An interview with

CLEVE MOLER

Conducted by Thomas Haigh

On

8 and 9 March, 2004

Santa Barbara, California

Interview conducted by the Society for Industrial and Applied Mathematics, as part of grant # DE-FG02-01ER25547 awarded by the US Department of Energy.

Transcript and original tapes donated to the Computer History Museum by the Society for Industrial and Applied Mathematics

ABSTRACT

In this interview, Cleve Moler, Chairman and Chief Scientist of The MathWorks and creator of MATLAB, discusses his career to date. The first session focuses on his academic career, beginning with his education, and his experiences as an undergraduate student of John Todd at Caltech, and as a graduate student of George Forsythe in Stanford's mathematics department. Moler then discusses the development of his intellectual interests, and his subsequent academic appointments at the Universities of Michigan (in mathematics) and New Mexico (in both mathematics and computer science). Moler discusses the origins of the Gatlinberg conferences, and his role in the EISPACK and LINPACK projects run by the Argonne National Laboratory to produce high quality, tested and portable mathematical software during the early- to mid-1970s. These packages are discussed in detail, and framed within the broader context of the development of numerical analysis as a discipline. Moler also discusses his involvement within ACM SIGNUM and SIAM, his managerial roles during the 1980s with Intel's supercomputing division and with failed supercomputer vendor Ardent, and his work as a teacher and author of several important textbooks. The second session turns to MATLAB, the world's most commercially successful piece of numerical analysis software. Moler outlines its development from a freely distributed package he created for educational purposes, and its commercialization by Jack Little and Steve Bangert, originally for the IBM PC in 1984. He recounts the subsequent development of MATLAB in some detail, and analyzes its current and competitors (including several derived from the same original code base), its relationship to open source software, its changing user base, the corporate philosophy and rapid growth of MathWorks, and the development of MATLAB through a number of major revisions including the addition of toolboxes and a graphical user interface.

THOMAS HAIGH: This is an oral history interview for the SIAM history of scientific computing and numerical analysis software project. It's March 8, 2004. This interview is with Cleve Moler of The MathWorks, it's being conducted at his home in Santa Barbara, California, and it's being conducted by Thomas Haigh.

So, to begin with I wonder if you could say a little bit about your family background and your earlier education, particularly with respect to mathematics.

CLEVE MOLER: I was born and raised primarily in Utah. Both my mother and father were journalists, my father worked for United Press for twenty-five years and then was editor of the paper in Ogden, Utah, for twenty-five years. He was a war correspondent at the end of the Second World War for United Press. We moved around some, I actually lived in California, Montana, when I was a little kid, but it was primarily in Utah. The earliest I can really remember being, one of the earliest mathematical experiences was in junior high school, the teacher there was a man named Alfred Persch, I think, I don't imagine he's alive anymore, I'd like to thank him, but he recognized that I had done, was good in math and when it came time to take the math class, probably in ninth grade, he said you know all this stuff. He gave me a book and sent me to the library to read it. The book was left over from the Second World War and had been used to teach a crash course in calculus and college mathematics for Air Force officers. I read through this book when I was in ninth grade and I didn't retain that stuff but that showed my interest in math. I also had very good math teachers in high school. I went to East High School in Salt Lake City, Utah, and Miss Davis was a wonderful teacher and a great motivation. When I was in my senior year in high school there was a statewide math contest and I took the statewide math contest and came in second, the guy who beat me was from a school in another city and he also beat me in the debate tournament. I wonder what happened to that guy? I don't remember his name.

HAIGH: Were you an outstanding student in many subjects, or were your talents focused more on mathematics?

MOLER: Pretty much everything. I was what they called "the brain" there. It was a large high school, six hundred students, and there was a group of us that hung out that were regarded as over the top in terms of mental capacity, and underwhelming in terms of social capacity. There was a girl in that group too, named Nancy Martin. We started dating in high school, she went off to Stanford, and we eventually got married my last year at Caltech.

I was interested in technical things. I was a ham radio operator, but there weren't any computers then. I got out of high school in 1957, and hardly anybody heard of computers there, but I did know that I wanted to go into a technical school, and so the only schools that I applied to were Caltech and MIT. I got admitted to both. They were interested in having me come because I was from Utah, I guess that would be called diversity back then, but both of them were interested in attracting students from the far regions, and Utah qualified. I went to Caltech instead of MIT because it was small; there were just a few students there. I went to Caltech with my Dad and visited before I decided to go

there. I never visited MIT. I turned down a big scholarship at MIT and they were very dismayed that I turned down one of their best scholarships.

HAIGH: In those days did Caltech have the same reputation that it's acquired, pretty much equivalent to MIT or would it be viewed as -

MOLER: It was coming. I went to Caltech in the fall of 1957, and two weeks after I got on campus Russia launched Sputnik, and it was the start of the space age. That was really the beginning that led to the real strength at Caltech's reputation.

HAIGH: When you arrived at university did you know you'd be majoring in mathematics?

MOLER: No. I never was sure I was going to major in mathematics. I even majored in other things. I was going to major in chemistry. I was going to major in physics. I was editor of the school paper my junior year, and a freelance writer, another name I wish I could remember but I don't know it, came to Caltech to do a piece for Fortune magazine, He spent a week at Caltech and did a long piece on Caltech (this must have appeared in Fortune around 1960). As the student newspaper editor I was sort of his representative with the students and I spent a lot of time with him, and I told him I was interested in being a technical writer like he was. My parents were writers, I was actually taking journalism at Caltech in connection with the school newspaper, and I thought that's what I wanted to do, and he said, "Well, it's a good life but you should continue to study math and science, you can only learn that when you're young. You can always learn to be a writer, but study math and science now. Don't give up on that." That was terrific advice. I decided to major in math because it was the most flexible major. You had the most options, and fewer required courses. I took one of the first courses Caltech offered in computing, I took courses in other subjects –

HAIGH: What year would that have been?

MOLER: The first programming I did was not a regular class. It was some extra thing they offered on campus. I think it must have been the Spring of 1959, the end of my sophomore year. The beginning of junior year, I took a numerical analysis class from John Todd. John Todd was… well he's the guy who got me into this business. He's British, or probably Irish, I think, he'd been at the center for numerical analysis at UCLA, in the earlier fifties, and had come to Caltech as a faculty member, and I took the numerical analysis class from him (Larry Shampine was in the same course actually) and we did our homework on the Burroughs 205 Datatron that Caltech had. We would sign up for time and go in and run this. There wasn't any operating system the computer filled the whole room but only one person at a time could use it, you had to be there in the room with the computer to use it.

HAIGH: Correct. So it was an "open shop" system, just like a piece of lab equipment that you could sign up for.

MOLER: That's right, yeah.

HAIGH: And was that the first time you'd seen a computer?

MOLER: Yes, well there was a small computer that we used the year before in this beginning class, the Burroughs 205, was essentially my first computer and that was 1959, my junior year at Caltech. When it came time to go to graduate school Todd recommended that I go to Stanford and work with [G.E.] Forsythe, so that was pretty much my only choice, I was going to go to Stanford.

HAIGH: Right. How was your initial exposure to the computer? Did you enjoy the experience -

MOLER: Oh, very much, yes.

HAIGH: You always knew from that point on that you wanted to work in that kind of area?

MOLER: Well, I guess so. Yes, I think that must be true. I wasn't sure, you know, whether I wanted an academic career or an industrial career. I'm not even sure I knew whether there were careers in computing.

HAIGH: Right. There weren't really well defined ones at that point.

