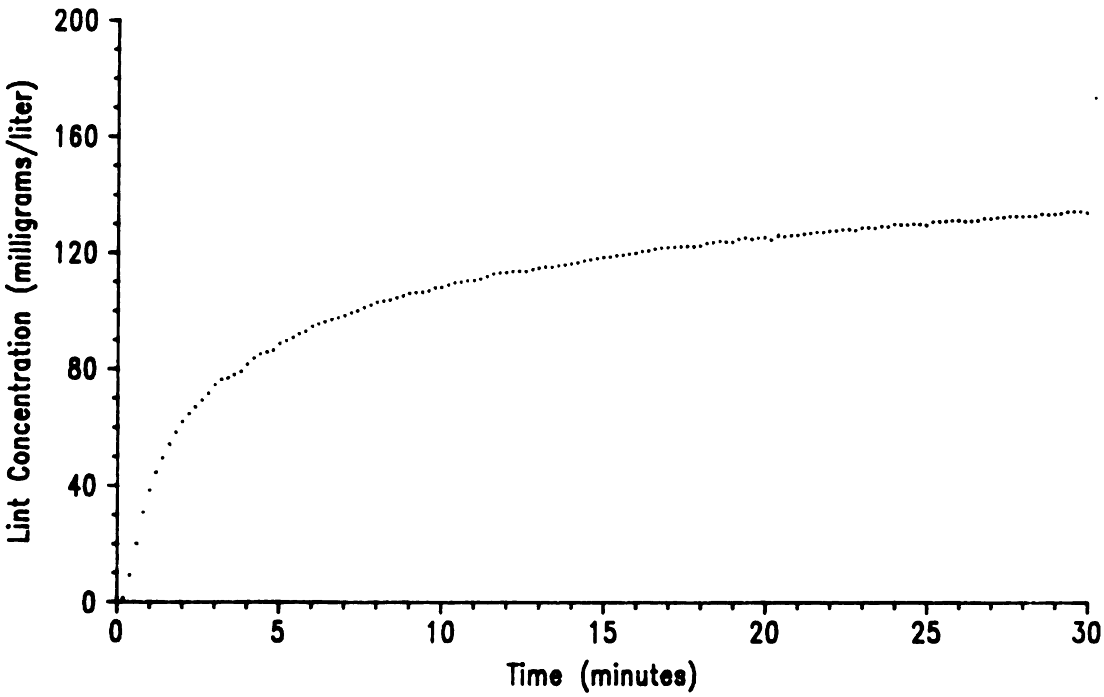
RUN - 10 Frequency - 1.75 Stroke Length - 140



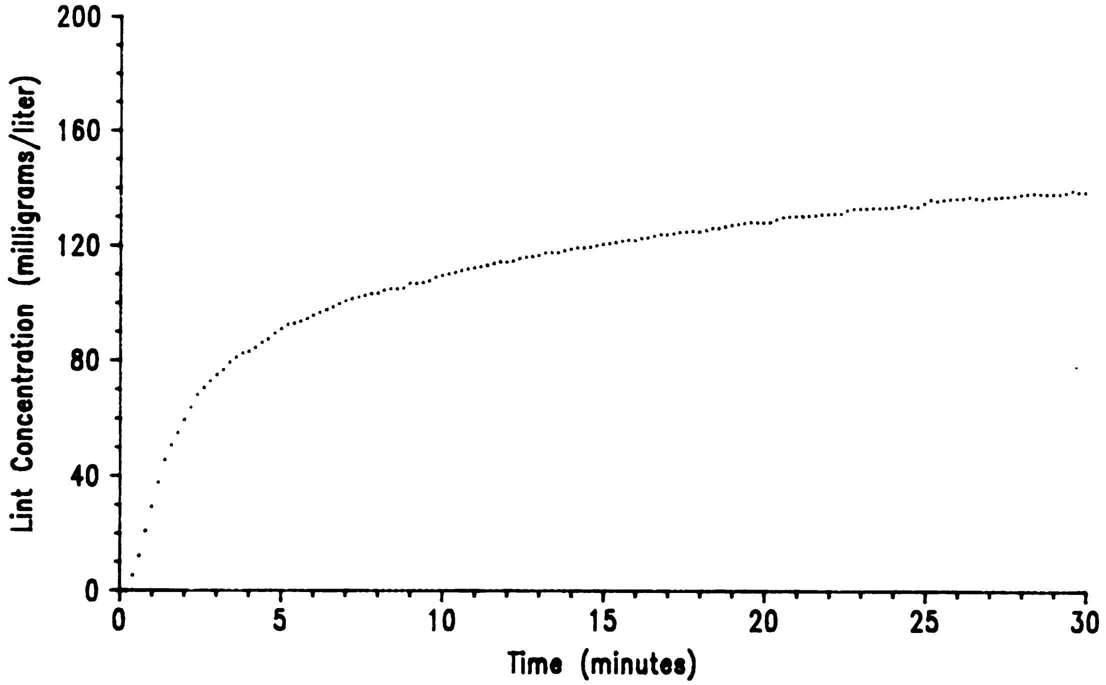
RUN - 11 Frequency - 2.00 Stroke Length - 140



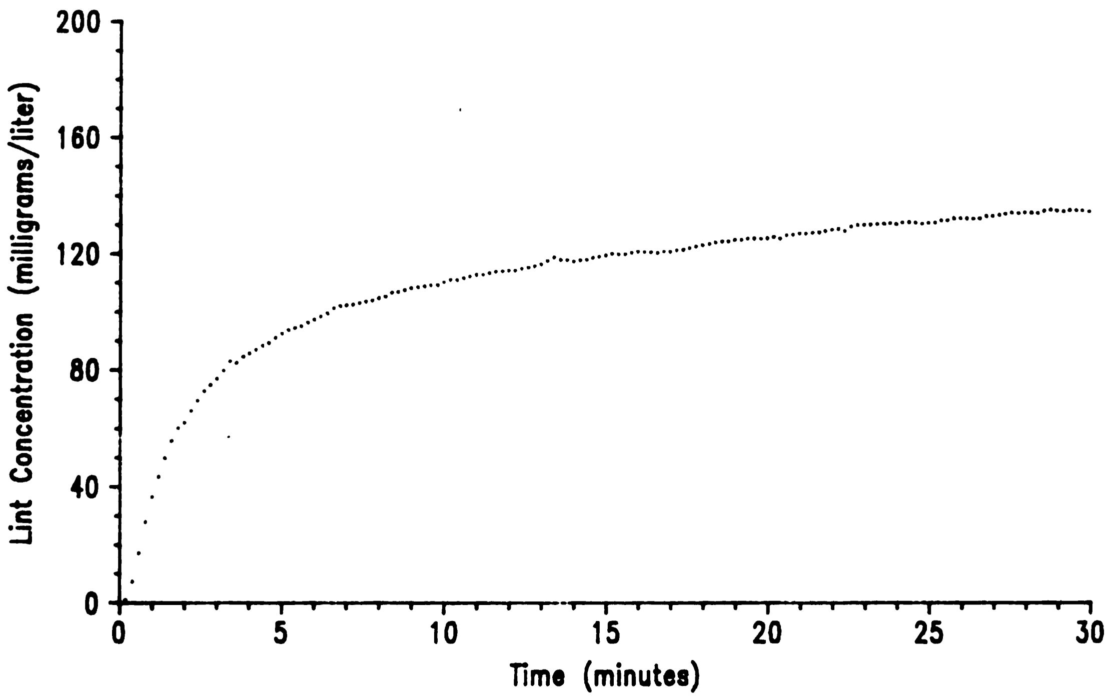
RUN - 12 Frequency - 2.25 Stroke Length - 140



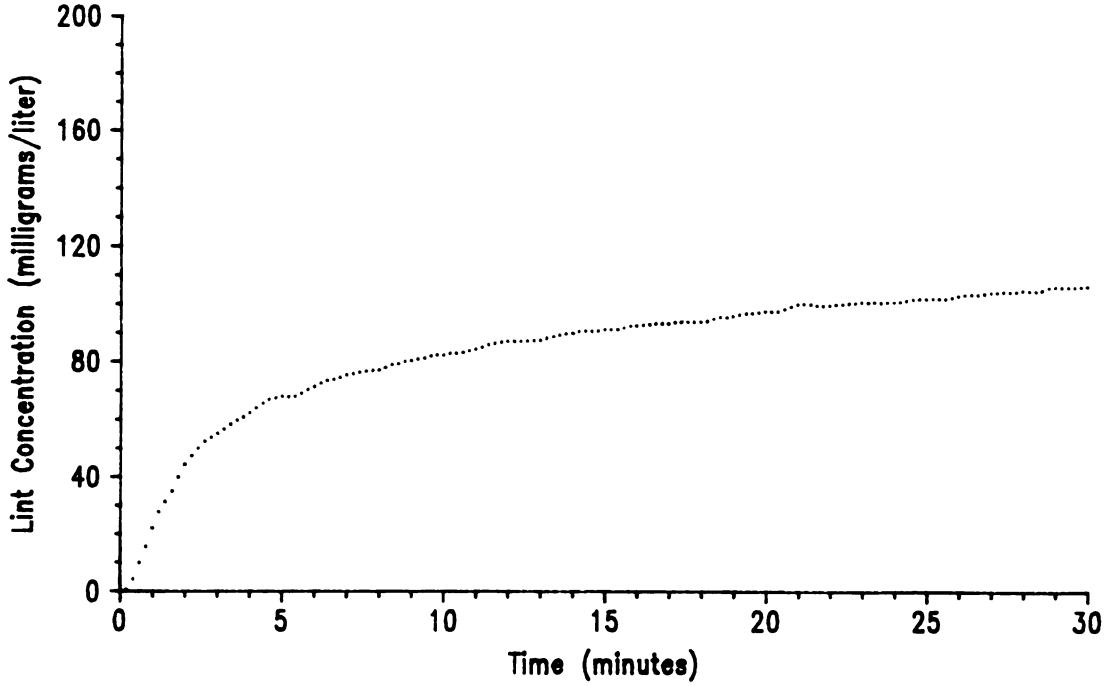
RUN - 13 Frequency - 2.25 Stroke Length - 140



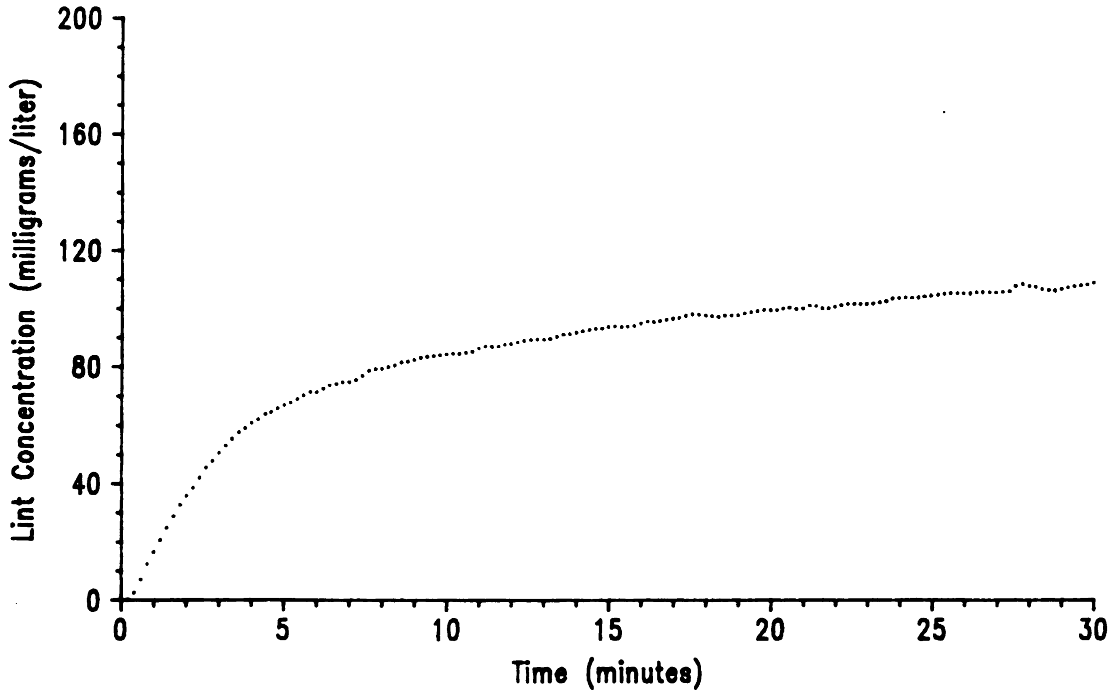
RUN - 14 Frequency - 2.00 Stroke Length - 140

