To the Graduate Council:

I am submitting herewith a dissertation written by Kevin D. Heath entitled "Carbon Dioxide Solvent Substitute: Fundamental Study of Aqueous Dispersions in Supercritical Carbon Dioxide." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Chemical Engineering.

H D Cochran

Dr. H. D. Cochran, Major Professor

We have read this dissertation

and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor

and Dean of The Graduate School

CARBON DIOXIDE SOLVENT SUBSTITUTE: FUNDAMENTAL STUDY OF AQUEOUS DISPERSIONS IN SUPERCRITICAL CARBON DIOXIDE

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Kevin D. Heath

August, 1999

DEDICATION

This dissertation is dedicated to my wife Angela, whom I will always love and cherish. Without her support this dissertation would not have been possible. This dedication is also extended to my parents Robert and Kay, and my sister Shauna. Shauna has given me the motivation, love, and support to accomplish anything.

ACKNOWLEGEMENTS

The author wishes to thank Oak Ridge National Laboratory for the financial support. Several people contributed in significant measure to this program. The author's deepest appreciation and thanks go to the following people:

Hank Cochran, major professor, for his invaluable guidance and friendship.

Alexander Van Hook, committee member, for his support and kindness starting back when I was an undergraduate.

Charlie Moore, committee member, for his encouragement and understanding.

Dr. Paul Bienkowski, committee member, for advice and sharing his laboratory at the University of Tennessee for the preliminary testing of the apparatus.

Doug Fielden and coworkers at the University of Tennessee machine shop for their innumerable suggestions in constructing the apparatus.

Vinod Shah for his assistance with initial operation.

Max Cronan and coworkers at the Oak Ridge National Laboratory's Plant and Equipment Division for their continuous help with modifications.

David DePaoli and Costas Tsouris for their technical advice on electrode geometry and electrodispersions.

Horng (James) Dai for his assistance with light scattering and LabVIEW programming.

Hung-Chih (Frank) Li for his assistance with the ternary Peng-Robinson code.

Lois Hamm for her colorful conversations and entertaining gossip that kept me laughing throughout the day.

My friends Danny Cribbs and Josh Nichols for all the enjoyable memories on the lake, at concerts, and living life—"Bawitdaba".

Last , but definitely not least, the author is eternally grateful to his wife for her love, understanding, and patience as he strives for academic excellence.

ABSTRACT

A new method for dispersing aqueous liquid into supercritical carbon dioxide (SCO_2) was demonstrated using pulsed high-voltage electric fields. System conditions were varied from 35°C to 65°C at pressures from 10.34 MPa to 17.24 MPa. The apparatus was equipped with a laser light scattering system so that it was possible to determine the dependence of mean droplet size with various operating parameters such as field strength, pulse frequency, flow rate, temperature, and pressure. An extraction of ethanol from a 10 vol. % aqueous ethanol solution was performed, and compared with theoretical flash calculations using a ternary form of the Peng-Robinson equation of state. The extraction process was shown to occur with 100 % efficiency for a single theoretical stage.

The effect of varying field strength, pulse frequency, and liquid flow rate was examined on $<d_p>$, the number averaged droplet diameter, for dispersing pure water in SCO_2 at 10.34 MPa and 40°C. This was the first study of electrospraying of a liquid into a dense, supercritical fluid. Increases in field strength caused $<d_p>$ to increase. This is opposite the effect normally observed when operating in the cone-jet mode of electrosprays. It is believed that the mode of electrospray in this research may extend past the cone-jet mode, because beyond the cone-jet mode increases in instability and polydispersity can increase $<d_p>$. In all cases, whether increasing flow rate or field strength, lowering the pulse frequency appeared to give smaller $<d_p>$. The effect of increasing $<d_p>$ with increasing flow rate demonstrated by other researchers was confirmed.

A study varying temperature, pressure, and flow rate was completed using a liquid feed containing 10 vol. % aqueous ethanol (fermentation limit). These experiments examined the effect of flow rate (2-12 ml/min), pressure (10.34, 13.79, 17.24 MPa), and temperature (35, 47, 65°C) on the mean droplet size of the dispersion while holding field strength and pulse frequency constant. For a pressure of 10.34 MPa, it was observed that increasing temperature increases $<d_p>$; however, at 17.24 MPa increasing temperature decreases $<d_p>$. This reversal was also seen when plotting the data for constant temperature. At 35°C, it was observed that increasing pressure increases $<d_p>$, again however, for 65°C increasing pressure decreased $<d_p>$. Although, these observations initially appear puzzling, they were shown to correlate with surface tension, one of the several physical properties of the system that are changing with both temperature and pressure.

Extraction of ethanol from a 10 vol. % aqueous ethanol solution using SCO_2 in the electrodispersion cell (EDC) appears to be possible. It occurs with essentially 100 % efficiency for a single theoretical stage. This was confirmed for four state conditions (10.34 and 17.24 MPa each at 35 and 65°C) and verified using a ternary Peng-Robinson equation of state by examining the experimental and theoretical separation factors . This observation was further supported by linear rates of extraction for both ethanol and water with molar flow rates of CO_2. An important conclusion from the ethanol mass transfer experiments is that the micron-size ethanol-water dispersion was very efficiently coalesced by the wire-mesh electrode so that entrainment of liquid in the CO_2 flowing from the cell was negligible. It was hoped that CO_2 flow rates could be reached where the rate of extraction was much higher than that observed in the null experiment (no

electric field) and where the ethanol extraction was mass transfer limited. Unfortunately, this was not achieved because of the limited capacity of the condensers.

LIST OF FIGURES

FIGURE **PAGE**

CHAPTER I

INTRODUCTION

1.1 Background

For some time the United States chemical industries and government have been pursuing alternate substances to replace organic solvents which are currently used in most industrial synthesis and separation operations. These organic solvents can be toxic, flammable, explosive, carcinogenic, polluting, etc. Within their search, supercritical carbon dioxide (CO_2) has been noted as their leading choice for solvent substitution (1). An essentially benign solvent, CO_2 is produced in large excess; so its use as a solvent would entail no additional release to the atmosphere that would contribute to the greenhouse effect. Technologies have been developed for many separation and reaction processes using supercritical CO_2 as a solvent, and several large commercial applications are in operation throughout the U.S., Europe, and Japan. However, because of the low solubility of water in supercritical CO_2 and poor mass transfer performance of conventional contactors between water and CO_2 phases, many potential commercial processes involving aqueous solutions have so far been beyond the range of commercial application of supercritical CO_2.

Techniques for producing aqueous dispersions in supercritical CO_2 have been sought for over a decade, and only recently (2-7) have a few successful surfactants been found that result in reverse miscelles in supercritical CO_2 with small quantities of water in their cores. Small angle neutron and x-ray scattering (SANS and SAXS) methods have been used to characterize the size and shape of many these systems (8-13). The present

thesis is part of a broader ongoing research program focused on systems with supercritical solvents. In this broader research program, SAXS and SANS studies have played an important role, and as a member of the research team, the present author participated in several of these scattering studies (see Appendix A). In order to accurately interpret the results of scattering experiments, one needs to know the concentration and density of the system under study. The present author initiated density measurements which are being continued by others to help with the analysis of prior scattering work (see Appendix B).

These successes of incorporating aqueous dispersions in supercritical CO_2 using surfactants have attracted a great deal of attention because of the widespread importance of their potential applications. To date, the highest water-to-surfactant molar ratio attained has been 32, which allows a maximum of 2 wt. % water to be soluble in CO_2 (2). This value is 10 times the amount of water that is soluble in pure CO_2. Although much progress has been achieved, the known surfactants (perfluoro polymers) are very costly, both to manufacture and to separate from the final product when finished.

An alternate approach may exist to produce dispersions of submicron-sized aqueous droplets in supercritical CO_2 without the use of surfactants. Over the past fifteen years, researchers in the Chemical and Energy Research Section of the Chemical Technology Division of Oak Ridge National Laboratory (ORNL) have explored drop and bubble formation behavior in electric fields from fundamental and applied perspectives. Techniques have been developed for producing aqueous dispersions in non-conducting organic solvents using intense electric fields; these developments have resulted in patents for an emulsion phase contactor to be used in separations (14, 15) and for an electric

dispersion reactor (16-18). A number of applications have been explored (19-27), and successful technology transfer has resulted in commercialization. To date the solvents in which the submicron-sized dispersions have been produced have included many organic liquids but never supercritical fluids. Fundamental understanding of the production and stability of these dispersions produced by electric fields (dc, ac, or pulsed) is currently insufficient to make an a priori assessment of the feasibility of applying the technology using supercritical CO_2 as the non-conducting fluid.

1.2 Research Objectives

The goal of this project is to test feasibility of a technique which would broaden the range of separation and reaction processes in which supercritical carbon dioxide is applicable. The main objectives are first to test the feasibility of producing aqueous dispersions in supercritical carbon dioxide using high-voltage pulsed electric fields. This dispersion would be in the form of very fine droplets—most likely in the submicron range. Droplets of this size would greatly increase the amount of surface area compared to conventional contactors.

Once a dispersion has been demonstrated, characterization will follow. A study of the mean droplet size of the dispersion will be made using laser light scattering. The effect of various operating variables on the mean droplet size will be determined. In addition to droplet size, a characterization of mass transfer properties will be completed. This will be achieved by performing an ethanol extraction from water using supercritical carbon dioxide.

1.3 Overview

A literature review will follow containing recent advances in electrospraying of a fluid into a gas. Various aspects of electrospraying will be discussed including the effects of applied voltage, liquid flow rate, capillary and electrode geometry, liquid electrical conductivity, and other physical properties. In addition, prior work of extracting ethanol from water using supercritical carbon dioxide will be reviewed supporting the importance of accurate phase equilibria. The experimental approach will be presented along with complete descriptions for electrospraying aqueous liquid into supercritical carbon dioxide (SCO_2), size determination through the use of laser light scattering, and ethanol extraction using SCO_2. Next, the results and discussion are introduced demonstrating the feasibility of dispersing aqueous dispersions in SCO_2, the effects of field strength, pulse frequency, flow rate, temperature, and pressure on mean droplet size, and the effect of temperature and pressure on the extraction of ethanol from a 10 vol. % aqueous ethanol solution. After a presentation of the conclusions and recommendations, Appendix A cites four publications resulting from scattering experiments on various polymers in SCO_2 using both small angle x-ray and small angle neutron scattering sources. Appendix B describes the initiation of density measurements for various monomers in SCO_2. Various computer programs based on the Peng-Robinson equation of state were utilized and included in Appendix C for regression of binary interaction parameters, ternary code for comparison with literature data, and flash calculations using mass balances and ternary code.

CHAPTER II

LITERATURE REVIEW

2.1 Electrospraying of Liquid into Gas

Electrospraying is a process that relies on electrical forces to break a liquid into fine charged droplets. One of the simplest ways of implementing such a process is to feed a liquid with sufficient electric conductivity through a grounded metal capillary a few centimeters away from a conducting mesh screen, which is charged with a sufficiently high electric potential. Under the influences of the electric field, the liquid is often observed to form a cone-shaped interface at the outlet of the capillary, through the apex of which a fine thread of liquid is ejected. The liquid thread breaks up farther downstream to disperse into a spray of droplets which often shows a remarkably narrow size distribution (28-30). This mode of spray operation is commonly referred to as cone-jet mode (31). The reason for the monodispersity of the cone-jet electrospray is that axisymmetric surface wave instabilities dominate the liquid breakup, which results in a constant ratio between the primary droplet diameter and the thread diameter (32). For liquids with low viscosity, the ratio has been measured at about 1.89 (28, 30-33), consistent with the prediction of Rayleigh's capillary instability theory (33). Electrospraying is capable of producing monodisperse droplets in the size range from 5 nm up to 100 μm when operating in the cone-jet mode (28-31, 34-38). Cloupeau and Prunet-Foch (39) classify the main modes of electrospraying as: dripping, microdripping, cone-jet, spindle, simple jet, and ramified jet. When the mode of electrospraying extends past the cone-jet mode into the spindle, simple jet, or ramified jet mode, monodispersity

is no longer observed as inconsistencies in the liquid apex cause a large range of different droplet sizes to be emitted.

2.1.1 Applied voltage and pulse frequency

The applied voltage is a key parameter in establishing the cone-jet mode; various modes of spraying can be observed depending on the voltage applied (39-41). However, droplet size has been reported to be virtually independent of the applied voltage for liquids of relatively high electrical conductivity (42, 43). Tang and Gomez (32) observed that at low flow rates, the applied voltage seems to have a modest effect, and at larger flow rates, the droplet size decreases almost linearly with the increase of the applied voltage. This effect appears to be more pronounced for lower conductivity solutions. Figure 1 from their work illustrates this effect. This figure shows the effect of the applied voltage on the droplet size for different liquid flow rates and two solutions of heptane with 0.3 wt. % and 2 wt. % of Stadis 450. Stadis 450 is an antistatic additive used to enhance the electrical conductivity of heptane. The 0.3 % sample has an electrical conductivity and dielectric constant of 1.4×10^{-8} Ω^{-1} cm^{-1} and 1.93 compared to 6.7×10^{-8} Ω^{-1} cm^{-1} and 1.91 for the 2 % solution. Within the range of applied voltage that ensures a stable cone-jet, as much as 50 % reduction in size was observed for a fixed flow rate of the lower conductivity solution. Similar to the work of Hayati (41), as the voltage is increased, the liquid cone-jet is observed to shrink toward the capillary and the angle of the cone decreases, which results in an increase in the jet velocity and a decrease of the droplet size. This is required for mass conservation and the constancy of the ratio between the droplet diameter and the liquid thread diameter (28-30). If the mode of

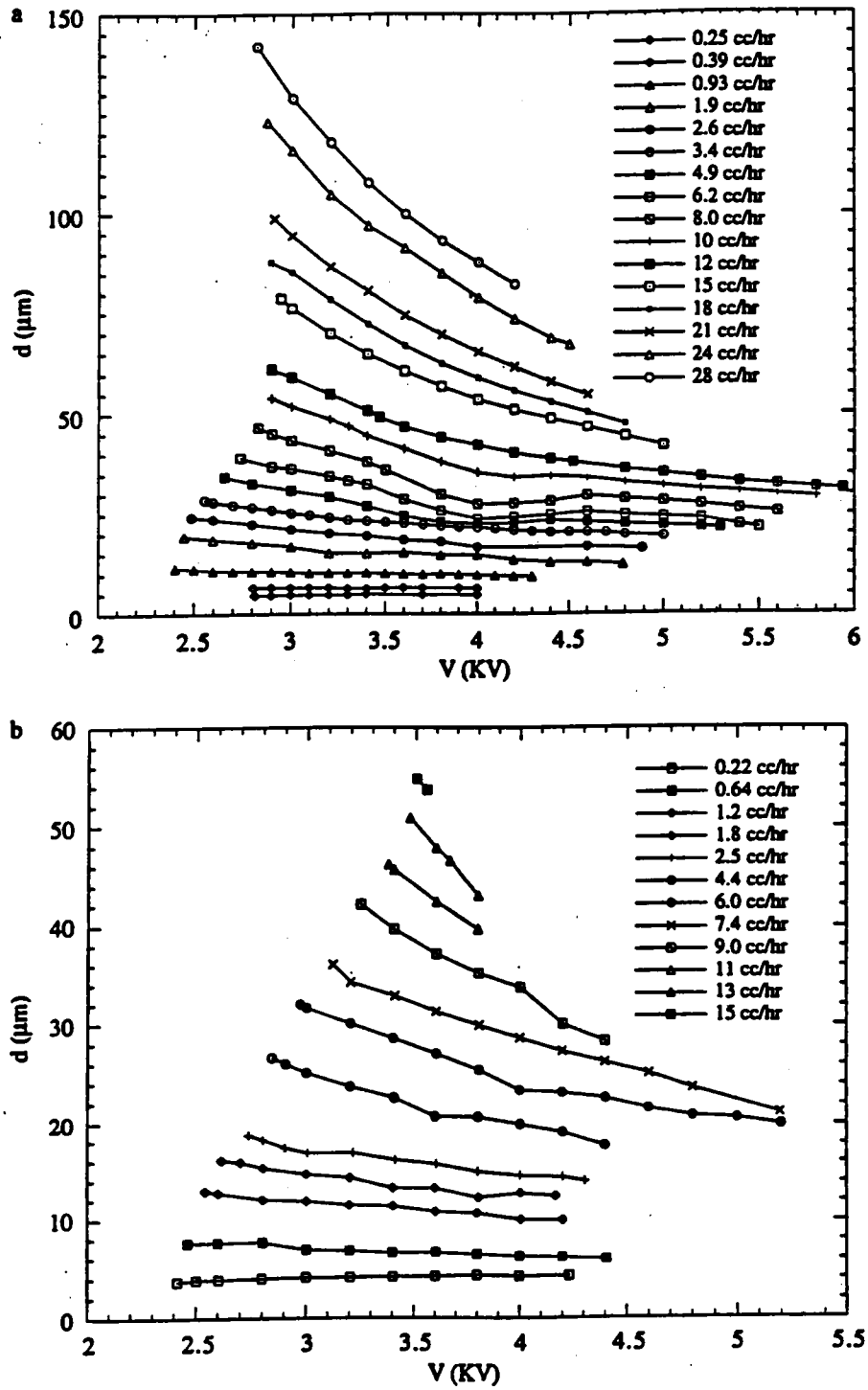


Figure 1. Droplet diameter versus voltage applied to the capillary for two heptane solutions at different flow rates: (a) heptane + 0.3 wt. % Stadis 450; (b) heptane + 2.0 wt. % Stadis 450, taken from Tang and Gomez (32).

electrospray is beyond the cone-jet mode, polydispersity is observed, and an increase in droplet size can be seen from increasing the field strength (39).

The frequency of a pulsed electric field affects the size of the droplets (44-46) as well. The most interesting behavior is found to occur when the electric field pulse frequency is in the vicinity of the natural oscillation frequency of the droplet (47). Near this natural frequency, a significant increase in field strength is required for droplet rupture (44); hence, this implies that this is the condition of maximum stability for the droplet. In addition, stability of the droplet drastically decreases on either side of the natural frequency. Varying the pulse frequency has also been used to study the effects of droplet oscillation on the rate of mass transfer for the case of continuous-phase control (45-46). If the proper combination of field strength and pulse frequency is utilized, the droplet will shatter, thereby creating a large amount of interfacial area.

2.1.2 Liquid flow rate

The flow rate is the dominant factor controlling the droplet size of the electrospray in the cone-jet mode; as shown in Figure 1 from Tang and Gomez (32), droplet sizes ranging more than two orders of magnitude can be easily generated from the cone-jet electrospray solely by varying the liquid flow rate. The droplet size dependence on liquid flow rate has been studied by various researchers (30, 32, 35, 36, 41, 42, 48). When the electrospray is operated at the onset voltage conditions, defined as the minimum voltage required for a cone-jet spray, the droplet size increases monotonically with the liquid flow rate; and the dependence can be fitted by a power law (29, 38, 49). If the mode of electrospray extends beyond the cone-jet mode, polydispersity dominates the

dispersion, however, similar trends of increasing mean droplet size with flow rate are still observed (39).

2.1.3 Capillary diameter and electrode configuration

In the cone-jet mode, droplet diameter is independent of capillary diameter (32). The capillary size, however, was found to significantly affect the cone-jet domain of the electrospray (31). The maximum liquid flow rate for the cone-jet electrospray decreases dramatically as the capillary diameter increases, which means that the cone-jet domain of the electrospray becomes smaller as the capillary size increases. Consequently, the larger the capillary size the narrower the operational domain of the cone-jet electrospray.

When varying the electrode position at different liquid flow rates, it was found that both the droplet size and the total electric current of the spray were independent of the electrode separation for a fixed liquid flow rate. This occurred as long as the spray was operated at the onset voltage conditions. The onset voltage was shown to increase monotonically with the increase of the separation between the capillary and the electrode at a given flow rate (33). This suggests that, once the cone-jet is established, the droplet size is independent of the geometric dimensions of the electrodes. It also implies that the onset condition of a cone-jet electrospray depends only on the electric field at the thread which is affected by the combination of the applied voltage and the electrode separation.

2.1.4 Liquid electric conductivity and other physical properties

From the above description, evidence indicates that the most important liquid physical property in controlling both the stability of the electrospray (31, 41) and the size of the droplet (42, 50) is the liquid electrical conductivity. A literature survey offers conflicting accounts of whether or not there is an upper limit to the liquid electrical

conductivity for the establishment of a stable electrospray. Jones and Thong (33) indicated that the maximum liquid electrical conductivity for a stable electrospray was $10^{-5}\ \Omega^{-1}\ cm^{-1}$. While others (51) claimed that a stable electrospray would be impossible to establish when the liquid conductivity was higher than $10^{-7}\ \Omega^{-1}\ cm^{-1}$. A similar conclusion was drawn (41) that a stable electrospray could be established only for semiconducting liquids, although no specific upper conductivity limit was mentioned. Yet another paper (50) suggests that no such limit exists. This opinion is supported by the fact that even stable liquid metal sprays can be established under vacuum conditions for the direct production of metal ions (52, 53). De la Mora and Loscertales (42) showed that stable electrosprays can be established for liquid conductivities spanning a five order of magnitude range from $10^{-2}\ \Omega^{-1}\ cm^{-1}$ to $10^{-7}\ \Omega^{-1}\ cm^{-1}$, with resulting electrosprays in the submicron size range. Perhaps, there is no upper limit to electrical conductivity for the establishment of stable electrosprays, but electrospraying of high conductivity liquids may require higher field strength at the cone tip. This higher field strength would lead to breakdown (arcing) in systems where the dielectric constant is high, but may be permitted with media which are more insulating such as under a vacuum.

In general, for the cone-jet electrospraying domain, results have been demonstrated that the maximum flow rate decreases as the liquid electrical conductivity increases (32). This behavior suggests that the cone-jet domain shifts toward smaller flow rates as the liquid electric conductivity increases, which implies that the maximum droplet size of the spray significantly decreases (31). Extending past the cone-jet mode may produce polydisperse dispersions, but from a practicality standpoint it also achieves greater operational flow rates which increases throughput.

Other liquid properties that are relevant to electrospraying are liquid viscosity, surface tension, and dielectric constant. The liquid viscosity is expected to play a significant role in the jet breakup process. As shown experimentally (28, 31, 33, 47), the jet breakup process of electrospray is controlled by a natural surface wave instability. The size of the droplet is consequently determined by the wavelength with maximum growth rate along the liquid thread, which will depend on the viscosity of the liquid. Since electrostatic liquid atomization occurs when the electric force overcomes the surface tension of the liquid, it is expected that the onset voltage for the stable cone-jet spray will increase with the liquid surface tension. In fact, the experimental results of Smith (46), using water mixed with different concentrations of nonionic surfactants, indicate that onset voltage scales with the square root of the liquid surface tension under fixed electrode configuration and liquid flow rate. Consequently, if the surface tension is large enough, a stable electrospray may not be established because the field required exceeds that for the electric breakdown in the gas surrounding the cone-jet. The same conclusion was also reached by Zeleny (54). Special considerations must be taken into account for the establishment of cone-jet electrospray with high surface tension liquids, such as pure water and aqueous solutions. It has been experimentally demonstrated that although stable aqueous electrosprays may be difficult to achieve in air, replacing the air with a better insulating medium such as gaseous CO_2 results in attainable and stable electrosprays (32,54). The effect of dielectric constant on the electrospray was addressed by de la Mora and Loscertales (42). Their results show that the dielectric constant does have an important effect on the current transmitting process in both the cone and the thread of the electrospray through charge relaxation in the liquid.

The charge relaxation time is defined as the ratio of the liquid's permeability to its electrical conductivity. For the case of a perfect conductor, the charge relaxation time is zero, and the entire drop surface carries the potential with no electric field internally. However, when the liquid droplet is not a perfect conductor, there will be a difference in electrical potential between the tip of the capillary and the apex of the drop. This potential drop ensures that the surface is subject to a tangential electric field. This tangential electric field along with a free surface charge, which is also supported by a non-perfect conductor, causes an electrical shear stress that can only be balanced by a viscous shear stress exerted by the drop liquid on the interface (55). Such electric shear stresses cause the sides of the droplet to move inward supporting the development and extension of the liquid thread. Hayati at al. (41) and others (56) agree that electrical shear stresses must be present if the cone-jet mode is to be achieved.

Ganan-Calvo (38) reported some initial conditions for operating in the cone-jet mode of electrospray for dispersing a polar liquid into a gas. The relationships involved physical properties such as K, ε_i, ε_o, γ, ρ, β, defined as liquid electrical conductivity, gas permittivity, permittivity in a vacuum, surface tension, liquid density, and dielectric constant, respectively; and system parameters such as R_c, Q, L_c, λ, t_h, and t_e, defined as characteristic jet radius (same order as droplet diameter), liquid flow rate, characteristic jet length, $\lambda = R_c / L_c << 1$, $t_h = R_c^3 / Q\lambda$, and $t_e = \varepsilon_i / K$, respectively. Ganan-Calvo's paper stated that when the residence time, t_h, becomes of the same order as charge relaxation time, t_e, the following relationships occur:

$$Q_{\min} \sim \frac{\gamma \varepsilon_i \beta^{1/2}}{\rho K} \qquad (1)$$

$$d_{\min} \sim \beta^{1/3}\left(\frac{\gamma\varepsilon_o^{\,2}}{\rho K}\right)^{1/3} \qquad (2)$$

$$I_{\min} \sim \left(\frac{\varepsilon_o\gamma^{\,2}}{\rho}\right)^{1/2} \qquad (3)$$

where Q_{min}, is the minimum liquid flow rate, d_{min} is the minimum average droplet diameter, and I_{min}, represents the minimum emitted current. These useful relationships allow one to speculate changes in physical or system properties and relate them to operational parameters.

Stable electrosprays in the cone-jet mode have been investigated for several decades now. The effects of electrospraying in liquids has been well documented (24, 26, 44-46) as well as electrospraying into gases (28, 32, 43, 54, 55), but no literature has been found on electrospraying into supercritical fluids where the continuous phase can be controlled to achieved properties ranging from those of liquids to those of gases. This project will focus specifically on this region to explore the interesting range of property variation not previously studied.

2.2 Supercritical Carbon Dioxide (SCO_2) Extraction of Ethanol

Since the commercialization of supercritical fluid (SCF) extraction processes in the late 1960s (57), interest in the separation technique has increased considerably. The continued interest in supercritical extraction is attributed to the advantages that the process offers, such as high mass-transfer rates, favorable selectivities, and low operating temperatures (57). An important characteristic of a SCF is that small changes in pressure or temperature can dramatically alter its density. This control of density ultimately determines a SCF's solvent power. This can be illustrated by the solubility parameter, δ,

which is shown for gaseous, liquid, and supercritical carbon dioxide as a function of pressure in Figure 2 (58). Developed in the late 40's independently by both G. Scatchard and J. H. Hildebrand, δ, the solubility parameter, is defined as the positive square root of the cohesive energy density, where the cohesive energy density is the energy change upon isothermal vaporization of the saturated liquid to the ideal-gas state (infinite volume). It is noticed that at –30 °C for gaseous carbon dioxide, the solubility parameter is essentially zero whereas the solubility parameter upon condensation to a liquid is comparable to that of a hydrocarbon. Above the critical temperature, it is possible to tune the solubility parameter continuously over a wide range by simply adjusting temperature or pressure. This phenomenon is the basis for many extraction processes that are currently being used around the world. Some of these extraction processes include extraction of caffeine from coffee and tea, and color and taste ingredients from hops, spices, and red peppers (57).

In the present research the extraction of ethanol from an aqueous ethanol solution by SCO_2 has been selected as a model in which to study the mass transfer performance of the electrodispersion cell. Although supercritical fluids have many favorable properties, they do possess some limitations. Perhaps the most often misinterpreted properties of SCFs are those of mass transport. It's often claimed that SCFs do not experience the mass transfer limitations of liquid extraction processes. This is untrue. If the extraction is from a liquid phase into a SCF phase, then the resistance to mass transfer in the liquid phase will often be rate determining. In this case, the gas-like mass transport properties of a SCF will not have an effect on the overall mass transfer rate. However, if the surface area of the liquid phase were greatly increased through electrodispersion, then the process

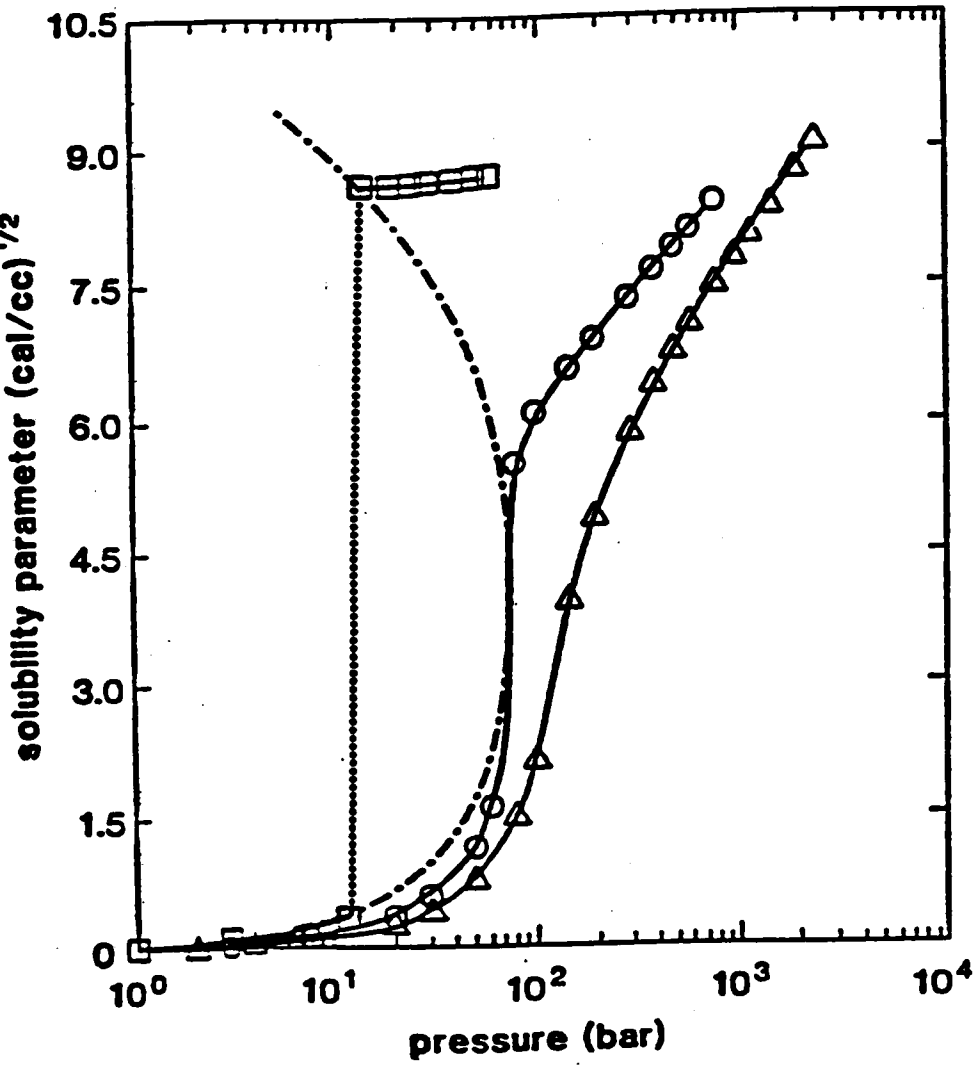


Figure 2. Solubility parameter of CO_2, δ, dependence on pressure at various temperatures (□: -30 °C, O: 31 °C, Δ: 70 °C), taken from Barton (58).

should be made much more efficient, and the SCF solvent would allow easy recovery of the solute through condensation.

When a supercritical extraction process is designed, the key factor to consider is the phase equilibria of the systems. These equilibria determine the maximum separation possible for the process, the solvent/feed ratio, and the selectivity for the extracted solutes. Many studies on the phase equilibria of carbon dioxide-ethanol-water systems have been reported in the literature (59-80). Inomata *et al.* (67) measured the equilibrium concentrations in the dilute ethanol region, and calculated separation factors well over 100 for ethanol feeds up to 10 wt. %. While these data were obtained for subcritical CO_2, they demonstrate the importance of knowing the phase equilibria. Kuk and Montagna (61) whose data were supported by many others (60, 62, 63, 67) noted that at 10.3 MPa and 313.2K, 90 wt. % ethanol could be recovered from feed mixtures of 10 wt. % ethanol by SCO_2 extraction. At atmospheric pressure, the ethanol-water binary system exhibits an azeotrope at 89.4 mol. %, which makes production of pure ethanol more difficult and costly. Furuta *et al.* (65, 66) discovered that it is possible to exceed the atmospheric pressure ethanol-water azeotropic composition (89.4 mol %) with a simple CO_2 extraction. Although Furuta *et al.* investigated experimentally and reported that an ethanol solution of more than 90 mol % could be obtained by extraction with CO_2 at 333.2 K and 10.1 MPa, the conditions to break the azeotropic limit were not specifically investigated. Lim *et al.* (64) did directly investigate Furuta's work, and concluded that aqueous ethanol can be concentrated above atmospheric azeotropic composition to as much as 95 mol % when the pressure in the ternary system CO_2-ethanol-water is below the critical pressure of the binary system CO_2-ethanol. Since then, others have repeated

this work and concentrated ethanol above the azeotrope concentration both with (68, 75) and without the use of additional entrainers (73, 76, 79).

Many researchers have reported experimental data for the ternary CO_2-ethanol-water system. Along with their experimental data almost all publications include an equation of state. For the current project, the Peng-Robinson (PR) (90) equation of state (EOS) was chosen for its relative simplicity and reasonable accuracy. The PR-EOS is written as

$$P = \frac{RT}{v-b} - \frac{a(T)}{v(v+b)+b(v-b)} \qquad (4)$$

The PR-EOS can be rewritten as

$$Z^3 - (1-B)Z^2 + (A-3B^2-2B)Z - (AB-B^2-B^3) = 0 \qquad (5)$$

where

$$A = \frac{a(T)P}{R^2T^2} \qquad B = \frac{bP}{RT} \qquad Z = \frac{Pv}{RT}$$

Equation 5 yields one or three roots depending on the number of phases in the system. In the two-phase region, the largest root is for the compressibility factor of the vapor while the smallest positive root corresponds to that of the liquid. The parameter *a(T)* is related to the intermolecular attraction forces dependent on temperature. The constant *b* is related to repulsive forces, which depends on the molecular size.

At the critical point,

$$a(T_c) = 0.45724\frac{R^2T_c^{\,2}}{P_c^{\,2}} \qquad (6)$$

$$b(T_c) = 0.0778\frac{RT_c}{P_c} \qquad (7)$$

and at other temperatures,

$$a(T) = a(T_c)\left[1 + \kappa_i\sqrt{1 - T_r}\right]^2 \qquad (8)$$

where T_r is the reduced temperature ($T_r = T / T_c$), $\kappa_i = 0.37464 + 1.54226\omega - 0.26992\omega^2$, and ω is the accentric factor.

Applying the thermodynamic relationship

$$\ln\frac{f}{p} = \int_0^P \left(\frac{v}{RT} - \frac{1}{P}\right)dP \qquad (9)$$

to equation 4, the following expression for the fugacity of a pure component can be derived

$$\ln\frac{f}{p} = Z - 1 - \ln(Z - B) - \frac{A}{2\sqrt{2}B}\ln\left(\frac{Z + 2.414B}{Z - 0.414B}\right) \qquad (10)$$

At equilibrium, an exact thermodynamic relationship exists that equates fugacity in each of the phases. The relation is

$$f^L = f^V \qquad (11)$$

When applying the PR-EOS to mixtures, mixing rules must be followed. Here the van der Waals mixing rules are used

$$a_m = \sum_i \sum_j x_i x_j a_{ij} \qquad (12)$$

$$b_m = \sum_i x_i b(T_c) \qquad (13)$$

$$a_{ij} = (1 - k_{ij})\sqrt{a(T)_i a(T)_j} \qquad (14)$$

The fugacity coefficient for mixtures, which is defined by $\phi_i = \frac{f_i}{y_i P}$ can be expressed in volume explicit terms

$$\ln \phi_i = \frac{b_i}{b_m}(Z-1) - \ln(Z-B) - \frac{A}{2\sqrt{2}B}\left[\frac{2\sum x_i a_{ij}}{a_m} - \frac{b_i}{b_m}\right]\ln\frac{Z+2.414B}{Z-0.414B} \qquad (15)$$

In equation 14, k_{ij}, is an empirically determined binary interaction parameter characterizing the binary formed by component i and component j. For this project, the binary systems are water-ethanol, water-CO_2, and ethanol- CO_2. Regression of binary literature data for these systems is performed, and the optimum value for k_{ij} is then obtained by minimizing the average absolute deviation (AAD) for the ethanol mole fraction in the vapor. The AAD is defined as

$$AAD = \frac{1}{n}\sum_{1}^{n}\frac{|\, y_i^{\,predicted} - y_i^{\,observed}\,|}{y_i^{\,observed}} \qquad (16)$$

CHAPTER III

EXPERIMENTAL APPROACH

3.1 Electrospraying of Aqueous Liquid into SCO_2

The central component in the experimental apparatus is a high pressure electrodispersion cell (EDC). The pressure cell obtained from Pressure Products Industries, Incorporated is capable of operating at pressures up to 34.47 MPa and temperatures from -30 to 175 °C. An EDC was fabricated from this pressure cell by installing fluid supply and withdrawal lines, electrical feedthroughs, and internal electrodes (see Figures 3 and 4). The inside chamber is in the shape of a cylinder with a diameter of 5.08 cm and a length of 17.78 cm. It has axial view windows on both sides allowing direct visualization of the cell contents. Experiments can be observed and recorded through these view windows with a high-speed video camera. High-speed video offers good resolution and immediate results. Analysis is carried out by selectively digitizing the video image. The camera signal is fed to a video recorder and then to a video monitor screen for visual observation. Visible light cannot be used to reliably observe objects less than ~0.5 μm; hence, any apparent disappearance of the dispersion may indicate droplets in the submicron range.

The aqueous liquid stream was fed into the pressure cell through a 0.55 mm OD capillary. The end of the capillary has been coned to minimize the surface area in contact with the aqueous droplet and maximize the electric field gradient. This procedure consists of a normal tube with a relatively heavy wall thickness (0.25 mm ID and 1.59

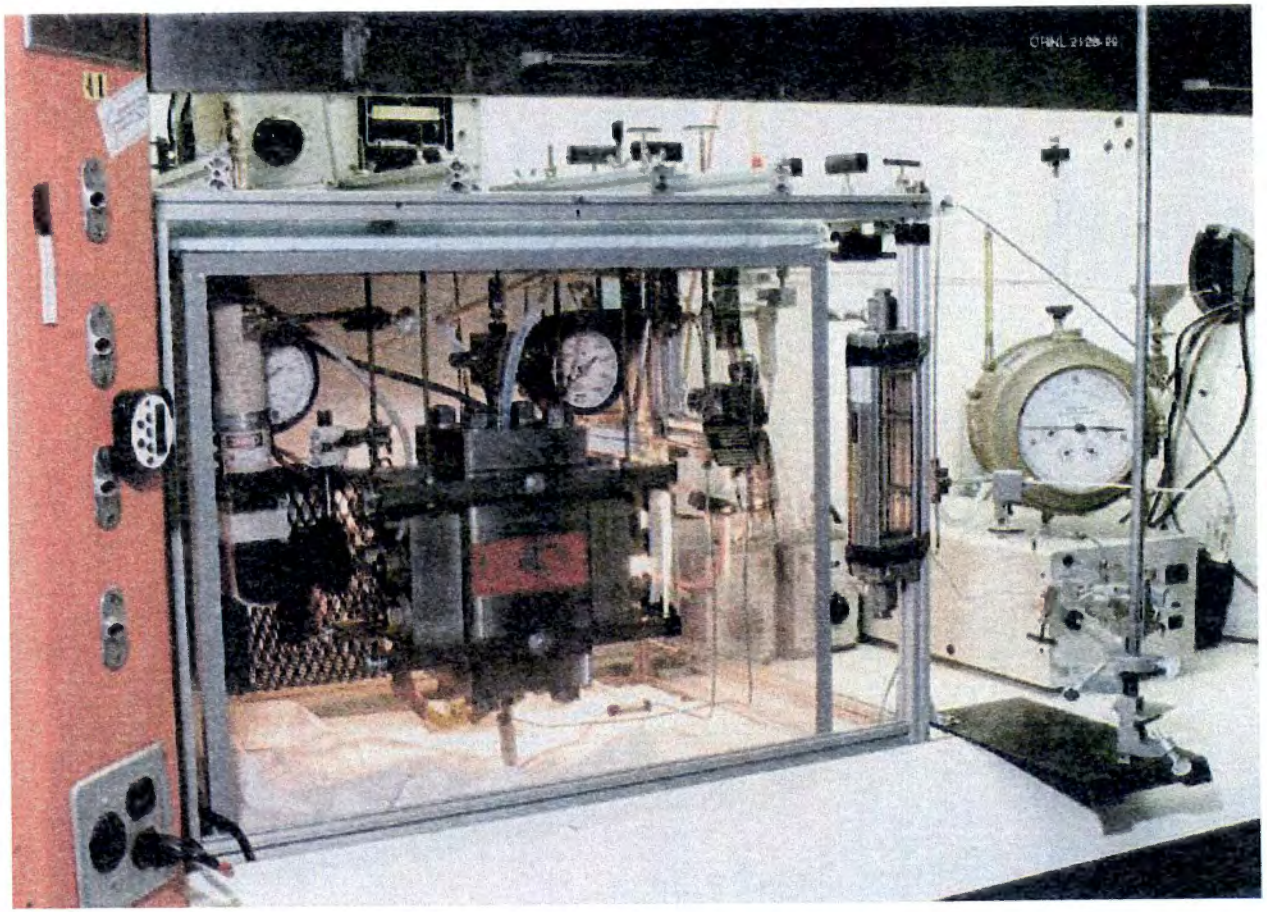


Figure 3. Electrodispersion cell (EDC) located at ORNL X-10 facility, building 4501, lab 208.

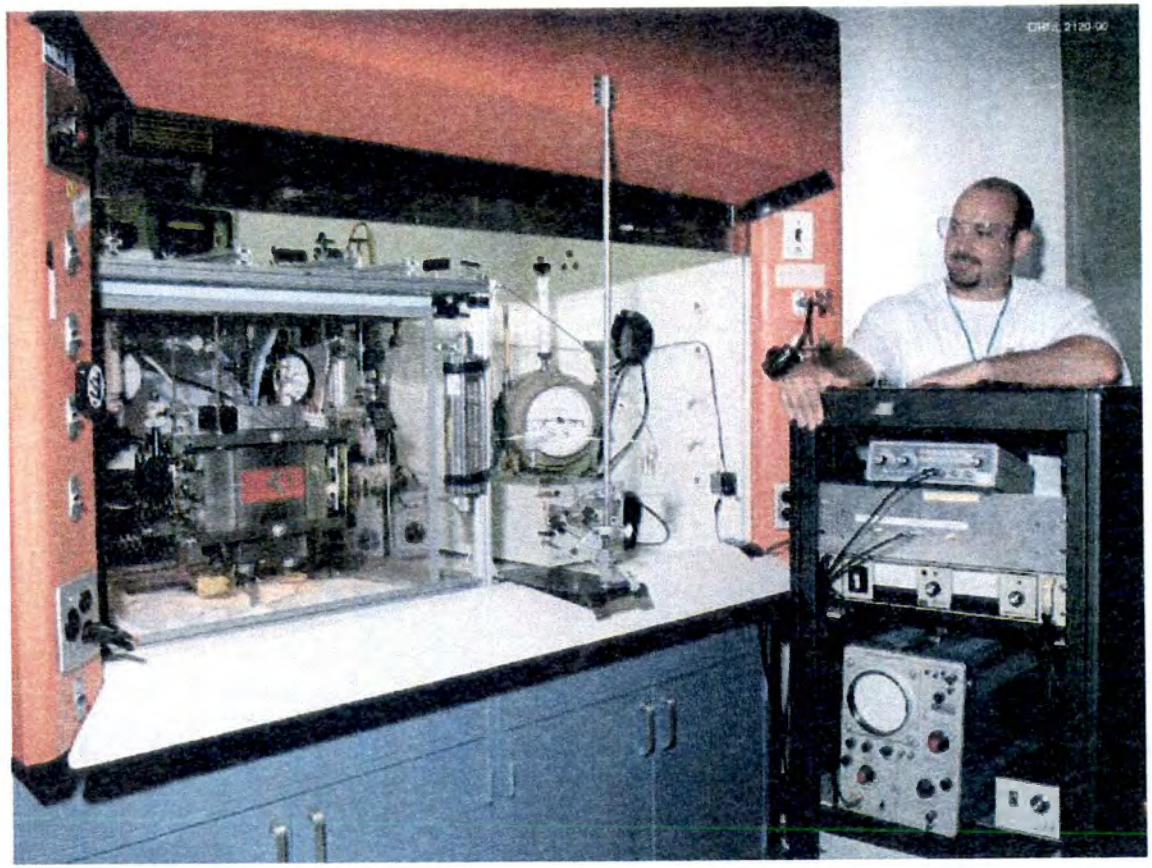


Figure 4. Electrodispersion cell (EDC) containing electric field generator located at ORNL X-10 facility, building 4501, lab 208.

mm OD see Figure 5a), that is mechanically ground with a taper resulting in a final outside tip diameter of 0.55 mm (see Figure 5b).

Initial testing of electrically dispersing aqueous liquid in SCO_2 was performed with a cylindrical glass cell, which had a 2.54 cm OD and a very restricting ID of 0.48 cm. This limitation of the ID forced an electrode design that was inappropriate for a continuous fine dispersion of water droplets. The glass cell design is shown in Figure 6. The inlet capillary was electrically grounded while the high voltage was delivered through a copper band (0.75 cm in width) wrapped around the OD of the tube. The cell was operated in co-current flow with at maximum operating pressure of 10.34 MPa.

Essentially the EDC disperses an aqueous medium by applying an electric field at or about a capillary tip, which is extended into the cell as shown in Figure 7. This electric field can cause the aqueous medium flowing from the capillary to instantaneously shatter into very small droplets (see Section 2.1). Two basic configurations are considered in this approach for applying the electrical field. The first method uses parallel metal plates which are approximately 2.54 cm squares that are 0.32 cm thick. These plates are positioned 1.27 cm on either side of the capillary such that the droplet falls directly between them. The distance between the plates is approximately 2.54 cm. The capillary tip in this configuration is made of an insulating material such as Teflon. An advantage of this configuration includes the reduction of coalescence of droplets on the electrode surface, which increases their residence time. This design has been successfully demonstrated to produce submicron water droplets in organic solvents (25) using field strengths of 9.5 kV/cm. The second configuration uses a metal wire mesh screen (3.81 cm diameter) placed 2.54 cm below the capillary (see Figure 8). For this

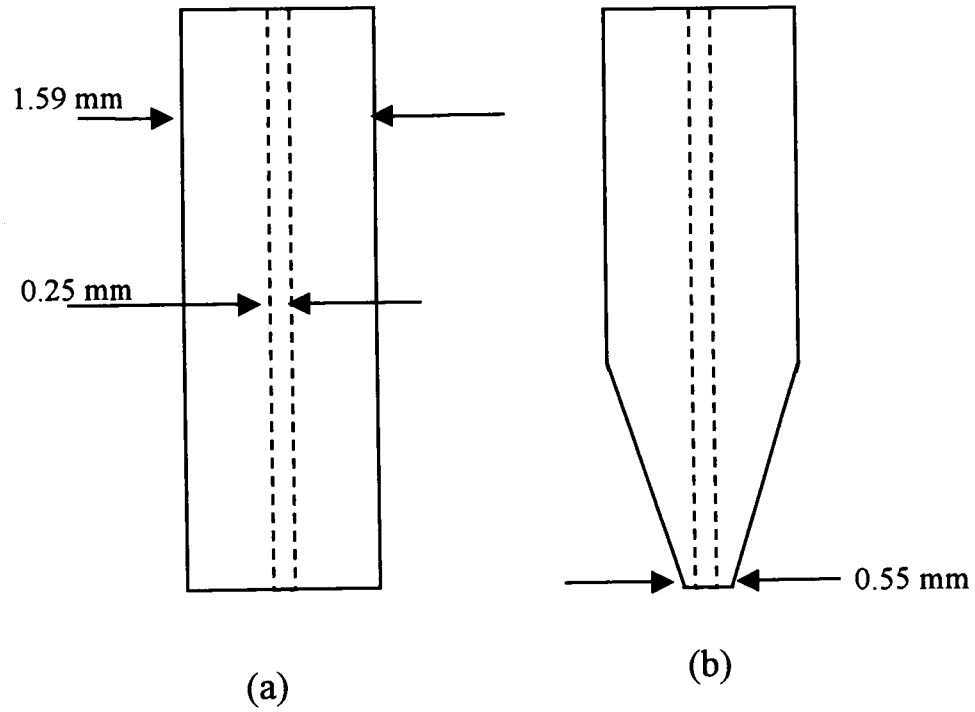


Figure 5. The end of the capillary has been coned to minimize the surface area in contact with the aqueous droplet and maximize the electric field gradient; (a) Normal tubing; (b) Same tube with coned tip.

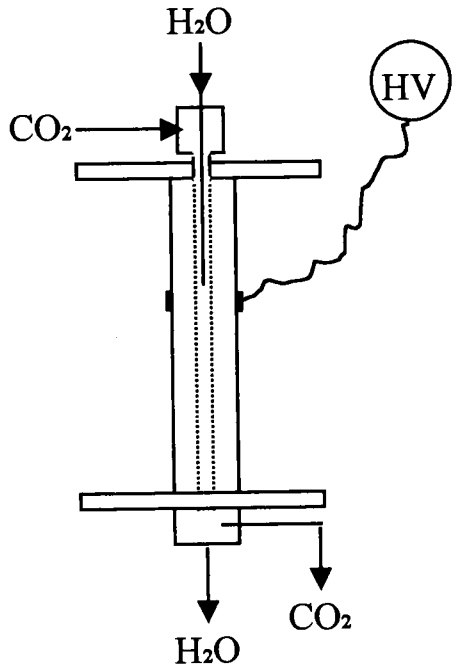


Figure 6. Glass tube cell with band electrode used in preliminary experiments.

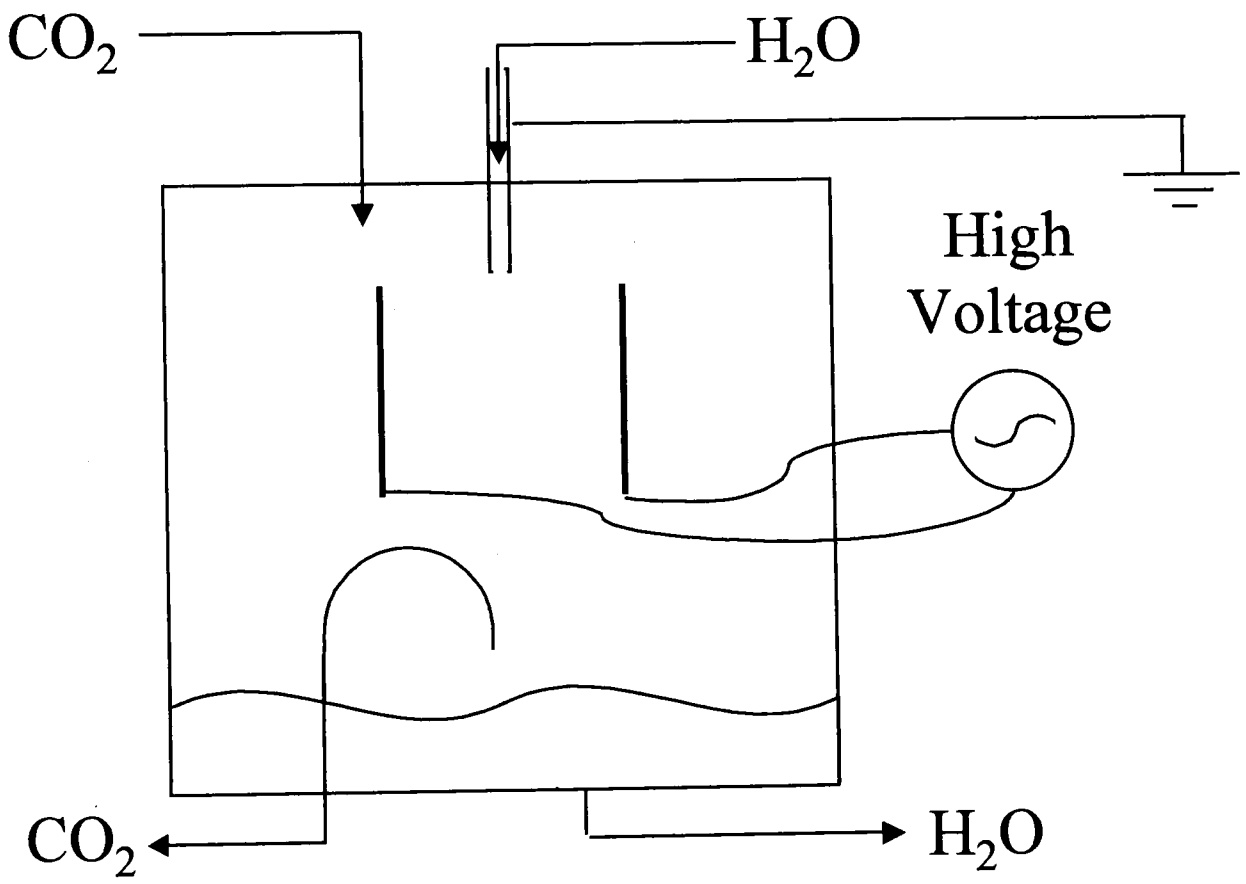


Figure 7. Electrodispersion cell with parallel electrode configuration.

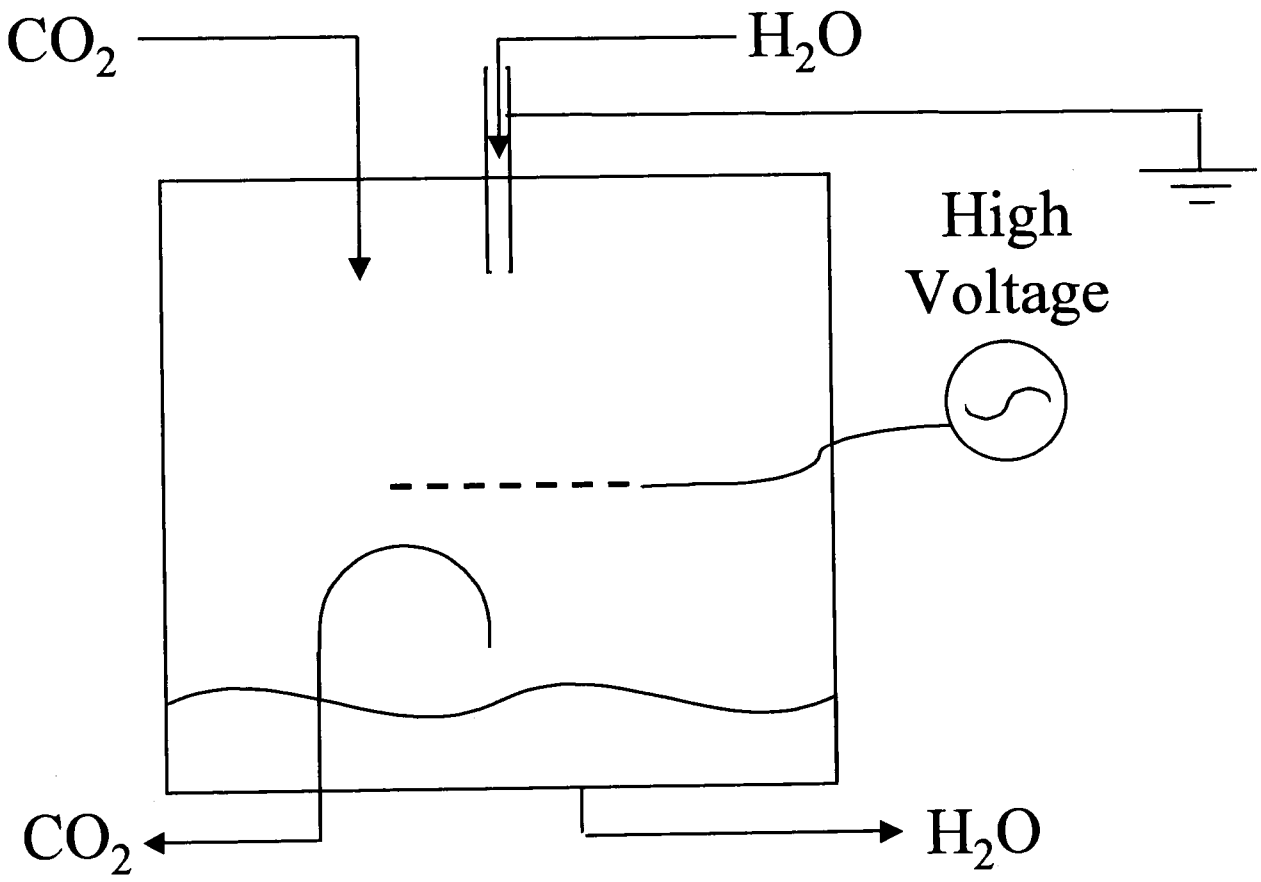


Figure 8. Electrodispersion cell with mesh screen electrode configuration.

design, the capillary is a grounded stainless steel tube; and the mesh screen carries the high voltage. This design is similar to that used in a previous study (26) for spraying water into cyclohexane except that in the previous work the mesh screen was used as the ground and the capillary carried the high voltage. Both electrode configurations have their advantages. The most important practical point to consider is water entrainment in the CO_2 phase. While the configuration in Figure 7 may improve residence time in the cell, the configuration in Figure 8 minimizes entrainment because the dispersion is immediately coalesced on the mesh screen. For this reason, the configuration found in Figure 8 was chosen and implemented in all of the following experiments.

The electric field was created by a custom-built pulsed dc voltage generator (see Figure 9). This generator utilizes a Global Specialties 4001 pulse generator and an Electronics Measurements, Inc., SCR dc power source that are supplied to a custom built pulser. The pulser then energizes a 12 V dc automotive high-voltage coil (Mallory Promaster). The coil produces the high voltage which is sent to the system through high-voltage diodes (Collmer Semiconductor, Inc., CS4107X30). The voltage is measured using a high-voltage step-down probe (Farnell HV40B; 1000:1) coupled with a Tektronix Inc., Type 504 oscilloscope to accurately determine the optimum voltages and frequencies. The oscilloscope was calibrated by Oak Ridge National Laboratory's Instrumentation and Controls Division to within 2 % error. This error is less than the width of the trace line. This system is capable of pulses up to 30 kV at 0-10 kHz.

