CACHE NEWS

NEWS ABOUT COMPUTERS IN CHEMICAL ENGINEERING EDUCATION.

No. 28

Spring 1989

The CACHE CORPORATION

WHAT IS CACHE?

CACHE is a not-for-profit organization whose purpose is to promote cooperation among universities, industry and government in the development and distribution of computer-related and/or technology-based educational aids for the chemical engineering profession

CREATION OF THE CACHE CORPORATION

During the 1960s the rapid growth of computer technology challenged educators to develop new methods of meshing the computer with the teaching of chemical engineering. In spite of many significant contributions to program development, the transferability of computer codes, even those written in FORTRAN, was minimal. ıecause of the disorganized state of university-developed codes for chemical engineering, fourteen chemical engineering educators met in 1969 to form the CACHE (Computer Aids for Chemical Engineering) Committee. The CACHE Committcc was initially sponsored by the Commission on Education of the National Academy of Engineering and funded by the National Science Foundation. In 1975, after several successful projects had been completed, CACHE was incorporated as a not-for-profit corporation in Massachusetts to serve as the administrative umbrella for the consortium activities.

CACHE ACTIVITIES

All CACHE activities are staffed by volunteers including both educators and industrial members and coordinated by the Board of Trustees through various Task Forces. CACHE actively solicits the participation of interested individuals in the work of its ongoing projects. Information on CACHE activities is regularly disseminated through CACHE News, published twice yearly. Individual inquiries should be addressed to:

CACHE Corporation
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933
CACHE@UTA3081

CACHE NEWS

The CACHE News is published twice a year and reports news of CACHE activities and other noteworthy developments of interest to chemical engineering educators. Persons who wish to be placed on the mailing list should notify CACHE at the aforementioned address. Contributions from CACHE represcntatives are welcome. This issue was editied by D. M. Himmelblau with contributions from a number of CACHE members and representatives.

CACHE NEWS

No. 28 Spring 1989

CACHE Trustees

President:

H. Scott Fogler, University of Michigan

Vice President

Jeffrey J. Siirola, Eastman Kodak

Secretary

Michael B. Cutlip, University of Connecticut
(BITNET: MCUTLIP@UCONNVM

Executive Officer

David M. Himmelblau, University of Texas at Austin

Academic Trustees

Yaman Arkun, Georgia Institute of Technology
L. T. Biegler, Carnegie-Mellon University
Brice Carnahan, University of Michigan,
(BITNET:BRICE_CARNAHAN@UB.CC.UMICH.EDU
James F. Davis, Ohio State University
Michael F.Doherty, University of Massachusetts
James M. Douglas, University of Massachusetts
Thomas F. Edgar, University of Texas at Austin
Bruce A. Finlayson, University of Washington
Ignacio Grossmann, Carnegie-Mellon University
Richard S. H. Mah, Northwestern University
(BITNET: DICK_MAH@NUACC.ACNS.NWU.EDU
Manfred Morari, California Institute of Technology
(BITNET: MM%IMC@CITROMEO
Gintaras V. Reklaitis, Purdue University
(BITNET: GVR@PURCHE
Peter R. Rony, Virginia Polytechnic Institute and State University (BITNET: RONY@VTVM1)
J. D. Seader, University of Utah
(Bitnet: SEADER@UTAHCCA)
Warren D. Seider, University of Pennsylvania
John H. Seinfeld, California Institute of Technology
George Stephanopoulos, Massachusetts Institute of Technology

Industrial Trustees

John C. Hale, E. I. DuPont de Nemours & Co.
J. J. Haydel, Shell
Norman E. Rawson, IBM,
(BITNET: RAWSON@MILVM1, Gateways must be open at both ends of the BITNET link.)
Edward M. Rosen, Monsanto Company
H. Dennis Spriggs, Linnhoff March
Joseph D. Wright, Xerox
(BITNET: WRIGHT@MCMASTER
or WRIGHT.XRCC-NS@XEROX.COM

Advisory Committee

Lawrence B. Evans, Massachusetts Institute of Technology
James R. Fair, University of Texas at Austin
Thomas H. Laffrerre, Monsanto Company
John J. McKetta, University of Texas at Austin

CACHE Task Forces and Committees

STANDING COMMITTEES

Publications

Prof. Brice Carnahan
Dept. of Chemical Engineering
University of Michigan
Ann Arbor, Michigan 48109
(313) 764-3366

Newsletter

Prof. David M. Himmelblau
Dept. of Chemical Engineering
University of Texas
Austin, Texas 78712
(512) 471-7445

Development

Dr. H. Dennis Spriggs
Linnhoff March
P. O. Box 2306
Leesburg, Virginia 22075-7617
(703) 777-1118

Conferences

Prof. Richard S. H. Mah
Dept. of Chemical Engineering
Northwestern University
Evanston, Illinois 60201
(312) 491-5357

AD HOC COMMITTEES

Laboratory Applications of Microcomputers

Prof. Yaman Arkun
School of Chemical Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0100

Artificial Intelligence

Prof. George Stephanopoulos
Dept. of Chemical Engineering
Massachusetts Institute of Technology
66-562
Cambridge, Massachusetts 02139
(617) 253-3904

TASK FORCES

Process Engineering

Dr. Jeffrey Siirola
Eastman Kodak Co.
P. O. Box 1972
Kingsport, Tennessee 37662
(615) 229-3069

Case Studies

Prof. Manfred Morari
Dept. of Chemical Engineering
206-41
California Institute of Technology
Pasadena, California 91125
(818) 356-4186

Curriculum

Prof. Warren Seider
Dept. of Chemical Engineering
University of Pennsylvania
220 S. 33rd Street/D3
Philadelphia, Pennsylvania 19104
(215) 898-7953

Electronic Mail

Prof. Peter Rony
Dept. of Chemical Engineering
Virginia Polytechnic Institute & State University
Blacksburg, Virginia 24061
(703) 961-7658

CACHE Process Engineering Task Force

Edward M. Rosen
Monsanto Corporation

Formerly the Large Scale Systems Task Force, the CACHE Process Engineering Task Force is the oldest task force in CACHE. This article is intended to briefly review the history of the Task Force and indicate its purpose, its accomplishments to date, and its future potential products and directions.

Members

Membership in the task force is generally open to all who are interested/partake in its activities. At any particular time the membership includes those actively involved in a task force project.

Task Force Mission and Purpose

The mission of the Process Engineering Task Force is the acquisition, evaluation and distribution of process engineering software for educational use. Such software includes programs for process synthesis, heat and material balancing, equipment modelling, rating, or design, dynamic simulation, control, optimization and the like. Task force activities include negotiations with third party software developers to obtain programs, preparation of illustrative test cases, software modifications for selected hardware/operating system combinations, preparation of user documentation and supporting instructional materials, program promotion and distribution, development of software enhancements, and similar activities.

History

The Large Scale Systems Task Force was first first organized at the CACHE meeting at Buck Hill Falls, Pennsylvania (Sept 2-3, 1971). The initial membership was J. D. Seader (Univ. of Utah Chairman), L. B. Evans (MIT), R. R. Hughes (Univ. of Wisconsin), W. D. Seider (Univ. of Pennsylvania) , Paul T. Shannon (Dartmouth), and Imre Zwiebel (Worcester Polytechnic Institute).

The task force's initial project was to send a letter

Current Task Force Members

Name	Affiliation	Current Activity
Trustees		
Lorenz T. Biegler	Carnegie Mellon	FLOWTRAN
Gary E. Blau*	Dow	SimuSolv
Michael F. Doherty	University of Massachusetts	PIP
James M. Douglas	University of Massachusetts	PIP
Ignacio E. Grossmann	Carnegie Mellon	MAGNETS, RESHEX, ADVENT
G. V. Reklaitis	Purdue	SPEEDUP, THEN, GPSS/PC
Edward M. Rosen	Monsanto	Chairman, FLOWTRAN, GPSS/PC
Jeffrey J. Siirola	Eastman Kodak	Past Chairman, RESHEX,MAGNETS, PIP, THEN
J. D. Seader	University of Utah	FLOWTRAN, SPEEDUP
Warren D. Seider	University of Pennsylvania	FLOWTRAN
Other Affiliates		
F. Carl Knopf	Louisiana State	THEN
Angelo Lucia	Clarkson	SPEEDUP
A. C. Pauls	Monsanto	FLOWTRAN
Jude T. Sommerfeld	Georgia Tech	GPSS/PC
H. Dennis Spriggs	Linnhoff March	Target II

*Newly Designated Member

to all major companies known to have a computer-aided process design program to inquire as to the program's accessibility to universities. Their initial publication was "CACHE Guidelines for Large Scale Computer Programs" (Feb 1973).

On December 10, 1973 Monsanto announced approval of the use of FLOWTRAN for educational use via a national computer network. Many Monsanto personnel were instrumental in assisting CACHE in this venture. Final arrangements by Monsanto were due to the efforts of M. C. Throdahl, F. E. Reese, J. R. Fair, and S. I. Proctor.

In 1978 the task force cosponsored a workshop with EDUCOM in which they reported the results of using FLOWTRAN on a service bureau. (See article 7 under Articles About CACHE Task Force Activities.)

In 1987 the name of the task force was changed from Large Scale Systems Task Force to Process Engineering Task Force in recognition of the broad scope of the task force's activities.

Past Chairmen

Large-Scale Systems	J. D. Seader	1971- 1982
Large-Scale Systems	R. R. Hughes	1983- 1985
Large Scale Systems/ Process Engineering	J. J. Siirola	1985- 1988
Process Engineering	E. M. Rosen	1989-

Currently Available Computer Products

1. FLOWTRAN

Applicability

FLOWTRAN is a steady state simulation system applicable to a wide range of chemical processes. It is based on sequential modular technology and requires the user to specify the feed streams and topology of the system. It has been used in courses in thermodynamics and unit operations, as well as the capstone senior design course.

Background

In 1974, the Monsanto Company made available their FLOWTRAN system. This project has continued until the current time. Some of the key dates in this project are listed below:

1974: FLOWTRAN made available on United Computing Systems service bureau

1982: FLOWTRAN Load Modules made available for Universities to run on their own computers

CACHE managed the effort to convert the FLOWTRAN system to a wide variety of target machines. Representatives from a number of universities participated in this effort.

1987: New Optimization Enhancement added (L. T. Biegler)

Task Force Members

Lorenz T. Biegler is Associate Professor of Chemical Engineering at Carnegie Mellon University. He obtained his Ph.D. degree from the University of Wisconsin in 1981. His research interests include flowsheet optimization, reactor network synthesis, and optimization strategies for dynamic systems. His main involvement with CACHE has been through the FLOWTRAN project and the optimization capability.

Michael F. Doherty is currently Professor and Department Head of the Chemical Engineering Department at the University of Massachusetts. He joined the staff of the University of Massachusetts in 1977 after previous teaching and industrial experience. He received his B.S. in chemical engineering from Imperial College, the University of London in 1973 and his Ph.D. from Trinity College, University of Cambridge in 1977. Dr. Doherty's research interests are in design and synthesis of nonideal separation systems, thermodynamics, and nonlinear process dynamics and control. He is also a member of the Center for Process Design and Control at the University of Massachusetts.

James M. Douglas is currently Professor of Chemical Engineering at the University of Massachusetts. He joined the department in 1968 after previous teaching at the University of Rochester and industrial experience at ARCO. He also served as department head of chemical engineering at the University of Massachusetts from 1979 to 1982. He received his B.S. in chemical engineering from Johns Hopkins University in 1954, his Ph.D. from the University of Delaware in 1960, and did postdoctoral studies at the Imperial College of Science and Technology in London in 1964. Dr. Douglas' research interests are in process design and control and reaction engineering. He is also the original and current director of the Center for Process Design and Control at the University of Massachusetts.

Since 1982 some 156 schools throughout the world have received FLOWTRAN for use in the classroom. The following indicates the growth in FLOWTRAN usage.

Year	Service Bureau Users	Total Systems Distributed
1974-1975	25	
1975-1976	34	
1980-1981	57	
1983		26
1984		62
1985		30
1986		27
1987		7
1988		11
		163

As of the end of 1988 some 29 schools signed an agreement with Stanford University for the use of QPSOL in the Optimization Enhancement.

Availability

A FLOWTRAN system tape or the Optimization Enhancement may be obtained from:

Professor J. D. Seader
CACHE
3290 MEB
University of Utah
Salt Lake City, Utah 84112

It is necessary to sign a three way agreement with Monsanto and CACHE for FLOWTRAN. In addition if the optimization enhancement is used an agreement with Stanford University is required.

A tape of FLOWTRAN is available on the following systems:

DEC VAX computers running with VMS operating systems

DEC VAX computers running with Ultrix-32 (UNIX) operating system

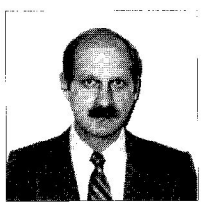
DEC 20XX mainframe computer running with the FORTRAN-20 compiler (9-track, 1600 BPI tape)

UNIVAC 1100 series computers running under the EXEC 1100 operating system with the FORTRAN compiler (9 track, 1600 BPI tape)

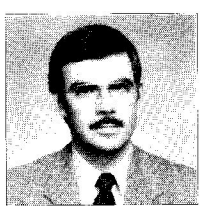
Amdahl computers running under the MTS (Michigan Terminal System) operating system with a FORTRAN Level G or H compiler (9 track, 6250 BPI tape).

IBM and IBM-plug-compatible mainframe computers such as the 370, 30XX and 43XX with the following operating system and FORTRAN compiler combinations:

Version	Operating System	FORTRAN Compiler
a	VM/CMS	VS
b	OS1/MVS	IV-H ext.
c	OS2/VS2 MVS	VS
d	CMS	IV-G1



Ignacio E. Grossmann is currently Professor of Chemical Engineering at Carnegie Mellon University. He received his B.S. degree in chemical engineering from Universidad Iberoamericana, Mexico City, in 1974, and his M.S. and Ph.D. from Imperial College, London, in 1975 and 1977. Professor Grossmann's research interests are in optimization models and methods for process synthesis, process flexibility and planning and scheduling of process operations. Professor Grossmann is also a member of the CACHE Design Case Studies Task Force, where he was a coauthor of the first two volumes.



G. V. Rex Reklaitis is Professor and Head of Chemical Engineering at Purdue University. He received his B.S. in chemical engineering from the Ilinois Institute of Technology in 1965 and Ph.D. from Stanford University in 1969. He joined the Purdue faculty in 1970, after a year as NSF Postdoctoral Fellow in Zurich, Switzerland. He was Senior Fulbright Lecturer in Lithuania, S.S.R., in 1980, served as Assistant Dean of Engineering for Research in 1985-1988, was President of CACHE 1986-88, and since 1986 has been Co-Editor-in-Chief of *Computers & Chemical Engineering*. Dr. Reklaitis' general research interests are in computer aided process design and operations, with current focus on design, scheduling, and simulation methodology applied to batch chemical processing.

Edward M. Rosen is a Senior Fellow in the Engineering Technology Department of the Monsanto-Chemical Company in St. Louis, Missouri. He received his B.S. and M.S. degrees in chemical engineering from Illinois Institute of Technology and his Ph.D. from the University of Illinois. He has been with the Monsanto Co. since 1959, except for a one-year academic leave of absence at Stanford University. He is coauthor (with E. J. Henley) of the book *Material and Energy Balance Computations* (John Wiley, 1969). He is a registered professional engineer in Missouri and has been a CACHE Industrial Trustee since 1977, its secretary from 1984-86 and is currently the chairman of its Process Engineering Task Force.

IBM PX-XT 370 PC operating in conjunction with an IBM mainframe

CDC Cyber mainframe computers with the NOS operating system and FORTRAN V compiler

Apollo Domain workstations running AEGIS operating system (program on floppy disk)

Data General MV superminicomputers running the AOS/VS operating system

Honeywell computers with CP6 operating system

Cray XMP supercomputers

The Optimization Enhancement to FLOWTRAN was developed by Prof L. T. Biegler of Carnegie Mellon University. It is based on Sequential Quadratic Programming and uses the QPSOL Quadratic Programming subroutine from Stanford University. The Enhancement is available on a 5 1/4 inch floppy disk.

Cost

The cost of FLOWTRAN is $250. For departments supporting CACHE, the cost is $175.

New users of FLOWTRAN automatically get the Optimization Enhancement with the tape. Other users may obtain the Optimization Enhancement for $75 (non-CACHE-supporting schools) or $50 for CACHE supporting institutions.

2. TARGET II

Applicability

Target II is a program that can be used to illustrate the basics of heat integration Pinch Technology. The program will construct a composite curve and calculate the minimum energy targets.

The program can be used in synthesis course work or in the capstone design course. It runs on an IBM PC or compatible.

Background

Target II was donated to CACHE by Dr. Dennis Spriggs, President of Linnhoff March, Inc.

Availability

The program is available on 5 1/4 inch floppy disk (360 K or 1.2 Meg) from:

CACHE Corporation
Attn: Janet Sandy
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933

Cost

The cost is $5. In addition, an agreement with CACHE and Linnhoff-March is required.