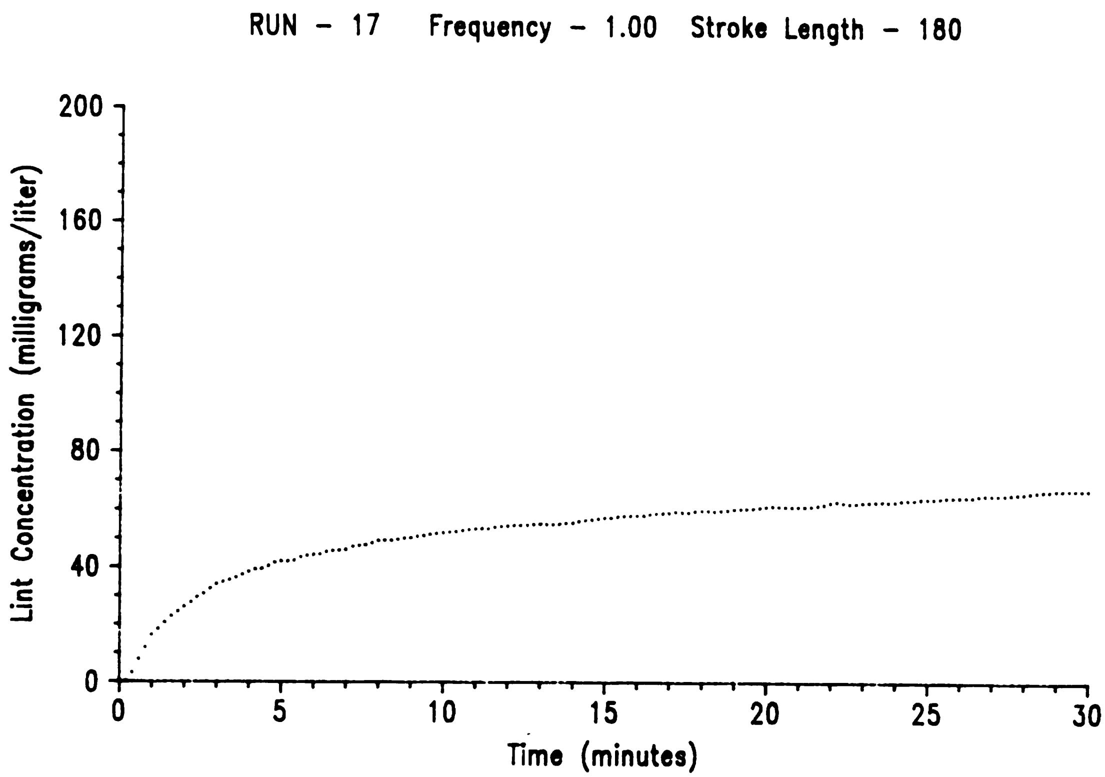


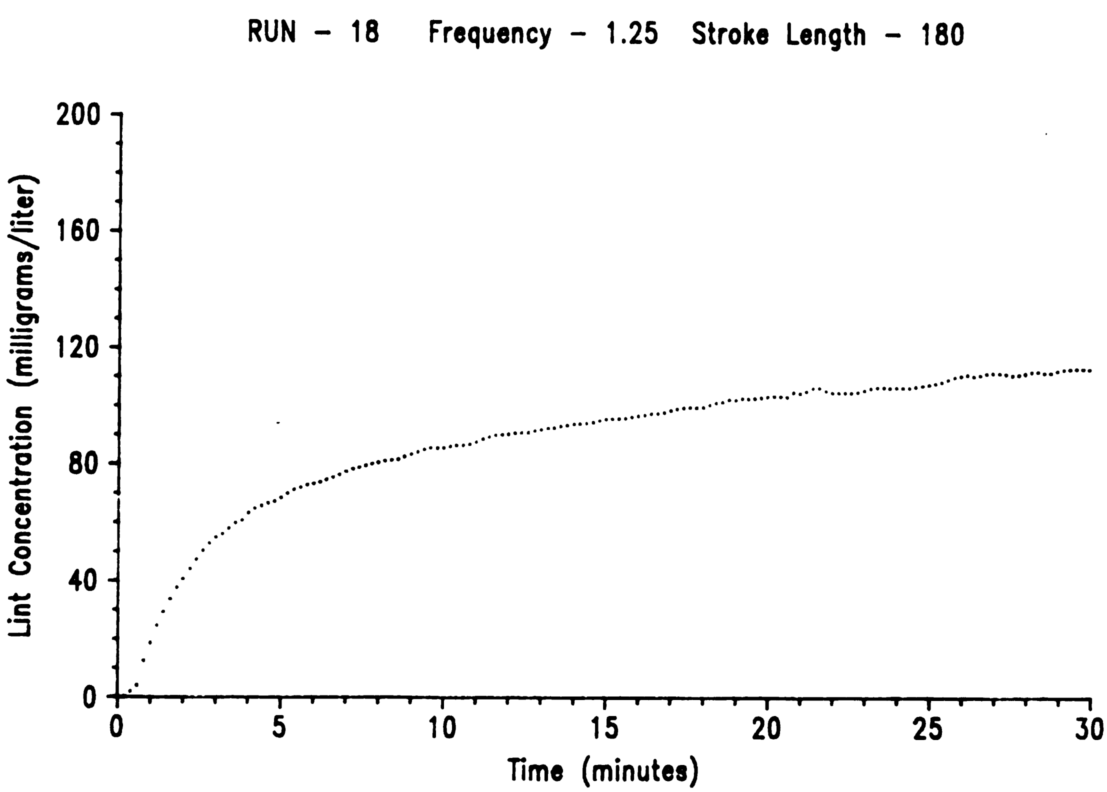
RUN - 15 Frequency - 1.75 Stroke Length - 140



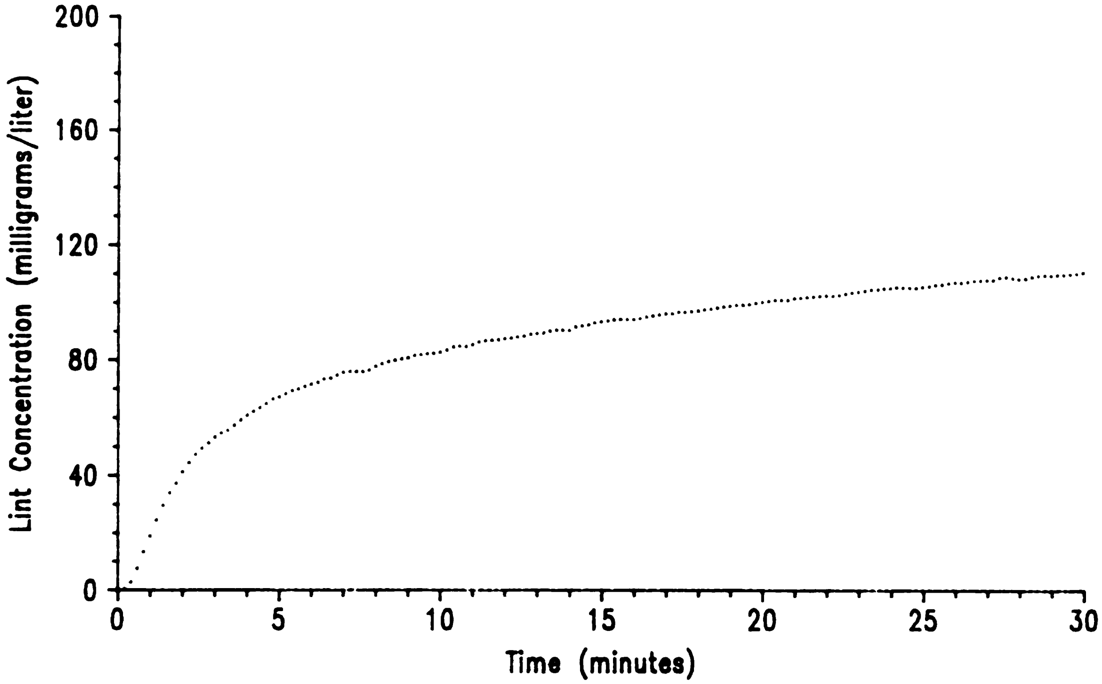
RUN - 16 Frequency - 1.50 Stroke Length - 140



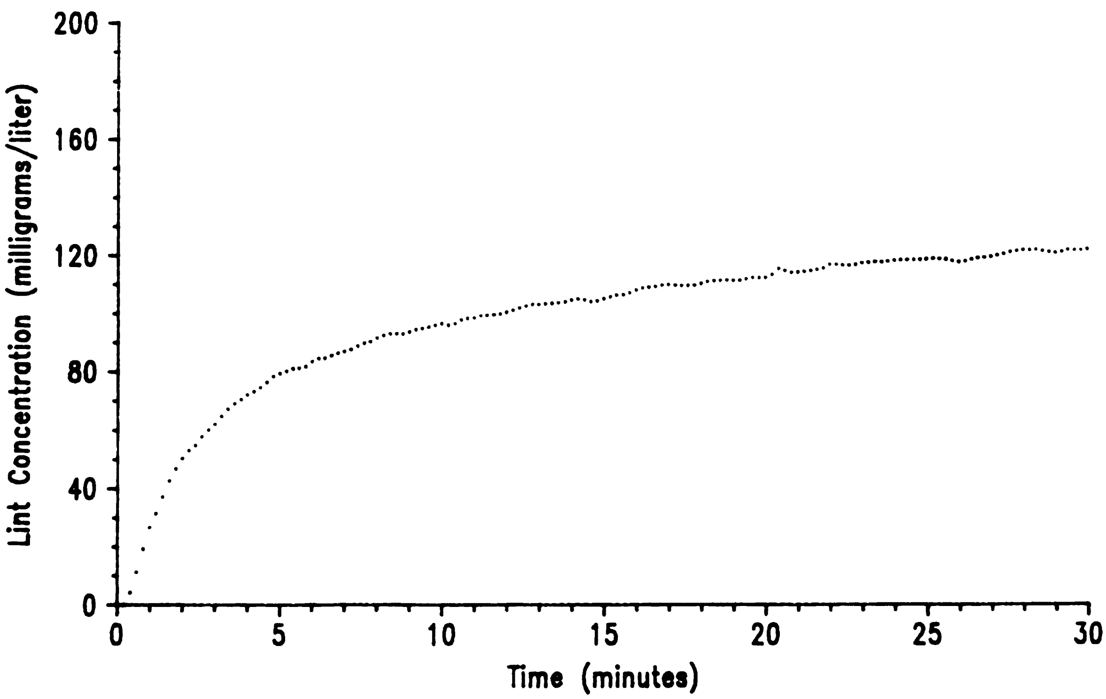

RUN - 17 Frequency - 1.00 Stroke Length - 180
Lint Concentration (milligrams/liter)
200
160
120
80
40
0
0
5
10
15
20
25
30
Time (minutes)


RUN - 18 Frequency - 1.25 Stroke Length - 180
Lint Concentration (milligrams/liter)
200
160
120
80
40
0
0
5
10
15
20
25
30
Time (minutes)

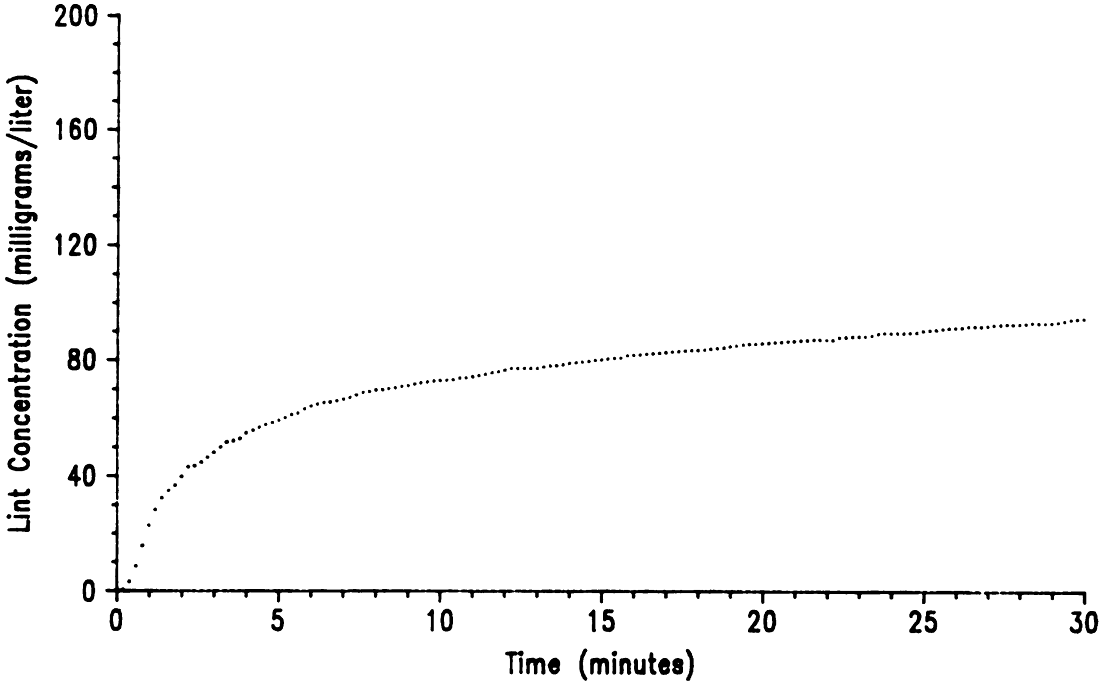
RUN - 19 Frequency - 1.50 Stroke Length - 180



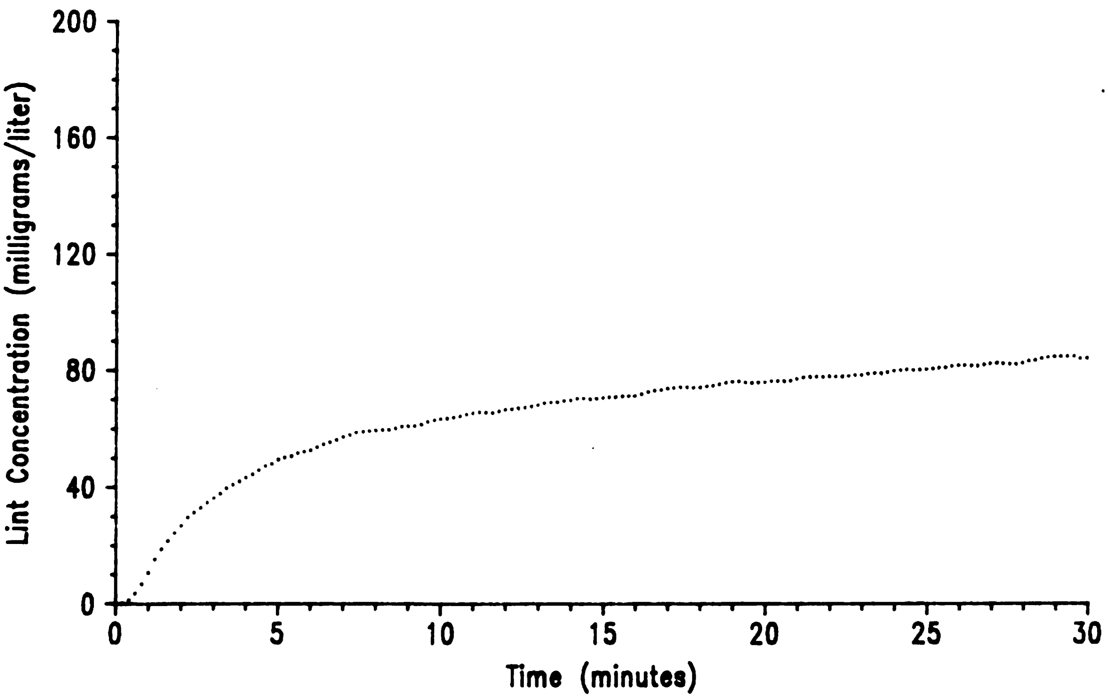
RUN - 20 Frequency - 1.50 Stroke Length - 180