Water flow is established with an SSI 222C high pressure liquid chromatograph (HPLC) pump, and the level of accumulated water in the cell is maintained by adjusting the water let down valve (Autoclave Eng., micrometering valve 30VRMM4812). Then,

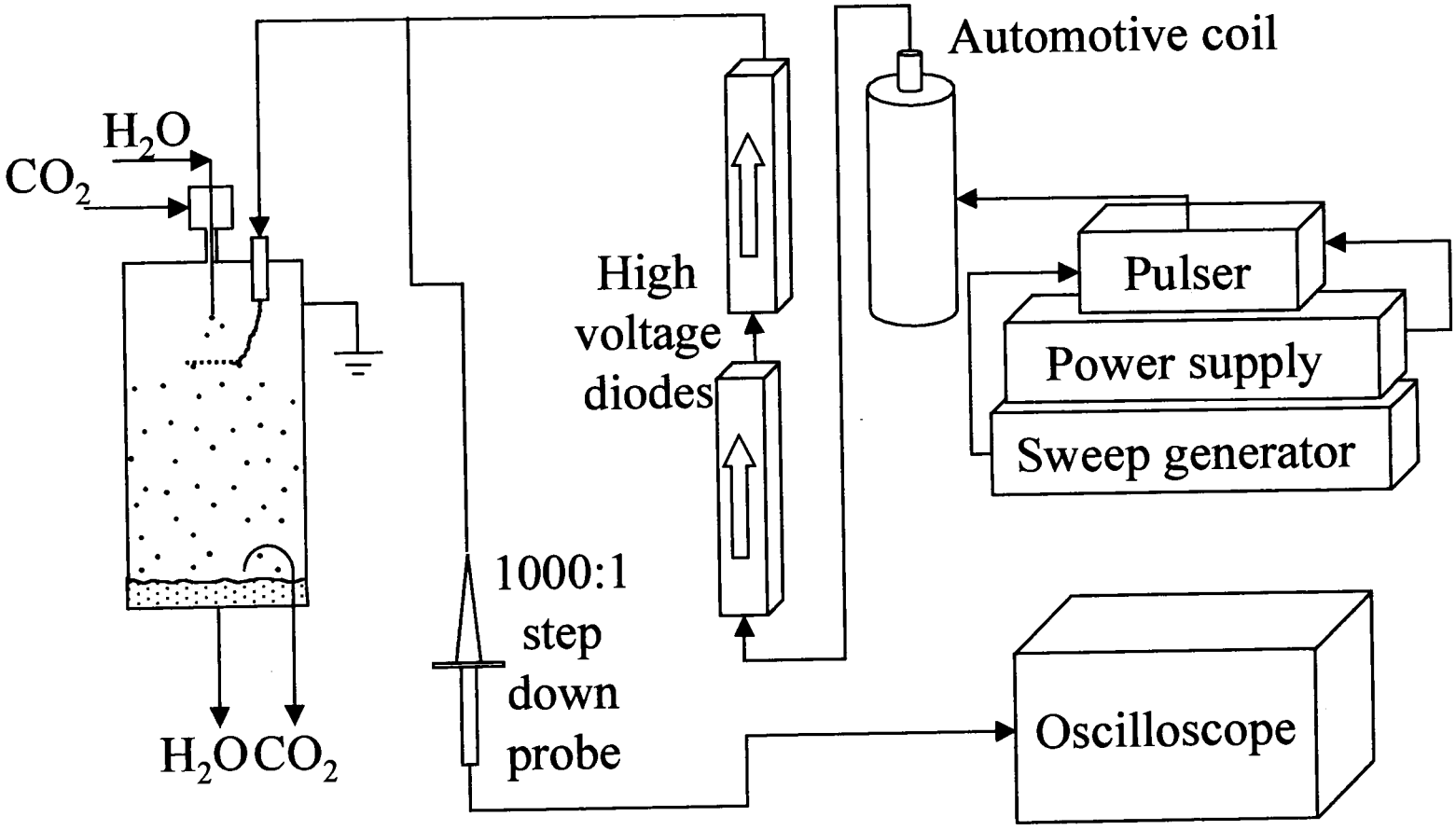


Figure 9. Electrical schematic for providing pulsed high-voltage electric fields.

CO_2 flow is established with a Teledyne Sprague Engineering Model S-86-JN-60 booster compressor, and the pressure of the system is maintained by adjusting the CO_2 let down valve (Autoclave Eng., micrometering valve 30VRMM4812). For the recovery of extract, the CO_2 outlet stream is sent to a series of U-tube condensers (1.27 cm OD and 0.635 cm ID), which contain glass wool for increased effectiveness. A Fisher Porter gas flow meter is used for online flow rate indication, and a wet test meter (Singer American Meter Division Model 802, serial no. P-843) is used for precise volumetric measurement of the CO_2 outlet stream. A bypass valve is located before the condensers so that the system can reach steady state before the condensers are operated. System operation proved to be simple and stable, so that controlled electrodispersion experiments could be performed successfully. The pressure of the system is obtained from an analog gauge supplied by Sprague Engineering Corp. with an accuracy of $\pm$ 0.34 MPa.

An air bath with a high intensity circulating fan has been used to control temperature and minimize electrical hazards. The heating unit is a custom built 300 Watt resistance heater operating through energy supplied by a Variac, variable output voltage transformer. The temperature of the air bath is kept constant through the balance of natural heat loss through the bath walls. Adjustments to the air bath heater are made through the Variac setting. The inlet streams of aqueous solution and carbon dioxide are initially fed through twenty feet each of coiled 0.3175 cm diameter tubing in order to raise and regulate their temperatures before entering the cell as shown in Figure 10. Since the cell consists of an almost solid block of stainless steel with an approximate weight of 34 kg, a separate electrical resistance heater (Omegalux, 505/5-P) and digital

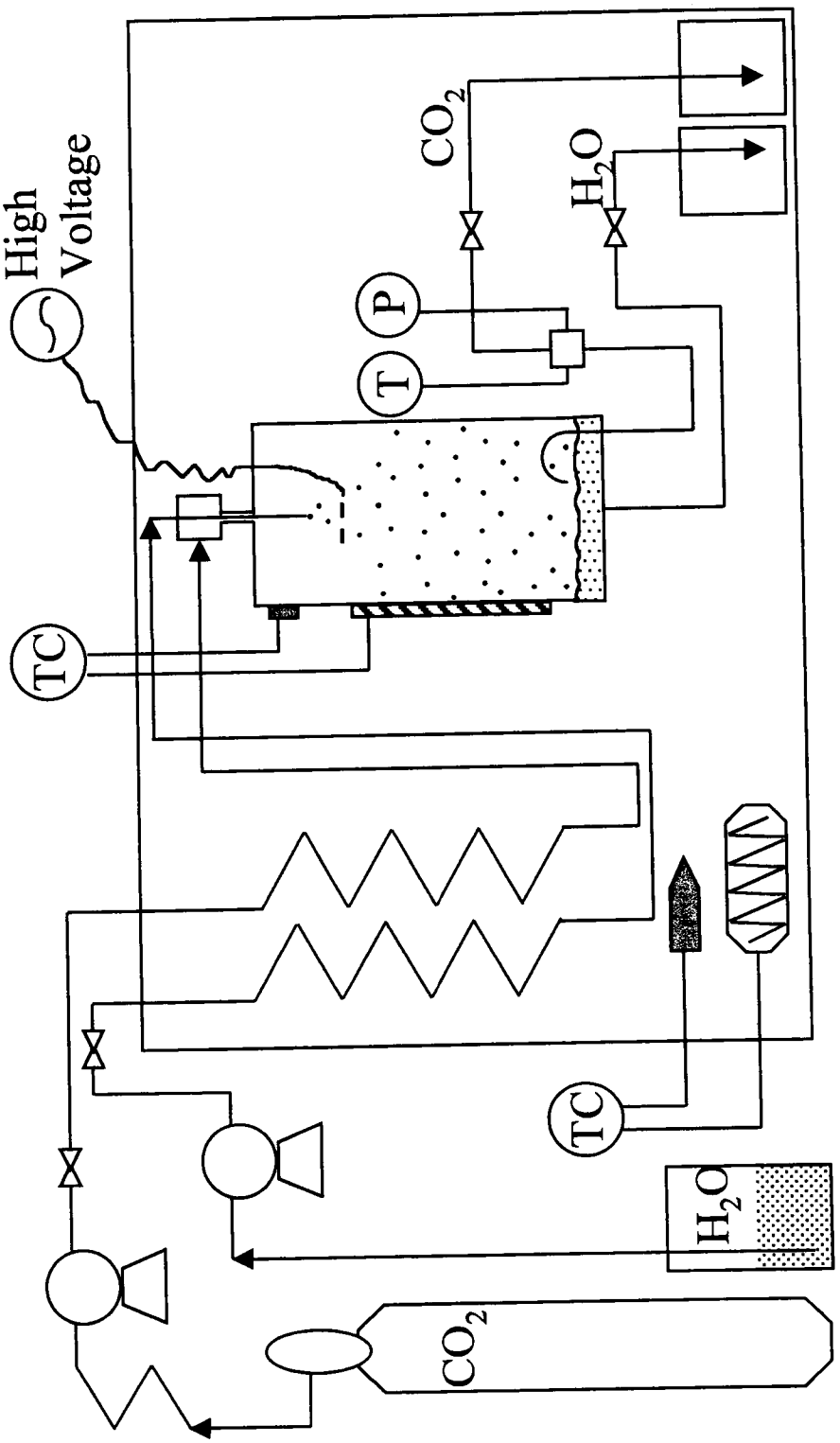


Figure 10. Full schematic of electrodispersion system.

temperature controller (Omega; custom built) have been adapted to the cell itself to assist in temperature control. Temperature control can be maintained to ± 0.1 °C.

The electrodispersion cell is capable of operating with an aqueous medium and CO_2 flowing in either co-current or counter current mode. For the present work, the cell will be described with the contact in co-current flow, which was the mode used in the experiments below. The carbon dioxide enters the chamber through an annular tube surrounding the water capillary. However, it should be noted that although the streams share an inlet port they clearly enter in the chamber with no pre-mixing. There are two outlet ports. A center outlet port is used for draining the accumulated aqueous solution, whereas the carbon dioxide is removed via an extended crooked sheltering tube (see Figure 8). The lengths of both the aqueous inlet tube (capillary) and the carbon dioxide outlet crooked tube can be manipulated for optimization and control of accumulation levels.

3.2 Size Characterization Using Laser Light Scattering

To properly characterize a dispersion, the property of foremost interest is its mean droplet size. Many prior studies involved optically viewing the droplets with a video camera adapted with a magnifying lens (14, 18-27). While this method has its advantages, it is limited in its ability to observe droplets less than 1 micron. The primary reason for this limit is because 1 micron is very close to the wave length of visible light. In addition to this limitation, the focal length shortens with higher magnification. Dispersion in SCO_2 requires a cell capable of maintaining high pressures. In the pressure cell used for the present studies, the closest an optical apparatus could get to the actual dispersion is well out of the focal length for obtaining micron level clarity. Hence, a

different method was required for this work. The method chosen involves laser light scattering.

A 10 mW He-Ne laser and power supply purchased from Melles Griot are used as the light source. This laser is positioned vertically inside the air bath because of physical constraints. The laser is redirected with a 90 degree mirror before entering the cell (see Figure 11). A contractible iris is placed between the mirror and the cell to control the laser beam diameter and to block any scattered light from the mirror. Upon exiting the cell, the high intensity of the beam is blocked with a beam stop. The remaining scattered light is displayed onto a screen designed especially for uniformity. The image of the scattered light on the screen is then acquired through a video camera (Pulnix TM-200) and sent to a DigitalVision RT Mono frame grabber for computer analysis. A two dimensional profile is created based on the observed pattern on the screen. A profile of the image is displayed in the form of light intensity versus distance from the center of the beam. For analysis, these data have to be converted to light intensity versus q, the magnitude of the scattering vector, which is derived from Figure 12 and the following relationships:

$$\bar{q} = \bar{k}_f - \bar{k}_i \qquad (17)$$

where $\bar{q}$ is the scattering vector or difference between $\bar{k}_i$, the incident beam vector and $\bar{k}_f$, the scattered beam vector. The magnitude of the scattering vector is given by

$$q = |\bar{q}| = \frac{4\pi}{\lambda}\sin\left(\frac{\theta}{2}\right) \qquad (18)$$

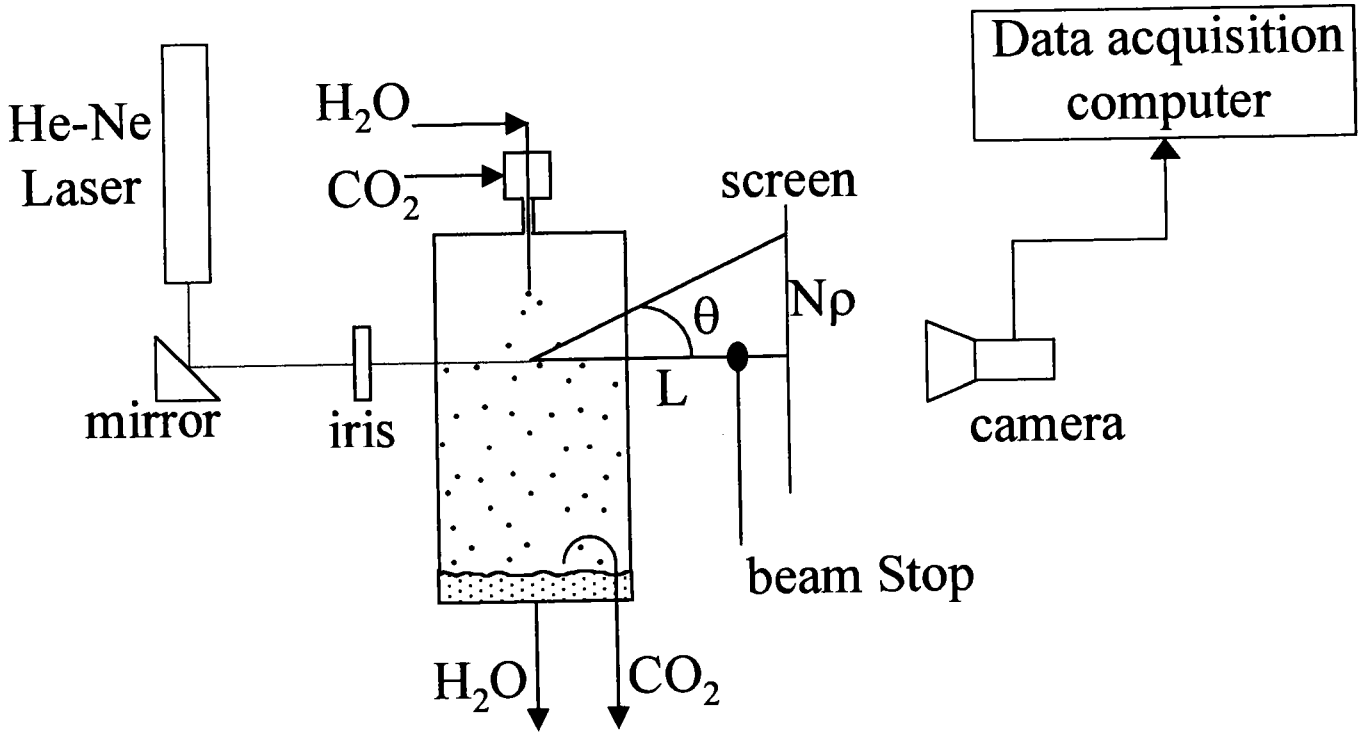
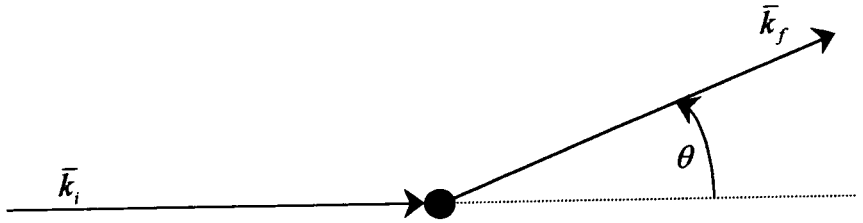


Figure 11. Laser light scattering apparatus for measuring number average droplet diameter.



$$\bar{q} = \bar{k}_f - \bar{k}_i \qquad q = |\bar{q}| = \frac{4\pi}{\lambda}\sin\left(\frac{\theta}{2}\right)$$

Figure 12. Definition of scattering vector, $\bar{q}$.

where λ is the wavelength of the scattered light (He-Ne laser), and θ is the scattering angle. Through a simple geometrical relationship, the distance from the intensity-versus-distance plot can be converted to scattering angle.

$$\theta = \arctan\left(\frac{N\rho}{L}\right) \tag{19}$$

where N is the pixel number, ρ is the distance per number of pixels ($N\rho$ is the actual distance), and L is the length between the scattering sample and the screen.

The pixel density ρ, can be determined very easily by placing a bulls-eye of concentric rings onto the observation screen. Figure 13 contains an illustration. SnapShot software by Viewpoint Software Solutions for use with LabVIEW v.5.0.1 was used to control the frame grabber. Additional code was written to adapt the software, which allows one to draw a line on the image. Then, the light intensity along the line is plotted versus pixel number. Subsequent averaging based on the known distances between the rings allows one to calculate the average pixel density. Once this is done, profiles can easily be converted from intensity-versus-pixel-number to intensity-versus-distance. In addition, substituting equation 19 into 18 and including the distance from the sample to the screen, L, and wavelength, λ, one can now represent the signal in the form of intensity versus q. This is the final form used for most analysis. However, before a correct evaluation of an unknown size can be accomplished, one must calibrate the q-scale. This is a multiplicative correction factor that is based on a sample of known size.

Monodispersed calibration samples are very common in size characterization systems. Calibration based on known samples are often used to correct for any undefined variation between instruments. So before data can be taken from the system, a calibration

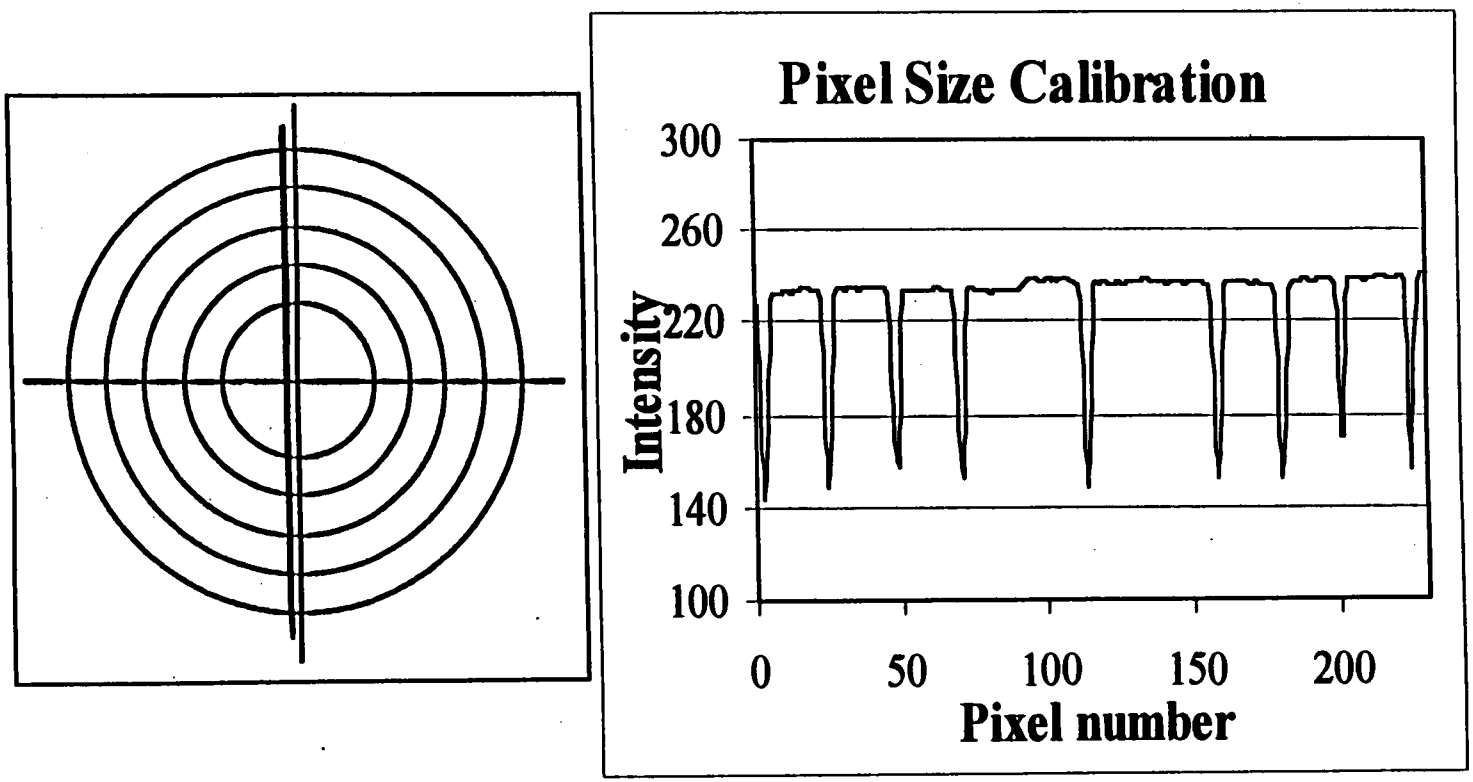


Figure 13. Calibration procedure for determining pixel density ρ.

must be completed. Monodisperse samples have scatter patterns of concentric rings with equal distances between the rings. This distinctive characteristic is the basis for their use as calibration samples. The following equation is used for correcting the q-scale.

$$\frac{2\pi}{q_{max}} = size \qquad (20)$$

where q_{max} is the value of q where intensity is maximum. The formula tells for a particular size sample where the first ring should be located on the q axis. Figure 14 displays the intensity profile for a 10 μm monodispersed sample. The first ring should be located at $q_{max} = 2\pi / 10\mu m = 0.628$ with units μm^{-1} . The q axis is adjusted with a multiplicative constant until the ring is located at 0.628 μm^{-1}. To be certain that calibration is correct, the process should be repeated with a second monodisperse sample. For a 2 μm monodispersed sample (Figure 15), the location of q_{max} is $2\pi / 2\mu m = 3.14$ μm^{-1}. If the correction factors are the same value for each calibration sample (10 μm and 2 μm), then the system is aligned properly, and data can be confidently taken on an unknown sample. The correction factors from the 10 μm and 2 μm calibration standards were determined to be 0.7006 and 0.7016, respectively. The close agreement of the two permitted quantitative use of the technique on unknown droplet sizes.

Once the profile of intensity versus q is established, one must choose a model on which to base the analysis. The model establishes the form of the particle size distribution and the shape of the individual particles (spherical, rod shaped, etc.). The distribution of aqueous droplets created in the electrodispersion cell has been assumed to be Lorentzian, and the droplets are assumed to be spherical. The square Lorentzian model is represented by the following formula:

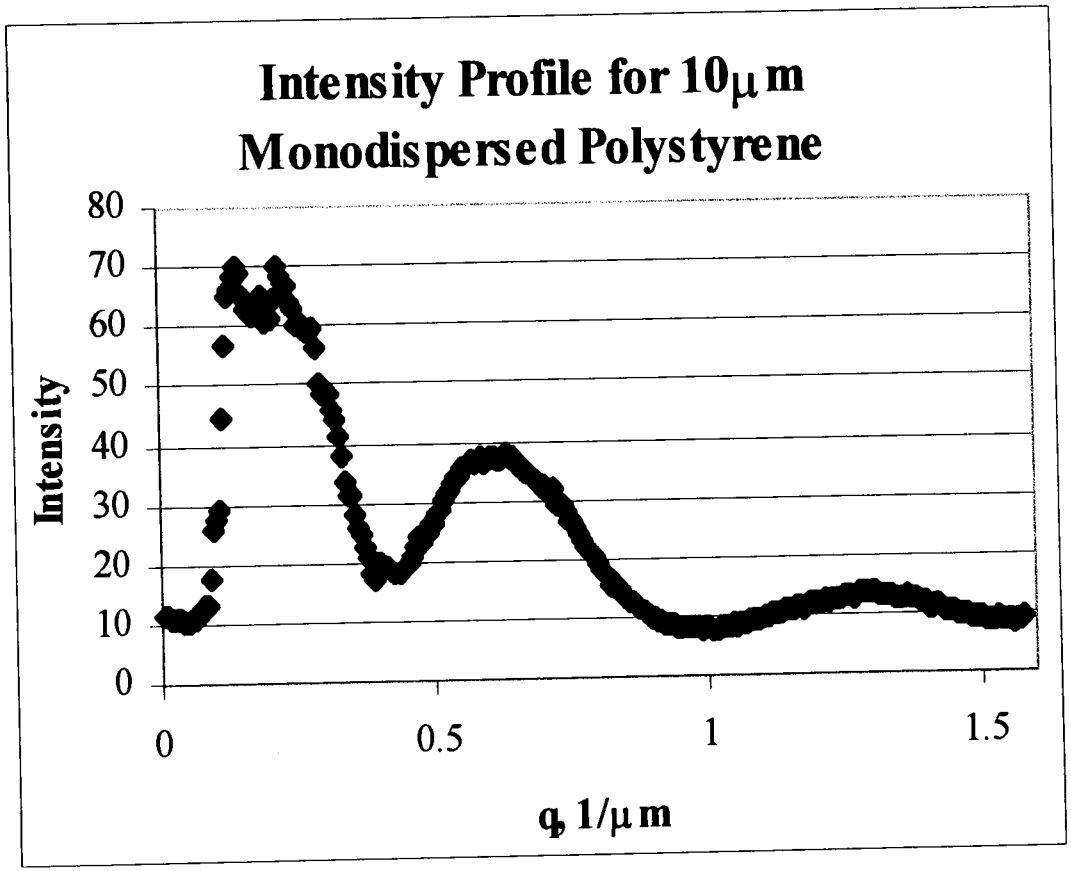


Figure 14. Intensity versus magnitude of scattering vector, q, for a 10 μm monodispersed polystyrene calibration standard.

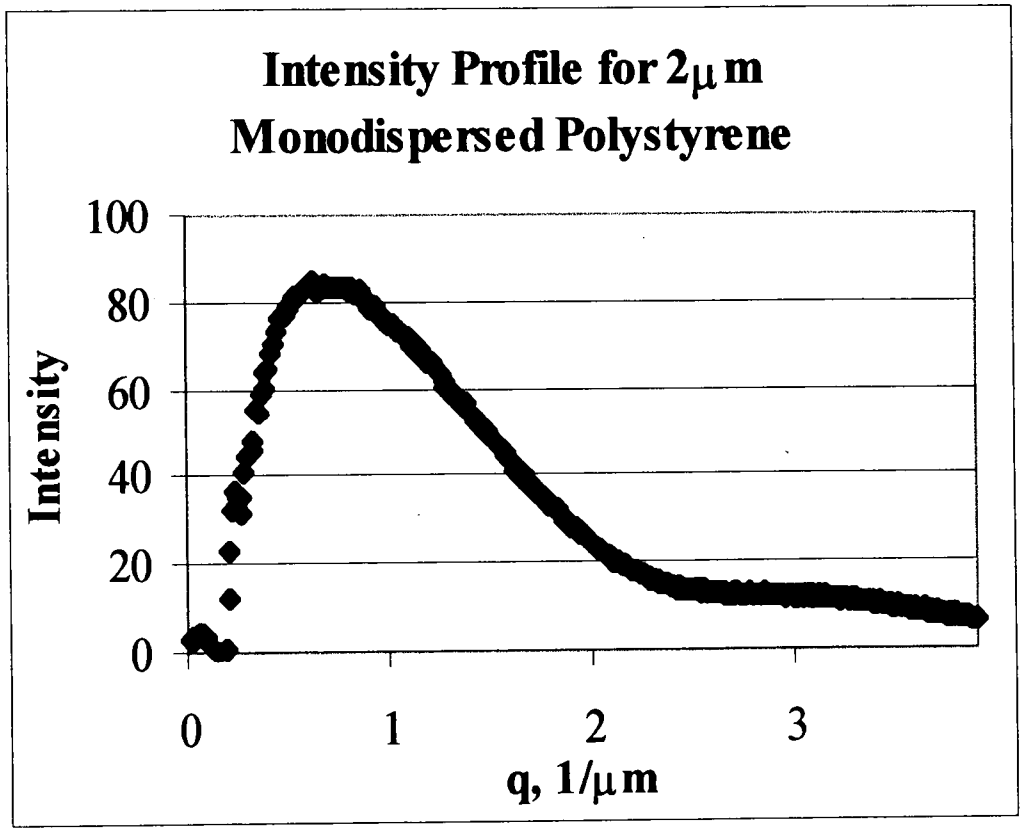


Figure 15. Intensity versus magnitude of scattering vector, q, for a 2 μm monodispersed polystyrene calibration standard.

$$I(q) = \frac{I(0)}{\left(1 + q^2 < d_p >^2\right)^2} \qquad (21)$$

where $I(0)$ represents the intensity at zero q and $<d_p>$ is the number average droplet diameter. For the low q regime, equation 21 can be linearized by plotting reciprocal intensity versus q^2 resulting in the following form:

$$\frac{1}{I(q)} = \frac{2 < d_p >^2}{I(0)} q^2 + \frac{1}{I(0)} \qquad (22)$$

where the intercept is the reciprocal intensity at zero q. The number average droplet diameter, $<d_p>$, can then be calculated from the measured slope and intercept.

Background scattering effects must first be established before unknown scattering experiments can be done. Therefore, an initial intensity profile is taken with the empty cell and is later subtracted from the experimental intensity profile. The difference is the response obtained from only the addition of the sample. This is the primary signal from which all analysis is done. Figure 16 contains an experimental example using the electrodispersion cell (EDC) for the residual intensity (experimental minus background) plotted versus the magnitude of the scattering vector q. Figure 17 illustrates the linearity of the data from Figure 16 when re-plotted in terms of reciprocal intensity, $1/I(q)$, versus q^2. Although considerable noise is evident in the scattering signals shown in Figure 16 and 17, it is also clear from Figure 17 that a reasonable estimate of the slope and intercept can be determined leading to reasonable estimates of $<d_p>$.

3.3 Ethanol Extraction Using SCO_2

Ethanol with a purity of 99.5 vol. % supplied by Wako Pure Chemical Ind. and distilled water was used for preparing all aqueous ethanol solutions. Liquefied CO_2 with

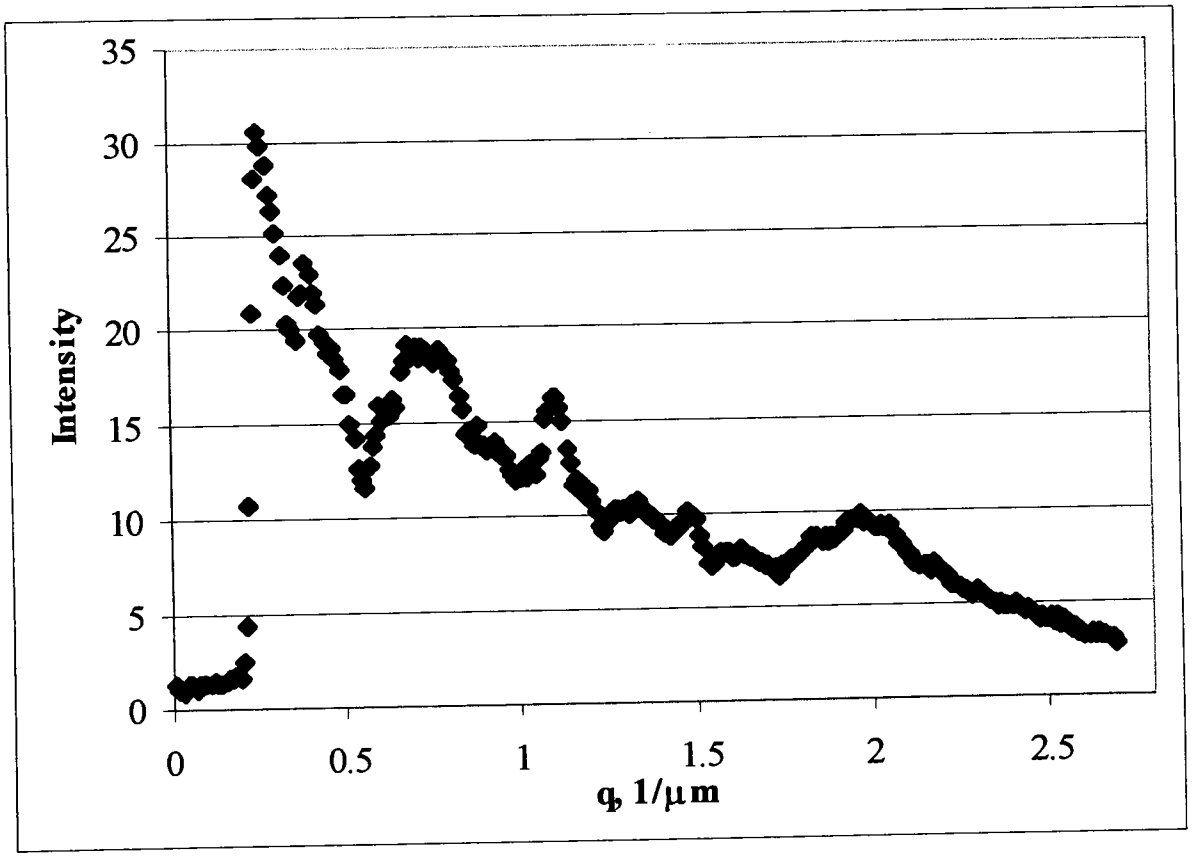


Figure 16. Residual intensity profile of an electrospray in the electrodispersion cell (EDC)

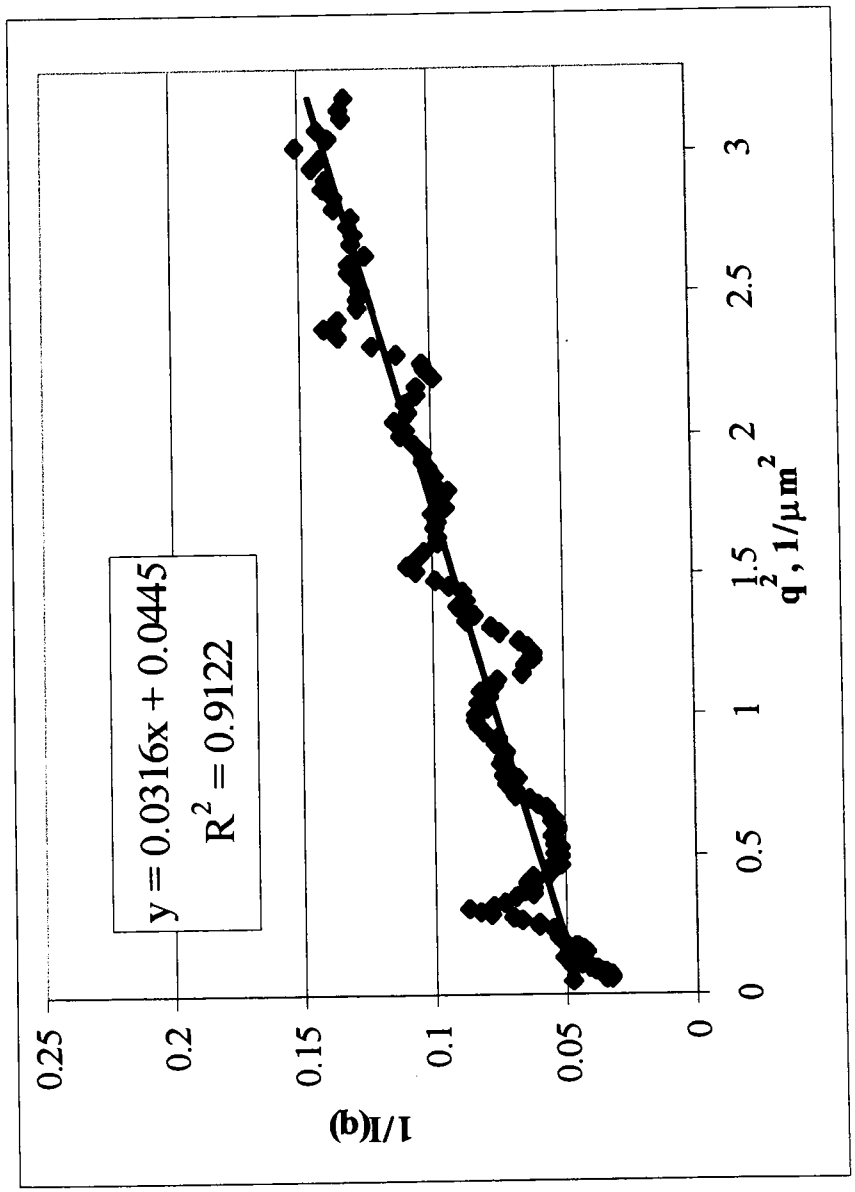


Figure 17. Linearized form of the data from Figure 16.

a purity of 99.99 vol. % was obtained by Holston Gases, Inc. The flow diagram for ethanol extraction from water, the system chosen for examining mass transfer performance, is displayed in Figure 18. The inlet streams are the 10 vol. % aqueous ethanol liquid stream and the stream of SCO_2. The outlet streams are the liquid from which ethanol has been extracted and the SCO_2 which will contain the ethanol and some water. The CO_2 outlet stream containing amounts of extracted ethanol and water is sent outside the air bath to a series of condensers (acetone/dry ice baths). The line from the air bath to the condensers is heating using a heating tape powered by a Variac. This heating tape is to keep the exit line from freezing from the Joule-Thomson effect. Consequently, it also calms the outlet flow, which can get sporadic from small pockets of solid of liquid in the outlet line. Fixed stainless steel plumbing with 0.3175 cm OD travels to and from the condensers. The U-tube condensers are made from heavy wall glass tubing (1.27 cm OD and 0.635 cm ID) and are interconnected with Tygon flexible tubing. The U-tubes are filled with glass wool, to increases contacting with CO_2 gas phase, and are therefore limited to lower CO_2 flow rates. Attempts to condense at too high of flow rates may result in only partial condensing of the outlet stream. The CO_2 outlet stream normally bypasses the condensers until a sample is taken. From the condensers, the outlet stream goes through a gas flow meter for online flow indication (which was previously calibrated with the wet test meter), and then on to a wet test meter for precise total volumetric accumulation. The liquid inlet feed rate is set by a high pressure liquid chromatograph pump. The liquid outlet flow is balanced against the inlet flow so that a constant liquid level is maintained within the cell. At sufficiently low CO_2 flow rates, the CO_2 will be saturated and the mass transfer will be limited by the

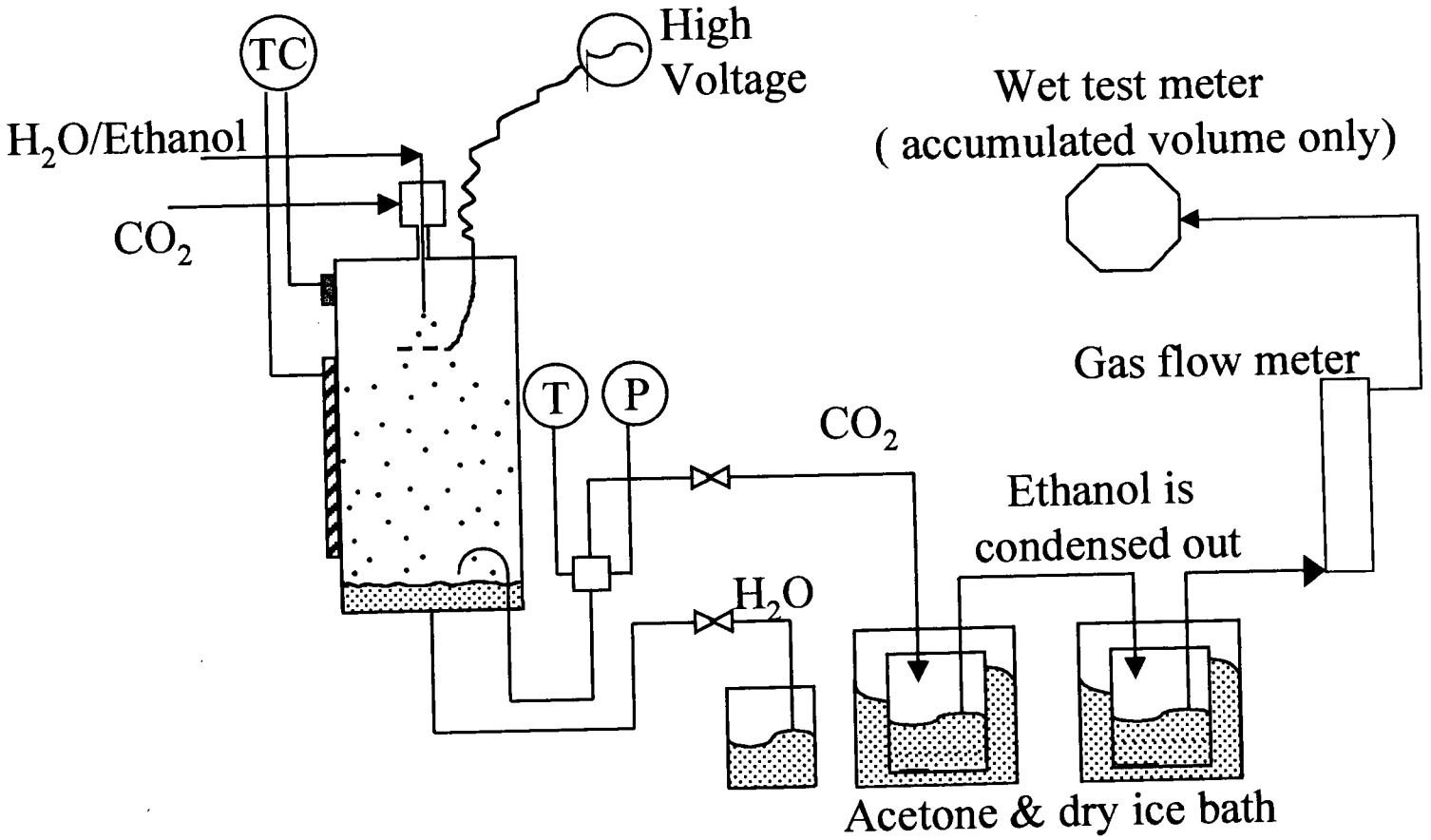


Figure 18. Flow diagram of ethanol extraction/recovery apparatus.

equilibrium conditions. However, at higher flow rates the concentration of ethanol in the CO_2 may be determined by the rate of mass transfer. It is specifically in this regime that variations in interfacial area will have the greatest effect on mass transfer performance; the effects of CO_2 flow rate, system temperature, and system pressure on mass transfer performance can also be studied. The system will be operated at steady state where the temperature, pressure, and liquid level within the cell are constant.

Since the ethanol is much more soluble than water in CO_2, it is believed that condensing ethanol and water from the outlet stream of the CO_2 will provide an extract of increased ethanol concentration. Depending on the phase equilibria, a concentration exceeding the atmospheric azeotropic concentration has been demonstrated to be possible (64-66, 73, 75, 76, 79). The inlet aqueous solution is comprised of 10 vol. % ethanol in distilled water (percentage typical of biological fermentations). The introduction of 10 vol. % of ethanol causes the interfacial tension between the mixture and CO_2 to drop from 26 mN/m (for the case of pure water and CO_2) to 16 mN/m at the operational pressure of 10.34 MPa and 35°C (81).

The procedure for operating the EDC for extraction of ethanol from an aqueous ethanol solution consists of first achieving constant temperature conditions. Depending on the flow rates, adjustments are required for maintaining a consistent temperature for both air bath and CO_2 outlet stream (connecting air bath to condensers). Next, the pressure is brought to the desired level, and maintained with the booster pump and CO_2 outlet metering valve. Then, the liquid inlet stream is turned on at the HPLC pump. However, there is no accumulated liquid in the cell. So, the liquid outlet stream should be closed until the liquid fills the cell to a specified level (aligned with the bottom of the

view windows). Liquid flow rate is not a parameter in this series of experiments. So, the inlet liquid flow rate is always constant. Once the liquid level inside the cell is correct, the level is maintained by adjusting the liquid outlet stream. This is the point of steady state operation where the temperature, pressure, and liquid level are all constant. Before and after each measurement, the U-tubes are weighed to determined condensed mass. After a condensing cycle is complete, the extracted volume is captured. The U-tubes are then placed in a drying oven at 150°C to remove any excess extract, and after cooling are ready to be used again. Subsequent analysis, using a refractometer calibrated with fresh solutions prior to each set of measurements, provides quantitative ethanol content in the CO_2 (extract) and liquid outlet streams. Before each new data point the liquid level is completely drained and refilled while at the new CO_2 gas flow rate. Once steady state operation is achieved, the process repeats itself with a new flow rate, temperature, or pressure.

Before an extraction experiment is carried out, a calculation is made using a ternary Peng-Robinson code (see Section 2.2 for theory and Appendix C.3 for code) with the H_2O-C_2H_5OH-CO_2 system. This code is used to predict the single stage equilibrium conditions for supercritical extraction of ethanol from water.

CHAPTER IV

RESULTS AND DISCUSSION

4.1 Demonstration of Feasibility of Aqueous Dispersions in SCO_2

Before testing under supercritical conditions, preliminary experiments were carried out in air to determine effective electrode geometries and relative distances between electrode and capillary tube to roughly determine operating conditions where electrodispersion existed without arcing. Once the effects of the electric field with various electrode geometries were explored, preliminary experiments were completed in SCO_2 using an available cylindrical glass cell. After limited success, the new pressure cell (described in Section 3.1) was utilized and after certain modifications was used as the electrodispersion cell (EDC) in this work. Testing with the EDC proved to be very fruitful, its greater inside volume allowing for improved electrode design over the limited glass cell design. In addition, the EDC is capable of operation at much higher pressures and temperatures.

An initial feasibility study was demonstrated by testing the two electrode designs of Figure 7 and 8 in air at atmospheric conditions without the use of a pressure cell. It was important to establish working geometries and distances before adding the difficulties of supercritical conditions. Figure 19 shows a series of captured video frames of the dispersion of distilled water into air using the mesh screen electrode design of Figure 8 placed 2.54 cm below a 0.55 mm OD capillary. The end of the capillary has been coned to minimize the surface area in contact with the aqueous droplet and maximize the electric field gradient (see Figure 5b). The 30 frames-per-second capture

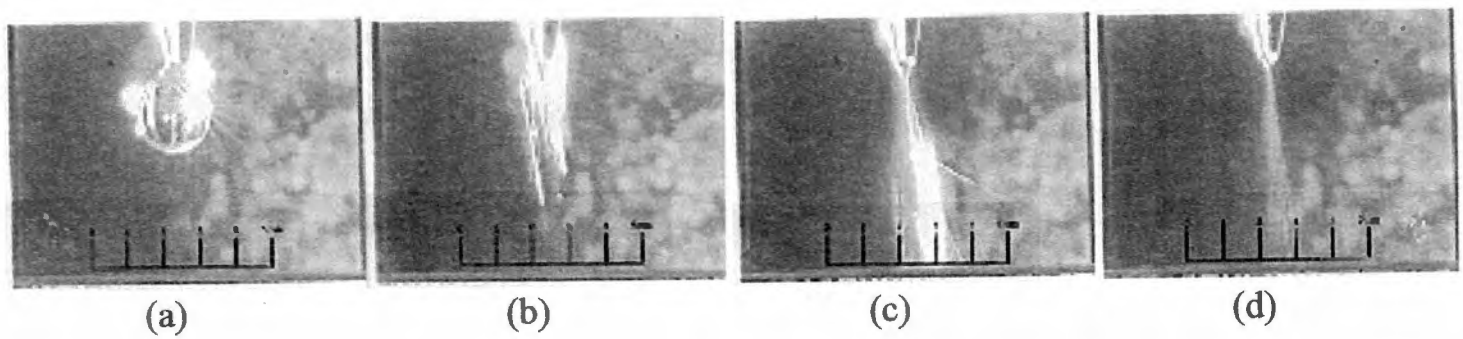


Figure 19. Series of captured video frames of the dispersion of distilled water into air at atmospheric conditions using the mesh screen electrode of Figure 8 placed 2.54 cm below the capillary. Total scale is 5 mm with each division being 1 mm.

rate in Figure 19 is somewhat limiting, because frames two and three seem to be blurred. However, it is clear, that once the high-voltage electric field was energized, the droplet almost instantaneously shatters into a fine spray of water droplets by the fourth frame, 4/30 of a second later.

Next, the dispersion of water in SCO_2 was studied. However, the only view cell available at that time that was capable of withstanding the supercritical conditions of CO_2 was a cylindrical glass tube which had 2.54 cm OD and a very restricting ID of 0.48 cm. This limitation of the ID forced an electrode design that was inappropriate for a continuous fine dispersion of water droplets. The glass tube cell design is shown in Figure 6. The inlet capillary was electrically grounded while the high voltage was delivered through a copper band (0.75 cm in width) wrapped around the OD of the tube. Upon energizing the field there was an instantaneous spray of the water droplets directly from the capillary to the glass surface. The electric field caused the droplets to immediately flow toward and adhere to the wall, resulting in almost immediate coalescence, which made it difficult to determine the fineness of the dispersion. Although it was difficult, limited success was achieved with this setup. For certain conditions a fine mist was observed on the glass surface just above the capillary exit. This observation indicated that a fine dispersion may have been created at least partially before the droplets reached the glass wall. The experimental conditions that proved favorable were the following: pulsed dc was favored over constant dc at a frequency of 50 Hz using the maximum allowable field capability (–30kV/cm) before breakdown was observed. The cell was operated in cocurrent flow while regulating the pressure at 8.27 MPa with a CO_2 high pressure liquid chromatograph (HPLC) pump and CO_2 letdown

valve. The temperature of the system was held at a constant 40°C, while the flow rate of water was 0.6 ml/min.

All studies involving dispersions in SCO_2 from here on were completed using the EDC apparatus. Figure 20 displays the dependence on a small electric field of the droplet size just prior to detachment. The aqueous medium is 10 vol. % ethanol in distilled water, and the surrounding environment is SCO_2 at a pressure of 7.58 MPa and 40 °C. This series of frames closely resembles the previous work of Borzabadi and Bailey (27). Figure 21 was taken from their work (27), and illustrates glycerol drops just about to detach for different voltages between a capillary tube and a plate. In their system the plate is grounded, and the droplets are formed in air at ambient conditions.

Figure 22 demonstrates attainment of the first objective of demonstrating feasibility for electrospraying a fine dispersion of aqueous droplets into SCO_2. The series is a frame by frame (30 frames/sec rate) description of the effect of increasing the voltage beyond that of merely reducing the droplet size to that of the desired effect of producing a fine dispersion. Figure 22a shows two droplets; one is located on the tip of the capillary and the other has just fallen from it. The fallen droplet can only be seen as a streak stretching vertically below the capillary. Figure 22b displays the effect of a stepwise increase in field strength. Here the single droplet that was previously on the tip is now represented as a streak, and the next droplet is forming the desired electrospray which provides the fine dispersion. The third frame of Figure 22c shows that no longer is there a streak because all the droplets are of a small size. The conditions that allow for this observation are the following: A flow rate of 0.9 ml/min of aqueous solution which was balanced by a steady outlet flow of the accumulated liquid. The pressure of 7.58 MPa

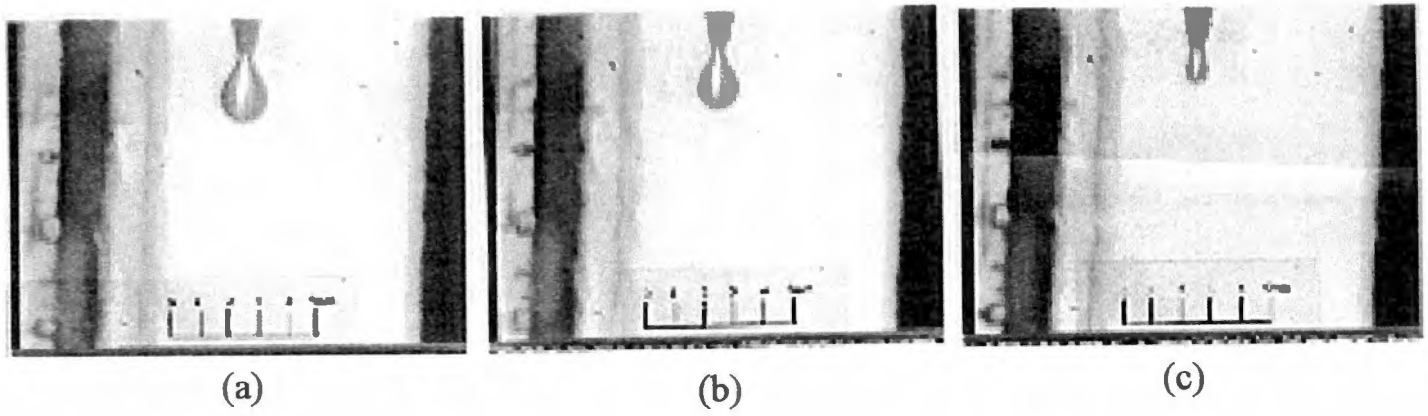


Figure 20. Dependence of droplet size (just prior to detachment) on electric field. Aqueous medium is 10 vol. % ethanol in distilled water. Supercritical CO_2 is at 40°C and 7.58 MPa. (a) -0.75 kV/cm, (b) -1 kV/cm, (c) -1.5 kV/cm. Total scale is 5 mm with each division being 1 mm.

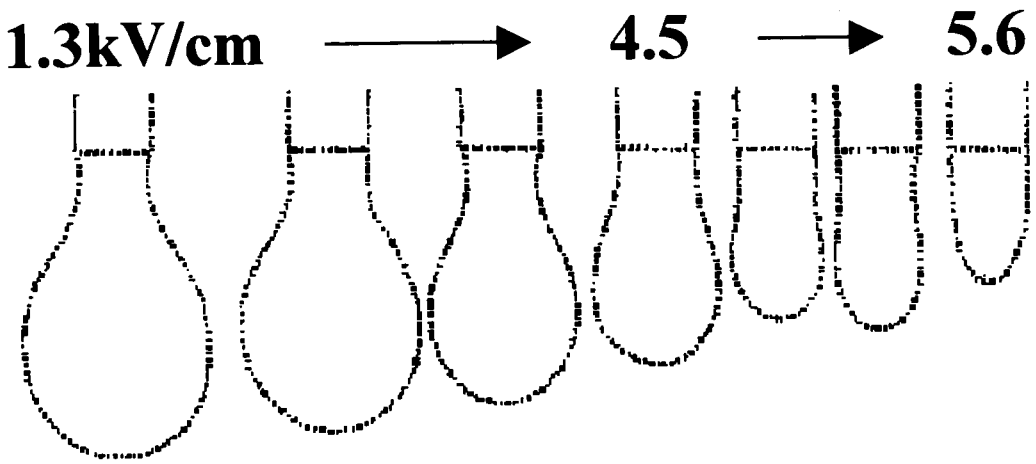


Figure 21. This figure was taken directly from Borzabadi and Bailey (27). Data represents experimental profiles of electrified drops just prior to dripping. Electrode design is similar to Figure 8 except that a plate was used instead of a screen. In addition, the plate was grounded while the capillary carried the high-voltage. The liquid was glycerol into an environment of air at ambient conditions. The distance between capillary and plate was 7.7 mm.

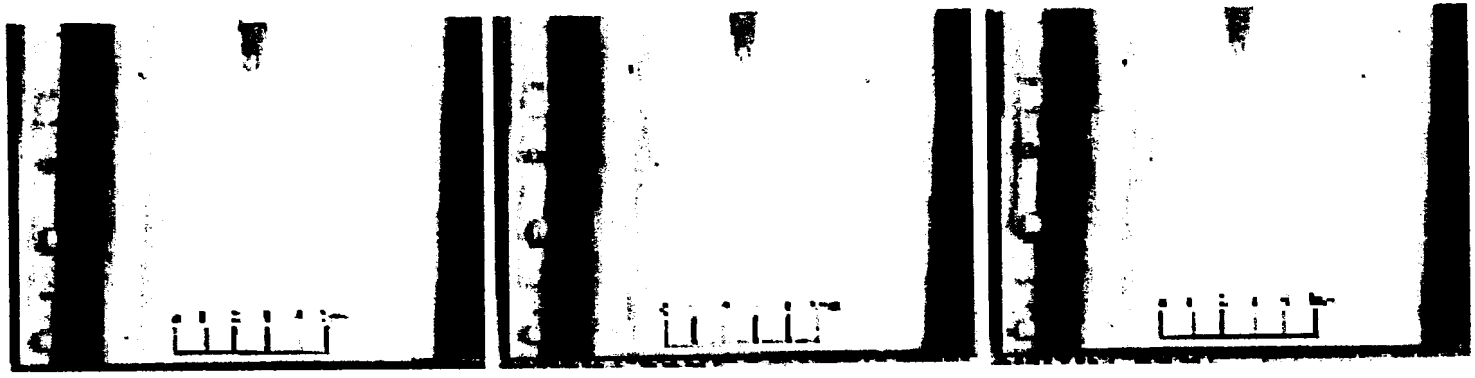
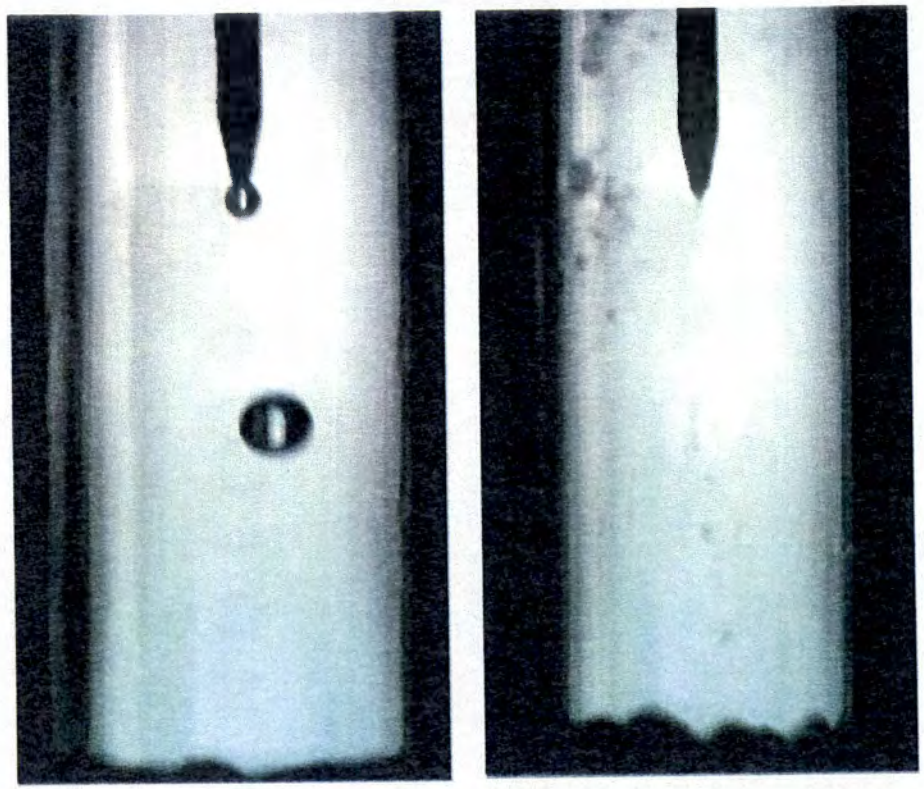


Figure 22. Frame by frame depiction of the effect of increasing the electric field until a fine dispersion is achieved. Figure c depicts a fine dispersion of aqueous droplets (10 vol. % ethanol in distilled water) into supercritical CO_2 at 40°C and 7.58 MPa. The field strength had been increased step-wise from -1.5 kV/cm to -2 kV/cm. Total scale is 5 mm with each division being 1 mm.

was maintained by the balance of CO_2 flow in and out. The temperature was held at a constant 40°C. The frequency of the applied voltage was 400 Hz at a maximum peak of –5.2 kV (2.0 kV/cm). Figure 23 contains a better visual depiction of inside the EDC. Figure 23a displays the typical size of aqueous droplet without a field, and when the field is engaged (Figure 23b) the droplets almost completely disappear. The mode of electrospray found in Figure 23 appears to be that of the cone-jet mode (39). There exists a distinct cone emanating from the tip of the capillary, which defines the characteristic cone-jet mode.