3. PIP (Academic Version)

Applicability

PIP (Process Invention Procedure) is a generalized

J. D. (Bob) Seader is Professor of Chemical Engineering at the University of Utah. He received B.S. and M.S. degrees from the University of California at Berkeley, and a Ph.D. degree from the University of Wisconsin at Madison. Following a 13-year inudstrial career (first with Chevron in process design and engineering, and then with Rocketdyne Division of North American Aviation where he supervised research on all engines that took man to the moon), he began an academic career. His current research interests include processing of tar sands to produce synthetic crude oil, restricted diffusion in porous catalysts, and finding all roots to systems of nonlinear algebraic and transcendental equations in chemical engineering.

Warren D. Seider is Professor of Chemical Engineering at the University of Pennsylvania. He served as the first Chairman of CACHE and has been a member of the Process Engineering (Large Scale Systems) Task Force since its inception in 1971. Warren is a coauthor of *Flowtran Simulation--An Introduction*. He received his B.S. degree from Polytechnic Institute of Brooklyn and his Ph.D. from the University of Michigan. He was elected a director of the AIChE in 1983.

Jeffrey J. Siirola is currently a Senior Research Associate in the Eastman Chemicals Division of Eastman Kodak Company. He received his B.S. degree in chemical engineering in 1967 from the University of Utah and his Ph.D. in 1970 from the University of Wisconsin. He is currently Programming Chairman of the Computing and Systems Technology Division of AIChE. His research interests include chemical process synthesis, technology assessment, applications of artificial intelligence and symbolic programming, and chemical engineering education.

synthesis procedure based on shortcut design methods. The program tightly integrates design calculations with generalized economics and enables the user to rapidly scan alternatives. The program is useful in the capstone course or in a synthesis course.

The program is designed to run on an IBM PC or compatible. The academic version is limited to working with the examples supplied.

Background

PIP has been made available by Prof. James M. Douglas of the University of Massachusetts. It reflects Prof. Douglas's synthesis/design philosophy as discussed in his recent textbook *Conceptual Design of Chemical Processes*, McGraw-Hill (1988).

Availability

The program is available on six 5 1/4 inch floppy disks (360 K) from:

CACHE Corporation
Attn: Janet Sandy
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933

Documentation (in the form of a thesis) and an installation procedure is distributed with the floppies.

Cost

The cost is $75 for non-supporting institutions. For CACHE-supporting institutions the cost is $50. The cost includes the thesis "PIP—Process Invention Procedure, A Prototype Expert System for Synthesizing Chemical Process Flowsheets," by R. L. Kirkwood (May 1987).

The thesis alone is $20. The Floppies alone are $10.

4. THEN—Version 1

Applicability

THEN (Teaching Heat Exchange Networking) is a PC-based program which carries on where TARGET II leaves off. The program will generate heat exchanger matches attempting to minimize the number of units.

Briefly, the program first locates the pinch region; above the pinch only external heating is allowed and below the pinch only external cooling is allowed. Heat exchangers are first placed at the pinch because this is the most constrained region. Away from the pinch, a variety of heuristic rules are used for matching exchangers.

Background

Prof. Carl Knopf of Louisiana State University has made this program available to CACHE.

Availability

The program is available on one (1.2 Meg) 5 1/4 in disk from:

CACHE Corporation
Attn: Janet Sandy
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933

Cost

The Cost is $5. Initial distribution is free to all supporting departments.

5. GPSS/PC (Student Version)

Applicability

GPSS is a useful tool in batch processing simulation. It can be used in the capstone course or special courses devoted to batch processing and/or simulation.

Background

This is a student version of GPSS/PC limited to fifty blocks.

Availability

The program is available on one (1.2 Meg) 5 1/4 inch disk from:

CACHE Corporation
Attn: Janet Sandy
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933

Cost

The program will be distributed free to all CACHE-supporting departments together with a copy of *Exercises in Chemical Engineering Using GPSS*. After the initial distribution, the package will be available, as noted below, for $25.

Currently Available Books and Manuals

1. *FLOWTRAN Simulation —An Introduction*

by J. D. Seader, University of Utah
W. D. Seider, University of Pennsylvania
A. C. Pauls, Monsanto Co.

1st Edition, 1974
2nd Edition, 1977
3rd Edition, 1987 ($16.95/copy)
(approximately 300 pages)

Available from:

Ulrich's Bookstore
Attn: Heather Senior
549 E. University Avenue
Ann Arbor, Michigan 48109

2. *Exercises in Chemical Engineering Using GPSS*

by D. J. Schultheisz and J. T. Sommerfeld

December 1988
(approximately 250 pages)

Available from:

CACHE Corpration
Attn: Janet Sandy
P. O. Box 7939
Austin, Texas 78713-7939
(512) 471-4933

The exercises come with the student version of GPSS/PC from Minuteman software of Stow, Massachusetts on a 5 1/4 inch floppy disk. Problems from the exercise book are included on the disk.

The cost is $25.

Current Projects

A number of projects are currently being pursued by members of the Process Engineering Task Force. These projects involve negotiating with the vendor, setting up a distribution mechanism, and obtaining documentation.

1. GPSS/PC

GPSS/PC is a an implementation of the popular discrete simulation language GPSS (General Purpose Simulation System). Version 2 introduces graphics and animation into a highly interactive simulation environment.

GPSS/PC is a product of Minuteman Software, P.O. Box 171, Stow Massachusetts (508-897-5662, 800-223-1430). It requires an IBM compatible personal computer, 640K RAM memory, one 5 1/4 inch or 3 1/2 inch diskette drive, and a graphics monitor and adapter compatible with IBM EGA with 256K video buffer.

The use of GPSS in the chemical engineering curriculum has been discussed by Daniel J. Schultheisz and Jude T. Sommerfeld in "Discrete-Event Simulation in Chemical Engineering," *Chemical Engineering Education*, Spring 1988.

GPSS/PC was the system used in the CACHE publication "Exercises in Chemical Engineering Using GPSS" by Schultheisz and Sommerfeld.

The current price to educational institutions for the *full version* of GPSS/PC is $595.

The task force has recently completed negotiations with Minuteman Software for a student version. That version and "Exercises in Chemical Engineering Using GPSS" initially will be distributed free to all universities which support CACHE.

2. RESHEX

The program RESHEX (RESilient Heat EXchanger network synthesis) was developed at the University of Wisconsin and the California Institute of Technology (Manfred Morari) with three objectives in mind:

1. To serve as a tool for continuing research in the area of heat exchanger network synthesis.

2. To serve as an educational aid for undergraduate and graduate students or anybody else learning new synthesis techniques.

3. To reduce the drudgery of hand calculations and graphical procedures.for anybody applying the techniques on a routine basis

The synthesis method for streams with fixed properties is based on the problem table of Linnhoff and Flower. The method for handling constraints and variations in the flowrates is described in *CACHE NEWS*, April 1983, p. 14.

A number of improvements have been made to the program. It is currently being evaluated for distribution.

3. MAGNETS

MAGNETS is an iteractive program for the synthesis of heat exchanger networks. It was developed at Carnegie Mellon University under the direction of Ignacio Grossmann. The program consists of three major phases: minimum utility calculation, determina ton of matches for fewest units, and derivation of network with minimum investment cost. Each of these phases involves the solution of an optimization problem: LP transshipment, MILP transshipment, and NLP optimization of super structure, respectively. MAGNETS generates the formulations and interfaces with LINDO for the first two, and with MINOS for the latter.

Capabilities of MAGNETS include handling of multiple utilities, options for selection of pinch points for network partitioning, specification of various types of constraints for matches and handling of stream splitting.

It is currently being evaluated for distribution.

4. SPEEDUP

SPEEDUP is an equation-oriented flowsheeting

package with the ability to simultaneously solve sets of equations and procedures. Its ability to handle mixed sets of linear and non-linear algebraic and differential equations allows it to tackle design and rating problems as easily as it handles simulation. Its facilities for evolutionary design allow the engineer to build up his flowsheet in the steady-state and then exploit the capabilities for dynamic simulation and optimization.

SPEEDUP was developed at Imperial College, London and is currently distributed by ProsysTech Inc., 30 Vreeland Road, Florham Park, New Jersy 07932-1986 (201-377-4855).

CACHE is negotiating with ProsysTech for an educational version of SPEEDUP.

5. SIMUSOLV

The SimuSolv computer program is an integrated, multifunctional software package designed to help engineers develop and use mathematical models of dynamic chemical and physical systems. The heart of the program is its ability to solve sets of differential equations through numerical integration techniques and analyze nonlinear experimental data in a statistically sound fashion.

SimuSolv was developed by DOW based on ACSL (Advanced Continuous Simulation Language) and is marketed by Mitchell and Gauthier Associates, 73 Junction Square Dr., Concord, MA , 01742-3090 (508-369-5115).

CACHE plans to approach Dow and MGA for an educational version of SimuSolv.

Current Dow fees for educational institutions is $2700 plus an anual fee of $1000.

6. ADVENT

ADVENT is a comprehensive process synthesis system developed by Union Carbide Corporation. It applies and builds on Pinch Technology and incorporates chemical engineering and other technologies in a state-of-the-art engineering work station.

Included in the system are energy targets (minimum practical energy requirements); capital targets (minimum capital requirements); optimum energy/capital tradeoffs (optimum delta T); process modification to reduce energy and/or capital, and improve capacity, operability, and raw material efficiency; optimum stream matches in heat exchangers; and integration of process with furnaces, compressors, and heat utilities.

ADVENT will be marketed by AspenTech, Cambridge, Massachusetts.

CACHE has submitted a proposal to Union Carbide to obtain ADVENT for educational use.

7. THEN—VERSION 2

Often the Grand Composite Curve (GCC) for a given process has an unusual shape—for example, an evaporator train will produce severe "kinks" in the GCC. These kinks, as well as other factors (see E. Rev and Z. Fonyo, *Computers and Chemical Engineering*, Vol. 10, p. 601 (1986)), can produce hidden pinches during the synthesis algorithm. THEN—version 2 allows solution of these problems, thereby expanding its utility in the synthesis of many heat exchanger networks.

The task force is working with Prof. F. Carl Knopf to obtain this version.

8. FLOWTRAN Exercise Book

The second edition of *Exercises in Process Simulation Using FLOWTRAN*, edited by J. Peter Clark, T. P. Koehler, and J. T. Sommerfeld of Georgia Tech., is now out of print. The task force is promoting the possibility of a 3rd edition.

Articles About CACHE Task Force Activities

1. Clark, J. Peter and J.T. Sommerfeld, "Use of FLOWTRAN Simulation in Education," *Chemical Engineering Education*, Vol. 10, No 2, Spring 1976, p. 90.
2. Hughes, R. R. "CACHE Use of FLOWTRAN on UCS," CACHE Inc., Cambridge, Massachusetts, 1975.
3. Biegler, L., B. Carnahan, J. D. Seader, and G. V. Reklaitis, "CACHE and the Dissemination of Software for ChE Education," Proceedings of UP CAEDM-87, pp. 11–19, Ohio State University (1987).
4. Edgar, T. F., R.S.H. Mah, G. V. Reklaitis, and D. M. Himmelblau, "Computer Aids in Chemical Education," *CHEMTECH*, Vol. 18, 1988, p. 227-283.
5. Edgar, T. F., R.S.H. Mah, and G. V. Reklaitis, "Use of Computers in Chemical Engineering Education," *Chemical Engineering Progress*, Sept. 1988.
6. Seader, J. D., "New Optimization Feature for FLOWTRAN," *CACHE News*, Spring 1988.
7. Seider, W. D. and A. W. Westerberg, "Utilization of Networks for the Sharing of Computer-Based Resources within Chemical Engineering," NSF Report on the Workshop held September 28–29, 1978, Washington, D.C.

Laboratory Applications of Microcomputers Task Force

Yaman Arkun, Georgia Institute of Technology
Ali Cinar, Illinois Institute of Technology
Duncan A. Mellichamp, University of California, Santa Barbara

Mission Statement

The mission of the Laboratory Applications of Microcomputers Task Force is to assist chemical engineering departments in developing undergraduate laboratory experiments involving on-line computer applications. The goal is to give a picture of the wide array of classical and modern engineering concepts that have been successfully built into computer-aided laboratory experiments, of the very different hardware and software approaches, and of the pedagogical objectives that were fulfilled when the computer was introduced.

Completed Projects

During 1986 CACHE conducted a survey of U.S. and foreign chemical engineering departments to determine just how widely on-line computers were being applied in the undergraduate chemical engineering laboratory. Our prior expectations were that microcomputers and personal computers would be found in many new laboratory applications, primarily because of technical advances that have led to greatly improved computer performance with significantly lower prices. Also, the easy availability and increased utilization of interfacing hardware and software in the last half dozen years have eliminated much of the mystery and difficulty involved in connecting a computer to a laboratory process.

The survey results, published by CACHE in September 1986 [1], confirmed the expectations: many departments, including many smaller and less research-oriented departments, have introduced microcomputers and personal computers into the classical undergraduate laboratory. In accomplishing this, a great deal of innovation was required and much diversity has resulted. From the schools that responded to the survey, 54 different groups (classifications) of experiments such as reactors and distillation columns were counted. Within each group there were as many as 16 different variations.

The survey was intended to serve as a forerunner to the CACHE anthology on on-line computer applications in the undergraduate chemical engineering laboratory. In this anthology, which was published in the summer of 1988 [2], three major experimental areas from the classical chemical engineering curriculum are covered: (1) thermodynamics, fluid dynamics, heat and mass transfer, (2) chemical and biochemical reactors, and (3) process dynamics and control. Of the twenty-one experiment descriptions included, three are in the reactor area and nine each in thermodynamics/transport and in control.

The distinguishing feature of all write-ups included in the anthology is an attempt to focus on the fundamental engineering phenomena that are involved. In each case, the authors show how an on-line computer, usually an inexpensive micro- or personal computer with off-the-shelf input/output cards, can be used effectively to take data from a process or operating unit, analyze these raw results using theory at the level of a typical undergraduate lecture course, and display the final results. Student responses and the cost of the experiment are also given. Particular details concerning construction of the experimental unit and instrumentation are included. In most cases, the type of computer used is relatively unimportant; and the software is easy to develop, can be obtained from the authors or is available as a commercial package. These features make program development quite simple.

The experiments selected show the diversity of approaches that can work in the laboratory. For example, at the very simple end of the spectrum, two experiments involving unsteady heat transfer—in a cylinder (Karman) and in a sphere (Heist, Olsen, Saltsburg)—require only that a single thermocouple embedded in the cylinder/sphere be interfaced to the computer. Each experiment emphasizes its own variation of data analysis in line with the goals of the developer.

For example, in the experiment involving unsteady state heat transfer in a cylinder, students recognize the significance of nondimensional quantities such as reduced temperature difference, Biot and Fourier numbers, and they take a critical look at the convective boundary condition in unsteady state heat transfer. The apparatus is shown in Fig. 1. The QUICKI/O Software supplied with ADALAB analog/digital card provides extensions to Applesoft BASIC so that data acquisition

commands are easily incorporated into programs in BASIC. Interaction with the student is through the video monitor and the printer. A sample of the dialogue and the plot of the experimental results on the Heisler chart are shown in Figs. 2a and 2b.

Several contributors describe inexpensive but flexible experimental units that can serve multiple purposes, particularly in a control laboratory. For example, Stadtherr and Masel use several copies of the same experiment (air flow temperature control) sketched in Figs. 3a and 3 b. The students use this apparatus for about eight weeks (three hours/week) for various purposes, including introduction to the use of PC/XT hardware and ASYST software, data acquisition, sam-

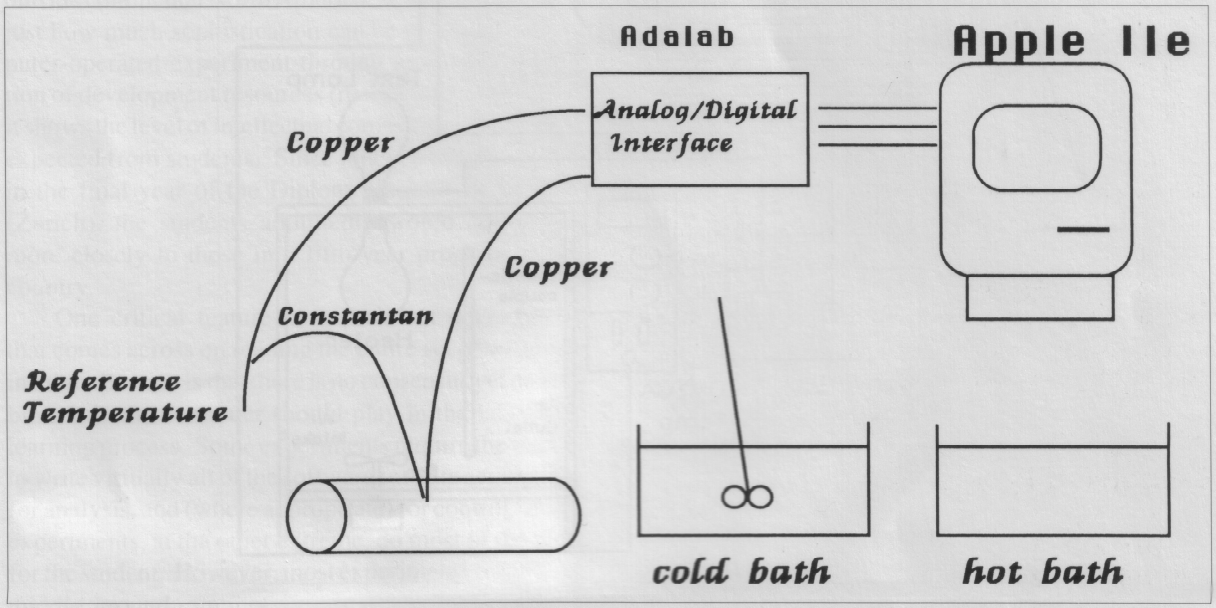


FIGURE 1. Experimental Apparatus.