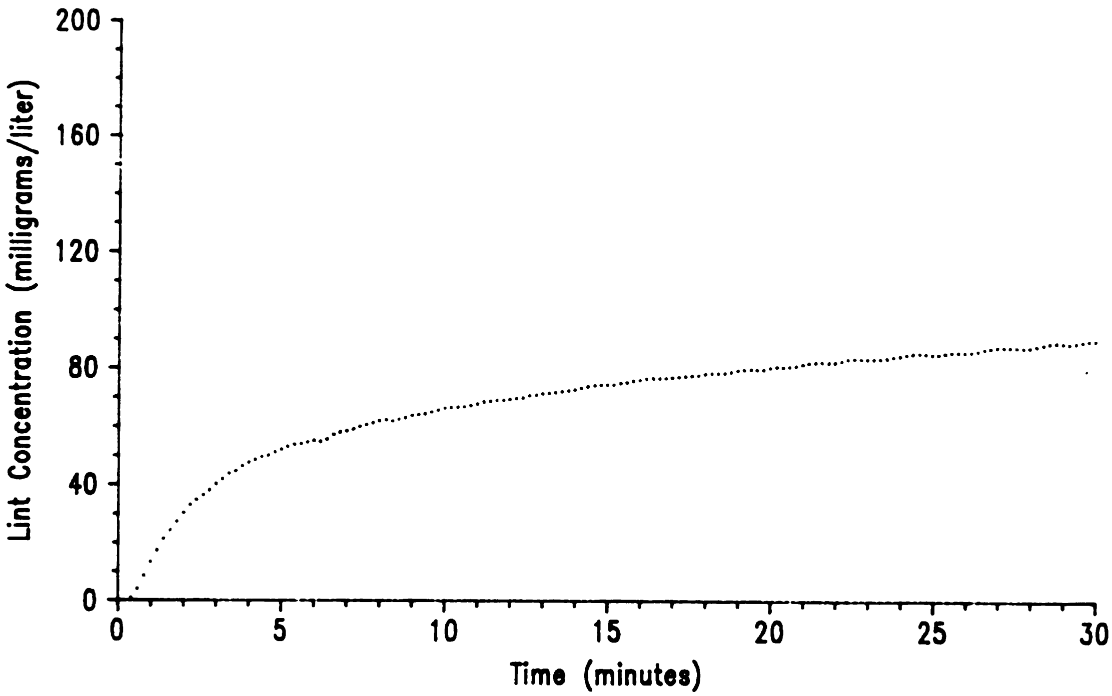
RUN - 21 Frequency - 1.25 Stroke Length - 180



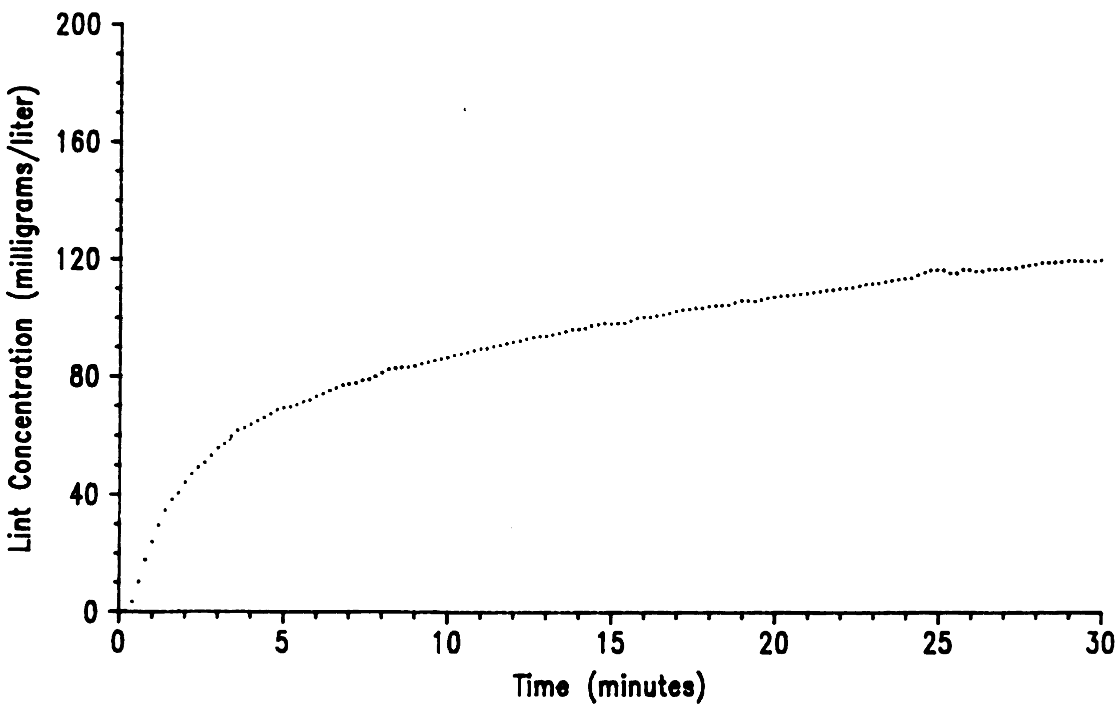
RUN - 22 Frequency - 1.00 Stroke Length - 180



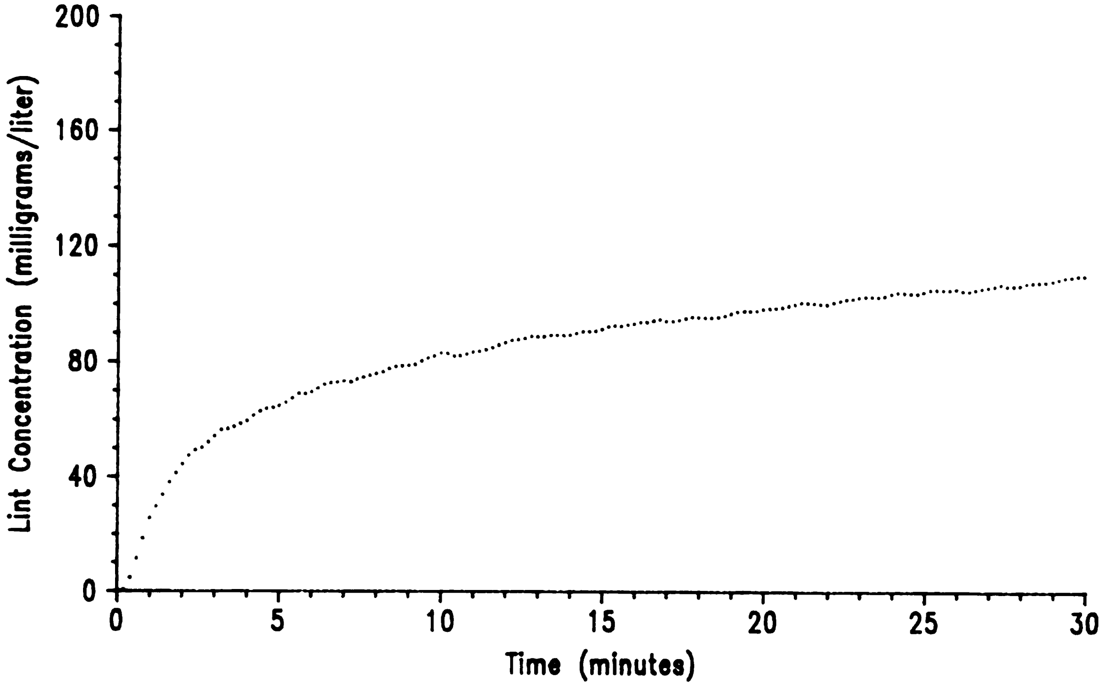
RUN - 23 Frequency - 1.75 Stroke Length - 120



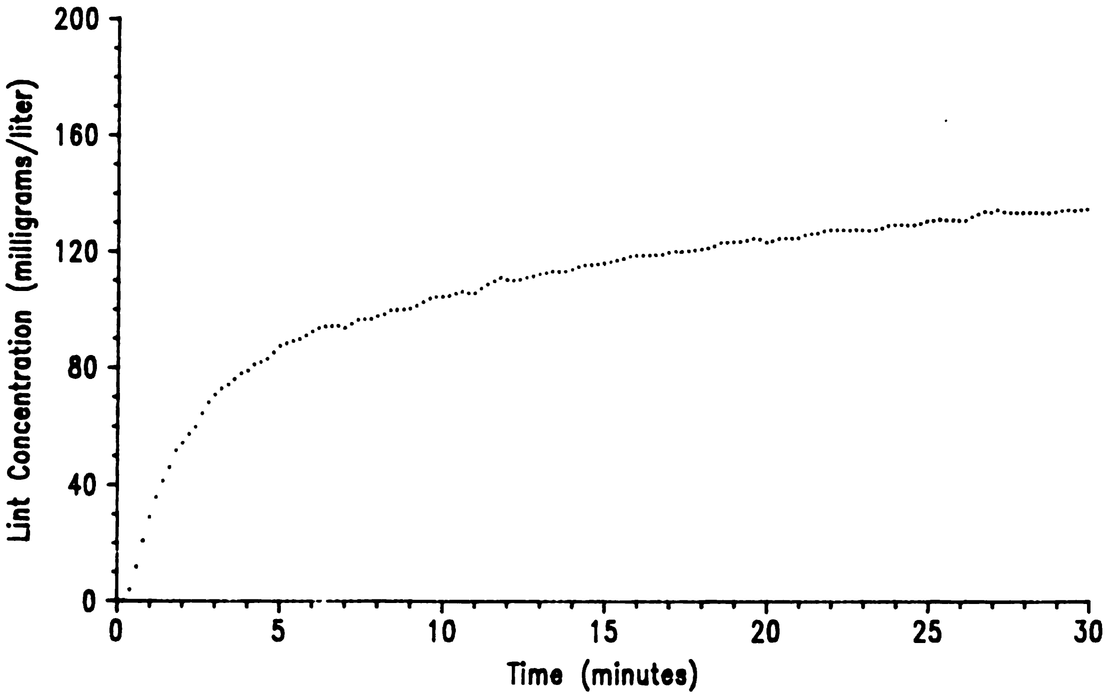
RUN - 24 Frequency - 2.25 Stroke Length - 120



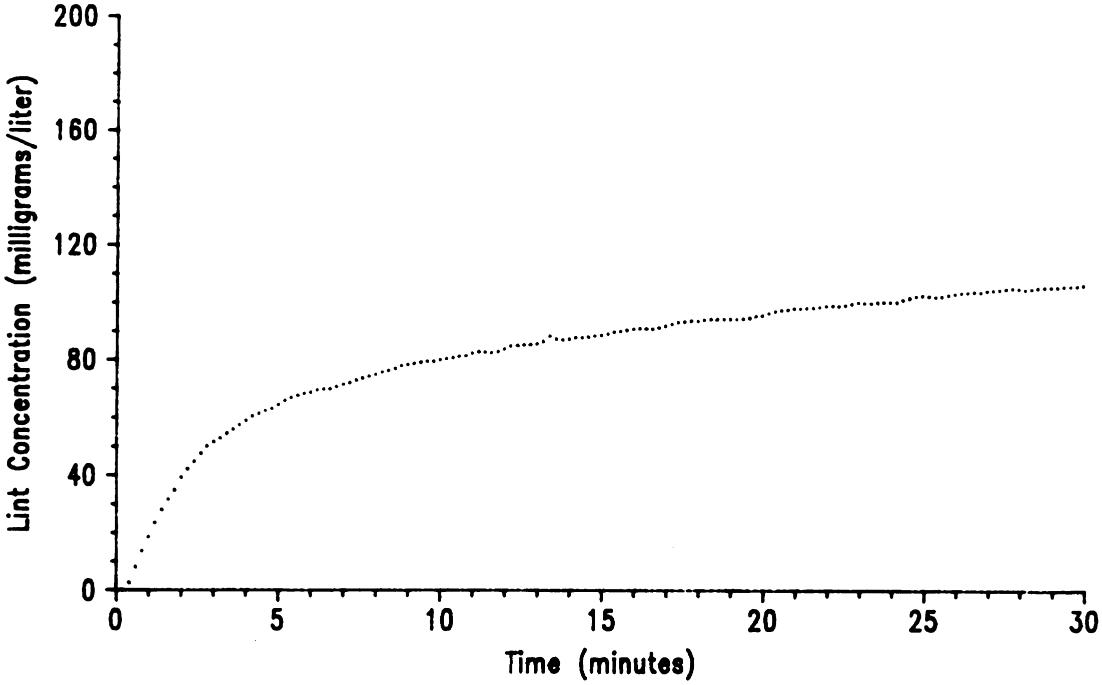
RUN - 25 Frequency - 2.00 Stroke Length - 120