MOLER: I remember one thing, about that time IBM gave a computer to UCLA, must have been a 704, and set up something called the Western Data Processing Center, and this computer was supposed to handle all the computing needs of all the universities in thirteen western states, so nobody had any idea how, where computing was going, how pervasive it was going to become.

HAIGH: Sure. Can you remember who taught your first course in computing?

MOLER: It wasn't a course in computing. There were some lectures on how to program the 205, by a guy from, his may have been named Wilcox, I'm not sure, he wasn't a regular faculty member, he was staff in electrical engineering.

HAIGH: And this wasn't a regular course, this was some optional training session?

MOLER: That right. And then also in the numerical analysis course by Todd, we did our homework on the computer.

HAIGH: And did all the students do that, or was it-

MOLER: No, no, just a very few, very few. Well, everybody who took 205 used the computer, but there were only a handful of students in that class. Only a few students at Caltech at the time were doing anything in computers.

HAIGH: Right. So that was quite an opportunity in those days?

MOLER: That's right, yes.

HAIGH: Now was it also true that from the moment you took Todd's course…. did you know at that point that you wanted to go to graduate school, did you know at that point that you wanted to focus on numerical analysis?

MOLER: I think that's probably true. At that time when I took Todd's course I also took a course in advanced calculus, or beginning real analysis, using a book by [Tom] Apostol (a famous professor at Caltech), taught my a man named [J.K.] Knowles, he was in engineering, engineering mechanics, or something like that at Caltech. I didn't do very well in that course, I think I probably got a B or a C in that class, which was the more theoretical math, and I got an A plus from Todd, so that sort of crystallized things.

HAIGH: Do you think that your attraction to computing and to applied topics is related at all to your broader range of interest, your earlier interest in things like ham radio as opposed to more narrow, purer kinds of mathematics?

MOLER: I think that's right, I think that's right. I've always been interested in the applied aspect of math; I've never been so good at math for its own sake. There's the Putnam exam, it's an exam that undergraduate math students take. I didn't take it cause I could just look down the hallway at Caltech and see half a dozen guys that were better than I was at the Putnam exam so I didn't ever have any pretension of expertise in those areas.

HAIGH: Do you remember what the main topics were in Todd's course?

MOLER: He was working on a book and we had purple, what was it called, dittograph, pages. That was a way of reproducing mimeographed notes that were eventually going to become the book. We could go find Todd's book and see what the chapters are, but it was matrix computation and zero finding and the numerical solution of differential equations. I actually took it, in my senior year I took an individual projects course from Todd and one of the things I did was I wrote a program in absolute numeric machine language for the Burroughs 205 to invert Hilbert matrices. That was the beginning of my matrix computation. I also remember studying an obscure algorithm called reduced penultimate remaindering, which was a way of finding roots of polynomials. I didn't even know what penultimate meant until I learned about that algorithm, invented by [A.C.] Aiken, an obscure crazy algorithm, not very useful, but I learned that with my experiments then.

HAIGH: At the point that you took that course, how well established was numerical analysis as a field?

MOLER: Oh, that's hard to say. The term was certainly there but the UCLA project had been called the "Institute for Numerical Analysis", so that name existed, the course was titled "Numerical Analysis". So the term was certainly known and used, but it hadn't been around for many years, and there weren't very many people involved in it.

HAIGH: So it might only have been done at a few places. Do you think that the outlines of what numerical analysis was, and what the main topics within in it were, would have been pretty well understood?

MOLER: I think so, I think so. I've just finished writing a book called "Numerical Computing with MATLAB" that SIAM is going to publish, and the table of contents of that book could well have been the outline of the course I took at Caltech in 1959.

HAIGH: So, interest in all those broad areas within numerical analysis predated the automatic computer?

MOLER: Absolutely, absolutely, yes. We did some things by hand, we had some Marchant calculators in the library that we used to do hand calculations, the method we used to solve linear equations called Gaussian elimination, it was known to the Chinese two thousand years before Christ, and it's still the same method today, and things Crout's method and Doolittle's method that were done on hand calculators before computers.

HAIGH: So, on the recommendation of Todd you went to graduate school at Stanford -

MOLER: At Stanford –

HAIGH: And you knew from the beginning that you'd be working for Forsythe?

MOLER: Well, it wasn't an automatic thing but that's what I expected to do. My first year at Stanford I was just a math grad student and I didn't really have anything officially to do with Forsythe at all, I had some standard math courses to take, but I soon got connected with him (I forget when the first actual course was I took from him) and when I got to Stanford there was a program in computing. You could get a masters degree in computer something or other from the math department. It was a separate degree program in the math department having to do with computing, and Forsythe and another professor named John Herriott were the professors associated with this. About my second or third year at Stanford they got interested in starting a separate department. They actually wanted to hire John McCarthy, the inventor of LISP, and artificial intelligence and so on, they wanted to hire him from MIT, but Stanford's math department didn't want to hire him because he wasn't a mathematician anymore. So Forsythe started the computer science department in part so they could hire McCarthy, that wasn't the only reason but that was one of the motivating factors, so by 1965, my fourth year at Stanford, the computer science department had came into existence. My last year at Stanford I was a graduate student in math and an instructor in computer science and Forsythe had resigned his professorship in math and become head of the new computer science department. So that was happening while I was there.

HAIGH: All right. And did you feel to any extent torn between the two programs?

MOLER: Oh no, no, no. I wasn't torn. Some of the math courses I had to take I didn't do very well in, abstract algebra, I took a course in Lie groups, didn't do very well in that, there's a famous Danish mathematician named [Lars] Hormander and he visited Stanford and gave a course on partial differential equations that I started to take and then quit because I didn't understand it. The qualifying exams to get onto the Ph.D. program, the major hurdle besides your thesis for the Ph.D. program, you had some choice of the exams and I took three exams, real analysis, complex analysis, and applied analysis, I

didn't take topology, I didn't take algebra, so I was as applied as you could get and still be in the math department.

HAIGH: Right. So, the reason you weren't torn was that your interests were very much on the computation side?

MOLER: Yeah, I was interested in the computation and it was also what I was good at.

HAIGH: How did other people in the math department view the new program? Was the importance widely understood, or did some think of it as a distraction?

MOLER: I think they'd probably still say it's a distraction [laughs]. Numerical analysis has always been the black sheep of mathematics. I like to say that mathematics is the art of avoiding computation. In pure mathematics if you have to do a computation then it's because your model is not sophisticated enough, or not elegant enough, or you haven't got the right abstraction. It was just different, it was a new subject and most of the mathematicians at Stanford, I'd say, were happy to see it get out of the department and go its own way elsewhere.

HAIGH: And apart from Forsyth, whom we shall talk more about in a moment, were there any other graduate students or faculty members that you worked particularly closely with?

MOLER: Several, one. My official academic advisor for my first two years was a man named Karel Delew. He's most famous because several years after I left Caltech he was killed by a graduate student there, he was actually murdered by somebody with a hammer.

HAIGH: That's a sad claim to fame.

MOLER: That's right. He was one of the best lecturers I ever had. I took real analysis, and functional analysis from him and he was immaculately organized and it was beautiful work, I still have his notes somewhere in my library. A student named Bill McKeeman showed up, originally as a physics student then as a math student, but then he became a computer science student and he was one of Stanford's first two Ph.D.s in computer science. He was a very good friend of mine in graduate school and he now works at MathWorks, he's now one of the guys behind our compiler at Mathworks. I would ask Forsyth, "what courses should I take?" And he says take anything that [Menahem] Max Schiffer taught. Schiffer taught methods of mathematical physics and things like that, and again a very good lecturer, a famous teacher as well as researcher. I took partial differential equations from Paul Finn at Stanford, I remember that. I've seen him several times since.