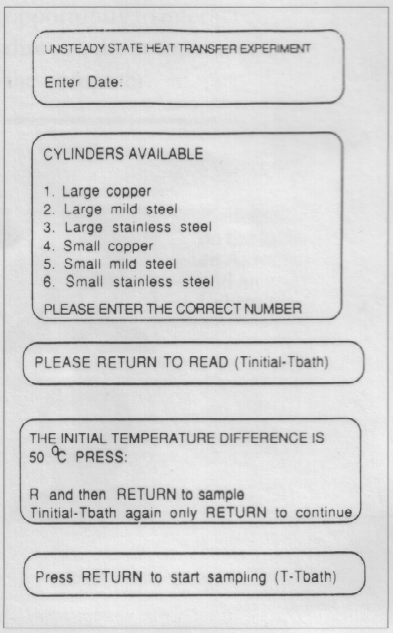


FIGURE 2a. Prompts appearing on the video screen during a run.

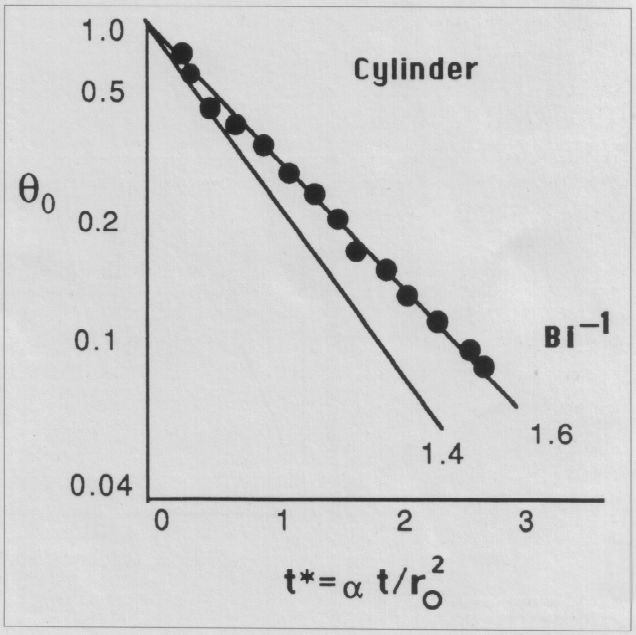


FIGURE 2b. Typical experimental results plotted on a Heisler Chart.

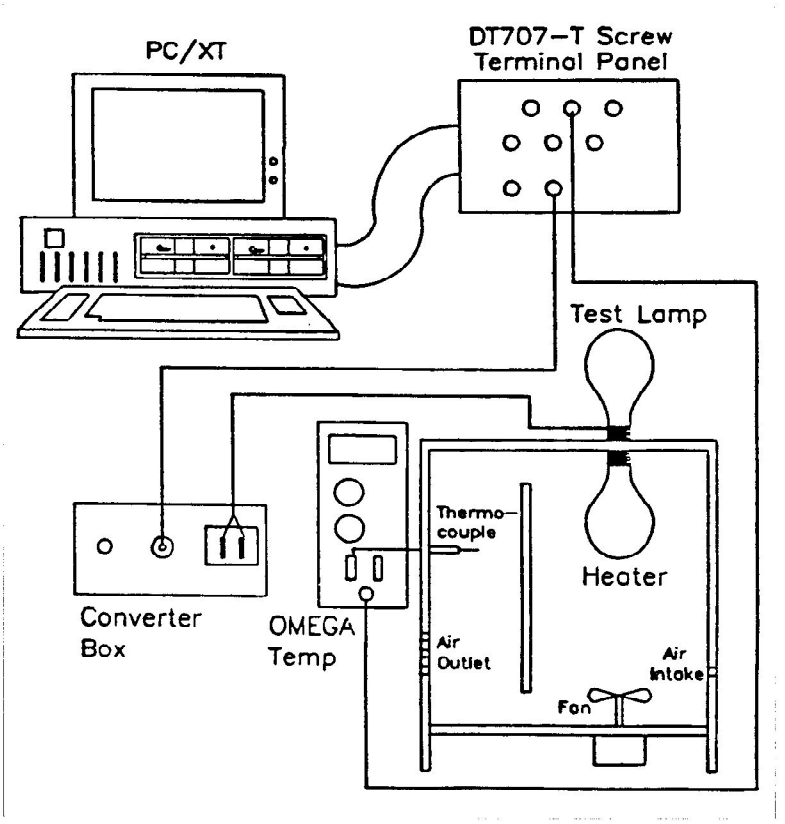


FIGURE 3a. Schematic of of apparatus.

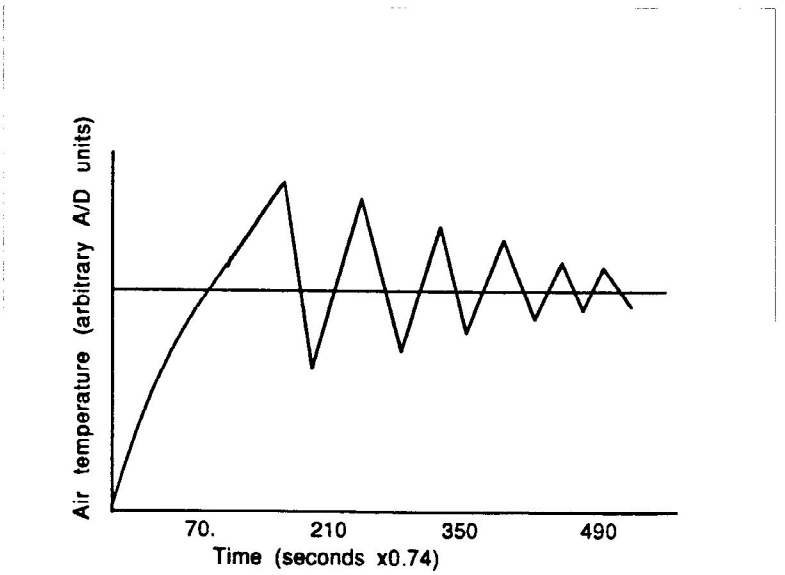


FIGURE 3b. Set point response for a poor PID Controller.

pling and filtering, open-loop response analysis, and finally PID controller design.

At the other end of the complexity spectrum are the biochemical reactor experiment (Cooney and McDonald) and the sequential control experiment—a coffee grinder/brewer/dispense (Keller and Reinhart). The latter unit is in a class by itself for several reasons. First, it is the only contribution we obtained from outside continental North America; second, it illustrates just how much sophistication can be built into a computer-operated experiment through generous application of development resources (money and time); third, it shows the level of intellectual commitment that can be expected from students. Since this experiment is used in the final year of the Diplom program at the ETH (Zurich), the students admittedly would correspond more closely to those in a fifth-year program in this country.

One critical feature of experiment development that comes across on reading the entire set of write-ups in this collection is that there is no consensus yet on how big a role the computer should play in the laboratory learning process. Some experiments require the student to write virtually all of the software for data acquisition, for analysis, and (where appropriate) for control. A few experiments, at the other extreme, do most of the work for the student. However, most experiments occupy the middle ground, attempting to enhance the student's learning experience by eliminating much of the drudgery that often is associated with undergraduate laboratory work. At the same time, the student is given the opportunity to interact easily with the experimental unit through the computer and to interact with data taken by the computer.

The objective in designing experiments should be to let students apply analysis at a higher level, and not to replace analysis entirely in the laboratory procedure. Some of the experimental systems described in the anthology have achieved this goal better than others, but all suggest ways in which the laboratory experience can be revitalized and strengthened.

Needs

The task force is soliciting feedback on the anthology and descriptions of new microcomputer applications in all areas of interest to chemical engineers. Due to rapidly changing hardware and software technology and emerging new fields, we plan to monitor these developments closely and hope to continue assisting our colleagues in developing innovative and useful undergraduate laboratory experiments using on-line computer applications.

References

1. Cinar, A and Mellichamp, D. A., *Survey: Experiments Involving On-Line Computer Applications in Chemical Engineering Undergraduate Laboratories*, CACHE, Austin, TX (Sept. 1986).

2. Mellichamp, D.A. and Cinar, A, *On-Line Computer Applications in the Undergraduate Chemical Engineering Laboratory: A CACHE Anthology*, CACHE, Austin, TX (June 1988).

The Authors

Professor Arkun received his Ph.D. from the University of Minnesota in 1979. From 1979 to 1985 he was on the faculty at Rensselaer Polytechnic Institute. He joined Georgia Tech in July 1985, where he is currently an Associate Professor. His research areas are in process control. Dr. Arkun is a member of AIChE and IEEE, and an associate editor for the *Journal of Automatica*. He is a CACHE trustee and heads the Task Force on Labortatory Applications of Microcomputers.

Dr. Ali Cinar is currently an associate professor at Illinois Institute of Technology. His research interests are chemical process control and artificial intelligence applications. His teaching activities include process design, process control, expert systems courses and development of computer interfaced experiments in reaction engineering and process control. Dr. Cinar received his B.S. in chemical engineering from Robert College, Turkey. After completing his Ph.D. studies at Texas A & M University in 1976, he became a faculty member at Bogazici University, Turkey. He spent a year as a visiting faculty member at the University of Wisconsin, Madison in 1981, before joining the Illinois Institute of Technology in 1982.

Duncan A. Mellichamp is Professor of Chemical Engineering at the University of California, Santa Barbara. He was CACHE trustee for fourteen years and president in 1977-78. He is the editor of a book, *Real-Time Computing with Applications to Data Acquisition and Control*, published in 1983 and coauthor of a book, *Process Dynamics and Control*, to be published in May of this year.

Discrete-Event Simulation in the Curriculum

Daniel J. Schultheisz and Jude T. Sommerfeld
Georgia Institute of Technology

Abstract

This paper describes experience with introducing a discrete event simulation program — specifically, GPSS—into the chemical engineering curriculum. Primary applications here are in the area of computer-aided design of batch processes. Research work resulting in the publication of numerous practical applications is also summarized. This work has culminated in the preparation of an instructor's manual on this topic, to be published and distributed by CACHE.

Introduction

With the commissioning of many world-scale chemical commodity plants in the early 1980's, an intense revival of commercial interest in batch chemical processes occurred in most Western countries. Such processes are generally associated with the manufacture of fine and specialty chemicals, in addition to allied products.

At about the same time, and in response to greater educational needs in the design of flexible manufacturing systems (FMS), the IBM Corporation funded the initiation of graduate computer-integrated manufacturing systems (CIMS) programs at Georgia Tech and several other schools. The chemical engineering contributions to this program included the development of a new graduate level course on the computer-aided design of batch chemical processes. Early offerings of this new course emphasized discrete-event simulation of batch chemical processes, using the general purpose simulation system (GPSS) [1,2]. This topic was subsequently downloaded into our senior-level undergraduate course on computer-aided process design. The original graduate course has since concentrated on more sophisticated combined (discrete + continuous) systems for batch process simulation, such as SLAM [3] and BATCHES [4]. A considerable amount of applied research work has also been performed in conjunction with this educational effort, resulting in a variety of published applications (Table 1). This work has culminated in the preparation of an instructor's manual [15], entitled "Exercises in Chemical Engineering, using GPSS," similar to the previously published (by CACHE) manual [16] of FLOWTRAN exercises in chemical engineering. This new manual and accompanying diskette of software are also distributed by the CACHE Corporation, the availability of which is described in another part of this newsletter. The purpose of this article is to assist users of these materials (particularly those who use PC's) in this exciting new area. Interested readers are also referred to recent tutorial articles [17, 18] on discrete-event simulation in chemical engineering.

Discrete-Event Simulation

Discrete-event simulators were originally developed as numerical aids to solve complex queuing theory problems [19] not amenable to analytical solution. Such problems occur routinely in the field of industrial

Table 1

Recent Applications of GPSS to Chemical and Allied Processes

Application	Ref.
DDT manufacture	5
Chocolate manufacturing	6
Large-scale poliomyelitis vaccine production	7
Sequence of batch distillation columns	8
PVC manufacture	9
Choline chloride manufacture	10
Penicillin synthesis	11
Textile finishing mill	12
Wastewater treatment	13
Thermotropic polymer plant design	14

engineering; typical example applications include machine shops, materials handling facilities, customer service stations, and transportation networks.

Discrete simulators have a simulation time clock. The proper scheduling of events in the time domain is a formidable task and is typically implemented by the internal logic of the simulator. A time event is one whose timing depends upon some predecessor event and can generally be predicted. Examples of such time events would include the arrival of a part or a batch (following a previous one) and completion of a movement or processing activity. Most discrete simulation systems feature stochastic capabilities to model human factors and thus have one or more built-in random-number generators. Output from these latter is used to sample event times (or durations between time events) from various probability distributions. Thus, for example, a processing time can follow the uniform or rectangular distribution; that is, this time can take the form $A \pm B$, where A represents the mean value and B is the half-width, in appropriate time units, of the distribution. Empirical, user-developed distribution functions, e.g., from a plant histogram, can also be supplied by an analyst.

GPSS Processor

The progenitor of discrete-event simulation systems is GPSS, which dates back to 1959 and is still used extensively in many manufacturing sectors. Because of its easy use, availability, reliability, and efficient operation (integer arithmetic only in many versions), GPSS is a very effective tool if only discrete simulation capability is required. Other popular discrete-event simulation systems include SIMULA [20], more prevalent in Europe, and SIMSCRIPT [21].

There are 35-40 different precoded functional subroutines, known as blocks and generally written in FORTRAN, in GPSS. The items which move from block to block in a GPSS model are known as transactions. Such transactions could represent machine assembly parts or batches of material in a production facility. The capabilities of the blocks themselves vary from simple to complex. In GPSS, there is the ability to detain transactions in a queue, waiting for processing or some other event to occur. Single servers (e.g., a single machine or vessel) are designated as facilities, and there is a block (named ADVANCE) to represent the time lapse associated with processing a transaction in a facility. A battery of multiple, parallel, identical servers (facilities) in GPSS is known as a storage. A bank of identical, parallel reactors in a batch chemical process would then be represented by a storage in GPSS.

GPSS Output

As with a system such as FLOWTRAN [22], which provides a summary table of the streams passing through the model and output results from each of the blocks in the model, GPSS automatically prints a variety of output statistics at the conclusion of a simulation. These statistics pertain primarily to the various facilities, queues, and storages in the model.

Thus, from an inspection of the facility output statistics from a GPSS simulation, an analyst might find that the average holding time per transaction for a given facility is considerably greater than the user-supplied average service time for that facility. In a chemical engineering application, for example, this could indicate that a reactor, after finishing processing of a batch (transaction), often cannot discharge the batch because of an unavailable downstream facility. The latter might correspond to a storage tank which is full or another processing unit (e.g., still, centrifuge, dryer) which is engaged. The regular occurrence of such a situation would normally be accompanied by an average utilization (fraction of total time busy) approaching unity for the original upstream facility, and would suggest the existence of some downstream bottleneck. The existence of similar bottleneck situations can also be deduced from the output statistics for GPSS storages.

The output statistics for queues also represent very valuable information. Thus, high average values for the queue contents and high average waiting times per transaction would again result from some downstream bottleneck in the process being modelled. The model could then be easily and appropriately modified, perhaps by additional or more efficient downstream facilities, representing proposed process modifications. The productivity (number of batches produced) of the modelled process is, of course, related to the number of transactions passing through the GPSS model.

GPSS on a PC

The PC version of GPSS used in development of the instructor's manual [15] was developed by Minuteman Software of Stow, Massachusetts. It is a completely interactive software package and as powerful as a mainframe version of GPSS. It is available in both student and industrial versions. The student version lacks some of the more advanced features, such as the Session Journal, and also runs somewhat more slowly. GPSS/PC requires at least 320K of memory, but larger

models will require up to 640K. All of the simulations presented in the manual were performed with a 640K capacity, using the industrial version (Version 2).

Output Reports

GPSS/PC includes its own report program, GPSSREPT, which formats the output report. The user can designate the output report file and supply the title for the report. If no file is specified, the report will be placed in the default report file. The user can also suppress the report, or have an unformatted report written immediately, as soon as the statement is read. The user has the option of having the formatted report displayed on the screen or sent to a printer. The user can also designate a DOS file for the unformatted report in the REPORT statement or, by using GPSSREPT, send the formatted report to a DOS file.