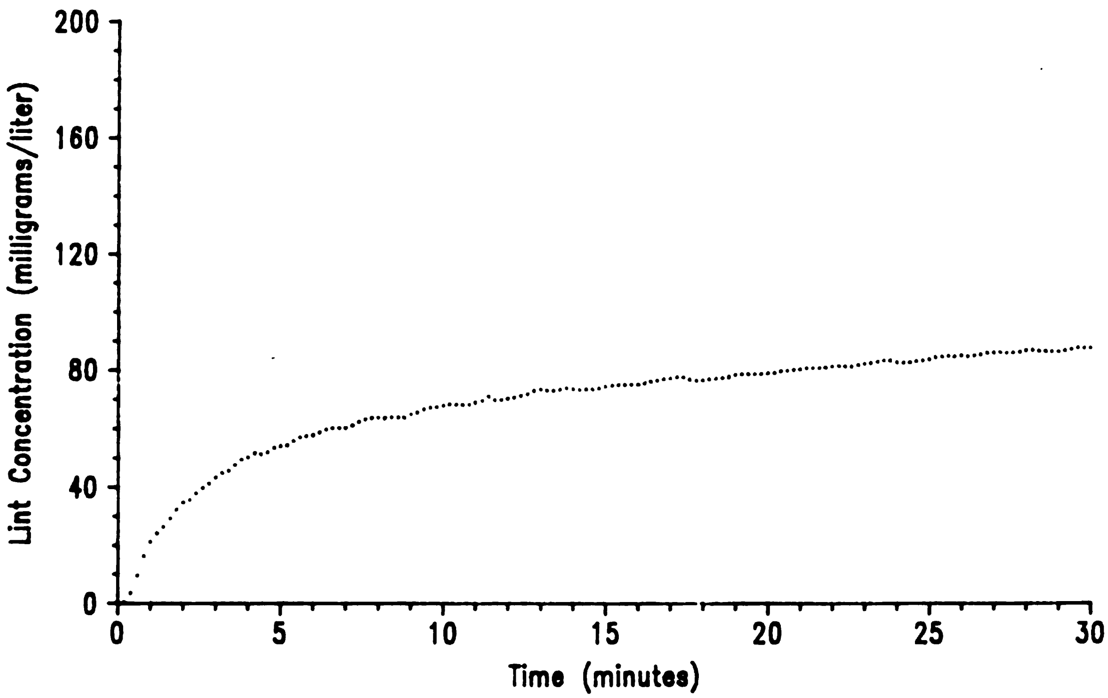
RUN - 26 Frequency - 2.25 Stroke Length - 120



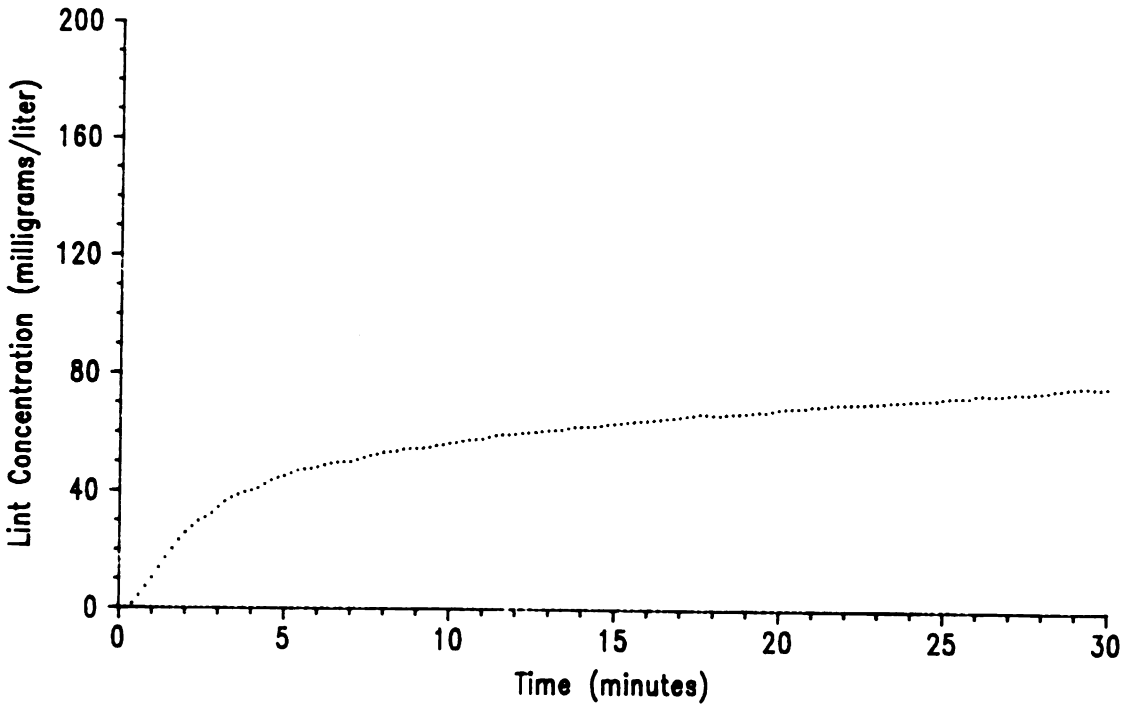
RUN - 27 Frequency - 2.00 Stroke Length - 120



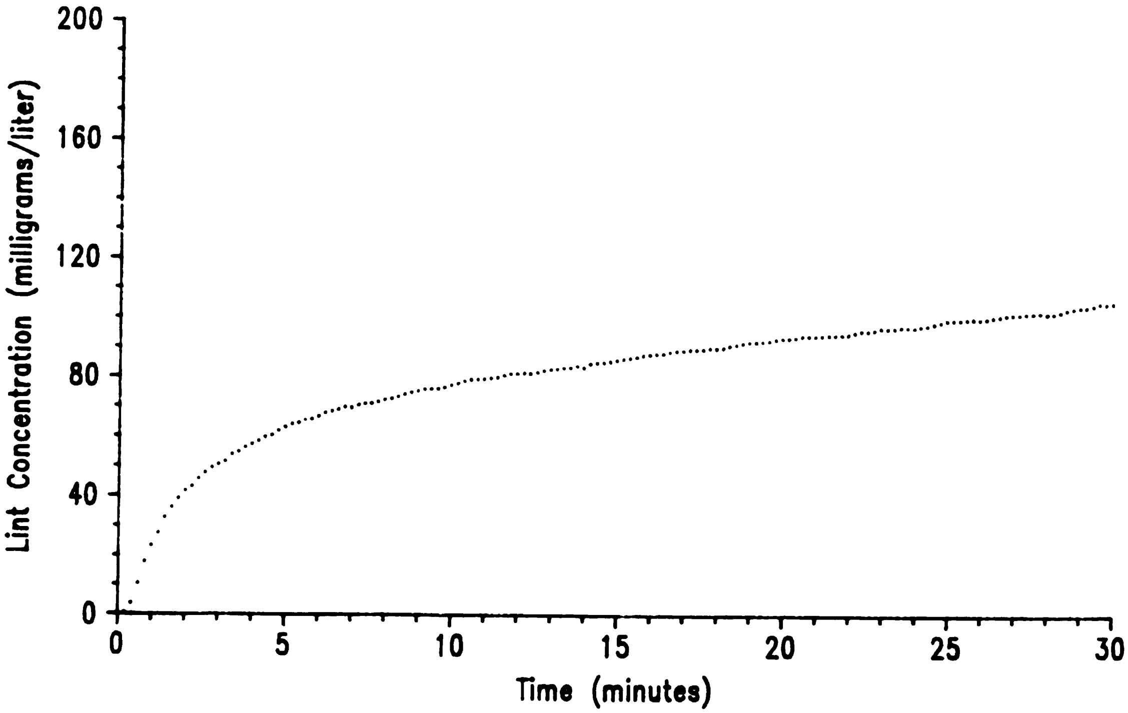
RUN - 28 Frequency - 1.75 Stroke Length - 120



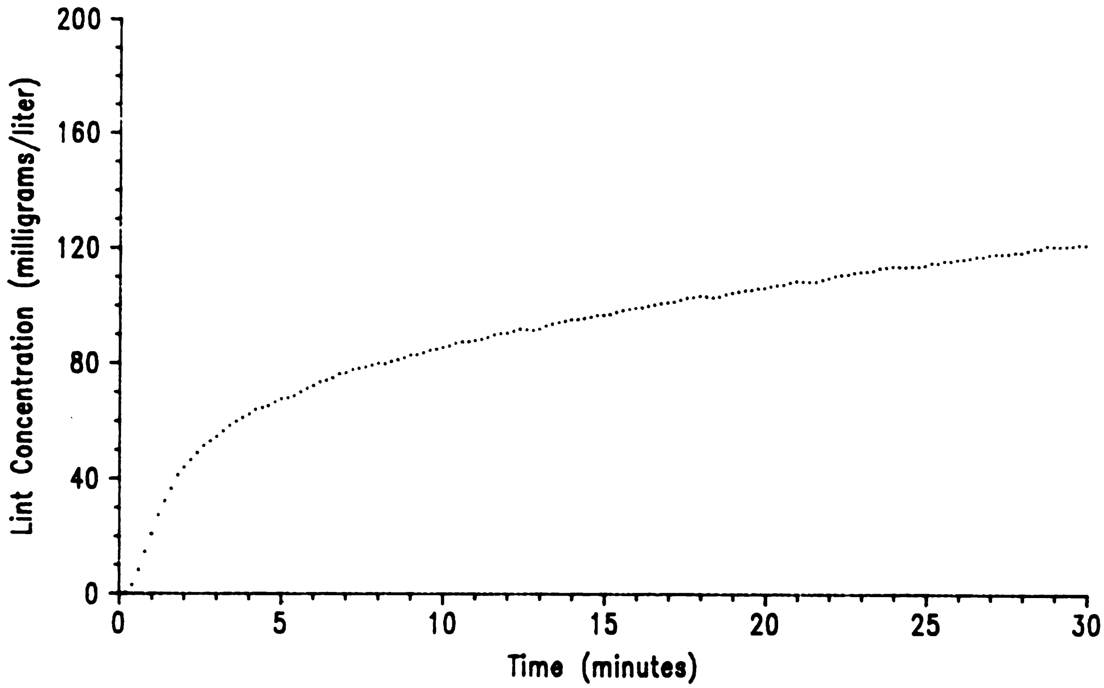
RUN - 29 Frequency - 1.25 Stroke Length - 160



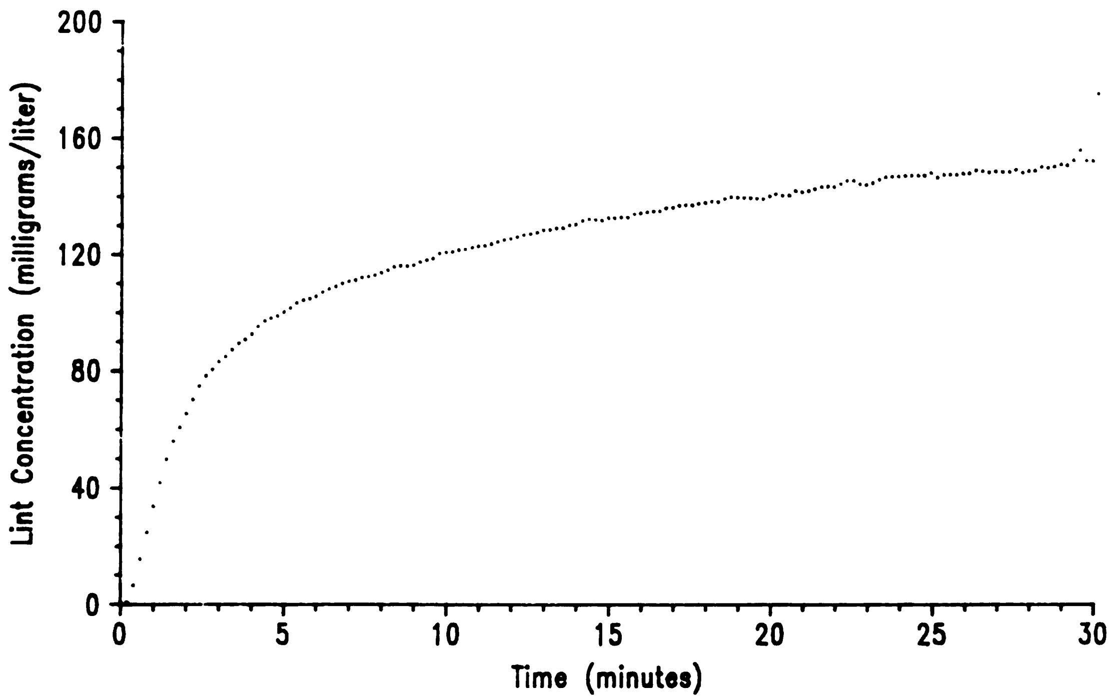
RUN - 30 Frequency - 1.50 Stroke Length - 160



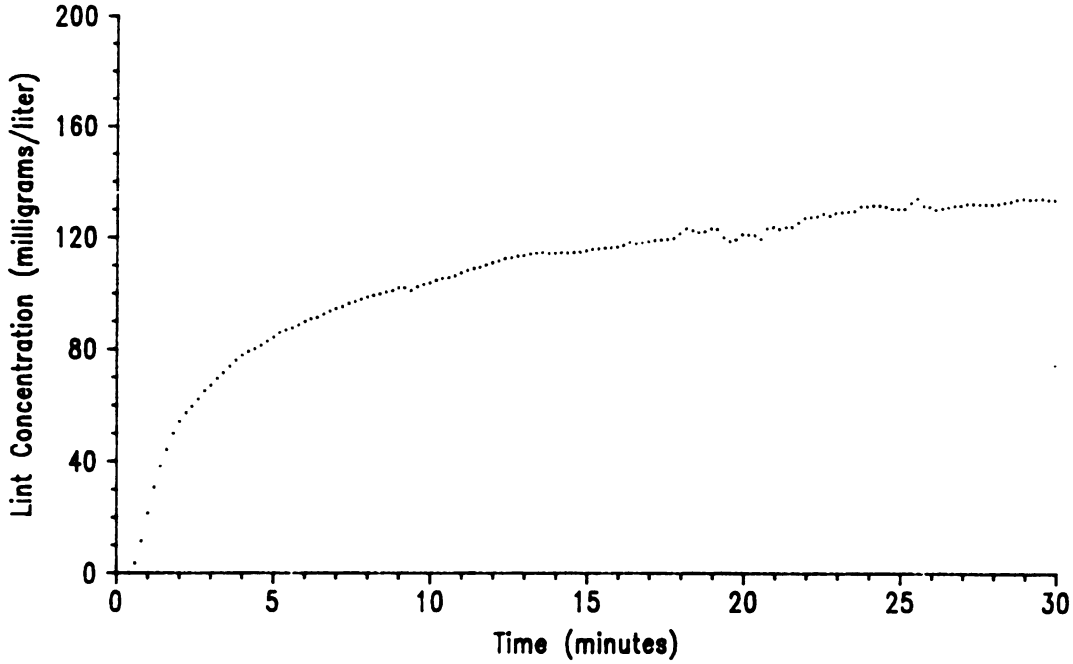
RUN - 31 Frequency - 1.75 Stroke Length - 160