The series of frames in Figure 24 display the effect of increasing the aqueous flow rate while maintaining a constant voltage of –8.8 kV (3.5 kV/cm). All other parameters except flow rate are consistent with those in Figure 22 including frequency. Figure 24a uses a flow rate of 4.0 ml/min. Figure 24b has an increased flow rate of 5.0 ml/min, while the last, Figure 24c, shows the effect of using 6.0 ml/min. The dispersion effect remains in all three cases. The only noticeable difference between the three is the extension of the initial stream length. This shows that the process is feasible at even higher flow rates as well. The limit of the dispersion is balanced by the flow of the aqueous solution and the electric field which shatters it. The mode of electrospray for this figure appears to be that of the ramified jet mode (39). This mode of electrospray is characterized by the liquid thread widening at the apex and its edges emitting many fine jets.

Feasibility of dispersing aqueous fluid in SCO_2 was achieved for both pure water and a 10 vol. % aqueous ethanol solution for the condition of 10.34 MPa and 40°C. Figure 23 displays a clear demonstration of this objective. Depending on the conditions,



(a) (b)

Figure 23. Electrodispersion cell close-up for (a) field off and (b) field on conditions. Specific conditions are 7.58 MPa, 40°C, 2.0 kV/cm at 400 Hz, and 1.0 ml/min liquid flow rate.

optical examinations support the mean diameter of the droplets as ranging from more than a hundred microns to less than one micron, with corresponding production frequencies ranging from a few thousand to a few tens of millions of droplets per second.

4.2 Effect of Field Strength, Pulse Frequency, and Flow Rate on the Mean Droplet Size of Pure Water in SCO_2

A laser light scattering system was adapted to the EDC apparatus for the measurement of number average droplet diameter ($<d_p>$). This system was calibrated with both 10 μm and 2 μm calibration standards before collecting experimental data. The method of analysis has already been presented (see Section 3.2). Various relationships with $<d_p>$ were explored including the effect of field strength, pulse frequency, and aqueous flow rate. The results in the following paragraphs were obtained with pure distilled water and SCO_2 at 10.34 MPa and 40°C.

Figure 25 shows the relationship of field strength (2-5 kV/cm) on $<d_p>$ for three pulse frequencies (100, 1,000, and 10,000 Hz) while holding the aqueous flow rate constant at 3 ml/min. Here $<d_p>$ increases with increasing field strength. Specifically an increase from 0.9 μm at 2 kV/cm to 1.75 μm at 5 kV/cm was observed. This result appears contradictory to prior work (32) in the cone-jet mode of electrospray, but it cannot be said for certain that these results and those in the following paragraphs were obtained in the cone-jet mode. Prior work has established that increasing the field strength while in the cone-jet mode causes a decrease in droplet size. However, an electrospray which operates beyond the onset of cone-jet mode (at higher field strength, flow rate or both) becomes unstable and polydisperse. Polydispersity is believed to cause the observed increase in $<d_p>$ with increasing field strength seen in Figure 25. The

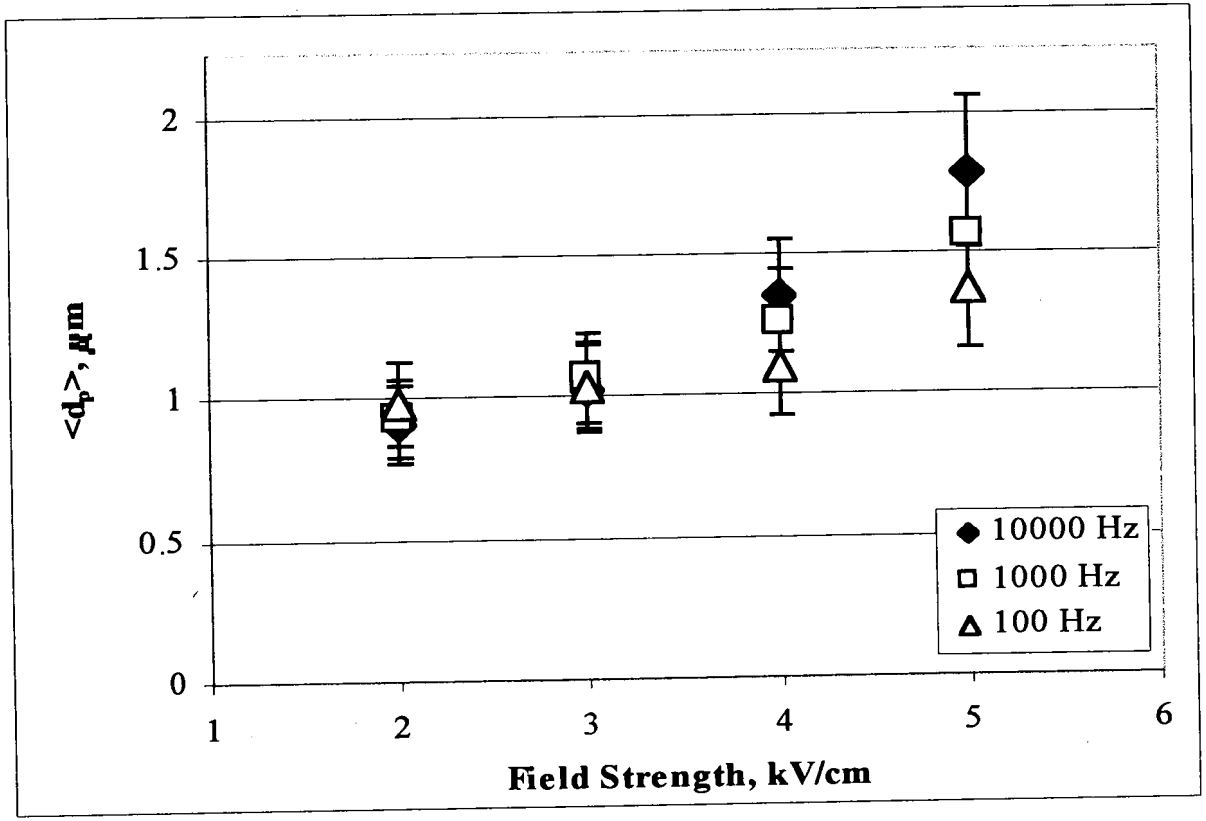


Figure 25. Effect of field strength on $<d_p>$ with distilled H_2O at flow rate of 3 ml/min, 10.34 MPa, and 40°C.

results in Figure 25 also show the effect of increasing the pulse frequency. All three pulse frequencies give similar $<d_p>$ at lower field strength with higher frequency giving a larger $<d_p>$ at higher field strength. The error bars included in Figure 25 and subsequent figures relating to mean droplet size were calculated based on the standard deviation from the least squares linear regression of the scattering data using

$$S.D. = \sqrt{\frac{\sum_i (y_i - \bar{y})^2}{(n-1)}} \qquad (23)$$

where $\bar{y}$ is the mean average of the reciprocal intensity similar to the description presented in Figure 17. An error propagation analysis was completed and the average deviation is represented as error bars.

Figure 26 displays the effect of increasing pulse frequency (1-10,000 Hz) on $<d_p>$ for two aqueous flow rates (4 and 7 ml/min) while holding the field strength constant at 4 kV/cm. Here it is shown, that at the lower pulse frequency of 10 Hz, $<d_p>$ is smaller than 1.0 μm for both flow rates. However, as the pulse frequency is increased to 10,000 Hz, the $<d_p>$ for both flow rates increase. The 7 ml/min data set shows a larger dependence with an increase to 2.3 μm for increased pulse frequency, while the 4 ml/min data set has a more modest dependence with a $<d_p>$ of 1.5 μm at the highest pulse frequency.

Figure 27 gives the effect of increasing flow rate (1-9 ml/min) on $<d_p>$ for two pulse frequencies (10 and 1,000 Hz) while holding the field strength constant at 4 kV/cm. At lower aqueous flow rates (1-6 ml/min), $<d_p>$ appears fairly constant for these conditions, but increases once the flow increases. For all flow rates the 10 Hz data set

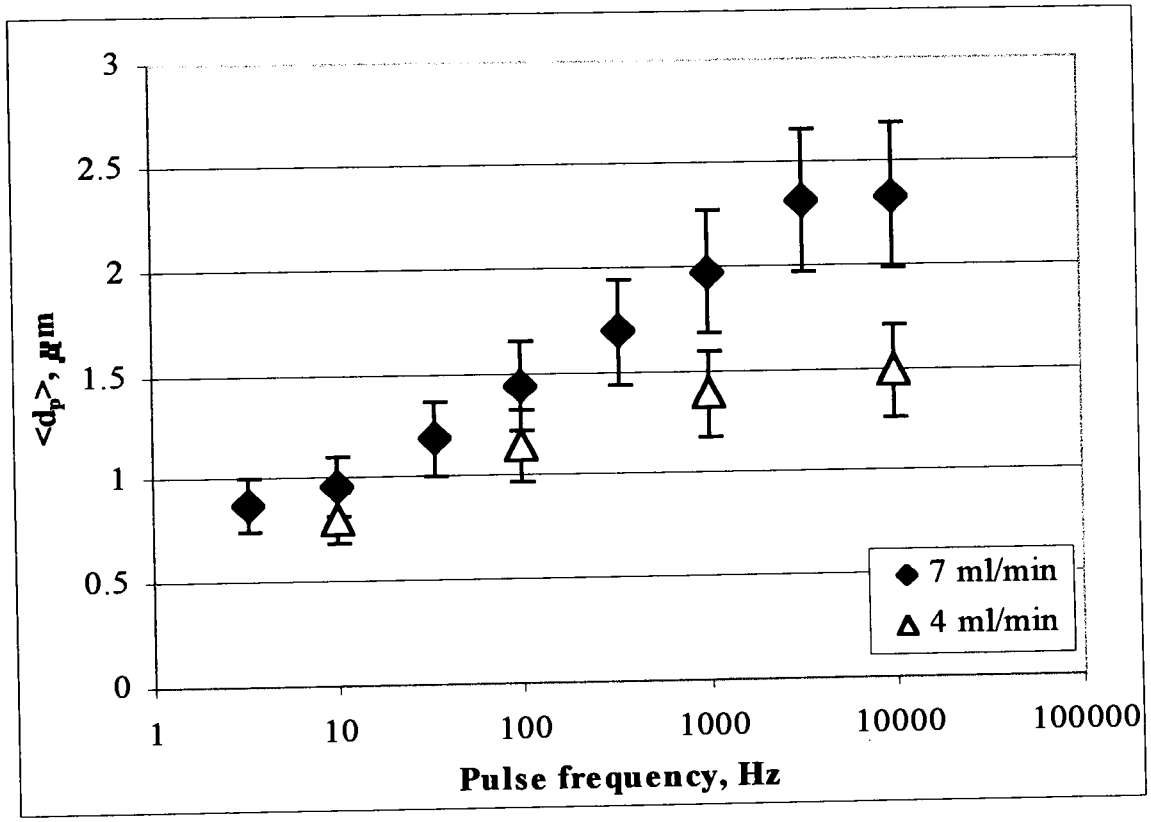


Figure 26. Effect of pulse frequency on $<d_p>$ at 4kV/cm, 10.34 MPa, and 40°C for distilled H_2O.

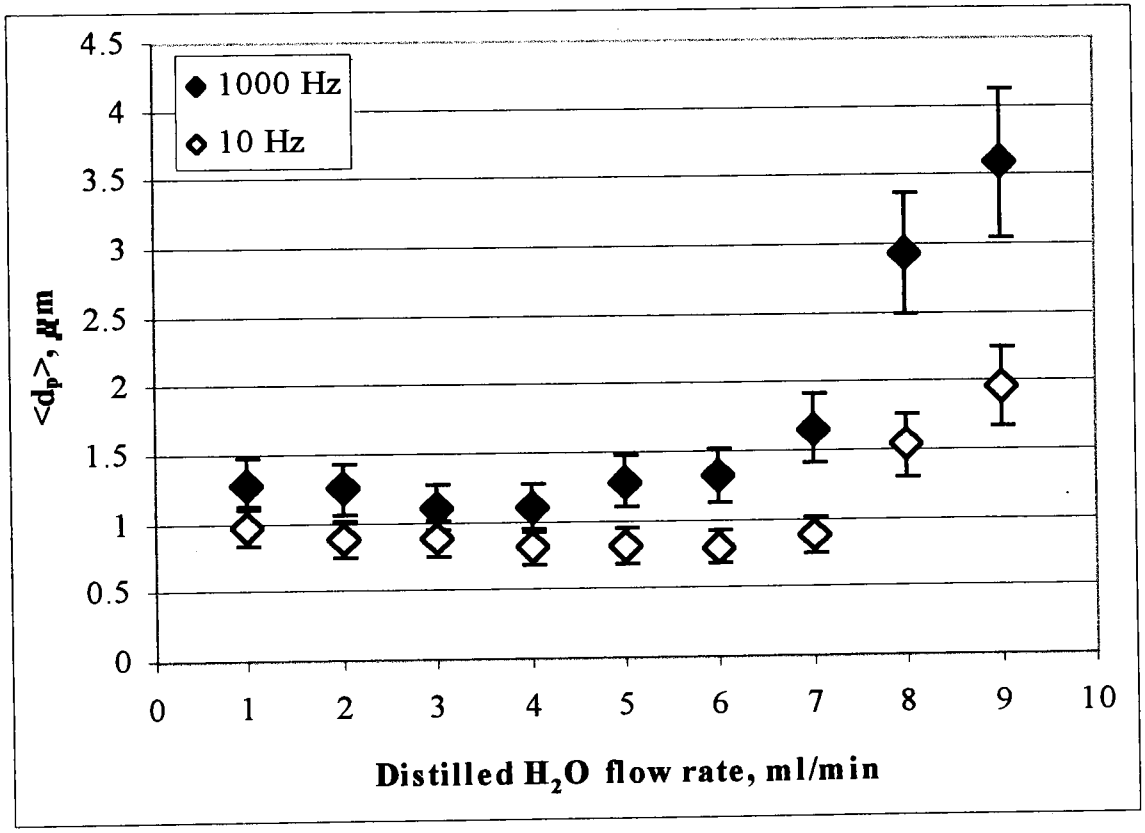


Figure 27. Effect of flow rate on <d_p> at 4kV/cm, 10.34 MPa, and 40°C.

gives smaller $<d_p>$ than the 1,000 Hz data set. The 10 Hz data set shows $<d_p>$ less than 1.0 μm until a flow rate of 8 ml/min where it climbs to 1.5 μm with a maximum of 2 μm at 9 ml/min. For the 1,000 Hz data set the $<d_p>$ stays below 1.5 μm until the flow rate is raised to 8 ml/min where it increases to almost 3 μm with a maximum of 3.5 μm at 9 ml/min. These findings of increased $<d_p>$ with increasing flow rate are common and have been observed by many researchers (32-38). A tabulation of all data contained in Figures 25-27 is located in Appendix D.

Indeed there are interesting effects of field strength, pulse frequency, and flow rate on $<d_p>$. It would be valuable to confirm in future work whether the electrospray is in the cone-jet mode. Beyond the cone-jet mode increases in instability and polydispersity may confuse the effects of field strength and flow rate. In all cases, whether increasing flow rate or field strength, lowering the pulse frequency appeared to give smaller $<d_p>$. The effect of increasing $<d_p>$ with increasing flow rate demonstrated by other researchers was confirmed even though evidence suggests that much of the current data may be beyond the cone-jet mode of electrospray.

4.3 Effect of Liquid Flow Rate, Temperature, and Pressure on the Mean Droplet Size of 10 vol. % Aqueous Ethanol in SCO_2

Understanding relationships which determine mean droplet size, $<d_p>$, of electrosprays in SCO_2 will allow the optimization and ultimately the improvement of their use. The effects of varying flow and field characteristics on electrodispersion of pure water in SCO_2 were presented in Section 4.2 for a constant state condition of 10.34 MPa and 40°C. In this section, the effects of liquid flow rate and various temperature and pressure conditions on $<d_p>$ are presented with the focus on electrospraying 10 vol.

% aqueous ethanol in SCO_2. Temperatures of 35, 47, and 65°C with pressures of 10.34, 13.79, and 17.24 MPa at each temperature were examined with a constant field strength of 2.7kV/cm and constant pulse frequency of 133 Hz. An attempt to understand the data is presented with a focus on surface tension, one of the several physical properties that change with state conditions.

Figure 28 displays the effect of temperature (35, 47, and 65°C) on $<d_p>$ at a constant pressure of 10.34 MPa. The 35°C data set has $<d_p>$ that are almost all less than 1.0 μm at all liquid flow rates from 2-12 ml/min. The 47°C data set has larger $<d_p>$ than the 35°C set ranging from 1.3 μm to almost 2.0 μm at the highest flow rate. The 65°C data set exhibits $<d_p>$ even larger than the other two with a very slight dependence on flow rate ranging from 2.2 μm at 2 ml/min to 2.5 μm at 12 ml/min. In this figure increasing the temperature causes an increase in $<d_p>$ over all flow rates at 10.34 MPa.

Figure 29 displays the effect of temperature (35, 47, and 65°C) on $<d_p>$ at a constant pressure of 13.79 MPa. All the temperatures at this pressure exhibit similar $<d_p>$ especially at the lower flow rates from 2-6 ml/min. At higher flow rates (8-12 ml/min), the 35°C data set has the largest $<d_p>$ at 2.4 μm for 12 ml/min flow. The 65°C data set has the smallest $<d_p>$ at 1.6 μm for 12 ml/min flow. There exists a slight dependence (increase of $<d_p>$) with flow rate for all three temperatures at this pressure.

Figure 30 displays the effect of temperature (35, 47, and 65°C) on $<d_p>$ at a constant pressure of 17.24 MPa. This plot shows the reverse effect of that observed for the 10.34 MPa data set. The lowest temperature (35°C) now produces the largest $<d_p>$, while the highest temperature (65°C) gives the lowest $<d_p>$. The 65°C data set shows a

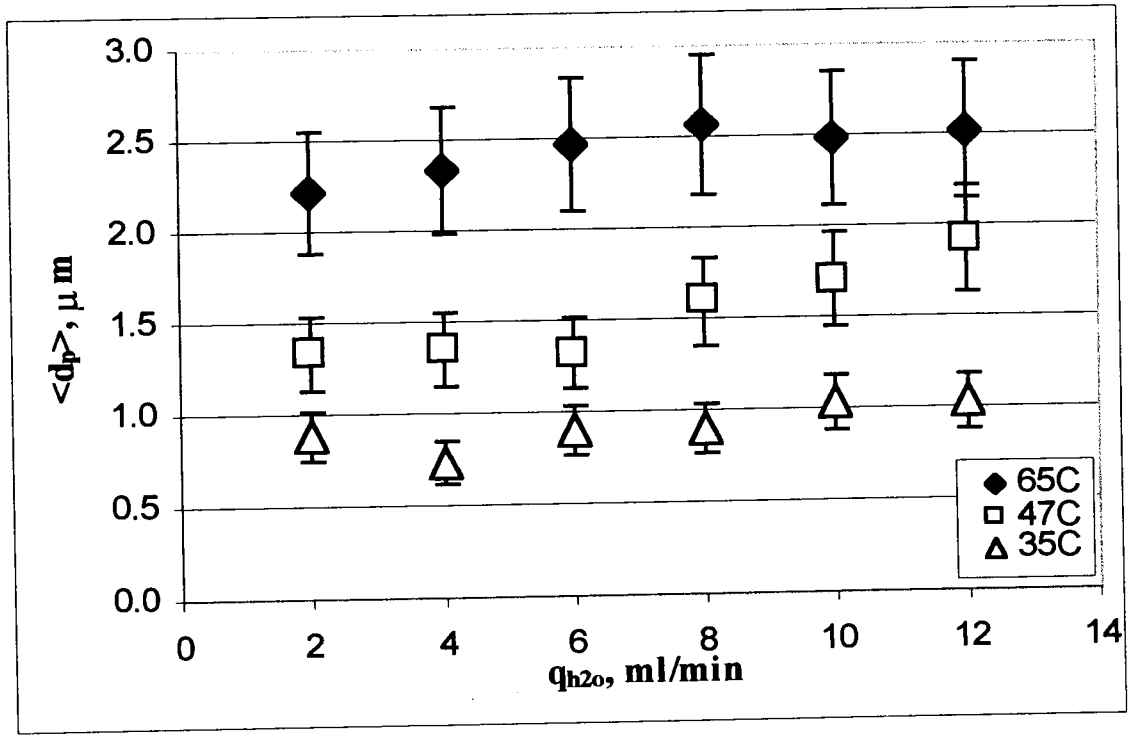


Figure 28. Droplet Size Characterization at 2.7 kV/cm, 133 Hz, and 10.34 MPa.

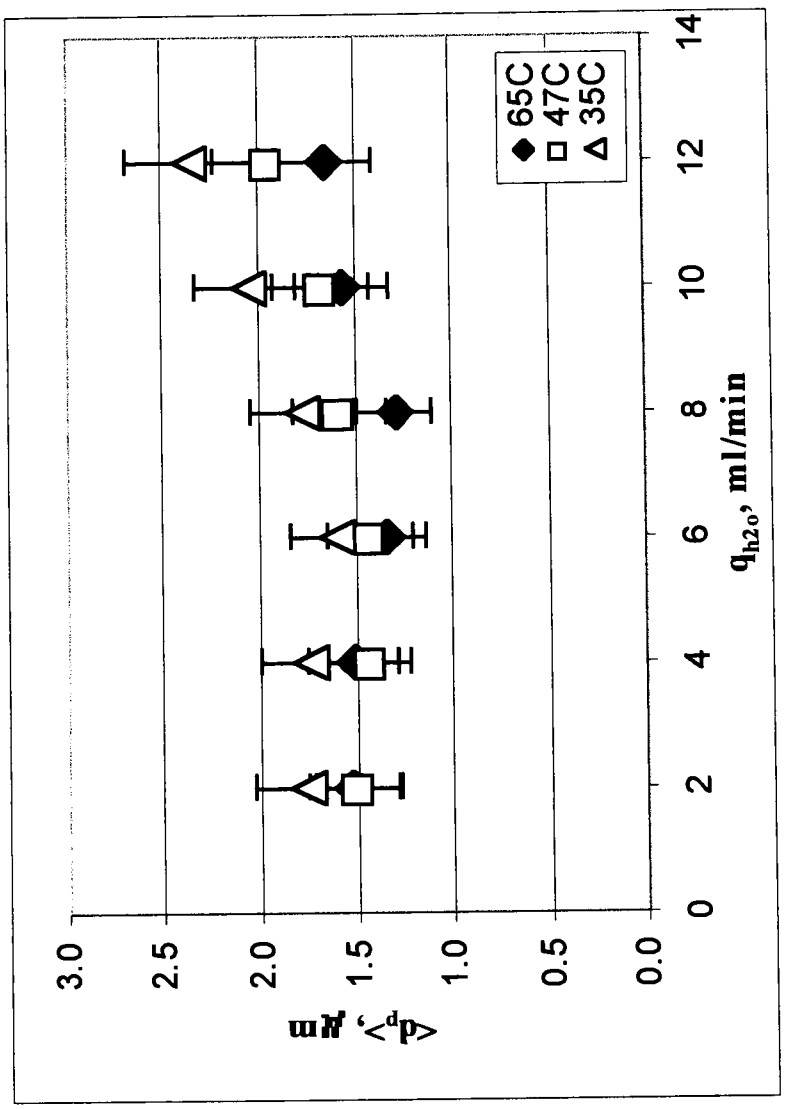


Figure 29. Droplet Size Characterization at 2.7 kV/cm, 133 Hz, and 13.79 MPa.

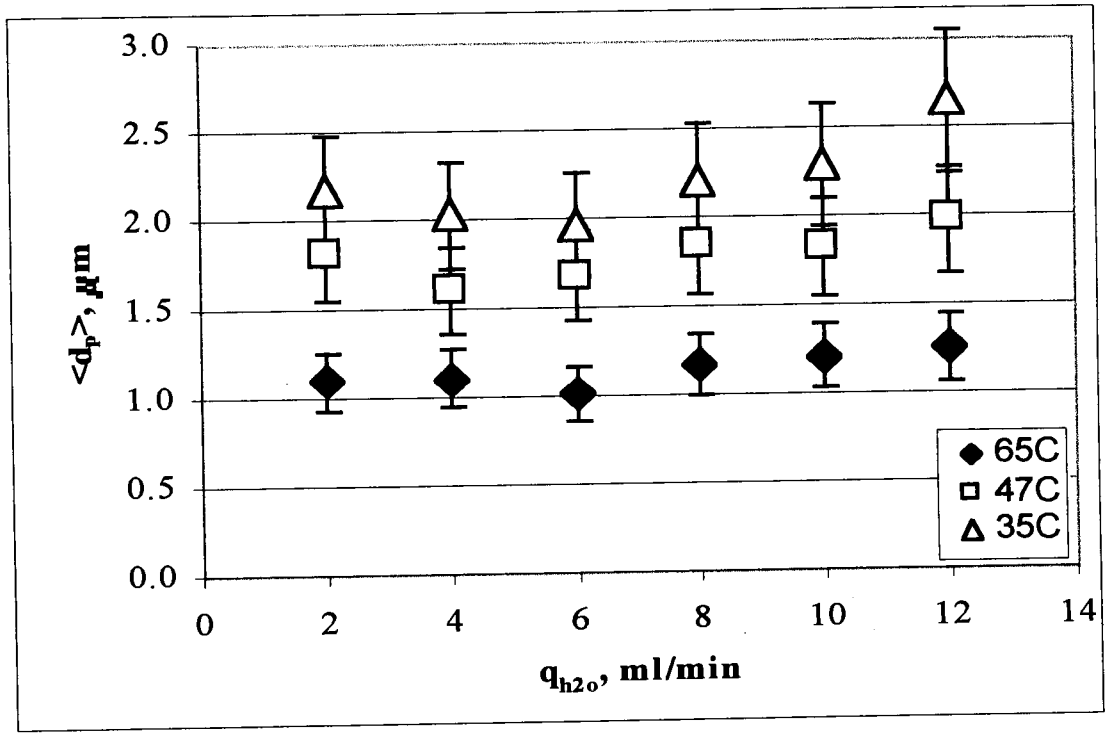


Figure 30. Droplet Size Characterization at 2.7 kV/cm, 133 Hz, and 17.24 MPa.

very weak dependence of $<d_p>$ on liquid flow rate ranging from 1.1 μm at 2 ml/min to 1.3 μm at 12 ml/min. While the 35°C data set exhibits $<d_p>$ that range from a low of 2 μm at 6 ml/min to as high as 2.6 μm at 12 ml/min. The 47°C data set has $<d_p>$ which are all intermediate between the 35°C and 65°C data.

Figure 31 displays the effect of pressure (10.34, 13.79, and 17.24 MPa) on $<d_p>$ at a constant temperature of 35°C. Here the same data are re-plotted to elucidate the effect of raising the pressure for a constant temperature of 35°C. The lowest value of $<d_p>$ at the pressure (10.34 MPa) are almost all less than 1.0 μm at all flow rates (2-12 ml/min), while the higher pressure data sets both show much larger $<d_p>$. In fact, the highest pressure data set (17.24 MPa) contain $<d_p>$ as high as 2.6 μm at 12 ml/min. A trend of increasing $<d_p>$ with increasing pressure for this temperature is observed.

Figure 32 displays the effect of pressure (10.34, 13.79, and 17.24 MPa) on $<d_p>$ at a constant temperature of 47°C. The data are all very similar with more overlap at the higher flow rates than the lower flow rates. From 2 ml/min to 8 ml/min a noticeable trend exist with increasing $<d_p>$ for increases of pressure. However, at higher flow rates (10-12 ml/min) the data almost all fall along the same $<d_p>$. The 17.24 MPa data set has a noticeably higher $<d_p>$ of 1.8 μm at 2 ml/min compared with $<d_p>$ at 10.34 and 13.24 MPa, which are 1.3 and 1.5 μm respectively.

Figure 33 displays the effect of pressure (10.34, 13.79, and 17.24 MPa) on $<d_p>$ at a constant temperature of 65°C. Again the reversal of trends is seen for this temperature as was seen for the highest pressure plot in Figure 30. Figure 33 shows that at 2500 and 13.79 MPa $<d_p>$ is considerably smaller than at the lowest pressure of 10.34

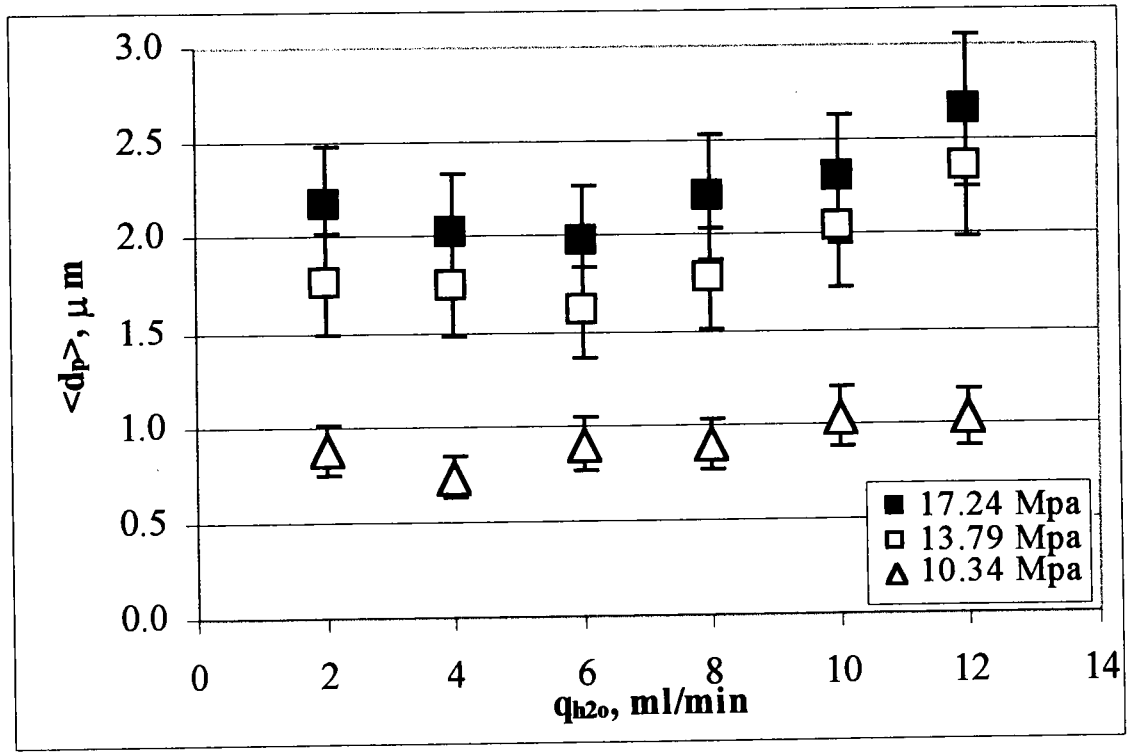


Figure 31. Droplet size characterization at 2.7 kV/cm, 133 Hz, and 35°C.

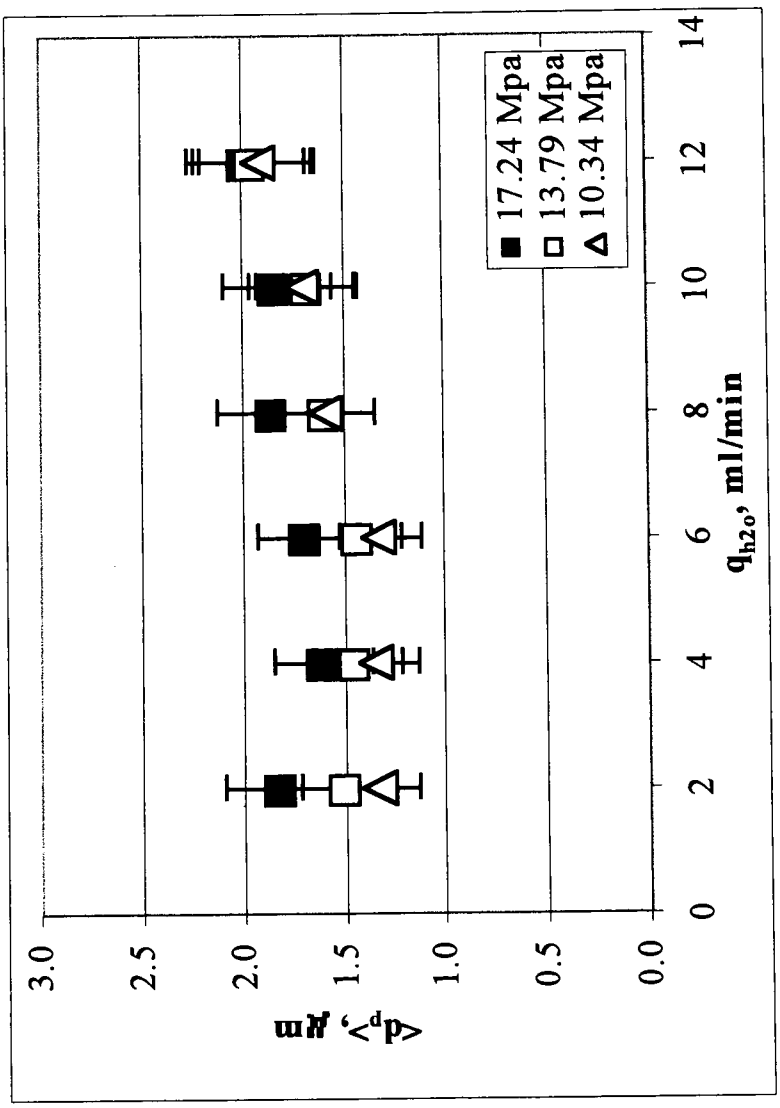


Figure 32. Droplet size characterization at 2.7 kV/cm, 133 Hz, and 47°C.

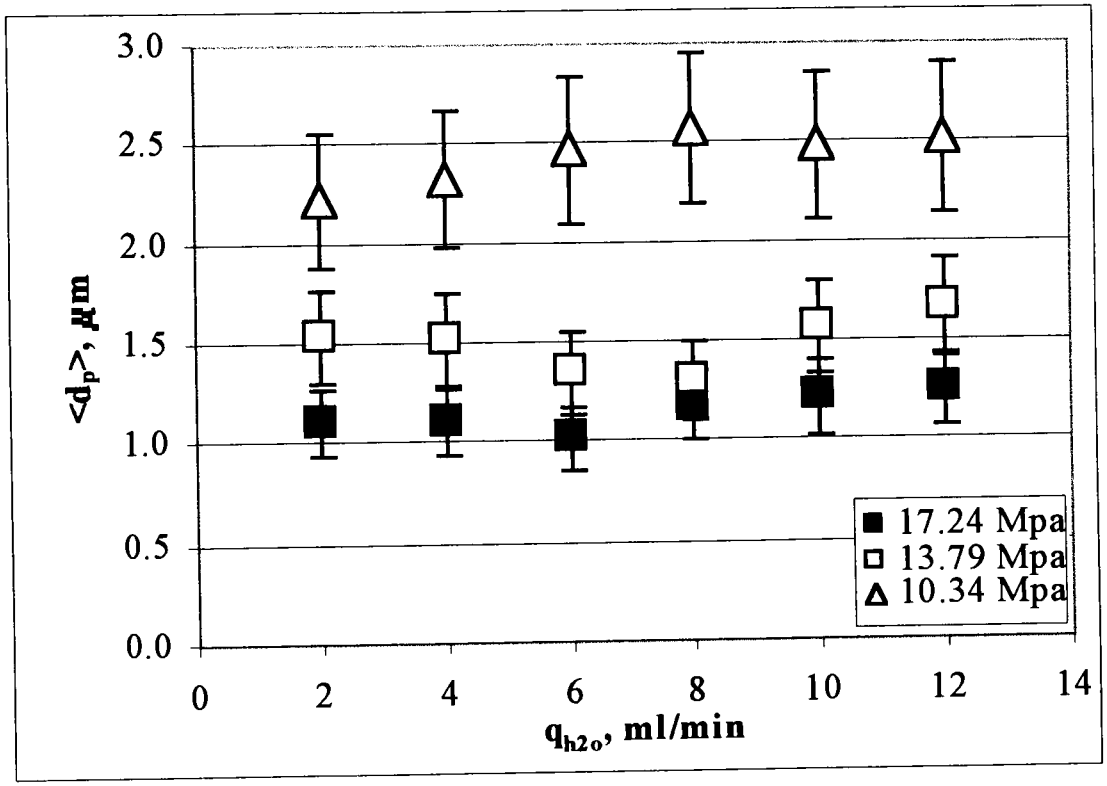


Figure 33. Droplet size characterization at 2.7 kV/cm, 133 Hz, and 65°C.

MPa. All sets show a slight increase of $<d_p>$ with increasing flow rate. The 17.24 MPa set range from 1.1 μm at 2 ml/min to 1.3 μm at 12 ml/min. The 13.79 MPa data set shows slightly larger $<d_p>$ than the 17.24 MPa data set at all flow rates. The 10.34 MPa data set shows much larger $<d_p>$, 2.2 μm at 2 ml/min and 2.5 μm at 12 ml/min.

Figure 34 contains literature data taken from Chun and Wilkinson (81) for interfacial tension of a 10 vol. % aqueous ethanol solution in CO_2. The plot displays the effect of pressure on interfacial tension for three temperatures (35, 40, and 65°C). At low to moderate pressure, the reduction in interfacial tension with pressure is a result of the increased solubility of carbon dioxide in the aqueous phase. This plot shows that there is a minimum in the interfacial tension near the critical condition of SCO_2 for lower temperatures. The higher temperature data set (65°C) does not show a minimum while the deepest minimum occurs for the 35°C data set. As described by Chun and Wilkinson, the rise in interfacial tension after the minimum (as pressure increased) implies a desorption process, and the minimum itself occurs as a consequence of the appearance of a small quantity of a third-phase intermediate in composition between the aqueous solution and carbon dioxide rich phases.

Figure 35 is a plot of experimental $<d_p>$ obtained from the present work versus interfacial tension data from Figure 34 (81). The details for each data point are included in the plot. The data points represented by squares are slightly mismatched because the $<d_p>$ was measured for a temperature of 47°C while the interfacial tension data point was obtained for a temperature of 40°C. A linear regression of the remaining data with a correlation coefficient (R^2) of 0.9184 is included. This plot shows that there exists a weak dependence of $<d_p>$ on interfacial tension. This observation coincides with the

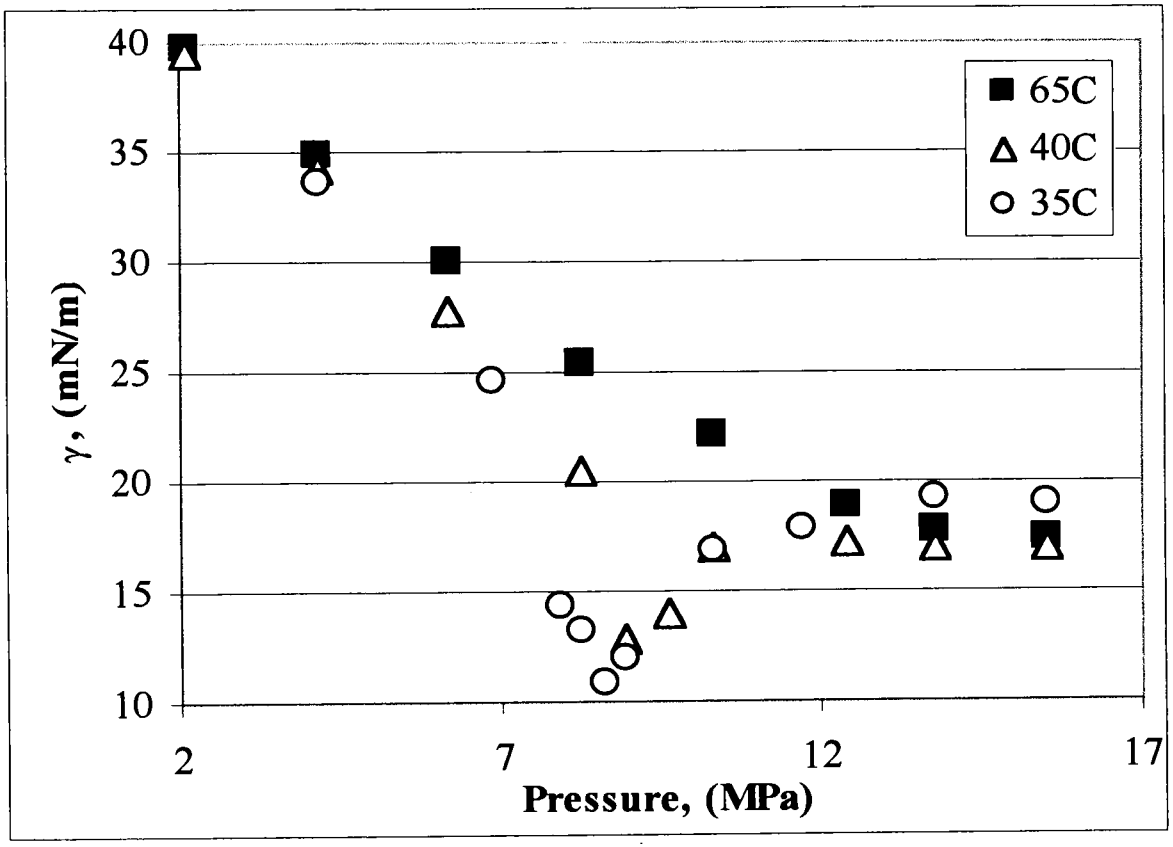


Figure 34. Interfacial tension data taken from Chun and Wilkinson (81) for 10 vol. % ethanol in CO_2.

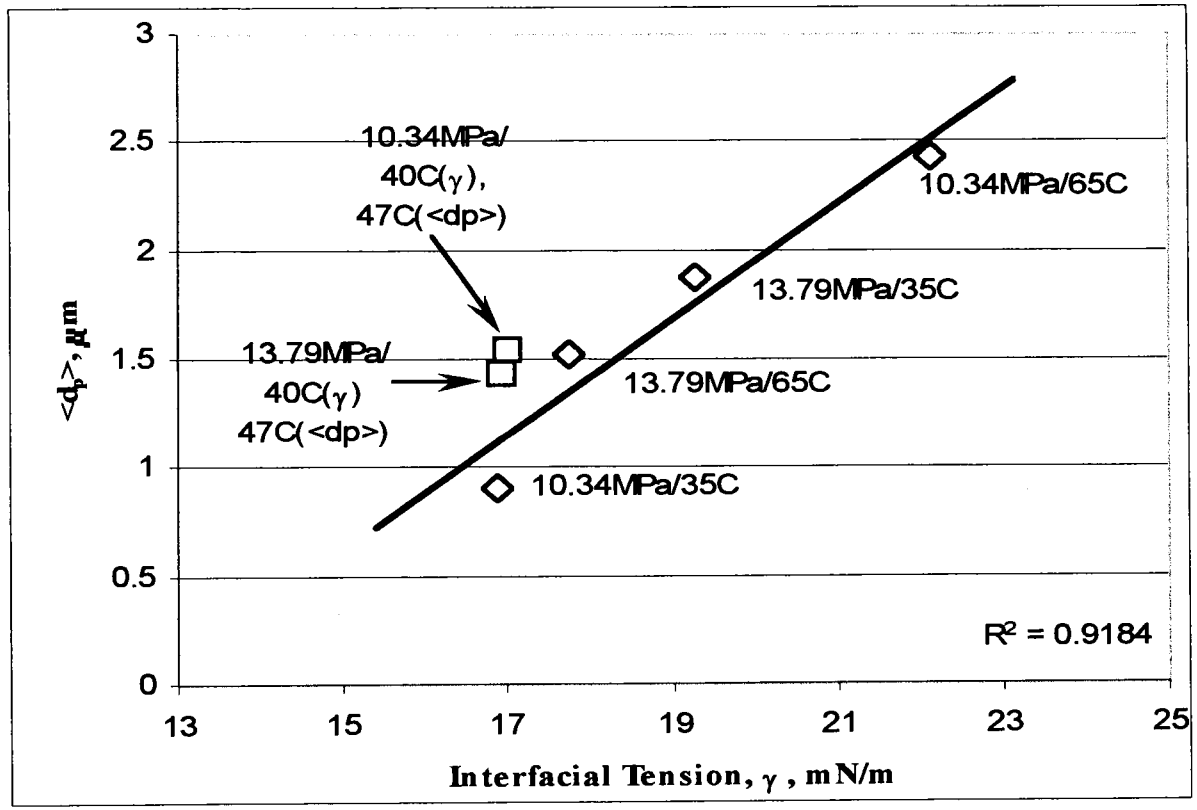


Figure 35. Droplet size correlation with interfacial tension for 10 vol. % aqueous ethanol in SCO_2.

weak dependence of surface tension on the minimum droplet diameter relationship presented in equation 2. Although this plot of surface tension presents evidence for why the observed reversal of the $<d_p>$ occurs at higher pressures and higher temperatures, it is not the only cause for changes of $<d_p>$ with state conditions. Several other physical properties are changing as well, including the liquid density and especially the CO_2 density, which both will vary with temperature and pressure. In addition, electrical conductivity and dielectric constant of the system will also change with state conditions. A tabulation of all data in Figures 28-35 is presented in Appendix D.

Several interesting relationships have been presented. For a pressure of 10.34 MPa, it was observed that increasing temperature increases $<d_p>$; however, at 17.24 MPa increasing temperature decreases $<d_p>$. This reversal was also seen when plotting the data for constant temperature. At 35°C, it was observed that increasing pressure increases $<d_p>$, again however, for 65°C increasing pressure decreased $<d_p>$. Although, these observations initially appear puzzling, they can be correlated with at least one (surface tension) of the several physical properties of the system that are changing with both temperature and pressure.

4.4 Effect of Temperature and Pressure on the Extraction of Ethanol from an Electrospray Solution of 10 vol. % Aqueous Ethanol Using SCO_2

In prior sections the attention has been the characterization of the EDC by studying the effects of operating conditions on $<d_p>$. In this section, the focus is on using the EDC to perform an extraction of ethanol from a 10 vol. % aqueous ethanol solution using SCO_2. The effects of varying temperature (35 and 65°C) and pressure (10.34 and 17.24 MPa) are presented and compared to theoretical equilibrium predictions

based on flash calculations using the Peng-Robinson (PR) equation of state model. A total of four state conditions was studied: 35°C/10.34 MPa, 35°C/17.24 MPa, 65°C/10.34 MPa, and 65°C/17.24 MPa.

Before an analysis of a ternary system is started, equilibrium data for each of the binary systems should be studied. A binary form of the PR equation of state (see Appendix C.1) was used to regress literature data (64, 74) for each of the binary systems H_2O-C_2H_5OH, H_2O-CO_2, and C_2H_5OH-CO_2 at each of the four state conditions. Table 1 contains the binary interaction parameters (k_{ij}) from this regression. Although k_{ij} does not vary much with pressure, it is apparent that tuning is required to obtain accurate k_{ij} for different temperatures. When the regression is complete, there are four sets of k_{ij} to be used for the four state conditions under study.

The binary k_{ij} values are used in the ternary form of the PR equation of state (see Appendix C.2) to model the ternary system H_2O-C_2H_5OH-CO_2. Table 2 contains average absolute deviations (AAD) for x_2 (ethanol mol fraction in liquid phase), y_1, and y_2 (water and ethanol mol fractions in vapor phase, respectively) in comparison to literature data (64, 74) at each of the four state conditions. Along with AAD, the number of data points used in the regression and literature sources are included. This table is presented to demonstrate the accuracy of the PR model with the ternary system at state conditions similar to those tested in this work. Next, flash calculations were completed using a mass balance and the ternary PR equation of state to model the EDC at each of the four state conditions. This code has been included in Appendix C.3.

Figure 36 displays the effect of increasing CO_2 flow rate on the separation factor, α, at two pressures (10.34 and 17.24 MPa) for a constant temperature of 35°C. The

Table 1. Binary Interaction Parameters (k_{ij})					
		35 C		45 C	
kij	Binary system	1500 psi	2500 psi	1500 psi	2500 psi
k_{12}	H_2O-C_2H_5OH	-0.127	-0.124	-0.079	-0.099
k_{13}	H_2O-CO_2	0.115	0.114	0.090	0.090
k_{23}	C_2H_5OH-CO_2	0.103	0.103	0.085	0.085

Table 2. Average Absolute Deviation (AAD) for Regression of Ternary PR with Literature Data				
	35 C		45 C	
	10.34 MPa	17.24 MPa	10.34 MPa	17.24 MPa
x_2	0.0095696	0.063243	0.040333	0.051512
y_1	0.001477	0.0117	0.0006363	0.0153
y_2	0.00385	0.01954	0.002931	0.03612
Literature source:	74	64	64	64
Data points:	14	4	7	3
H_2O-C_2H_5OH-CO_2 system, (1)-(2)-(3) x & y--denotes liquid & vapor mol fractions, resepectively				

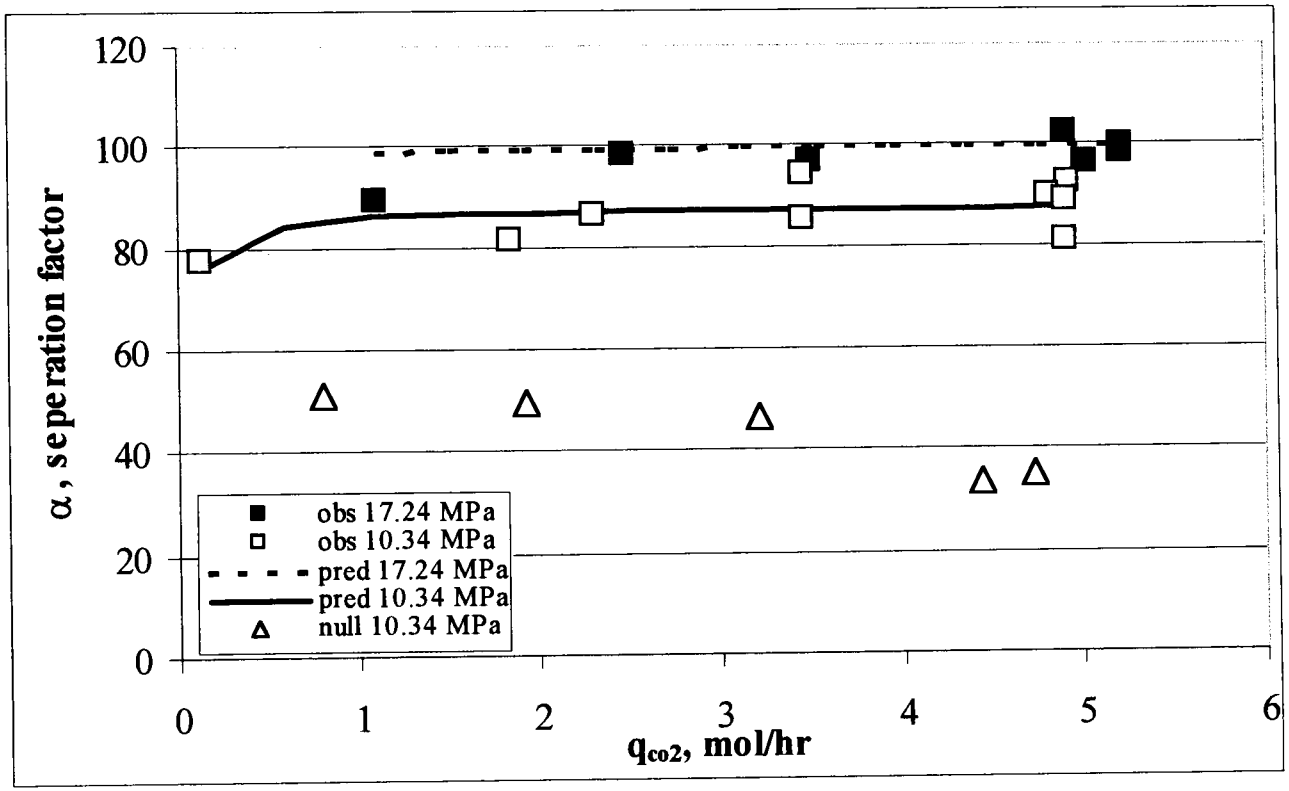


Figure 36. Mass transfer evaluation of EDC at 35°C for 2.7kV/cm, 133Hz, and 10 vol. % aqueous ethanol flow rate of 4 ml/min.

separation factor, α, is defined as $(y_2/x_2)/(y_1/x_1)$ where x and y denote mol fractions in the liquid and vapor phases, respectively. The experimental data are represented by points and the theoretical predictions are represented as lines. The null experiment consists of having all experimental conditions the same except the electric field has not been engaged. This results in an approximate droplet size of around 2.4 mm, which is over 1000 times larger than with the field on where the mean droplet size is around 1.0 μm. The separation factors for both pressures when the field was on appear to remain constant with molar flow rate of CO_2. The higher pressure data set (17.24 MPa), however, has slightly larger separation factors (100 compared to 88). For both data sets, the experiments correlate well with the theoretical data, leading to the conclusion that the system achieves essentially 100 % efficiency of a single theoretical stage at these conditions. The null experiments show separation factors as high as 55 % of the predicted equilibrium values. From the comparison between null and experimental separation factors, the full benefit of the electric field is not apparent. If the CO_2 molar flow rate could be increased substantially, perhaps the true benefit of reducing aqueous droplet size by three orders of magnitude would be observed. However, since the apparatus was limited to only lower CO_2 flow rates, the enhancement was not seen. A more appropriate set of conditions is recommended for future mass transfer characterization.

Figure 37 displays the effect of increasing CO_2 flow rate on the separation factor, α, for the same two pressures (10.34 and 17.24 MPa), but at a higher constant temperature of 65°C. As with the 35°C plot the separation factors at 65°C appear to be constant with molar flow rate of CO_2, and correlate well with theoretical calculations

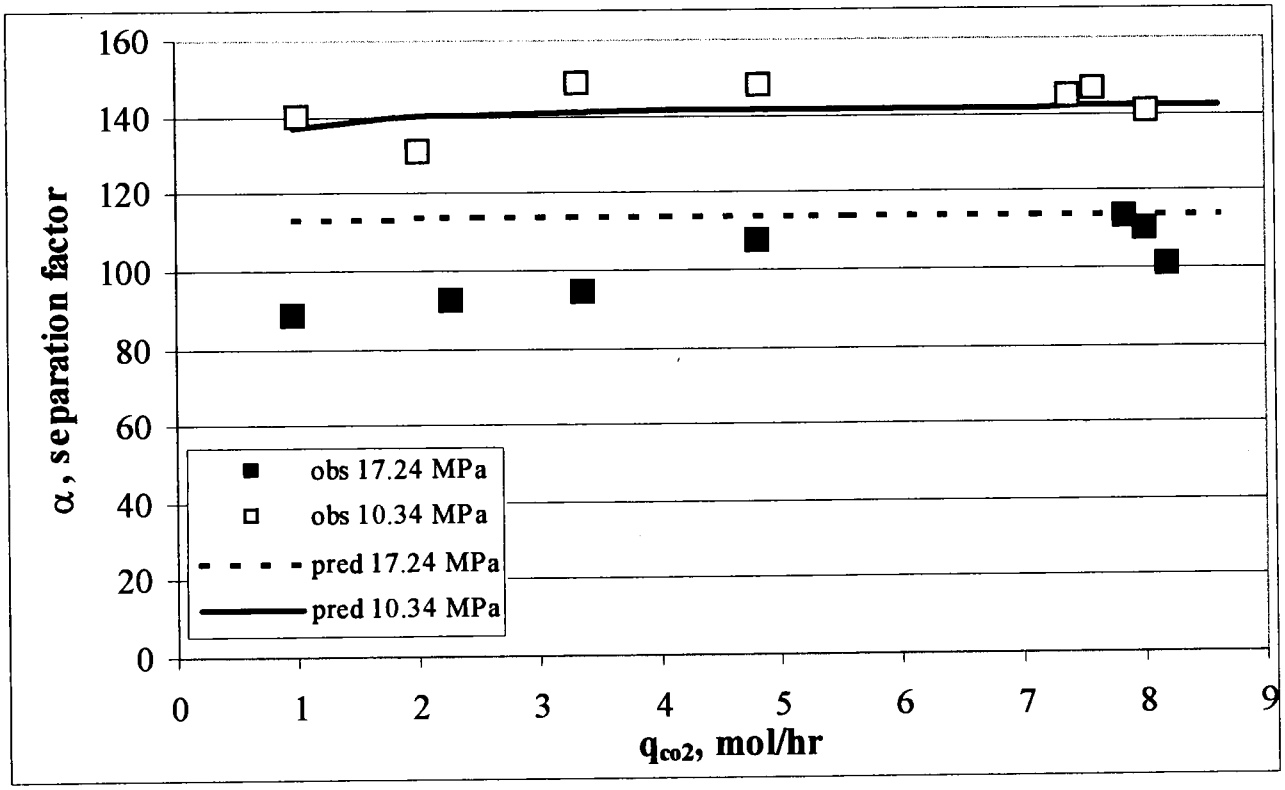


Figure 37. Mass transfer evaluation of EDC at 65°C for 2.7kV/cm, 133Hz, and 10 vol. % aqueous ethanol flow rate of 4 ml/min.

indicating they also achieved 100 % efficiency of a single theoretical stage. The higher pressure (17.24 MPa) data set has separation factors (90-110) similar to the data taken at 35°C. The lower pressure (10.34 MPa) data set, however, has considerably larger separation factors around 140-145. Based on the published equilibrium data for the H_2O-C_2H_5OH-CO_2 system (64-66, 73, 75, 76, 79) at 65°C and 10.34 MPa, one would have expected an extract composition above the atmospheric concentration under conditions where the composition of the extract liquid was above 89.4 mol % C_2H_5OH. This could have been achieved with higher feed concentrations.