The formatted report contains all of the information relevant to the simulation. It begins with general information, such as the length of the simulation, number of blocks, facilities, storages, and free memory. The report then gives a detailed listing of each GPSS block involved in the simulation, supplying the line number, location (number or name), block type, the number of transactions to enter the block, the number of transactions currently in the block, and the number of transactions waiting for a specific condition that depends on the state of the block. Only executable GPSS blocks, such as ADVANCE, are listed here. GPSS control statements, which create non-block entities or set up conditions for the running of the simulation, are not included. VARIABLE and MATRIX, for example, are control statements.

The report then lists such entities as facilities, queues, and storages, listing such statistics as average utilization, average time, remaining storage capacity, maximum queue content, and number of entries. Following these will be tables, logic switches (set or reset), savevalues, and matrices. The transaction group POSITION is always listed, as it is reserved for the Positions Window, to be discussed later. Table statistics will include the mean and standard deviation of tabulated values, the range (lower and upper limits of tabulated values), frequency (weighted count of items falling within the range), and the cumulative frequency count expressed as a percentage of the total count. Reports will also include information on user chains, numeric groups, and the current events chain, if necessary.

Running a Simulation

GPSS/PC makes it very easy to initiate a simulation, as the software itself institutes the proper format. The user inputs the line number. The space bar then provides the correct spacing to the location field. The next space prompts for the block name, then block operands, and finally for the comments field. GPSS/PC will not allow the use of location names that indicate block or control statements. For example, a location could not be labelled GENE or LOG, as these are reserved for GENERATE and LOGIC statements, respectively. Also, the use of SNAs (Standard Numerical Attributes) are not permitted in the location field. A block could not be labelled MX1, as MX1 indicates the values stored in the matrix whose name is 1.

GPSS/PC is self-correcting in that it calls the attention of the user to any syntax or spelling errors by use of a warning sound. GPSS/PC will not allow any input that would lead to an incorrect result. For example, if the desired block is GENERATE, the user could not accidentally misspell the word as GENREATE. The warning would sound at the input of the R, and the cursor would remain at its proper place.

The comments field is denoted by a semi-colon (;). If it is desired to use an entire line for comments, it can be done by following the line number with a semi-colon or asterisk (*). Some statements, such as VARIABLE, do not have a comments field.

When the entire model has been input, the simulation can be initiated by use of the START command, with whatever operand is relevant to the simulation. GPSS/PC allows the user to stop the simulation at any time. This may be useful if the user wishes to delete or insert a block, or alter the order of the existing blocks. By using the STEP or STOP commands, the user can control how far the simulation will proceed. The user can then CONTINUE the simulation until it reaches completion.

Visual Innovations/Windows

Because the user cannot view the formatted report without accessing the report program, GPSS/PC has developed a number of graphic capabilities (windows) for use during and after the simulation. The most useful of these innovations allows the user to observe the simulation from several different points of view. By looking into windows, the user can closely follow the development of the simulation, which can greatly aid understanding of the dynamics of the situation. The Blocks Window (Figure 1) graphically illustrates the

movement of transactions through the model. Each block is represented and labelled on the screen. The block is highlighted as the active transaction enters. The number of transactions currently in the block is listed. In this way, the user can see if an unexpected breakdown occurs and immediately take corrective action. The user can also display a cumulative Blocks Window, which will indicate the total number of transactions to enter each block.

The Facilities Window (Figure 2) depicts each facility. It provides the information that is usually found in the report, such as the average utilization and the size of the queue. It also gives the average transaction time and the transaction number of the current owner. For those terminals with color capability, the display color depends on the fractional utilization. The Storage Window (Figure 3) provides similar information. It displays the amount of storage available, average utilization, queue size, and the maximum storage in use at any time during the simulation. The Tables Window (Figure 4) gives the mean and standard deviation of the table data, and displays the upper bound of the frequency counts versus the frequency boundaries (intervals). The Matrices Window (Figure 5) displays the values of the defined matrices. The user can watch to see that the proper values, as defined by an MSAVEVALUE block, are placed in the matrix.

The Positions Window (available only in Version 2) allows the user to pictorially represent the simulation. By varying the shape and/or color of the icons, there are 1500 distinct combinations. The monitor definitely requires color graphics capability. The procedure for this type of animation is far too complex for inclusion here.

All of these windows can be opened either through a command in the model or through the keyboard. In addition, microwindows can be opened through a command. Up to four microwindows can be observed at any time, each displaying a relevant SNA. The microwindows are placed at the edge of the larger window. As noted earlier, these windows allow observation of the simulation in progress. However, opening these windows will cause the simulation to proceed much more slowly, as the time it takes to represent the movement of a transaction on the screen (through the simulation, or its effect on a facility or storage) is far greater than that required for the transaction actually to complete its task. Thus a simulation normally requiring only two minutes to complete may require 20 minutes or more if the windows are opened. This is especially true if the time units involved in the simulation vary widely. For example, if one step takes one minute and another takes 10 days, the great disparity in these time units will considerably lengthen the simulation time.

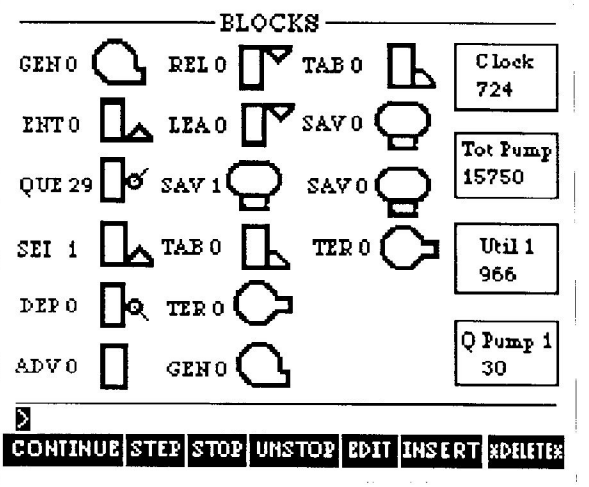


FIGURE 1. Typical view of a Block Window from GPSS/PC (courtesy of Minuteman Software).

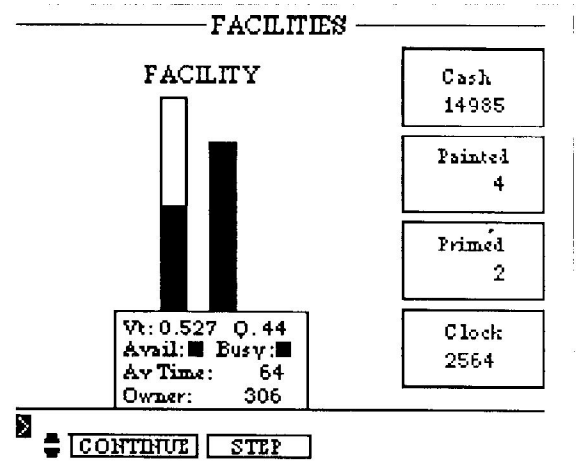


FIGURE 2. Typical view of a Facilities Window from GPSS/PC (courtesy of Minuteman Software).

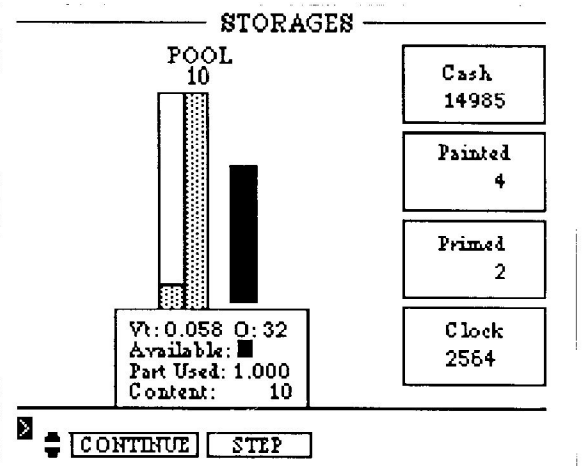


FIGURE 3. Typical view of a Storage Window from GPSS/PC (courtesy of Minuteman Software).

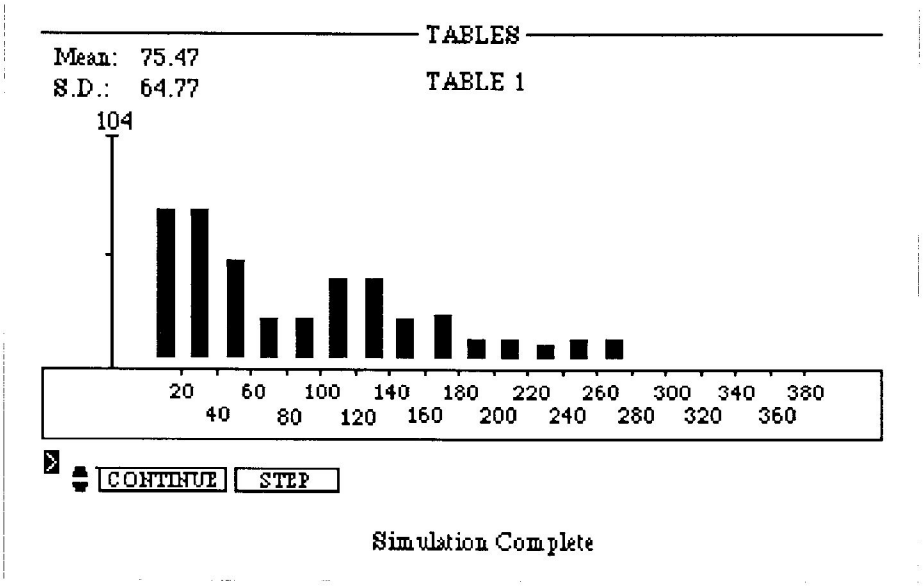


FIGURE 4. Typical view of a Tables Window from GPSS/PC (courtesy of Minuteman Software).

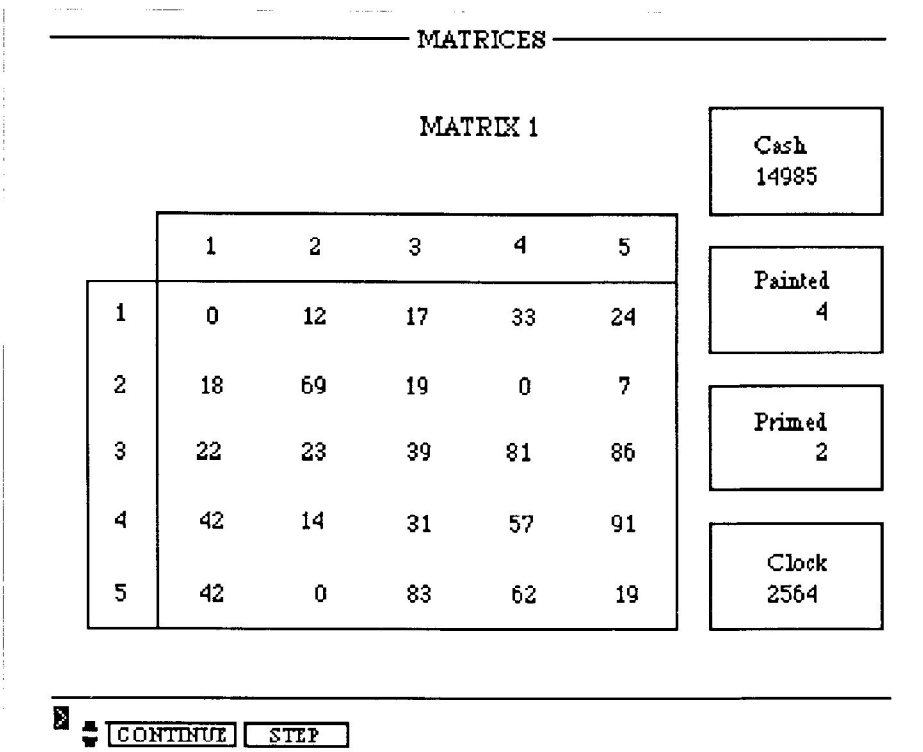


FIGURE 5. Typical View of a Matrices Window from GPSS/PC (courtesy of Minuteman Software).

These windows are extremely useful for checking the results of the completed simulation, and for multiple runs if a parameter is altered. Use of the CLEAR statement following a simulation, however, will cause a matrix to be initialized to zero. Either the model must be read again or INITIAL statements must be introduced.

Plots/Mouse

GPSS/PC also allows the user to create a plot of a relevant SNA in the Data Window during a simulation (Figure 6). This is done by use of the PLOT command. In addition, GPSS/PC allows the use of a mouse or light pen during the simulation. This can be useful for selecting specific blocks that one may wish to alter or delete. It is also possible to force a transaction to become active.

Standard Numerical Attributes

Following a simulation, GPSS/PC allows the user to access any SNA. This is especially useful if the desired SNA is not one available from one of the windows. For example, if the user wished to know the average residence time in a queue named LINE, it would only require typing of SHOW QT$LINE. The relevant statistic would then be displayed on the screen. The SHOW command can be used for any of the approved SNA codes, of which there are 52 in GPSS/PC. This makes it easier to run the simulation under different conditions, if a full report is not desired for each outcome. In some cases, the user may want to use indirect addressing, which can be very useful. The user should remember that if an SNA is part of the expression used in the address, that SNA cannot be evaluated unless there is an active transaction.

Statistical Analysis

GPSS/PC offers a unique and powerful statistical analysis option, called ANOVA. Extremely useful for large and complex models, this package allows the user to analyze data for confidence levels. In a statistical program such as GPSS, there is very often a noticeable range in the results. ANOVA will take the results from several runs, most likely performed using different random number seeds, and return such important statistical quantities as the mean, standard deviation, and 95% confidence level (Figure 7). ANOVA can also be employed to test whether the results obtained are statistically significant. That is, it can tell the user whether further runs are required. It will only accept results that have not been perturbed by interactions, either within the simulation or from the user.

Session Journal

The Session Journal is an optional ASCII file that collects all statements made by the user and any GPSS/PC messages. It can be used to collect trace messages, ANOVA results, error messages, and entity definitions, among other things. It can be especially useful if the user desires to load a model designed under a different GPSS system. The Session Journal will associate any error statements with the appropriate source statements. In this way, the user will find it easier to correct any syntax errors.

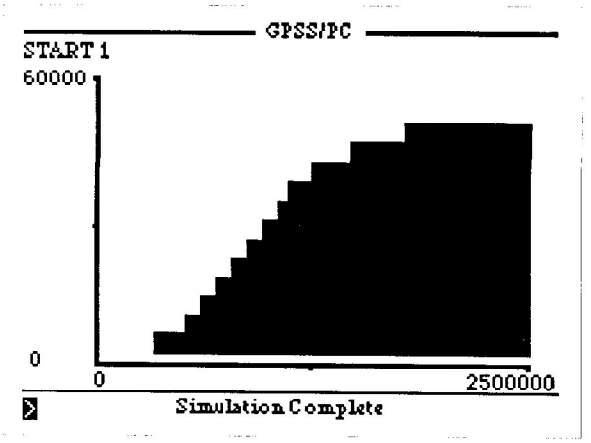


FIGURE 6. Typical plot of an SNA for GPSS/PC (courtesy of Minuteman Software).

GPSS/PC

ANOVA

Source of Variance	Sum of Squares	Degrees of Freedom	Mean Square	F
Treatments	76480.333	1	76480.333	0.410
Error	747007.000	4	186751.750	
Total	823487.333	5		

Treatment	Count	Mean	Std. Dev	Min	Max.
2	4	251.50	499.00	1	1000
3	2	12.00	1.41	11	13

FIGURE 7. Example of statistical analysis (ANOVA) output from GPSS/PC (courtesy of Minuteman Software).

Further details on GPSS/PC, as well as areas of difference between it and mainframe versions (e.g., GPSS V), are given in the instructor's manual [15].

References

1. Schriber, T. J., *Simulation Using GPSS*. Wiley, New York (1974).

2. Dunning, K. A., *Getting Started in GPSS*, Engineering Press, San Jose, CA (1985).

3. Pritsker, A. A. B., *An Introduction to Simulation and Slam II*, 3rd Ed., Halsted Press, New York (1986).

4. Carmichael, D. S., S. M. Clark, and G. S. Joglekar, *BATCHES Reference Manual*, Batch Process Technologies, Inc., West Lafayette, IN (1987).

5. Blaylock, C. R., C. O. Morgan and J. T. Sommerfeld, "GPSS Simulation of DDT Manufacture," *Proc. 14th IASTED Intl. Conf. on Appl. Sim. and Mod.*, Vancouver (June, 1986), p. 314.

6. Passariello, I. and J. T. Sommerfeld, "GPSS Simulation of Chocolate Manufacturing" in *Tools for the Simulation Profession*, Soc. for Comp. Sim., San Diego (1986), p. 1.

7. Kenvin, J. C. and J. T. Sommerfeld, "Discrete-Event Simulation of Large-Scale Poliomyelitis Vaccine Production," *Proc. Biochem.*, 22, 74 (1987).