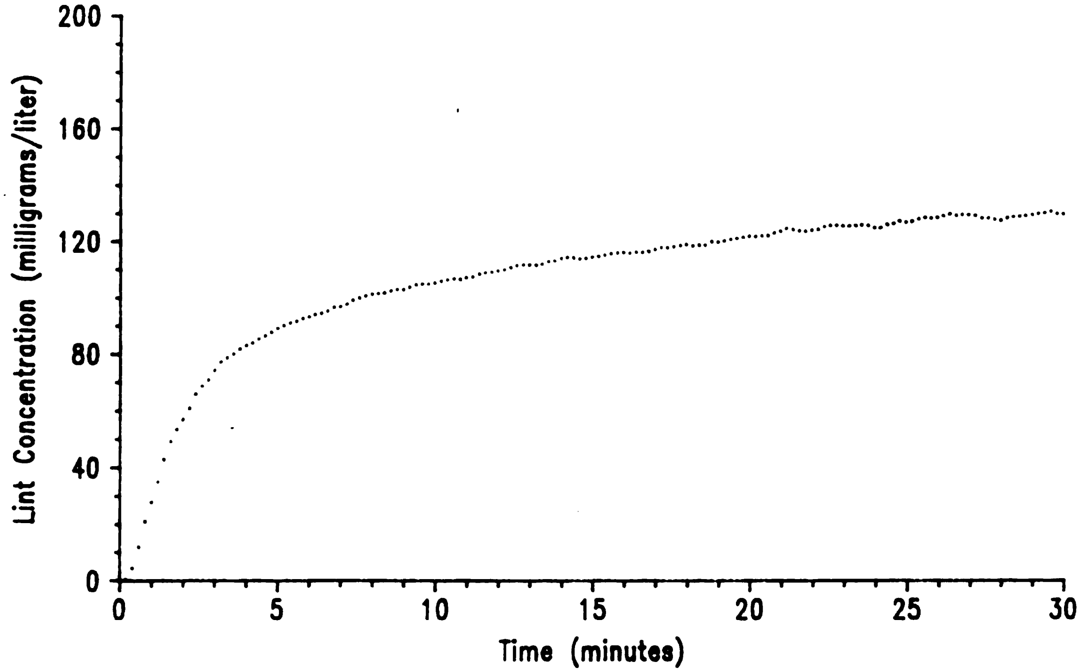
RUN - 32 Frequency - 2.00 Stroke Length - 160



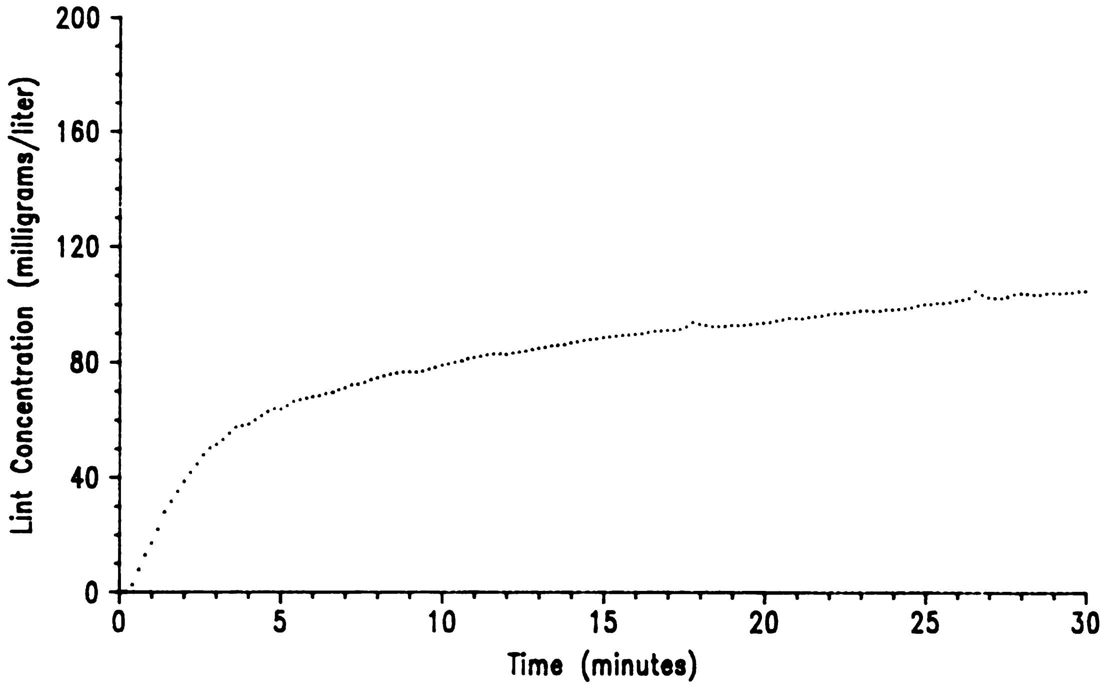
RUN - 33   Frequency - 2.00   Stroke Length - 160



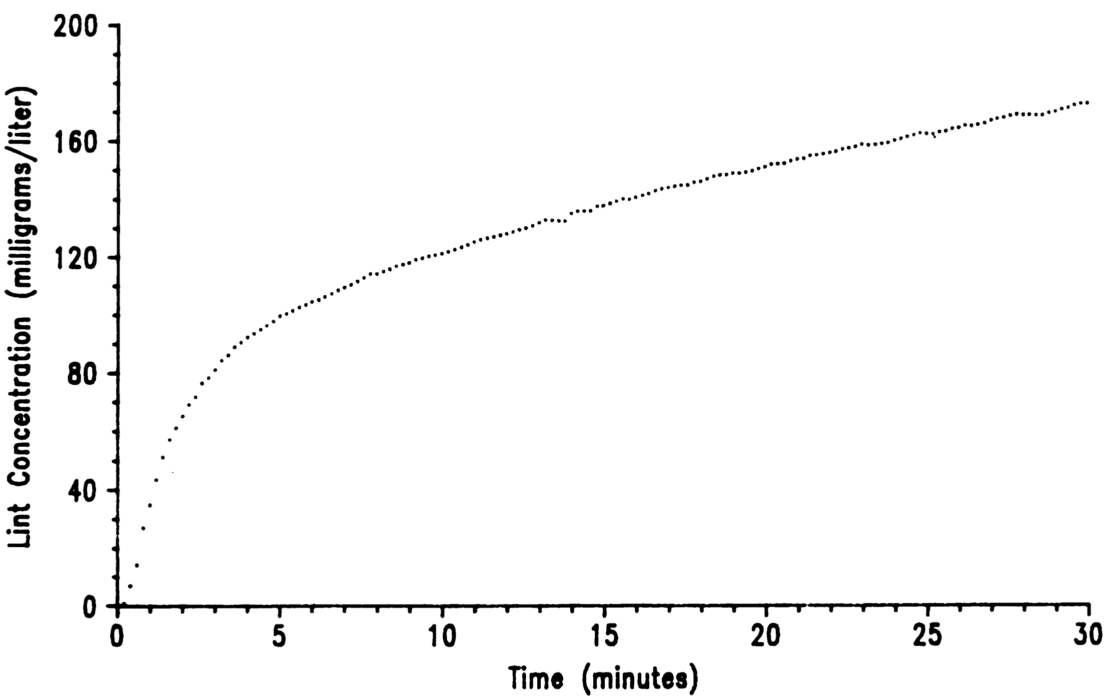
RUN - 34   Frequency - 1.75   Stroke Length - 160



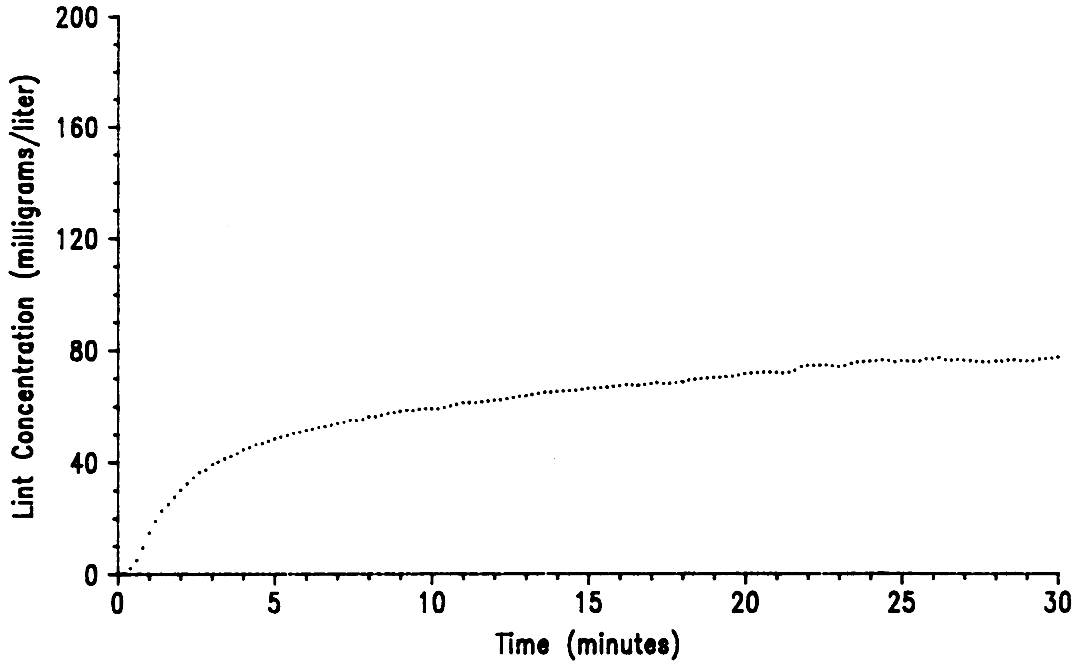
RUN - 35   Frequency - 1.50   Stroke Length - 160



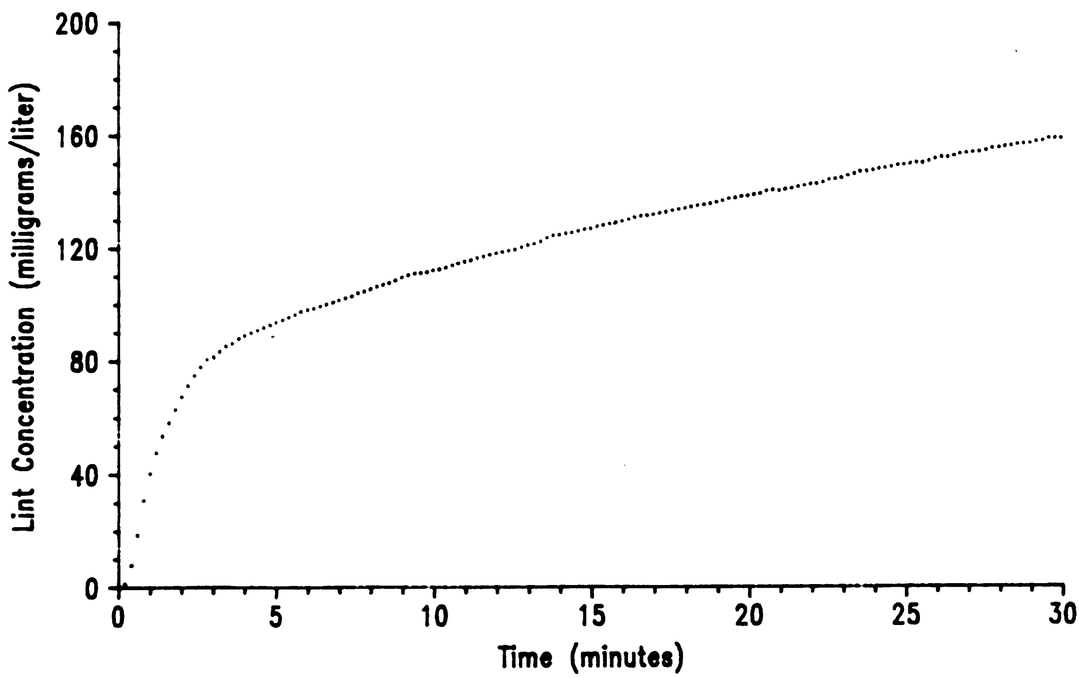
RUN - 36   Frequency - 2.25   Stroke Length - 160