It has been well documented that for a given temperature, there exists a specific pressure below which it is possible to concentrate aqueous ethanol above the atmospheric azeotropic composition (89.4 mol. %). This is possible when the pressure in the ternary system is below the critical pressure of the binary system C_2H_5OH-CO_2 (64). For a temperature of 35°C, the critical pressure is 8.62 MPa. However, for 65°C the critical pressure is 12.20 MPa (64). This may be the reason for the larger separation factors in the 65°C and 10.34 MPa state condition. While ethanol concentrations in the vapor phase (y_2) were fairly constant at approximately 72.4 mol. % (CO_2-free basis) for three of the state conditions, the y_2 for the 65°C and 10.34 MPa data set rose to 81.4 mol. % (CO_2-free basis). Although these data do not exceed the atmospheric azeotropic composition, they do support an increase in effectiveness at this condition.

There is an important conclusion that can be drawn from the observation that the separation achieved in the EDC is determined only by equilibrium for all conditions studied up to about 8 mol/hr CO_2 flow rate. Entrainment of micron-size liquid droplets from the EDC is apparently negligible; otherwise the measured extract ethanol

concentration would have been lower than the equilibrium concentration. The liquid feed concentration and the liquid effluent concentration were always lower than the extract concentration. Any entrainment of the liquid would have decreased the measured extract concentration below the equilibrium value. It is generally believed that the micron-size droplets produced by electrospraying are electrically charged. Charged droplets would be strongly attracted to the wire mesh electrode. Apparently, a high degree of droplet coalescence at the wire mesh electrode was achieved in the EDC, resulting in negligible liquid entrainment.

An attempt was made to increase the CO_2 flow rate such that the extraction of ethanol would no longer be equilibrium limited. Figure 38 shows three additional data points at substantially higher CO_2 flow rates. These flow rates were beyond the limit of the tube type condensers, and the extractions are not accurate. They are included to examine for indication of possible departure from equilibrium data. Even at these high flow rates, the rate of extraction appears to be controlled by equilibrium; however, due to the large scatter of the data, this result is not conclusive.

The experimental results were analyzed from a different approach by plotting the rate of extraction of ethanol and water versus the molar flow rate of CO_2. Figure 39 gives the rates of extraction of ethanol and water at a pressure of 10.34 MPa and temperature of 35°C. This figure includes experimental data with the field on and the null experiment where the field was turned off. The rates of extraction for both ethanol and water with the field on have a strong linearity supported by correlation coefficients of 0.8693 and 0.8803, respectively. The null experiment, however, does not demonstrate consistent linear relationships between the rates of extraction and CO_2 molar flow rates. The initial

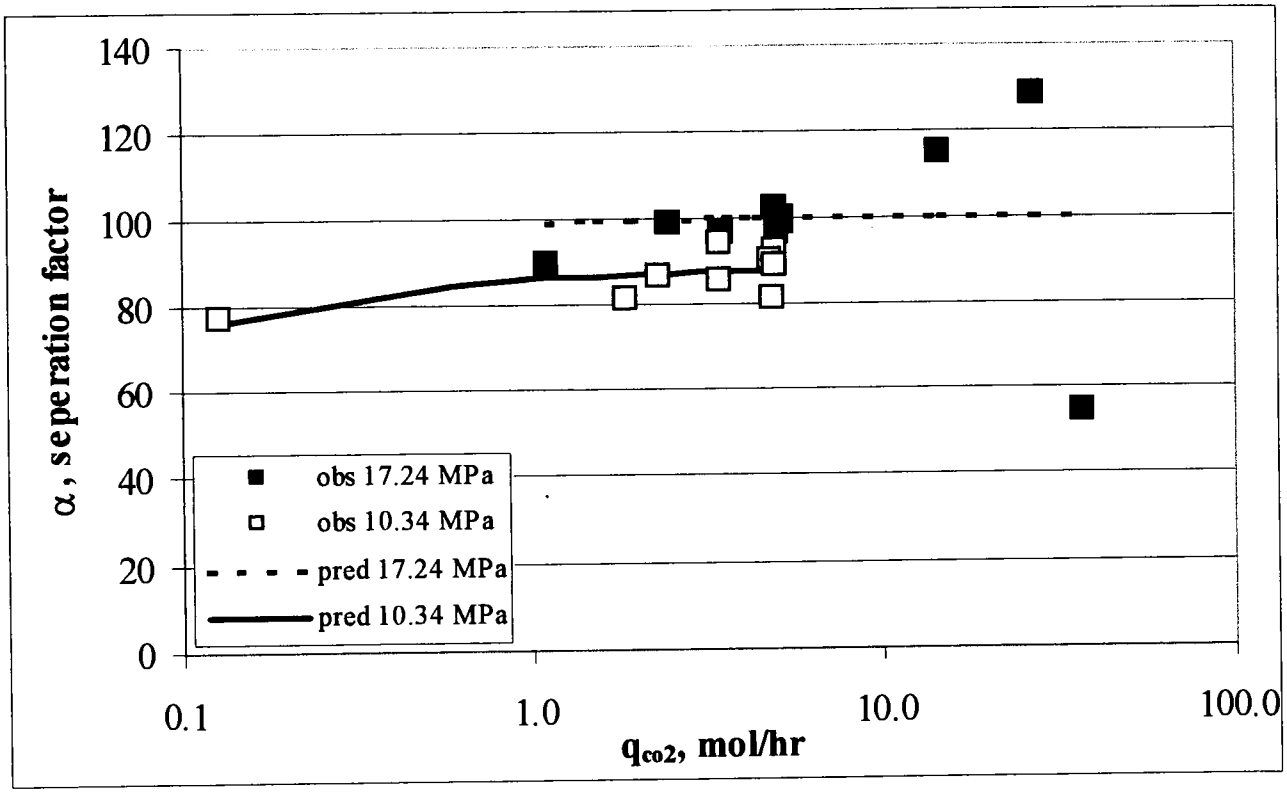


Figure 38. Mass transfer evaluation of EDC at 35°C for 2.7kV/cm, 133Hz, and 10 vol. % aqueous ethanol flow rate of 4 ml/min at high flow rates.

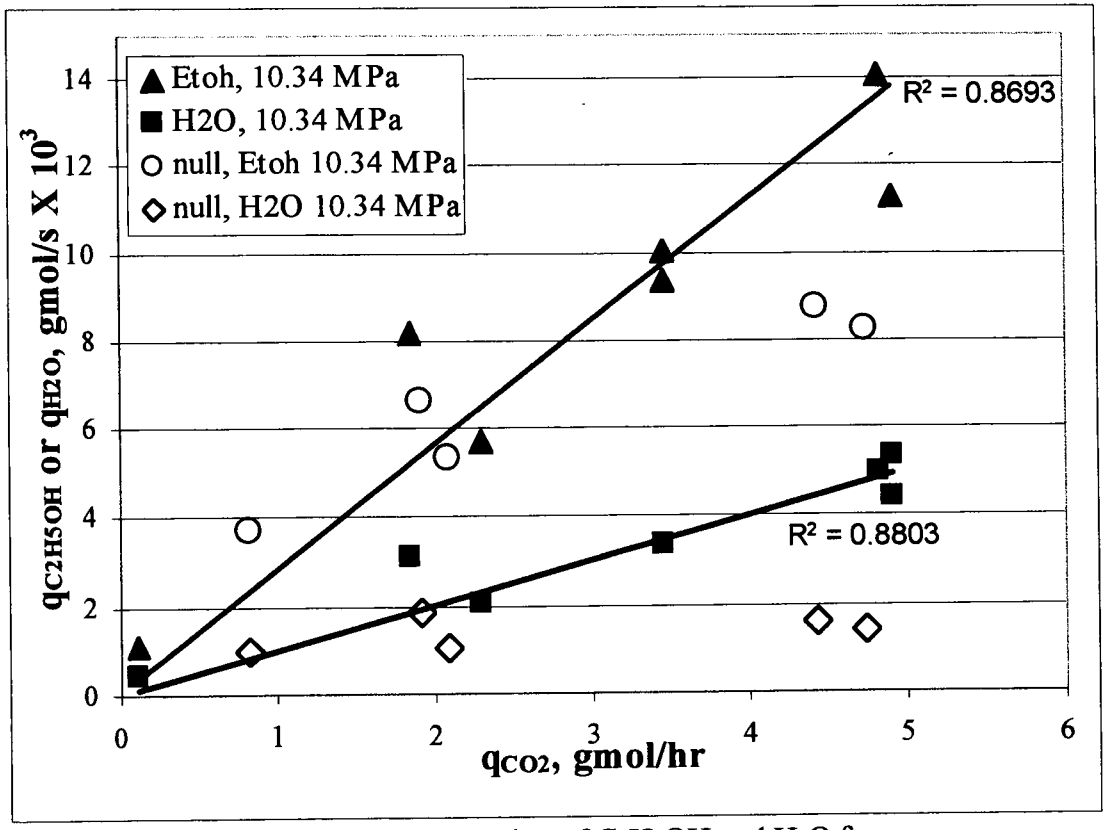


Figure 39. Rate of extraction of C_2H_5OH and H_2O from a 10 vol. % aqueous ethanol flow rate of 4 ml/min, 2.7 kV/cm, and 133 Hz at 35°C with null experiment included.

rates of extraction (< 2 gmol/hr) for the null experiments are similar to the field on experiments. At slightly higher CO_2 molar flow rates (> 2 gmol/hr), however, the rates appear to reach the mass transfer limit and level off.

Figure 40 gives the rates of extraction of ethanol and water at both pressures (10.34 and 17.24 MPa) for 35°C. The plot shows that for a CO_2 molar flow rate of 3.5 gmol/hr, the extraction rate of ethanol nearly doubles going from 10 to 20 gmol/s * 10^{-3} when increasing the pressure from 10.34 to 17.24 MPa. A similarly noticeable increase is observed for the rate of extraction of water at the same CO_2 molar flow rate. Here an increase from 3 to 7 gmol/s * 10^{-3} occurs when increasing the pressure from 10.34 to 17.24 MPa. The maximum attainable rate of extraction of ethanol for this temperature is 23 gmol/s * 10^{-3}, and occurs at 17.24 MPa with a CO_2 molar flow rate of 5 gmol/hr.

Figure 41 gives the rates of extraction of ethanol and water at both pressures (10.34 to 17.24 MPa) for 65°C. Unlike the data from Figure 40, these data have regression lines that have very similar slopes. Across all CO_2 molar flow rates, however, the rate of extraction was higher for the increased pressure data set. The maximum attainable rate of extraction of ethanol for this temperature is 48 gmol/s * 10^{-3}, and occurs at 17.24 MPa with a CO_2 molar flow rate of 8 gmol/hr. Clearly the change in temperature from 35 to 65°C has a greater impact on the rate of extraction of ethanol with resulting increases from 15 to 37 gmol/s * 10^{-3} at 10.34 MPa and 23 to 48 gmol/s * 10^{-3} at 17.24 MPa. Whereas if the focus is shifted to changing pressure from 10.34 to 17.24 MPa more mild increases are obtained from 15 to 23 gmol/s * 10^{-3} at 35°C and 37 to 48 gmol/s * 10^{-3} at 65°C. The linear increase of ethanol and water extraction rates with CO_2 molar flow rate (indicated by strong correlation coefficients, R^2) at each of the four state

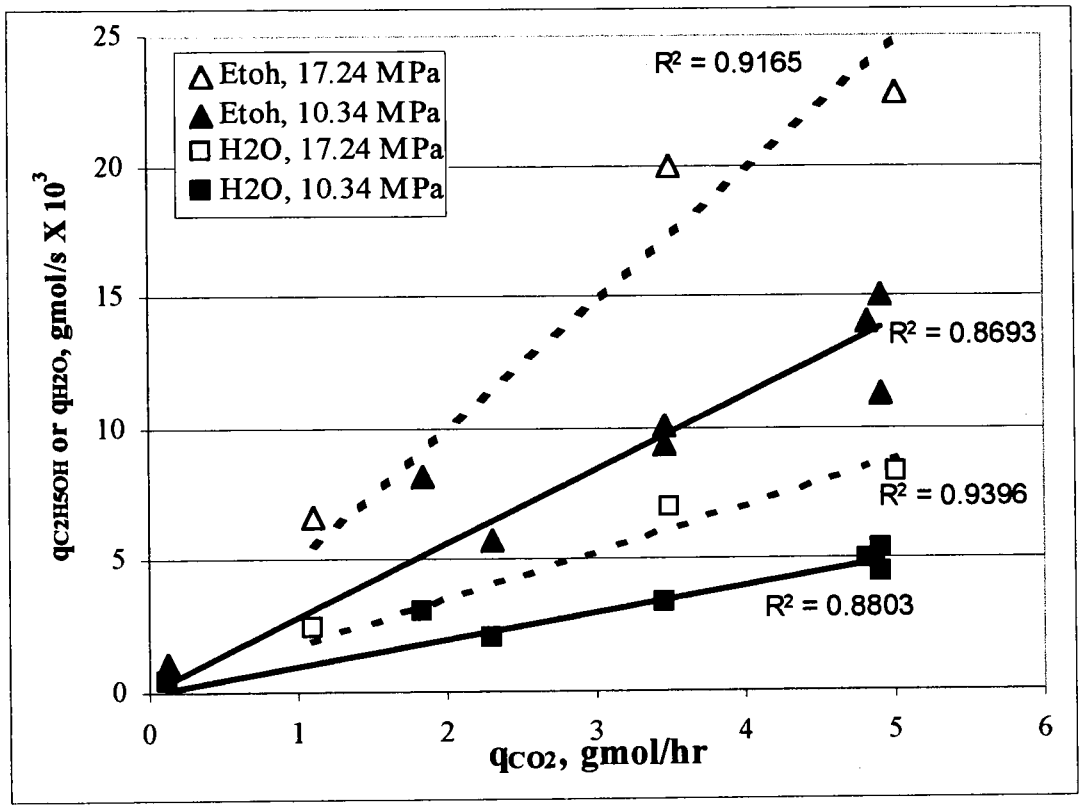


Figure 40. Rate of extraction of C_2H_5OH and H_2O from a 10 vol. % aqueous ethanol flow rate of 4 ml/min, 2.7 kV/cm, and 133 Hz at 35°C.

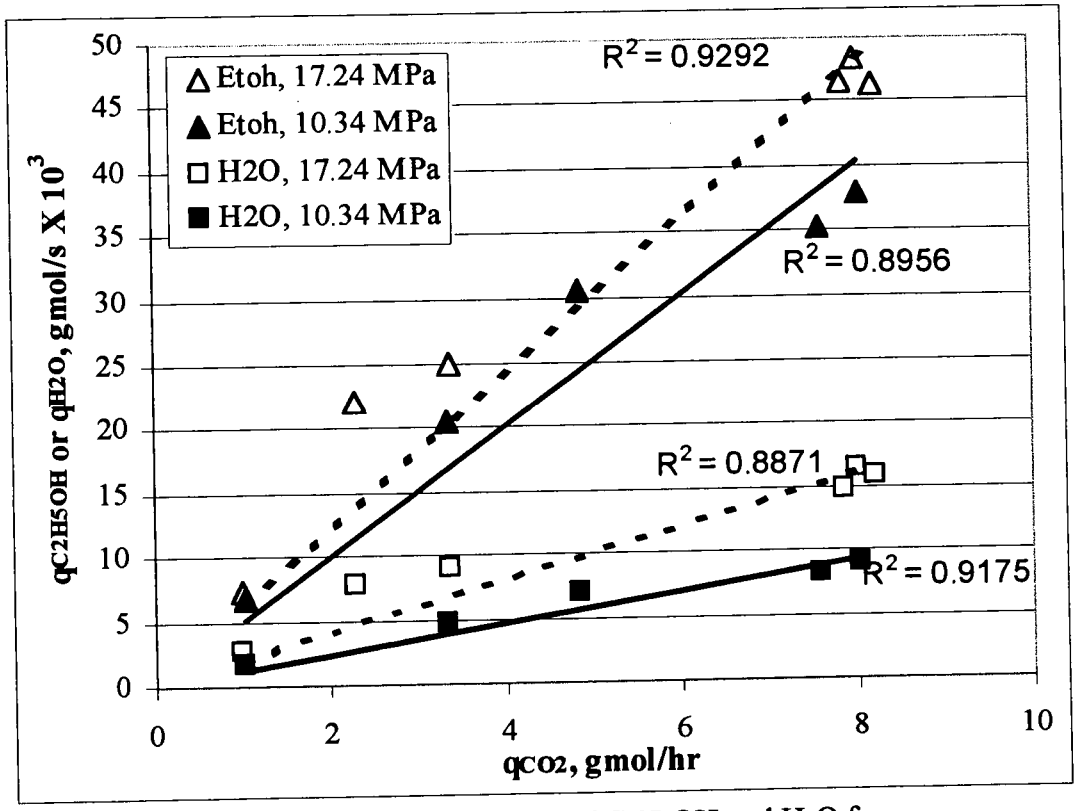


Figure 41. Rate of extraction of C_2H_5OH and H_2O from a 10 vol. % aqueous ethanol flow rate of 4 ml/min, 2.7 kV/cm, and 133 Hz at 65°C.

conditions with the electric field on is further confirmation that extraction in the EDC is limited by equilibrium at these flow rates. A tabulation of all data in Figures 36-41 is presented in Appendix D.

Extraction of ethanol from a 10 vol. % aqueous ethanol solution using SCO_2 in the EDC appears to be possible. Entrainment of micron-size liquid droplets from the EDC is apparently negligible; otherwise the measured extract ethanol concentration would have been lower than the equilibrium concentration. In all extraction experiments the results were controlled by equilibrium, and, as shown by the null experiment, limitations of the apparatus prevented study of mass transfer performance of the EDC. It was hoped that CO_2 flow rates could be reached where the observed rate of mass transfer was much greater than that observed in the null experiment and where the ethanol extraction was mass transfer limited. Unfortunately, this was not achieved because of the limited capacity of the condensers. This would have allowed evaluation of the intrinsic mass transfer rate of the EDC.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

A new method for dispersing aqueous liquid into SCO_2 was demonstrated using pulsed high-voltage electric fields. System conditions were varied from 35°C to 65°C at pressures from 10.34 to 17.24 MPa. The apparatus was fitted with a laser light scattering system so that it was possible to determine the dependence of mean droplet size with various operating parameters such as field strength, pulse frequency, flow rate, temperature, and pressure. An extraction of ethanol from a 10 vol. % aqueous ethanol solution was performed, and compared with theoretical flash calculations using a ternary form of the Peng-Robinson equation of state.

The effect of varying field strength, pulse frequency, and liquid flow rate was examined on $<d_p>$, the number averaged droplet diameter, for dispersing pure water in SCO_2. All experiments using pure water were carried out at 10.34 MPa and 40°C. Increases in field strength caused $<d_p>$ to increase. This is opposite the effect normally observed when operating in the cone-jet mode of electrosprays. It is believed that the mode of electrospray in this research may extend past the cone-jet mode, because beyond the cone-jet mode increases in instability and polydispersity can increase $<d_p>$. In all cases, whether increasing flow rate or field strength, lowering the pulse frequency appeared to give smaller $<d_p>$. The effect of increasing $<d_p>$ with increasing flow rate demonstrated by other researchers was confirmed. This was the first study of electrospraying a liquid into a dense supercritical fluid.

A study varying temperature, pressure, and flow rate was completed using a liquid feed containing 10 vol. % ethanol in distilled water (concentrations found in fermentation broths). These experiments examined the effect of flow rate (2-12 ml/min), pressure (10.34, 13.79, 17.24 MPa), and temperature (35, 47, 65°C) on size of the dispersion. The field strength and pulse frequency were held constant throughout these experiments. Several interesting relationships have been presented. For a pressure of 10.34 MPa, it was observed that increasing temperature increases $<d_p>$; however, at 17.24 MPa increasing temperature decreases $<d_p>$. This reversal was also seen when plotting the data for constant temperature. At 35°C, it was observed that increasing pressure increases $<d_p>$, again however, for 65°C increasing pressure decreased $<d_p>$. Although, these observations initially appear puzzling, variation of $<d_p>$ correlate with surface tension, one of the several physical properties of the system that are changing with both temperature and pressure.

Extraction of ethanol from a 10 vol. % aqueous ethanol solution using SCO_2 in the EDC appears to be possible. It occurs with 100 % efficiency for a single theoretical stage with mass transfer rates up to 1.34×10^{-4} gmol/cm^{3}s. (Since less than half of the reactor volume was utilized, a similar rate may have occurred with a smaller cell volume.) This was confirmed for four state conditions (10.34 and 17.24 MPa each at 35 and 65°C) and verified using a ternary PR equation of state by examining the experimental and theoretical separation factors. This observation was further supported by linear rates of extraction for both ethanol and water with molar flow rates of CO_2.

Since the separation achieved in the EDC is determined only by equilibrium for all conditions studied up to about 8 mol/hr CO_2 flow rate, entrainment of micron-size

liquid droplets from the EDC is apparently negligible; otherwise the measured extract ethanol concentration would have been lower than the equilibrium concentration. The liquid feed concentration and the liquid effluent concentration were always lower than the extract concentration. Any entrainment of the liquid would have decreased the measured extract concentration below the equilibrium value. It is generally believed that the micron-size droplets produced by electrospraying are electrically charged. Charged droplets would be strongly attracted to the wire mesh electrode. Apparently, a high degree of droplet coalescence at the wire mesh electrode was achieved in the EDC, resulting in negligible liquid entrainment.

It was hoped that CO_2 flow rates could be reached where the rate of extraction was much higher than that observed in the null experiment and where the ethanol extraction was mass transfer limited. Unfortunately, this was not achieved because of the limited capacity of the condensers. This would have allowed evaluation of the intrinsic mass transfer rate of the EDC.

Recommended future work would include measuring the mass transfer coefficients. It would be desirable to design and install condensers capable of operating at substantially higher CO_2 flow rates so that the intrinsic mass transfer performance of the EDC can be studied. Since the current operation measures a number average dispersion size, it is difficult to accurately determine the total surface area in contact with the gas phase. It would be beneficial to disperse a water soluble dye so that water holdup could be measured photometrically. If it were possible to measure the suspended liquid volume, then a calculation could be made of total surface area which would then lead to the determination of mass transfer coefficients.

The use of pulsed electric fields to produce a fine dispersion of aqueous droplets into supercritical CO_2 appears to be an attractive possibility for improved operations where an environmentally friendly system is desired. This project establishes a basis for (1) broadening the range of applicability of solvent substitution by supercritical CO_2, (2) contributing to the fundamental knowledge of droplet formation in a unique experimental way (supercritical CO_2 affords the range of variables not available in most experiments), and (3) exploring the behavior of micro-droplets in supercritical fluids. In addition, these dispersions may have the capability of lending themselves to a variety of applications including dyeing operations, reactions, and the formation of fine particles. The adaptation of such schemes to industrial applications could provide systems which are an order of magnitude more efficient as well as contain enormous environmental benefits. Likely initial commercial applications of this technology might be found for systems where a small volume of high-value product must be extracted from an aqueous medium. Possible applications of this type might be found in the pharmaceutical industry.

REFERENCES

REFERENCES

1. Widrig, C.L., J. Tao, and D. Allen, Basic Research Needs for Environmentally Responsive Technologies of the Future, results of a workshop sponsored by NSF and DOE, Jan 4-6, 1996, page 13.

2. Harrison, K.L., J. Goveas, and K.P. Johnston, "Water-in-Carbon Dioxide Microemulsions with a Fluorocarbon-Hydrocarbon Hybrid Surfactant," *Langmuir*, **10,** 3536 (1994).

3. Clark, M.J., K.L. Harrison, K.P. Johnston, and S.M. Howdie, "Water in Supercritical Carbon Dioxide Microemulsions: Spectroscopic Investigation of a New Environment for Aqueous Inorganic Chemistry," *J. Am. Chem. Soc.*, **119,** 6399 (1997).

4. Heitz, M.P., C. Carlier, J. deGrazia, K.L. Harrison, K.P. Johnston, T.W. Randolph, and F.V. Bright, "Water Core within Perflurorpolyether-Based Microemulsions Formed in Supercritical Carbon Dioxide," *J. Phys. Chem. B,* **101,** 6707 (1997).

5. Fulton, J.L., D.M. Pfund, J.B. McClain, T.J. Romack, E.E. Maury, J.R. Combes, E.T. Samulski, J.M. DeSimone, and M Capel, "Aggregation of Amphiphilic Molecules in Supercritical Carbon Dioxide: A Small Angle X-ray Scattering Study," *Langmuir*, **11,** 4241 (1995).

6. Beckman, E.J., "Carbon Dioxide Extraction of Biomolecules," *Science*, **271**, 613 (1996).

7. Johnston, K.P., *et al.*, "Water-in-Carbon Dioxide Microemulsions: An Environment for Hydrophyles Including Proteins," *Science*, **271**, 624 (1996).

8. McClain, J.B.J., J.D. Londono, D. Chillura-Martino, R. Triolo, D.E. Betts, D.A. Canelas, H.D. Cochran, E.T. Samulski, J.M. DeSimone and G.D. Wignal, *Science*, **274**, 2049 (1996).

9. Londono, J.D., R. Dharmapurikar, H.D. Cochran, G.D. Wignal, J.B. McClain, J.R. Combes, D.E. Betts, D.A. Canelas, J.M. DeSimone, E.T. Samulski, D. Chillura-Martino, and R. Triolo, *J. Appl. Cryst.*, **30**, 690 (1997).

10. Cooper, A.I., J.D. Londono, G.D. Wignal, J.B. McClain, E.T. Samulski, J.S. Lin, A. Dobrynin, M. Rubinstein, J.M.J. Frechet and J.M. DeSimone, *Nature*, in press.

11. Hayter, J.B., Physics of Amphiphiles, Miscelles, Vesicles and Microemulsions, V. de Giogio and M. corti eds., North Holland, Amsterdam, 1985.

12. Magid, L.J., "Elucidation of Miscellar and Microemulsion Architecture using SANS", *Colloids and Surfaces*, Elsevier, **19**, 129 (1986).

13. Caponetti, E., and R. Triolo, *Adv. Colloid and Interface Sci.*, **32**, 235 (1990).

14. Scott T.C., and R.M. Wham, "Surface Area Generation and Droplet Size Control in Solvent Extraction Systems Utilizing High Intensity Electric Fields," U. S. Patent No. 4,767,929 (1988).

15. Scott, T.C. "Magnetic/Electric Field Solvent Extraction," U. S. Patent No. 4,941,959 (1989).

16. Harris, M.T., W.G. Sisson, O.A. Basaran, "Improved Nozzle for Electrical Dispersion Reactor," U. S. Patent No. 5,464,195 (1995).

17. Harris, M.T., T.C. Scott, and C.H. Byers, "Method and Apparatus for Preparing Metal Oxide Powders," U.S. Patent No. 5,122,360 (1992).

18. Harris, M.T., T.C. Scott, and C.H. Byers, "Method and Apparatus for the Production of Metal Oxide Powder," U. S. Patent No. 5,207,973 (1993).

19. Scott T.C., and R.M. Wham, "An Electrically Driven Multistage Countercurrent Solvent Extraction Device: the Emulsion-Phase Contactor," *Ind. Eng. Chem. Res.*, **28**, 94 (1989).

20. Harris, M.T., T.C. Scott, O.A. Basaran, and C.H. Byers, " Morphology Control in Precursor Ceramic Powder Production by the Electrical Dispersion Reactor," *Mat. Res. Soc. Symp. Proc.*, **180**, 153 (1990).

21. Harris, M.T., T.C. Scott, O.A. Basaran, and C.H. Byers, "Formation of Y-Ba-Cu (1-2-3) Hydrous Oxide Precursor Powders in the Electric Dispersion Reactor," *Superconducting Engineering, AIChE Symp. Ser.*, **88**, 44 (1992).

22. Harris, M.T., W.G. Sisson, T.C. Scott, O.A. Basaran, C.H. Byers, W. Renand, and T. Meek, "Multiphase Electrodispersion Precipitation of Zirconia Powders," *Mat. Res. Soc. Symp. Proc.*, **271**, 945 (1992).

23. Harris, M.T., T.C. Scott, and C.H. Byers, "The Synthesis of Metal HydrousOxide Particles by Multiphase Electrodispersion," *Mat. Sci. Eng.*, **A168**, 125 (1993).

24. Scott, T.C., D.W. DePaoli, and W.G. Sisson, "Further Development of the Electrically Driven Emulsion-Phase Contactor," *Ind. Eng. Chem. Res.*, **33**, 1237 (1994).

25. Harris, M.T., W.G. Sisson, S.M. Hayes, S.J. Bobrowski, and O.A. Basaran, "Porous Spherical Shells and Microspheres by Electrodispersion Precipitation," *Mat. Res. Soc. Symp. Proc.*, (1995).

26. Scott, T.C., J.M. Cosgrove, D.W. DePaoli, and W.G. Sisson, "Use of the Electrically Driven Emulsion-Phase Contactor for a Biphasic Liquid-Liquid Enzyme System: the Oxidation of p-Cresol in Toluene by Aqueous-Phase Horseradish Peroxidase," *Appl. Biochem. Biotech.*, **51/52**, 283 (1995).

27. Borzabadi, E., Bailey, A. G., "The properties of Axially Symmetric Electrified Pendent Drops," *J. Electrostatics,* **5**, 369, (1978).

28. Tang, K., A. Gomez, *Phy. Fluids A*, **6**, 2317 (1994a).

29. Fernandez de la Mora, J., and J. Navascues, *et al., J.Aerosol Sci.*, **21**, S673 (1990).

30. Chen, D., D.Y.H. Pui, and S.L. Kaufman, Electrospraying of Conducting Liquids for Monodisperse Aerosol Generation in the 4 nm to 1.8 μm Diameter Range, *J. Aerosol Sci.* **26**, 963 (1995).

31. Choupeau, M., B. Prunet-Foch, *J. Electrostatics*, **22**, 135 (1989).

32. Tang, K., A. Gomez, *J. Colloid Interface Sci.*, **184**, 500 (1996).

33. Jones, A.R., and K.C. Thong, *J.Phys. D. Appl. Phys.* **4**, 1159 (1971).

34. Vaonnegut, B., and R.L. Neubauer, Production of monodispersed Liquid Particles by Electroatomization, *J. Colloid Sci.*, **7**, 616 (1952).

35. Rossell-Llompart, J., and J. Fernandez de la Mora, Generation of Monodispersed Droplets 0.3 to 4 mm in Diameter from Electrified Cone-Jets of Highly Conducting Liquids, *J. Aerosol Sci.*, 25, 1093 (1994).

36. Chen, D., and D.Y.H. Pui, *Aerosol Sci. and Tech.*, **27**, 367 (1997).

37. Loscertales, I.G., and Fernandez de la Mora, *J. Aerosol Sci.* **27**, 695 (1993).

38. Ganan-Calvo, A.M., The Size and Charge of Droplets in the Electrospray of Polar Liquids in Cone-Jet mode, and the Minimum Droplet Size, *J. Aerosol Sci.* **25S**, S309 (1994).

39. Cloupeau, M.B., B. Prunet-Foch, *J. Electrostatics* **25**, 165 (1990).

40. Cloupeau, M.B., B. Prunet-Foch, *J. Aerosol Sci.* **25,** 1021 (1994).

41. Hayati, I., A.I. Bailey, TH.F. Tadros, *J. Colloid Interface Sci.* **117**, 205 (1987).

42. Fernandez de la Mora, J., and I.G. Loscertales, *J. Fluid Mech.* **260**, 155 (1994).

43. Tang, K., and A. Gomez, *J. Aerosol Sci.* **25**, 1237 (1994).

44. Scott, T.C., *AIChE Journal,* **33,** 1557 (1987).

45. Wham, R.M., and C.H. Byers, *Sep. Sci. Tech.,* **22,** 447 (1987).

46. Scott, T.C., *Chem. Sep., I: Principles,* C.J. King, J.D. Navratil, Eds., Litarvan, Denver (1986).

47. Lamb, H., *Hydrodynamics,* 6th ed., Dover, N.Y., 475 (1945).

48. Tang, K., and A. Gomez, *J. Colliod Interface Sci.* **175**, 326 (1995).

49. Tang, K., " The electrospray: Fundamentals and Feasibility of Its Application to Targeted Drug Delivery by Inhalation." Ph.D. Dissertation, Yale University, 1994.

50. Smith, D.P.H., *IEEE Trans. Ind. Appl.* **IA-22**, 527 (1986).

51. Mutoh, M., S. Kaieda, and K. Kamimura, *J. Appl. Phys.* **50**, 3174 (1979).

52. Swatik, D.S., and C.D. Hendricks, *AIAA J.* **6**, 1596 (1968).

53. Mahoney, J.F., A.Y. Yahiku, H.L. Daley, R.D. Moore J. Perel, *J. Appl. Phys.* **40**, 5101 (1969).

54. Zeleny, J., *Proc. Cambridge Phil. Soc.* **18,** 71 (1915).

55. Zhang, X., and O.A. Basaran, *J. Fluid Mech.* **326,** 239 (1996).

56. Ganan-Calvo, A.M., *J. Fluid Mech.,* **335,** 165 (1997).

57. McHugh, M.A., and V.J. Krukonis, Supercritical Fluid Extraction: Principles and Practice, Butterworths; Boston, Mass, 1986.

58. Barton, A., Handbook of Solubility Parameters and Other Cohesive Parameters, CRC: Boca Raton, FL, 1983.

59. Wiebe, R., and V.L. Gaddy *J. Amer. Chem. Soc.,* **61,** 319 (1939); **62,** 315 (1940); **63,** 475 (1941).

60. Baker, L.C.W., and T.F. Anderson, *J. Amer. Chem. Soc.* **79**, 2071 (1957).

61. Kuk, M.S., and J.C. Montagna, Chemical Engineering at Supercritical Conditions, Ann Arbor Sci. Pub. Ch.4, 101 (1983).

62. Gilbert, M.L., and M.E. Paulaitis, *J. Chem. Eng. Data,* **31,** 296 (1986).

63. Takishima, S., K. Saiki, K. Arai, and S, Saito, *J. Chem. Eng. Japan,* **19,** 48 (1986).
64. Lim, J.S., Y.Y. Lee, and H.S. Chun, *J. Supercritical Fluids,* **7,** 219 (1994).

65. Furuta, S., N. Ikawa, R. Fukuzato, N. Imanishi, *Kagaku Kogaku Ronbushu,* (in Japanese, with English abstract), **15,** 519 (1989).

66. Furuta, S., N. Ikawa, and R. Fukuzato, "Extraction of Ethanol From Aqueous Solutions Using Compressed Carbon Dioxide", In: G. Vetter (Eds.), *2nd Int. Symp. on High Pressure Chem. Eng.* Erlangen (Germany) DECHEMA, 345 (1990).

67. Inomata, H., K. Arai, S. Saito, S. Ohba, and K. Takeuchi, *Fluid Phase Equilibria,* **53,** 23 (1989).

68. Inomata, H., A. Kondo, K. Arai, and S. Saito, *J. Chem. Eng. Japan,* **23,** 199 (1990).

69. Yoon, J.H., H. Lee, and B.H. Chung, *Fluid Phase Equilibria,* **102,** 287 (1994).

70. Hirohama, S., T. Takatsuka, S. Miyamoto, and T. Muto, *J. Chem. Eng. Japan,* **26,** 409 (1993).

71. Lahiere, R., and J. Fair, *J. Chem. Eng. Data,* **34,** 275 (1989).

72. Feng, Y.S., X.Y. Du, C.F. Li, and Y.J. Hou, *1st Int. Symp. Supercritical Fluids,* **1,** 75 (1988).

73. Nagahama, K., J. Suzuki, and T. Suzuki, *1st Int. Symp. Supercritical Fluids,* **1,** 143 (1988).

74. Martinez de la Ossa, E., V. Brandani, G. Del Re Gabriele, D. Giacomo, and E. Ferri, *Fluid Phase Equilibria,* **56,** 325 (1990).

75. Pan, A.M., Bachelor Eng. Thesis. Zheijiang University, Hangzhou, China (in Chinese).

76. Yao, S., Y. Guan, and Z. Zhu, *Fluid Phase Equilibria,* **99,** 249 (1994).

77. Hirohama, S., M. Ojiro, T. Takatuka, F. Kanegae, and T. Muto, *2nd Int. Symp. Supercritical Fluids,* 97 (1991).

78. Guvenc, A., U. Mehmetoglu, and A. Calimli, *4th Int. Symp. Supercritical Fluids,* **B,** 597 (1997).

79. Nagase, Y., T. Tada, N. Ikawa, and R. Fukuzato, *4th Int. Symp. Supercritical Fluids,* **B,** 617 (1997).

80. Hirohama, S., T. Takatsuka, S. Miyamoto, and T. Muto, *J. Chem. Eng. Japan,* **26,** 247 (1993).

81. Chun, B.S., and G.T. Wilkinson, *3rd Int. Symp. Supercritical Fluids,* **1,** (1994).

82. Wignal, G.D., Encyclopedia of Polymer Science and Engineering, M. Grayson and J. Kroschwitz, Eds., Wiley N.Y. (1987).

83. Magid, L.J., *Colloids and Surfaces,* **19,** 129 (1986).

84. Hendricks, R.W., *J. Appll. Phys.* **11,** 15 (1978).

85. Fulton, J.L., D.M. Pfund, and Y. Ma, *Rev. Sci. Intrum.,* **66,** 1 (1995).

86. Koehler, W.C., *Physica (Utrecht),* **137B,** 320 (1986).

87. Melnichenko, Y.B., E. Kiran, G.D. Wignal, K.D. Heath, S. Salaniwal, H.D. Cochran, and M. Stamm, *Science* in press (1999).

88. Wells, S. *et al., Amer. Chem. Soc. conf.,* March 1999.

89. DeSimone J., *et al., Amer. Chem. Soc. conf.,* March, 1999.

90. Peng, D., and D.B. Robinson, *Ind. Eng. Chem., Fundam.,* **15,** 59 (1976).

91. Shen, Z., F. Tan, J. Yang, and J. Wang, *1st Int. Symp. Supercritical Fluids,* **1,** 223 (1988).

APPENDICES

The following sections contain related work done as a part of this research in conjunction with the main project. This related work was tangential to the main focus, but supportive of the project overall objectives.

APPENDIX A

SMALL ANGLE NEUTRON AND X-RAY SCATTERING IN SCO_2

SAXS and SANS are important tools for studying solutions in SCO_2. They elucidate much information about the system including the size and shape of both single polymer chains (82), and supramolecular structures (83), and with a resolution in the range of 5-2000 A. The 10 m SAXS instrument (84) located at the Oak Ridge National Laboratory (ORNL) has been used in this research to study systems such as perfluoroalkanes (perfluorononane, perfluoropentadecane), perfluoroethers (perfluoroether carboxylic salt, perfluoroether monosulfate), and commercially available lubricants such as hexaflouropropylene oxide (HFPPO) (Dupont's Krytox). By utilizing a sample cell modeled after that of Fulton and coworkers (85), these solutions were studied in CO_2 at concentrations from 1-15 mol % solute with temperature and pressures ranging from 20 °C and 8.27 MPa to 40 °C and 20.68 MPa. Additional work included in this research was completed on the W. C. Koehler (86) 30m SANS facility also at ORNL. In fact, some of the experiments performed by SAXS, such as the perfluoroethers, were repeated by SANS for independent confirmation. Other systems studied by SANS include poly(dimethylsiloxane) (PDMS), and various block copolymers: poly(vinyl acetate)-b-poly(tetrahydroperfluoro octylmethacrylate) (PVAC-b-PTAN), poly(polystyrene/1-phenyl-1-(4-vinylphenyl)ethylene)-b-poly(vinylidene fluoride) (PS/PVPE-b-VF2), and poly(n,n-dimethyl amino ethyl methacrylate)-b-poly(1,1-dihydroperfluoro octylmethacrylate) (PDMAEMA-b-PFOMA). All these experiments

were performed above the critical temperature of CO_2 with pressures from 17.24 MPa to 68.95 MPa. More experimental detail can be found in the publications that have resulted from these experiments (87-89).

A.1 Paper 1

Y. B. Melnichenko, E. Kiran, G. D. Wignall, K. D. Heath, S. Salaniwal, H. D. Cochran, M. Stamm, "Pressure And Temperature-Induced Transitions In Solutions Of Poly(Dimethyl Siloxane) In Supercritical Carbon Dioxide," *Macromolecules*, in press (1999).

PRESSURE AND TEMPERATURE-INDUCED TRANSITIONS IN SOLUTIONS OF POLY(DIMETHYL SILOXANE) IN SUPERCRITICAL CARBON DIOXIDE

Y. B. Melnichenko[1], E. Kiran[2], G. D. Wignall[1], K. D. Heath[3], S. Salaniwal[3], H. D. Cochran[3], M. Stamm[4]

[1]*Oak Ridge National Laboratory*, Oak Ridge, TN 37831, USA*

[2]*Department of Chemical Engineering, University of Maine, Orono, ME 04469*

[3]*Chemical Technology Division, Oak Ridge National Laboratory and Department of Chemical Engineering, University of Tennessee, Knoxville, TN 37996*

[4]*Max Planck-Institut fur Polymerforschung, 55021 Mainz, Germany*

Supercritical fluids (SCFs) have great technological potential for minimizing the organic wastes associated with polymer manufacturing and processing. However, significant challenges remain for developing the same level of understanding of the behavior of polymers in SCFs as has been reached for polymers in traditional organic solvents. Small-angle neutron scattering was used to study the effect of pressure and temperature on the phase behavior of poly(dimethylsiloxane) (PDMS) in supercritical carbon dioxide (SC CO_2). It was demonstrated that PDMS-SC CO_2 solutions reproduce all main features of the temperature-concentration phase diagram for polymers in organic solvents. Moreover, because of their continuously adjustable solubility, SCFs exhibit novel effects, such as a pressure-induced transition to the Θ point and to the good solvent domain, in addition to a polymer-solvent demixing at a lower critical solution pressure.

* Managed by Lockheed Martin Energy Research Corporation under contract DE-AC05-96OR-22464 for the U. S. Department of Energy.

A supercritical fluid (SCF) is a substance at a pressure and temperature above the liquid-vapor critical point where the coexisting liquid and vapor phases become indistinguishable. The physical properties of SCFs are similar to those of dense gases, although when highly compressed, their density may be comparable to that of the sub-critical liquids. SCFs in general and supercritical carbon dioxide (SC CO_2) in particular have emerged as an attractive alternative to the organic solvents used for polymer manufacturing and processing. One key advantage of SCFs is the possibility of continuously tuning the solvent quality by varying the pressure (P) in addition to the temperature (T), which offers an unparalleled means for controlling polymer solubility. Despite recent progress in many specific areas of SCF science and technology (see e.g. [1,2] and references therein), significant challenges remain for developing the same level of understanding of the behavior of polymers in SCFs as has been reached for polymers in traditional organic solvents. Here we address the effect of temperature and pressure on the phase behavior of poly(dimethyl siloxane) (PDMS) in SC CO_2. We apply small-angle neutron scattering (SANS) to extract information on the concentration fluctuations and the dimensions of polymer chains at different conditions and find that the supercritical solutions reproduce all salient features of the temperature-concentration phase diagram for polymers in liquid solvents [3]. Moreover, because of their continuously adjustable properties, SCFs exhibit novel effects, such as a pressure-induced transition to the Θ point and to the good solvent domain, in addition to a polymer-solvent demixing at a lower critical solution pressure.

At the phenomenological level, liquid solvents for polymers are divided into three classes, namely poor, theta (Θ), and good solvents. The solvent quality is directly related

to the ability of the solvent molecules to mediate the attractive intrachain forces responsible for the polymer – solvent demixing. Poor solvents can hardly impede the intrachain interactions and hence can dissolve only short chain (or low molecular weight M_W) polymers with a limited number of contacts between the segments. The pairwise attractive and repulsive interactions compensate at the Flory or the "Θ temperature" (T_Θ) which is defined as the upper critical solution temperature (UCST) for a polymer with infinite molecular weight $M_W = \infty$ [4,5]. At $T=T_\Theta$ the radius of gyration $R_g(T_\Theta)$ of polymer chains is unperturbed by excluded volume effects and is also independent of the long-range critical concentration fluctuations. In the good solvent domain $T > T_\Theta$ the repulsive forces between the segments work to expand R_g above the unperturbed dimensions at $T=T_\Theta$.

The dimension of polymer chains R_g and the correlation length of the concentration fluctuations ξ in polymer solutions may be determined at different (P,T) using SANS combined with a high concentration isotope labeling method [6]. The coherent scattering intensity, *I*, for a mixture of identical hydrogenated and deuterated polymer chains in a solvent is:

$$I(Q,x) = K * S_S(Q) + L * S_t(Q) \quad , \qquad (1)$$

where $S_s(Q)$ is the single-chain structure factor which contains information on the dimensions (R_g) of the polymers. Similarly, $S_t(Q)$ is the total scattering structure factor which contains information on the correlation length of the concentration fluctuations, ξ.

The scattering vector is given by $Q = 4\pi\lambda^{-1}\sin\theta$, where 2θ is the scattering angle and λ is the neutron wavelength. The prefactor $K^* \sim (b_H - b_D)^2\, x(1-x)$ in Eq.(1) is a function of the neutron scattering lengths of the hydrogenated (b_H) and deuterated (b_D) monomers as well as on the mole fraction (x) of hydrogenated polymers. If all chains are hydrogenated (x=1), K^*=0, and one can obtain ξ(T,P) directly from $I(Q,x=1) \sim S_t(Q)$ using the Ornstein-Zernike formula

$$S_t(Q) = S(0)/(1+Q^2\xi^2) \quad . \qquad (2)$$

In addition to b_H and b_D, the prefactor $L^* \sim [b_H x + (1-x)b_D - b_S']^2$ in Eq.(1) depends on the scattering length b_S' of a solvent normalized via the ratio of the specific volumes of the monomer and the solvent molecule. For PDMS-CO_2 mixtures, b_H =0.086 10^{-12} cm, b_D = 6.33 10^{-12} cm, and b_S' =1.47 10^{-12} to 3.76 10^{-12} cm in the range between the critical (ρ_{CO2}=0.469 g/cm^3) and the liquid state densities (ρ_{CO2}=1.2 g/cm^3). For a given combination of the neutron scattering lengths ($b_H < b_S' < b_D$), it is possible to make the pre-factor L^* zero at any density of SC CO_2 by choosing an appropriate concentration (x) of hydrogenated PDMS (e.g. L^*=0 at x=0.512 and ρ_{CO2}=0.95 g/cm). In this case, Eq.(1) gives the single chain (intramolecular) scattering directly and the R_g may be obtained by fitting $I(Q)$ with the Debye function [7,8]:

$$S_s^D = (2/y^2)(y-1+e^{-y}) \quad , \quad y = Q^2 R_g^2 \quad . \tag{3}$$

For solutions of non-overlapping polymer chains [9] under the Θ condition, Eq.(3) yields in the limit of small Q [7,8]:

$$S(Q) = \frac{S(0)}{1+Q^2\xi^2(\Theta)} \quad , \quad \xi(\Theta) = R_g(\Theta)/\sqrt{3} \quad . \tag{4}$$

Thus, the SANS measurements of ξ vs. T (or P) may be used to determine the Θ temperature (or the Θ pressure) using Eq.(4) [10].

The experiments were performed on the 30-m SANS spectrometer at the Oak Ridge National Laboratory over Q-range of $0.005<Q/Å^{-1}<0.05$ (λ=4.75 Å). The data were radially averaged and converted to an absolute coherent cross section (in units of cm^{-1}) using procedures described elsewhere [11]. The structure factors $S_s(Q)$ and $S_t(Q)$ were obtained from Eq.(1) and used to calculate ξ and R_g at different thermodynamic conditions using Eqs. (2) and (3). Polymer samples of PDMS-h (M_W =22500, 47700, and 79900 of polydispersity $M_W/M_N \leq 1.03$) and PDMS-d (M_W =27600, $M_W/M_N \leq 1.11$) were synthesized and characterized at the Max Planck Institute for Polymer Research, Germany. PDMS-d (75600, $M_W/M_N \leq 1.3$) was purchased from Polymer Standards Service GmbH, Mainz, Germany. All solutions were prepared at the overlap concentration [9] (C*=0.1397, 0.0959, and 0.0741 g/ml for M_W 22500, 47700, and 79900, respectively). H-PDMS or isotopic mixtures of (h+d) PDMS at x=0.512 were

loaded into a stainless steel cylindrical cell which was fitted with optically polished sapphire windows virtually transparent to neutron radiation. The samples were pressurized at T=50 ^{O}C to 45.7 MPa (i.e. at ρ_{CO2}=0.95 g/cm^3) with CO_2 (SFC purity 99.99%, Matheson Gas products, Inc., USA) and stirred thoroughly until a transparent homogeneous solution was obtained. After completing the temperature scans ξ(T,P) or R_g(T,P) at constant density (ρ_{CO2}=0.95 g/cm^3), the variation of ξ(P,T=const) or R_g(P,T=const) was measured with pressure at isothermal conditions.

The temperature variation of ξ in PDMS - SC CO_2 solutions at constant density ρ_{CO2}=0.95 g/cm^3 is shown in Fig.1. As monitored by the intermolecular correlation length, the concentration fluctuations diverge ($\xi \Rightarrow \infty$) as T approaches the critical temperature of phase demixing T_C. The intermolecular correlations decline rapidly with the temperature, become comparable to the dimension of polymer chains at ($T-T_C$) ~ 12 - 16 ^{O}C and reach the theoretical value $\xi(\Theta)$ for each M_W (Eq.4) at $T=T_\Theta$ = 65±5 ^{O}C. To our knowledge, this is the first experimental observation of the temperature-induced transition to the Θ point in polymer solutions in SCFs. In the inset in Fig.1 we compare the temperature variation of the concentration fluctuations in supercritical CO_2 and in organic solvents [i.e. polystyrene (PS) in the Θ solvent cyclohexane-d (CH-d)]. The reduced correlation lengths $\xi/\xi(T_\Theta)$ for both systems fall on the same master curve when plotted vs. the reduced temperature $\tau' = \frac{T_\Theta}{T_\Theta - T_C}\frac{T-T_C}{T}$ which accounts for the temperature distance from both T_Θ and T_C. A typical variation of ξ vs. ($T - T_C$) in PDMS-SC CO_2 solutions is shown in Fig.2. As is seen, the critical index ν in a scaling

law for the correlation length $\xi \sim (T - T_c)^{-\nu}$ exhibits a sharp crossover from the mean-field value (ν=0.5) in the Θ region to the Ising model value (ν=0.630±0.001 [12]) in the critical region around T_C. Accordingly, the critical index γ=1.23±0.02 for the susceptibility (i.e. the osmotic compressibility) in $\chi \sim (T - T_c)^{-\gamma}$ at T⇒T_C for all solutions studied agrees within experimental errors with the Ising model value (1.239±0.002 [12]). The crossover takes place when $\xi \sim R_g$ and thus reproduces the main features of the crossover observed in solutions of PS in CH-d (see the inset in Fig.2) [13,14]. These observations delineate an intrinsic analogy between the temperature behavior of polymers in SCFs and in liquid Θ solvents in the vicinity of the UCST and show that polymer solutions in SCFs belong to the universality class of the Ising model.

The effect of pressure on the thermodynamic condition of PDMS – SC CO_2 solutions is illustrated in Fig.3. The concentration fluctuations diverge as P approaches the upper critical solution pressure (UCSP) P_C where the system exhibits pressure-induced phase demixing. The solvent quality increases rapidly with P in the range ($P-P_C$) ≤ 7 MPa and more gradually after ξ becomes comparable to the radius of gyration of polymer chains $R_g(\Theta)$=75 Å. At P_Θ = 52±4 MPa the correlation length becomes equal to the theoretical value $\xi(\Theta)$≅41 Å (Eq.4) which indicates that the Θ condition has been reached. This is the first experimental observation of the pressure-induced transition to the Θ point in polymer solutions in SCFs, the existence of which was predicted earlier in [15]. It is interesting that the effect of pressure may be quite opposite at less favorable thermodynamic conditions. If the experiment is performed at a temperature close to a UCST, ξ(P) goes through the minimum after which the lower critical solution pressure

(LCSP) is approached (see the inset in Fig.3). The best solvent quality is reached close to the theta pressure P_Θ=52 MPa, though the correlation length $\xi(P_\Theta)\cong$74 Å is still much larger than the dimension of polymer chains in the solution $R_g(\Theta)$ =58 Å. This indicates that pressure alone may not always be capable of moving supercritical polymer solutions out of the poor solvent domain. The situation is similar to polymer solutions in poor organic solvents (e.g. PS in acetone) where the intermolecular correlations never decrease below the dimension of the constituent polymer [10]. We are not aware of previously published data on the pressure-induced demixing upon an increase in pressure in supercritical polymer solutions and believe this result to be the first experimental observation of the LCSP.

Fig.4 illustrates the behavior of the radius of gyration of PDMS in SC CO_2 with P and T. As is seen, R_g of the polymer remains invariant during both pressure and temperature quenches which extend to the immediate vicinity of the (T,P) polymer-SCF demixing locus and agrees well with the dimension of unperturbed chains with ($R_g = 0.267\ M_w^{1/2} \cong 40$ Å [16]). This observation indicates the universality of the constancy of R_g in semidilute polymer solutions below T_Θ and P_Θ which was previously demonstrated for liquid solvents [10,17] and has now been shown to extend to supercritical fluids. At $P>P_\Theta$ and/ or $T>T_\Theta$ the dimensions of polymer chains increase due to the excluded volume effects, as was observed in PS- CH-d solutions at $T>T_\Theta$ [18], indicating that SC CO_2 is becoming a good solvent for PDMS in this range of the thermodynamic parameters.

In conclusion, the experiments demonstrate universality between the structure and thermodynamic properties of the liquid and supercritical polymer solutions. In the latter

case, the threshold of unlimited miscibility (the Θ point) may be reached by varying either the pressure or the temperature. The experimental data agree with the results of Monte Carlo simulations which indicate that polymer chains may adopt unperturbed and expanded conformations at high densities [19]. This provides a foundation for a better understanding the underlying physics of SCF – polymer assemblies.

Stimulating discussions with M. Muthukumar and T. Wagner are greatly acknowledged. The research was supported by the Divisions of Advanced Energy Projects and Materials Sciences, U. S. Department of Energy under contract No. DE-AC05-96OR-22464 (with Lockheed Martin Energy Research Corporation). KDH and HDC were supported by the Division of Chemical Sciences of the U. S. Department of Energy. The financial support of YBM by Alexander von Humboldt-Stiftung is greatfully acknowledged. SS was supported by the National Science Foundation under grant number CTS-9613555 at the University of Tennessee.

REFERENCES

1. Kiran, E.; Levelt Sengers, J. M. H. Eds. *Supercritical Fluids: Fundamentals for Application;* Kluwer Acad. Publ.: Dodrecht, Boston, London, 1993.
2. McHugh, M. A.; Krukonis, V. J. *Supercritical fluid extraction: principles and practice;* Butterworth-Heinemann: Boston, 1994.
3. Daoud, M.; Jannink, G.; *J. Phys. (Paris)* **1976,** *37*, 973.
4. Sanchez, I. C. *Encyclopedia of Physical Science and Technology;* Academic Press: New York, 1992, vol. 13, pp.153-170.
5. The Θ condition may be viewed as corresponding to the threshold of unlimited polymer – solvent miscibility in the sense that a polymer of an arbitrary M_W becomes miscible in any proportion with the given solvent.
6. Williams, C. E.; Nierlich, M.; Cotton, J. P.; Jannink, G.; Boue, F.; Daoud, M; Farnoux, B., Picot, C.; de Gennes, P.-G.; Rinaudo, M.; Moan, M.; Wolf, C. *J. Polym. Sci., Polym. Lett. Ed.* **1979**, *17*, 379; Akasu, A. Z.; Summerfield, G. C.; Jahansan, S. N.; Han C. C.; Kim, C. Y.; Yu, H. J. *Polym. Sci., Polym. Phys. Ed.* **1980**, *18*, 863; King, J. S.; Boyer, W.; Wignall, G. D.; Ullman, R. *Macromolecules* **1985**, *18*, 709.
7. Des Cloizeaux, J. Jannink, G. *Polymers in Solution. Their Modeling and Structure;* Claredon Press: Oxford, 1990.
8. Fujita, H. *Polymer Solutions;* Elsevier: Amsterdam, Oxford, NY, Tokyo, 1990.
9. The concentration C* at which polymer chains begin to overlap may be defined as $C^* = 3M_W / 4\pi R_g^3 N_A$, where N_A is the Avogadro number [7,8]. The C* thus

defined provides the closest proximity to the critical concentration of polymer solutions in organic solvents in the range $M_W \leq 80000$ [Melnichenko, Wignall, to be published].

10. The validity of Eq.(4) was verified using SANS for solutions of polystyrene in deuterocyclohexane at the Θ condition over a wide range of pressures 0.1 – 50 MPa in [Melnichenko, Y. B.; Wignall., G. D.; Van Hook, W. A.; Szydlowsky, J.; Wilczura H.; Rebelo, L. P. *Macromolecules* **1998**, *31*, 8436].
11. Wignall G. D.; Bates, F. S. *J. Appl. Crystallogr.* **1986**, *20*, 28.
12. Le Guillou J. C.; Zinn-Justin, J. *Phys.Rev.* **1980**, *B 21*, 3976.
13. Melnichenko Y. B. et al. *Europhys.Lett.* **1992**, *19*, 355.
14. Melnichenko, Y. B.; Anisimov, M. A.; Povodyrev, A. A.; Wignall, G. D.; Sengers J. V.; Van Hook, W. A. *Phys. Rev. Lett.*, **1997**, *79*, 5266.
15. Kiran E.; Sen, Y. L. in: *Supercritical Fluid Engineering Science,* Kiran, E.; Brenneke, J. F. Eds., *ACS Symposium Series*, 1993, ch. 9.
16. Mark, J. E. Ed. *Physical Properties of Polymers Handbook;* AIP Press: Woodbury, NY 1996.
17. Melnichenko Y. B.; Wignall, G. D. *Phys. Rev. Lett.* **1997**, *78*, 686.
18. Cotton., J. P.; Nierlich, M.; Boue, F.; Daoud, M.; Farnoux, B.; Jannink, G.; Duplessix, R.; Picot, C. *J. Chem. Phys.*, **1976**, *65*, 1101.
19. Luna-Barcenas, G.; Meredith, J. C.; Sanchez, I. C.; Johnson, K. P.; Gromov, D. G.; de Pablo, J. J., *J. Chem. Phys.* **1997**, *107*, 10782; Gromov,; D. G.; de Pablo, J. J.; Luna-Barcenas, G.; Sanchez, I. C.; Johnson, K. P. *J. Chem. Phys.* **1998**, *108*, 4647.