Let me tell you a story about my final oral exam as a Ph.D. So I'd written a thesis. I wasn't around the math department at all my last year; I was off writing my thesis in the computer science department as well as elsewhere on campus. Came time to defend my thesis and there was a committee meeting. When committee was formed at Stanford, somebody from another department was chairman of the committee. Forsythe still had

credentials in the math department, I guess he was still a professor, so he was on the committee, he said it's going to be defense of thesis, you're going to have to stop and talk about the mathematical background, so talk about your thesis and talk about the mathematical background. So I spent a little time boning up on some of the math that I'd used in the thesis and I gave a talk about my thesis and everybody listened politely and then it was time to ask questions. And the first question came from one of the other professors and he said, "Let's see, you took a course in Lie groups didn't you?", and then he started asking me questions about group theory and I was just destroyed. I couldn't even remember what a group was. That doesn't happen to me very often, that I get completely rattled, and one of the other professors started asking me some topology questions, and I did a terrible job with these questions, it was really embarrassing and really awkward.

We came to the end of this awful hour and they asked me wait outside and I waited outside for half an hour, while, apparently, these guys argued with each, and finally they all came out and with sort of grim smiles shook my hand and said congratulations. The next day a notice went up on the bulletin boards around the math department that Ph.D. oral exams are defense of thesis, those are the ground rules, there had been disagreement with these guys as to what the ground rules were. I went on to spend a post doc year in Zurich, and one of the professors from the committee, Robert Ornstein, visited Zurich a year later, and I was still so embarrassed that I didn't even go see him when he came to Zurich, didn't go to his talk, didn't want to face him.

HAIGH: But I think that's something that you find in the academic world, that people have the same few questions that they just ask repeatedly, whatever the occasion.

MOLER: Yeah, but I'm telling this story because it shows my lack of ability in many of the standard topics of a classical mathematics education.

HAIGH: Right. So can we talk about your relationship with Forsyth, and your thesis topic?

MOLER: Okay. George Forsythe got his Ph.D. at Brown in 1939, he actually worked on numerical meteorology, he was one of the members of this important Center for Numerical Analysis at UCLA, at the end of the Second World War He came to Stanford shortly after that in the math department, and went on to found the computer science department. There's a Forsythe building at Stanford. He was president of the Association for Computing Machinery, a very important figure. He wrote a couple of books, some of them with me. He wasn't a famous researcher, there's no, well there's sort of a Forsythe algorithm but it's pretty obscure, and he's known for his organizational abilities, his teaching abilities, rather than his research abilities. There's something called the Forsythe tree, which is the tree of Ph.D. students, you can find it on the web, and there's 17 students in the tree, I'm about in the middle of the tree, and almost all these people have gone on to important positions, some academic and some industrial. The first half of his students were in the math department, and I'm at the end of that group, and the second half of his students were in the computer science department.

For a while he was not only head of the new computer science department, he was also director of the computer center at Stanford. His wife, Alexandra, was also a computer science educator, she taught programming at Gunn High School at Palo Alto, which is a prominent high school at Palo Alto, and for a long time there was a best selling book, or series of books, on programming in computer science by four authors Forsythe, [E.I.] Organick and two other people, and the Forsythe in that is Sandra Forsythe, not George Forsythe.

The man was a Quaker, a very quiet, honest, sincere man. As I was working on the thesis (I'll get back to thesis talk in a minute) he said one day, "When do you think you're going to be done?" It was up to me to decide when I was done. He didn't tell me what to do or how much to do, that was my decision. He said, "No one's ever going to read your thesis unless they think it's wrong."

The thesis topic was something that Forsythe had worked on himself, it was on computing eigenvalues of a Laplace equation, on two dimensional domains. People had worked on obtaining upper and lower bounds for eigenvalues. You can obtain upper bounds by the Rayleigh-Ritz method and choosing test functions, and since the eigenvalue is the minimum of the quadratic form, putting test functions in that quadratic form gives you an upper bound on the eigenvalues. It's harder to get lower bounds on the eigenvalues, and Forsythe had proved that finite difference methods in certain situations produced lower bounds, and so my thesis was extending Forsythe's work in this area, about finite difference methods and under what conditions they produced bounds for the eigenvalues.

HAIGH: And this was something that he suggesting that you work on?

MOLER: Yes.

HAIGH: Were there other topics that you had considered before that?

MOLER: No, I don't think so. I don't remember any, I don't remember what they would have been.

HAIGH: Did the project go reasonably smoothly?

MOLER: Yes it did, it did. I did some computing. I wrote some code, did some experiments. I also proved some theorems. One of the main things I wanted to prove I couldn't prove in the generality that I wanted, but the sort of watered down version is the theorem that's in the thesis. Later other people went on and proved much more general things about finite difference methods. The main example in the thesis was this L shaped membrane, so it's a domain made up of three unit squares arranged in a L, and the important thing is that is has this 270 degree reentrant corner. Forsythe's theory about when finite difference methods provided lower bounds didn't apply to the L shaped membrane, because the gradient is unbounded, if you actually made an L shaped membrane it would rip at that corner and the computations we did showed that in fact the finite difference eigenvalues approached the answer from above, they didn't provide lower bounds even when the mesh size got small. That example, that L shaped

membrane, has gone on to be the MathWorks logo, and we still do things with it today with graphics and it's on our business cards and on a clean highway sign on Route 9 in Natick near our office, and all that got started with the thesis.

HAIGH: So, that wouldn't be a new method but what it would do, am I correct, is to give you improved guidance about the situations under which an existing method would be appropriate to use?

MOLER: That's exactly right. So, it wasn't a new method, I was proving things about the convergence rate of the method.

HAIGH: And would those things be useful to know to implement it on a computer?

MOLER: Yes, they would tell you whether or not you would wanted to implement it. In the case with this unbounded gradient it shows the method isn't very accurate because the convergence is slow there because of the singularity, and so it tells you that you need to modify the method, or use other methods, to get accurate results.

HAIGH: And is this a method that people might have used without a computer or is it a method came in with the computers?

MOLER: It was conceivable, yeah it was possible to use it without a computer. Wilkinson, in England, had done some experiments with the same L shaped membrane and he regarded it as a test example for when he first started doing matrix Eigen calculations. Some of the very first calculations done on computers were done on this problem.

HAIGH: Right. And did you use a computer in the course of your thesis research?

MOLER: Yes, certainly. Very definitely.

HAIGH: So for you at this point the computer was an experimental mathematical tool?

MOLER: Absolutely, absolutely. That's right, you do experiments on the computer and that guides you to theorems you can prove about the method you're using or sometimes just other mathematical topics you can use the computer as a mathematical laboratory.

HAIGH: And was that the main way you continued to use the computer over the next decade or so of your career?

MOLER: I don't think so. I haven't proved a theorem in a long long time, and I haven't done any mathematical analysis of rates of convergence, or that kind of numerical analysis, for a long time. In MATLAB and MathWorks, we don't prove theorems.

HAIGH: I wonder, did you make that shift immediately after your thesis or was it something that happened gradually during your time in academia?

MOLER: It's gradual. And even the fact that there are theorems in the thesis is because I was a math student, so that's what you have to do. I mean, already the thing that was interesting was to me the computation. Well, no that's not fair, I take that back, that was an honest theorem, and that was an honest result about this particular method.

HAIGH: Right. But what fired you up personally was the computation more than the theorem?

MOLER: Right.

HAIGH: What was Forsythe like to work with?