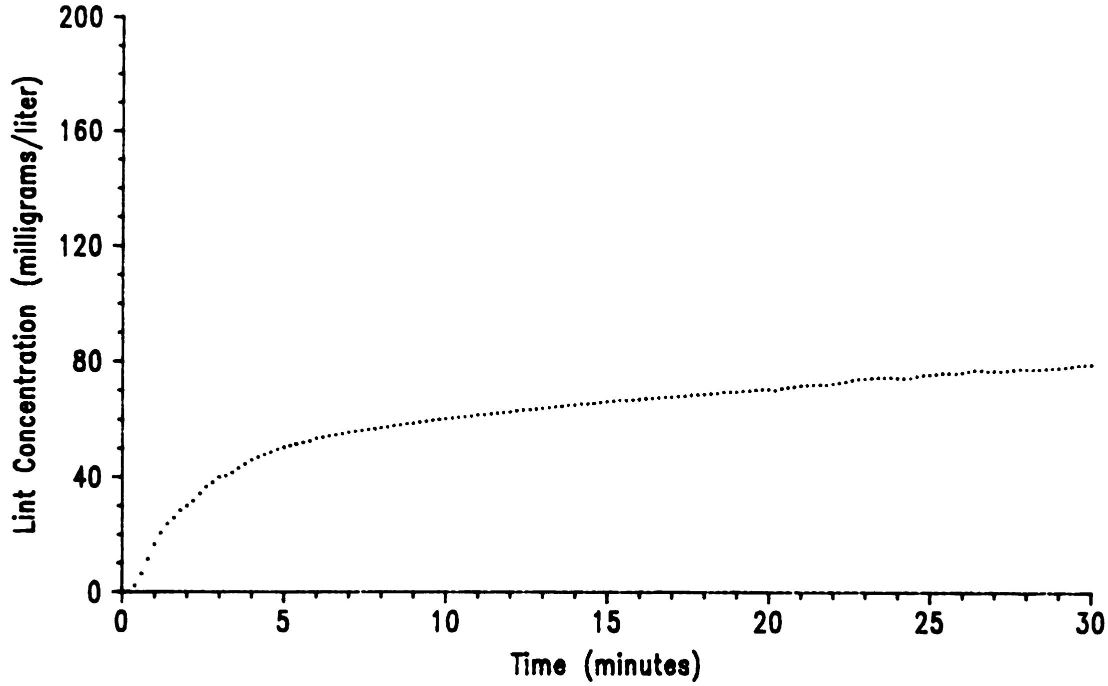
RUN - 37 Frequency - 1.25 Stroke Length - 160



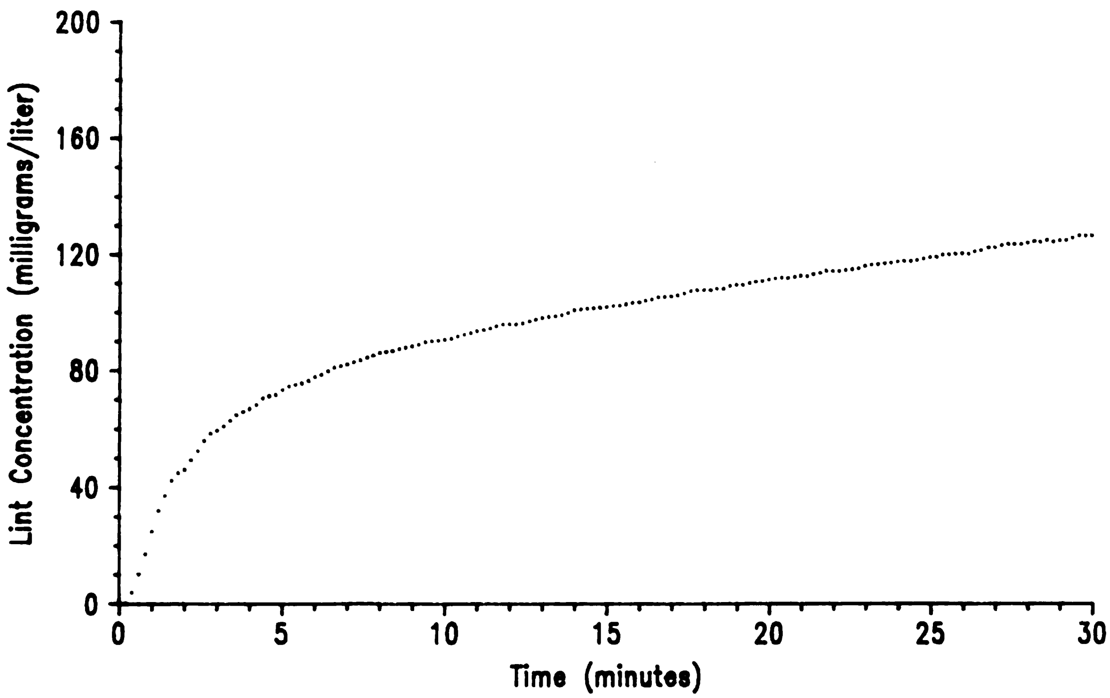
RUN - 38 Frequency - 2.25 Stroke Length - 160



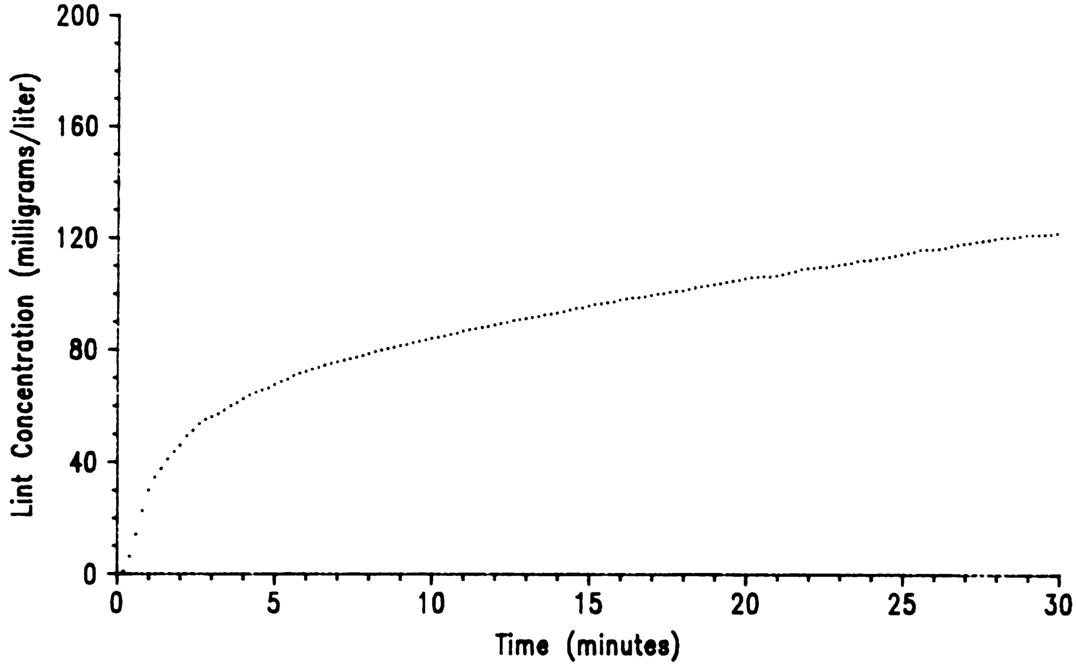
RUN - 39 Frequency - 1.00 Stroke Length - 200



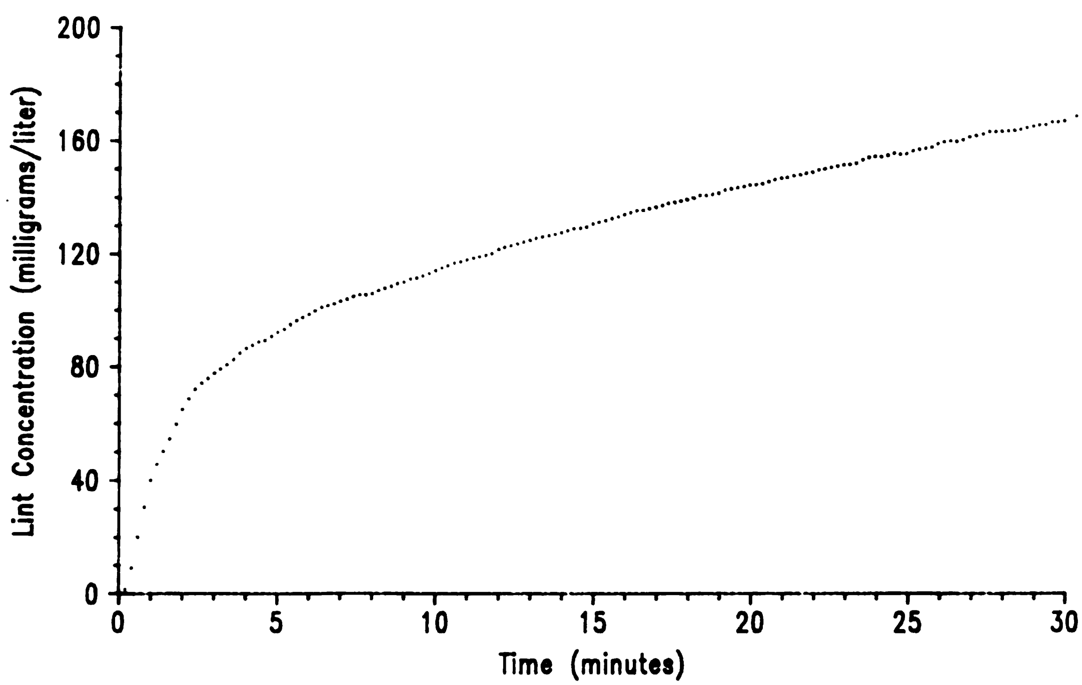
RUN - 40 Frequency - 1.25 Stroke Length - 200



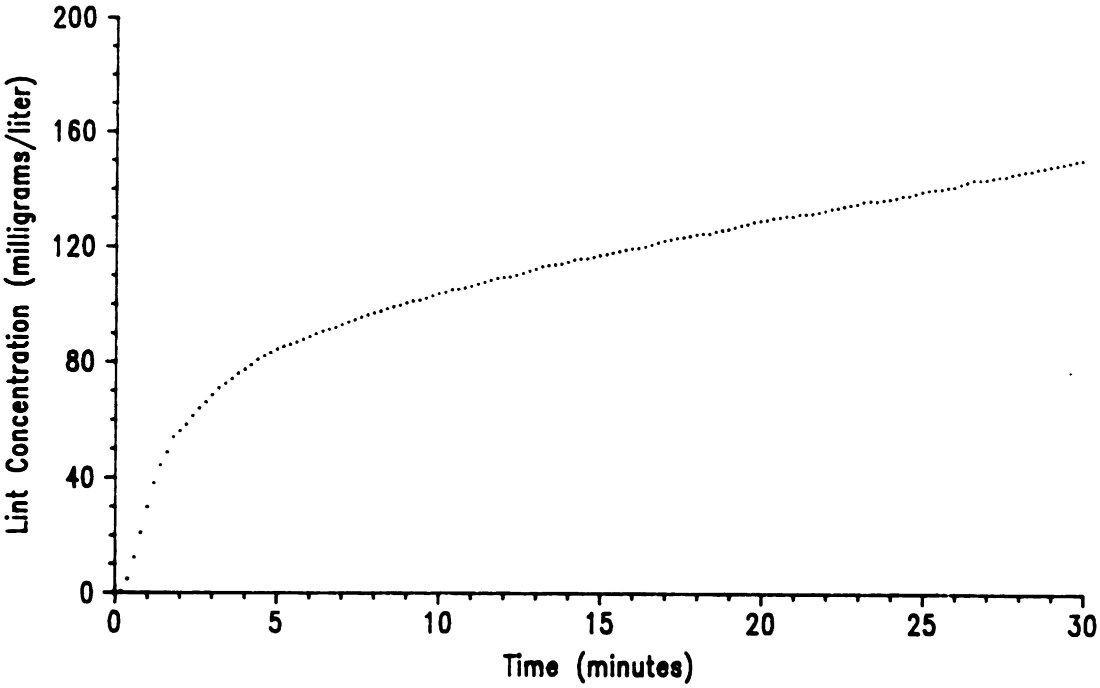
RUN - 41 Frequency - 1.50 Stroke Length - 200



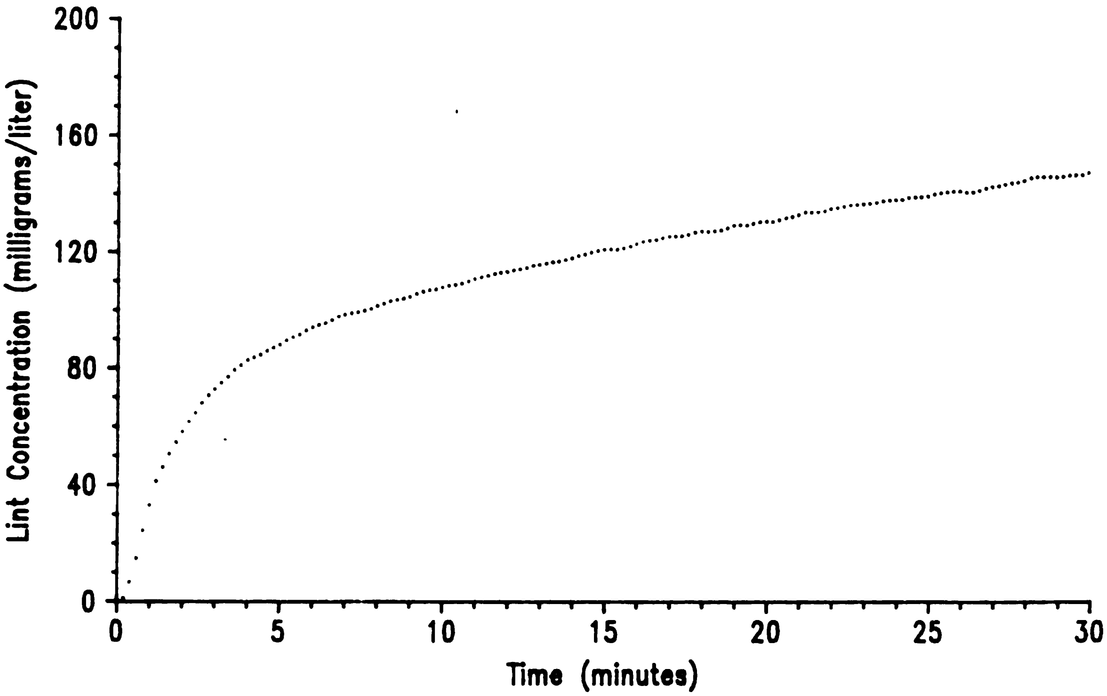
RUN - 42 Frequency - 1.75 Stroke Length - 200