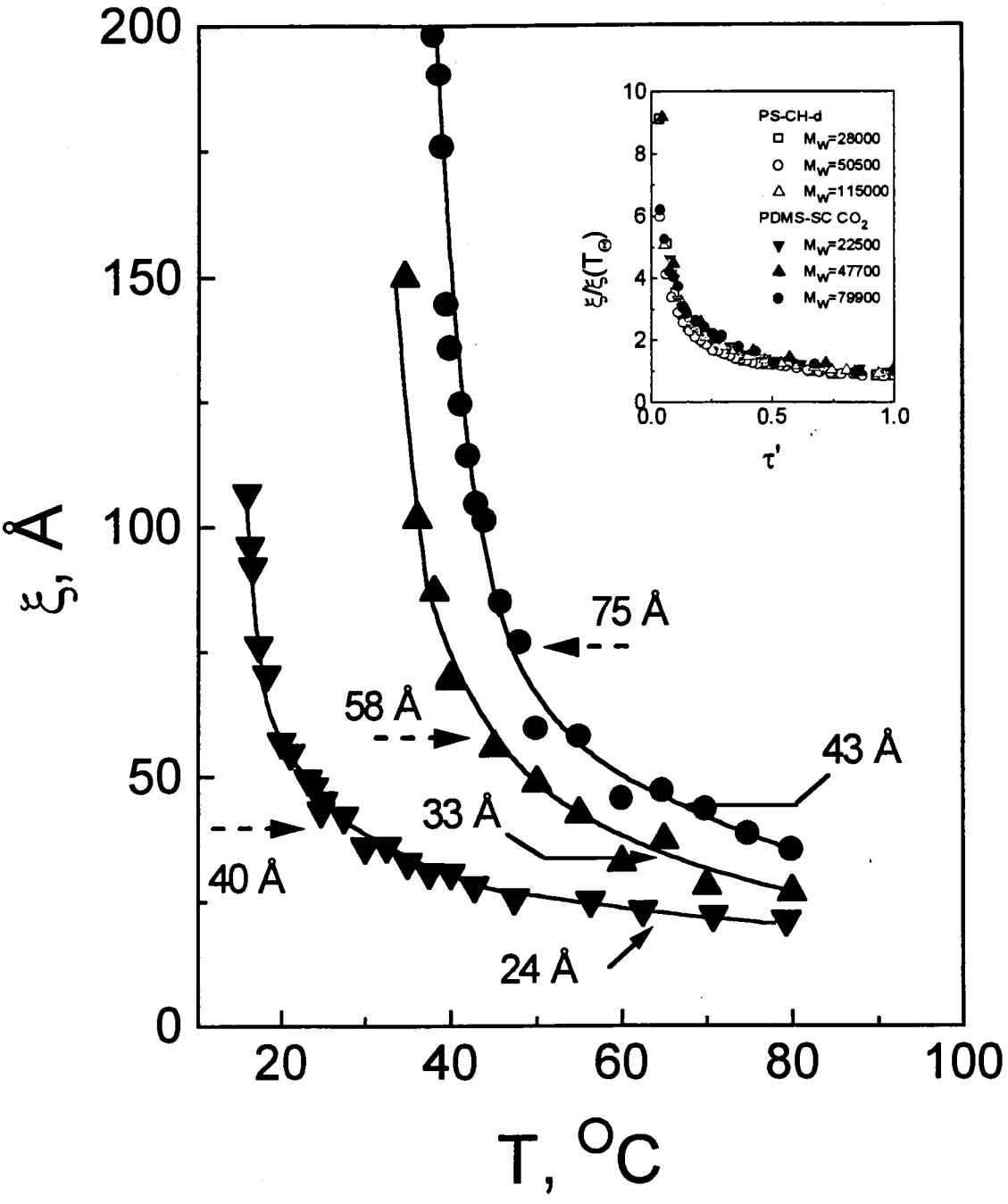
LEGENDS:

Fig.1. $\xi(T)$ for h-PDMS with M_W=22500 (▼), 47700 (▲), and 79900 (•). ρ_{CO2}=0.95 g/cm^3. Dashed and solid arrows show the value of $R_g(\Theta)=0.267\ M_w^{1/2}$ [16] and $\xi(\Theta)=R_g(\Theta)/\sqrt{3}$ (Eq.4), respectively. The inset shows the variation of the reduced correlation length vs. the reduced temperature $\tau'=\frac{T_\Theta}{T_\Theta - T_c}\frac{T-T_c}{T}$ for solutions of PS in CH-d [17] and PDMS in SC CO_2. In the latter case, the parameters used for calculations are: T_Θ=65 ^oC, T_C = 12.2 ^oC (M_W=22500), T_C = 31.8 ^oC (M_W=47700), and T_C = 36.1 ^oC (M_W=79900).

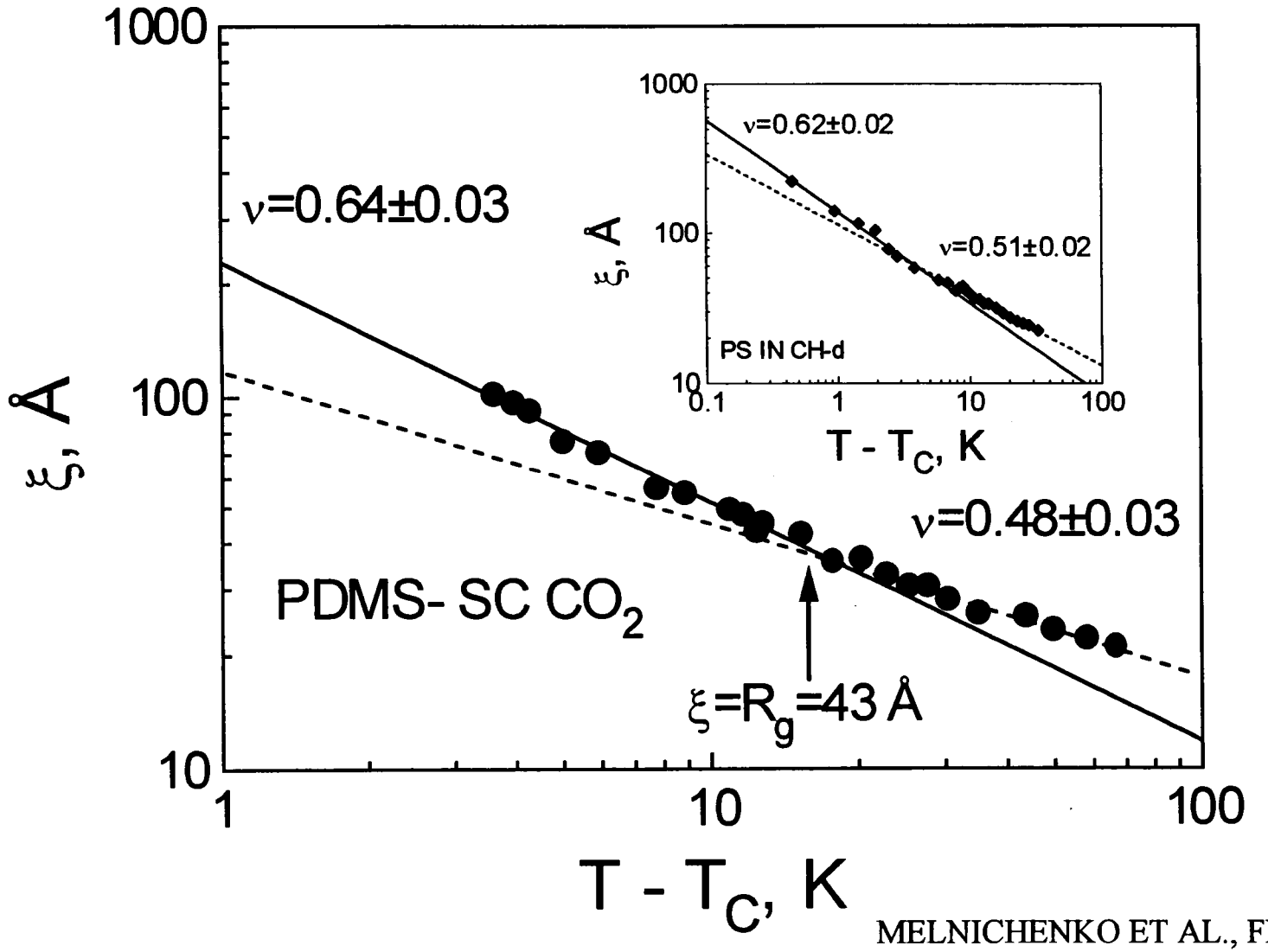
Fig.2. Variation of ξ as a function of ($T-T_C$) for solution of PDMS (M_W =22500) in SC CO_2. ρ_{CO2}=0.95 g/cm^3. The slope gives the value of the critical index ν. The inset shows ξ vs. ($T-T_C$) for PS – CH-d solution with a similar M_W =28000 of the polymer.

Fig.3. Variation of the correlation length ξ as a function of pressure for PDMS (M_W =79900) in SC CO_2 at T=50 ^oC. Dashed and solid arrows show the value of $R_g(\Theta)$ and $\xi(\Theta)$ (Eq.4), respectively. The inset shows $\xi(P)$ for solution of PDMS (M_W =47700) in SC CO_2 at T=32 ^oC close to the UCST for this solution (T_C =31.8 ^oC). The extrapolation of $\xi^{-1}(P)$ to zero gives $P_C \cong 32$ MPa and $P_C \cong 90$ MPa for the UCSP and the LCSP, respectively.

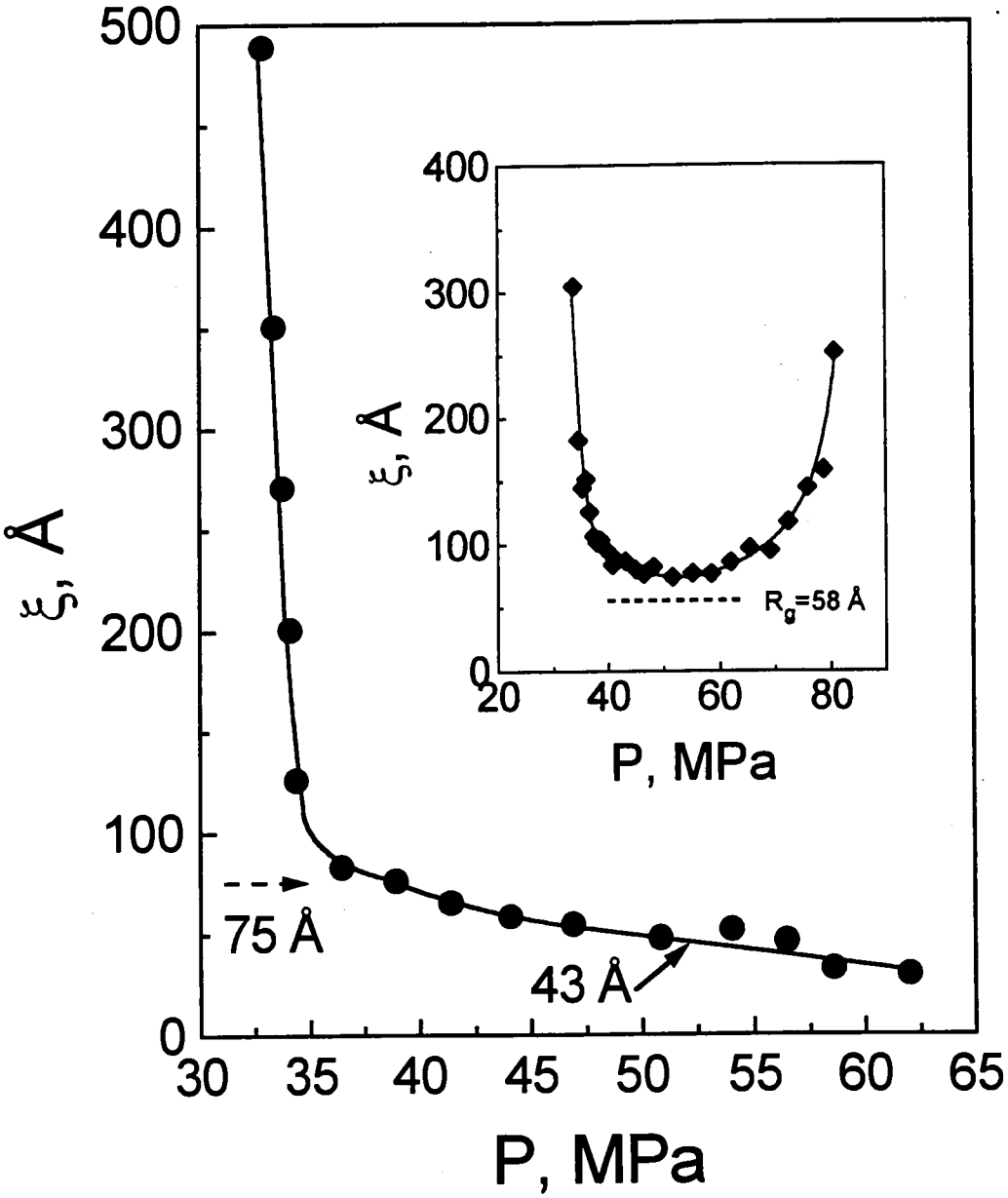
Fig.4. Variation of the radius of gyration for PDMS with M_W=22500 in solution of (h+d) PDMS in SC CO_2 vs. pressure at T=70 ^oC. The inset shows R_g as a function of temperature at ρ_{CO2}=0.95 g/cm^3.



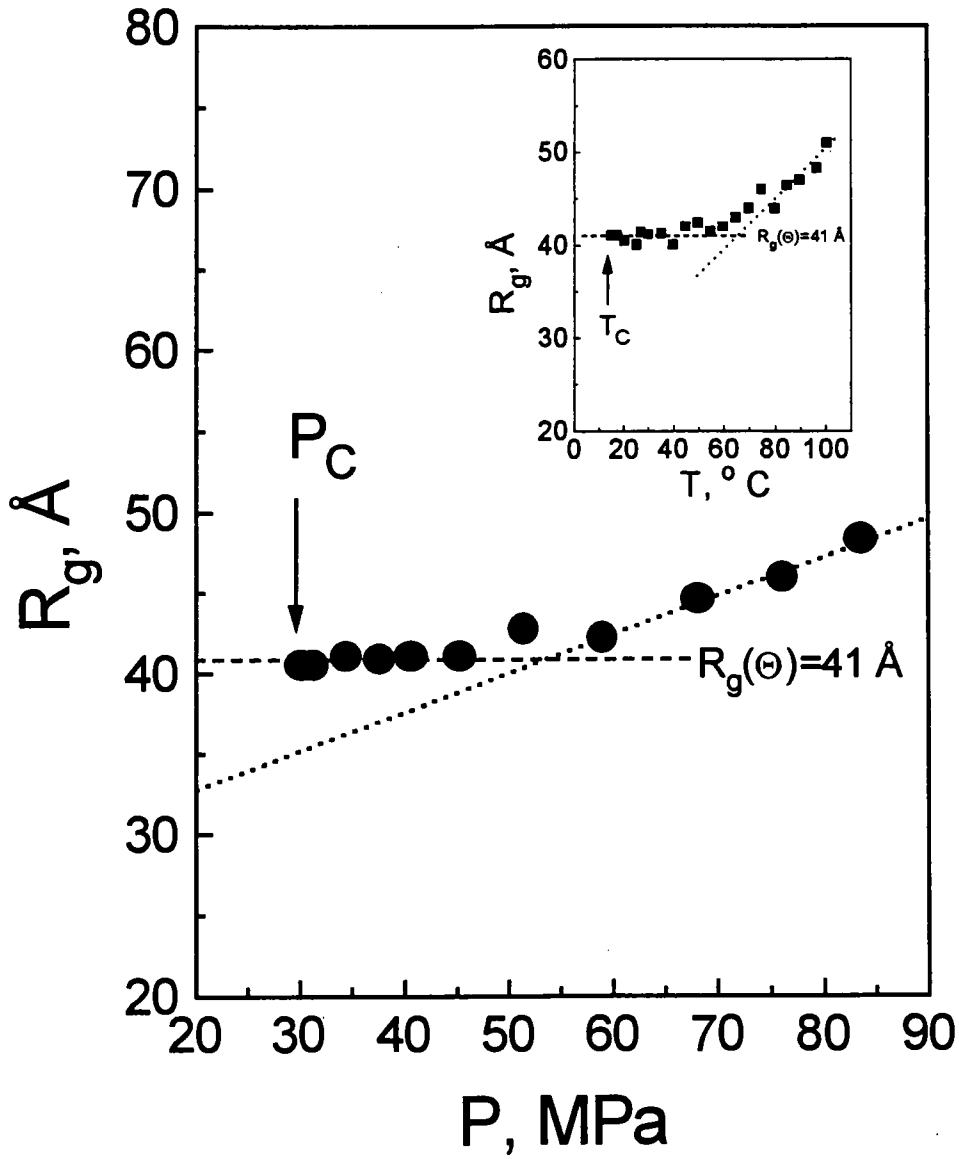
MELNICHENKO ET AL., FIG. 1



MELNICHENKO ET AL., FIG. 2



MELNICHENKO ET AL., FIG. 3



MELNICHENKO ET AL. FIG. 4

A.2 Paper 2

Y. B. Melnichenko, E. Kiran, K. D. Heath, S. Salaniwal, H. D. Cochran, M. Stamm, W. A. Van Hook, G. D. Wignall, "SANS studies of Polymers in Organic Solvents and Supercritical Fluids in the Poor, Theta and Good Solvent Domains," *ACS Symp. Ser. 216*, in press (1999).

SANS Studies of Polymers in Organic Solvents and Supercritical Fluids in the Poor, Theta and Good Solvent Domains

Y. B. Melnichenko[1], E. Kiran[2], K. Heath[1], S. Salaniwal[1], H. D. Cochran[1],M. Stamm[3], W. A. Van Hook[4] and G. D. Wignall[1]

[1]Solid State Division, Oak Ridge National Laboratory*, Oak Ridge, TN 37831
[2]Department of Chemical Engineering, University of Maine, Orono, ME 04468
[3]Max Planck Institut für Polymerforschung, 55021 Mainz, Germany
[4]Chemistry Department, University of Tennessee, Knoxville TN 37996

We demonstrate that semidilute polymer solutions in supercritical fluids (SCFs) reproduce all main features of the polymer behavior in organic solvents which is indicative of the intrinsic similarity between the thermodynamic properties of polymers in SCFs and in the far sub-critical liquids. Using small-angle neutron scattering, we studied the effect of temperature and pressure on the phase behavior of polystyrene (PS) in organic solvents as well as of poly(dimethylsiloxane) (PDMS) in supercritical carbon dioxide (SC CO_2). The radius of gyration R_g of polymer chains in both organic solvents and in SC CO_2 is invariant during both temperature and pressure quenches down to the critical point of demixing. The limit of infinite polymer miscibility (the Θ condition) of PDMS - SC CO_2 solutions may be reached by varying the pressure (P_Θ=52±4 MPa at the density of SC CO_2 ρ_{CO2} = 0.95 g/cm^3) and or the temperature (T_Θ = 65±5 ^oC). A sharp crossover between the critical and mean field behavior in PDMS – SC CO_2 is similar to that in solutions of PS in cyclohexane, and at $T > T_\Theta$, $P > P_\Theta$ the solutions reach the good solvent domain where R_g expands beyond the unperturbed dimensions $R_g(\Theta)$ at the Θ condition.

* Managed by Lockheed Martin Energy Research Corporation under contract DE-AC05-96OR-22464 for the U. S. Department of Energy.

Introduction

In organic solvents, it is well known that the radius of gyration, R_g (i.e. the r.m.s. distance of scattering elements from the center of gravity) of polymer molecules depends on the sign and magnitude of the interactions between the chain segments and the molecules of the surrounding liquid. In "good" solvents, the dominating repulsive forces between the segments (excluded volume effects) work to expand the R_g, and the second virial coefficient (A_2) is positive. In less favorable solvents, the pairwise attractive and repulsive segment interactions may compensate at the "Flory" or "theta temperature" (T_Θ), where $A_2=0$, and R_g corresponds to the dimension of a volume-less polymer coils, "unperturbed" by the excluded volume effects. By definition, T_Θ is the critical solution temperature (T_C) for a polymer with the molecular weight $M_W = \infty$ and thus corresponds to the threshold of unlimited polymer – solvent miscibility. One of the first significant applications of SANS was to confirm Flory's prediction that polymer chains would adopt such random-walk configurations in the condensed (amorphous) state [1].

In the poor solvent regime ($T<T_\Theta$, $A_2<0$), dominating attractive interactions between the segments work to collapse polymer chains into compact polymer globules. This phenomenon is well understood for dilute solutions of separated polymers which collapse as the temperature approaches T_C [2]. SANS has recently been used to extend the experimental observations to "semidilute" solutions of strongly interacting macromolecules. In accord with the de Gennes' concept [3], in the critical region $T \sim T_C$ of the solutions polymer chains do not interpenetrate significantly and thus should be collapsed [i.e. $R_g(T_C) < R_g(T_\Theta)$] as in dilute concentration regime. However, experiments on PS in cyclohexane (CH) [4,5] and acetone (AC) [6] have demonstrated that the predicted decrease in R_g is not observed as $T \Rightarrow T_C$. Instead, diverging concentration fluctuations near T_C lead to the formation of distinct microdomains of strongly interpenetrating molecules, which prevent the expected collapse [4-6]. Measurements have also been made in supercritical fluids to contrast such systems with organic solvents and to test the prediction [7] that the molecules will adopt the unperturbed dimensions at a critical "theta pressure"(P_Θ) as they do in polymer solutions at the theta temperature. A direct comparison of the phase behavior of PDMS in SC CO_2 with that of PS in CH and in AC is indicative of the intrinsic similarity between the structure and thermodynamic properties of polymers in SCFs and in sub-critical organic liquids.

Experimental

The SANS data were collected on the W. C. Koehler 30m SANS facility [8] at Oak Ridge National Laboratory (ORNL). The neutron wavelength was λ = 4.75 A ($\Delta\lambda/\lambda \sim 5\%$) and the 64 x 64 cm^2 area detector with cell size ~ 1 cm^2 was placed at various sample-detector distances to give an overall range of momentum transfer of $0.005 < Q = 4\pi \lambda^{-1} \sin\theta < 0.1$ A^{-1}, where 2θ is the angle of scattering. The data were corrected for instrumental backgrounds and detector efficiency on a cell-by-cell basis, prior to radial (azimuthal) averaging and the net intensities were converted to an

absolute (± 3%) differential cross section per unit solid angle, per unit sample volume [$d\Sigma(Q)/d\Omega$ in units of cm^{-1}] by comparison with pre-calibrated secondary standards [9]. The experiments in supercritical CO_2 were conducted in a high-pressure cell similar to the one used previously for polymer synthesis [10] and the SANS studies of block copolymer amphiphiles in supercritical CO_2 [11]. The cell's stainless body was fitted with sapphire windows with virtually no attenuation (cell transmission ~ 93%) or parasitic scattering. The sample cross sections were obtained after subtracting the intensities of the cell, and the signal from the CO_2 amounted to a virtually flat background (~ 0.04 cm^{-1}), which formed only a minor correction to the scattering from the homopolymer solutions.

To obtain R_g and the correlation length (ξ) of the concentration fluctuations we make use of the SANS high concentration isotope labeling method [12-14], which allows the determination of both parameters for semidilute polymer solutions. The coherent scattering cross section ($d\Sigma/d\Omega$ in units of cm^{-1}) of an incompressible mixture of identical protonated and deuterated polymer chains dissolved in a solvent is given by [4,5,14]:

$$I(Q,x) = I_S(Q,x) + I_t(Q,x) \quad , \tag{1}$$

$$I_S(Q,x) = KnN^2 S_S(Q) \quad , \tag{2}$$

$$I_t(Q,x) = LnN^2 S_t(Q) \quad . \tag{3}$$

The subscripts "s" and "t" correspond to scattering from a single chain and total scattering, respectively and thus $S_s(Q)$ is the single-chain structure factor, containing information on the intramolecular correlations (and hence R_g). Similarly, the total scattering structure factor, $S_t(Q)$ embodies information on the total (both intra- and intermolecular) correlations between monomer units and is related to the correlation length of the concentration fluctuations, ξ. The structure factors are normalized so that $S_s = 1$ at ($Q = 0$) and $S_t = S_s$ at infinite dilution. Also, x is the mole fraction of protonated chains in the solvent, and n and N are the number density of the polymer molecules and the degree of polymerization, respectively. The prefactors K and L are:

$$K = (b_H - b_D)^2 x(1-x); \qquad L = [b_H x + (1-x)b_D - b_S']^2 \quad , \tag{4}$$

where b_H and b_D are the scattering lengths of the protonated and deuterated monomers and b_S' is the scattering length of a solvent molecule, normalized to the ratio of specific volumes of the monomer and the solvent molecule.

The prefactor, L, in equation (4) controls the "total" scattering contribution and it has been shown [4] that for isotopic PS mixtures dissolved in fully deuterated acetone (AC-d), $L = 0$ at $x = 0.214$. Similarly for PDMS in CO_2, an isotopic ratio of $x = 0.512$ gives $L = 0$ at the density of the solvent ρ_{CO2}=0.95 g/cm^3. Thus, the intramolecular

scattering function may be obtained directly from the measured cross section at all temperatures and equation (1) gives the R_g directly [6]. For PS in CH-d, however, there is no isotopic ratio, $0 < x < 1$, which satisfies the condition $L(x) = 0$, and $d\Sigma/d\Omega(Q,x)$ always contains a minor ($\leq 10\%$) contribution from the total (intermolecular) scattering, which must be subtracted to extract R_g [6].

If all chains are all protonated (x=1), the prefactor, $K = 0$, and $d\Sigma/d\Omega \sim S_t(Q)$. Thus, the size of the concentration fluctuations may be measured via the Ornstein-Zernike (O-Z) formalism [9]: $d\Sigma/d\Omega(Q) = d\Sigma/d\Omega(0)/(1+Q^2\xi^2)$, where ξ is the composition fluctuation correlation length, which may be obtained from the slope of an O-Z plot of $[d\Sigma/d\Omega(0)]^{-1}$ vs. Q^2.

Samples of PS-h, defined by their weight (w) and number (n) averaged molecular weights (M_W = 10,200, 11,600 and 533,000 Dalton) and PS-d (M_W = 10,500, 11,200, and 520,000 Dalton) of polydispersity $M_W/M_n \leq 1.06$ were purchased from Polymer Laboratories. Deuterated solvents AC-d and CH-d (D/(H+D)=0.995) were purchased from Sigma Chemical and were dried over a molecular sieve prior to preparing solutions near the critical concentration C(PS/CH-d)=4.4 wt% [4] and C(PS/AC-d)=20.3wt% [15]. Solutions of (h+d) PS in AC-d and CH-d were prepared at x=0.214 and x=0.20, respectively. The latter condition provides a reasonably high signal-to-noise ratio, which minimizes the contribution from the intermolecular "total scattering" term in Eq.1.

Samples of protonated PDMS-h with M_W = 22,500 47,700, and 79,900 and polydispersity $M_W/M_n \leq 1.03$ were synthesized and characterized at Max Planck Institut für Polymerforschung, Germany, and PDMS-d (M_w = 27600 and 75600, $M_w/M_n \leq 1.11$) was purchased from Polymer Standards Service GmbH, Mainz, Germany. All solutions were prepared at the volume fraction of the polymer equal to the concentration of polymer coil overlap (ϕ^* = 0.1372, 0.0941, and 0.0727 for M_w = 22,500 47,700, and 79,900, respectively) which is practically indistinguishable from the critical concentration of phase demixing [3].

Samples were either run at atmospheric pressure in quartz cells as a function of temperature (e.g. PS-h in CH-d [4,5] or in AC-d [6]) or were loaded into the pressure cell with the appropriate isotopic ratio to eliminate (x = 0.512 for PDMS in CO_2) or minimize (e.g. x = 0.2 for PS-h in CH-d) the contribution of the total scattering term [Eq.(1)]. The maximum area accessible to the neutron beam is ~ 2 cm^2 and the path lengths of the quartz or pressure cells may be adjusted over the range, 0.1 to 2.0 cm. to optimize the transmission. The temperature was controlled by circulating fluids ($\pm$ 0.1°K) and pressure was applied and measured using a screw-type pressure generator and a precision digital pressure indicator, respectively.

Results and Discussion

Fig.1 shows the temperature variation of the correlation length ξ for PS-h in CH-d at the critical concentration of the polymer and it may be seen that the chains do not collapse as $T \Rightarrow T_C$ as observed in dilute solutions [2]. Instead, they maintain their

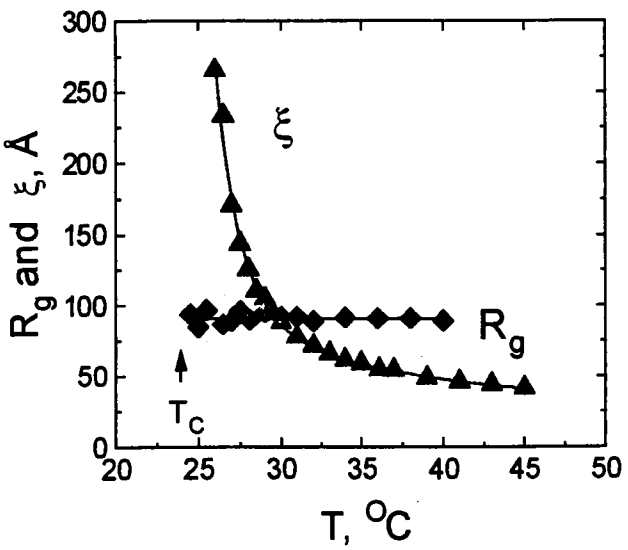


Figure 1. $R_g(T)$ and $\xi(T)$ for the solution of PS with M_W =115,000 in CH-d between T_Θ = 40 °C and the critical temperature of phase demixing (T_C).

unperturbed dimensions in accordance with the theoretical predictions of Muthukumar [16] and Raos and Alegra [17]. The size of the concentration fluctuations is very small in the Θ region, but increases dramatically near the critical point where it exceeds the chain dimensions [$\xi >> R_g(T_\Theta)$]. These findings indicate that critical polymer solutions can no longer be considered as an ensemble of collapsed, non-interpenetrating chains. Instead, the diverging thermodynamic fluctuations in the critical region lead to the formation of distinct microdomains, representing unperturbed, strongly interpenetrating macromolecules.

Fig. 2 shows values of ξ and R_g over a wide range of pressure and temperature extending from the Θ domain to the poor solvent condition near T_C. It may be seen that all isobars for $\xi(T)$ merge at the theta temperature, Θ=40 °C, where the average correlation length $\xi(\Theta) = 109 \pm 5$ Å. According to Fujita [18] and Des Cloizeaux and Jannink [19], $\xi(\Theta)=R_g(\Theta)/3^{1/2}$ at the theta condition, so $\xi(\Theta)$ = 114 Å for M_W = 533000 which agrees well with the experimental value.

Unlike the PS-CH system, which may undergo the transition from the poor solvent to the theta solvent by adjusting the temperature, the solvent quality is much poorer for PS - AC. Fig.3 shows the effect of pressure on the thermodynamic state of PS/AC-d solutions and the chain dimensions [$R_g \cong 29$ Å] are independent of pressure and temperature and remain close to the unperturbed dimensions for Gaussian chains with M_W = 11,600 [i.e. $R_g = 0.27\ M_W^{1/2} \sim 29$ Å]. Conversely, the concentration

fluctuations as monitored by the correlation length, ξ, diverge near the critical point and fall when $P >> P_C$. However, they do not decay sufficiently to reach $\xi(\Theta)$ =

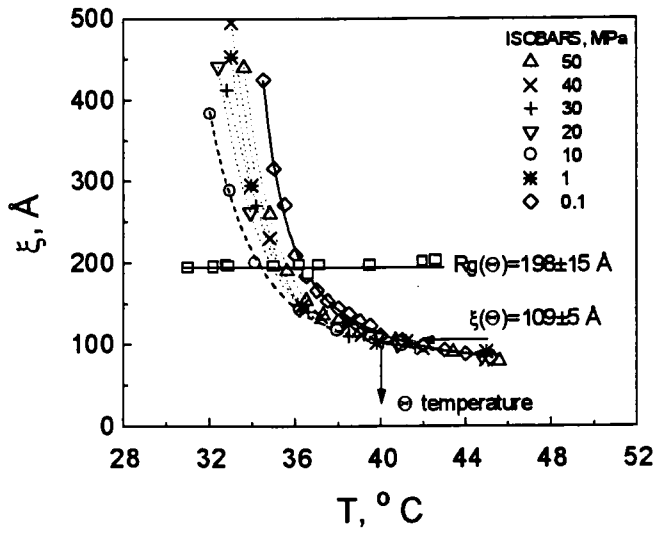


Figure 2. $\xi(T)$ for PS-h (M_W = 533000) in CH-d at different pressures shown in the inset. $R_g(P,T)$ of PS-h chains in the solution (h+d) PS in CH-d is also shown (open squares).

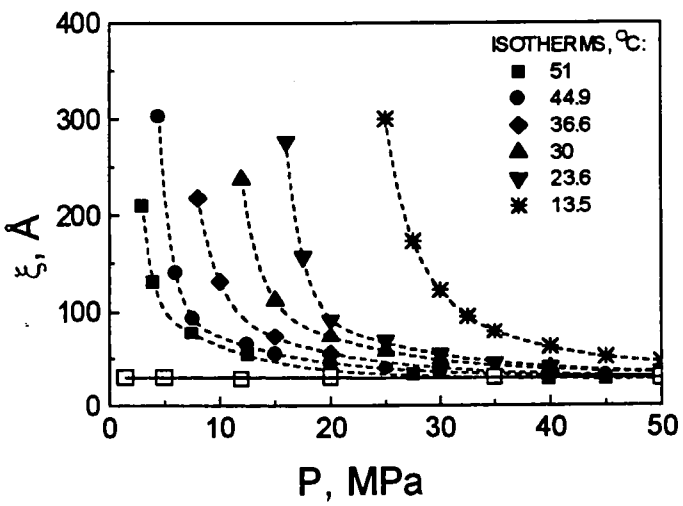


Figure 3. $\xi(P)$ of PS-h (M_W = 11,600) in AC-d at the different temperatures shown in the inset. $R_g(P,T)$ of labeled PS-h chains in the solution of (h+d) PS in AC-d is also plotted (open squares).

$R_g(\Theta)/3^{1/2} \cong 17$ Å, and thus, the system never leaves the poor solvent domain. Despite this, the macromolecules always exhibit their unperturbed (theta) dimensions, even though they never reach the theta domain. It is remarkable that the diverging concentration fluctuations which prevent chain collapse in PS-CH, also operate in PS-AC. So effective is this mechanism, that even for solutions which never leave the poor solvent domain, the macromolecules are always "stabilized" to exhibit the theta dimensions over wide ranges of pressure and temperature (Fig. (3).

The SANS experiments on PDMS in CO_2 were designed to compare the chain dimensions with those in organic solvents and to test the prediction [7] that they will adopt "ideal" configurations, unperturbed by excluded volume effects, at a critical "theta pressure"(P_Θ) as they do in polymer solutions at the theta temperature. The effect of temperature on the thermodynamic state of PDMS – CO_2 solution is illustrated in Fig. 4. The concentration fluctuations diverge as the temperature approaches the phase boundaries, which are themselves a function of pressure. When the correlation length decreases to the theoretical value $\xi(\Theta) = R_g(\Theta)/3^{1/2} \cong 41$ Å, the Θ condition is reached and this demonstrates that, unlike PS-AC, the solvent quality may be changed from the "poor" to theta solvent by varying the temperature, as in PS-

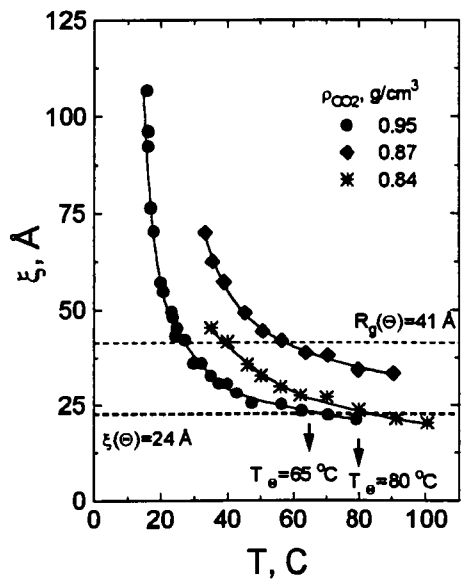


Figure 4. $\xi(T)$ of PDMS in SC CO_2 at different density of the supercritical solvent shown in the inset. The value of the Θ temperature depends crucially on the density of SC CO_2, e.g. $T_\Theta = 65 \pm 5\ ^oC$ at $\rho_{CO2} = 0.95\ g/cm^3$, however $T_\Theta = 80 \pm 5\ ^oC$ at $\rho_{CO2} = 0.87\ g/cm^3$. The Θ temperature may hardly be reached at $\rho_{CO2} \leq 0.84\ g/cm^3$.

CH solutions. A typical variation of ξ vs. $(T - T_C)$ in PDMS-SC CO_2 solutions is shown in Fig.5. As is seen, the critical index ν in a scaling law for the correlation length $\xi \sim (T - T_C)^{-\nu}$ exhibits a sharp crossover from the mean-field value (ν=0.5) in the Θ region to the Ising model value (ν=1.24) in the critical region around T_C. The crossover takes place when the correlation length becomes equal to the radius of gyration of the polymer and thus reproduces the main features of the crossover of ξ observed in solutions of PS in CH-d (see the inset in Fig.5) [4,5]. These observations delineate an intrinsic analogy between the temperature behavior of polymers in SCFs and in organic solvents [20] and show that polymer solutions in SCFs belong to the universality class of Ising model.

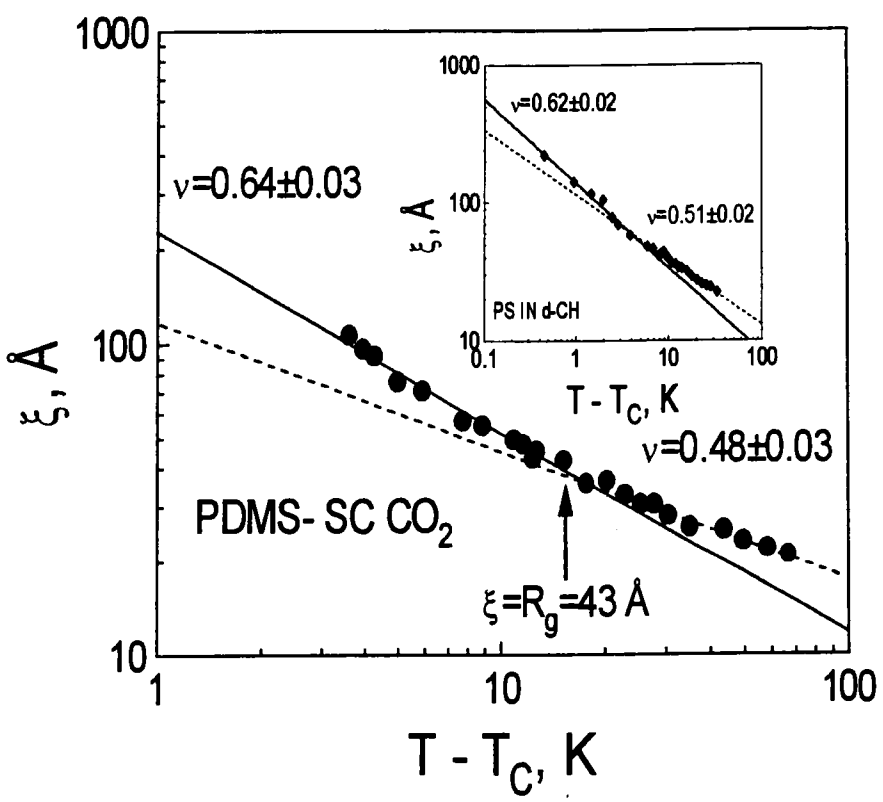


Figure 5. Variation of ξ as a function of $(T-T_C)$ for solution of PDMS (M_W =22500) in SC CO_2. The slope gives the value of the critical index ν. The inset shows ξ vs. $(T-T_C)$ for a solution of PS with M_W =28000 in CH-d [4,5].

Fig.6 shows that PDMS may also enter the "good solvent" domain at a "theta" temperature T_Θ = 65 ±5 °C, as observed in PS-CH solutions [21]. However, unlike organic solvents, this transition can also be made to occur at a critical "theta pressure" (P_Θ = 52 ± 4 MPa) in CO_2, as illustrated in the inset in Fig.6. To our knowledge, this is the first time that the existence of a "theta pressure" has been demonstrated experimentally and a more detailed description of this phenomenon is given in reference [22].

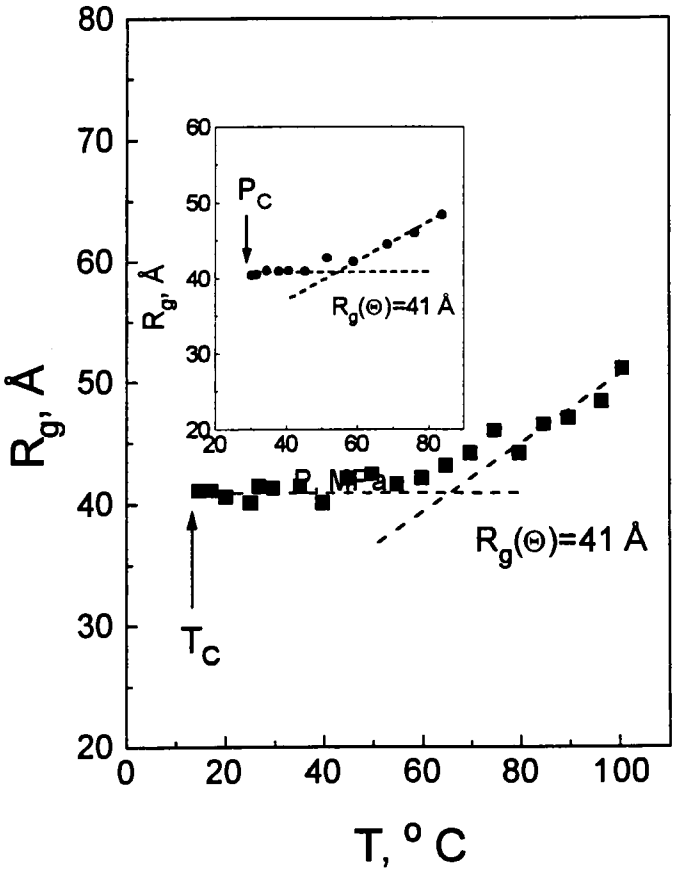


Figure 6. Expansion of PDMS chains in SC CO_2 above the Θ temperature $T_\Theta \sim 65$ °C. Below T_Θ, diverging concentration fluctuations prevent the coil collapsing as observed in organic solvents. The inset illustrates first observation of a Θ pressure in solutions of PDMS in SC CO_2. As observed in organic solvents, the polymer chains do not collapse below P_Θ.

For $P > P_\Theta$ and $T > T_\Theta$, the system exhibits a "good solvent" domain, where the polymer molecules expand beyond the unperturbed R_g, in a good agreement with results of computer simulations [23]. However, for $T < T_\Theta$, and $P < P_\Theta$, the chains do not collapse and maintain their unperturbed dimensions, as observed in organic solvents [4-6], where the size of the concentration fluctuations, ξ diverges as $T \Rightarrow T_C$. Thus, the deterioration of the solvent quality again leads to the formation of microdomains, consisting of interpenetrating polymer coils, as in organic solvents. Near the critical point, the growth of the polymer concentration fluctuations brings together the initially non-overlapping chains, and the macromolecules adopt the unperturbed dimensions, as in highly concentrated systems and in the condensed state. Thus, the stabilization of the molecular dimensions in the poor solvent domain by diverging concentration fluctuations is a universal phenomenon, observed not only in "classical" polymer solutions in organic solvents (e.g. PS in CH), but also in supercritical fluids (e.g. PDMS in CO_2).

A unique attribute of SCFs is that the solvent strength is easily tunable with changes in the system density, offering exceptional control over the solubility. Thus, for PDMS, CO_2 becomes a "theta" solvent at $P_\Theta \sim 52$ MPa and $T_\Theta \sim 65$°C, whereas it behaves as a "good" solvent for $P > P_\Theta$ and $T > T_\Theta$. Below this transition, the chain dimensions never fall below the "theta" R_g, as observed in organic solvents, though for supercritical CO_2, the system may be driven through this transition as a function of pressure in addition to temperature. Understanding the solubility mechanisms is a necessary condition for the development of CO_2-based technologies and SANS promises to give the same level of insight into polymers in supercritical media that it has provided in the condensed state, organic solvents and in aqueous systems.

Acknowledgments

YBM and GDW wish to acknowledge stimulating discussions with M. Muthukumar. The research was supported the Divisions of Advanced Energy Projects and Materials Sciences, U. S. Department of Energy under contract No. DE-AC05-96OR-22464 (with Lockheed Martin Energy Research Corporation) and Grant No. DE-FG05-88ER-45374 (with the University of Tennessee).

References

1. Wignall, G. D. in: *Encyclopedia of Polymer Science and Engineering;* John Wiley & Sons, Inc.: New York, 1987; Vol. 10, p. 112.

2. Chu, B; Ying, Q; Grosberg, A. *Macromolecules* **1995,** 28, 180, and references therein
3. De Gennes, P. G. *Scaling Concepts in Polymer Physics*; Cornell University Press: Ithaca, 1979.
4. Melnichenko, Y. B.; Wignall, G. D. *Phys. Rev. Lett.* **1977,** *78*, 686.
5. Melnichenko, Y. B.; Anisimov, M. A.; Povodyrev, A. A.; Wignall, G. D.; Sengers, J. V.; Van Hook, W. A. *Phys. Rev. Lett.* **1997**, *79*, 5266.
6. Melnichenko, Y. B.; Wignall, G. D.; Van Hook, W. A.; Szydlowski, J.; Rebello, L. P.; Wilczura, H. *Macromolecules* **1998,** *31*, 8436.
7. Kiran, E.; Sen, Y. L. in: *Supercritical Fluid and Engineering Science*; ACS Symposium Series 1993.
8. Koehler, W. C., *Physica* **1986**, *137B*, 320.
9. Wignall, G. D.; Bates, F. S. *J. Appl. Cryst.* **1986**, *20*, 28.
10. DeSimone, J. M.; Guan, Z.; Elsbernd, C. S. *Science*, **1992,** *257*, 945.
11. McClain, J. B.; Betts, D. E.; Canelas, D. A.; Samulski, E. T.; DeSimone, J. M.; Londono, J. D.; Cochran, H. D.; Wignall, G. D.; Chillura-Martino, D.; Triolo, R. *Science* **1996**, 274, 2049.
12. Williams, C. E.; Nierlich, M.; Cotton, J. P.; Jannink, G.; Boue, F.; Daoud, M; Farnoux, B., Picot, C.; de Gennes, P.-G.; Rinaudo, M.; Moan, M.; Wolf, C. *J. Polym. Sci., Polym. Lett. Ed.* **1979**, *17*, 379.
13. Akasu, A. Z.; Summerfield, G. C.; Jahansan, S. N.; Han C. C.; Kim, C. Y.; Yu, H. J.: *Polym. Sci., Polym. Phys. Ed.* **1980**, *18*, 863.
14. King, J. S.; Boyer, W.; Wignall, G. D.; Ullman, R. *Macromolecules* **1985**, *18*, 709.
15. Szydlowski, J.; Van Hook, W. A. *Macromolecules* **1991**, *24*, 4883.
16. Muthukumar, M. *J. Chem. Phys.* **1986**, *85*, 4722.
17. Raos, G.; Allegra, G. *J. Chem. Phys.* **1996,** *104*, 1626.
18. Fujita, H. *Polymer Solutions*; Elsevier: Amsterdam, Oxford, NY, Tokyo; 1990, p. 192.
19. Des Cloizeaux, J.; Jannink, G. *Polymers in Solution. Their Modeling and Structure*; Claredon Press: Oxford; 1990, page 815.
20. This conclusion is strongly supported by recent theoretical calculations which demonstrated a close similarity between the scaling behavior of grafted polymer chains in supercritical and in liquid solvents [Meredith, J. C. and Johnston, K. P., *Macromolecules* **1998**, *31*, 5518].
21. Cotton, J. P.; Nierlich, M.; Boue, F.; Daoud, M.; Farnoux, B.; Jannink, G.; Duplessix, R.; Picot, C. *J. Chem. Phys.* **1976**, *65*, 1101.
22. Melnichenko, Y. B.; Kiran, E.; Wignall, G. D.; Heath, K.; Salaniwal, S.; Cochran, H. D.; Stamm, M., submitted to *Phys.Rev.Lett.*
23. Luna-Barcenas, G.; Meredith, J. C.; Sanchez, I. C.; Johnson, K. P.; Gromov, D. G.; de Pablo, J. J., *J. Chem. Phys.* **1997**, *107*, 10782; Gromov, D. G.; de Pablo, J. J.; Luna-Barcenas, G.; Sanchez, I. C.; Johnson, K. P., *J. Chem. Phys.* **1998**, *108*, 4647.

A.3 Paper 3

Y. B. Melnichenko, E. Kiran, K. D. Heath, S. Salaniwal, H. D. Cochran, M. Stamm, W. A. Van Hook, G. D. Wignall, "Comparison of the Behavior of Polymers in Supercritical Fluids and Organic Solvents via SANS," *XI International Small Angle Scattering Conference, Upton, New York*, in press (1999).

Comparison of the Behavior of Polymers in Supercritical Fluids and Organic Solvents via Small Angle Neutron Scattering

Y. B. Melnichenko[1], E. Kiran[2], K. Heath[1], S. Salaniwal[1], H. D. Cochran[1], M. Stamm[3], W. A. Van Hook[4] and G. D. Wignall[1]

[1]Solid State Division, Oak Ridge National Laboratory*, Oak Ridge, TN 37831

[2]Department of Chemical Engineering, University of Maine, Orono, ME 04468

[3]Max Planck Institut für Polymerforschung, 55021 Mainz, Germany

[4]Chemistry Department, University of Tennessee, Knoxville TN 37996

Small-angle neutron scattering has been used to study the effect of temperature and pressure on the phase behavior of semidilute solutions of polymers dissolved in organic and supercritical solvents. Above the theta temperature (T_θ), these systems exhibits a "good solvent" domain, where the molecules expand beyond the unperturbed dimensions in both organic solvents and in CO_2. However, unlike organic solvents, this transition can be made to occur at a critical "theta pressure" (P_θ) in CO_2 and this represents a new concept in the physics of polymer-solvent systems. For $T < T_\theta$, and $P < P_\theta$, the system enters the "poor solvent" domain, where diverging concentration fluctuations prevent the chains from collapsing and allow them to maintain their unperturbed dimensions.

* Managed by Lockheed Martin Energy Research Corporation under contract DE-AC05-96OR-22464 for the U. S. Department of Energy.

Introduction

In organic solvents, it is well known that the radius of gyration, R_g (i.e. the r.m.s. distance of scattering elements from the center of gravity) of polymer chains depends on the sign and magnitude of the interactions between the chain segments and the molecules of the surrounding liquid. In "good" solvents, the dominating repulsive forces between the segments (excluded volume effects) work to expand the R_g, and the second virial coefficient (A_2) is positive. In less favorable solvents, the pairwise attractive and repulsive segment interactions may compensate at the "theta temperature" (T_θ), where $A_2 = 0$, and R_g corresponds to the dimension of a volume-less polymer coils, "unperturbed" by the excluded volume effects. One of the first applications of SANS was to confirm Flory's prediction that polymer chains would adopt such random-walk configurations in the amorphous state. By definition, T_θ is the critical solution temperature (T_c) for a polymer with the molecular weight $M_w = \infty$ and thus corresponds to the threshold of unlimited polymer–solvent miscibility.

In the poor solvent regime ($T < T_\theta$, $A_2 < 0$), attractive interactions between the segments work to collapse polymer chains into compact polymer globules. This phenomenon is well understood for dilute solutions of widely separated polymer chains, which collapse as $T \sim T_c$ (Chu, Ying and Grosberg, 1995). SANS has recently been used to extend the experimental observations to semidilute solutions of strongly interacting macromolecules . According to the de Gennes' concept, in the critical region ($T \sim T_c$), the chains do not interpenetrate significantly and thus should be collapsed [i.e. $R_g(T_c) < R_g(T_\theta)$] as in dilute concentration regime. However, experiments on PS in cyclohexane (CH) and acetone (AC) have demonstrated that the predicted decrease in R_g is not observed.

Instead, diverging concentration fluctuations near T_c lead to the formation of distinct microdomains of strongly interpenetrating molecules, which prevent the expected collapse (Melnichenko et al., 1997, 1998). Measurements have also been made in supercritical fluids to test the prediction of Kiran and Sen (1993), that the molecules will adopt the unperturbed dimensions at a critical "theta pressure"(P_θ) as they do in polymer solutions at the theta temperature and to explore the similarity and differences between the structure and thermodynamic properties of polymers in SCFs and in sub-critical organic liquids.

Experimental

The SANS data were collected on the 30m SANS facility (Koehler, 1986) at Oak Ridge National Laboratory. The wavelength was $\lambda = 4.75$ Å ($\Delta\lambda/\lambda \sim 5\%$) and the 64 x 64 cm^2 area detector with cell size ~ 1 cm^2 was placed at sample-detector distances of 12m and 5m to give an overall range of momentum transfer of $0.005 < Q = 4\pi\lambda^{-1}\sin\theta < 0.1$ A^{-1}, where 2θ is the angle of scatter. The data were corrected for instrumental backgrounds and detector efficiency on a cell-by-cell basis, prior to radial (azimuthal) averaging and the net intensities were converted to an absolute (± 3%) differential scattering cross section per unit solid angle, per unit sample volume [$d\Sigma(Q)/d\Omega$ in units of cm^{-1}] by comparison with precalibrated secondary standards (Wignall and Bates, 1986). The experiments in supercritical CO_2 were conducted in a cell similar to the one used previously for polymer synthesis and SANS studies of block copolymer amphiphiles in supercritical CO_2 (McClain et al., 1996). The cell was fitted with sapphire windows which caused virtually no attenuation of the neutron beam (cell transmission ~ 93%) or parasitic scattering. The signal from the CO_2 amounted to a virtually flat background (~ 0.04 cm^{-1}), which formed only a minor correction to the scattering from the homopolymer solutions.

High concentration isotope labeling methods [Williams (1979), Akasu (1980), King (1985) and co-workers] were used to obtain the R_g of the polymer molecules and the correlation length (ξ) of the concentration fluctuations. The coherent scattering cross section [$d\Sigma/d\Omega(Q)$] of an incompressible mixture of identical protonated and deuterated polymer chains dissolved in a solvent is given by:

$$I(Q,x) = I_s(Q,x) + I_t(Q,x) \qquad (1)$$

$$I_s(Q,x) = KnN^2 S_s(Q) \qquad (2)$$

$$I_t(Q,x) = LnN^2 S_t(Q) \qquad (3)$$

where the subscripts "s" and "t" correspond to scattering from a single chain and total scattering, x is the mole fraction of protonated chains and n and N are the number density and degrees of polymerization. $S_s(Q)$ is the single-chain structure factor, containing information on the intramolecular R_g, and the total scattering structure factor, $S_t(Q)$ embodies both both intra- and intermolecular correlations between segments. ξ is the correlation length of the concentration fluctuations and the structure factors are normalized so that $S_s = 1$ at ($Q = 0$) and $S_t = S_s$ at infinite dilution. The prefactors K and L are:

$$K = (b_H - b_D)^2 x(1 - x); \qquad L = [b_H x + (1 - x)b_D - b_S]^2 \qquad (4)$$

where b_H and b_D are the scattering lengths of the H^1- and D^2-labeled segments and b_S' is the scattering length of a solvent molecule, normalized to the same specific volume.

The prefactor, L, in equation (4) controls the "total" scattering contribution and it has been shown (Melnichenko et al., 1977) that for isotopic polystyrene (PS) mixtures dissolved in deuterated acetone (AC-d), $L = 0$ at $x = 0.214$. Similarly for PDMS in CO_2, an isotopic ratio of $x = 0.512$ gives $L = 0$ at a solvent density of ρ_{CO2}=0.95 g/cm^3. Thus, the intramolecular scattering function may be obtained directly from the measured cross section at all temperatures and equation (1) gives the R_g directly. For PS in CH-d, however, there is no isotopic ratio, $0 < x < 1$, which satisfies the condition $L(x) = 0$, and $d\Sigma/d\Omega$ always contains a minor (~10%) contribution from the total (intermolecular) scattering, which must be subtracted to extract R_g. If all chains are all protonated (x=1), the prefactor, K = 0, and $d\Sigma/d\Omega \sim S_t(Q)$. Thus, the size of the concentration fluctuations may be measured via the Ornstein-Zernike (O-Z) formalism

$$\frac{d\Sigma}{d\Omega}(Q) = \frac{d\Sigma}{d\Omega}(0) / (1 + Q^2\xi^2) \qquad (5)$$

where ξ is the composition fluctuation correlation length, which may be obtained from the slope of an O-Z plot of $[d\Sigma/d\Omega(0)]^{-1}$ vs. Q^2.

Samples of PS-h, defined by their weight (w) and number (n) averaged molecular weights (M_w = 10,200, 11,600 and 533,000 Dalton) and PS-d (M_w = 10,500, 11,200, and 520,000 Dalton) of polydispersity M_w/M_n ~ 1.06 were purchased from Polymer Laboratories. Deuterated solvents AC-d and CH-d (D/(H+D)=0.995) were purchased from Sigma Chemical and were dried over a molecular sieve prior to preparing solutions near the critical concentration C(PS/CH-d) = 4.4 wt% (Melnichenko and Wignall, 1997) and C(PS/AC-d)=20.3wt% (Szydlowski and Van Hook, 1991). Solutions of (h+d) PS in AC-d and CH-d were prepared at x=0.214 and x=0.20, respectively.

Samples of protonated polydimethylsiloxane (PDMS-h) with M_w = 22,500 47,700, and 79,900 (M_w/M_n ~ 1.03) and PDMS-d (M_w = 27600 and 75600, M_w/M_n ~ 1.11) were used to prepare solutions at a volume fraction of the polymer equal to the concentration of polymer coil overlap (φ^* = 0.1372, 0.0941, and 0.0727 for M_w = 22,500 47,700, and 79,900, respectively) which is close to the critical concentration of phase demixing. Samples were either run at atmospheric pressure in quartz cells as a function of temperature (e.g. PS-h in CH-d or in AC-d or were loaded into the pressure cell with the appropriate isotopic ratio to eliminate (x = 0.512 for PDMS in CO_2) or minimize (e.g. x = 0.2 for PS-h in CH-d) the contribution of the total scattering term in equation (1). The temperature was controlled by circulating fluids (± 0.1°K) and pressure was applied using a screw-type pressure generator.