MOLER: Wonderful. He wasn't harsh, he wasn't demanding, he was inspiring, he would give you guidance when you need it. He told me one time, not just about me but about everybody, he says, "It's our job as professors to teach you students what we know and then get out of your way", and I've always remembered that. I'm thinking that about MathWorks today. There are young people at MathWorks today working on parts of our software that I've always been very much involved in, and I don't always agree with them, but I'm getting out of their way. It's time for them to do it their way, and Forsythe would have had that attitude.

HAIGH: So that's something you learned from him?

MOLER: Yeah.

HAIGH: Now your book with Forsythe, *Computer Solutions of Linear Algebraic Systems*, Prentice-Hall 1967, now that's almost the first thing that you published –

MOLER: That's correct.

HAIGH: How did that come to be?

MOLER: Forsythe taught a numerical analysis course, it was math or computer science, or both, 137, and he was using a text book from his friend and colleague, Peter Henrici, called *Elements of Numerical Analysis*, so an important little text book in numerical analysis, but it doesn't have anything about matrices in it, it's about differential equations and zero finding and interpolation, and so on, so it's half of the course, so Forsythe wrote some notes for the other half. And then the next year I taught the course. I was still a graduate student, I was only an instructor in the department, but I taught that course, I guess that would have in 1965, and I expanded on the notes.

HAIGH: That was the course you mentioned earlier, teaching while you were finishing up your Ph.D. –

MOLER: Yes, that's right. Well one of them. I also taught a beginning programming course. But this numerical analysis course… in this book there are some computer codes, there's programs for solving systems of linear equations and they're written in Algol FORTRAN and PL/1, and this is one of the first text books that had serious software in it.

It's been recognized, it's in a list from the Association of Computing Machinery of the 20 most influential text books in the history of computing, or something like that, because of its use of software early on. It took a while to finish that book, I went on to Zurich on a post doc and the PL/1 program in that book was actually written at an IBM research center in Rüschlikon, Switzerland, after I left Stanford.

HAIGH: So how were responsibilities for the book divided between you?

MOLER: Oh, we both worked on it. We wrote pieces of it, we read it, we worked on individual chapters. I was in charge of the software in the book, I wrote a chapter on Hilbert matrices that actually reflects my experience from Todd at Caltech. The introductory stuff, Forsythe wrote.

HAIGH: Right. So looking at the book, my impression was that it's designed to take students who have an understanding of the mathematics and guide them towards being able to do some practical implementations of it.

MOLER: And visa versa. It also, you know people who just know about programming computers can learn some of the mathematics that underlies it from this book, too.

HAIGH: So, before you began work on the book, did you and Forsythe collectively, have experience with all the areas that are covered in different chapters, or were there some that you have to go away and work on in order to write the book?

MOLER: No, no. We knew this stuff, we knew all of this stuff. The book evolved from notes from the course so we knew that stuff when we were teaching the course, and we were writing those notes. I think the only thing new in the book is the PL/1 program, we thought at the time that maybe PL/1 was going to become important programming language, it turns out, well it didn't have the staying power that we thought it might have. That's the only PL/1 program I've written in my life.

HAIGH: Oh.

[End of Tape 1, Side A] [Start of Tape 2, Side B]

Although there is software in the book it seems that the approach in it, to use the cliché, is very much, to teach the man to fish rather than to give him the fish in terms of preparing people. It's not a collection of libraries, but a tutorial on how to write their own routines.

MOLER: No, no, no. This is a little book, it's only less than 150 pages, the software is just, it's not a complete set of software for doing anything. It's a very important computation. Solving systems of simultaneous linear equations is arguably the most important computation in scientific computing, it underlies everything else, but it's rare that you do it just by itself. There's very little in here about why you want to solve systems of equations, there's very little here about the applications of this to serious scientific computing.

HAIGH: Yes. So at that point, if you were an engineer or a scientist, say a Ph.D. student, who had a practical need, for your own research to do this would it have been normal, would it have been normal to have taken a basic course or two explaining the mathematics –

MOLER: Right.

HAIGH: And take another course with this book and then to write your own code?

MOLER: You would use, if you were doing let's say studying control theory or finite element computations in engineering at this time, you would have used this subroutine from this book in your own program –

HAIGH: Right.

MOLER: You'd write more stuff for the particular problem you were solving but this subroutine out of this book would be one of the things you would use in your own code.

HAIGH: Right. So back in those days you couldn't assume that at the computer center there'd by a nice collection of subroutines and libraries, so you could just plug one in?

MOLER: Oh you could. Well, there was beginning to be that. At that time, in the middle 60s, there was an organization of IBM users called SHARE, and there was a collection of software that SHARE maintained and distributed, and the FORTRAN program that's in this book for solving systems of liner equations, I also contributed that same program to SHARE. That same program, or pretty much that same program, was also published in the Communications of the ACM algorithms collection. This particular program in this book was the first FORTRAN program that they had in the ACM algorithm section (the programs were originally all in Algol). Stanford had a computer center and there was a library of subroutines at the computer center, and this was one of them, so this is a book about one of the basic subroutines that was in any early math library.

HAIGH: And how was the book received?

MOLER: It was widely cited and widely used. It's not a complete text book, it's not a text book for a whole course, but it had quite an influence there.

HAIGH: Were the ways in which it was used the same ways you'd expected it to be?

MOLER: Oh, I think so. Yes. I don't know, but it was, you know, other people used it in other ways in other courses. We referred to, you know somebody's teaching it, teaching a physics course or mechanical engineering course or something like that, in which this would be one of many topics there and they just referenced this book and say, "here's the latest word on solving linear equations."

HAIGH: One of the things, as someone who's not trained in this specific area, I'm having some trouble with, is understanding which bits of mathematics are particularly important

for which application areas. So, could you say a little bit more about the specific disciplines in which this book might have been used most?

MOLER: As I said, solving systems of simultaneous linear equations, I would argue, is *the* most important topic in scientific computing. They come in many forms. This particular book handles fairly small systems and modern computing today solves much larger systems to which this software doesn't apply, but still the topic is there, so I can't think of an area of scientific computing which doesn't rely on solving linear equations. Control theory and signal processing, my wife was just looking for a hearing aid for her mother on the Web and came across hearing aid companies, and these guys use MATLAB and they solve, they use this software, they use what's become of this software in designing hearing aids. All the big codes at the national laboratories doing weapons effects are solving systems of simultaneous linear equations. Wall Street predicting stock prices and futures prices, and so on, does this. Statistics, automobile, aerospace, any place.

HAIGH: So it's extremely general?

MOLER: Yes.

HAIGH: There isn't a strong coupling of specific methods that you might use in the application area, is that true?

MOLER: That's right. Simultaneous, matrices are matrices no matter what discipline your in, and solving linear equations is independent of where they come from until they get to be really big.

HAIGH: Right. So it's something that would have been foundational over an extremely wide area?

MOLER: Right.

HAIGH: Now you mention at this point that the book dealt only with very small systems. Was that a limitation for the field as a whole at that point?

MOLER: No, no. People were also solving large systems at that time but by different methods, so this is about direct methods for the solution of dense linear systems, there are also iterative methods for solving the kind of equations you get in finite difference methods for partial differential equations, for example. So, my thesis was about a matrix problem that came from partial differential equations. I didn't use the stuff in this book in my thesis… different kind of matrix.

HAIGH: Now one complaint that people in science sometimes have is that writing text books doesn't necessarily get you the same academic credit that other kinds of work would. Do you feel that –

MOLER: That's certainly true, absolutely. It ain't research.

HAIGH: So as a young academic did you feel that this didn't count for as much in terms of tenure and hiring, and those kinds of things?