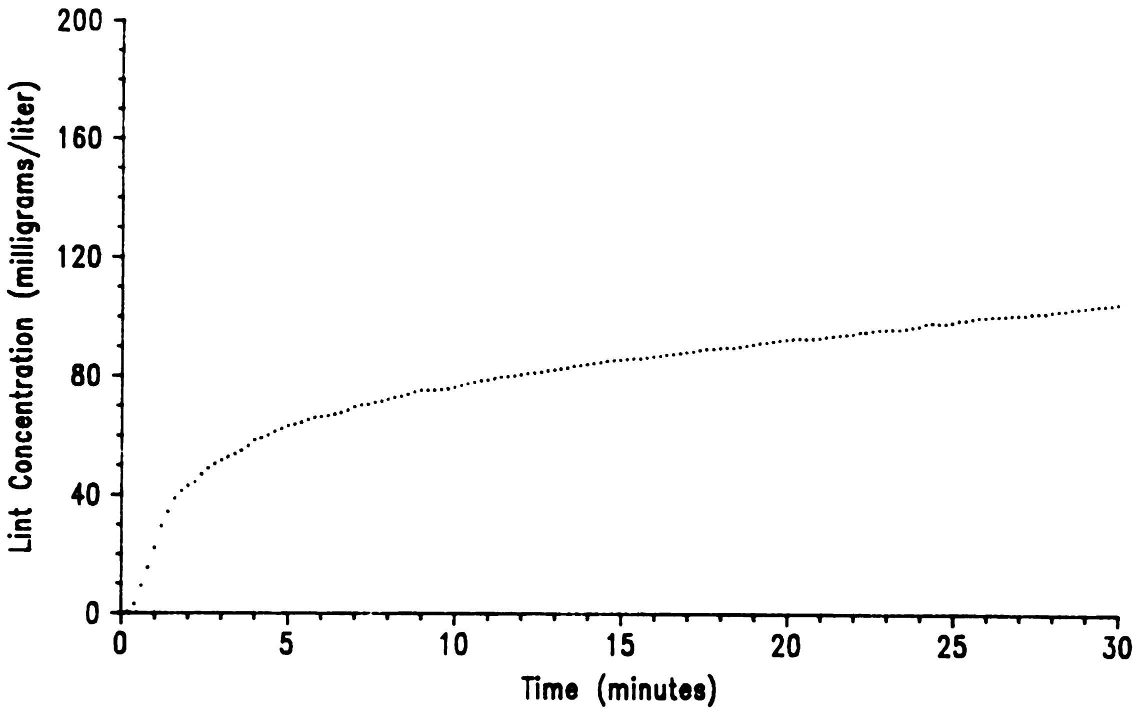
RUN - 43 Frequency - 1.75 Stroke Length - 200



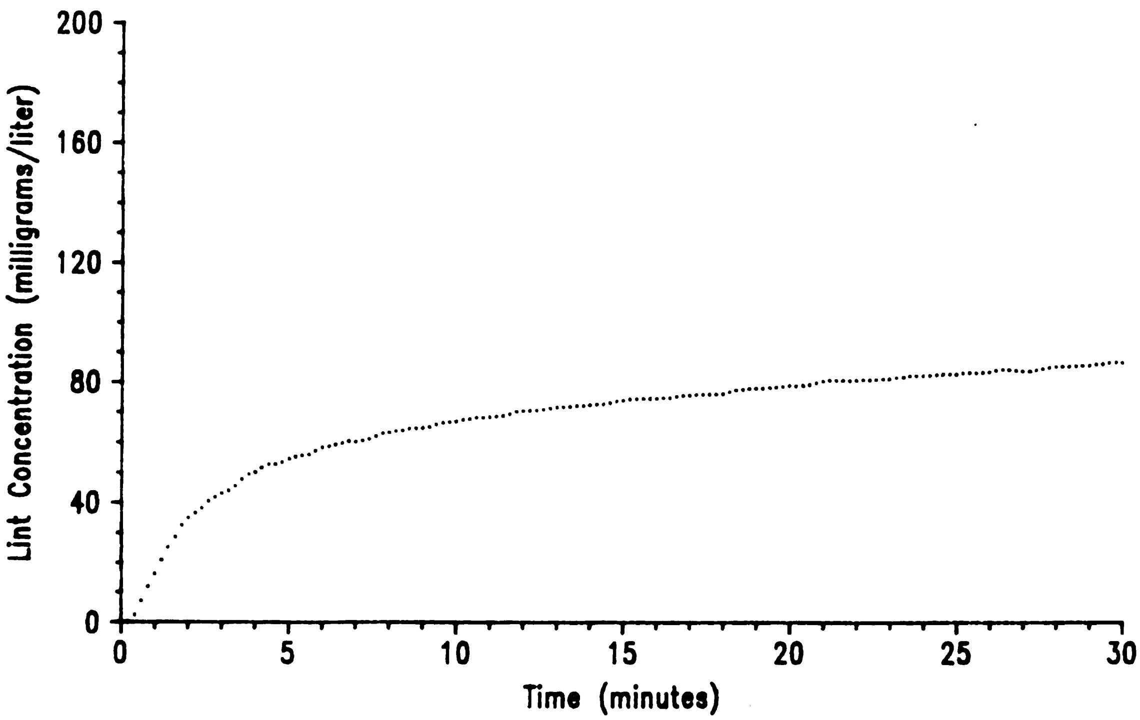
RUN - 44 Frequency - 1.50 Stroke Length - 200



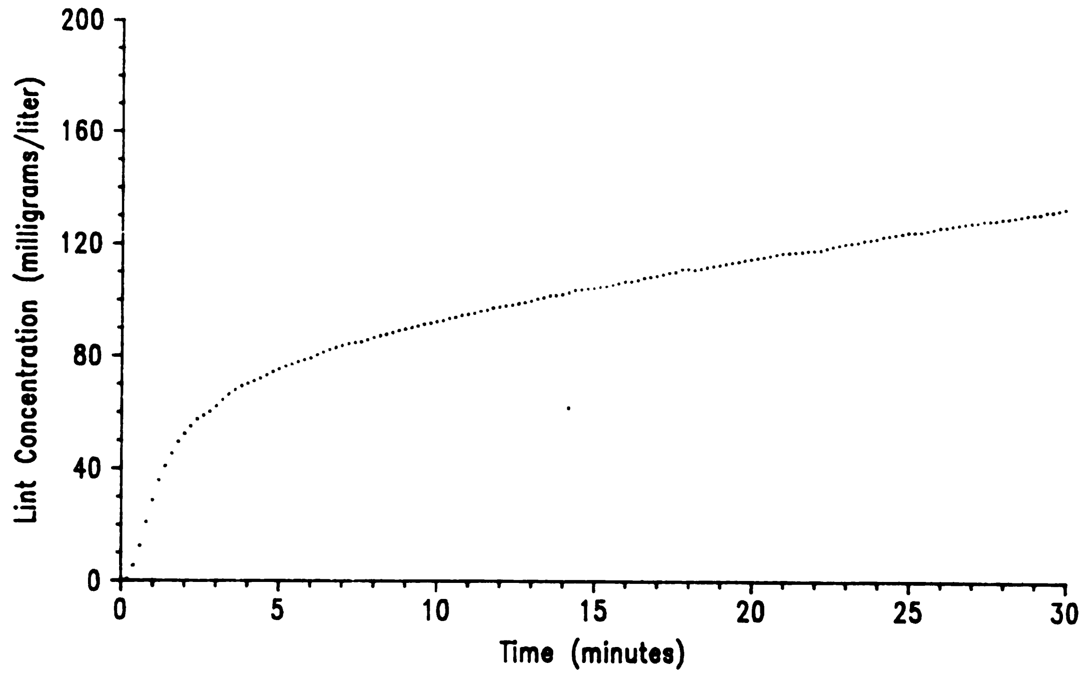
RUN - 45 Frequency - 1.25 Stroke Length - 200



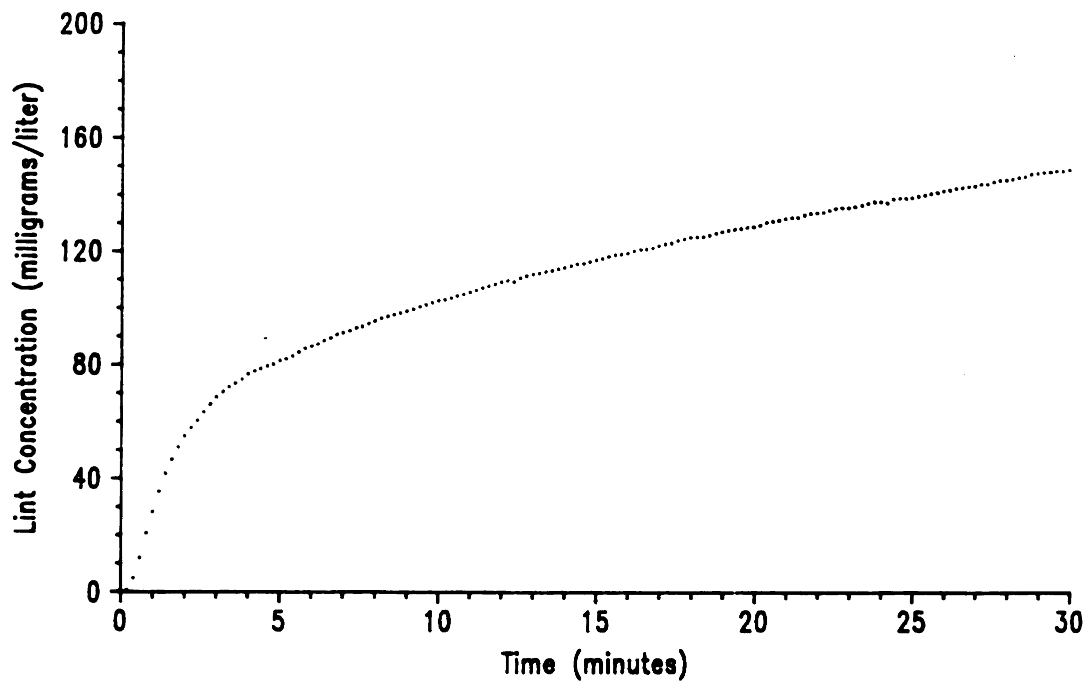
RUN - 46 Frequency - 1.00 Stroke Length - 200



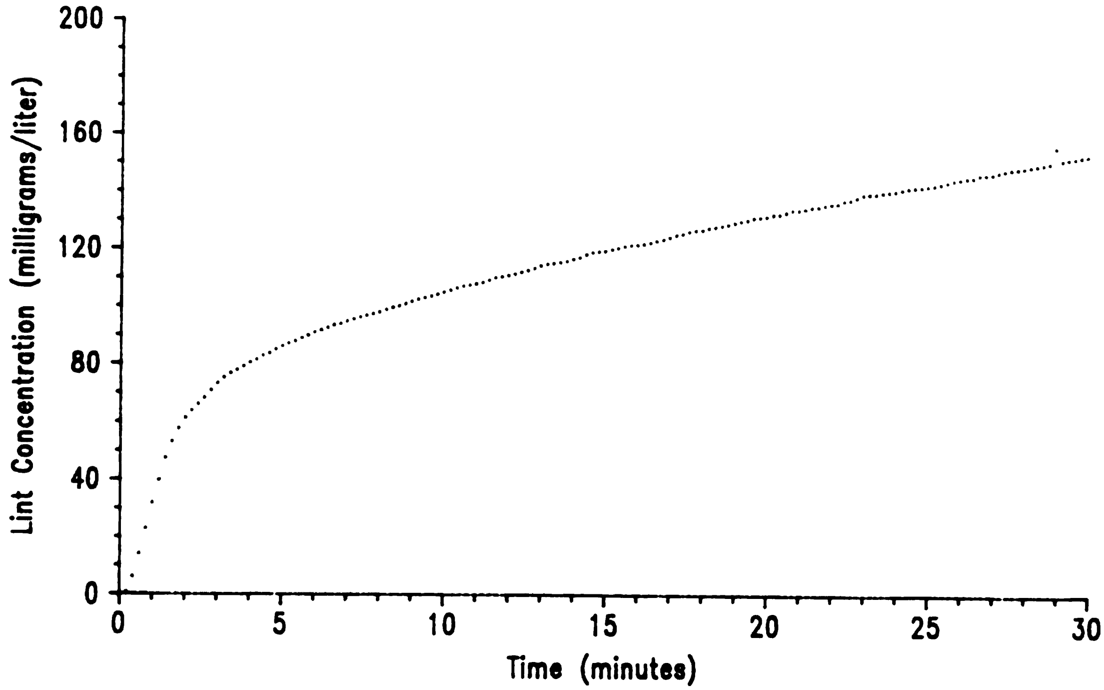
RUN - 47 Frequency - 1.75 Stroke Length - 180



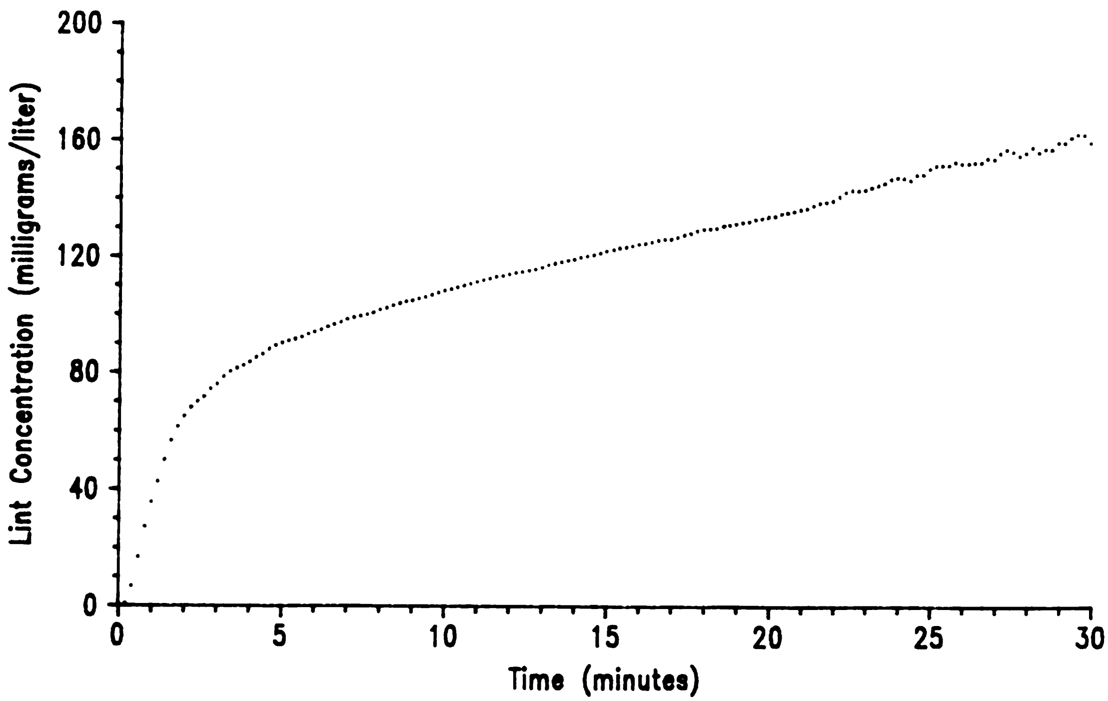
RUN - 48 Frequency - 1.75 Stroke Length - 180