Results and Discussion

Fig.1 shows the temperature variation of the correlation length ξ for PS-h in CH-d at the critical concentration and it may be seen that the chains do not collapse as $T \sim T_c$, as observed in dilute solutions. Instead, they maintain their unperturbed dimensions in accordance with the theoretical predictions of Muthukumar (1986) and Raos and Alegra (1996). The size of the concentration fluctuations is small in the Θ-region, but increases dramatically near the critical point where it exceeds the chain dimensions [$\xi >> R_g(T_\Theta)$]. These findings indicate that critical polymer solutions can no longer be considered as an ensemble of collapsed, non-interpenetrating chains. Instead, the diverging thermodynamic fluctuations in the critical region lead to the formation of distinct microdomains, representing unperturbed, strongly interpenetrating macromolecules.

Fig. 2 shows values of ξ and R_g over a wide range of pressure and temperature extending from the Θ-domain to the poor solvent condition near T_c. It may be seen that all isobars for $\xi(T)$ merge at the theta temperature, T_θ = 40 °C, where the average correlation length $\xi(Q)$ = 109 ± 5 Å. According to Fujita (1990) and Des Cloizeaux and Jannink (1990)], $\xi(T_\theta) = R_g/3^{1/2}$ at the theta condition, so $\xi(T_\theta)$ = 114 Å for M_w = 533,000 which agrees well with the experimental value.

Unlike the PS-CH system, which may undergo the transition from the poor solvent to the theta solvent by adjusting the temperature, the solvent quality is much poorer for PS - AC. Fig.3 shows the effect of pressure on the thermodynamic state of PS/AC-d solutions and the chain dimensions [R_g - 29 Å] are independent of pressure and temperature and remain close to the unperturbed dimensions for M_w = 11,600 [i.e. R_g - 0.27 $M_w^{1/2}$ ~ 29 Å]. Away from the critical point ($P >> P_c$), the concentration fluctuations fall, though they do not decay sufficiently to reach $\xi(T_\theta) = R_g/3^{1/2}$ - 17Å, so the system never leaves the poor solvent domain. Nevertheless, the macromolecules always exhibit their unperturbed (theta) dimensions, and the diverging concentration fluctuations, which prevented chain collapse in PS-CH, also operate in PS-AC. Remarkably, this mechanism is so effective that even for solutions which never reach the theta solvent domain, the macromolecules are always "stabilized" to exhibit the theta dimensions over wide ranges of pressure and temperature.

SANS has also been used (Melnichenko et al., 1999) to study the dimensions of polymer molecules in supercritical CO_2 to test the prediction of Kiran and Sen (1993) that they will adopt "ideal" configurations, unperturbed by excluded volume effects, at a critical "theta pressure"(P_θ) as they do in polymer solutions at the theta temperature. Experiments on PDMS in CO_2 [figs. (4) and (5)] confirm that the system exhibits both of these

phenomena at a theta pressure, P_θ ~520 bar and temperature T_θ ~ 65°C. To the authors' knowledge, this is the first time that the existence of a "theta pressure" has been demonstrated and this effect constitutes a new concept in the physics of polymer-solvent systems. For $P > P_\theta$ and $T > T_\theta$, the system exhibits a "good solvent" domain, where the polymer molecules expand beyond the unperturbed R_g, measured in the condensed (solid) state. However, for $T < T_\theta$, and $P < P_\theta$, the chains do not collapse as expected [figs. (4) and (5)]. Instead, they maintain their unperturbed dimensions, as observed in organic solvents. Thus, the deterioration of the solvent quality again leads to the formation of microdomains, consisting of interpenetrating polymer coils. Near T_c, the growth of the polymer concentration fluctuations brings together the initially diluted chains, which adopt the unperturbed dimensions, as in highly concentrated systems and in the condensed state. Thus, the stabilization of the molecular dimensions in the poor solvent domain by diverging concentration fluctuations is a universal phenomenon, observed not only in "classical" polymer solutions, but also in supercritical fluids. Thus, there is a close similarity between the behavior of polymer molecules in organic solvents and in CO_2.

However, a unique attribute of SCFs is that the solvent strength is easily tunable with changes in the system density, offering exceptional control over the solubility. Thus, for PDMS, CO_2 becomes a "theta" solvent at P_θ ~520 bar and T_θ ~65°C, whereas it behaves as a "good" solvent for $P > P_\theta$ and $T > T_\theta$. However, in SC CO_2, the system may be driven through this transition as a function of pressure in addition to temperature. Understanding the solubility mechanisms is a necessary condition for the development of CO_2-based technologies and SANS promises to give the same level of insight into polymers and in supercritical media that it has provided in the condensed state and organic solvents.

Acknowledgments

YBM and GDW wish to acknowledge stimulating discussions with M. Muthukumar. The research was supported the Divisions of Advanced Energy Projects and Materials Sciences, U. S. Department of Energy under contract No. DE-AC05-96OR-22464 (with Lockheed Martin Energy Research Corporation) and Grant No. DE-FG05-88ER-45374 (with the University of Tennessee). The financial support of YBM by Alexander von Humboldt-Stiftung is gratefully acknowledged.

References

Akasu, A. Z.; Summerfield, G. C.; Jashan, S. N.; Han C. C.; Kim, C. Y.; Yu, H. J.: *Polym. Sci., Polym. Phys. Ed.* **1980**, *18*, 863.

Chu, B; Ying, Q; Grosberg, A. *Macromolecules* **1995**, 28, 180, and references therein.

Cotton, J. P.; Nierlich, M.; Boue, F.; Daoud, M.; Farnoux, B.; Jannink, G.; Duplessix, R.; Picot, C. *J. Chem. Phys.* **1976**, *65*, 1101.

De Gennes, P. G. *Scaling Concepts in Polymer Physics*; Cornell University Press, 1979.

Des Cloizeaux, J.; Jannink, G. *Polymers in Solution. Their Modeling and Structure*; Claredon Press: Oxford; 1990, page 815.

Fujita, H. *Polymer Solutions*; Elsevier: Amsterdam, Oxford, NY, Tokyo; 1990, p. 192.

King, J. S.; Boyer, W.; Wignall, G. D.; Ullman, R. *Macromolecules* **1985**, *18*, 709.

Kiran, E.; Sen, Y. L. in: *Supercritical Fluid and Engineering Science*; ACS Symposium Series 1993.

Koehler, W. C., *Physica* **1986**, *137B*, 320.

Melnichenko, Y. B.; Wignall, G. D. *Phys. Rev. Lett.* **1977**, *78*, 686.

Melnichenko, Y. B.; Wignall, G. D.; Van Hook, W. A.; Szydlowski, J.; Rebello, L. P. and Wilczura, H. *Macromolecules* **1998**, *31*, 8436.

Melnichenko, Y. B.; Kiran, E.; Wignall, G. D.; Heath, K.; Salaniwal, S.; Cochran, H. D.; Stamm, M., submitted to *Macromolecules*.

McClain, J. B.; Betts, D. E.; Canelas, D. A.; Samulski, E. T.; DeSimone, J. M.; Londono, J. D.; Cochran, H. D.; Wignall, G. D.; Chillura-Martino, D.; Triolo, R. *Science* **1996**, 274, 2049.

Muthukumar, M. *J. Chem. Phys.* **1986**, *85*, 4722.

Raos, G.; Allegra, G. *J. Chem. Phys.* **1996**, *104*, 1626.

Szydlowski, J.; Van Hook, W. A. *Macromolecules* **1991**, *24*, 4883.

Wignall, G. D.; Bates, F. S. *J. Appl. Cryst.* **1986**, *20*, 28.

Williams, C. E.; Nierlich, M.; Cotton, J. P.; Jannink, G.; Boue, F.; Daoud, M; Farnoux, B., Picot, C.; de Gennes, P.-G.; Rinaudo, M.; Moan, M.; Wolf, C. *J. Polym. Sci., Polym. Lett. Ed.* **1979**, *17*, 379.

Figure Captions

Figure 1. $R_g(T)$ and $\xi(T)$ for the solution of PS with M_W=115,000 in CH-d between T_Θ=40 °C and the critical temperature of phase demixing (T_C).

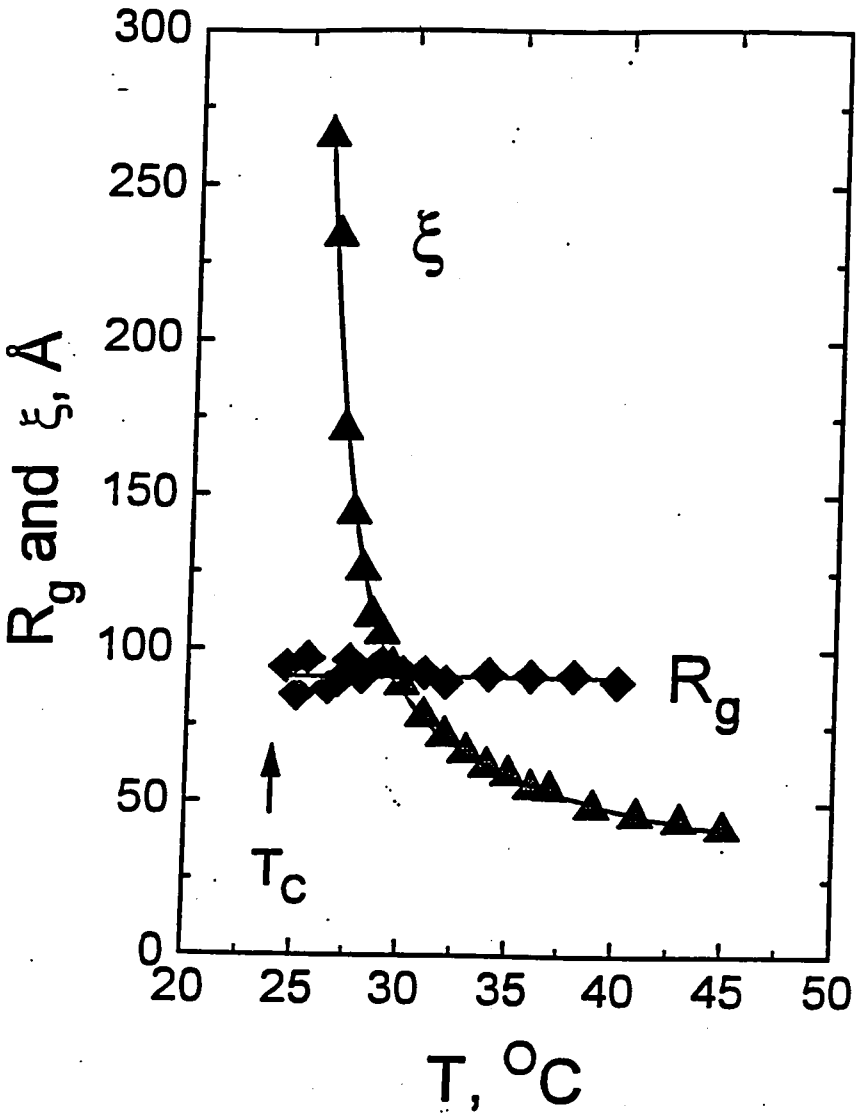
Figure 2. $\xi(T)$ for PS-h (M_W=533,000) in CH-d at different pressures shown in the inset. $R_g(P,T)$ of PS-h chains in the solution of (h+d) PS in CH-d is also shown (□).

Figure 3. $\xi(P)$ of PS-h (M_W=11,600) in AC-d at the different temperatures shown in the inset. $R_g(P,T)$ of labeled PS-h chains in the solution of (h+d) PS in AC-d is also shown (□).

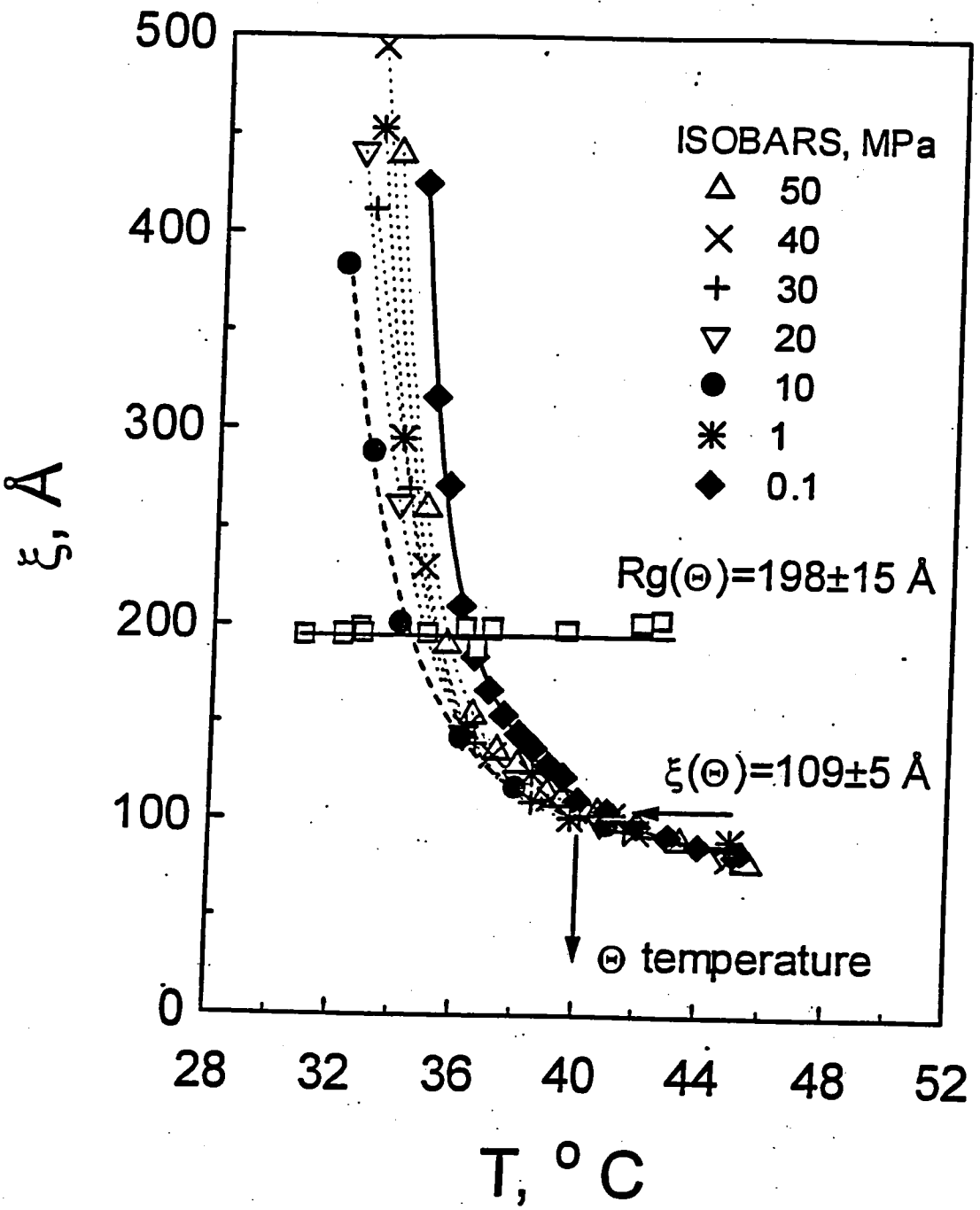
Figure 4. $\xi(T)$ of PDMS in SC CO_2 as a function of density (inset). The value of the Θ temperature depends crucially on the density of SC CO_2, e.g. T_Θ=65±5 °C at ρ_{CO2}= 0.95 g/cm^3, however T_Θ=80±5 °C at ρ_{CO2}= 0.87 g/cm^3. The Θ temperature may hardly be reached at ρ_{CO2}= 0.84 g/cm^3.

Figure 5. Expansion of PDMS in SC CO_2 above the Θ temperature (T_Θ ~ 65 °C) and the Θ pressure (P_Θ ~ 520 bar, inset). Below T_Θ and P_Θ, diverging concentration fluctuations prevent the coil collapsing as observed in organic solvents (Figs. 1 - 3).

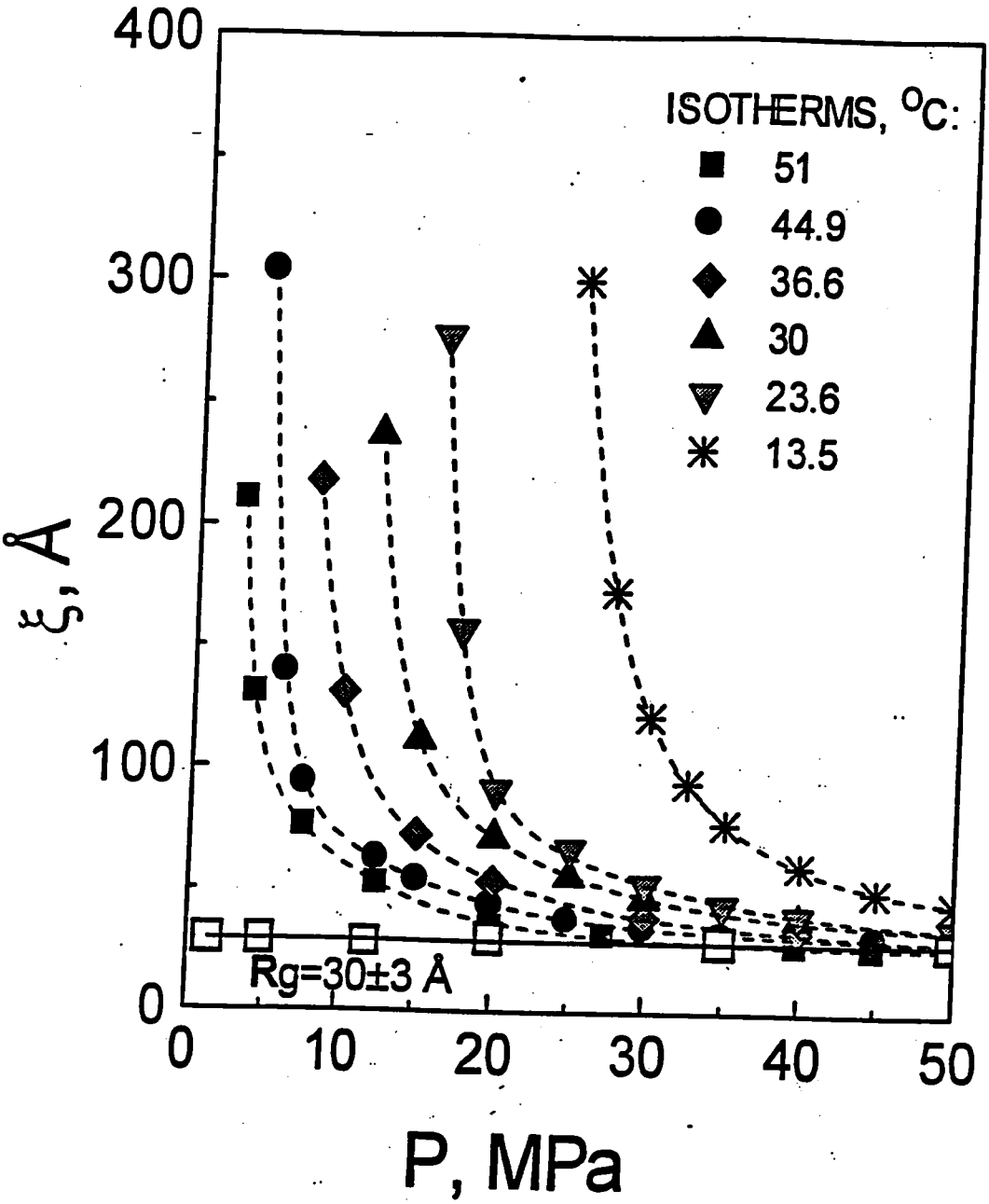
MELNICHENKO ET AL., FIG.1



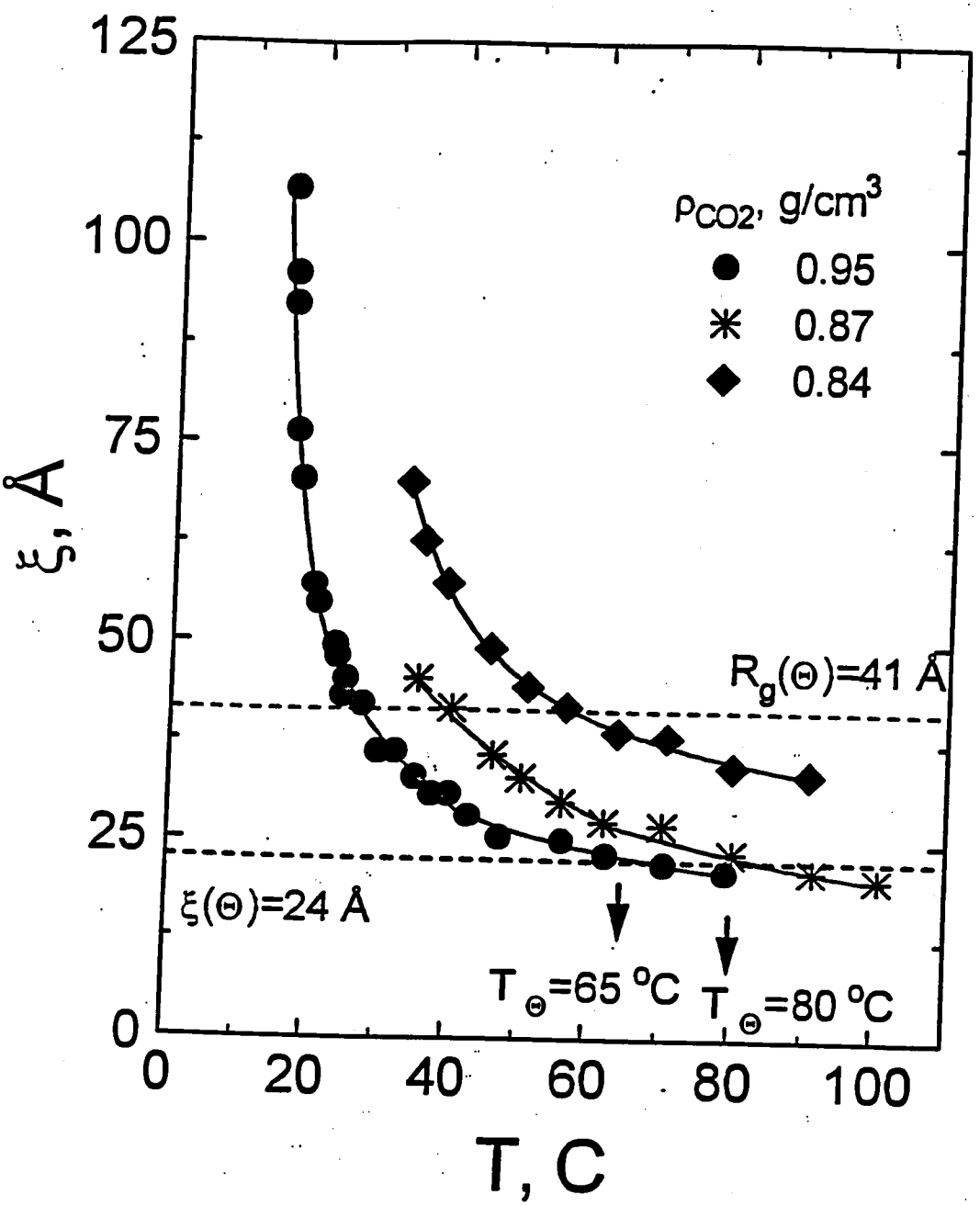
MELNICHENKO ET AL., FIG.2

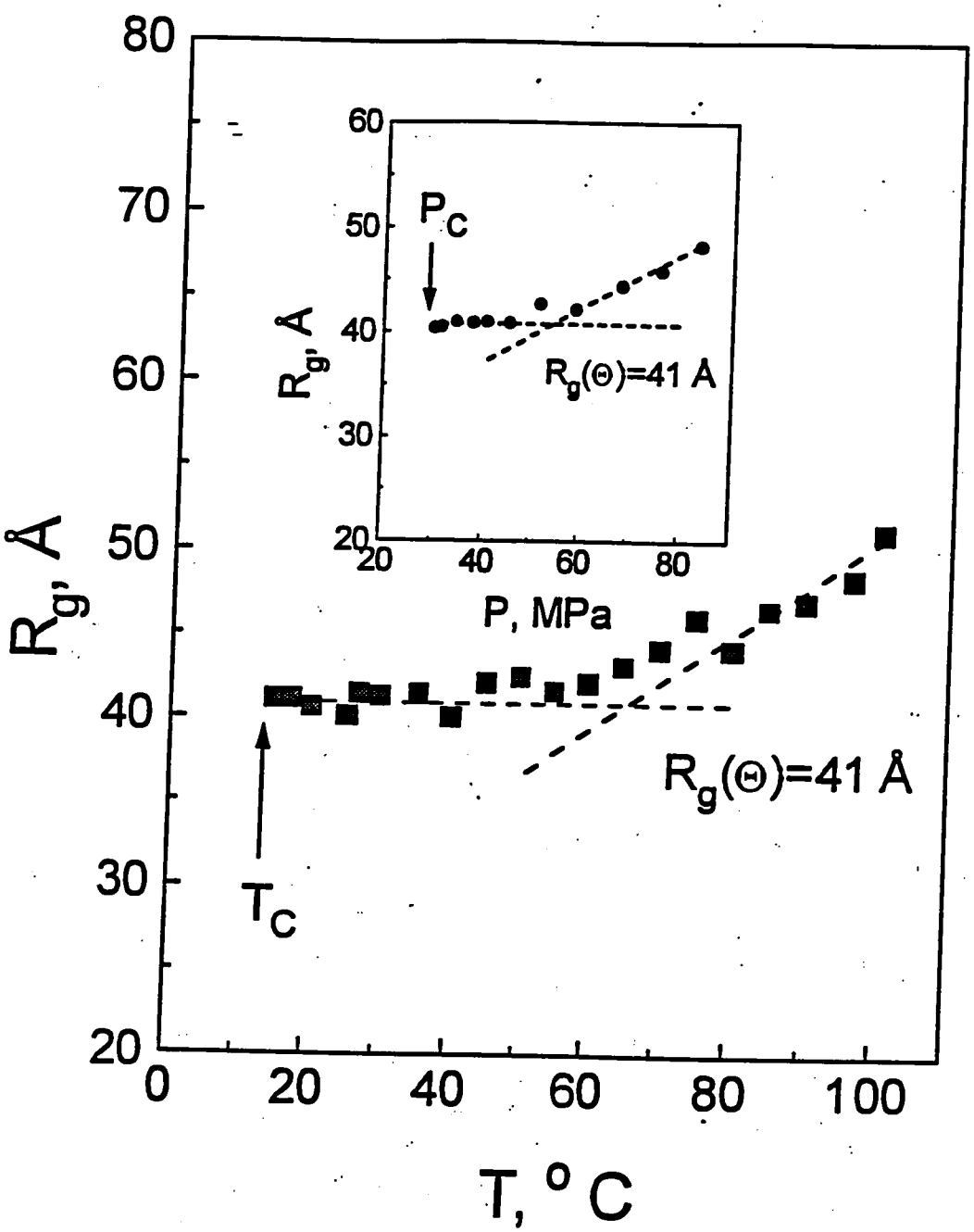


MELNICHENKO ET AL., FIG.3



MELNICHENKO ET AL., FIG.4




80
70
60
50
40
30
20
0
20
40
60
80
100
$R_{g'}$, Å
T, °C
T_C
$R_g(\Theta)$=41 Å
60
50
40
30
20
20
40
60
80
$R_{g'}$, Å
P, MPa
P_C
$R_g(\Theta)$=41 Å

A.4 Paper 4

Y. B. Melnichenko, G. D. Wignall, W. Brown, E. Kiran, H. D. Cochran, S. Salaniwal, K. D. Heath, W. A. Van Hook, M. Stamm, "Static and Dynamic Critical Phenomena in Solutions of Polymers in Organic Solvents and Supercritical Fluids," *Computational Studies, Nanotechnology, and Solution Thermodynamics of Polymer Systems*, in preparation (1999).

APPENDIX B

DENSITY OF MONOMER-SCO_2 SOLUTIONS

For an accurate analysis of scattering experiments, the composition and molar density of the solution is required. Much of this information can be found in the literature. However, some data are missing because a) there was no prior need for such data or b) the measurement of the properties was too difficult. Some SANS and SAXS data that have already be taken lack certain density information for their analysis. Thus for analysis of scattering data, density measurements were completed as a part of this research on various compositions of 2-propanol in carbon dioxide. Density dependence on pressure (ranging from 0.69 to 34.47 MPa) was recorded for three temperatures (25, 45, and 65 °C). The measurements were carried out at Oak Ridge National Laboratory using a vibrating-tube densitometer (Paar DMA 60 with 512 HP remote cell containing a stainless steel tube). The densitometer tube is insulated by a jacket, which is thermostated by an ethylene glycol/water bath. Currently, density measurements are continuing using the same apparatus for the study of various other monomers in SCO_2 to fill the gap in the literature. Four compositions of styrene monomer and methyl methacrylate monomer in carbon dioxide will also be studied. The results will be an independent density publication(s), and a subsequent scattering publication(s).

APPENDIX C

TERNARY PENG-ROBINSON EQUATION OF STATE

Phase equilibria are studied by the Peng-Robinson (PR) (90) equation of state (EOS). It has been shown in the literature (72, 91) that this model predicts SCO_2 behavior satisfactorily. Correlation of the binary systems from the density measurements was one of the uses for this code. In addition, this code has been used to study the binary systems of CO_2-ethanol, CO_2-water, and water-ethanol. Binary interaction parameters were regressed, and subsequently used to study the ternary system of CO_2-ethanol-water (see Table 1). Regression of ternary data from the literature using the above mentioned binary interaction parameters has been demonstrated with acceptable average absolute deviations (see Table 2). This PR EOS was then used to plan the single stage extraction experiment, and predictions for the results were made before an experiment started. The ternary form of the PR EOS was then used to complement the analysis of the ethanol extraction of water using supercritical carbon dioxide.

C.1 Code for Regression of Binary Interaction Parameters

Filename	**Description**
c2oh-h2o.in	system properties input file
c2oh-h2o.323	literature data file for temperature of 323°K
co2-c2oh.in	system properties input file
co2-c2oh.333	literature data file for temperature of 333°K
co2-h2o.in	system properties input file
co2-h2o.323	literature data file for temperature of 323°K
prmx2.in	input and output designation file
prmx2.f	main code and subroutines

```
%
$INPUT
c2oh-h2o-2.323
$
$critical-constants
component # 1
pc = 6.14
vc = 0.1671
tc = 513.9
omega = 0.644
dpm = 1.7
component # 2
pc = 22.12
vc = 0.0571
tc = 647.3
omega = 0.344
dpm = 1.8
$
```

```
%
$INPUT
%co2-c2oh-2.313
co2-c2oh-2.333
$
$critical-constants
component # 1
pc = 7.38
tc = 304.1
vc = .0939
omega = 0.239
dpm = 0.0
component # 2
pc = 6.14
tc = 513.9
vc = .1671
omega = 0.644
dpm = 1.7
$
```

%T	P	X	Y
323.15	0.012921	0.00434	0.0478
323.15	0.014249	0.01522	0.145
323.15	0.015555	0.02722	0.2259
323.15	0.017379	0.04628	0.3182
323.15	0.019088	0.06776	0.3893
323.15	0.021522	0.10983	0.4738
323.15	0.023585	0.17103	0.5355
323.15	0.024961	0.24681	0.5753
323.15	0.025933	0.3238	0.6044
323.15	0.026572	0.38844	0.627
323.15	0.026836	0.4175	0.637
323.15	0.027564	0.50485	0.6681
323.15	0.028131	0.58081	0.6989
323.15	0.028489	0.63429	0.7239
323.15	0.029006	0.72451	0.7735
323.15	0.029176	0.76377	0.7981
323.15	0.029346	0.80838	0.8286
323.15	0.02947	0.85785	0.8664
323.15	0.029511	0.89064	0.8942
323.15	0.029527	0.89934	0.9019
323.15	0.029529	0.92445	0.925
323.15	0.029529	0.9537	0.9531
323.15	0.029154	0.97315	0.9724
323.15	0.02951	0.98153	0.9808

%T	P	X1	Y1
333.4	0.544	0.02	0.905
333.4	0.991	0.038	0.947
333.4	2.11	0.088	0.972
333.4	3.03	0.129	0.978
333.4	4.043	0.169	0.981
333.4	4.993	0.217	0.982
333.4	6.094	0.278	0.981
333.4	7.529	0.372	0.979
333.4	9.024	0.494	0.972

```
%
$INPUT
co2-h2o.323
$
$critical-constants
component # 1
pc =7.38
vc = 0.0939
tc = 304.1
omega = 0.239
dpm = 0.0
component # 2
pc = 22.12
tc = 647.3
vc = .0571
omega = 0.344
dpm = 1.8
$
```

%t	p	x	y
323.15	7.599375	0.0179	0.996181
323.15	10.13250	0.0203	0.995093
323.15	15.19875	0.0219	0.993340
323.15	20.26500	0.0230	0.992606

```
%  the "%" character in the first coulmn means ignore this line.
% The $INPUT-FILES makes the program look for the filename on the next
line
%  which contains information about the components and the actual
%  p, t, x, y data files. Restricted to 1 file name only.
%
$INPUT-FILES
c2oh-h2o.in
$
$OUTPUT
c2oh-h2o-2.out
$
% Use  either $LEAST-SQUARES if using NL2SOL or $NELDER-MEAD
% if you want to use the NELDER MEAD (NM) method for regression.
%$LEAST-SQUARES
$NELDER-MEAD
$SOLVER
% 0 : for analytical solver
% 1 : for numerical solver (default)
SOLVER = 1
$
% the "$NON-DEFAULT-VALUES" field can be used to override
%  default values used by NL2SOL or NM. Look at Algorithm 573
% in ACM TOMS (Transactions on Mathematical Software) Vol 7, pg 369
%  1981.
$NON-DEFAULT-VALUES
SOLPRT = 1
STATPR = 1
X0PRT = 1
MXITER = 200
MXFCAL = 500
AFCTOL = 1.D-5
RFCTOL = 1.D-5
NM-CODE = 1
NELDER-MEAD-TOLERANCE = 1.d-12
$
% The "$INITIAL-GUESS" field should have the initial guess on the
% kij. The format is ( all numbers seperated by semicolon) all in one
line.
%   initial guess (number) ; > low limit on kij ; < high_limit_on_kij ;
%   step number ; The step size is only used by the NM method
% the second line should be left as it is. This has not been implemented
yet.
% If you want to hold the parameter constant use the "@" sign in the
beginning
% of the line ( as shown in line 2).
$INITIAL-GUESS
% If you want to hold the parameter constant use the "@" sign in the
beginning
% of the line ( as shown in line 2)
  -.0000d0  ; > -5.d0 ; < 5.d0 ; step 0.00010d0 ;
@  0.0d0 ; > -1.d0 ; < 1.d0 ; step 0.05d0 ;
$
$RESIDUALS
$ERROR
```

```
c       program prmx2
      implicit double precision (a-h,o-z)
c
c calculates vle using the peng-robinson equation of state
c
c for the pr equations see
c     parameter (ndmax=5000,npmax=4)
      include 'defp.m'
      parameter (nnvd = 93+ndmax*(npmax+3)+npmax*(3*npmax+33)/2)
      real*8 urparm(npmax), v(nnvd)
c     real*8 vc(2), tc(2), ow(2), pc(2), dm(2), rhoc(2)
      include 'dimcr.m'
c     real*8 xexp(ndmax), yexp(ndmax), pexp(ndmax), texp(ndmax)
      include 'dimexp.m'
c     real*8 s(npmax),xub(npmax), xlb(npmax), x(npmax)
      include 'dimpar.m'
      real*8 k12, k0, k1
      integer*4 uiparm(npmax), iv(npmax+60)
      character*50 fname1(1), fnamei(10), fdate*24, fnameo,
     $     fname, sinp*78, fchno*3
      logical*1 ires, fnm, fls, ierr, ivir
      include 'c-kij.m'
      include 'c-const.m'
      include 'c-expdat.m'
      include 'c-critp.m'
      common /dbg1/ ndbg1, ndbg2
      common /res/ ixd
      external cfunpr, alpha, calpha, beta
      data r /8.31434d-3/

      call dfault(iv,v)

        open (unit=10,file='prmx2.in',status='old')
      call scrpa2 (10, fname, fnameo, infil, iofil, np, iv, v,
     $                  x, xub, xlb, s, inmi, fnm, fls, ires, ierr,
     $                  ivir, uiparm)
            if (iofil.gt.0) then
                  open (55,file=fnameo,status='unknown')
            else
            fchno = '000'
                open (34,file='prmx2.xxx',status='unknown')
            read (34,*,end=34) ifno
            write (fchno,33) ifno
33          format(i3)
34          rewind (34)
            write (34,*) ifno+1
            close (34)
            if (fchno(1:1).eq.' ') fchno(1:1) = '0'
            if (fchno(2:2).eq.' ') fchno(2:2) = '0'
                fnameo = 'output/prmx2.'//fchno
                  open (55,file=fnameo,status='new')
      endif

c     write (*,*) fdate()
        write (*,*)  ' ********** prmx2 ********** '
c     write (55,*) fdate()
```

```
        write (55,*)' ********** prmx2 ***************** '
        write (*,*)  '  Peng-Robinson EOS     double precision version '
        write (55,*)'  Peng-Robinson EOS    double precision version '

c
c print the input script file
c
      rewind (10)
      do 7 j = 1,100
            read (10,'(a)',end=8) sinp
            isl = ilen(sinp)
            write (*,'(1x,a)') sinp(1:isl)
            write (55,'(a)') sinp(1:isl)
7     continue

8           close (10)

c read the input data
c
      if (infil.eq.0) then
            fnamei(1) = 'input.dat'
            infil = 1
      endif
      nd = 0
         open (unit=65,file=fname,status='old')
          ncomp = 2
         call scrpc2 (65, fname1, pc, tc, rhoc,
     $                   ow, dm, ncomp, inf )
      do ii = 1,2
      vc(ii) = 1.d0/rhoc(ii)
      write (*,'(1x,'' component # '',i3)') ii
      write (*,'(1x,'' critical density     '',g14.7)') rhoc(ii)
      write (*,'(1x,'' critical volume      '',f10.7)') vc(ii)
      write (*,'(1x,'' critical pressure    '',f10.5)') pc(ii)
      write (*,'(1x,'' critical temperature '',f10.3)') tc(ii)
      write (*,'(1x,'' acentric factor       '',f10.5,/)') ow(ii)
      write (55,'(1x,'' component # '',i3)') ii
      write (55,'(1x,'' critical density     '',g14.7)') rhoc(ii)
        write (55,'(1x,'' critical volume      '',f10.7)') vc(ii)
      write (55,'(1x,'' critical pressure    '',f10.5)') pc(ii)
      write (55,'(1x,'' critical temperature '',f10.3)') tc(ii)
      write (55,'(1x,'' acentric factor      '',f10.5,/)') ow(ii)
      end do
        call inpdt3 (fname1, inf, nd, ires, 1)

      close (65)

c calculate the constants for the equation of state
c
c
c

      call rempar (x, xlb, xub, s, uiparm , np, nc)
      call parsum (nd, x, xlb, xub, s, np, nc, uiparm, npmax)
c
c store the array address for the residual vector at iv(50)
c
       iv(50) = 94 + 2*nd + np*(3*np + 31)/2 + np
```

```
c
c  call vnlvsn to perform nonlinear regression
c   note that
c     np  is the number of parameters being regressed
c     nc is the number of parameters being held constant
      irs = iv(50)

      if ((fls).and.(np.ne.0)) then
        call vnlvsn (nd, np, nc, x, xlb, xub, cfunpr, iv, v, uiparm,
     $                  urparm, ufparm)

      elseif ((fnm).and.(np.ne.0)) then

        call nlm5 (x, xlb, xub,s, np, nc, nd, v(irs), iv, inmi, inmo,
     $                  cfunpr, uiparm, urparm, ufparm)

      endif

c
c calculate the sum of squares and residuals only if vnlvsn returns
c  with a valid code
c
      if ((fnm).and.(inmo.eq.0.or.inmo.eq.-2.or.inmo.eq.-3)) then
      write (*,*) '    '
      elseif (fnm) then
            write (  *,'(1x,'' inmo = '',i3)') inmo
            write (55,'(1x,'' inmo = '',i3)') inmo
      endif

      if (ires) then
      ixd = 1
      call cfunpr (nd, np, x, xlb, xub, nf, v, uiparm
     $      , urparm, ufparm)
      endif

c     write (*,*) fdate()
c     write (55,*) fdate()
      end
```

```
      subroutine scrpa2 (nu, fname1, fname2, infil, iofil,
     $                       p, iv, v, x, xub, xlb, s, inmi,
     $                        fnm, fls, ires, ierr, ivir, uiparm)
      implicit double precision (a-h,o-z)
      character*(*) fname1, fname2
      integer nu, p, uiparm(*)
      dimension fname1(*), iv(*), v(*), x(*), xlb(*),
     $               xub(*), s(*), ww(11)
      character sinpu*90, sinp*90
      logical*1 finp, fout, ing, ndv, ires, fnm, fls, ivir, ierr

      common /out2/ iflag, iwd, iwh, iwb, ww

      infil = 0
      iofil = 0
      inpg  = 0
      inpc  = 0
c analyze the input script file
c
      do 5 j = 1,200
            sinp =
     $'
            read (nu,'(a)',end=6) sinp
c           write (*,*) sinp
                if ((finp) .or. (fout)) then
                   sinpu = sinp
                   goto 57
                endif
            call lower (sinpu, sinp)
57              continue
c
c check to see if the line is not a comment statement
c
        if ((ilen(sinpu).ne.0).and.(sinpu(1:1).ne.'%')) then
c
c if the line is not a comment line  and the input file flag has
c  been set .....
c
            if ((finp).and.(sinpu(1:1).ne.'$')) then
                  if (ilen(sinpu).ne.0) then
                        infil = infil + 1
                        fname1(infil) = sinpu
                  endif

            elseif ((finp).and.(sinpu(1:1).eq.'$')) then
                  if (infil.eq.0) then
                        infil = -1
                  endif
                  finp = .false.

            elseif ((fout).and.(sinpu(1:1).ne.'$')) then
                  if (ilen(sinpu).ne.0) then
c                       iofil = iofil + 1
c                       fname2(iofil) = sinpu
                        fname2 = sinpu
                        iofil = 1
                  endif
```

```
            elseif ((fout).and.(sinpu(1:1).eq.'$')) then
                  if (iofil.eq.0) then
                        iofil = -1
                  endif
                  fout = .false.
            elseif ((ndv).and.(sinpu(1:1).ne.'$')) then

                if (ilen(sinpu).ne.0)
     $               call tunen8 (sinpu, iv, v, iwd, iwh, iwb, inmi,
ww)

            elseif ((ndv).and.(sinpu(1:1).eq.'$')) then
                  ndv = .false.
            elseif ((ing).and.(sinpu(1:1).ne.'$')) then
                  if (ilen(sinpu).ne.0) then
                        inpg = inpg + 1
            call pscan( sinpu, x(inpg), xlb(inpg), xub(inpg),
     $                        s(inpg), ipc)
                        uiparm(inpg) = -ipc
                  endif

            elseif ((ing).and.(sinpu(1:1).eq.'$')) then
                  p = inpg
                  ing = .false.

            elseif ((index1(sinpu,'$input-file')).ne.0) then
                  finp = .true.
            elseif ((index1(sinpu,'$output')).ne.0) then
                  fout = .true.
            elseif ((index1(sinpu,'$non-default-values')).ne.0) then
                  ndv = .true.
            elseif ((index1(sinpu,'$nelder-mead')).ne.0) then
                  fnm = .true.
            elseif ((index1(sinpu,'$least-squares')).ne.0) then
                  fls = .true.
            elseif ((index1(sinpu,'$initial-guess')).ne.0) then
                  ing = .true.
            elseif ((index1(sinpu,'$residual')).ne.0) then
                  ires = .true.
            elseif ((index1(sinpu,'$virial')).ne.0) then
                  ivir = .true.
            elseif ((index1(sinpu,'$error')).ne.0) then
                  ierr = .true.
            endif
          elseif (sinpu(1:1).eq.'%') then
c           write (*,*)sinpu
          endif
5     continue
6     continue
      rewind (20)
      return
      end
```

```
      subroutine scrpc2 (nu, fname1, pc, tc, rhoc, ow, dm, nc, inf)
c-----------------------------------------------------------------------
-------
c
c   vinod shah     3/19/92
c
c   this subroutine is to be used only for binary mixtures
c
c   this subroutine reads an input file opened as unit # nu and sorts
c    out information on the input filename(s) and critical properties
c    of the two components that form the binary mixture.
c
c   the input file has to have a specified format shown below
c
c   filenames that have input data to be read in have to be preceded by
a
c    '$input' on a new line and the list terminated by a '$' on a new
line.
c    the limit on the number of filenames
c    that can be read is set by the declaration of the fname1 array in
the main program.
c   critical constants are preceded by '$critical-constants' and the
list terminated by
c    a '$' on a new line. constants for each component are preceded by
'component # id'
c    on a new line, where id is the component number (1 or 2). this has
to be followed by
c    one or more constants, each on a seperate line. the format for the
constants is
c
c     pc = 123
c       tc = 456
c       vc = 789
c       rhoc = 123
c       omega = 456
c     dpm = 789
c

c    e.g if the input file contains
c    12345678901234567890123456789O    this denotes the column no in the
file.
c    $input
c    vms/f1.dat
c    vms/f2.dat
c    f3.dat
c    $
c
      implicit double precision (a-h,o-z)
      character*50  fname1
      integer nu
      real*8 pc(nc), tc(nc), rhoc(nc), ow(nc), dm(nc)
      dimension fname1(*)
      character*50 sinpu*78, sinp*78
      logical*1 finp, finc, finco
      logical*1 stod

      common /out2/ iflag, iwd, iwh, iwb, ww
```

```
      inf = 0
      iofil = 0
      inpg  = 0
      inpc  = 0

c
c analyze the input script file
c
      do 5 j = 1,200
            sinp =
     $'                                                                '
            read (nu,'(a)',end=6) sinp
c           write (*,*) sinp
            call lower (sinpu, sinp)
c
c check to see if the line is not a comment statement
c
         if ((ilen(sinpu).ne.0).and.(sinpu(1:1).ne.'%')) then

c
c if the line is not a comment line  and the input file flag has
c  been set .....
c
            if ((finp).and.(sinpu(1:1).ne.'$')) then
                  if (ilen(sinpu).ne.0) then
                        inf = inf + 1
                        fname1(inf) = sinp
                  endif

            elseif ((finp).and.(sinpu(1:1).eq.'$')) then
                  if (inf.eq.0) then
                        inf = -1
                  endif
                  finp = .false.

            elseif ((finc).and.(sinpu(1:1).ne.'$')) then
               if ((index1(sinpu,'component')).ne.0) then
                  finco = .true.
                  il1 = ilen(sinpu)
                  rhoci = 0.0
                  ipo = index1(sinpu,'#')
                  if (ipo.ne.0) then
                     if (stod(sinpu(ipo+1:il1),val)) then
                        id = int(val)
                     else
                        write (*,*) ' error reading component # '
                     endif
                  else
                     write (*,*) ' error reading component statemant'
                  endif
               elseif ((ilen(sinpu).ne.0).and.(id.gt.0)) then

                  call crits2 (sinpu, pci, tci, rhoci, omi, dmi)
                  pc(id) = pci
                  tc(id) = tci
                  ow(id) = omi
                  dm(id) = dmi
                  rhoc(id) = rhoci
```

```
            endif

          elseif ((finc).and.(sinpu(1:1).eq.'$')) then
                finc = .false.

          elseif ((index1(sinpu,'$input')).ne.0) then
                finp = .true.
          elseif ((index1(sinpu,'$critical-constants')).ne.0) then
                finc = .true.
          endif
        elseif (sinpu(1:1).eq.'%') then
c         write (*,*)sinp
        endif

5     continue
6     continue


      return
      end
```

```
      subroutine inpdt3 (fnamei, infil, nd, ires, idno)

c-----------------------------------------------------------------------
c  vinod shah 7/24/91
c
c
c reads the input files
c
c  on input :
c
c     infil : integer   : the number of input files to be read
c     ires  : logical*1 : true if residuals are to be printed
c                         false otherwise
c
c  on output :
c     nd    : integer    : the number of data points read in
c
c when reading the files the following points to be noted
c
c    a '%' sign in the 1st 5 columns of the line denotes a comment
c     line. the comment line(s) must appear before any valid data is
c     read in. the subroutine skipl1 scans out any comment lines in
c     the data files. skipl1 also returns "isc" or the scan-code to
c     which determines the format of the input data. the default value
c     is 3.
c

      implicit double precision (a-h,o-z)
      integer*4   npmax, ndmax
      include 'defp.m'
      include 'dimexp.m'
      character*(*) fnamei
      character*80 ssp
      dimension fnamei(1)
      logical*1 ires
      include 'c-expdat.m'

      ndn = nd + 1
c
c default value of isc
c
      isc = 1
      do k = 1,infil
          open( 20,file=fnamei(k),status='old')
          write (*,85) fnamei(k)
          write (55,85) fnamei(k)
85    format(1x,' reading input file : ',a)

c
c skip the comment lines and print them to the screen and file-no 55
c
          call skipl1 (20, 55 , 1, isc)

          do 10 i = 1,500
           if (isc.eq.1) then
                read (20,'(a)',end=99) ssp
               if (index(ssp,'%') .eq. 1) then
              write (*,*) ssp
```

```
                go to 10
              else
            backspace (20)
            read (20,*,end=99) texp(ndn), pexp(ndn), xexp(ndn),
yexp(ndn)
            idc(ndn) = idno
              if (.not.ires) then
                write (* ,'(1x,4(1x,g12.5,2x:),i5)')texp(ndn)
     $                , pexp(ndn), xexp(ndn), yexp(ndn)
                write (55,'(1x,4(1x,g12.5,2x:),i5)')texp(ndn)
     $                , pexp(ndn), xexp(ndn), yexp(ndn)
              endif
              ndn = ndn + 1
              if (ndn.gt.ndmax) then
                write (  *,20) ndmax
                write (55,20) ndmax
                goto 990
              endif
             endif
           endif
10        continue
99     close (20)
       end do

990      nd = ndn - 1
20     format(1x,' * * * warning: ndmax exceeded - rest of the data ',
     $ 'not read  * * * ')
       return
       end
```

```
      subroutine cfunpr (ndata, np, xn, xlbn, xubn, nf, r1, uiparm
     $      , urparm, ufparm)
      implicit real*8 (a-h,o-z)
      integer*4 uiparm(*)
      include 'defp.m'
      include 'dimcr.m'
      include 'dimexp.m'
      real*8 xn(*),xlbn(*), xubn(*), urparm(*),r1(*), eqk(2)
      real*8 k12, x(2), y(2), ye
      real*8 a(2), b(2)
      include 'c-kij.m'
      include 'c-expdat.m'
      include 'c-critp.m'
      common /res/ ixd
      data r /8.31434d-3/


      nf = 1
      do kk = 1, np
          if ((xn(kk).gt.xubn(kk)).or.(xn(kk).lt.xlbn(kk))) then
            nf = 0
            return
          endif
      end do
      k12 = xn(1)
      ck12 = xn(2)
      dpt = 0.d0
      dp1 = 0.d0
      dps = 0.d0
      dy1 = 0.d0
      dp1 = 0.d0
      dyt = 0.d0
        if (ixd.ne.0) then
                write ( *,105)
                write (55,105)
105     format(1x,'  Temp         x      Pcalc     Pexp    ycalc    yexp'
     $,'     rhov         rhol        id ' )
        endif


      nad = 0
c calculate the constants for the equation of state
c     prb  = 0.0778d0*r*tc/pc
c     praa = 0.45724d0*r**2*tc**2/pc
      oka1  = 0.37464 + 1.54226*ow(1) - 0.26992*ow(1)**2
      ob1 = 0.0778d0
      oa1 = 0.45724d0
      bc1 = ob1*r*tc(1)/pc(1)
      ac1 = oa1*r**2*tc(1)**2/pc(1)
      oka2  = 0.37464 + 1.54226*ow(2) - 0.26992*ow(2)**2
      ob2 = 0.0778d0
      oa2 = 0.45724d0
      bc2 = ob2*r*tc(2)/pc(2)
      ac2 = oa2*r**2*tc(2)**2/pc(2)
      do n = 1, ndata
          x(1) = xexp(n)
            ye = yexp(n)
c         y(1) = 0.9999d0
```

```
            y(1) = yexp(n)
          p = pexp(n)/.8d0
          t = texp(n)
          y(2) = 1.d0 - y(1)
          x(2) = 1.d0 - x(1)
          tr1 = t/tc(1)
          alpha1 = (1.d0 + oka1*(1.d0 - dsqrt(tr1)))**2
            a(1) = alpha1*ac1
          b(1) = bc1

          tr2 = t/tc(2)
          alpha2 = (1.d0 + oka2*(1.d0 - dsqrt(tr2)))**2
            a(2) = alpha2*ac2
          b(2) = bc2
C     calculate the mixture

      if (x(1).eq.1.0) then
         call nrfgp2 (p, t, a(1), b(1), vv, vl, ier)
         y(1) = 1.d0
         y(2) = 0.d0
         if (ierr.lt.0) then
            idc(n) = -1
            r1(n) = 0.d0
            goto 98
         endif
      elseif (x(1).eq.0.0) then
         call nrfgp2 (p, t, a(2), b(2), vv, vl, ier)
         y(1) = 0.d0
         y(2) = 1.d0
         if (ierr.lt.0) then
            idc(n) = -1
            r1(n) = 0.d0
            goto 98
         endif
        else
         call nreqp1 (p, t, x, a, b, y, eqk, vv, vl, ierr, ye)
         if (ierr.lt.0) then
            idc(n) = -1
            r1(n) = 0.d0
            goto 98
         endif
      endif

        r1(n)= (p/pexp(n)-1.d0)
        r1(n+ndata) = (y(1)/yexp(n)-1.0)

        if (ixd.ne.0) then
c         write (*,*) ' results '

          write (*,100) t, x(1), p, pexp(n), y(1), yexp(n), 1.0/vv,
     +      1.0/vl, idc(n)
          write (55,100) t, x(1), p, pexp(n), y(1), yexp(n), 1.0/vv,
     +      1.0/vl, idc(n)

          dpt = dabs(p/pexp(n)-1.d0) + dpt
          dps = r1(n)**2 + dps
          dp1 = dabs(p-pexp(n)) + dp1
          dyt = dabs(y(1)/yexp(n)-1.0) + dyt
```

```
        endif
100     format(1x,f9.3,1x,f6.4,1x,2(f8.4,1x),2(f6.4,1x),2(E11.4,1x)
     + ,1x,i3)
98      continue
        end do

          if (ixd.ne.0) then
                dp1 = dp1/dble(ndata)
                dpt = dpt/dble(ndata)
                dpsn = dps/dble(ndata)
                dyt = dyt/dble(ndata)
104     format(1x,10(g11.4,2x:))
                write (*,102)
                write (55,102)
102             format(5x,
     $'ndata      p_aad          yaad      P-aad           SS          SS*N')

                write (*,103) ndata, dpt, dyt, dp1, dpsn, dps
                write (55,103) ndata, dpt, dyt, dp1, dpsn, dps
          endif
103     format(2x,i5,3x,g12.5,5(1x,g11.4:))

        return
        end
```

C.2 Ternary Code for Comparison with Literature Data

Filename	**Description**
h2o-c2oh-co2.in	ternary system input file with kij
h2o-c2oh-co2.1	literature data file
terpr.in	input and output designation file
terpr.f	main code

```
%
$INPUT
h2o-c2oh-co2-333-10.1
%h2o-c2oh-co2-2.1
%h2o-c2oh-co2-333-18.1
$
$kij
%k12
%-0.15618
%-0.112165
-0.079
%k13
%0.1
0.09
%0.1108
%0.114
%-0.0884
%0.192
%k23
%0.058756
%0.06544
%0.101592
%0.0950915
%0.0530
%0.103256
%0.0852177
0.0852
$
$critical-constants
component # 1
pc = 22.12
vc = 0.0571
tc = 647.3
omega = 0.344
dpm = 1.8
component # 2
pc = 6.14
vc = 0.1671
tc = 513.9
omega = 0.644
dpm = 1.7
component # 3
pc = 7.38
tc = 304.1
vc = .0939
omega = 0.239
dpm = 0.0
$
```