8. Barnette, D. T. and J. T. Sommerfeld, "Discrete-Event Simulation of a Sequence of Multi-Component Batch Distillation Columns," *Comp. and Chem. Engr.*, 11, 395 (1987).

9. Adebekun, A. K., Z-Q. Song and J. T. Sommerfeld, "GPSS Simulation of PVC Manufacture," *Polymer Proc. Eng.*, 5, 145 (1987).

10. Bales, W. J., J. R. Johnson and J. T. Sommerfeld, "Use of a Queueing Simulator in Design of a Batch Chemical Production System," *Prodn. and Inventory Mgmt. J.*, 29(2), 36 (1988).

11. Ransbotham, S. B., J. A. Schwarzhoff and J. T. Sommerfeld, "Discrete-Event Simulation of Penicillin Production," *Proc. Biochem.*, 23, 182 (1988).

12. Livingston, D. L. and J. T. Sommerfeld, "Discrete-Event Simulation in the Design of Textile Finishing Processes," *Textile Rsch. J.*, in press.

13. Glenn, S. L., R. T. Norris, Jr. and J. T. Sommerfeld, "Discrete-Event Simulation of Sequencing Batch Reactors (SBR) for Wastewater Treatment," *Proc. Amer. Chem. Soc. 1989 Ind. & Eng. Chem. Spring Symp.*, Atlanta (May, 1989), in press.

14. Lin, L., M. L. McKendry and J. T. Sommerfeld, "Design of a Thermotropic Liquid Crystal Polymer Plant Using GPSS," *Chem. Engr. Commun.*, submitted for publication.

15. Schultheisz, D. J. and J. T. Sommerfeld, *Exercises in Chemical Engineering, using GPSS*, CACHE Corp., Austin, TX (1988).

16. Clark, J. P., T. P. Koehler and J. T. Sommerfeld, *Exercises in Process Simulation Using FLOWTRAN*, 2nd Ed., Ulrich's Bookstore, Ann Arbor, MI (1980).

17. Schultheisz, D. J. and J. T. Sommerfeld, "Discrete-Event Simulation in Chemical Engineering," *Chem.Engr. Educ.*, 22(2), 98 (1988).

18. Schultheisz, D. J. and J. T. Sommerfeld, "Discrete Simulation Can Optimize Your Batch Process," *Hydrocarbon Proc.*, submitted for publication.

19. Banks, J. and J. S. Carson, II, *Discrete-Event System Simulation*, Prentice-Hall, Englewood Cliffs, NJ (1984).

20. Franta, W. R., *The Process View of Simulation*, Elsevier North-Holland, New York (1977).

21. Wyman, F. P., *Simulation Modeling: A Guide to Using SIMSCRIPT*, Wiley, New York (1970).

22. Seader, J. D., W. D. Seider, and A.C. Pauls, *FLOWTRAN Simulation--An Introduction*, 2nd Ed., Ulrich's Bookstore, Ann Arbor, MI (1977).

Daniel J. Schultheisz and Jude T. Sommerfeld
School of Chemical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Image Processing and Analysis in Chemical Engineering: The Justification for Specialized Workstations and the Need for Computational Power

Robert S. Brodkey
The Ohio State University

Image processing is the reduction of visual data to a form that is convenient to manipulate on a computer. Image analysis is further manipulation of the computer data to extract quantitative information. The analysis depends on the specific research problem and is a step beyond image enhancement where the visual material is modified just for improved visual interpretation.

In many research areas there are operations in which understanding could be enhanced if simple photos or multiple photographic records could be analyzed easily. In the past, we have used films (both normal speed and high speed) to gain qualitative understanding of what is occurring but were unable to extract additional quantitative information that would be very useful. Unfortunately, one cannot just take visual information, reduce it to digital form, and expect to obtain anything useful. One must first establish the type of digital information needed from the visual record and then develop the software necessary to extract that information in an efficient manner, recognizing that massive amounts of data are involved.

Actually, image processing and analysis techniques have been utilized by researchers for many years. Usually though, these imaging facilities are for one specific purpose such as particle size analysis. However, recently the field has experienced rapid expansion due to solid-state sensor development which has resulted in high-resolution devices. The newer developments permit critical quantitative information to be obtained directly from visual imagery in a more general manner than has been previously possible. With this in mind, we have established a high-resolution, color image processing and image analysis system. This system is integrated with the full range of computer facilities at Ohio State. The imaging system operates with its own MicroVax II, which has a captive 400 Mbyte disk. The MicroVax itself is networked to our graphics laboratory VAX 8550. The entire system is connected in turn to the University's X-MP/28 supercomputer that will be upgraded to a Y-MP/864.

First a few words about the facilities: The front end is an Eikonix high resolution linear diode array system. The camera has a resolution of 2048x2048 digital elements (pixels) for any image size from an 8mm film negative to an 8x11 in. format (positive or negative). Such high-resolution systems are inherently slow (about 10 minutes for a full color picture). The Eikonix system digitizes directly into our Dipix Aries-III workstation. The Eikonix is equipped with the camera unit itself, array detector, color filter wheel, monitor scope, normalizer, several high quality 35mm lens, a 5-in. condenser, reflective and transmissive illumination sources, and all the necessary interface and drivers to allow it to be run as a subset of the workstation.

The workstation is a Dipix, Inc. ARIES-III system, which is strong on software when compared to other systems. Of major importance is the design principle that does not restrict the size of the array being processed. First, the memory is dynamic in that any size can be assigned to any specific array. When an operation is done on the array, the entire array can be processed. Thus, fitting at boundaries, either for filtering or warping, is not needed. It is very important for our work to be able to use large arrays. The output array from the Eikonix camera is 4 Mbytes for each of three color filters and for a black and white composite; thus, we have 16 Mbytes of information. Note that the disk-to-disk processing means that the Dipix design is not limited in size and can process images of any size and depth. The trade-off is size for time. Although only a limited part of a very large array can be displayed at any one moment, one can move the view to anywhere within the array that is in memory and zoom to any magnification. The batch operations can be done on the MicroVax, the 8550, or even on the CRAY. So far the latter option has not been used.

The motivation for the facilities was our research on coherent structures in turbulent shear flows. This will be discussed further, but our experience has been that image operations are a tool and that anytime visual perception can be used, imaging can help obtain accurate data normally not available. Thus, we have found that applications in chemical engineering have gone far beyond our original desire to study coherent structures. For example, we have used the facility to study liquid drop breakup in a well characterized (extensional) turbulent flow field, to interpret stereoscopic scanning electron microscopic (SEM) pictures of catalyst surfaces, for the surface characterization of polymeric composite materials, and to obtain descriptive details

about the orientation and possible entanglement of fibers in paper flocs and in polymers. Still other applications are visual observations obtained during rheological measurements, low velocity vector field estimation for control in microgravity crystallization experiments, determination of sheer effect on cell viability, and estimation of the stresses caused by the biting action of human teeth. In effect, we have a new tool that can be used to extract additional information beyond just looking at pictures.

To illustrate the usefulness of the approach, a few additional words on the coherent structures problem are in order. Turbulence and mixing go hand in hand; thus, it is understandable that the turbulence generated by a mixing unit can have a major effect on the time required and the degree of mixing obtained. While our understanding of turbulence is still relatively poor, the approaches used for turbulence research have changed drastically during the last fifteen years. The concept of using overall averages for the mean and fluctuating components has given way to a more fundamental view that turbulent shear flows are composed of a sequence of complex coherent motions or structures. Progress in turbulence research depends on better understanding of the mechanism of turbulent flows gained through the study of such motions or structures. Indeed, improvements in processes involving complex turbulent shear flows will require understanding the effect of coherent structures on the processes. Thus, our goal is to produce a model picture of the complicated unsteady, three-dimensional flow patterns that occur during turbulent transport. Such knowledge can facilitate our understanding of turbulent flow structure, leading later to rational methods of modeling and modifying the effects of turbulence.

Research in fluid mechanics and more specifically in turbulence has been hampered by inadequate probing techniques. These techniques are usually limited to point measurements and do not represent adequately the underlying three-dimensional physics. Moreover, vorticity and strain rate require spatial differentiation of velocity field data and thus are essentially inaccessible to the experimentalist. Probes are good for time dependent information at a few points in the flow, but one can imagine the problem of probe interference that would develop if one wanted to place probes in a three-dimensional array. With the advent of modern image analysis, visual information can be converted into useful quantitative information that will enhance understanding of these very complex processes.

The immediate goal of our current turbulence research is to obtain three-dimensional quantitative data to help establish a mechanistic picture of coherent structures in turbulent shear flows. This would help close the gap between visual observations and anemometry measurements. To accomplish this, the detailed nature of visually observed coherent motions needs to be converted into quantitative information. To obtain the proper information involves determination of the vector velocity field for individual events by high resolution image analysis and, from this, establishing the vorticity and strain fields. This is not an easy task: recall that vorticity involves the difference between the velocity gradients. But once accomplished, it will be easier to identify coherent motions by alternate visual methods and by more extensive anemometry measurements. The former will provide the mechanistic picture and the latter will provide the statistical ensemble averages. The final step will be to incorporate the physical understanding obtained into an analytical model of the turbulent field and of the turbulent production and dissipation mechanisms.

The specific tasks that we have had are 1) set up a simple turbulent mixing tank, mark the flow with colored particles, investigate oil-soluble dyes that can enhance the color and light output from the particles, and generally improve the photographic techniques for better images of the flow field; 2) make available the entire image for analysis without compressing the data and losing information; 3) adapt filtering techniques for removal of background noise and improving edge detection by derivative methods; 4) develop procedures to reduce the large volume of data (up to 16 Mbytes) that would be generated from a full color image and its black and white counterpart and develop procedures for particle identification; and 5) develop automated techniques to match, locate, and track each particle from frame to frame; this will be used to determine the velocity and derivatives for vorticity evaluation.

Each of the preceeding steps have been demonstrated and our current efforts are directed to the perfection of the techniques before large-scale reduction of films is done. We have established a group of color oil-based dyes that allow color to be used as a dimension to allow higher particle concentrations to be used. The in-between steps are part of image processing and are relatively conventional and available to anyone with an imaging facility. The final step is the most difficult and has occupied a great deal of our time. Particle identification was not a major problem. We have found that there is a considerable reduction in computer effort if one transfers the data base from the normal red, green, blue model of color to the hue, intensity, and saturation model. Stereomatching to find particle pairs is in principle an easy task since the number of criteria to establish pairs is large. However, our preliminary results show that even though the stereomatching criteria used are more than adequate for the human vision to identify

stereopairs, the criteria did not resolve the stereomatching problem completely in our work. This is not a satisfactory situation, if one wants to automate the process. Tracking of the particles is easier if the film frame rate is high enough so that the particles only move a short distance between frames. In the scheme of matching and tracking, stereo and motion are considered and analyzed by different processes; however, human vision unifies the two processes in a way that each one overcomes the other's drawbacks. Separation of the two processes brought out their inherent shortcomings.

Separating motion from its counterpart process of stereomatching creates problems. Tracking is based only on information obtained from the previous frame. Therefore, an error in the previous frame will propagate rapidly and errors will accumulate. Due to inertia, we know that the motion of a particle cannot change abruptly if the filming rate change between two consecutive frames is small. Also, the velocity of a particle will not change dramatically under such conditions. Therefore, there are two basic characteristics of the motion process, path and velocity coherence, that are not being utilized in the analysis. These basic characteristics of the motion suggest that the position of a particle in the next frame should be predicted by using information of the particle's motion in the previous frames. This approach allows stereo and motion processes to be unified in a way that simulates the human operation to a greater extent. Our present efforts lie in this direction, which has proved to be very successful and has even allowed us to use a lower framing rate and still follow and match the particles adequately.

FLOWTRAN Tapes

_____ I am interested in preparing a FLOWTRAN tape.

_____ I am interested in obtaining a FLOWTRAN tape.

Please complete the following information. If you have two computers you would like to consider, duplicate this form and submit both completed forms and your preference.

1. Computer make and complete model number:

   _____

2. Operating System version:

   _____

3. FORTRAN compiler version:

   _____

4. Magnetic tape facility: No. of tracks _____ Drive speed in bits/inch _____

Name _____

Address _____

_____

_____

Please send this form to:

Prof. J. D. Seader
CACHE
3062 MEB
University of Utah
Salt Lake City, UT 84112

Please fill out this form only if you have not previously contacted Prof. Seader.

IBM PC Lessons in Fluid Mechanics

W. D. Seider
University of Pennsylvania

It is well recognized that computers are the key to the analysis of complex flows. Yet, of all the courses in the chemical engineering curriculum, computers are perhaps the least used in the fluid mechanics course. This may be related to the conceptual difficulties of introducing and applying momentum balances for all but the simplest fluids and geometries. Perhaps it is also because the most widely used textbooks do not utilize computers. And, perhaps it is because computer software has not been widely available to permit the solution of open-ended problems in the fluid mechanics course.

The IBM PC has been effective in providing interactive lessons, involving open-ended problems, in other courses and, consequently, it is our expectation that similar lessons will be effective in fluid mechanics. With this incentive, and armed with the UNISON Authoring System (a Courseware Applications, Inc., product), several members of the CACHE Curriculum Task Force are undertaking to prepare lessons in fluid mechanics that have two foci—macroscopic and microscopic analyses. These lessons, which are being developed during the spring and summer, will be class tested in the fall and should become available in 1990. A brief summary of each lesson follows.

Macroscopic Analysis

1. Pipe Flow—Deniz Karman, New Brunswick.

This module will treat both compressible and incompressible flow in cylindrical conduits, allowing for a variation of physical properties with temperature or solids loading. The conservation of mass, momentum, and energy will be demonstrated graphically for a conduit with typical elements (pumps, pipes, venturi meters, packed beds, valves, etc.) using hydraulic and energy grade lines. Open-ended problems will enable the student to explore various configurations and parameter variations. The software will permit the matching of pump and conduit characteristics graphically.

2. Piping and Pumping Networks—James O. Wilkes, Michigan.

A 600 statement FORTRAN program exists to analyze piping networks. The program, which currently executes in batch mode without computer graphics, accepts as input the topology of the piping network. This includes the location and elevation of each node, the length and diameter of pipe segments, the position of pumping stations, the location and capacity of storage tanks, and the flowrate and pressure demands for the particular liquid involved. The program solves the continuity and momentum balances in the steady-state and has been used successfully to analyze alternative designs for the expansion of a residential piping network. It has the potential to introduce students in the fluid mechanics course to the synthesis of alternative designs and to fault analysis (e.g., in locating blockages that develop).

The emphasis in preparing this lesson will be on the graphics interface. UNISON will be used, to the extent possible, to permit students to specify the network topology graphically and to display the results graphically.

Microscopic Analysis

1. Stress and Deformation: Couette and Squeezing Flow Between Parallel Plates—Kevin Ellwood and Tasos Papanastasiou, Michigan.

A computer problem is being developed for the study of Couette flow between parallel plates that creates shear stresses and shear deformation which can be displayed by tracing control volumes of material squares as they flow and deform, driven by a moving boundary or alternatively by a pressure difference. Squeezing flow between the same flows creates normal stresses and deformation, which can also be displayed by appropriate deformable control volumes. Lubricated squeezing flow which relaxes the no-slip boundary condition induces pure extensional deformation and

normal stress. Unlubricated squeezing flow gives rise to mixed deformation and stress.

2. Problems that Require Differential Balances—Mark J. McCready, Notre Dame.

These lessons will focus on the use of differential analysis. Flows will be in common geometries (tubes, annuli, and flat plates) caused by gravity, pressure gradients, and moving boundaries, and involving Newtonian and non-Newtonian fluids. The students will solve problems with either differential force balances or application of the Navier-Stokes equations. The lessons will be designed to lead the student to solutions, before enabling the student to explore the effect of parameter changes and alternate fluid models.

One lesson will focus on the wire-coating problem. Here, the flow will be due to the speed of drawing and external pressure gradients. Shear thinning fluids will be compared visually against shear thickening fluids. Other lessons will involve flow down a flat plate, rotational flow, and flow between flat parallel plates.

With experience, it is anticipated that the elements of computer lessons described by Scott Fogler will be incorporated.

3. Fluid Identification—Stacy Bike, Michigan.

In this lesson, the student will be given a fluid with unknown rheological characteristics and will identify the fluid type (i.e, Newtonian, shear-thinning, etc.) by performing "experimental" tests. The experimental tests will include a cone/plate rheometer and drainage of fluid from a tank. "Help" screens will aid the student in understanding the basis of each test, provide the limitations of the test, and describe the type of data to be obtained, together with an analysis of the data. The results will involve experimental error and an occasional result will be spurious. Given the constraints of cost and time, the students will decide which experiments to perform and how much data to obtain. The student's determination of fluid type will be graded upon completion of the analysis. This module may be created in the framework of a mystery.

For more information about the lessons, please contact the authors.