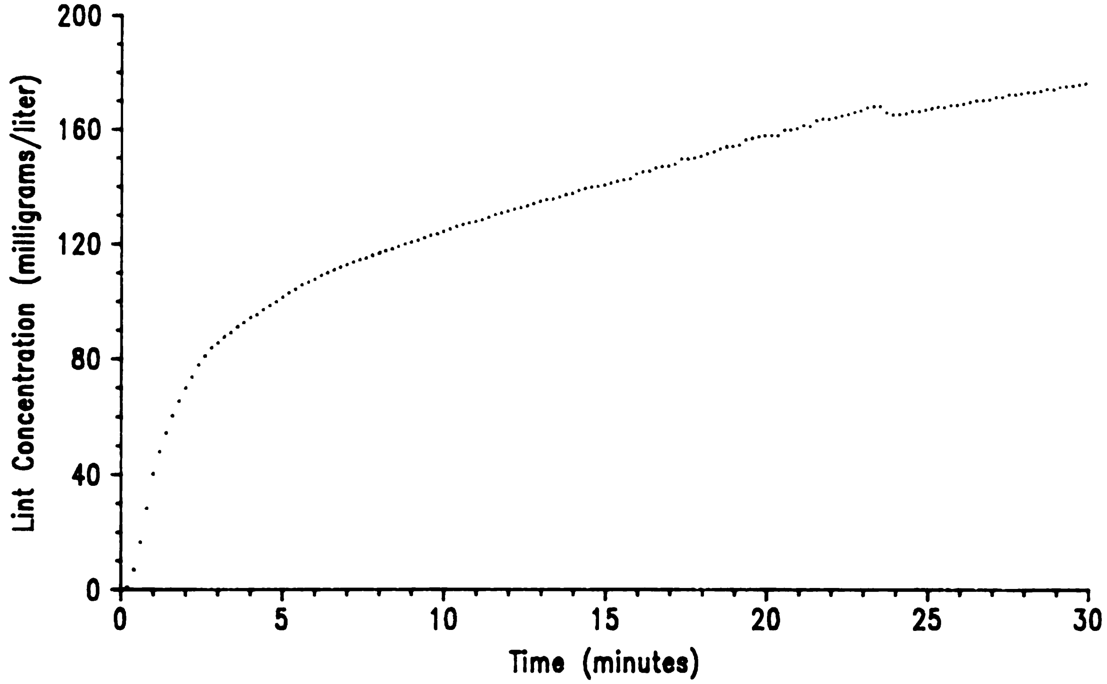
RUN - 49 Frequency - 2.00 Stroke Length - 180



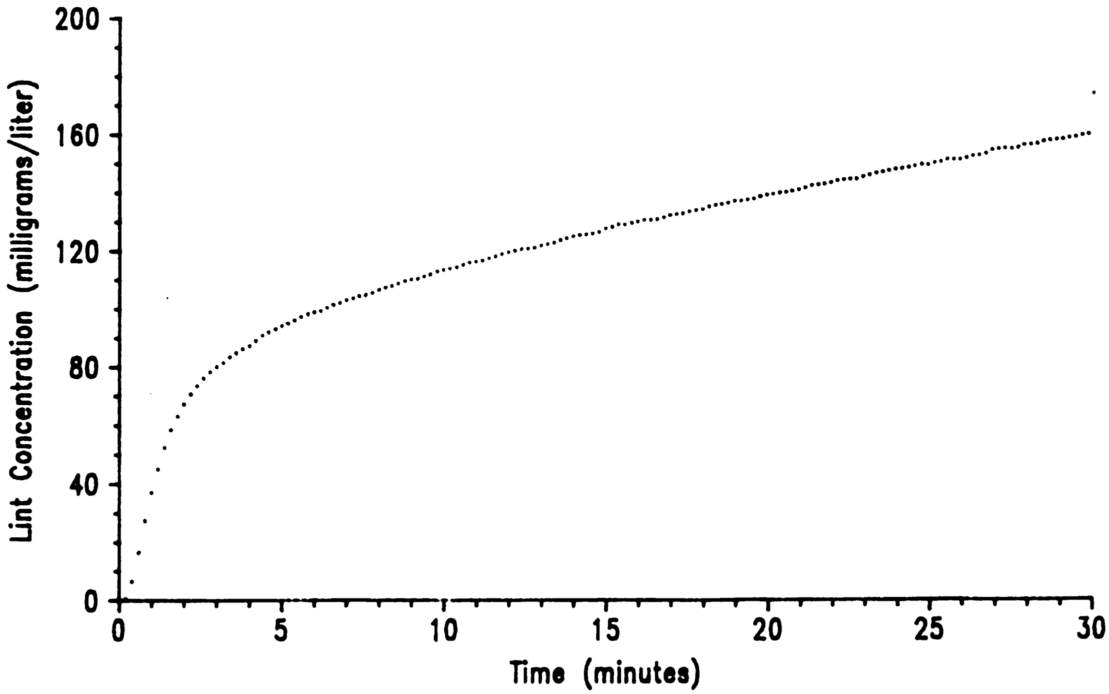
RUN - 50 Frequency - 2.00 Stroke Length - 180



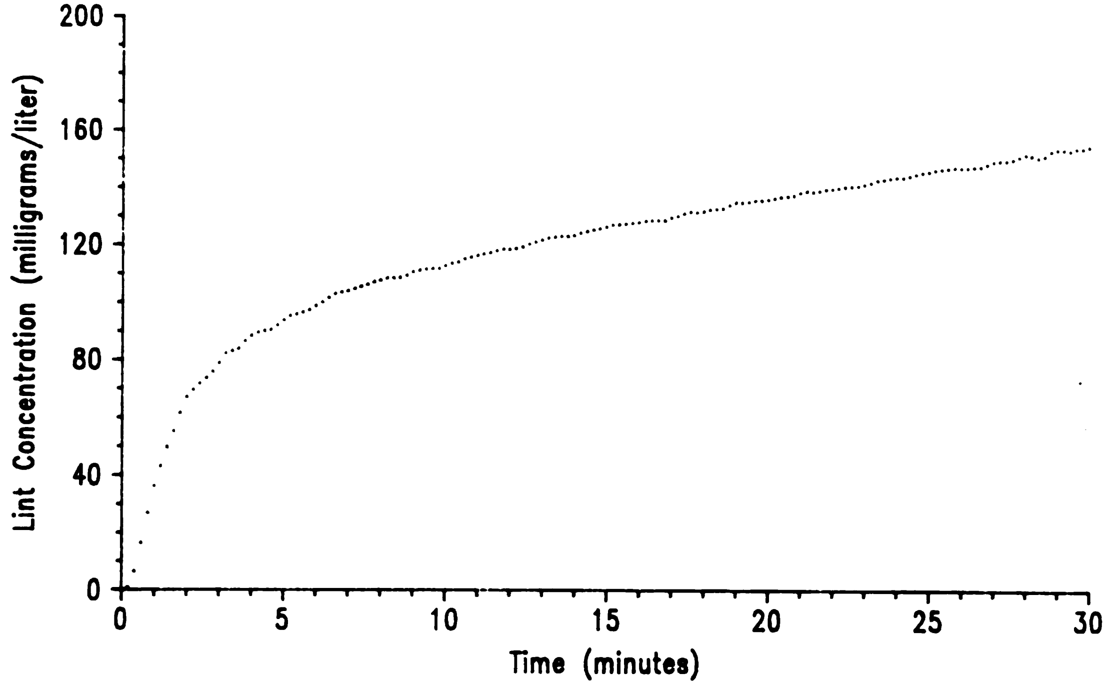
RUN - 51 Frequency - 2.50 Stroke Length - 140



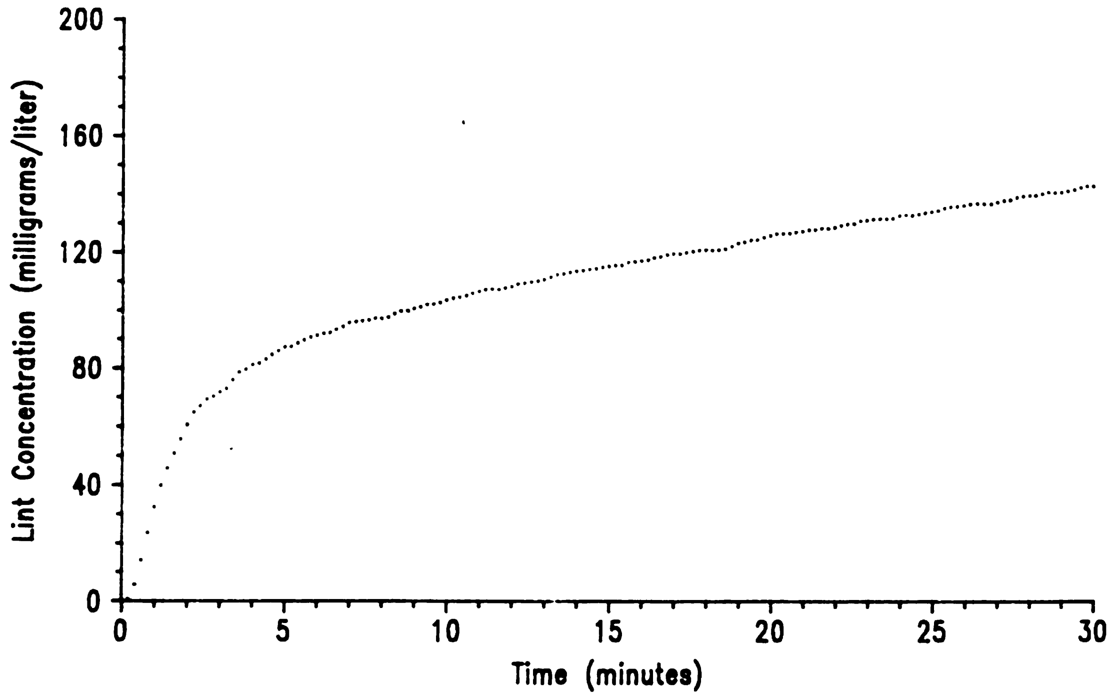
RUN - 52 Frequency - 2.50 Stroke Length - 140



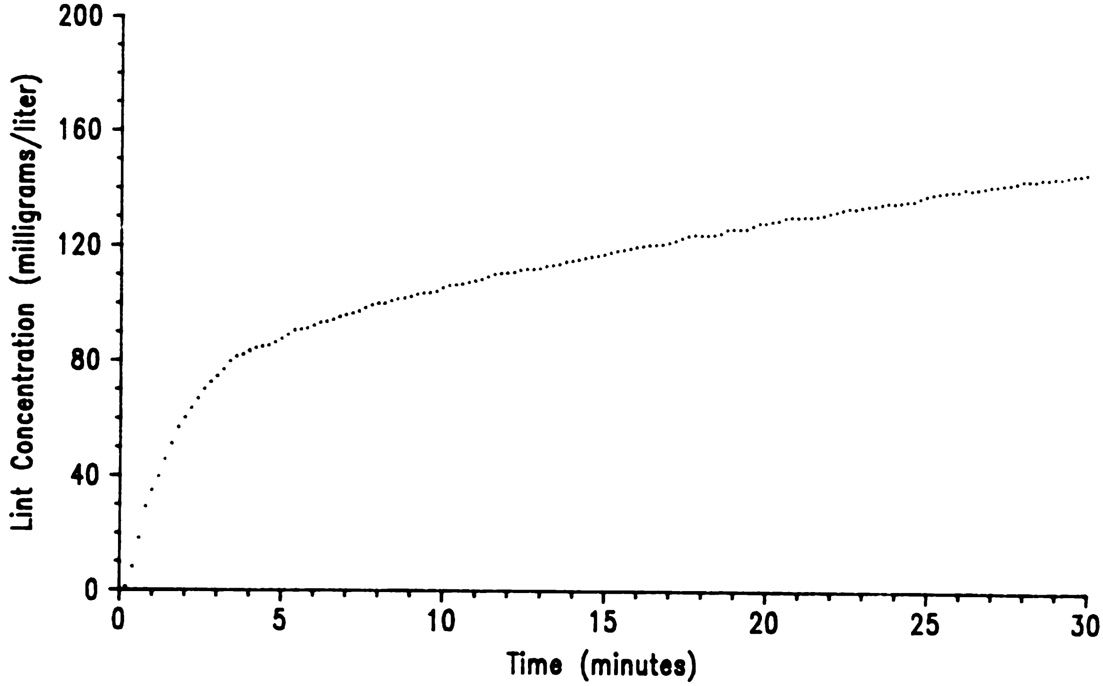
RUN - 53 Frequency - 2.50 Stroke Length - 120



RUN - 54 Frequency - 2.75 Stroke Length - 120



RUN - 55 Frequency - 2.75 Stroke Length - 120



RUN - 56 Frequency - 2.50 Stroke Length - 120