% T	P	X1	x2	Y1	y2
313.2	10.3	0.7356	0.2644	0.0076	0.0381
313.2	10.3	0.7486	0.2514	0.0073	0.0368
313.2	10.3	0.7505	0.2495	0.0071	0.0358
313.2	10.3	0.7617	0.2383	0.0069	0.0354
313.2	10.3	0.7751	0.2249	0.0066	0.0347
313.2	10.3	0.7768	0.2232	0.0062	0.0341
313.2	10.3	0.7892	0.2108	0.0056	0.0322
313.2	10.3	0.8101	0.1899	0.0050	0.0295
313.2	10.3	0.8226	0.1744	0.0047	0.0264
313.2	10.3	0.9134	0.0866	0.0044	0.0144
313.2	10.3	0.9155	0.0845	0.0043	0.0140
313.2	10.3	0.9160	0.0840	0.0042	0.0137
313.2	10.3	0.9222	0.0778	0.0036	0.0117
313.2	10.3	0.9233	0.0767	0.0034	0.01111

```
$input-file
h2o-c2oh-co2.in
%n2-co2-c2h6.in
%nc6-cycc6-c7.in
%cycc6-3m2b-c8.in
%n2-ar-c1.in
$
$output-file
%dewn2-co2-c2h6.out
%h2o-c2oh-co2.out
%qnc6-cycc6-c7.out
%qcycc6-3m2b-c8.out
%qn2-ar-c1.out
%qn2-co2-c2h6.out
test.out
$
```

```
      Program TERPR

      real*8 vc(3), tc(3), ow(3), pc(3), dm(3), rhoc(3), kij(3)
      parameter(ndmax=500)
      real*8 x1exp(ndmax), y1exp(ndmax), pexp(ndmax), texp(ndmax)
      real*8 x2exp(ndmax), y2exp(ndmax), idc(ndmax)
        integer infil, iofil, isl, nd, inf
      character sinp*90, sinpu*90
      character*50 fname1,fname2,fname3(10)
      logical finp, fout
      logical*1 ires
      common /dbg1/ ndbg1, ndbg2
      common /res/ ixd
      common /expdat/pexp,texp,x1exp,x2exp,y1exp,y2exp,idc
      common /critp/pc,vc,tc,ow,dmr
      external cfunpr, alpha, calpha, beta
      data r /8.31434d-3/



C
C     Read main input file 'terpr.in'
C

      open (unit=10,file='terpr.in',status='old')
      do 5 j = 1,50
         sinp =
     $'
         read (10,'(a)',end=6) sinp
         sinpu = sinp
         if ((ilen(sinp).ne.0).and.(sinp(1:1).ne.'%')) then
              if ((finp).and.(sinp(1:1).ne.'$')) then
                   if (ilen(sinp).ne.0) then
                        infil = infil + 1
                        fname1 = sinpu
                   endif
              elseif ((finp).and.(sinp(1:1).eq.'$')) then
                   if (infil.eq.0) then
                        write(*,*) 'Please specify input file!!'
                  stop
                   endif
                   finp = .false.
              elseif ((fout).and.(sinp(1:1).ne.'$')) then
                   if (ilen(sinp).ne.0) then
                        fname2 = sinpu
                        iofil = 1
                   endif
              elseif ((fout).and.(sinp(1:1).eq.'$')) then
                 if (iofil.eq.0) then
                        write(*,*) 'Please specify output file!!'
                  stop
                   endif
                   fout = .false.
             elseif ((index1(sinp,'$input-file')).ne.0) then
                  finp = .true.
             elseif ((index1(sinp,'$output-file')).ne.0) then
                  fout = .true.
```

```
            endif
         elseif (sinp(1:1).eq.'%') then
         endif
5     continue
6     continue


C
C     write to the output file
C

      open (55,file=fname2,status='unknown')
      write (*,*)  ' ********** terpr ***************** '
      write (55,*)' ********** terpr ***************** '
      rewind (10)
      do 7 j = 1,100
            read (10,'(a)',end=8) sinp
            isl = ilen(sinp)
            write (*,'(1x,a)') sinp(1:isl)
            write (55,'(a)') sinp(1:isl)
7     continue
8     close (10)
      open (unit=65,file=fname1,status='old')
      ncomp = 3

C
C     read system input file
C
      call scrpc2 (65,fname3,pc,tc,rhoc,
     $                  ow, dm, ncomp, inf, kij )
      write(*,*)  'k12 =',kij(1)
      write(55,*) 'k12 =',kij(1)
      write(*,*)  'k13 =',kij(2)
      write(55,*) 'k13 =',kij(2)
      write(*,*)  'k23 =',kij(3)
      write(55,*) 'k23 =',kij(3)
      do ii = 1,3
         vc(ii) = 1.d0/rhoc(ii)
         write (*,'(1x,'' component # '',i3)') ii
         write (*,'(1x,'' critical density     '',g14.7)') rhoc(ii)
         write (*,'(1x,'' critical volume      '',f10.7)') vc(ii)
         write (*,'(1x,'' critical pressure    '',f10.5)') pc(ii)
         write (*,'(1x,'' critical temperature '',f10.3)') tc(ii)
         write (*,'(1x,'' acentric factor       '',f10.5,/)') ow(ii)
         write (55,'(1x,'' component # '',i3)') ii
         write (55,'(1x,'' critical density     '',g14.7)') rhoc(ii)
         write (55,'(1x,'' critical volume      '',f10.7)') vc(ii)
         write (55,'(1x,'' critical pressure    '',f10.5)') pc(ii)
         write (55,'(1x,'' critical temperature '',f10.3)') tc(ii)
         write (55,'(1x,'' acentric factor      '',f10.5,/)') ow(ii)
      end do
      close (65)


C
C     read real data files
C
```

```
      nd = 0
      call inpdt3 (fname3, inf, nd, ires, 1)


C
C     begin the calculation
C

      call funpr (nd, nf, kij)
      stop
      END


      INTEGER FUNCTION ILEN (STRW)
C-----------------------------------------------------------------------
--
C
C     INTEGER FUNCTION ILEN DETERMINE THE TRUE LENGTH OF THE STRING,
C      i.e. IT NEGLECTS ANY TRAILING SPACES IN THE STRING.
C
C     Vinod Shah  7/13/90
C
C-----------------------------------------------------------------------
--

      CHARACTER*(*) STRW
      INTEGER L, I

      L = LEN(STRW)
      DO 100 I = L,1,-1
            IF ((STRW(I:I).NE.' ').AND.(STRW(I:I).NE.CHAR(09)))
     $              GOTO 110
100   CONTINUE
110   ILEN = I
      RETURN
      END


      INTEGER*4 FUNCTION INDEX1 (SSR1, SSR2)
      CHARACTER*(*) SSR1, SSR2

*-----------------------------------------------------------------------*
*                                                                       *
*  INDEX1  - FINDS THE LOCATION OF A SUBSTRING IN A STRING              *
*                                                                       *
*  SSR1 IS THE STRING IN WHICH THE SUBSTRING SSR2 IS TO BE FOUND        *
*                                                                       *
*     Vinod Shah              Final version   7/12/90                   *
*                                                                       *
*  Note that ssr2 must not have unnecessary leading blanks              *
*                                                                       *
*-----------------------------------------------------------------------*

      LOGICAL FOUND
      INTEGER IL, I, L, I1, I2

      INDEX1 = 0
```

```
      FOUND = .FALSE.
C-----------------------------------------------------------------------
C
C FIND THE LENGTH OF SSR1 NEGLECTING ANY TRAILING BLANKS
C
C-----------------------------------------------------------------------
      L = LEN(SSR1)
      DO I = L,1,-1
            IF (SSR1(I:I).NE.' ') GOTO 110
      END DO
110   I1 = I

C-----------------------------------------------------------------------
C
C FIND THE LENGTH OF SSR2 NEGLECTING ANY TRAILING BLANKS
C
C-----------------------------------------------------------------------

      L = LEN(SSR2)
      DO I = L,1,-1
            IF (SSR2(I:I).NE.' ') GOTO 120
      END DO
120   I2 = I

C-----------------------------------------------------------------------
C
C IF THE LENGTH OF SSR2 IS GRATER THAN SSR1 RETURN AS A COMPARISION
C  CANNOT BE DONE
C
C-----------------------------------------------------------------------

      IF (I2.GT.I1) RETURN

      IL = 1

      DO I = 1, I1

           IF ((SSR1(I:I).EQ.SSR2(IL:IL)).AND.(.NOT.FOUND)) THEN
            FOUND = .TRUE.
            IL = IL + 1
            INDEX1 = I
           ELSEIF ((SSR1(I:I).EQ.SSR2(IL:IL)).AND.(FOUND)) THEN
            IL = IL + 1
           ELSE
            FOUND = .FALSE.
            IL = 1
            INDEX1 = 0
           ENDIF

           IF ((IL.EQ.(I2+1)).AND.(FOUND)) RETURN

      END DO

      IF (IL.LT.(I2+1)) INDEX1 = 0

      RETURN
      END
```

```
      LOGICAL*1 FUNCTION STOD (STR,RR)

C-----------------------------------------------------------------------
--
C
C     THIS SUBROUTINE CONVERTS A CHARACTER STRING NUMBER TO A
C      REAL*8 NUMBER.
C
C     STR  : A CHARACTER STRING CONTAINING THE NUMBER
C     RR   : A REAL*8 NO. IN WHICH THE VALUE IS RETURNED
C     STOD : A LOGICAL VARIABLE WHICH IS RETURNED .FALSE. IF AN ERROR
C             WAS FOUND DURING THE CONVERSION.
C
C     Vinod Shah        7/13/90
C                Modified   10/21/90
C
C-----------------------------------------------------------------------
--

      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*(*) STR
      REAL*8 X,RR
      CHARACTER STRT*30,MANT*30,EXP*5
      LOGICAL*1 CTOD

      STOD = .FALSE.
      X = 0.D0

C-----------------------------------------------------------------------
--
C     GET THE LENGTH OF STR AS DEFINED IN THE CALLING PROGRAM
C-----------------------------------------------------------------------
--
      L = LEN(STR)

C-----------------------------------------------------------------------
--
C     DETERMINE (IF ANY) THE NUMBER OF LEADING BLANKS &/OR TABS
C-----------------------------------------------------------------------
--
      DO 50 I=1,L
            IF ((STR(I:I).NE.' ').AND.(STR(I:I).NE.CHAR(09)))
     $               GOTO 70
50    CONTINUE

C-----------------------------------------------------------------------
-------
C     GET THE ACTUAL LENGTH OF THE STRING. THAT IS NEGLECT TRAILING
C      BLANKS
C-----------------------------------------------------------------------
-------

70    LA = ILEN(STR(I:L))

      IF (LA.EQ.0) RETURN
```

```
      STRT = STR(I:L)

C-----------------------------------------------------------------------
-------
C     CHECK TO SEE IF THE FIRST CHARACTER IN THE STRING IS A SIGN
C      DESCRIPTOR. IF SO NOTE THE SIGN
C-----------------------------------------------------------------------
-------
      IF (STRT(1:1).EQ.'-') THEN
            SIGN = -1.D0
            ISM = 2
      ELSEIF (STRT(1:1).EQ.'+') THEN
            SIGN = +1.D0
            ISM = 2
      ELSE
            ISM = 1
            SIGN = 1.D0
      ENDIF

C-----------------------------------------------------------------------
-------
C     FIND OUT THE LOCATION OF 'E' OR 'D' AND A DECIMAL POINT IN THE
C      STRING
C-----------------------------------------------------------------------
-------

      IE1 = INDEX(STRT(1:LA),'E')
      IE2 = INDEX(STRT(1:LA),'e')
      IE  = MAX0(IE1,IE2)
      ID1 = INDEX(STRT(1:LA),'D')
      ID2 = INDEX(STRT(1:LA),'d')
      ID  = MAX0(ID1,ID2)
      IDEC = INDEX(STRT(1:LA),'.')

C-----------------------------------------------------------------------
-------
C     IF A DECIMAL POINT IS ABSENT IT IS ASSUMED TO BE AT THE END
C      OF THE STRING FOR THE CASE OF A FIXED FORMAT
C-----------------------------------------------------------------------
-------

      IF (IDEC.EQ.0) IDEC = LA+1

C-----------------------------------------------------------------------
-------
C     CHECK TO SEE IF AN EXPONENTIAL FORMAT IS USED IN THE STRING
C-----------------------------------------------------------------------
-------

      IF ((IE.EQ.0).AND.(ID.EQ.0)) THEN
            IEXP = 0
      ELSE
            IEXP = 1
      ENDIF

      IF (IEXP.EQ.1) THEN

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

```
-------
C     IF AN EXPONENTIAL FORMAT IS BEING USED SEPERATE THE NUMBER INTO
C      A MANTISSA AND AN EXPONENT.
C-----------------------------------------------------------------------
-------

            IA = MAX(IE,ID)
            MANT = STRT(1:IA-1)
            EXP  = STRT(IA+1:LA)
            IM = ILEN(MANT)
            IEE = ILEN(EXP)
            EX = 0.0

C-----------------------------------------------------------------------
-------
C     DETERMINE THE SIGN OF THE EXPONENT
C-----------------------------------------------------------------------
-------
            IF (EXP(1:1).EQ.'+') THEN
                  SEXP = 1.D0
                  ISE = 2
            ELSEIF (EXP(1:1).EQ.'-') THEN
                  SEXP = -1.D0
                  ISE = 2
            ELSE
                  SEXP = 1.D0
                  ISE = 1
            ENDIF

C-----------------------------------------------------------------------
-------
C     DETERMINE THE EXPONENT
C-----------------------------------------------------------------------
-------
            DO 400 I = IEE,ISE,-1
            IF (CTOD(EXP(I:I),RX)) THEN
                  IF (I.NE.0) EX = RX*10.D0**(IEE-I)+EX
            ELSE
                  RETURN
            ENDIF
400         CONTINUE
            EX = EX*SEXP
      ELSE

C-----------------------------------------------------------------------
-------
C     OTHERWISE THE EXPONENT = 1 AND THE ENTIRE NO. IS THE MANTISSA
C-----------------------------------------------------------------------
-------
            EX = 0.0D0
            MANT = STRT
            IM = ILEN(MANT)
      ENDIF

C-----------------------------------------------------------------------
-------
C     DETERMINE THE MANTISSA
C
```

```
C       FIRST DETERMINE THE PORTION BEFORE THE DECIMAL POINT
C-------------------------------------------------------------------
-------
      DO 200 I = IDEC-1,ISM,-1
            IF (CTOD(MANT(I:I),RX)) THEN
                  IF (I.NE.0) X = RX*10.D0**(IDEC-I-1)+X
            ELSE
                  RETURN
            ENDIF
200   CONTINUE

C-------------------------------------------------------------------
--
C     NEXT DETERMINE THE PORTION AFTER THE DECIMAL POINT
C-------------------------------------------------------------------
--
      IF (IDEC.NE.0) THEN
            DO 300 I = IDEC+1,IM
            IF (CTOD(MANT(I:I),RX)) THEN
                  IF (I.NE.0) X = RX*10.D0**(-I+IDEC)+X
            ELSE
                  RETURN
            ENDIF
300         CONTINUE
      ENDIF

C-------------------------------------------------------------------
--
C     EVALUATE THE FINAL NUMBER BASED ON THE SIGN AND THE MANTISSA
C      AND THE EXPONENT
C-------------------------------------------------------------------
--
      RR = X*SIGN*10.D0**INT(EX)
      STOD = .TRUE.

      RETURN
      END



      LOGICAL*1 FUNCTION CTOD (CH,XX)

C-------------------------------------------------------------------
--
C     REAL FUNCTION CTOD CONVERTS A CHARACTER TO A NUMBER (REAL*8)
C      IF THE CHARACTER IS NOT WITHIN '0-9' IT RETURNS WITH
C      CTOD = .FALSE.
C
C     Vinod Shah        7/13/90
C
C-------------------------------------------------------------------
--

      CHARACTER*1 CH
      REAL*8 XX

      CTOD = .FALSE.
      IF ((ICHAR(CH).GE.48).AND.(ICHAR(CH).LE.58)) THEN
```

```
            I = ICHAR(CH)-48
            XX = DBLE(I)
            CTOD = .TRUE.
      ENDIF
      RETURN
      END


      SUBROUTINE  LOWER (SLR1,SLR2)
C-----------------------------------------------------------------------
C
C     SUBROUTINE LOWERCONVERTS THE STRING SLR2 TO LOWER CASE AND
C      RETURNS IT IN SLR1
C
C     Vinod Shah  7/13/90
C
C-----------------------------------------------------------------------

      CHARACTER*(*) SLR1,SLR2
      INTEGER LEN2, LEN1, MAXLEN, I

      LEN1 = LEN(SLR1)
      LEN2 = LEN(SLR2)
      MAXLEN = MIN0 (LEN2, LEN1)
      DO 100 I = 1,MAXLEN
            IVAL = ICHAR(SLR2(I:I))
            IF ((IVAL.GE.65).AND.(IVAL.LE.90)) THEN
                  SLR1(I:I) = CHAR(IVAL+32)
            ELSE
                  SLR1(I:I) = SLR2(I:I)
            ENDIF
100   CONTINUE
      RETURN
      END


      subroutine scrpc2 (nu,fname3,pc,tc,rhoc,ow,dm,nc,inf,kij)
c-----------------------------------------------------------------------
-------
c
c   vinod shah     3/19/92
c
c   this subroutine is to be used only for binary mixtures
c
c   this subroutine reads an input file opened as unit # nu and sorts
c    out information on the input filename(s) and critical properties
c    of the two components that form the binary mixture.
c
c   the input file has to have a specified format shown below
c
c   filenames that have input data to be read in have to be preceded by
a
c    '$input' on a new line and the list terminated by a '$' on a new
line.
c    the limit on the number of filenames
c    that can be read is set by the declaration of the fname1 array in
```

```
the main program.
c    critical constants are preceded by '$critical-constants' and the
list terminated by
c     a '$' on a new line. constants for each component are preceded by
'component # id'
c     on a new line, where id is the component number (1 or 2). this has
to be followed by
c     one or more constants, each on a seperate line. the format for the
constants is
c
c      pc = 123
c        tc = 456
c        vc = 789
c        rhoc = 123
c        omega = 456
c      dpm = 789
c

c     e.g if the input file contains
c     12345678901234567890123456789O    this denotes the column no in the
file.
c     $input
c     vms/f1.dat
c     vms/f2.dat
c     f3.dat
c     $
c
       implicit double precision (a-h,o-z)
       character*50  fname3
       integer nu, kc
       real*8 pc(3), tc(3), rhoc(3), ow(3), dm(3), kij(3)
       dimension fname3(*)
       character*50 sinpu*78, sinp*78
       logical*1 finp, finc, finco , fkij
       logical*1 stod
       external ilen, lower, stod

       common /out2/ iflag, iwd, iwh, iwb, ww

       inf = 0
       iofil = 0
       inpg  = 0
       inpc  = 0
       kc = 0

c
c analyze the input script file
c
       do 5 j = 1,200
             sinp =
      $'                                                                '
             read (nu,'(a)',end=6) sinp
             call lower (sinpu, sinp)
c
c check to see if the line is not a comment statement
c
          if ((ilen(sinpu).ne.0).and.(sinpu(1:1).ne.'%')) then
```

```
c
c if the line is not a comment line  and the input file flag has
c  been set .....
c


            if ((fkij).and.(sinpu(1:1).ne.'$')) then
             kc = kc + 1
               il1 = ilen(sinpu)
               kij(kc) = 0.0
                    if (stod(sinpu(1:il1),val)) then
                         kij(kc) = real(val)
                    else
                   write(*,*)'Error reading kij'
                endif
            elseif ((fkij).and.(sinpu(1:1).eq.'$')) then
             fkij = .false.

            elseif ((finp).and.(sinpu(1:1).ne.'$')) then
                  if (ilen(sinpu).ne.0) then
                        inf = inf + 1
                        fname3(inf) = sinp
                  endif
            elseif ((finp).and.(sinpu(1:1).eq.'$')) then
                  if (inf.eq.0) then
                        inf = -1
                  endif
                  finp = .false.
            elseif ((finc).and.(sinpu(1:1).ne.'$')) then
               if ((index1(sinpu,'component')).ne.0) then
                  finco = .true.
                  il1 = ilen(sinpu)
                  rhoci = 0.0
                  ipo = index1(sinpu,'#')
                  if (ipo.ne.0) then
                     if (stod(sinpu(ipo+1:il1),val)) then
                        id = int(val)
                     else
                        write (*,*) ' error reading component # '
                     endif
                  else
                     write (*,*) ' error reading component statemant'
                  endif
               elseif ((ilen(sinpu).ne.0).and.(id.gt.0)) then
                  call crits2 (sinpu, pci, tci, rhoci, omi, dmi)
                  pc(id) = pci
                  tc(id) = tci
                  ow(id) = omi
                  dm(id) = dmi
                  rhoc(id) = rhoci
               endif
            elseif ((finc).and.(sinpu(1:1).eq.'$')) then
                  finc = .false.
            elseif ((index1(sinpu,'$kij')).ne.0) then
                  fkij = .true.
            elseif ((index1(sinpu,'$input')).ne.0) then
                  finp = .true.
            elseif ((index1(sinpu,'$critical-constants')).ne.0) then
```

```
                finc = .true.
            endif
          elseif (sinpu(1:1).eq.'%') then
c           write (*,*)sinp
          endif
5     continue
6     continue
      return
      end



      subroutine crits2 (strt, pc, tc, rhoc, om, dm)

c-----------------------------------------------------------------------
-------
c     vinod shah    11/16/90
c
c     the subroutine analyzes a string containing critical parameters
c
c
      implicit real*8 (a-h,o-z)
      character strt*(*), sts*80, snum*80, strt1*80
      logical*1 stod
      logical fil, open1
      save iflt
      external ilen, lower, stod
      data fil,open1/.false.,.false./
      data iflt /0/


      if (iflt .eq.0) then
           inquire (unit=55 ,opened=open1)
           if (open1.eqv..true.) then
              fil = .true.
              iflt = 1
           endif
      else
              iflt = 1
      endif

      do 10 j = 1,80
            sts(j:j) = ' '
            snum(j:j) = ' '
            strt1(j:j) = ' '
10    continue

      call lower (strt1,strt)

      il = ilen(strt1)
      if (il.eq.0) then
            return
      else
            sts = strt1(1:il)
      endif


      i1 = index(sts,'pc')
```

```
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'pc','pc'
            if (fil) write (55,20) 'pc','pc'
            return
      else
            snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            pc = (val)
      else
            write (*,30) 'pc','pc'
            if (fil) write (55,30) 'pc','pc'
            stop
      endif
      return
endif

i1 = index(sts,'tc')
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'tc','tc'
            if (fil) write (55,20) 'tc','tc'
            return
      else
            snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            tc = (val)
      else
            write (*,30) 'tc','tc'
            if (fil) write (55,30) 'tc','tc'
            stop
      endif
      return
endif

i1 = index(sts,'rhoc')
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'rhoc','rhoc'
            if (fil) write (55,20) 'rhoc','rhoc'
            return
      else
         snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            rhoc = (val)
      else
            write (*,30) 'rhoc','rhoc'
            if (fil) write (55,30) 'rhoc','rhoc'
            stop
      endif
      return
endif
```

```
i1 = index(sts,'vc')
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'vc','vc'
            if (fil) write (55,20) 'vc','vc'
            return
      else
         snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            vc = (val)
            if (rhoc.eq.0.0) rhoc = 1.0/vc
      else
            write (*,30) 'vc','vc'
            if (fil) write (55,30) 'vc','vc'
            stop
      endif
      return
endif

i1 = index(sts,'omega')
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'omega','omega'
            if (fil) write (55,20) 'omega','omega'
            return
      else
            snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            om = (val)
      else
            write (*,30) 'omega','omega'
            if (fil) write (55,30) 'omega','omega'
            stop
      endif
      return
endif


i1 = index(sts,'dpm')
if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
            write (*,20) 'dpm','dpm'
            if (fil) write (55,20) 'dpm','dpm'
            return
      else
            snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
            dm = (val)
      else
            write (*,30) 'dpm','dpm'
            if (fil) write (55,30) 'dpm','dpm'
```

```
                stop
            endif
            return
        endif


20      format(1x,'*** crits2 error -- error finding "=" following : ',
     $            a,/,1x,10x,a,4x,' not changed ')
30      format(1x,'*** crits2 error -- error reading number following : ',
     $            a,' = ',/,1x,10x,a,4x,' not changed ')

        return
        end



        subroutine inpdt3 (fname3, infil, nd, ires, idno)

c-----------------------------------------------------------------------
c  vinod shah 7/24/91
c
c
c reads the input files
c
c  on input :
c
c     infil : integer   : the number of input files to be read
c     ires  : logical*1 : true if residuals are to be printed
c                         false otherwise
c
c  on output :
c     nd    : integer    : the number of data points read in
c
c when reading the files the following points to be noted
c
c    a '%' sign in the 1st 5 columns of the line denotes a comment
c     line. the comment line(s) must appear before any valid data is
c     read in. the subroutine skip11 scans out any comment lines in
c     the data files. skip11 also returns "isc" or the scan-code to
c     which determines the format of the input data. the default value
c     is 3.
c

        implicit double precision (a-h,o-z)
        integer*4   ndmax
        parameter(ndmax=500)
        real*8 x1exp(ndmax), y1exp(ndmax), pexp(ndmax), texp(ndmax)
        real*8 x2exp(ndmax), y2exp(ndmax),idc(ndmax)
        character*(*) fname3(10)
        character*80 ssp
        logical*1 ires
        common /expdat/pexp,texp,x1exp,x2exp,y1exp,y2exp,idc

        ndn = nd + 1


c
c default value of isc
```

```
c
      isc = 1
      do k = 1,infil
          open( 20,file=fname3(k),status='old')
          write (*,85) fname3(k)
          write (55,85) fname3(k)
85    format(1x,' reading input file : ',a)

c
c skip the comment lines and print them to the screen and file-no 55
c
          call skip11 (20, 55 , 1, isc)

          do 10 i = 1,500
           if (isc.eq.1) then
                read (20,'(a)',end=99) ssp
               if (index(ssp,'%') .eq. 1) then
                    write (*,*) ssp
                  go to 10
                else
             backspace (20)
             read (20,*,end=99) texp(ndn), pexp(ndn)
     &          , x1exp(ndn),x2exp(ndn), y1exp(ndn), y2exp(ndn)
             idc(ndn) = idno
                 if (.not.ires) then
                     write (* ,'(1x,6(g12.5,1x:),i5)')texp(ndn)
     $,pexp(ndn),x1exp(ndn),x2exp(ndn),y1exp(ndn),y2exp(ndn)
                     write (55,'(1x,6(g12.5,1x:),i5)')texp(ndn)
     $,pexp(ndn),x1exp(ndn),x2exp(ndn),y1exp(ndn),y2exp(ndn)
                endif
                ndn = ndn + 1
                if (ndn.gt.ndmax) then
                    write (  *,20) ndmax
                    write (55,20) ndmax
                    goto 990
                endif
               endif
            endif
10        continue
99     close (20)
       end do

990      nd = ndn - 1
20     format(1x,' * * * warning: ndmax exceeded - rest of the data ',
      $ 'not read  * * * ')
       return
       end


      SUBROUTINE SKIPL1 (NIN, NOUT, NTOUT, ISC)

C-----------------------------------------------------------------------
C
C     VINOD SHAH   11/21/90
C
C     THE SUBROUTINE READS THE FILE WITH UNIT NO. NIN
C      AND SKIPS ANY LINES BEGINNING WITH A "%" IN THE COLUMNS 1-10
```

```
C      THE LINES READ CONTAINING A % SIGN ARE PRINTED ONTO THE
C      SCREEN AND NTOUT UNITS ... NOUT(1), NOUT(2), ,,, NOUT(NTOUT)
C
C     IT ALSO LOOKS FOR A NUMBER FOLLOWING " SCAN-CODE " AND RETURNS
C      IT IN ISC
C
C-----------------------------------------------------------------------

      DIMENSION NOUT(NTOUT)
      CHARACTER*78 SKIPS, SKIPSU, SNUM*40
      LOGICAL FIL, OPEN1
      LOGICAL*1 STOD
      REAL*8 VAL
      SAVE IFLT

      DATA FIL,OPEN1/.FALSE.,.FALSE./
      DATA IFLT /0/


      IF (IFLT .EQ.0) THEN
           INQUIRE (UNIT=55 ,OPENED=OPEN1)
           IF (OPEN1.EQV..TRUE.) THEN
               FIL = .TRUE.
               IFLT = 1
           ENDIF
      ELSE
               IFLT = 1
      ENDIF

      ISC = 1
      DO 10 I = 1,100
          READ (NIN,'(A)',ERR=90)SKIPS
          CALL LOWER (SKIPSU, SKIPS)
C-----------------------------------------------------------------------
C IF A "%" SIGN WAS FOUND IN THE FIRST TEN COLUMNS OF THE FILE
C  IT IS A COMMENT LINE
C-----------------------------------------------------------------------
          IF (INDEX(SKIPSU(1:10),'%').NE.0) THEN
            I1 = INDEX(SKIPSU,'SCAN-CODE')
            IL = ILEN(SKIPSU)
            IF (I1.NE.0) THEN
                IEQ = INDEX(SKIPSU,'=')
                IF (IEQ.EQ.0) THEN
                  WRITE (*,20) 'SCAN-CODE','SCAN-CODE'
                  IF (FIL) WRITE (55,20) 'SCAN-CODE','SCAN-CODE'
                ELSE
                  SNUM = SKIPSU(IEQ+1:IL)
                ENDIF
                IF (STOD(SNUM,VAL)) THEN
                  ISC = INT(VAL)
                ELSE
                    WRITE (*,30) 'SCAN-CODE','SCAN-CODE'
                    IF (FIL) WRITE (55,30) 'SCAN-CODE','SCAN-CODE'
                ENDIF
            ENDIF

            WRITE (*,'(1X,''%'',A)') SKIPS(1:IL)
            DO 15 K = 1,NTOUT
```

```
                WRITE (NOUT(K),'(''%'',A)') SKIPS(1:IL)
15          CONTINUE

            ELSE
                GOTO 90
            ENDIF

10      CONTINUE

C-----------------------------------------------------------------------
C DATA IS PRESENT IN THE LAST LINE THAT WAS READ, HENCE SET THE FILE
C  POINTER TO THE THAT LINE AND RETURN
C-----------------------------------------------------------------------

90      BACKSPACE(NIN)
20      FORMAT(1X,'*** SKIPL1 ERROR -- ERROR FINDING "=" FOLLOWING : ',
     $            A,/,1X,10X,A,4X,' NOT CHANGED ')
30      FORMAT(1X,'*** SKIPL1 ERROR -- ERROR READING NUMBER FOLLOWING : ',
     $            A,' = ',/,1X,10X,A,4X,' NOT CHANGED ')

        RETURN

C---------- END OF SUBROUTINE SKIPL1 ----------------------------------
-

        END



        subroutine prfmx2 (p, t, v, a, b, x, phic, ier, k)
        implicit real*8 (a-h,o-z)
        data r /8.31434d-3/
C calculates the fugacity coefficient of the mixture using the peng-
robinson
C equation of state
C       dimension a(nc), b(nc) , x(nc), phic(nc)
C currently nc = 3
        dimension a(3), b(3) , x(3)
        real*8 k(3), phic(3)


c        write(*,*)'in prfmx x=',x(1),x(2),x(3)

        sqr2 =dsqrt(2.d0)
C calculate the compressibility z
        prt = p/r/t
        z = prt*v
C calculate the mixture parameters
C  calculate aij


c        write(*,*)'a=',a(1),a(2),a(3)
c       write(*,*)'kij=',k(1),k(2),k(3)


        a12 = (1.d0-k(1))*dsqrt(a(1)*a(2))
        a13 = (1.d0-k(2))*dsqrt(a(1)*a(3))
        a23 = (1.d0-k(3))*dsqrt(a(2)*a(3))
```

```
c     write(*,*)'aij=',a12,a13,a23


      am = x(1)**2*a(1)+x(2)**2*a(2)+x(3)**2*a(3)
     &     +2.d0*x(1)*x(2)*a12+2.d0*x(1)*x(3)*a13
     &     +2.d0*x(2)*x(3)*a23
      bm = x(1)*b(1)+x(2)*b(2)+x(3)*b(3)
      oa = am*prt/r/t
      ob = prt*bm
C calculate the fugacity coefficient
      zb = dlog(z-ob)
      zb1 = dlog((z+(sqr2+1.d0)*ob)/(z+(1.d0-sqr2)*ob))
      zb2 = oa/2.d0/sqr2/ob
      phicl1 = b(1)/bm*(z-1) - zb -
     $        zb1*(2.d0*(x(1)*a(1)+x(2)*a12+x(3)*a13)/am -b(1)/bm)
     &      *zb2
      phicl2 = b(2)/bm*(z-1) - zb -
     $        zb1*(2.d0*(x(2)*a(2)+x(1)*a12+x(3)*a23)/am -b(2)/bm)
     &      *zb2
      phicl3 = b(3)/bm*(z-1) - zb -
     $        zb1*(2.d0*(x(3)*a(3)+x(1)*a13+x(2)*a23)/am -b(3)/bm)
     &      *zb2


c     mark 4
c      write(*,*)'am=',am
c      write(*,*)'bm=',bm
c      write(*,*)'zb=',zb
c      write(*,*)'zb1=',zb1
c      write(*,*)'zb2=',zb2
c      write(*,*)'phicl1=',phicl1
c      write(*,*)'phicl2=',phicl2
c      write(*,*)'phicl3=',phicl3
c     pause
c     end 4


C     phic(1) = phicl1
C     phic(2) = phicl2
c     phic(3) = phicl3


      phic(1) = dexp(phicl1)
      phic(2) = dexp(phicl2)
      phic(3) = dexp(phicl3)
      return
      end



      subroutine anasov(pr, t, roots, a, b, ndeg, fail)
      real*8 m, n, rvcub, rucub, norm, theta, root(3), error(3), r
      real*8 coeff(4), pr ,t, b, a, tol
      complex*16 roots(3), w, u, v, ucub, vcub
c     mark 0
c      real(8) m, n, rvcub, rucub, norm, theta, root(3), error(3), r
c      real(8) coeff(4), pr ,t, b, a, tol
```

```
c       double complex roots(3), w, u, v, ucub, vcub
        real*8 im1,im2,im3,im4,real1,real3,r1,r2,r3,i1,i2,i3
c       end 0
        logical*1 fail
          integer j

        r = 8.31434e-3
        w = (-0.5,3**0.5/2)
          root(1) = 0
          root(2) = 0
          root(3) = 0
        coeff(1) = 1.0
        coeff(2) = (b - r*t/pr)
        coeff(3) = (a/pr - 3.0*b**2.0 - 2.0*b*r*t/pr)
        coeff(4) = (b**3.0 + r*t*b**2.0/pr - a*b/pr)
          a1 = coeff(2)
          a2 = coeff(3)
          a3 = coeff(4)
        m = coeff(3) - 1.0/3.0*coeff(2)**2.0
        n = 2.0/27.0*coeff(2)**3.0 - 1.0/3.0*coeff(2)*coeff(3) + coeff(4)
          delta = 1.0/4.0*n**2.0+1.0/27.0*m**3.0
          if (delta .ge. 0) then
              rucub = -0.5*n + delta**0.5
              if (rucub .le. 0) then
                 u = -(-rucub)**(1.0/3.0)
              else
                 u = rucub**(1.0/3.0)
              endif
              rvcub = -0.5*n - delta**0.5
              if (rvcub .le. 0) then
                 v = -(-rvcub)**(1.0/3.0)
              else
                 v = rvcub**(1.0/3.0)
              endif
          else
              norm = (1.0/4.0*n**2+abs(delta))**0.5
              theta = acos(-0.5*n/norm)
        real1 = norm*cos(theta)
        im1 = norm*sin(theta)
        im2 = norm*sin(-theta)
        real3 = norm**(1.0/3.0)*cos(theta/3.0)
        im3 = norm**(1.0/3.0)*sin(theta/3.0)
        im4 = norm**(1.0/3.0)*sin(-theta/3.0)
              ucub = dcmplx(real1,im1)
              vcub = dcmplx(real1,im2)
              u    = dcmplx(real3,im3)
              v    = dcmplx(real3,im4)
          endif



          roots(1) = u + v - 1.0/3.0*coeff(2)
          roots(2) = w*u + w**2.0*v - 1.0/3.0*coeff(2)
          roots(3) = w**2.0*u + w*v - 1.0/3.0*coeff(2)
          t1 = roots(1)**3+a1*roots(1)**2+a2*roots(1)+a3
          t2 = roots(2)**3+a1*roots(2)**2+a2*roots(2)+a3
          t3 = roots(3)**3+a1*roots(3)**2+a2*roots(3)+a3
```

```
         do 10 i = 1, 3
            if (abs(dimag(roots(i))) .lt. 10d-8) then

               root(i) = dble(roots(i))
               tol = abs(root(i)*0.001)
               error(i) = root(i)**3+a1*root(i)**2+a2*root(i)+a3
               r1 = root(i)*1.1
               do 20 while(abs(error(i)) .ge. tol)
                  e1 = r1**3+a1*r1**2+a2*r1+a3
                  if ((error(i) .gt. 0.0) .and. (e1 .ge. error(i))) then
                     r1 = root(i)*0.9
                  elseif ((error(i) .lt. 0.0) .and.
     &                    (e1 .le. error(i))) then
                     r1 = root(i)*0.9
                  elseif ((e1*error(i)) .ge. 0.0) then
                     root(i) = r1
                     error(i) = e1
                     r1 = 1.1*root(i)
                  elseif ((e1*error(i)) .lt. 0) then
                     r2 = (r1 + root(i) )/2.0
                     e2 = r2**3+a1*r2**2+a2*r2+a3
                     if ((e2*e1) .ge. 0) then
                        r1 = r2
                     else
                        root(i) = r2
                        error(i) = e2
                     endif
                  endif
20             continue
               roots(i) = root(i)

            endif
          error(i) = roots(i)**3+a1*roots(i)**2+a2*roots(i)+a3
10        continue
         return
         end



         subroutine srtrp1 (roots, vv, vl, t, ndeg,
        $                          im, icc)
         implicit double precision (a-h,o-z)
         complex*16 roots(4), cr(4)
         common /debug/ idbgs, idbgr
         dimension rreal(4)
         rreal(1) = 0.0
         rreal(2) = 0.0
         rreal(3) = 0.0
         irf = 0
         vv = -1.0
         vl = -1.0
         icc = 0

c-----------------------------------------------------------------------
c sort out the real and complex roots
c  the real roots are stored in the array rreal
c
c-----------------------------------------------------------------------
```

```
      nreal = 0
      npos  = 0

      do 20 j = 1, ndeg
          r1 = dimag(roots(j))
          if (abs(r1).lt.1.d-15) then
            nreal = nreal + 1
            rreal(nreal) = dble(roots(j))
            if (rreal(nreal).gt.0) npos = npos + 1
          endif
20    continue
c-----------------------------------------------------------------------
c  sort the real roots in decreasing order
c
c-----------------------------------------------------------------------
      do 25 l = 2,nreal
          do 30 k = nreal,1,-1
            if (rreal(k).gt.rreal(k-1)) then
                temp = rreal(k-1)
                rreal(k-1) = rreal(k)
                rreal(k) = temp
            endif
30        continue
25    continue

            if (im.eq.3) then
                vv = rreal(1)
                vl = rreal(3)
            elseif (im.eq.1) then
                vv = rreal(1)
            elseif (im.eq.2) then
                  do lm = nreal,1,-1
                  if (rreal(lm).gt.0.0) then
                        vl = rreal(lm)
                        goto 89
                  endif
                  end do
89                continue
            endif
      return
      end



      subroutine nrfgp2 (pp, t, a, b, vv, vl, ierr)
      implicit double precision (a-h,o-z)
      complex*8 roots(4)
      real*8 fugv, fugl
      real*8 k0, k1
      logical*1 fail
      logical*4 fil, open1
      save iflt
      common /crit/ pc,tc,rhoc,zc,g,om
      external slvp1, srtrp1, fugp1
      data fil,open1/.false.,.false./
      data iflt /0/

        if (iflt .eq.0) then
```

```
            inquire (unit=100,opened=open1)
            if (open1.eqv..true.) then
                fil = .true.
                iflt = 1
            endif
          else
                iflt = 1
          endif
c
c sets the maximum no. of iterations
c
      imax = 1000
      pr = pp/0.8
      ndeg = 3
      nf = 1
      icn = 0
      id = 3
      do j = 1,imax
         ddiff = pr*1.0d-5
         call anasov(pr, t, roots, a, b, ndeg, fail)
         if (.not.fail) then
            call srtrp1 (roots, vv, vl, t, ndeg,id,icn)
         else
            write (*,200) 'nrfgp2'
            if (fil) write (100,200) 'nrfgp2'
            ierr = -1
            return
         endif
         if ((vv.lt.0.0).or.(vl.lt.0.0)) then
            write (*,201) 'nrfgp2'
            if (fil) write (100,201) 'nrfgp2'
            ierr = -1
            return
         endif
         call fugp1 (pr, t, vv, fugv, a, b, nf)
         if (nf.eq.0) then
            write (*,202) 'nrfgp2'
            if (fil) write (100,202) 'nrfgp2'
            return
         endif
         call fugp1 (pr, t, vl, fugl, a, b, nf)
         if (nf.eq.0) then
            write (*,202) 'nrfgp2'
            if (fil) write (100,202) 'nrfgp2'
            return
         endif
         func = real(fugv) - real(fugl)
         prd = pr + ddiff
         call anasov(prd, t, roots, a, b, ndeg, fail)
         if (.not.fail) then
            call srtrp1 (roots, vv1, vl1, t, ndeg,id,icn)
         else
             write (*,200) 'nrfgp2'
             if (fil) write (100,200) 'nrfgp2'
             ierr = -1
             return
         endif
         if ((vv1.lt.0.0).or.(vl1.lt.0.0)) then
```

```
            write (*,201) 'nrfgp2'
            if (fil) write (100,201) 'nrfgp2'
            ierr = -1
            return
         endif
         call fugp1 (prd, t, vv1, fugv1, a, b, nf)
         if (nf.eq.0) then
            write (*,202) 'nrfgp2'
            if (fil) write (100,202) 'nrfgp2'
            return
         endif
         call fugp1 (prd, t, vl1, fugl1, a, b, nf)
         if (nf.eq.0) then
            write (*,202) 'nrfgp2'
            if (fil) write (100,202) 'nrfgp2'
            return
         endif
         func1 = fugv1 - fugl1
         derf = (func1 - func)/ddiff
         pnew = pr - func/(derf+1.d-9)
         func2 = func/pr
         if (abs(func2).lt. 0.00001) then
              pp = pnew
              goto 100
         else
              if (pnew.lt.0.d0) then
                   pnew = pr/2.0
              endif
                 pr = pnew
         endif
      end do
100   continue
      if (j.ge.imax) then
         write (*,203) imax,'nrfgp2'
         if (fil) write (100,203) imax,'nrfgp2'
         ierr = -1
         return
      endif
200   format(1x,' * * * * spoly failed in ',a,' * * * * ')
201   format(1x,' * * no real valid root found in ',a,' * *')
202   format(1x,' * * * * fugacity failure in ',a,' * * * *')
203   format(1x,
     $      ' * * maximum # of iterations ',i4,' exceeded in ',a,' * *
')
      return
      end



      subroutine fugp1 (pr, t, v, fug, a, b, nf)
      implicit double precision (a-h,o-z)
      data g/8.31434d-3/
      z = pr*v/g/t
      bb = b*pr/g/t
      aa = a*pr/g**2/t**2
      ff1 = -aa/2.d0/dsqrt(2.d0)/bb
      ff3 = z - 1.d0 - dlog(z-bb)
      ftmp = ff1*dlog((v+2.414d0*b)/(v-0.414d0*b)) + ff3
```

```
      fug = dexp(ftmp)*pr
      return
      end



      subroutine funpr (ndata,nf,kij)
      implicit real*8 (a-h,o-z)
      parameter(ndmax=500)
      real*8 xlexp(ndmax), y1exp(ndmax), pexp(ndmax), texp(ndmax)
      real*8 x2exp(ndmax), y2exp(ndmax), idc(ndmax)
      real*8 eqk(3)
      real*8 vc(3), tc(3), ow(3), pc(3), dm(3), rhoc(3), kij(3)
      real*8 x(3), y(3)
      real*8 a(3), b(3)
      common /expdat/pexp,texp,x1exp,x2exp,y1exp,y2exp,idc
      common /critp/pc,vc,tc,ow,dmr
      common /res/ ixd
      data r /8.31434d-3/


      nf = 1
      dpt = 0.d0
      dp1 = 0.d0
      dps = 0.d0
      dxt2 = 0.d0
      dyt1 = 0.d0
      dyt2 = 0.d0

      write(*,*)'Results'
                write ( *,105)
                write (55,105)
105     format(1x,'Temp    P        x1       x2calc   x2exp    y1calc
     &y1exp    y2calc   y2exp' )
C
C     $,'    y2cal    id ' )


      nad = 0
c calculate the constants for the equation of state
c     prb  = 0.0778d0*r*tc/pc
c     praa = 0.45724d0*r**2*tc**2/pc
      oka1  = 0.37464 + 1.54226*ow(1) - 0.26992*ow(1)**2
      ob1 = 0.0778d0
      oa1 = 0.45724d0
      bc1 = ob1*r*tc(1)/pc(1)
      ac1 = oa1*r**2*tc(1)**2/pc(1)
      oka2  = 0.37464 + 1.54226*ow(2) - 0.26992*ow(2)**2
      ob2 = 0.0778d0
      oa2 = 0.45724d0
      bc2 = ob2*r*tc(2)/pc(2)
      ac2 = oa2*r**2*tc(2)**2/pc(2)
      oka3  = 0.37464 + 1.54226*ow(3) - 0.26992*ow(3)**2
      ob3 = 0.0778d0
      oa3 = 0.45724d0
      bc3 = ob2*r*tc(3)/pc(3)
      ac3 = oa2*r**2*tc(3)**2/pc(3)
```

```
      do n = 1, ndata
          x(1) = x1exp(n)
          x(2) = x2exp(n)
          y(1) = y1exp(n)
          y(2) = y2exp(n)
          p = pexp(n)
          t = texp(n)
          y(3) = 1.d0 - y(1) - y(2)
          x(3) = 1.d0 - x(1) - x(2)

          tr1 = t/tc(1)
          alpha1 = (1.d0 + oka1*(1.d0 - dsqrt(tr1)))**2
            a(1) = alpha1*ac1
          b(1) = bc1
          tr2 = t/tc(2)
          alpha2 = (1.d0 + oka2*(1.d0 - dsqrt(tr2)))**2
            a(2) = alpha2*ac2
          b(2) = bc2
          tr3 = t/tc(3)
          alpha3 = (1.d0 + oka3*(1.d0 - dsqrt(tr3)))**2
            a(3) = alpha3*ac3
          b(3) = bc3
      if (x(1).eq. 1.0) then
         call nrfgp2 (p, t, a(1), b(1), vv, vl, ier)
         y(1) = 1.d0
         y(2) = 0.d0
         y(3) = 0.d0
         if (ierr.lt.0) then
            idc(n) = -1
            goto 98
         endif
      elseif (x(2).eq. 1.0) then
         call nrfgp2 (p, t, a(2), b(2), vv, vl, ier)
         y(1) = 0.d0
         y(2) = 1.d0
         y(3) = 0.d0
         if (ierr.lt.0) then
            idc(n) = -1
            goto 98
         endif
      elseif (x(1).eq.x(2) .and. x(1).eq. 0.0) then
         call nrfgp2 (p, t, a(3), b(3), vv, vl, ier)
         y(1) = 0.d0
         y(2) = 0.d0
         y(3) = 1.d0
         if (ierr.lt.0) then
            idc(n) = -1
            goto 98
         endif
      else
         call nreqp1 (p, t, x, a, b, y, eqk, vv, vl, ierr,kij)
         if (ierr.lt.0) then
            idc(n) = -1
            goto 98
         endif
      endif

c      write(*,*)'stop here'
```

```
C     write(*,*)'t=',t
c     write(*,*)'x=',x(1),x(2)
c     write(*,*)'p=',p,pexp(n)
c     write(*,*)'y=',y(1),y(2)
c     write(*,*)'yexp=',y1exp(n),y2exp(n)
C     pause


          write (*,100) t,p, x(1),x(2),x2exp(n)
     &                     ,y(1),y1exp(n),y(2),y2exp(n)
          write (55,100) t,p,x(1),x(2),x2exp(n)
     &                     ,y(1),y1exp(n),y(2),y2exp(n)

          dxt2 = dabs(x(2)-x2exp(n)) + dxt2
          dyt1 = dabs(y(1)-y1exp(n)) + dyt1
          dyt2 = dabs(y(2)-y2exp(n)) + dyt2
C        endif
100     format(1x,f6.2,1x,f6.2,1x,3(f8.4,1x),4(f8.4,1x))
98      continue
        end do

C          if (ixd.ne.0) then
               dxt2 = dxt2/dble(ndata)
               dyt1 = dyt1/dble(ndata)
               dyt2 = dyt2/dble(ndata)
104     format(1x,10(g11.4,2x:))
               write (*,102)
               write (55,102)
102            format(4x,
     $'ndata   x2aad        y1aad       y2aad   ')

               write (*,103) ndata, dxt2, dyt1, dyt2
               write (55,103) ndata, dxt2, dyt1, dyt2
C          endif
103   format(2x,i5,3x,g12.5,3(1x,g11.4:))
      return
      end


      subroutine nreqp1 (p, t, x, a, b, y, eqk, vv, vl, ierr,kij)
      implicit real*8 (a-h,o-z)
      dimension a(3), b(3), x(3), eqk(3), y(3), kij(3)
C solves the equation sum kixi - 1 = 0 using the newton raphson method
C maximum number of iterations
      ierr = 0
      imax = 300000
      p1 = p
      scale = 1.d0
      isd = 0
        x2 = x(2)
        cyc = 0
        x2old1 = 0.0
        x2old2 = x(2)
      yold1 = y(1)
      yold2 = y(2)
      ups = 0.005d0
      dps = 0.005d0
      fold = 1.d0
```

```
c      write(*,*)'x=',x(1),x(2),x(3)
c      write(*,*)'y=',y(1),y(2),y(3)

      do j = 1, imax

      count2 = count2 + 1

            if (x(2) .gt. 1.0) then
             x(2) = 1.0
          endif
          if (x(2) .eq. 0.0) then
             x(2) = 0.1
          endif
          x(3) = 1.0-x(1)-x(2)
        y(1) = yold1/1.00
        y(2) = yold2/1.00
        y(3) = 1.0-y(1)-y(2)


        do k = 1, imax

      count1 = count1+1

            call bubpr2 (p1, t, x, a, b, y, eqk, vv, vl, ier, kij)
C           call bubpr2 (p1, t, y, a, b, x, eqk, vl, vv, ier, kij)
            if (ier.lt.0) then
               ierr = -1
               return
            endif
            if (k.eq.1) then
                ysum = eqk(1)*x(1) + eqk(2)*x(2) + eqk(3)*x(3)
                y(1) = eqk(1)*x(1)/ysum
                y(2) = eqk(2)*x(2)/ysum
                y(3) = eqk(3)*x(3)/ysum
              ysumo = ysum
C               xsum = eqk(1)*y(1) + eqk(2)*y(2) + eqk(3)*y(3)
C               x(1) = eqk(1)*y(1)/ysum
c               x(2) = eqk(2)*y(2)/ysum
c               x(3) = eqk(3)*y(3)/ysum
         elseif (k.eq.imax) then
                 write (*,*) ' Maximum number of iterations exceeded '
            ierr = -2
                return
            else
                ysum = eqk(1)*x(1) + eqk(2)*x(2) + eqk(3)*x(3)
C               xsum = eqk(1)*y(1) + eqk(2)*y(2) + eqk(3)*y(3)
                if (dabs(ysum-ysumo).lt. 5.d-5) then
C               if (dabs(xsum-xsumo).lt.5.d-5) then
                  goto 91
                else
                y(1) = eqk(1)*x(1)/ysum
                y(2) = eqk(2)*x(2)/ysum
                y(3) = eqk(3)*x(3)/ysum
                ysumo = (y(1) + y(2) + y(3))*ysum
c               x(1) = eqk(1)*y(1)/ysum
c               x(2) = eqk(2)*y(2)/ysum
c               x(3) = eqk(3)*y(3)/ysum
```

```
c                xsumo = (x(1) + x(2) + x(3))*xsum
                 endif
              endif
           end do
91          continue

            if (dabs(ysum-1.d0).lt. 1.0d-7 .and.
     &             dabs(x2old1-x2old2).lt. 1.0d-8) then
       dx = x2old2-x2old1
c      write(*,*) "ysum=",ysum
c      write(*,*) "dx=",dx
c      write(*,*) "count1",count1
c      write(*,*) "count2",count2
c      count2 = 0.0
c      pause


       count1 = 0.0
c           if (dabs(xsum-1.d0).lt.1.0d-6) then
              goto 92
            endif
            f1 = ysum-1.d0
            if (j.eq.imax) then
                write (*,*) ' Maximum number of iterations exceeded '
                write (*,*) t,p
                ierr = -2
                return
             elseif (j .eq. 1) then
             x(2) = x(2)*1.1
             fold = f1
            elseif (j .gt. 1) then
                  if (cyc .eq. 0) then
                      if (f1*fold .lt. 0.0) then
                        cyc = 1
                        if (x(2) .gt. x2) then
                          x2old1 = x2
                          x2old2 = x(2)
                          x(2) = (x2old1+x2old2)/2.0
                          fold1 = fold
                          fold2 = f1
                        else
                          x2old1 = x(2)
                          x2old2 = x2
                          x(2) = (x2old1+x2old2)/2.0
                          fold1 = f1
                          fold2 = fold
                        endif
                      elseif (x(2) .gt. x2) then
                         x(2) = x2/(x(2)/x2*1.1)
                      else
                         x(2) = x2*(x2/x(2)*1.1)
                      endif
                      elseif (cyc .eq. 1) then
                     if (f1*fold1 .lt. 0.0) then
                         x2old2 = x(2)
                         fold2 = f1
                         x(2) = (x2old1+x2old2)/2.0
                      elseif (f1*fold2 .lt. 0.0) then
```

```
                        x2old1 = x(2)
                        fold1 = f1
                        x(2) = (x2old1+x2old2)/2.0
                     else
                        write(*,*) ' Something wrong 1 ! '
                        pause
                     endif
                  else
                     write(*,*) ' Something wrong 2 ! '
                     pause
                  endif
               endif

c     write(*,*) "x(2)=",x(2)
c     write(*,*) "cyc=",cyc
c     pause

      end do
92    return
      end


      subroutine bubpr2 (p, t, x, a, b, y, eqk, vv, vl, ier, k)
      implicit real*8 (a-h,o-z)
      real*8 phicl(3), phicv(3), k(3)
      dimension a(3), b(3), x(3), y(3), eqk(3)
      logical*1  fail
      complex*16 roots(3)
C   P is the pressure (guess on input)
C   x(i) is the liquid mole fraction
C   y(i) is the vapor mole fraction (guess on input)
C   eqk(i) is the equilibrium K value on return
C   Uses the peng robinson equation of state
C     calculate the mixture parameters for the liquid phase
      a12 = (1.d0-k(1))*dsqrt(a(1)*a(2))
      a13 = (1.d0-k(2))*dsqrt(a(1)*a(3))
      a23 = (1.d0-k(3))*dsqrt(a(2)*a(3))
      am = x(1)**2*a(1)+x(2)**2*a(2)+x(3)**2*a(3)
     &     +2.d0*x(1)*x(2)*a12+2.d0*x(1)*x(3)*a13
     &     +2.d0*x(2)*x(3)*a23
      bm = x(1)*b(1)+x(2)*b(2)+x(3)*b(3)
C  Solve the peng-robinson equation of state for the liquid phase
      call anasov(p, t, roots, am, bm, 3, fail)
      if (.not.fail) then
            call srtrp1 (roots, vvo, vl, t, 3, 2 , icc)
            if (icc.ge.0) then

c     write(*,*)'kij=!',k(1),k(2),k(3)
c      write(*,*)'in bub x=',x(1),x(2),x(3)
c      write(*,*)'in bub y=',y(1),y(2),y(3)


               call prfmx2 (p, t, vl, a, b, x, phicl, ier, k)

c     write(*,*)'stop 2 !'

            else
               write (*,*) 'Error in sorting roots '
```

```
                ier = -1
                return
              endif
      else
              write (*,*) ' Error in finding roots'
              ier = -3
              return
      endif

C   Calculate the fugacity coeff of the vapor phase

C     calculate the mixture parameters for the vapor phase
      a12 = (1.d0-k(1))*dsqrt(a(1)*a(2))
      a13 = (1.d0-k(2))*dsqrt(a(1)*a(3))
      a23 = (1.d0-k(3))*dsqrt(a(2)*a(3))
      am = y(1)**2*a(1)+y(2)**2*a(2)+y(3)**2*a(3)
     &     +2.d0*y(1)*y(2)*a12+2.d0*y(1)*y(3)*a13
     &     +2.d0*y(2)*y(3)*a23
      bm = y(1)*b(1)+y(2)*b(2)+y(3)*b(3)
C  Solve the peng-robinson equation of state for the vapor phase

c      write(*,*)'stop 3 !'

                call anasov(p, t, roots, am, bm, 3, fail)
      if (.not.fail) then
         call srtrp1 (roots, vv, vlo, t, 3, 1 , icc)
         if (icc.ge.0) then
            call prfmx2 (p, t, vv, a, b, y, phicv, ier, k)

c        Write(*,*)'vv,vl=',vv,vl
c        pause

c     mark 5
c      write(*,*)'stop 4 !'
c      write(*,*)'phicl1=',phicl(1)
c      write(*,*)'phicl2=',phicl(2)
c      write(*,*)'phicl3=',phicl(3)
c      write(*,*)'phicv1=',phicv(1)
c      write(*,*)'phicv2=',phicv(2)
c      write(*,*)'phicv3=',phicv(3)
c     end 5
         else
            write (*,*) 'Error in sorting roots '
              ier = -2
              return
         endif
      else
            write (*,*) ' Error in finding roots'
            ier = -3
            return
      endif
c calculate the equilibrium ratios


C           if ((phicl(1)-phicv(1)) .gt. 100000.0) then
C                 eqk(1) = 1000000000.0
c      else
c          eqk(1) = dexp(phicl(1)-phicv(1))
```

```
c     endif
c           if ((phicl(2)-phicv(2)) .gt. 100000.0) then
c                 eqk(2) = 1000000000.0
c      else
c          eqk(2) = dexp(phicl(2)-phicv(2))
c     endif
c           if ((phicl(3)-phicv(3)) .gt. 100000.0) then
c                 eqk(3) = 1000000000.0
c      else
c          eqk(3) = dexp(phicl(3)-phicv(3))
c     endif
      eqk(1) = phicl(1)/phicv(1)
      eqk(2) = phicl(2)/phicv(2)
      eqk(3) = phicl(3)/phicv(3)

C     write(*,*)'stop 5 !'

      return
      end
```

C.3 Flash Calculations Using Mass Balances and Ternary Code

Filename	**Description**
h2o-c2oh-co2.in	ternary system input file with kij
terpr.in	input and output designation file
isotf.f	main code

```
%
$INPUT
h2o-c2oh-co2.1
$
$kij
%k12
%-0.15618
%-0.112165
%-.04928
-0.124
%k13
0.114
%0.0952
%.1108
%k23
%0.058756
0.103256
%0.0852
$
$critical-constants
component # 1
pc = 22.12
vc = 0.0571
tc = 647.3
omega = 0.344
dpm = 1.8
component # 2
pc = 6.14
vc = 0.1671
tc = 513.9
omega = 0.644
dpm = 1.7
component # 3
pc = 7.38
tc = 304.1
vc = .0939
omega = 0.239
dpm = 0.0
$
```

```
$input-file
h2o-c2oh-co2.in
$
$output-file
test.out
$
```

```
      Program isotflash
c-----------------------------------------------------------------------
c     Main program, handle the input, output and call the subroutine,
c     flash, to do the flash vaporization
c-----------------------------------------------------------------------
      real*8 vc(3), tc(3), ow(3), pc(3), dm(3), rhoc(3), kij(3)
      parameter(ndmax=500)
      real*8 x1exp(ndmax), y1exp(ndmax), pexp(ndmax), texp(ndmax)
      real*8 x2exp(ndmax), y2exp(ndmax), idc(ndmax)
      real*8 fg, fl, xf(3), yf(3), a, x(3), y(3), t, p, k(3)
      real*8 fgg, fll
      integer infil, iofil, isl, inf, key, erf
      character sinp*90, sinpu*90
      character*50 fname1,fname2,fname3(10)
      logical finp, fout
      common /dbg1/ ndbg1, ndbg2
      common /res/ ixd
      common /expdat/pexp,texp,x1exp,x2exp,y1exp,y2exp,idc
      common /critp/pc,vc,tc,ow
      external cfunpr, alpha, calpha, beta
      data r /8.31434d-3/
c-----------------------------------------------------------------------
--
c     Read main input file 'terpr.in'
c-----------------------------------------------------------------------
--
      open (unit=10,file='terpr.in',status='old')
      do 5 j = 1,50
        sinp =
     &'                                                              '
        read (10,'(a)',end=6) sinp
        sinpu = sinp
        if ((ilen(sinp).ne.0).and.(sinp(1:1).ne.'%')) then
            if ((finp).and.(sinp(1:1).ne.'$')) then
              if (ilen(sinp).ne.0) then
                infil = infil + 1
                  fname1 = sinpu
              endif
            elseif ((finp).and.(sinp(1:1).eq.'$')) then
              if (infil.eq.0) then
                  write(*,*) 'Please specify input file!!'
              stop
              endif
              finp = .false.
            elseif ((fout).and.(sinp(1:1).ne.'$')) then
              if (ilen(sinp).ne.0) then
                fname2 = sinpu
                iofil = 1
              endif
            elseif ((fout).and.(sinp(1:1).eq.'$')) then
              if (iofil.eq.0) then
                  write(*,*) 'Please specify output file!!'
              stop
              endif
              fout = .false.
          elseif ((index1(sinp,'$input-file')).ne.0) then
              finp = .true.
          elseif ((index1(sinp,'$output-file')).ne.0) then
```