MOLER: I think that's true. I did not write this book as a young academic… well when a graduate student I wrote my thesis and this was something I worked on… it was a side line, it wasn't ever the main activity I was involved in. It's become one of the things I'm best known for, but it didn't get me tenure.

HAIGH: Okay. So, unless you have anything else to say about the book at that point let's shift to that other screen with your other publications and what did get you tenure.

MOLER: Okay.

HAIGH: So, immediately after graduating you went to Zurich you mentioned -

MOLER: Right.

HAIGH: That was for post doc –

MOLER: Interesting there. There was a famous institute at ETH, the Swiss Federal Institute of Technology, a famous computing institute, run by a man named Eduard Stiefel. Stiefel was the epitome of the classic German Herr Professor Doctor. He had been at this Institute for Numerical Analysis at UCLA, along with Forsythe and Todd, and that Institute had gotten it's sponsorship from the Office of Naval Research, so the Office of Naval Research had a post doctoral graduate fellowship for the study of numerical analysis at Stiefel's institute in Zurich, it was that specialized. Four people in history have had that fellowship, and all of them went on to make names for themselves in numerical analysis business. I was the third one, so I went off, I went off to Zurich, but I was actually still paid from the Office of Naval Research through Stanford. Zurich was a famous important place, there were three world famous numerical analysts there, Stiefel, Rutishauser, and Henrici had just returned from UCLA, he'd gotten the call to go back home to also become a full professor, which in Switzerland is far more than a full professor in the United States is. We used to joke about three body problem, these three eminent figures in orbit about each other because they were all their own personalities, but Stiefel ran the computer center as well this Institute, he had attracted, there were a lot of visitors of various kinds, there were students of various kinds, I never saw anything of Stiefel, I spent a lot of time with Henrici, Rutishauser was sick and I didn't have much to do with him, but there were some of the other visitors and other young people at the institute that I worked with.

HAIGH: How large was the Institute?

MOLER: It was a computer center as well as a numerical analysis research institute, there were probably 200 people on the staff, but a lot of those were computer center staff. So the computer center served the whole institute. The ETH is the Swiss MIT, and at that time these places had central computer centers so that was the place I was visiting. James Lyness, a Brit who had been in Australia, came at the same time, and he had a lot of sort of quirky little problems that he worked on, things that weren't central, but I got

interested in them and between us I think we wrote three papers while I was in Zurich, or shortly thereafter. A few years later, when it came time to get tenure at the University of Michigan, those papers had been published and that was one of the important things. Those are not really central stuff, but they're interesting little research papers. The more important thing there was the L shaped membrane, the paper with Fox and Henrici. So, as I say, I'd worked on this L shaped membrane problem in my thesis, Leslie Fox was a professor at Oxford, he'd actually had two graduate students work on related problems. There was a small IBM research center in the suburb of Zurich called, well near Zurich, that had a weekly math seminar and Fox came and lectured at that seminar and talked about the work his students had been doing and it gave me an idea about how to do the L shaped membrane that didn't involve finite differences, that took advantage of this singularity rather than be jeopardized by it. I worked on that very hard, I remember I got some important results just before Christmas in 1965, and our Christmas tree had a little L shaped membrane that I made out of tin foil. I told Henrici about it, he had an idea about some of the theoretical underpinnings and so I wrote a paper about what I'd done, and about what Henrici had added, and then because Fox had set the thing off with his lecture we invited him to join us in the paper, and he did. That became a very important paper. In fact, Nick Trefethen has just now written a paper for SIAM, where he revives that method and brings it up to date, improves on it. He's very active in SIAM today.

HAIGH: Right. So then that paper "Approximation and Bounds for Eigenvalues of Elliptic Operators," you consider that's the most important of your early papers?

MOLER: That's right, I agree. There's another one I see here on that list by Larry, with Larry Payne, he came through Zurich at the time, and I told him about this, and he said, "Oh, well here's a better way to do that," and so we wrote another little paper about that.

HAIGH: And was it a better way?

MOLER: It's not a change in the method, it's a change how you analyze the results. The paper with Payne is a theorem about how accurate the method is.

HAIGH: So, in the Fox and Henrici paper you're actually proposing a new method?

MOLER: Yes, that's right.

HAIGH: And was that method of practical importance?

MOLER: Yes, it was.

HAIGH: Who used it?

MOLER: Some of my students [laughs]. It's not of great practical importance, it's pretty specialized, it's only about eigenvalue problems and only for certain kinds of domains, I had thought that it would have more practical importance than it turned out it has. Two of my students later, [Norman] Schryer and [Stanley] Eisenstat, both wrote papers about it, and as I say Trefethen is resurrecting the method. It's not of great practical importance.

HAIGH: Just to get a sense of how big the area was at that point, how many reasonably practical methods would there have been at that point for the automatic computation of eigenvalues and their upper and lower bounds?

MOLER: How many methods?

HAIGH: Yeah, I mean is it the kind of thing where there might be a dozen methods –

MOLER: Fewer than that, fewer than that. A couple.

HAIGH: And at this point were new methods being created in order to use the potential of automatic computers?

MOLER: The work in this particular field is not about inventing methods, but about analyzing the methods that already exist, so there have been dozens if not hundreds of papers written about the rates of convergence of finite difference methods, but there's only one finite difference method… no that's not right, but there are more papers… few papers actually invent new methods, they mostly analyze methods that exist already.

HAIGH: Would typical users of methods actually be well aware of this research on what might be the best method you might say to use it or were they a little naïve about these things?

MOLER: Naïve. They use methods based on what software they can get hold of, or what software guys down the hall are using, rather than reading the background papers about the theory of the methods.

HAIGH: Yeah. That would have been completely true in the sixties?

MOLER: Even more true then, yeah.

HAIGH: Oh, by the way, in addition you mentioned Zurich, and obviously the INA, but were there other influential institutions in numerical analysis?

MOLER: Oh, certainly, yes. So the other prominent institutions: Oxford, Cambridge, Stockholm, University of Illinois, University of Maryland, University of Texas, each of these has got probably one famous guy there. Gear in Illinois, Young in Texas. University of Toronto, Tom Hull, University of Maryland. Curiously some of our top educational institutions wouldn't be on my list. MIT, Harvard and Yale, for example, none of them have particularly prominent numerical analysts or even… well, I'll limit it to that. So those would not be famous places in this area.

HAIGH: And had all those institutions moved into electronic computing to the same extent?

MOLER: Yes. Yes, by 1965 every major university in the world had a computer center. There were no personal computers. The mode of computing is you would write your program on punch cards, submit it to the computer center, it would be run on the main

frame, output printed on big sheets of paper with green lines across it, and then put into a box where you'd pick up your output a few hours later.

HAIGH: So in the sixties what do you think the main effects of electronic computing were on the intellectual content of numerical analysis, and the kinds of work that were being done by academics?

MOLER: Well, in numerical analysis it's always been about computation. Numerical analysis is the invention of algorithms, the study of algorithms, the convergence of algorithms, the rates of convergence, the errors in algorithms, and originally it was about hand computation, and then as soon as you got computers people started using computers, usually (originally at least) with the same methods that you were doing by hand. But it's always been that the motivation for numerical analysis is computation. The question is whether people writing the theoretical numerical analysis papers are actually doing their own computation, or whether they're having students do it for them, or whether they're talking about other people's computations. So you can do numerical analysis and write papers about numerical analysis without actually doing any computations yourself, but the computations are always the background of what you're writing about.

HAIGH: I would think that result of there being a lot of people using computers would be more interest in the field of numerical analysis in general.