The Akron Family of Test Problems for Research and Education

J. M. Berty, Atul Govil, Chandrasekhar Krishnan, and Rajul Rastogi
The University of Akron

Now you can put a disk in your personal computer and conduct experiments, instead of charging catalyst into your reactor. You can also make design calculations or evaluate the performance of simulated commercial reactors.

All this is made possible by the Akron Family of Test Problems [1,2]. These are the UCKRON-I and VEKRON-I problems that represent the low-pressure methanol synthesis alone and low-pressure methanol synthesis in conjunction with the reverse water-gas shift reaction, respectively. These reactions are representative of many other catalytic or homogeneous reactions. The UCKRON problem was created in cooperation with Union Carbide Corporation, and the VEKRON problem is an extension of the former, developed by the Veszprem University of Chemical Engineering in Veszprem, Hungary.

The UCKRON problem was constructed by assuming a four-step mechanism whereas the VEKRON problem was formulated using a six-step reaction mechanism. The kinetic expressions were derived assuming no rate limiting steps and are thermodynamically consistent. Therefore, the rate expressions can be considered as representing the real "truth" for educational purposes. You can incorporate these rate expressions in your industrial reactor simulation programs or simulate CSTR experiments. You can also develop your own kinetic models for methanol synthesis using the test problems as follows: To the rate data obtained from these "true" rate expressions, add random errors, make statistical evaluations, and develop various rate models. The resulting models can be compared to the original "true" expression, and the effects of simplification involved in modeling can be shown by comparing, for example, the reactor performance calculated on the basis of the "true" kinetics with that calculated from the kinetic models that approximate the "truth."

More information on this subject is available in the paper "The State of Kinetic Model Development," published in the February 1988 issue of *Chemical Engineering Progress (pp. 61-67)*. This paper includes a list of recommended studies [1]. The Akron Family of Test Problems was used as a basis for all papers presented at the two AIChE-sponsored "International Workshops on Kinetic Model Development." The papers will be published under the same title in the February 1989 Special Edition of *Chemical Engineering Communications* [2].

In a small way, the test problems are a step towards fulfilling the idea proposed by the Committee on Chemical Engineering Frontiers that "It is now realistic to imagine mathematical models of fundamental phenomena beginning to replace laboratory and field experiments" [3].

For copies of UCKRON-I and VEKRON-I programs, send a 5-1/4 inch diskette with a self-addressed and stamped diskette mailer to:

> Auburn Science Center 160
> Department of Chemical Engineering
> The University of Akron
> Akron, OH 44325-3906.

This diskette will contain FORTRAN-77 programs (both source and PC executable versions) to calculate the reaction rate for the UCKRON problem by specifying the temperature and partial pressures for hydrogen, carbon monoxide, and methanol. The VEKRON problem requires, in addition to all the inputs of the UCKRON problem, the partial pressures of carbon dioxide and water.

References

1. Berty, J. M., "The State of Kinetic Model Developments," *Chemical EngineeringProgress*, pp. 61-67 (February, 1988).

2. "International Workshops in Kinetic Model Development," Special Edition of Chemical *Engineering Communications*, (February, 1989).

3. Committee on Chemical Engineering Frontiers, *Research Needs and Opportunities, Board on Chemical Sciences and Technology, Commission on Physical Sciences, Mathematics, and Resources, National Research Council*, Ch. 8, p. 5, National Academic Press (1988).

DIFFPACK

W. E. Schiesser
Lehigh University

The Numerical Method of Lines (NMOL) is a comprehensive procedure for the computer integration of systems of ordinary and partial differential equations (ODE/PDEs) which has been gaining wide acceptance among scientists and engineers. In order to facilitate the use of NMOL, Lehigh University has developed a set of problems, DIFFPACK, illustrating the application of this procedure to the major classes of ODE/PDE problems. The problems are coded in transportable Fortran 77 and run under DSS/2, a general-purpose NMOL code.

The classes of problems included in DIFFPACK are:

(1) Initial Value ODEs

A system of two linear ODEs with a stiffness ratio of 10^6 and a system of nonlinear ODEs of arbitrary order and stiffness.

(2) Boundary Value ODEs

The solution of a nonlinear boundary value ODE is illustrated indirectly through the equilibrium solution of a one-dimensional PDE (see below).

(3) One-Dimensional PDEs

The one-dimensional Burgers' equation is the basis for this subset of one-dimensional PDE problems:

$$u_t = -uu_x + Du_{xx} \qquad (1)$$

(3.1) Hyperbolic PDEs—illustrated through the term $-uu_x$

(3.2) Parabolic PDEs—illustrated through the term u_{xx}

(3.3) Elliptic PDEs—illustrated through $u_t = 0$; this can also be considered a boundary value ODE.

The numerical solutions for these three cases are confirmed with an exact solution to equation (1):

$$u(x,t) = 1/(1 + e^{x/(2D) - t/(4D)}) \qquad (2)$$

(4) Two and Three-Dimensional PDEs

The two and three-dimensional (2-D, 3-D) Burgers' equations are the basis for this subset of 2-D and 3-D PDE problems:

$$u_t = -uu_x + Du_{xx} - uu_y + Du_{yy} \qquad (3)$$

$$u_t = -uu_x + Du_{xx} - uu_y + Du_{yy} - uu_z + Du_{zz} \qquad (4)$$

(4.1) Hyperbolic PDEs—illustrated through the terms $-uu_x$, $-uu_y$ (2-D) and $-uu_z$ (3-D)

(4.2) Parabolic PDEs—illustrated through the terms u_{xx}, u_{yy} (2-D) and u_{zz} (3-D)

(4.3) Elliptic PDEs—illustrated through $u_t = 0$

The numerical solutions for these three cases are confirmed with exact solutions to equations (3) and (4):

$$u(x,y,t) = 1/(1 + e^{x/(2D) + y/(2D) - 2t/(4D)}) \qquad (5)$$

$$u(x,y,z,t) = 1/(1 + e^{x/(2D) + y/(2D) + z/(2D) - 3t/(4D)}) \qquad (6)$$

The coding for all of these problems contains detailed comments defining the problems and explaining the steps of the NMOL solutions.

DIFFPACK is available in the following formats:

1. DIFFPACK only (previous purchase of DSS/2, Release 4 is assumed); single and double precision Fortran 77 source code and output from the problems:

   (1.1) Nine track, unlabelled, 1600 BPI, ASCII (or EBCDIC) tape.

   (1.2) DOS-formatted, 1.2 mb diskette.

2. DIFFPACK plus DSS/2, Release 4; single and double precision Fortran 77 source code, set of seven DSS/2 manuals, output from ten DSS/2 test problems and output from the DIFFPACK problems. Software is on nine-track tape.

Shipment is by Priority Mail in the U.S. and Canada, and by international air mail to all other countries. Delivery is two weeks ARO.

Purchase of either set of software may be initiated by a purchase order directed to:

Dr. W.E. Schiesser
Mountaintop—Bldg. A (111)
Room D307
Lehigh University
Bethlehem, PA 18015 USA

Prices are available on request.

Payment is net 30 by a check payable to Lehigh University in response to a Lehigh invoice in U.S. dollars through a U.S. bank.

Inquiries about this software, or any of the other numerical software available from Lehigh, may be directed to W.E. Schiesser at the address above or on (215) 758-4264.

Matrix Oriented Computation Using *Matlab*™

Jeffrey C. Kantor
University of Notre Dame

Matlab is a tool for interactive numerical computation. It contains as built-in functions essentially all of the numerical linear algebra algorithms in LINPACK and EISPACK. Coupled with a programmable interpreter and good scientific graphics capability, Matlab can be used for algorithm development in many areas of Chemical Engineering.

To demonstrate some of its functionality, I've included in this article several examples where Matlab has proven useful in my own teaching and research activities. These examples are not comprehensive since they neither fully exploit all of the features of Matlab or do they show all of our applications. The examples were chosen only because they seemed to be relatively straightforward and self-contained illustrations of how Matlab can be used.

Some Background

Matlab was originally conceived by Cleve Moler just over a decade ago while he was teaching numerical methods at the University of New Mexico. He found it frustrating to simultaneously teach numerical methods and the programming tricks it takes to implement them. The effort required to write FORTRAN code can simply overwhelm a student and not leave much time left over for doing applications. So to address the problem, Cleve Moler wrote a simple interpreter in portable FORTRAN for a high-level matrix oriented language. The interpreter was based on one given by N. Wirth for a model language called PL/0 (Wirth, 1976). The numerical algorithms were naturally based on the recently completed Linpack and Eispack projects to which Cleve Moler had made substantial contributions. This primitive Matlab[1] interpreter was evidently quite successful and ported to a number of machines during the late 1970's and early 1980's, undergoing minor revisions in the process.

Several companies subsequently adopted Matlab as a development platform for commercial control design and analysis software. Systems Control Technology produced a package called Control-C, and at about the same time, Matrix-X was developed by Integrated Systems, Inc. Both companies found many shortcomings in the original Matlab interpreter including workspace constraints, lack of function definitions, and overall performance. The Matlab interpreter was largely rewritten at each of these companies to support their products.

A few of the professional staff from these companies joined together to form a new company called the MathWorks, Inc. There they produced an entirely new version of Matlab written in C for portability and efficiency. The interpreter was greatly enhanced to include an ability for the user to program Matlab functions. They also developed an integrated facility for producing a basic set of publication quality scientific graphs. The MathWorks currently markets this version of Matlab for a variety of hardware platforms, the details are given at the end of this article.

Beyond the basic interpreter, there are several 'toolboxes' intended for specific application areas. A 'toolbox' is typically a collection of functions and scripts that implement specialized numerical algorithms. These generally are not finished applications in the sense of a well-developed user interface with a lot menus and the like, but are rather integrated collections of algorithms that you either can use directly or build into your own scripts. It is sort of like using a FORTRAN subroutine library, but with the advantage of being able to directly execute the routines in the interactive Matlab environment. The MathWorks distributes a Signal Processing Toolbox with Matlab, and markets several others including a Control Design Toolbox, Robust

[1] Matrix Laboratory.

Control Toolbox, System Identification Toolbox, a Chemometrics Toolbox. There are also toolboxes commercially available from third parties, in addition to a number that University researchers may have put together for their own purposes.

Now for the confusing part. There is a 'public domain' IBM PC version of Matlab. In addition, several universities sell very low cost versions of Matlab available for the Macintosh and IBM PC. These are based on Moler's original FORTRAN code, sometimes with enhanced graphics and macro writing facilities. A person should be careful with these since they are not of the same calibre as the MathWorks and simply don't include the tools necessary for doing real work. Nor will the toolboxes cited above work with these versions. A corollary of this advice is to not let an exposure to these other versions color your view of Matlab.

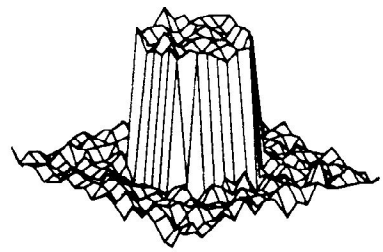
What is Matlab?

In some ways, the Matlab interpreter vaguely resembles a cross between BASIC and APL in the sense that it is programmable and endowed with a rich set of operators for matrix manipulations. The key distinction is that Matlab incorporates well-developed and reliable algorithms for numerical linear algebra. Moreover, the built-in graphics capability is often entirely sufficient for presenting results in final published form. (The graphics in this article, for example, were pasted in directly from Matlab).

Let me give an example of how these capabilities can be used for day-to-day 'scratchpad' kind of calculations that pop up. A few days ago a colleague of mine walked into my office with an idea for processing video images to enhance the edges of discs that appear in the picture. He acquires these images in his experiments on concentrated suspensions of non-colloidal particles. He started off by saying (roughly) "Suppose you have a noisy image of a disc ..." at which point I stopped him, turned on my computer, and typed the following commands in Matlab[1]

```
x = -1:.1:1;                % X mesh
y = -1:.1:1;                % Y mesh
[xx,yy] = meshdom(x,y);% 2D mesh

z=sqrt(xx.^2+yy.^2)<0.5; % make disk
rand('normal');          % white noise
z = z + 0.05*rand(z);    % add to disk
mesh(z);                 % 3D plot
xlabel('Noisy Disk');    % add title
```

which produced the following image



Noisy Disk

This code segment demonstrates several of the key features of Matlab. First of all, the variables `x, y, z` represent vectors and matrices. Matrices are an elementary data type within Matlab. Because matrices can be manipulated directly as single objects, much of tedium of writing loops to do element by element calculations is removed, along with the need for a lot of extraneous indexing. In the sixth line, for example, a matrix is constructed with the same dimensions as `z` consisting of normally distributed random numbers (`rand(z)`), multiplied by `0.05`, and the result added to `z`. The third line demonstrates how Matlab functions can return multiple results, which in this case are two matrices `xx` and `yy`.

Duly impressed, my colleague went on at the blackboard to describe a simple algorithm requiring that the image be processed by a pair of 2D convolutions.[2] Since this might be done more than once to different data sets, it seemed sensible to encapsulate the algorithm as a Matlab function.

```
function [y] = sobel(z)

% SOBEL
%  Do edge detection on a 2D array

s = [1 2 1; 0 0 0; -1 -2 -1];
h = conv2(z,s);      % 2D convolution
```

[1] The text appearing after the % symbol constitute Matlab comments. These have been included here to add a bit of explanation.

[2] This idea actually came fairly directly from the the *Mathworks Newsletter* of December, 1988.

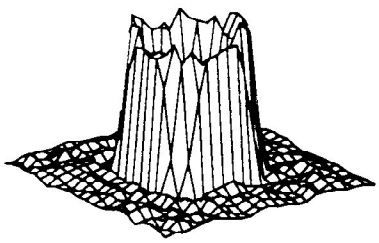
```
v = conv2(z,s');     % 2D convolution
y = sqrt(h.^2 + v.^2);
```

A function is prepared as a separate text file that is subsequently read by the Matlab interpreter when its name is encountered in a command line. A user written function behaves as the same way as any built-in function. In this example, a function named `sobel` is defined which takes a single input argument `z`, then utilizes a built-in Matlab function `conv2` to construct two 2D convolutions with a matrix, `s`, and its transpose, `s'`. The function output, `y`, is found by taking the harmonic mean of the two convolutions.

The edge detection function was used in the following commands

```
zf = sobel(z);
mesh(zf);
xlabel('Edge Filtered Disk');
```

to produced an edge enhanced picture



Edge Filtered Disk

In general, functions can have multiple-input and multiple-output arguments. Just as in FORTRAN, any variables used in writing a function are treated as local and will not be confused with other variables of the same name used in other functions or the command environment.

So during the course of a half-hour conversation, my colleague was able to (watch me) construct and test an edge detection algorithm. It is this ability to quickly prototype and test algorithms using a rich base of numerical tools that makes Matlab a valuable computational tool.

Using Matlab in the Classroom

I have used Matlab in teaching a graduate course on Process Control (Fall, 1987), the linear algebra portion of a course covering Mathematical Methods for first-year graduate students (Fall, 1988), and currently for a Junior-level course on Computer Methods for Chemical Engineers (Spring, 1989). Matlab seems to provide an appropriate software base for each of these courses.

In the case of teaching Advanced Process Control, the main goal in using Matlab was to provide the student with experience in doing time-series analysis, model identification, control design, and simulation. There are competing software packages that could also be used for these purposes, among them Program-CC, but none seemed to offer any significant advantage over Matlab for linear analysis. Besides, my teaching assistant had already had some experience with Matlab, and it was already installed on several Sun workstations in the Department. Overall, the teaching experience was a very good one. By the end of course the students demonstrated a real facility with Matlab, the Control Design Toolbox, and the System Identification Toolbox. Later in the article there is an example that came from homework problem assigned in the course.

For the undergraduate Computer Methods course, there were additional considerations that came up when considering a choice of software tools. Among them was the choice between using Matlab or a package of FORTRAN subroutines such as given in Press, *et al.* (1986). On the one hand, FORTRAN remains as the principle programming language for numerically intensive engineering applications, therefore a facility with FORTRAN is highly desirable. Moreover, our students all take a required Freshman Engineering course that teaches the elements of FORTRAN.

On the other hand, it is significantly faster to write and test small codes using the high-level Matlab interpreter. The students also indicated a strong preference for microcomputer based software tools which could be used on various workstation clusters about campus rather than be tied to a single minicomputer located in the Engineering College.

On balance, I felt that a more productive environment would allow the course to survey more topics with more emphasis on applications, so I chose to use Matlab. The semester is now half over and the students seem to agree with me. It is still too early to tell if the students will be able to transfer their new computational skills to other courses. In particular, it will be interesting to see how these students will choose to do calculations that will be required in their Senior Design courses.

Recent textbooks have appeared which incorporate various amounts of Matlab into the text and exercises. The third edition of the classic linear algebra text by Noble (1988) contains is number of Matlab exercises and examples. Another linear algebra textbook by Hill is basically centered on Matlab, with chapters regarding programming technique. It is so complete that it could serve as a low-cost Matlab manual for students.

Lennert Ljung's book on system identification (Ljung, 1987) is closely coupled to the System Identification Toolbox. The toolbox, in fact, was written by Ljung, and the text provides excellent technical documentation.