```
              fout = .true.
            endif
          elseif (sinp(1:1).eq.'%') then
          endif
5         continue
6       continue
c-----------------------------------------------------------------------
--
c     write to the output file
c-----------------------------------------------------------------------
--
        open (55,file=fname2,status='unknown')
        write (*,*)  ' ********** isotf ***************** '
        write (55,*)' ********** isotf ***************** '
        rewind (10)
        do 7 j = 1,100
          read (10,'(a)',end=8) sinp
          isl = ilen(sinp)
          write (*,'(1x,a)') sinp(1:isl)
          write (55,'(a)') sinp(1:isl)
7       continue
8       close (10)
        open (unit=65,file=fname1,status='old')
        ncomp = 3
c-----------------------------------------------------------------------
--
c     read system input file
c-----------------------------------------------------------------------
--
        call scrpc2 (65,fname3,pc,tc,rhoc,
       &             ow, dm, inf, kij )
        write(*,*)  'k12 =',kij(1)
        write(55,*) 'k12 =',kij(1)
        write(*,*)  'k13 =',kij(2)
        write(55,*) 'k13 =',kij(2)
        write(*,*)  'k23 =',kij(3)
        write(55,*) 'k23 =',kij(3)
        do ii = 1,3
          vc(ii) = 1.d0/rhoc(ii)
          write (*,'(1x,'' component # '',i3)') ii
          write (*,'(1x,'' critical density     '',g14.7)') rhoc(ii)
          write (*,'(1x,'' critical volume      '',f10.7)') vc(ii)
          write (*,'(1x,'' critical pressure    '',f10.5)') pc(ii)
          write (*,'(1x,'' critical temperature '',f10.3)') tc(ii)
          write (*,'(1x,'' acentric factor       '',f10.5,/)') ow(ii)
          write (55,'(1x,'' component # '',i3)') ii
          write (55,'(1x,'' critical density     '',g14.7)') rhoc(ii)
          write (55,'(1x,'' critical volume      '',f10.7)') vc(ii)
          write (55,'(1x,'' critical pressure    '',f10.5)') pc(ii)
          write (55,'(1x,'' critical temperature '',f10.3)') tc(ii)
          write (55,'(1x,'' acentric factor      '',f10.5,/)') ow(ii)
        end do
        close (65)
        write(*,*) "Please input the Pressure!! (Mpa)"
        read(*,*) p
        write(*,*) "Please input the Temperature!! (K)"
        read(*,*) t
        write(*,*) "Please input the max flowrate of gas stream! (ml/min)"
```

```
      read(*,*) fgu
      write(*,*) "Please input the min flowrate of gas stream! (ml/min)"
      read(*,*) fgl
      write(*,*) "Please input the flowrate of liquid stream! (ml/min)"
      read(*,*) fll
      write(*,*) "Please input the composition of gas stream! (wt% CO2)"
      read(*,*) cg
      write(*,*) "Please input the composition of liquid stream!
      &(vol% EtOH)"
      read(*,*) cl
      cg = cg/100.0
      cl = cl/100.0
      dfg = (fgu-fgl)/10.0
      app = 0.45724*r**2*tc(3)**2/pc(3)
      bpp = 0.07780*r*tc(3)/pc(3)
      cta = app*(1+(0.37464+1.54226*ow(3)-0.26992*ow(3)**2)*
      &      (1-(t/tc(3))**0.5))**2
      vbaro = 0.094
      do 10 i = 1,10000
        vbar = (r*t-cta*(vbaro-bpp)/(vbaro**2+2*bpp*vbaro-bpp**2))/p+bpp
        dv = abs(vbar-vbaro)/vbaro
        if (dv .lt. 0.00001) goto 10
        vbaro = vbar
        if (i .eq. 10000) then
          write(*,*) "No solution for Vbar!!"
          pause
        endif
10    continue
      open (55,file=fname2,status='unknown')
      write(*,*)
      write(*,*)
      write(55,*)
      write(55,*)
      write(*,*)  "Constant P, T flash vaporization!!"
      write(55,*) "Constant P, T flash vaporization!!"
      write(*,*)
      write(55,*)
      write(*,*)  "Pressure (Mpa) =",p
      write(55,*) "Pressure (Mpa) =",p
      write(*,*)  "Temperature (K) =",t
      write(55,*) "Temperature (K) =",t
      write(*,*) "Flowrate of liquid stream (ml/min) =",fll
      write(55,*) "Flowrate of liquid stream (ml/min) =",fll
      write(*,*) "Composition of gas stream! (wt% CO2) =",cg
      write(55,*) "Composition of gas stream! (wt% CO2) =",cg
      write(*,*) "Composition of liquid stream! (vol% EtOH) =",cl
      write(55,*) "Composition of liquid stream! (vol% EtOH) =",cl
      write(*,*)
      write(55,*)
      write(*,*)
      write(55,*)
      write(*,*)
      write(55,*)
      do 30 j=1,11
        fgg = fgl+(j-1)*dfg
        write(*,*) "Flowrate of gas stream (ml/min) =",fgg
        write(55,*) "Flowrate of gas stream (ml/min) =",fgg
        fg = fgg/vbar/1000.0
```

```
        yf(1) = ((1-cg)/18.0)/((1-cg)/18.0+cg/44.0)
        yf(2) = 0.0
        yf(3) = 1-yf(1)
        fl = fll*cl*0.7893/46.0+fll*(1-cl)*1.0/18.0
        xf(1) = (fll*(1-cl)*1.0/18.0)/fl
        xf(2) = 1.0-xf(1)
        xf(3) = 0.0
        call flash(key,fg,fl,xf,yf,a,x,y,t,p,k,erf,kij)
        write(*,*) "a,erf =",a,erf
        write(55,*) "a,erf =",a,erf
        write(*,*) "x =",x(1),x(2),x(3)
        write(55,*) "x =",x(1),x(2),x(3)
        write(*,*) "y =",y(1),y(2),y(3)
        write(55,*) "y =",y(1),y(2),y(3)
        write(*,*) "k =",k(1),k(2),k(3)
        write(55,*) "k =",k(1),k(2),k(3)
        alp = k(2)/k(1)
        write(*,*) "Alpha =",alp
        write(55,*) "Alpha =",alp
        write(*,*)
        write(55,*)
        write(*,*)
        write(55,*)
        write(*,*)
        write(55,*)
        write(*,*)
        write(55,*)
        write(*,*)
        write(55,*)
30      continue
        write(*,*) "Job finished!!"
        write(55,*) "Job finished!!"
        close (55)
        stop
        END


        INTEGER FUNCTION ILEN (STRW)
c-----------------------------------------------------------------------
--
c     INTEGER FUNCTION ILEN DETERMINE THE TRUE LENGTH OF THE STRING,
c     i.e. IT NEGLECTS ANY TRAILING SPACES IN THE STRING.
c     Vinod Shah   7/13/90
c-----------------------------------------------------------------------
--
        CHARACTER*(*) STRW
        INTEGER L, I
        L = LEN(STRW)
        DO 100 I = L,1,-1
          IF ((STRW(I:I).NE.' ').AND.(STRW(I:I).NE.CHAR(09)))
     &  GOTO 110
100     CONTINUE
110     ILEN = I
        RETURN
        END
```

```
      INTEGER*4 FUNCTION INDEX1 (SSR1, SSR2)
      CHARACTER*(*) SSR1, SSR2
c-----------------------------------------------------------------------
--
c     INDEX1  - FINDS THE LOCATION OF A SUBSTRING IN A STRING
c     SSR1 IS THE STRING IN WHICH THE SUBSTRING SSR2 IS TO BE FOUND
c     Vinod Shah                Final version   7/12/90
c     Note that ssr2 must not have unnecessary leading blanks
c-----------------------------------------------------------------------
--
      LOGICAL FOUND
      INTEGER IL, I, L, I1, I2
      INDEX1 = 0
      FOUND = .FALSE.
c-----------------------------------------------------------------------
c     FIND THE LENGTH OF SSR1 NEGLECTING ANY TRAILING BLANKS
c-----------------------------------------------------------------------
--
      L = LEN(SSR1)
      DO I = L,1,-1
        IF (SSR1(I:I).NE.' ') GOTO 110
      END DO
110   I1 = I
c-----------------------------------------------------------------------
--
c     FIND THE LENGTH OF SSR2 NEGLECTING ANY TRAILING BLANKS
c-----------------------------------------------------------------------
--
      L = LEN(SSR2)
      DO I = L,1,-1
        IF (SSR2(I:I).NE.' ') GOTO 120
      END DO
120   I2 = I
c-----------------------------------------------------------------------
--
c     IF THE LENGTH OF SSR2 IS GRATER THAN SSR1 RETURN AS A COMPARISION
c     CANNOT BE DONE
c-----------------------------------------------------------------------
--
      IF (I2.GT.I1) RETURN
      IL = 1
      DO I = 1, I1
        IF ((SSR1(I:I).EQ.SSR2(IL:IL)).AND.(.NOT.FOUND)) THEN
          FOUND = .TRUE.
          IL = IL + 1
          INDEX1 = I
        ELSEIF ((SSR1(I:I).EQ.SSR2(IL:IL)).AND.(FOUND)) THEN
            IL = IL + 1
        ELSE
            FOUND = .FALSE.
            IL = 1
            INDEX1 = 0
        ENDIF
        IF ((IL.EQ.(I2+1)).AND.(FOUND)) RETURN
      END DO
      IF (IL.LT.(I2+1)) INDEX1 = 0
      RETURN
```

```
      END


      LOGICAL*1 FUNCTION STOD (STR,RR)
c-----------------------------------------------------------------------
--
c     THIS SUBROUTINE CONVERTS A CHARACTER STRING NUMBER TO A
c     REAL*8 NUMBER.
c     STR  : A CHARACTER STRING CONTAINING THE NUMBER
c     RR   : A REAL*8 NO. IN WHICH THE VALUE IS RETURNED
c     STOD : A LOGICAL VARIABLE WHICH IS RETURNED .FALSE. IF AN ERROR
c            WAS FOUND DURING THE CONVERSION.
c     Vinod Shah           10/21/90
c-----------------------------------------------------------------------
--
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*(*) STR
      REAL*8 X,RR
      CHARACTER STRT*30,MANT*30,EXP*5
      LOGICAL*1 CTOD
      STOD = .FALSE.
      X = 0.D0
c-----------------------------------------------------------------------
--
c     GET THE LENGTH OF STR AS DEFINED IN THE CALLING PROGRAM
c-----------------------------------------------------------------------
--
      L = LEN(STR)
c-----------------------------------------------------------------------
--
c     DETERMINE (IF ANY) THE NUMBER OF LEADING BLANKS &/OR TABS
c-----------------------------------------------------------------------
--
      DO 50 I=1,L
        IF ((STR(I:I).NE.' ').AND.(STR(I:I).NE.CHAR(09)))
     &  GOTO 70
50    CONTINUE
c-----------------------------------------------------------------------
--
c     GET THE ACTUAL LENGTH OF THE STRING. THAT IS NEGLECT TRAILING
c     BLANKS
c-----------------------------------------------------------------------
--
70    LA = ILEN(STR(I:L))
      IF (LA.EQ.0) RETURN
      STRT = STR(I:L)
c-----------------------------------------------------------------------
--
c     CHECK TO SEE IF THE FIRST CHARACTER IN THE STRING IS A SIGN
c     DESCRIPTOR. IF SO NOTE THE SIGN
c-----------------------------------------------------------------------
--
      IF (STRT(1:1).EQ.'-') THEN
        SIGN = -1.D0
        ISM = 2
      ELSEIF (STRT(1:1).EQ.'+') THEN
        SIGN = +1.D0
```

```
        ISM = 2
      ELSE
        ISM = 1
        SIGN = 1.D0
      ENDIF
c-----------------------------------------------------------------------
--
c     FIND OUT THE LOCATION OF 'E' OR 'D' AND A DECIMAL POINT IN THE
c     STRING
c-----------------------------------------------------------------------
--
      IE1 = INDEX(STRT(1:LA),'E')
      IE2 = INDEX(STRT(1:LA),'e')
      IE  = MAX0(IE1,IE2)
      ID1 = INDEX(STRT(1:LA),'D')
      ID2 = INDEX(STRT(1:LA),'d')
      ID  = MAX0(ID1,ID2)
      IDEC = INDEX(STRT(1:LA),'.')
c-----------------------------------------------------------------------
--
c     IF A DECIMAL POINT IS ABSENT IT IS ASSUMED TO BE AT THE END
c     OF THE STRING FOR THE CASE OF A FIXED FORMAT
c-----------------------------------------------------------------------
--
      IF (IDEC.EQ.0) IDEC = LA+1
c-----------------------------------------------------------------------
--
c     CHECK TO SEE IF AN EXPONENTIAL FORMAT IS USED IN THE STRING
c-----------------------------------------------------------------------
--
      IF ((IE.EQ.0).AND.(ID.EQ.0)) THEN
        IEXP = 0
      ELSE
        IEXP = 1
      ENDIF
      IF (IEXP.EQ.1) THEN
c-----------------------------------------------------------------------
--
c     IF AN EXPONENTIAL FORMAT IS BEING USED SEPERATE THE NUMBER INTO
c     A MANTISSA AND AN EXPONENT.
c-----------------------------------------------------------------------
--
        IA = MAX(IE,ID)
        MANT = STRT(1:IA-1)
        EXP  = STRT(IA+1:LA)
        IM = ILEN(MANT)
        IEE = ILEN(EXP)
        EX = 0.0
c-----------------------------------------------------------------------
--
c     DETERMINE THE SIGN OF THE EXPONENT
c-----------------------------------------------------------------------
--
        IF (EXP(1:1).EQ.'+') THEN
          SEXP = 1.D0
            ISE = 2
        ELSEIF (EXP(1:1).EQ.'-') THEN
            SEXP = -1.D0
```

```
            ISE = 2
          ELSE
            SEXP = 1.D0
            ISE = 1
          ENDIF
c-------------------------------------------------------------------------
--
c     DETERMINE THE EXPONENT
c-------------------------------------------------------------------------
--
          DO 400 I = IEE,ISE,-1
            IF (CTOD(EXP(I:I),RX)) THEN
              IF (I.NE.0) EX = RX*10.D0**(IEE-I)+EX
            ELSE
              RETURN
            ENDIF
400       CONTINUE
          EX = EX*SEXP
        ELSE
c-------------------------------------------------------------------------
--
c     OTHERWISE THE EXPONENT = 1 AND THE ENTIRE NO. IS THE MANTISSA
c-------------------------------------------------------------------------
--
          EX = 0.0D0
          MANT = STRT
          IM = ILEN(MANT)
        ENDIF
c-------------------------------------------------------------------------
--
c     DETERMINE THE MANTISSA
c     FIRST DETERMINE THE PORTION BEFORE THE DECIMAL POINT
c-------------------------------------------------------------------------
--
        DO 200 I = IDEC-1,ISM,-1
          IF (CTOD(MANT(I:I),RX)) THEN
            IF (I.NE.0) X = RX*10.D0**(IDEC-I-1)+X
          ELSE
            RETURN
          ENDIF
200     CONTINUE
c-------------------------------------------------------------------------
--
c     NEXT DETERMINE THE PORTION AFTER THE DECIMAL POINT
c-------------------------------------------------------------------------
--
        IF (IDEC.NE.0) THEN
          DO 300 I = IDEC+1,IM
            IF (CTOD(MANT(I:I),RX)) THEN
              IF (I.NE.0) X = RX*10.D0**(-I+IDEC)+X
            ELSE
              RETURN
            ENDIF
300       CONTINUE
        ENDIF
c-------------------------------------------------------------------------
--
c     EVALUATE THE FINAL NUMBER BASED ON THE SIGN AND THE MANTISSA
```

```
c     AND THE EXPONENT
c-------------------------------------------------------------------------
--
      RR = X*SIGN*10.D0**INT(EX)
      STOD = .TRUE.
      RETURN
      END


      LOGICAL*1 FUNCTION CTOD (CH,XX)
c-------------------------------------------------------------------------
--
c     REAL FUNCTION CTOD CONVERTS A CHARACTER TO A NUMBER (REAL*8)
c     IF THE CHARACTER IS NOT WITHIN '0-9' IT RETURNS WITH
c     CTOD = .FALSE.
c     Vinod Shah          7/13/90
c-------------------------------------------------------------------------
--
      CHARACTER*1 CH
      REAL*8 XX
      CTOD = .FALSE.
      IF ((ICHAR(CH).GE.48).AND.(ICHAR(CH).LE.58)) THEN
        I = ICHAR(CH)-48
        XX = DBLE(I)
        CTOD = .TRUE.
      ENDIF
      RETURN
      END


      SUBROUTINE  LOWER (SLR1,SLR2)
c-------------------------------------------------------------------------
--
c     SUBROUTINE LOWERCONVERTS THE STRING SLR2 TO LOWER CASE AND
c     RETURNS IT IN SLR1
c     Vinod Shah   7/13/90
c-------------------------------------------------------------------------
--
      CHARACTER*(*) SLR1,SLR2
      INTEGER LEN2, LEN1, MAXLEN, I
      LEN1 = LEN(SLR1)
      LEN2 = LEN(SLR2)
      MAXLEN = MIN0 (LEN2, LEN1)
      DO 100 I = 1,MAXLEN
        IVAL = ICHAR(SLR2(I:I))
        IF ((IVAL.GE.65).AND.(IVAL.LE.90)) THEN
            SLR1(I:I) = CHAR(IVAL+32)
        ELSE
            SLR1(I:I) = SLR2(I:I)
        ENDIF
100   CONTINUE
      RETURN
      END
```

```
c-------------------------------------------------------------------
-------
        if ((ilen(sinpu).ne.0).and.(sinpu(1:1).ne.'%')) then
c-------------------------------------------------------------------
-------
c    if the line is not a comment line  and the input file flag has
c    been set .....
c-------------------------------------------------------------------
-------
          if ((fkij).and.(sinpu(1:1).ne.'$')) then
            kc = kc + 1
              il1 = ilen(sinpu)
              kij(kc) = 0.0
              if (stod(sinpu(1:il1),val)) then
                kij(kc) = real(val)
              else
                write(*,*)'Error reading kij'
            endif
            elseif ((fkij).and.(sinpu(1:1).eq.'$')) then
            fkij = .false.
            elseif ((finp).and.(sinpu(1:1).ne.'$')) then
              if (ilen(sinpu).ne.0) then
                  inf = inf + 1
                  fname3(inf) = sinp
              endif
            elseif ((finp).and.(sinpu(1:1).eq.'$')) then
              if (inf.eq.0) then
                  inf = -1
              endif
              finp = .false.
            elseif ((finc).and.(sinpu(1:1).ne.'$')) then
              if ((index1(sinpu,'component')).ne.0) then
                finco = .true.
                  il1 = ilen(sinpu)
                  rhoci = 0.0
                  ipo = index1(sinpu,'#')
                  if (ipo.ne.0) then
                    if (stod(sinpu(ipo+1:il1),val)) then
                      id = int(val)
                    else
                        write (*,*) ' error reading component # '
                    endif
                  else
                    write (*,*) ' error reading component statemant'
                  endif
              elseif ((ilen(sinpu).ne.0).and.(id.gt.0)) then
                call crits2 (sinpu, pci, tci, rhoci, omi, dmi)
                pc(id) = pci
                tc(id) = tci
                ow(id) = omi
                dm(id) = dmi
                rhoc(id) = rhoci
              endif
            elseif ((finc).and.(sinpu(1:1).eq.'$')) then
            finc = .false.
            elseif ((index1(sinpu,'$kij')).ne.0) then
              fkij = .true.
            elseif ((index1(sinpu,'$input')).ne.0) then
```

```
              finp = .true.
            elseif ((index1(sinpu,'$critical-constants')).ne.0) then
              finc = .true.
            endif
          elseif (sinpu(1:1).eq.'%') then
          endif
5       continue
6       continue
        return
        end


        subroutine crits2 (strt, pc, tc, rhoc, om, dm)
c-----------------------------------------------------------------------
-------
c     The subroutine analyzes a string containing critical parameters
c     vinod shah    11/16/90
c-----------------------------------------------------------------------
-------
        implicit real*8 (a-h,o-z)
        character strt*(*), sts*80, snum*80, strt1*80
        logical*1 stod
        logical fil, open1
        save iflt
        external ilen, lower, stod
        data fil,open1/.false.,.false./
        data iflt /0/
        if (iflt .eq.0) then
          inquire (unit=55 ,opened=open1)
          if (open1.eqv..true.) then
            fil = .true.
            iflt = 1
          endif
        else
          iflt = 1
        endif
        do 10 j = 1,80
          sts(j:j) = ' '
          snum(j:j) = ' '
          strt1(j:j) = ' '
10      continue
        call lower (strt1,strt)
        il = ilen(strt1)
        if (il.eq.0) then
          return
        else
          sts = strt1(1:il)
        endif
        i1 = index(sts,'pc')
        if (i1.ne.0) then
          ieq = index(sts,'=')
          if (ieq.eq.0) then
              write (*,20) 'pc','pc'
              if (fil) write (55,20) 'pc','pc'
              return
          else
              snum = sts(ieq+1:il)
```

```
  endif
  if (stod(snum,val)) then
      pc = (val)
  else
      write (*,30) 'pc','pc'
      if (fil) write (55,30) 'pc','pc'
      stop
  endif
  return
endif
i1 = index(sts,'tc')
if (i1.ne.0) then
  ieq = index(sts,'=')
  if (ieq.eq.0) then
    write (*,20) 'tc','tc'
      if (fil) write (55,20) 'tc','tc'
      return
  else
      snum = sts(ieq+1:il)
  endif
  if (stod(snum,val)) then
      tc = (val)
  else
      write (*,30) 'tc','tc'
      if (fil) write (55,30) 'tc','tc'
      stop
  endif
  return
endif
i1 = index(sts,'rhoc')
if (i1.ne.0) then
  ieq = index(sts,'=')
  if (ieq.eq.0) then
      write (*,20) 'rhoc','rhoc'
      if (fil) write (55,20) 'rhoc','rhoc'
      return
  else
      snum = sts(ieq+1:il)
  endif
  if (stod(snum,val)) then
      rhoc = (val)
  else
      write (*,30) 'rhoc','rhoc'
      if (fil) write (55,30) 'rhoc','rhoc'
      stop
  endif
  return
endif
i1 = index(sts,'vc')
if (i1.ne.0) then
  ieq = index(sts,'=')
  if (ieq.eq.0) then
    write (*,20) 'vc','vc'
    if (fil) write (55,20) 'vc','vc'
      return
  else
      snum = sts(ieq+1:il)
  endif
```

```
      if (stod(snum,val)) then
          vc = (val)
          if (rhoc.eq.0.0) rhoc = 1.0/vc
      else
          write (*,30) 'vc','vc'
          if (fil) write (55,30) 'vc','vc'
          stop
      endif
      return
    endif
    i1 = index(sts,'omega')
    if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
          write (*,20) 'omega','omega'
          if (fil) write (55,20) 'omega','omega'
          return
      else
        snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
          om = (val)
      else
          write (*,30) 'omega','omega'
      if (fil) write (55,30) 'omega','omega'
          stop
      endif
      return
    endif
    i1 = index(sts,'dpm')
    if (i1.ne.0) then
      ieq = index(sts,'=')
      if (ieq.eq.0) then
          write (*,20) 'dpm','dpm'
          if (fil) write (55,20) 'dpm','dpm'
          return
      else
          snum = sts(ieq+1:il)
      endif
      if (stod(snum,val)) then
          dm = (val)
      else
          write (*,30) 'dpm','dpm'
          if (fil) write (55,30) 'dpm','dpm'
          stop
      endif
      return
    endif
20  format(1x,'*** crits2 error -- error finding "=" following : ',
     $           a,/,1x,10x,a,4x,' not changed ')
30  format(1x,'*** crits2 error -- error reading number following : ',
     $           a,' = ',/,1x,10x,a,4x,' not changed ')
    return
    end


    SUBROUTINE SKIPL1 (NIN, NOUT, NTOUT, ISC)
```

```
c-----------------------------------------------------------------------
-------
c     THE SUBROUTINE READS THE FILE WITH UNIT NO. NIN
c     AND SKIPS ANY LINES BEGINNING WITH A "%" IN THE COLUMNS 1-10
c     THE LINES READ CONTAINING A % SIGN ARE PRINTED ONTO THE
c     SCREEN AND NTOUT UNITS ... NOUT(1), NOUT(2), ,,, NOUT(NTOUT)
c     IT ALSO LOOKS FOR A NUMBER FOLLOWING " SCAN-CODE " AND RETURNS
c     IT IN ISC
c     VINOD SHAH   11/21/90
c-----------------------------------------------------------------------
-------
      DIMENSION NOUT(NTOUT)
      CHARACTER*78 SKIPS, SKIPSU, SNUM*40
      LOGICAL FIL, OPEN1
      LOGICAL*1 STOD
      REAL*8 VAL
      SAVE IFLT
      DATA FIL,OPEN1/.FALSE.,.FALSE./
      DATA IFLT /0/
      IF (IFLT .EQ.0) THEN
        INQUIRE (UNIT=55 ,OPENED=OPEN1)
        IF (OPEN1.EQV..TRUE.) THEN
          FIL = .TRUE.
          IFLT = 1
        ENDIF
      ELSE
        IFLT = 1
      ENDIF
      ISC = 1
      DO 10 I = 1,100
        READ (NIN,'(A)',ERR=90)SKIPS
        CALL LOWER (SKIPSU, SKIPS)
c-----------------------------------------------------------------------
-------
c     IF A "%" SIGN WAS FOUND IN THE FIRST TEN COLUMNS OF THE FILE
c     IT IS A COMMENT LINE
c-----------------------------------------------------------------------
-------
        IF (INDEX(SKIPSU(1:10),'%').NE.0) THEN
            I1 = INDEX(SKIPSU,'SCAN-CODE')
            IL = ILEN(SKIPSU)
            IF (I1.NE.0) THEN
              IEQ = INDEX(SKIPSU,'=')
              IF (IEQ.EQ.0) THEN
                  WRITE (*,20) 'SCAN-CODE','SCAN-CODE'
                IF (FIL) WRITE (55,20) 'SCAN-CODE','SCAN-CODE'
              ELSE
                  SNUM = SKIPSU(IEQ+1:IL)
              ENDIF
              IF (STOD(SNUM,VAL)) THEN
                  ISC = INT(VAL)
              ELSE
                WRITE (*,30) 'SCAN-CODE','SCAN-CODE'
                IF (FIL) WRITE (55,30) 'SCAN-CODE','SCAN-CODE'
              ENDIF
            ENDIF
            WRITE (*,'(1X,''%'',A)') SKIPS(1:IL)
            DO 15 K = 1,NTOUT
```

```
              WRITE (NOUT(K),'(''%'',A)') SKIPS(1:IL)
15          CONTINUE
          ELSE
              GOTO 90
          ENDIF
10      CONTINUE
c-----------------------------------------------------------------------
-------
c     DATA IS PRESENT IN THE LAST LINE THAT WAS READ, HENCE SET THE FILE
c     POINTER TO THE THAT LINE AND RETURN
c-----------------------------------------------------------------------
-------
90      BACKSPACE(NIN)
20      FORMAT(1X,'*** SKIPL1 ERROR -- ERROR FINDING "=" FOLLOWING : ',
     $             A,/,1X,10X,A,4X,' NOT CHANGED ')
30      FORMAT(1X,'*** SKIPL1 ERROR -- ERROR READING NUMBER FOLLOWING : ',
     $             A,' = ',/,1X,10X,A,4X,' NOT CHANGED ')
        RETURN
        END



        subroutine prfmx2 (p, t, v, a, b, x, phic, kij)
c-----------------------------------------------------------------------
-------
c     calculates the fugacity coefficient of the mixture using the peng-
robinson
c     equation of state
c     dimension a(nc), b(nc) , x(nc), phic(nc)
c     currently nc = 3
c-----------------------------------------------------------------------
-------
        implicit real*8 (a-h,o-z)
        data r /8.31434d-3/
        dimension a(3), b(3) , x(3)
        real*8 kij(3), phic(3)
c-----------------------------------------------------------------------
-------
c        write(*,*)'in prfmx x=',x(1),x(2),x(3)
c-----------------------------------------------------------------------
-------
        sqr2 =dsqrt(2.d0)
c-----------------------------------------------------------------------
-------
c     calculate the compressibility z
c-----------------------------------------------------------------------
-------
        prt = p/r/t
        z = prt*v
c-----------------------------------------------------------------------
-------
c     calculate the mixture parameters
c     calculate aij
c-----------------------------------------------------------------------
-------
        a12 = (1.d0-kij(1))*dsqrt(a(1)*a(2))
        a13 = (1.d0-kij(2))*dsqrt(a(1)*a(3))
        a23 = (1.d0-kij(3))*dsqrt(a(2)*a(3))
```

```
      am = x(1)**2*a(1)+x(2)**2*a(2)+x(3)**2*a(3)
     &     +2.d0*x(1)*x(2)*a12+2.d0*x(1)*x(3)*a13
     &     +2.d0*x(2)*x(3)*a23
      bm = x(1)*b(1)+x(2)*b(2)+x(3)*b(3)
      oa = am*prt/r/t
      ob = prt*bm
c-----------------------------------------------------------------------
-------
c    calculate the fugacity coefficient
c-----------------------------------------------------------------------
-------
      zb = dlog(z-ob)
      zb1 = dlog((z+(sqr2+1.d0)*ob)/(z+(1.d0-sqr2)*ob))
      zb2 = oa/2.d0/sqr2/ob
      phicl1 = b(1)/bm*(z-1) - zb -
     &        zb1*(2.d0*(x(1)*a(1)+x(2)*a12+x(3)*a13)/am -b(1)/bm)
     &      *zb2
      phicl2 = b(2)/bm*(z-1) - zb -
     &        zb1*(2.d0*(x(2)*a(2)+x(1)*a12+x(3)*a23)/am -b(2)/bm)
     &      *zb2
      phicl3 = b(3)/bm*(z-1) - zb -
     &        zb1*(2.d0*(x(3)*a(3)+x(1)*a13+x(2)*a23)/am -b(3)/bm)
     &      *zb2
      phic(1) = dexp(phicl1)
      phic(2) = dexp(phicl2)
      phic(3) = dexp(phicl3)
      return
      end



      subroutine anasov(pr, t, roots, a, b)
c-----------------------------------------------------------------------
-------
c    Analytical solver to solve the cubic equation.
c-----------------------------------------------------------------------
-------
      real*8 m, n, rvcub, rucub, norm, theta, root(3), error(3), r
      real*8 coeff(4), pr ,t, b, a, tol
      complex*16 roots(3), w, u, v, ucub, vcub
      real*8 im1,im2,im3,im4,real1,real3,r1,r2
      r = 8.31434e-3
      w = (-0.5,3**0.5/2)
      root(1) = 0
      root(2) = 0
      root(3) = 0
      coeff(1) = 1.0
      coeff(2) = (b - r*t/pr)
      coeff(3) = (a/pr - 3.0*b**2.0 - 2.0*b*r*t/pr)
      coeff(4) = (b**3.0 + r*t*b**2.0/pr - a*b/pr)
      a1 = coeff(2)
      a2 = coeff(3)
      a3 = coeff(4)
      m = coeff(3) - 1.0/3.0*coeff(2)**2.0
      n = 2.0/27.0*coeff(2)**3.0 - 1.0/3.0*coeff(2)*coeff(3) + coeff(4)
      delta = 1.0/4.0*n**2.0+1.0/27.0*m**3.0
      if (delta .ge. 0) then
        rucub = -0.5*n + delta**0.5
```

```
      if (rucub .le. 0) then
        u = -(-rucub)**(1.0/3.0)
      else
        u = rucub**(1.0/3.0)
      endif
      rvcub = -0.5*n - delta**0.5
      if (rvcub .le. 0) then
        v = -(-rvcub)**(1.0/3.0)
      else
        v = rvcub**(1.0/3.0)
      endif
    else
      norm = (1.0/4.0*n**2+abs(delta))**0.5
      theta = acos(-0.5*n/norm)
      real1 = norm*cos(theta)
      im1 = norm*sin(theta)
      im2 = norm*sin(-theta)
      real3 = norm**(1.0/3.0)*cos(theta/3.0)
      im3 = norm**(1.0/3.0)*sin(theta/3.0)
      im4 = norm**(1.0/3.0)*sin(-theta/3.0)
      ucub = dcmplx(real1,im1)
      vcub = dcmplx(real1,im2)
      u    = dcmplx(real3,im3)
      v    = dcmplx(real3,im4)
    endif
    roots(1) = u + v - 1.0/3.0*coeff(2)
    roots(2) = w*u + w**2.0*v - 1.0/3.0*coeff(2)
    roots(3) = w**2.0*u + w*v - 1.0/3.0*coeff(2)
    t1 = roots(1)**3+a1*roots(1)**2+a2*roots(1)+a3
    t2 = roots(2)**3+a1*roots(2)**2+a2*roots(2)+a3
    t3 = roots(3)**3+a1*roots(3)**2+a2*roots(3)+a3
    do 10 i = 1, 3
      if (abs(dimag(roots(i))) .lt. 10d-8) then
        root(i) = dble(roots(i))
        tol = abs(root(i)*0.001)
        error(i) = root(i)**3+a1*root(i)**2+a2*root(i)+a3
        r1 = root(i)*1.1
        do 20 while(abs(error(i)) .ge. tol)
          e1 = r1**3+a1*r1**2+a2*r1+a3
          if ((error(i) .gt. 0.0) .and. (e1 .ge. error(i))) then
            r1 = root(i)*0.9
          elseif ((error(i) .lt. 0.0) .and.
&                 (e1 .le. error(i))) then
            r1 = root(i)*0.9
          elseif ((e1*error(i)) .ge. 0.0) then
            root(i) = r1
            error(i) = e1
            r1 = 1.1*root(i)
          elseif ((e1*error(i)) .lt. 0) then
            r2 = (r1 + root(i) )/2.0
            e2 = r2**3+a1*r2**2+a2*r2+a3
            if ((e2*e1) .ge. 0) then
              r1 = r2
            else
              root(i) = r2
              error(i) = e2
            endif
          endif
```

```
20        continue
          roots(i) = root(i)
        endif
        error(i) = roots(i)**3+a1*roots(i)**2+a2*roots(i)+a3
10    continue
      return
      end



      subroutine srtrp1 (roots, vv, vl, ndeg, im, icc)
      implicit double precision (a-h,o-z)
      complex*16 roots(4)
      common /debug/ idbgs, idbgr
      dimension rreal(4)
      rreal(1) = 0.0
      rreal(2) = 0.0
      rreal(3) = 0.0
      irf = 0
      vv = -1.0
      vl = -1.0
      icc = 0
c-----------------------------------------------------------------------
c    sort out the real and complex roots
c    the real roots are stored in the array rreal
c-----------------------------------------------------------------------
      nreal = 0
      npos  = 0
      do 20 j = 1, ndeg
        r1 = dimag(roots(j))
        if (abs(r1).lt.1.d-15) then
          nreal = nreal + 1
            rreal(nreal) = dble(roots(j))
            if (rreal(nreal).gt.0) npos = npos + 1
        endif
20    continue
c-----------------------------------------------------------------------
c    sort the real roots in decreasing order
c-----------------------------------------------------------------------
      do 25 l = 2,nreal
        do 30 k = nreal,l,-1
          if (rreal(k).gt.rreal(k-1)) then
              temp = rreal(k-1)
              rreal(k-1) = rreal(k)
              rreal(k) = temp
          endif
30      continue
25    continue
      if (im.eq.3) then
        vv = rreal(1)
        vl = rreal(3)
      elseif (im.eq.1) then
        vv = rreal(1)
      elseif (im.eq.2) then
        do lm = nreal,1,-1
            if (rreal(lm).gt.0.0) then
              vl = rreal(lm)
              goto 89
```

```
            endif
          end do
89        continue
        endif
        return
        end


        subroutine bubpr2 (p, t, x, y, eqk, vv, vl, ier, kij)
c-----------------------------------------------------------------------
c    P is the pressure (guess on input)
c    x(i) is the liquid mole fraction
c    y(i) is the vapor mole fraction (guess on input)
c    eqk(i) is the equilibrium K value on return
c    Uses the peng robinson equation of state
c     calculate the mixture parameters for the liquid phase
c-----------------------------------------------------------------------
        implicit real*8 (a-h,o-z)
        real*8 phicl(3), phicv(3)
        dimension a(3), b(3), x(3), y(3), eqk(3)
        real*8 vc(3), tc(3), ow(3), pc(3), kij(3), vl, vv
        complex*16 roots(3)
        logical fail
        common /critp/pc,vc,tc,ow
        data r /8.31434d-3/
        oka1  = 0.37464 + 1.54226*ow(1) - 0.26992*ow(1)**2
        ob1 = 0.0778d0
        oa1 = 0.45724d0
        bc1 = ob1*r*tc(1)/pc(1)
        ac1 = oa1*r**2*tc(1)**2/pc(1)
        oka2  = 0.37464 + 1.54226*ow(2) - 0.26992*ow(2)**2
        ob2 = 0.0778d0
        oa2 = 0.45724d0
        bc2 = ob2*r*tc(2)/pc(2)
        ac2 = oa2*r**2*tc(2)**2/pc(2)
        oka3  = 0.37464 + 1.54226*ow(3) - 0.26992*ow(3)**2
        ob3 = 0.0778d0
        oa3 = 0.45724d0
        bc3 = ob2*r*tc(3)/pc(3)
        ac3 = oa2*r**2*tc(3)**2/pc(3)
        tr1 = t/tc(1)
        alpha1 = (1.d0 + oka1*(1.d0 - dsqrt(tr1)))**2
        a(1) = alpha1*ac1
        b(1) = bc1
        tr2 = t/tc(2)
        alpha2 = (1.d0 + oka2*(1.d0 - dsqrt(tr2)))**2
        a(2) = alpha2*ac2
        b(2) = bc2
        tr3 = t/tc(3)
        alpha3 = (1.d0 + oka3*(1.d0 - dsqrt(tr3)))**2
        a(3) = alpha3*ac3
        b(3) = bc3
        a12 = (1.d0-kij(1))*dsqrt(a(1)*a(2))
        a13 = (1.d0-kij(2))*dsqrt(a(1)*a(3))
        a23 = (1.d0-kij(3))*dsqrt(a(2)*a(3))
        am = x(1)**2*a(1)+x(2)**2*a(2)+x(3)**2*a(3)
       &     +2.d0*x(1)*x(2)*a12+2.d0*x(1)*x(3)*a13
```

```
     &      +2.d0*x(2)*x(3)*a23
      bm = x(1)*b(1)+x(2)*b(2)+x(3)*b(3)
c-----------------------------------------------------------------------
c     Solve the peng-robinson equation of state for the liquid phase
c-----------------------------------------------------------------------
      call anasov(p, t, roots, am, bm)
      if (.not.fail) then
        call srtrp1 (roots, vvo, vl, 3, 2 , icc)
        if (icc.ge.0) then
            call prfmx2 (p, t, vl, a, b, x, phicl, kij)
        else
            write (*,*) 'Error in sorting roots '
            ier = -1
            return
        endif
      else
        write (*,*) ' Error in finding roots'
        ier = -3
        return
      endif
      a12 = (1.d0-kij(1))*dsqrt(a(1)*a(2))
      a13 = (1.d0-kij(2))*dsqrt(a(1)*a(3))
      a23 = (1.d0-kij(3))*dsqrt(a(2)*a(3))
      am = y(1)**2*a(1)+y(2)**2*a(2)+y(3)**2*a(3)
     &      +2.d0*y(1)*y(2)*a12+2.d0*y(1)*y(3)*a13
     &      +2.d0*y(2)*y(3)*a23
      bm = y(1)*b(1)+y(2)*b(2)+y(3)*b(3)
c-----------------------------------------------------------------------
c     Solve the peng-robinson equation of state for the vapor phase
c-----------------------------------------------------------------------
      call anasov(p, t, roots, am, bm)
      if (.not.fail) then
        call srtrp1 (roots, vv, vlo, 3, 1 , icc)
        if (icc.ge.0) then
          call prfmx2 (p, t, vv, a, b, y, phicv, kij)
        else
          write (*,*) 'Error in sorting roots '
            ier = -2
            return
        endif
      else
        write (*,*) ' Error in finding roots'
        ier = -3
        return
      endif
      eqk(1) = phicl(1)/phicv(1)
      eqk(2) = phicl(2)/phicv(2)
      eqk(3) = phicl(3)/phicv(3)
      return
      end
```

```
      Subroutine flash(key,fg,fl,xf,yf,a,x,y,t,p,k,erf,kij)
c-----------------------------------------------------------------------
c     This subroutine is a modification of the subroutine, FLASH, from
c     Prausnitz, J.M., Anderson T., etc. "Computer calculations for
c     multicomponent vapor-liquid and liquid-liquid equilibria",
c     Appendix F.
c-----------------------------------------------------------------------
      real*8 z(3),u(3),w(3),kl,kij(3),vl,vv
      real*8 fg, fl, xf(3), yf(3), a, x(3), y(3), t, p, k(3)
      integer erf, er
      data eps/0.01/
      vf = fg/(fg+fl)
100   erf = 0
      kee = key
      sx = 0.0
      sy = 0.0
      sxf = 0.0
      syf = 0.0
      do 10 I=1,3
        sx = sx+x(i)
        sy = sy+y(i)
        sxf = sxf+xf(i)
        syf = syf+yf(i)
10    continue
      if (abs(sxf-1.0) .gt. 0.01) goto 930
      if (abs(syf-1.0) .gt. 0.01) goto 930
      do 20 i = 1,3
        x(i) = xf(i)
        y(i) = yf(i)
        u(i) = xf(i)
        w(i) = yf(i)
        z(i) = fl*xf(i)+fg*yf(i)
20    continue
      a = vf
150   it = 0
      sl = 1.0
      kee = 2
      kd = 0
      tn = t
      an = a
      goto 280
200   it = it +1
      if (it .eq. 1) goto 230
      if (abs(dfa) .ge. 0.1)  then
        goto 210
      endif
      epf = eps*10.0*abs(dfa)
      if (abs(f) .le. epf) goto 400
      goto 220
210   if (fv .le. eps) goto 400
220   if (it .gt. 20000) goto  900
230   fo = fv
      kd = 0
      goto 380
250   an = a-sl*da
      if (an .lt. 0.0) an = 0.01
      if (an .gt. 1.0) an = 0.99
```

```
      if (kd .eq. 1.0) goto 290
280   call bubpr2 (p, t, x, y, k, vv, vl, er, kij)
290   sx = 0
      sy = 0
      dfa = 0
      do 300 i = 1,3
        k1 = k(i) - 1.0
        u(i) = z(i)/(k1*an+1.0)
        w(i) = k(i)*u(i)
        sx = sx + u(i)
        sy = sy + w(i)
        dfa = dfa - (w(i)-u(i))**2/z(i)
300   continue
      f = sy - sx
      fv = abs(f)
      if (it .eq. 0) goto 200
      if (it .le. 2) goto 370
      if (kd .eq. 2) goto 370
      if (fv .le. fo) goto 370
      kd = 1
      sl = 0.9*sl
      if (sl. lt. 0.2) goto 360
      goto 250
360   an = a
      tn = t
      kd = 2
      goto 280
370   a = an
      t = tn
      sl = 1.0
      goto 200
380   da = f/dfa
      do 390 i =1,3
        x(i) = u(i)/sx
        y(i) = w(i)/sy
390   continue
      goto 250
400   do 500 i = 1,3
        x(i) = u(i)/sx
        y(i) = w(i)/sy
500   continue
      erf = er
      return
900   erf = 2
      goto 940
910   erf = 3
      goto 940
920   erf = 4
      a = 1.0
      goto 950
930   erf = 5
940   a = 0.0
950   continue
      return
      end
```

APPENDIX D

EXPERIMENTAL DATA

The following are photographs and tabulated data from all the figures contained in the body of the dissertation. Further data and information can be found in the Oak Ridge National Laboratory notebook (serial no. A108454-G) and in electronic format, which are both in the possession of the major professor, H. D. Cochran.

Table 3. Tabulation of Figure 25 data				
kV	2	3	4	5
10000 Hz	0.91	1.03	1.35	1.79
1000 Hz	0.93	1.07	1.26	1.56
100 Hz	0.98	1.04	1.1	1.37

Table 4. Tabulation of Figure 26 data		
ms/kv	Hz	4 ml/min
100	10	0.81
10	100	1.15
1	1000	1.39
0.1	10000	1.49
pulse, ms	Hz	7 ml/min
0.1	10000	2.33
0.3	3333	2.31
1	1000	1.97
3	333	1.69
10	100	1.44
30	33	1.19
100	10	0.96
300	3	0.87

Table 5. Tabulation of Figure 27 data			
flow	10 Hz	flow	1000 Hz
1	0.98	1	1.29
2	0.87	2	1.25
3	0.88	3	1.11
4	0.81	4	1.11
5	0.82	5	1.29
6	0.80	6	1.33
7	0.89	7	1.66
8	1.54	8	2.94
9	1.96	9	3.59

Table 6. Tabulation of data in Figures 28-33								
	flow, ml/min	2	4	6	8	10	12	avg.
17.24MPa	65C	1.10	1.11	1.02	1.17	1.21	1.25	1.14
13.79MPa	65C	1.53	1.52	1.35	1.30	1.56	1.66	1.49
10.34MPa	65C	2.22	2.33	2.47	2.57	2.48	2.52	2.43
17.24MPa	47C	1.82	1.60	1.68	1.84	1.83	1.98	1.79
13.79MPa	47C	1.50	1.44	1.43	1.59	1.68	1.94	1.60
10.34MPa	47C	1.34	1.34	1.32	1.59	1.71	1.92	1.54
17.24MPa	35C	2.16	2.03	1.97	2.20	2.30	2.65	2.22
13.79MPa	35C	1.76	1.74	1.60	1.78	2.04	2.35	1.88
10.34MPa	35C	0.89	0.74	0.90	0.90	1.03	1.03	0.91

Table 7. Tabulation of Figure 34 data					
Mpa P	mN/m 35C	Mpa P	mN/m 40C	Mpa P	mN/m 65C
0.1	47.99	0.1	47.53	0.1	43.86
2.07	40.87	2.07	39.47	2.07	39.72
4.14	33.57	4.14	34.24	4.14	34.87
6.89	24.62	6.2	27.77	6.2	29.99
7.93	14.43	8.27	20.45	8.27	25.39
8.27	13.25	8.96	12.88	10.34	22.15
8.62	10.93	9.65	13.96	12.41	18.82
8.96	12.01	10.34	17.03	13.79	17.76
10.34	16.89	12.41	17.19	15.51	17.32
11.72	17.93	13.79	16.96		
13.79	19.28	15.51	17.06		
15.51	18.95				

Table 8. Tabulation of Figure 35 data	
γ, mN,m	<d>, μm
16.89	0.91
22.15	2.43
19.28	1.88
17.76	1.53
17.03	1.54
16.96	1.43

Table 9. Tabulation of data in Figure 36 & 38 for 10.34MPa/35C					
mol/hr	ml/min	pred	mol/hr	ml/min	obs
0.12	50	75.8	0.13	52	77.28
0.60	245	84.50	1.84	748	81.37
1.08	440	86.1	2.31	935	86.16
1.57	635	86.6	3.46	1403	85.32
2.05	830	86.9	3.46	1405	93.98
2.53	1025	87.2	4.93	2000	92.28
3.01	1220	87.3	4.82	1953	89.58
3.49	1415	87.4	4.92	1994	88.54
3.97	1610	87.5	4.91	1990	81.05
4.45	1805	87.5			
4.93	2000	87.6			

Figure 10. Tabulation of data in Figure 36 & 38 for 17.24MPa/35C			
mol/hr	ml/min	obs	pred
1.10	448	89.03	98.60
2.47	1000	98.42	99.30
3.50	1418	96.51	99.50
5.21	2114	99.46	99.70
5.02	2035	95.99	99.70
5.22	2119	98.36	99.70
4.91	1993	102.21	99.70
14.55	5900	114.88	99.90
26.82	10879	128.37	99.90
36.98	15000	54.29	100.00

Table 11. Tabulation of data in Figure 36 & 38 for 10.34MPa/35C

Normal experiment (field on)

L/min	raw ri data		VAPOR			LIQUID			L/min	ALPHA
qco2	vap	liq	H2O	Etoh	ETOH%	H2O	Etoh	ETOH%	qco2	obs
52	6414	3777	0.103	0.897	0.897	0.899	0.101	0.101	52	77.28
748	6415	3751	0.104	0.896	0.896	0.904	0.096	0.096	748	81.37
935	6405	3749	0.099	0.901	0.901	0.905	0.095	0.095	935	86.16
1403	6406	3751	0.100	0.900	0.900	0.904	0.096	0.096	1403	85.32
1405	6391	3745	0.092	0.908	0.908	0.905	0.095	0.095	1405	93.98
2000	6395	3743	0.094	0.906	0.906	0.906	0.094	0.094	2000	92.28
1953	6402	3739	0.098	0.902	0.902	0.907	0.093	0.093	1953	89.58
1994	6405	3737	0.099	0.901	0.901	0.907	0.093	0.093	1994	88.54
1990	6422	3737	0.107	0.893	0.893	0.907	0.093	0.093	1990	81.05

NULL experiment (no field)

L/min	raw ri data		VAPOR			LIQUID			L/min	ALPHA
qco2	vap	liq	H2O	Etoh	ETOH%	H2O	Etoh	ETOH%	qco2	null
327	5952	3334	0.150	0.850	0.850	0.900	0.100	0.100	327	51.14
781	5957	3329	0.156	0.844	0.844	0.902	0.099	0.099	781	49.53
1304	5985	3330	0.164	0.836	0.836	0.901	0.099	0.099	1304	46.54
1798	5939	3320	0.217	0.783	0.783	0.904	0.096	0.096	1798	33.79
1919	5942	3311	0.214	0.786	0.786	0.906	0.094	0.094	1919	35.41

Table 12. Tabulation of data in Figure 37 for 10.34MPa/65C					
mol/hr	ml/min	pred	mol/hr	ml/min	obs
0.99	400	137.1	1.01	411	140.03
1.75	710	139.5	2.02	821	130.88
2.51	1020	140.6	3.35	1358	148.48
3.28	1330	141.2	4.85	1967	147.37
4.04	1640	141.6	7.40	3000	144.37
4.81	1950	141.6	7.60	3083	146.56
5.57	2260	141.8	8.05	3264	140.36
6.34	2570	141.9			
7.10	2880	142			
7.86	3190	142.1			
8.63	3500	142.2			

Table 13. Tabulation of data in Figure 37 for 17.24MPa/65C					
mol/hr	ml/min	obs	mol/hr	ml/min	pred
0.99	402	88.36	0.99	400	112.7
2.29	930	92.20	1.75	710	113.3
3.38	1371	94.20	2.51	1020	113.5
4.82	1953	106.78	3.28	1330	113.6
7.86	3188	113.07	4.04	1640	113.7
8.03	3255	109.44	4.81	1950	113.8
8.21	3330	100.50	5.57	2260	113.8
			6.34	2570	113.9
			7.10	2880	113.9
			7.86	3190	113.9
			8.63	3500	113.9

Table 14. Tabulation of Figure 39 data

10.34MPa/35C Exp. #	vol, L	t, min	CO2 L/min	CO2 gmol/hr	total ext. wt.	yetoh'	goal seek	wh20, g	wetoh, g	ETOH Qetoh, gmol/s **10.34 MPa**	H2O Qh2o, gmol/s **10.34MPa**	MT rate etoh gmol/cm3s
20	40	20.10	1.99	4.91	0.2	0.7195	0.7195	0.0265	0.1735	11.2635	4.3905	3.13E-05
21	44	22.07	1.99	4.92	0.29	0.7369	0.7373	0.0355	0.2545	15.0438	5.3601	4.17E-05
22	51	26.11	1.95	4.82	0.32	0.7401	0.7401	0.0387	0.2813	14.0547	4.9352	3.90E-05
24	50	35.59	1.40	3.46	0.31	0.7518	0.7520	0.0354	0.2746	10.0619	3.3174	2.79E-05
25	53	37.78	1.40	3.46	0.31	0.7359	0.7359	0.0382	0.2718	9.3848	3.3677	2.60E-05
26	54.5	58.27	0.94	2.31	0.29	0.7369	0.7373	0.0355	0.2545	5.6975	2.0300	1.58E-05
27	50	66.87	0.75	1.84	0.48	0.7266	0.7267	0.0616	0.4184	8.1620	3.0698	2.26E-05
28	11.2	214.07	0.05	0.13	0.21	0.7276	0.7278	0.0268	0.1832	1.1162	0.4175	3.10E-06
NULL 10.34MPa/35C										ETOH **null**	H2O **null**	
52	57	29.71	1.92	4.73	0.2	0.8500	0.8501	0.0129	0.1871	8.2148	1.4480	2.28E-05
53	55	30.59	1.80	4.43	0.22	0.8440	0.8440	0.0148	0.2052	8.7493	1.6169	2.43E-05
54	50	58.79	0.85	2.10	0.26	0.8360	0.8361	0.0185	0.2415	5.3575	1.0504	1.49E-05
55	50	64.00	0.78	1.93	0.36	0.7826	0.7835	0.0351	0.3249	6.6210	1.8296	1.84E-05
56	25.5	76.52	0.33	0.82	0.24	0.7860	0.7861	0.0231	0.2169	3.6975	1.0060	1.03E-05

Table 15. Tabulation of data in Figures 40 and 41

10.34MPa/35C					total					ETOH Qetoh, gmol/s	H2O Qh2o, gmol/s	MT rate ETOH
Exp. #	co2 vol	t, min	L/min	gmol/hr	extract wt.	yetoh'	goal seek	wh20, g	wetoh, g	10.34 MPa	10.34 MPa	gmol/cm3s
20	40	20.10	1.990	4.91	0.2	0.719	0.720	0.026	0.174	11.26	4.39	3.13E-05
21	44	22.07	1.994	4.92	0.29	0.737	0.737	0.035	0.255	15.04	5.36	4.17E-05
22	51	26.11	1.953	4.82	0.32	0.740	0.740	0.039	0.281	14.05	4.94	3.90E-05
24	50	35.59	1.405	3.46	0.31	0.752	0.752	0.035	0.275	10.06	3.32	2.79E-05
25	53	37.78	1.403	3.46	0.31	0.736	0.736	0.038	0.272	9.38	3.37	2.60E-05
26	54.5	58.27	0.935	2.31	0.29	0.737	0.737	0.035	0.255	5.70	2.03	1.58E-05
27	50	66.87	0.748	1.84	0.48	0.727	0.727	0.062	0.418	8.16	3.07	2.26E-05
28	11.2	214.07	0.052	0.13	0.21	0.728	0.728	0.027	0.183	1.12	0.42	3.10E-06
17.24MPa/35C										Qetoh, gmol/s	Qh2o, gmol/s	ETOH
Exp. #	co2 vol	t, min	L/min	gmol/hr	extract wt.	yetoh'	goal seek	wh20, g	wetoh, g	17.24 MPa	17.24 MPa	gmol/cm3s
30	51	25.06	2.035	5.02	0.5	0.733	0.733	0.062	0.438	22.77	8.30	6.32E-05
32	40	28.21	1.418	3.50	0.49	0.741	0.741	0.059	0.431	19.93	6.96	5.53E-05
34	41	91.57	0.448	1.10	0.53	0.732	0.732	0.066	0.464	6.60	2.42	1.83E-05
17.24MPa/65C										Qetoh, gmol/s	Qh2o, gmol/s	ETOH
Exp. #	co2 vol	t, min	L/min	gmol/hr	extract wt.	yetoh'	goal seek	wh20, g	wetoh, g	17.24 MPa	17.24 MPa	gmol/cm3s
40	43	12.91	3.330	8.21	0.52	0.747	0.747	0.061	0.459	46.38	15.70	1.29E-04
41	41	12.60	3.255	8.03	0.53	0.746	0.746	0.062	0.468	48.44	16.48	1.34E-04
45	39	12.23	3.188	7.86	0.49	0.757	0.758	0.054	0.436	46.45	14.81	1.29E-04
42	32	23.34	1.371	3.38	0.51	0.736	0.736	0.063	0.447	24.99	8.96	6.94E-05
43	28	30.10	0.930	2.29	0.58	0.739	0.739	0.070	0.510	22.09	7.79	6.13E-05
44	27.5	68.45	0.402	0.99	0.44	0.732	0.732	0.055	0.385	7.33	2.68	2.04E-05
10.34MPa/65C										Qetoh, gmol/s	Qh2o, gmol/s	ETOH
Exp. #	co2 vol	t, min	L/min	gmol/hr	extract wt.	yetoh'	goal seek	wh20, g	wetoh, g	10.34 MPa	10.34 MPa	gmol/cm3s
45	40	12.26	3.264	8.05	0.39	0.808	0.808	0.033	0.357	37.97	9.03	1.05E-04
46	43	13.95	3.083	7.60	0.41	0.812	0.812	0.034	0.376	35.17	8.12	9.76E-05
48	35	25.77	1.358	3.35	0.44	0.816	0.816	0.036	0.404	20.46	4.62	5.68E-05
50	22	53.57	0.411	1.01	0.3	0.808	0.808	0.026	0.274	6.68	1.59	1.85E-05
51	32.5	16.52	1.967	4.85	0.42	0.815	0.815	0.034	0.386	30.44	6.93	8.45E-05

VITA

Kevin D. Heath was born in Knoxville, Tennessee on March 21, 1971. He attended public schools in East Tennessee, where he graduated the top 10% of his class in June, 1989. He entered the University of Tennessee the fall semester of the same year, where in August, 1994 after completing 4 semesters of Co-op work experience at General Electric Aircraft Engines in Cincinnati, Ohio, he received the Bachelor of Science in Chemical Engineering with honors of Magna Cum Laude. He then entered the Master's program in chemical Engineering at the University of Tennessee; and through a special program called "A Year in Japan", he was able to complete his research work at Kyoto University in Kyoto, Japan for a period of 1.5 years, which was fully funded through the Japanese government. He officially received his Master's degree in May of 1997. He then accepted a research assistantship with the University of Tennessee Chemical Engineering Department with funding through Oak Ridge National Laboratory's Chemical Technology Division for the pursuit of a Doctor of Philosophy degree. His concentration was in improved separation techniques in SCO_2 using electrically enhanced dispersions. He is a member of both the American Chemical Society (ACS) and the American Institute of Chemical Engineers (AIChE).