MOLER: Oh yeah, oh yes, it's not just numerical analysis. In 1965, well about 1965 computers were being used for lots of things, they were being used for technical computation by scientists and engineers, but we certainly had business data processing, we had airlines reservation systems…

HAIGH: By the mid sixties there were tens of thousands of computer systems –

MOLER: Yes, yeah.

HAIGH: And yet a substantial minority of them would have been used primarily for scientific research and technical computation.

MOLER: Yes, yes, that's right.

HAIGH: So clearly there are an enormous number of people working computer centers, and performing computations is something that graduate students and researchers in lots of areas were doing.

MOLER: Yes.

HAIGH: Now what I was wondering was how much that fed back into, and changed, the character of what numerical analysts were doing. Were they doing pretty much the same thing they'd always done, or were there new challenges and new methods?

MOLER: Yes, new challenges and new methods. There have been studies that show that the progress we've made in computing in terms of both speed and capacity have been as much have been as much due to algorithm and software improvements as hardware improvemens, so there've been all kinds of new methods invented and analyzed that didn't exist when I was in graduate school. The fast Fourier transform wasn't known, Golub and Kahan had developed the method for computing the singular value decomposition about l965, multi grid method, fast multi pole method, much of what we know about numerical solution of ordinary differential equations has been invented since I was in graduate school.

HAIGH: Right. And if I can push you to a gross generalization, are most of those methods, things just wouldn't have been practical to do without the electronic computer?

MOLER: Oh, absolutely, Yes that's right. That isn't a push.

HAIGH: Now, you mentioned that people working in technical areas and weren't specialists in numerical analysis were often ignorant of when methods should and shouldn't be used –

MOLER: Yes.

HAIGH: How was communication the other way. Did academic specialist in numerical analysis, have a good sense of the kinds of problems that people in disciplines were working with and what was and wasn't useful?

MOLER: No, I don't think so. There can be a pretty wide separation of it, if you look at the kind of computations that are going on in the automobile industry, or in the aerospace industry… there are a lot of computations done in the automobile industry to figure out the effect of fluid flow on a car and the mileage of the car, in order to reduce the mileage you want to reduce the aerodynamic drag. Cars now all look the same because they all solve the same partial differential equation… well except SUVs, they ignore that fact –

HAIGH: They're not motivated by practicality –

MOLER: That's right. But I don't think even today that many people working in numerical analysis really understand how the automobile industry is computing that flow.

HAIGH: And do you think that that's something they should know, or is the mathematics so much of a world of its own that it wouldn't make any difference if they did know?

MOLER: The real big industrial computations, the design of automobiles, the design of bombs, are big complicated problems that are using lots of methods and lots of mathematics, and the numerical analysts can't deal with something that complicated, that big. So if we're doing theoretical numerical analysis we have to concentrate on one particular method.

HAIGH: So that would seem to make the field in some ways a strange combination of pure and applied

MOLER: I wouldn't call it a strange combination, there's certainly both aspects to it. There's pure mathematics, there's theorems and proofs, and people look at numerical analysis in abstract function spaces, the kind of approximation theory that people like [I.] Babuska, and so on, studied for finite element methods is pretty far removed from what's being done when you design the fender of a car. There's a whole continuum, from the abstract theoretical to the practical computation and the software and the engineering practice.

HAIGH: Right. Is pure numerical analysis something that pure mathematicians as a whole would consider –

MOLER: Pure mathematicians would not consider it respectable. Pure mathematicians would still consider it applied. I mean this is what SIAM is about, SIAM exists because, in all the fields that SIAM covers, the work that's represented in the SIAM journals is not regarded as the kind of mathematics that would be published in the journals of the American Math Society. We have our own agenda here, and visa versa, stuff that the AMS publishes we wouldn't publish in SIAM.

HAIGH: Okay. I think that's helping me to understand how the field works in that way. Now back to your career, sir. After your, was it an entire in Zurich –

MOLER: Yes.

HAIGH: Okay, after your year in Zurich, your first –

MOLER: Job was at the University of Michigan –

HAIGH: As an assistant professor, and then as an associate professor?

MOLER: Right.

HAIGH: In the mathematics department.

MOLER: Yeah. While I was in Zurich I thought about what was I going to do after my post doc year, I considered several places, I'd even had some visits before I went to Zurich about where I might possibly want to go. I had a number of job offers for the beginnings of permanent positions, UC Berkeley, IBM Research was a possibility, in Yorktown Heights, but at the University of Michigan there was a numerical analysis professor named [R.H.] Bartels, who was just retiring and they were looking for a successor for Bartels. There was another famous computer scientist named [B.A.] Bernie Galler there, you know about that name from the history of computing, and Galler was recruiting a successor to Bartels… and I liked these people, they were nice people. It was in the math department. Computer science at Michigan was already a funny place, it was called the Department of Computing and Information Science, and it was not only computing but information theory and human language, and so on, and the numerical analysis was definitely in the math department. Though again, it was in another part of the math department, the math department's main offices were on one side of the campus and then they had a satellite office across campus in the College of Engineering, where

the applied mathematicians were, so my home was already across the campus from the center of the math department. The math department there was growing rapidly. The year I came they hired sixteen new assistant professors in the math department, it was incredible. And then there was a revolution, whole change of attitude about the department about what was going to lead to tenure, and many of these young people that were hired at the same time I was were very disappointed and unhappy about what happened with the tenure considerations. It ripped the math department apart a few years after I got there.

HAIGH: Did they just raised the bar, or was there –

MOLER: Yeah, they realized that all these sixteen people who had come they just couldn't keep them all and, in fact, only two of them got tenure out of that sixteen, everybody else left. You know these were your friends, you socialized with them, played poker with them, played volleyball with them, and the next year they're gone, it was like being in the Army.

HAIGH: So, as you said that this was within the mathematics department, although there was a computing operation. Were any of your job offers at that point from departments that would call themselves computer science?

MOLER: Yes, there were. The IBM offer was from IBM, that wasn't math. The Berkeley offer was from starting a computer science department at Berkeley, so that already would have been a computer science department. While at Michigan I actually entertained the possibility of going to the University of Texas, and went down there for a visit and had an offer from them, and that was in computer science.

HAIGH: So, at that point if you'd have had to choose one or the other would you have called yourself a mathematician or a computer scientist?

MOLER: Well, that's always been a question. I have a joke, I don't know if you know what the Lorenz Attractor looks like, that chaotic system with the butterfly. I have a map that shows that orbit and I say this is numerical analysis trying to decide whether it's mathematics or computer science, it's attracted by both centers but as it gets too close, it's repelled, and it's different at different places, it has a lot to do with personalities of people involved and history of people involved. Today numerical analysis really falls in the cracks between math and computer science, I can't think of any place where it's a central discipline in either department. Here at the University of California at Santa Barbara, I'm actually visiting here teaching a numerical analysis class that is one of the possible required courses for undergraduates in computer science. It's rare that computer science students take numerical analysis anymore. They don't at Stanford, for example.

HAIGH: I didn't at Manchester -

MOLER: Right, yes, yes.

HAIGH: We had one course at the beginning which had some differentiation, some integration, and some matrices, but they never explained how those numbers might have got into the matrix in the first place –

MOLER: That's right –

HAIGH: And why you would want to do all these operations to them, so I promptly forgot it all afterwards.

MOLER: You actually have a very famous, there's actually a very famous numerical analyst at Manchester, famous group, there by a guy named Nick Higham, I'm not even sure what department he's in I don't think it's computer science, UMIST, what's that?

HAIGH: That used to be an offshoot of the University, then it became almost independent, and just within the last couple of years they've merged it back in again. It stood for the University of Manchester Institute of Science and Technology.