The following two sections present two examples of incorporating Matlab into classroom activities.

Classroom Example - Linear Programming

Three years ago our Department introduced a new required course for our undergraduate majors entitled *Computer Methods for Chemical Engineers*. This course is normally taken by Spring semester Juniors after having completed the normal Mathematics sequence, and before commencing the two-semester Senior design sequence. The course covers elements of numerical methods with application to problems in chemical engineering.

Linear programming is discussed in some detail in the course because it is one of those skills that an engineer can transfer to a wide variety of problem areas. A key teaching goal is for the student to be able to recognize a problem as a linear program, and then to formulate the requisite objective and constraints.

I prefer to use the Active Set method as outlined by Fletcher (1987) to teach the principles behind linear programming. It seems to leave the student with a more intuitive understanding of the role of constraints and their sensitivities than does the Simplex method. If the students can understand the relatively simple strategy to solving linear program, it is then much easier to motivate and teach the numerical tricks it takes to implement an efficient algorithm.

The linear programming problem is formulated as minimizing the linear objective

$$\min_x z = c^T x$$

where x is a n-vector, subject to m linear constraints

$$a_i x \geq b_i \quad i = 1,2,\ldots,m$$

where $n \leq m$. If positivity constraints are present, then these are explicitly included in the constraint list. It is easy to show that if the feasible region is bounded, then optimum will always be found at a vertex defined by the intersection of n active constraints.

The basic algorithm is, firstly, to find any active set of n constraints forming a feasible vertex, then to move systematically from one vertex to another so as to reduce the value of the objective function at each step. Each step of the algorithm is defined by just two rules. The first rule identifies a constraint to throw out of the active constraint set in order to decrease the objective. The second rule determines which constraint to add to the active set to establish a new feasible vertex.

Let A be the set of active constraints that determine a feasible vertex. The vertex is given by solving a set linear equations to give

$$x = A_A^{-1} b_A$$

where A_A and b_A are constructed from the coefficients of the active constraints. Now suppose the right hand side of each active constraint is altered by a small positive amount ε_i. The vertex then shifts from x to x', where

$$x' = A_A^{-1} b_A + A_A^{-1} \varepsilon$$

Substituting x' into the objective function yields

$$z = c^T A_A^{-1} b_A + c^T A_A^{-1} \varepsilon$$

The second term shows the change in the objective function due to independent perturbations in the active constraint set. Thus the elements of the row vector

$$\lambda = c^T A_A^{-1}$$

play the role of 'sensitivity coefficients' revealing how the objective function responds to feasible perturbations in the active constraint set. If any of the elements of λ is negative, then the objective function can be reduced by removing that constraint from the active set. Just as in the Simplex method, we choose to remove the

constraint corresponding to the most negative element of λ.[1]

Let λ_p be the most negative element of λ. Then the effect of removing the p^{th} active constraint is given by

$$x' = x + \varepsilon_p s_p$$

where s_p is the p^{th} column of A_A^{-1}. How large can ε_p be before some other constraint becomes active? This can be computed explicitly as

$$\varepsilon_p = \min_{\substack{i \notin A \\ a_i s_p < 0}} \frac{b_i - a_i x}{a_i s_p}$$

The constraint which realizes the minimum ε_p is exactly the one to be added to the active constraint set. Having done that, the procedure repeats itself until no further improvement in the objective is possible, i.e., until all of the sensitivity coefficients are non-negative.

This basic algorithm cleanly translates to the following Matlab function. The function `lp` takes four arguments specifying the coefficients on the left and right hand sides of the constraints, coefficients of the objective function, and an initial feasible constraint set. The function returns the optimal value of the objective function, the optimal solution for the decision variables, the value of the sensitivity coefficients, and the final active constraint set.

```
function[z,x,lamb,activ]=lp(a,b,c,fea
s)

% Initialization

[m,n] = size(a);
activ = feas(:);

% Compute Initial Vertex

ainv = inv(a(activ,:));
x = ainv*b(activ,:);
lamb = c*ainv;

while any(lamb < 0),

  % Find which constraint to drop, p

  [tmp,p] = min(lamb);
  sp = ainv(:,p);

  % Find which constraint to add, q

  alpha = Inf;
  q = 0;
  for i=1:m,
    if ~any(i==activ),
      den = a(i,:)*sp;
      if den < 0,
        tmp = (b(i)-a(i,:)*x)/den;
        if tmp < alpha,
          alpha = tmp;
          q = i;
        end
      end
    end
  end

  % Recompute x, lamb, and z

  activ(p) = q;
  ainv = inv(a(activ,:));
  x = ainv*b(activ,:);
  lamb = c*ainv;

end

z = c*x;  % Compute objective
function
```

This example uses several of the Matlab control structures to simplify the coding process. The construction

```
while any(lamb < 0),
   [...]
end
```

controls the main iteration over vertices of the feasible region. The iteration continues as long as any element of the vector `lamb` is less than zero. Nested within this loop is an iteration

```
  for i=1:m,
     [...]
  end
```

which specifies a conventional indexed iteration loop where `i` successively takes values between `1` and `m`. Within this loop are several nested conditional statements such as

[1] Of course, this would be a good point to return to regarding the problems of cycling and Brand's rule (Fletcher, 1987).

```
if ~any(i==activ),
  [...]
end
```

In this case, the conditional code is executed if 'not any' of the elements of the vector `activ` are equal to `i`. The practice of indenting nested control structures graphically reveals program flow and is strongly urged on the students.

This function is a zeroth order cut at a practical algorithm for linear programming, it will work for small problems but will be inefficient and error prone when applied to larger problems. As exercises, the students are asked to correct several of the glaring deficiencies. Foremost is to avoid the repeated inversions of the active constraint matrix with a more efficient procedure using rank-one updates (i.e., the Sherman-Morrison formula). Having done this, the algorithm is then identical to the usual revised simplex method as discussed in most textbooks. Other exercises in algorithm development could include writing a code to identify an initial feasible constraint set, or to modify the algorithm to handle equality constraints.

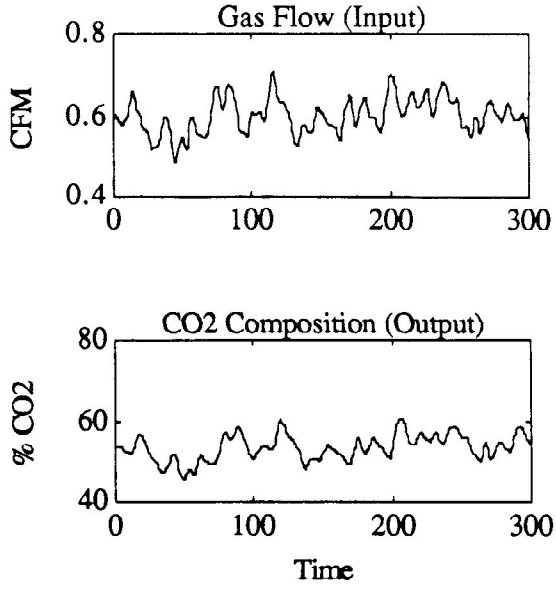
Classroom Example - Advanced Control

The next example illustrates the use of several toolboxes to do model identification and a simple control design. Students taking a graduate course in Advanced Process Control during Fall, 1987, were assigned a homework project in which they were to analyze input-output data for a small gas furnace. They were to obtain first a transfer function model, then use the model to design a PID, minimum variance, and optimal LQG controllers. The three controllers were to be evaluated by simulation. The students were given one week to complete the assignment.

The gas furnace data taken from Appendix B of Box and Jenkins (1976) consisted of 300 pairs of input-output measurements {u(k), y(k)} obtained at 9 second intervals. The manipulated input is a gas flowrate, and the measured output is the percentage of CO_2 in the stack gas. These data were given to the students as a Matlab file called `GasFurnaceData`. The file can be read and plotted using the following commands

```
%     Read data record

GasFurnaceData;
udata = u;
ydata = y;

%     Plot input-output data
subplot(211);  % Specifies upper plot
plot(udata);
title('Gas Flow (Input)');
ylabel('CFM');

subplot(212);  % Specifies lower plot
plot(ydata);
title('CO2 Composition (Output)');
ylabel('% CO2');
xlabel('Time');
```

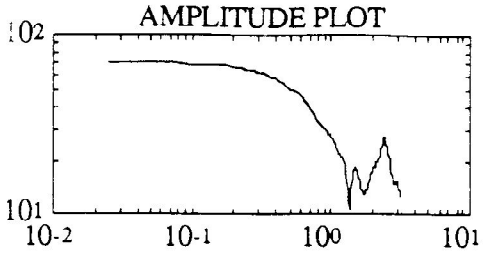
to produce the following plots:

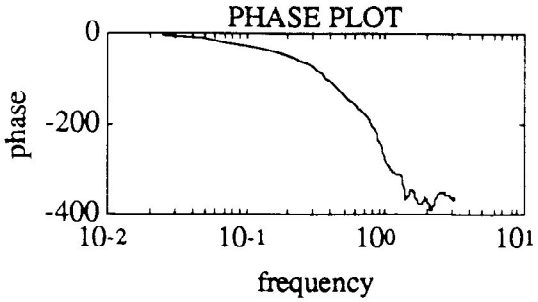


The first task for students was to identify a discrete-time transfer function model for the gas furnace. A non-parametric spectral analysis provides a starting point for estimating model order. This is done with the following commands:

```
y = detrend(ydata);
u = detrend(udata);
z = [y u];
g0 = spa(z);        % System_ID toolbox
bodeplot(g0);       % System_ID toolbox
```

The function `detrend` (from the Signal Processing Toolbox) is used to remove means and linear trends from the input and output data series. Then `spa` (from the System Identification Toolbox) is applied to construct a transfer function estimate that is stored as `g0`. The transfer function is displayed using `bodeplot`.





There are a number of possible models that could be used to describe this data. Of these, an ARMAX model in the form

$$y(t) = \frac{B(q)}{A(q)}u(t - n_k) + \frac{C(q)}{A(q)}e(t)$$

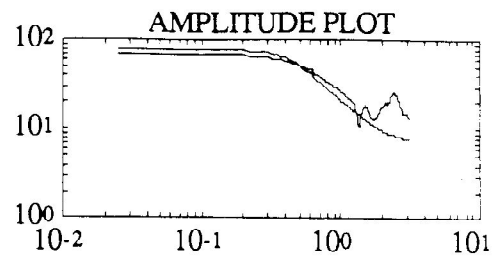
or, explicitly, as

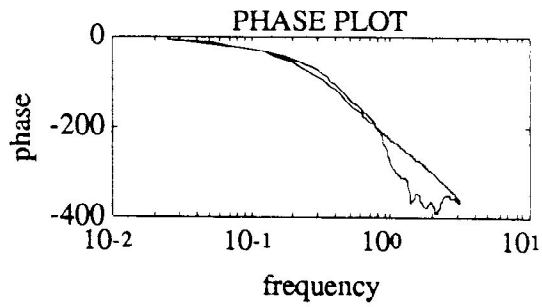
$$y(t) = \frac{b_1 q^{-1} + \ldots + b_{n_b} q^{-n_b}}{1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}} u(t - n_k) + \frac{c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}}{1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}} e(t)$$

does an adequate job (q^{-1} is the backward shift operator). The following commands use functions in the System Identification Toolbox to fit an ARMAX model for the case $n_a = n_b = n_c = 2$, $n_k = 1$. The fitted transfer function is then evaluated and Bode plot is displayed to compare the fitted transfer function to the previous non-parametric estimate.

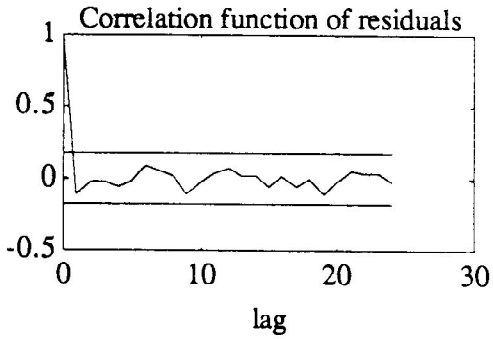
```
th = armax(z,[2 2 2 1]);
g = trf(th);
bodeplot([g g0]);
```

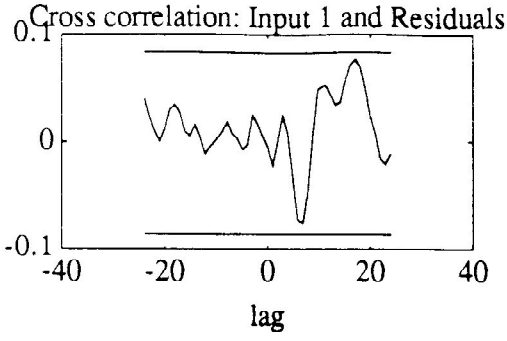
The resulting Bode plots are given below:





The Bode plot shows a reasonable fit of the data using a second order model. 'Goodness of fit' can also be explored by computing an estimated autocorrelation function for the residuals, and an estimated cross correlation between the input. This is done here and the results are plotted using the command `e = resid(z,th)` to produce





The horizontal lines in these plots mark the 95% confidence intervals for the null hypothesis. These plots indicate that there is little significant correlation left in the residuals so there is no statistical justification for employing higher order models. (Attempting to fit a first-order model to this data provides an example where statistically significant correlations do remain in the residuals.) The fitted model coefficients are displayed as follows:

```
»present(th)

This matrix was created by the
command      ARMAX  on 2/28 1989 at
10:47
Loss fcn: 0.09217   Akaike's FPE:
      0.09593 Sampling interval 1
The polynomial coefficients and their
      standard deviations are

B =
         0   -6.3133   16.9243
         0    1.9007    2.3403

A =
    1.0000   -1.3899    0.5299
         0    0.0516    0.0460

C =
    1.0000    0.1385    0.1307
         0    0.0856    0.0659
```

At this point in the exercise, the student has developed a transfer function model for the gas furnace that can be used for designing simple control systems. Omitting the details, an optimal LQG controller can be designed to minimize the loss function

$$J_u = E\,[y^2(k) + \rho u^2(k)]$$

by the computational method outlined in Chapter 12 of Astrom and Wittenmark (1984). The necessary calculations are encapsulated in the function `dlqg` given below. This function makes use of others defined in the Control Systems Toolbox. These are dlqr, which computes a solution to the algebraic discrete time Ricatti equation, and ss2tf, which converts a state-space model representation to a transfer function description.

```
function [s,r]=dlqg(th,rho)

%DLQG
%  [r,s] = DLQG(theta,rho) computes
%  the LQ optimal controller to
%  minimize the objective function
%
%              2            2
%          E[y (k) + rho*u (k)]
%
%  The resulting controller is given
%  in transfer function form
%
%                      S(q)
%            u(k) = - ---- y(k)
%                      R(q)
%
%  The plant model is given by theta
%  in the standard form of the System
%  Identification Toolbox.

% Ref:Chapter 12, Astrom & Wittenmark

% J.C. Kantor, 3 December 1987

  [a,b,c,d,f]=polyform(th);
  a=conv(a,f);
  na = length(a)-1;
  nb = length(b)-1;
  nc = length(c)-1;
  n = max([na,nb,nc]);
  A = [zeros(n,1),[eye(n-1);..
        zeros(1,n-1)]];
  A(1:na,1) = -a(2:na+1)';
  B = zeros(n,1);
  B(1:nb,1) = b(2:nb+1)';
  K = zeros(n,1);
  K(1:nc,1) = c(2:nc+1)';
  K = K + A(:,1);
  C = [1,zeros(1,n-1)];

  L = real(dlqr(A,B,C'*C,rho));

  [s,r] = ss2tf(A-K*C-B*L,K,L,[0],1);
```

Letting $\rho = 10^{-5}$ gives an approximation to minimum variance control. The resulting controller is given by $u(t) = -G_c(q)y(t)$ where

$$G_c = \frac{S(q)}{R(q)} = \frac{0.0762\,q^{-1} - 0.0512\,q^{-2}}{1 + 1.1555q^{-1} + 1.6364\,q^{-2}}$$

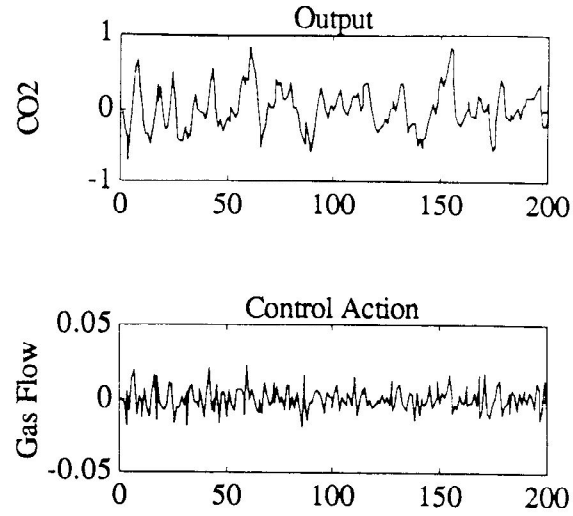
Finally, the student can compute the simulated response of the closed-loop gas furnace control system. The closed-loop transfer function between the output and exogenous disturbances $e(t)$ is given by

$$y(t) = \frac{C(q)R(q)}{A(q)R(q) + B(q)S(q)}e(t)$$

The following sequence of commands computes the products of polynomials using the Matlab convolution operator `conv`, does a simulation of the closed-loop plant models, and displays the results.

```
% Compute control and closed-loop
% transfer fuctions


[s,r] = dlqg(th,0.00001);
[a,b,c]=polyform(th);
p = conv(a,r) + conv(b,s);
qy = conv(c,r);
qu = conv(c,s);

% Construct a white noise input

rand('normal');
w = 0.1*rand(200,1);

% Output simulation

subplot(211);
plot(dlsim(qy,p,w));
title('Output');
ylabel('CO2');

% Control simulation

subplot(212);
plot(dlsim(qu,p,w));
title('Control Action');
ylabel('Gas Flow');
```

The simulated performance of the closed-loop regulator results in a 20.5% reduction in the variance of the CO_2 stack gas composition compared to the case of no control.