MOLER: It's not even an academic department it's a-

HAIGH: It was odd because UMIST specialized in mathematics, and engineering, and some management and the University of Manchester had its own parallel arms in those areas. I think they realized that consolidation made sense.

MOLER: Right, right. There's stories like that about lots of places today.

HAIGH: Correct. So then in those terms that's something that you think hasn't changed much over time, that numerical analysis has remained somewhat straddled between the fences of computer science and mathematics?

MOLER: That's right. I think it was Bob Floyd, the chairman of computer science at Stanford, who said many years ago that numerical analysis was the mother of computer science, but now it's acting like an anxious grandmother.

HAIGH: At Michigan, were there people either within the computer and information science department, or colleagues within the mathematics department, who you worked closely with?

MOLER: Yes, yes. I made good friends, we wrote papers together, there's a guy there named Smoller, we wrote a paper together about a problem he was interested in. That was again, that was an example of where we used computers to do experiments to discover a theorem in mathematics. In computer science, Galler. I didn't write any papers with but I knew him quite well, I had a lot to do with the computer center. Nothing else prominent comes to mind. You're looking at the resume there –

HAIGH: There's an L.P. Sullivan, was that a colleague?

MOLER: Oh, Lou, yes. Lou Sullivan, he was a guy in mechanical engineering that was interested in acoustics, underwater acoustics, and he came to me with this problem and we worked on it, and wrote a couple of papers about that. I'd forgotten about him.

HAIGH: At that point, then, you were in contact with people who were using computers to solve real problems?

MOLER: Yes. Question is, what's a real problem? I mean even the kind of research they do in a mechanical engineering department, is that a real problem? I don't know.

HAIGH: Okay. Well, with people in other disciplines and other application areas then?

MOLER: That's right. Another important thing that was happening was the connection with Argonne and the summer school at Michigan, we should talk about both of these.

First of all the summer school, so the University of Michigan ran short courses in the summer, and it became quite a business for Michigan, they actually built something called the Chrysler Center on the north campus. I'm not sure what the status is today, but it was a major operation at the University of Michigan, I don't know of any school that had anything comparable. There were dozens of short courses offered there during the summer and before I got, before 1965, they had one course, and then two courses, and then three courses in computing, and then it split off to one in languages, and one in operating systems, and one in numerical analysis. Galler ran one, Bartels ran one. The numerical analysis one had been going for several years before I got there and it was one of the major events in numerical analysis research in the world. Prominent numerical analyst researches were invited to Michigan for two weeks in the summer to give a series of lectures, students would come and pay to take these lectures, and these guys, initially these guys would listen to each others lectures because research in numerical analysis and graduate short courses were pretty much at the same level. Alston Householder, from Oak Ridge and the University of Tennessee, came every year for sixteen years, J.H. Wilkinson came from England every year for fifteen years, the father of modern matrix computation, and, besides Forsythe, my mentor. Todd came, Henrici came, I'm not sure whether Forsythe did. Others, Richard Varga, for example, were invited to give these lectures.

When I got there that became one of my jobs, organizing the summer conference, and it became my job to choose these guys, and sometimes I just invited the guys that had been coming before, like Householder and Wilkinson, other times I invited new guys. I ran this course every summer for the next six years. That was very important, very important in numerical analysis in general, and matrix computation in particular. Wilkinson's lecture notes in Michigan became his famous book, *The Algebraic Eigenvalue Problem,* and that was the foundation of modern matrix computation. Wilkinson was also working on publishing his algorithms for matrix computation in this journal called Numerische Mathematik…

[End of Tape 1, Side B] [Start of Tape 2, Side A]

MOLER: So, Wilkinson would come from England, he was at the National Physical Laboratory, would come to Michigan, spend a week or two there, give these lectures, and then go to Argonne Laboratory and spend a week or two there. He was working on these algorithms to be published in Algol, and at Argonne they had the idea, particularly a guy named Wayne Cowell, to make FORTRAN versions of these programs to make them more widely available, and that led to the EISPACK and LINPACK projects. The meetings between these guys, Wilkinson, Householder, Varga, also led….

At Michigan in the summer, they decided to organize a series of research conferences. Householder organized the first several of them in Gatlinburg, Tennessee, near the Oak Ridge National Laboratories, those were called the Gatlinburg conferences and then later the Householder conferences. They are select, invited, conferences where a self perpetuating international committee invites people to come to those conferences. At first it was very controversial because they were kind of secret and select. Now, you can apply to go, and the committee still selects who goes, but at least it's open. I've been to those conferences, every one since 1964. In 1964 I was still a grad student, and Forsythe was one of the organizers and he wanted me to come, the rest of the committee was opposed to having students come, Forsythe said, "If Moler can't come I won't come either," so I got to come. Pete Stewart also came to one early on, and those conferences have been a very important part of my career because I met early on these famous guys in matrix computation, particularly Wilkinson.

HAIGH: So, what you were just talking about were the Gatlinburg –

MOLER: Originally called the Gatlinburg Conferences, and then when Householder retired and they stopped having them in Gatlinburg, Tennessee, they went out and they're now held every three or four years at places around the world, they're now called the Householder Conferences.

HAIGH: Do you know when the Michigan summer schools in numerical analysis started

MOLER: I don't, it was –

HAIGH: You arrived in 1966 –

MOLER: Yeah, but it had been going since the fifties. They could have already been going ten years in 1966.

HAIGH: At the point you arrived and took it over were these big names still coming?

MOLER: Yes, yes.

HAIGH: Who would the students have been?

MOLER: They were mostly… well, I'm not exactly sure. They were probably mostly students from industry, from General Motors and Ford, the aerospace industry, you know, Lockheed, Hughes. Maybe academic students would come, but they maybe couldn't

afford it. These kind of short courses didn't fit within an academic program, they were intended for an industrial research audience.

HAIGH: So the conferences would have been an important channel of communication between the industrial community and the researchers in numerical analysis?

MOLER: Yeah, absolutely, absolutely, yes.

HAIGH: What made the instructors come?

MOLER: Each other. They would come to see each other, they were good friends, Wilkinson and Householder came out of habit, and then other people were honored to be asked to come and be part of this group. And they made money at it too, they go paid.

HAIGH: During this period from 1966 –

MOLER: I bet… there isn't anything about that on my resume, is there?

HAIGH: You've just got one line –

MOLER: One line –

HAIGH: "Co-chairman, Michigan Summer Conferences in Numerical Analysis."

MOLER: That was actually very important.

HAIGH: During the 1966 to 1972 period at Michigan, what were you working on and how did your interests change over time?

MOLER: Well, I was in the math department so I was supposed to be doing research on mathematics but I was also involved with the software effort, the Argonne effort for EISPACK was starting –

HAIGH: Right, and we should definitely talk about that.

MOLER: Yeah. I actually had a research contract from the Office of Naval Research for a couple of years, to support my sort of theoretical work in numerical analysis. It was a follow on to the fellowship I'd had in Zurich. And then, an important thing happened. After a couple of years at Michigan the Office of Naval Research asked me not to reapply because I was doing this software stuff and that wasn't mathematics, that wasn't the kind of thing that they sponsored. I got sponsorship elsewhere, from the National Science Foundation and through Argonne, and so on, but that was significant: the Office of Naval Research was sponsoring research in applied mathematics and when it turned to software they said, "that isn't research."

HAIGH: And was that a shock to you…

MOLER: Yes it was

HAIGH: …..to realize that you'd wandered out of the discipline without realizing it?

MOLER: I wouldn't put it that way, that I'd wandered out of the discipline… I thought it was shortsighted on their part but it's probably still, well I don't whether it's true today or not, but maybe it is. The government agencies have certain focus and certain interests and people that are participating in their programs can wander outside that field.