Many additional aspects of the problem can be readily treated using simple Matlab procedures.

Summary Remarks (Why Matlab Can't be Used for Everything?)

In spite of its many useful features, Matlab is not an appropriate tool for all applications. While it is difficult to draw precise boundaries, there are some general guidelines.

- *Matlab is useful when your problems are 'vectorizable'.*

  Matlab exhibits excellent floating point performance when using its matrix oriented primitive operations. However, because it is an interpreted (not compiled) language, it suffers some performance degradation on scalar and non-numeric operations. Some algorithms, such as for integrating ordinary differential equations, can be quite slow in Matlab for this reason.

- *Matlab is useful for prototyping algorithms.*

  Matlab is a high-level language with a large number of primitives so that even complex algorithms can be written in a minimal number of lines. The interpreter provides a convenient mechanism for debugging numerical algorithms. For example, simply by deleting the semicolon at the end of a line, the intermediate results of any computation are printed. There are also facilities for introducing keyboard interrupts and monitoring intermediate values.

• *Matlab is useful when you need results fast.*

In addition to the points given above, the available toolboxes and graphics facilities are often sufficient for solving problems from start to finish, including the production of publication graphics.

• *Matlab does not replace either FORTRAN or specialized application software.*

Matlab is not a replacement for a FORTRAN compiler and a good package of scientific subroutines. It not suited to truly large scale computation, nor can it be used effectively in a batch mode. Linear programming provides an example of the tradeoffs. Straightforward Matlab LP codes might be useful for problems with, say, up to a few hundred constraints. This is no match for commercial that can handle many thousands of constraints.

• *Matlab is not very effective for non-numerical algorithms.*

Matlab treats essentially all information as matrices of real or complex floating point numbers. The simple facilities for handling textual data in Matlab are inadequate for anything beyond manipulating titles and labels. It would be a mistake to use Matlab to do data base programming, for example, or for writing compilers.

Where to Obtain Matlab

Academic institutions can purchase Matlab directly from the MathWorks, Inc. Their address is

The MathWorks, Inc.
21 Eliot Street
South Natick, MA 01760

Phone: (508) 653-1415
Fax: (508) 653-2997
E-mail: na.mathworks@na-net.stanford.edu

The MathWorks has special licensing provisions for classroom and educational use. For commercial uses, Matlab is also distributed by

MGA, Inc.
73 Junction Square Dr.
Concord, MA 01742

Phone: (508) 369-5115

Versions of Matlab are available for IBM PC, AT, and 80386 platforms, including Weitek support. Also for the Apple Macintosh (with and without support for the 68881), Sun and Apollo workstations, DEC Vax, Gould, and Ardent machines. The Ardent version has facilities for 3D solids rendering.

References

Astrom, Karl J., and Bjorn Wittenmark (1984). *Computer Controlled Systems.* Prentice-Hall, Englewood Cliffs, NJ.

Box, George E. P., and Gwilym M. Jenkins (1976). *Time Series Analysis: Forecasting and Control.* Rev. Edition. Holden-Day, San Francisco.

Chapra, Steven C., and Raymond P. Canale (1988). *Numerical Methods for Engineers, Second Edition.* McGraw-Hill, New York.

Fletcher, R. (1987). *Practical Methods of Optimization, Second Edition.* John Wiley & Sons, New York.

Forsythe, George E., Michael A. Malcolm, and Cleve B. Moler (1977). *Computer Methods for Mathematical Computations.* Prentice-Hall, Englewood Cliffs, NJ.

Hill, David R. (1988). *Experiments in Computational Matrix Algebra.* Random House, New York.

Kahaner, David, Cleve Moler, and Stephen Nash (1989). *Numerical Methods and Software.* Prentice-Hall, Englewood Cliffs, NJ.

Ljung, Lennart (1987). *System Identification: Theory for the User.* Prentice-Hall, Englewood Cliffs, NJ.

Noble, Ben, and James W. Daniel (1988). *Applied Linear Algebra, Third Edition.* Prentice-Hall, Englewood Cliffs, NJ.

Press, William H., Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling (1986). *Numerical Recipes - The Art of Scientific Computing.* Cambridge University Press, Cambridge.

Wirth, Nicklaus (1976). *Algorithms + Data Structures = Programs.* Prentice-Hall, Englewood Cliffs, NJ.

Microcomputer Chemical Engineering Programs

Edited by Bruce A. Finlayson
University of Washington

Have you wondered what microcomputer programs are being used in other chemical engineering curricula? This column provides a mechanism for University professors to let others know about the programs they've developed and are willing to share on some basis.

The program should be described by a 250 word description, machine requirements, and ordering information. These programs should be ready to be shipped out the door, and should have been tested by students at more than one University. It would be helpful if the specific chemical engineering course were identified in which the program is useful. The programs will not be reviewed by Prof. Finlayson, nor will they be certified by CACHE.

In order to edit this column efficiently, submissions must be made to Finlayson via BITNET, using userid 27432 and node MAX. He will acknowledge receipt of the submission via BITNET, and will send the edited column to the CACHE office via BITNET. Letters will not be accepted. This requirement has two goals: to reduce the need for secretarial typing and to encourage academic chemical engineers to use electronic mail. You can find out how to use BITNET at your local installation—it is amazingly simple. You can then share the protocols with your colleagues for other uses.

This column can only be successful if you contribute. Let us hear from you!

A Rigorous Multicomponent Multistage Steady-State Distillation Rating Program

by E.C. Roche, Jr.

Program DISTIL performs the rating evaluation of an existing multicomponent multistage distillation column. The program capabilities are 100 stages in the column (plus the condenser and the reboiler) and 9 components. The program can handle 6 feeds and 10 (vapor-liquid) side-streams/intermediate heat exchangers. Each stage in the column is considered to be a theoretical contact stage. The reboiler is treated as a kettle, while the condenser/reflux separator is considered a fractional equilibrium stage depending on the distillate product vapor/liquid split. The rating program assumes that the column geometry has been established, the operating pressure level is known, and that the feeds, side-stream withdrawals, and the intermediate heat exchangers are defined.

The available design variables are: (1) the distillate flow rate (vapor and liquid), (2) the reflux ratio or its equivalent (the boil-up ratio, condenser duty, and the reboiler duty), (3) the number of stages in the column, (4) the feed stage, and (5) the degree of feed vaporization. The solution of the nonlinear set of algebraic equations yields the stage compositions, flow rates and temperature. The solution method is the combination of the linear algebra method and the Newton-Raphson method.

The componential equilibrium and thermodynamic data is provided to the program in the form of tabular data. This form of data entry provides the student in a mass transfer course the opportunity to concentrate on what a rating program can accomplish.

The program was developed for a mass transfer course and has been in continued use for over fifteen years. The program is executable on IBM PCs using PC-DOS 2.0 or higher. The current version of the program requires an 8087 chip and 230K of available memory for the executable load module. The program interactively requests the input data file, and then generates a standard output file plus two additional files: a convergence trace file (also displayed on the screen) and a profile file of the converged case. The data in the profile file can be externally graphed to observe the column's operating characteristics.

A disk containing the program, documentation and illustrative examples may be purchased for $20, payable to Edward C. Roche, Jr. Send requests to E.C. Roche, Jr., 81 Brooklawn Drive, Morris Plains, NJ 07950.

RESIM: A Reactor Design Teaching Tool

by B.W. Wojciechowski

The Microsoft Fortran program RESIM is designed for the investigation of the cost and performance of isothermal and adiabatic reactors. It contains data files which at present allow the design of reactors using any of four methanol synthesis catalysts, one SO2 oxydation catalyst or one ammonia synthesis catalyst. Other reactions and catalysts can be readily added as required.

The program has a wide range of design options which are entered in response to on-screen prompts. The numerical output is written to disk while graphical output can be sent to the screen or to an HP plotter. Pressure drop, Reynolds number in each bed, gas fugacities, heat exchange for cooling between beds, etc., are all included.

Finally, the program estimates the cost of the requested reactor design and specifies all reactor dimensions. The range of possible solutions for even one methanol synthesis catalyst is such that the program can be used to generate data for statistical experimental design studies.

The program is available in two versions: a version compiled on a Zenith 181 with 640K and an 8087 coprocessor and a version compiled on an IBM mainframe, suitable for use on IBM PC terminals via Kermit. Both versions can be obtained in .EXE form by agreement from Dr. B.W. Wojciechowski, Chemical Engineering Department, Queen's University, Kingston, Ontario K7L 3N6, Canada.

Real-time Multiloop Computer Control Program, UC ONLINE

by Alan Foss

This program is designed for instruction in control system operation in undergraduate and graduate process control courses. Students can rapidly configure any imaginable multiloop control system for either laboratory apparatus or simulated processes. The program runs in real time and is interactive; changes can be made at any time.

Three types of PID controllers are available, each with a nonlinear gain feature, an auxiliary input for gain changes, and automatic status changes. Computational elements include summers, multipliers, dividers, square root, high and low selectors, A/D and D/A conversion, and lead-lags. Detailed and summary information about controllers and variables is displayed on separately accessed screens. Real-time graphical trend displays of any variable are available. The control configuration may be saved to disk and read in again.

Scores of on-line help files and some 50 pages of documentation and examples are provided. The program runs on IBM PCs and PS/2s and compatibles, 256K (512K recommended for use with simulations). A math coprocessor is necessary for extensive simulations. CGA, EGA, or VGA displays. Printer optional. Driver routines for Data Translation #2801 data acquisition board included.

Departmental license for any number of machines: $495.00. Address inquiries to Prof. Alan S. Foss, Department of Chemical Engineering, Gilman Hall, University of California, Berkeley, CA 94720.

Real-time Dynamic Distillation Simulation and Relative Gain Program

by Alan Foss

The distillation simulation is linked with our multiloop control program UC ONLINE, with which many different control systems may be configured and run.

The simulation models concentration dynamics only in a tray column separating a binary mixture. The user may select the relative volatility, number of trays, separation specifications, etc. as a means of setting the conditions of operation. "Measured" outputs of the simulation include top and bottom concentrations (delayed and error corrupted if desired), top and bottom temperatures, and column pressure drop. Five manipulatable inputs are the flow rates of the feed, reflux, distillate, boilup, and bottoms. The simulation is time-scaled to complete a transient in about 2 minutes. Included with the simulation is a preprogram for rapid specification of the column operating conditions and preparation of data files for use by the UC ONLINE control system. This program also calculates the relative gains for the column, quantities useful for deciding on the control system configuration.

This combination of programs is an effective tool for instruction in distillation control in process control courses. Runs on IBM PCs and PS/2s and compatibles, 512K and math coprocessor required. Simulation requires UC ONLINE multiloop control program. Departmental license for any number of machines: $200.00.

Address inquiries to Prof. Alan S. Foss, Depart-

ment of Chemical Engineering, Gilman Hall, University of California, Berkeley, CA 94720.

The following programs have been listed in prior editions of the CACHE News. You can also obtain information about them from the conference "Chemical Engineering Software" on the Bulletin board GRAND@LSUCHE. To get started, send a Bitnet message:

To: Grand@LSUCHE

From: person id@node id

help

(1) Vapor Compression Refrigeration Cycle, by Stanley Sandler, University of Delaware.

(2) Compression of an Ideal Gas, by Stanley Sandler, University of Delaware.

(3) Computer-Aided Analysis for Process Systems, by Ted Cadman, University of Maryland.

(4) Discounted Cash Flow Analysis (and Present Worth), by Bruce A. Finlayson, University of Washington.

(5) Short-cut Distillation and Flash Calculations, by Bruce A. Finlayson, University of Washington.

(6) Convective Diffusion Equation (CDEQN), No. 25 and 26, by Bruce A. Finlayson, University of Washington.

(7) Engineering Plot (ENGNPLOT), No. 25 and 26, by Bruce A. Finlayson, University of Washington.

(8) Educational Software for Teaching Process Dynamics and Control, by Patrick Richard and Jules Thibault, Laval University.

(9) MIDAS—Microcomputer Integrated Distillation Sequences, by Andew Hrymak, McMaster University.

List of Chemical Engineering Departments Supporting CACHE

CACHE annually solicits universities for funds to carry out on-going CACHE activities and nurture new projects. The following is a list of our generous supporters:

University of Alabama, Tuscaloosa
Auburn University
Tuskegee University
University of Arizona
University of Arkansas
University of California, Berkeley
University of California, Davis
University of California, Los Angeles
University of California, Santa Barbara
California Institute of Technology
California State Polytechnic University
University of Southern California
Stanford University
University of Colorado
Colorado School of Mines
Colorado State University
University of Connecticut
Yale University
University of Delaware
Howard University
Florida State University
Florida Institute of Technology
Florida A&M University
University of South Florida
Georgia Institute of Technology
University of Idaho
University of Illinois, Urbana
University of Illinois at Chicago
Illinois Institute of Technology
Northwestern University
Purdue University
University of Notre Dame
Rose-Hulman Institute of Technology
Tri-State University
University of Iowa
Iowa State University
University of Kansas
Kansas State University
University of Kentucky
University of Louisville
Louisiana Technical University
Louisiana State University
University of of Southwestern Louisiana
University of Maine
Johns Hopkins University
University of Lowell
University of Massachusetts, Amherst
Massachusetts Institute of Technology
Northeastern University
Tufts University
Worcester Polytechnic Institute
University of Michigan
Michigan State University
Michigan Technological University
Wayne State University
University of Minnesota
Mississippi State University
University of Missouri, Rolla
Washington University
University of Nebraska, Lincoln
University of Nevada, Reno
Dartmouth College
University of New Hampshire
New Jersey Institute of Technology
Princeton University
Rutgers, The State University
Stevens Institute of Technology
University of New Mexico
City College of City U. of New York
Clarkson University
Cornell University
Manhattan College
Polytechnic University
Pratt Institute
University of Rochester
State U. of New York at Buffalo
Syracuse University
N. Carolina State University, Raleigh
N. Carolina A&T State University
University of North Dakota
University of Akron
Case Western Reserve University
University of Cincinnati
Cleveland State University
University of Dayton
Ohio State University
Ohio University
University of Toledo
Youngstown State University
University of Oklahoma
Oklahoma State University
University of Tulsa
Oregon State University
Bucknell University
Carnegie-Mellon University
Lafayette College
Lehigh University
University of Pennsylvania
Pennsylvania State University
University of Pittsburgh
Villanova University
Widener University
Brown University
Clemson University
University of South Carolina
S. Dakota School of Mines & Tech.
University of Tennessee, Knoxville
Tennessee Tech University
Vanderbilt University
University of Texas at Austin
University of Houston
Lamar University
Rice University
Texas A&M University
Texas Tech University
University of Utah
Brigham Young University
Hampton University
University of Virginia
Virginia Polytechnic Inst. & State U.
Washington State University
University of Washington
W. Virginia College of Graduate Studies
W. Virginia Institite of Technology
W. Virginia University
University of Wisconsin
University of Wyoming
University of Sydney, Australia
University of Adelaide, South Australia
University of Alberta
University of British Columbia
University of Calgary
Laval University
McMaster University
University of New Brunswick
Technical University of Nova Scotia
University of Ottawa
Ecole Polytechnique—U. of Montreal
KAIST, Korea
Queen's University
Royal Military College of Canada
University of Saskatchewan
University of Sherbrooke
University of Toronto
University of Waterloo
University of Western Ontario
University of Windsor
University of Concepcion, Chile
Ecole Nationale Superieure, France
King Abdulaziz U., Saudi Arabia
ETH-Zentrum, Switzerland

LIST OF INDUSTRIAL CONTRIBUTORS TO CACHE

The following companies have recently contributed financial support to specific CACHE activities:

Chevron Research Corporation
DuPont Committee on Educational Aid
Shell Companies Foundation
Xerox Foundation

Topics in this issue of the CACHE Newsletter:

CACHE Corp.
P. O. Box 7939
Austin, Texas 78713-7939

Nonprofit Org.
U.S. Postage Paid
Austin, TX
Permit No. 699

Thomas F. Edgar, Chairman
Department of Chemical Engineering
University of Texas
Austin, TX 78712-1062