HAIGH: So anything else on Michigan before we turn to LINPACK and EISPACK?

MOLER: I got tenure at Michigan, I could have stayed there forever. I'm glad I didn't… the woman I was married to at the time got her Ph.D. in this computing and information science program at Michigan, she worked on genetic algorithms, and we wanted to go someplace where we could both get an academic job, and she couldn't get a decent academic job in Michigan. So we looked around to a couple of places. Colorado was one possibility, but the University of New Mexico, in Albuquerque, offered us both jobs, me in the math department, and her in the computer science department, so I resigned my tenure position at Ann Arbor and we went to New Mexico.

HAIGH: And when you got tenure you were one of only two out of 16 people –

MOLER: That came that one year, yeah.

HAIGH: So, aside from the quality of your publications and intellectual merit, what made your case compelling where the others had failed?

MOLER: Well, I think it was those things that you just mentioned. I didn't particularly worry about getting tenure, it wasn't a big stressful thing, it just happened. I'd done enough work and the work I was doing was important. The department really wanted this numerical analysis stuff, it was important, this was one of the places where they got student credit hours from students in other disciplines, so that the students that took my courses were not just math students, there were math students but there were also students from engineering and physics. Nuclear engineering was a respected discipline at the time and some of the students with the best mathematical knowledge came from nuclear engineering, and the math department wanted to continue that to support that activity.

HAIGH: And within the department you were the main numerical analyst?

MOLER: That's correct, yes.

HAIGH: So your closest ties would be with people outside the department, within this community?

MOLER: Well, I'm not sure I'd say that. The students from nuclear engineering, and so on, I didn't ever have anything to do with, I don't think, with the professors that were in those departments. No, I didn't have close ties outside the math department.

HAIGH: No, I mean outside institutions as well. You know, with people like Forsythe, and other people you've mentioned who came to conferences.

MOLER: Well yes, and this connection with Argonne, was becoming important.

HAIGH: So let's move to Argonne: EISPACK and LINPACK, how did those begin?

MOLER: I'm not sure why Wilkinson was visiting Argonne, but he'd go there after Michigan and talk to people about this algorithmic work he was doing. A woman working there names Virginia Klema, took his Algol and translated it into FORTRAN for use in the Argonne library, without any thought of making it a bigger deal, and she did not produce professional quality distributable software. So a guy named Wayne Cowell (who is still alive, and I have his address here someplace, we might want to talk to him), it was his idea to turn this into a big project. Argonne was sponsored by what was then called the Atomic Energy Commission, I think, later it became the Department of Energy, but Cowell was also going to get separate funding from other government agencies, National Science Foundation… I'm not sure where all the money came from, certainly a separate budget from the Department of Energy…. The idea was to, careful how it's worded, we were doing research on the translation and porting and certification of mathematical software, we weren't writing new software. We were taking these algorithms that were developed by Wilkinson and his colleagues, and published in *Numerische Mathematik*, and doing research in the process of making a widely distributable library out of it. So the research in that process and the end product, the library itself, was sort of incidental. In a sense we failed, because we never wrote the paper about the process, but that was what the original proposal was for.

HAIGH: Now was that your real motivation, or was that a way of turning what you wanted to do into a respectable fundable piece of research?

MOLER: I think you said it just right, yes.

HAIGH: So what was the real motivation?

MOLER: Well, to produce software, that can be widely used.

HAIGH: Because you thought the world needed it?

MOLER: Yes.

HAIGH: Why did the world need it?

MOLER: Well, because these were important problems, these matrix computation problems, particularly the eigenvalue computation. By this time everybody had subroutines for solving linear equations, inverting matrices and so on, but the methods for computing matrix eigenvalues were not widely available as software. These methods had just been invented, really practical methods, and Wilkinson had codified them and analyzed them and made Algol versions of them. So the first project was EISPACK which is about eigenvalues. Now, in terms of complexity, and so on, those come after

linear equations, so if you were going to start off and do a matrix library you'd start with linear equations and do eigenvalues later. We did it backwards because everybody already had linear equations and so we started with the hard stuff and the easy stuff came later.

HAIGH: So, in this case, it wasn't that people already had eigenvalue routines but that they were terrible or somehow lacking. In most cases people who might want to get eigenvalues just couldn't?

MOLER: Well, there were a lot of methods around, there was software around, but the methods weren't very good and weren't very reliable, and weren't uniform. One important thing here was to provide a library, so that people throughout the world could use the same library.

HAIGH: Okay. And this was with Wilkinson's newly developed superior methods?

MOLER: That's correct, yeah.

HAIGH: So was this unprecedented, in that someone who has come up with a new method, and wants to spread it to the world doesn't just publish it in a journal and wait for nature to take it's course, but actually sets out to produce a high quality software package and systematically disseminate it.

MOLER: I think that was unprecedented. Well, people had been publishing software before… but there was a two step process. Wilkinson was already publishing the software because *Numerische Mathematik* had this series where they would publish algorithms as a traditional journal publication, so you'd write this paper, the paper would include Algol code, and it would get refereed and published. But there wasn't any distribution mechanism. The *Communications of the ACM* also had an algorithm section and there was SHARE, so there were ways of publishing software. Argonne Laboratory already had something called the Code Center which was to take and make software available to Department of Energy laboratories, and so what was original here was that these were new algorithms, that weren't widely available in any form, and that the emphasis here on making them portable across different machines. At the time there was an IBM library, and a CDC library, and the Univac library, and they were all different and the machine parameters were different, and software that would run on the IBM machine wouldn't run on the Univac machine and visa versa. What we did was make one library that could be used across all the machines.

HAIGH: And how was that achieved?

MOLER: By paying careful attention to that objective. For EISPACK there were actual different versions. There was an IBM version and there was a CDC version, because the FORTRANS were incompatible.

HAIGH: There wasn't a standard subset you could use?

MOLER: Not enough, and also machine parameters, like the floating point precision, and so on, were different. But all the software was developed from one common source and the variants for different machines were produced semi-automatically. With the second project, with LINPACK, we were able to achieve universal FORTRAN. The algorithms themselves didn't have that much machine dependence, to solve systems of linear equations you don't need to know the accuracy of the accuracy of the floating point arithmetic, for example, so we don't need that parameter there, and we knew about how to cope with the different kinds of FORTRAN by the time we got to LINPACK. Anyway, this was a project centered at Argonne Laboratory, and most of the people that worked on it were full time employees at Argonne –

HAIGH: Can you name them?

MOLER: Well, who are the authors of the manual there?

HAIGH: Is that under books or papers?

MOLER: Probably under books.

HAIGH: Books… oh here we are. We have *Matrix Eigensystem Routines -- EISPACK User's Guide*, First Edition 1974, with B.T. Smith –

MOLER: Right, Brian Smith is an Argonne employee.

HAIGH: J.M. Boyle –

MOLER: Boyle is another Argonne employee.

HAIGH: J.J. Dongarra.

MOLER: Dongarra is Argonne, he was just around as a summer kid at first, became a full time Argonne employee, but is now a famous professor at the University of Tennessee.

HAIGH: I'll be interviewing him next month.

MOLER: B.S. Garbow.

MOLER: Burt Garbow, another Argonne employee.

HAIGH: Y. Ikebe –

MOLER: Ikebe. That's a strange story. He was a young professor at the University of Texas, and his sponsor there was a famous old timer named David Young, and Young wanted to be in on this EISPACK project, and have Ikebe be part of it. So Ikebe was part of the grant proposal and the University of Texas got some money and he did his own version of things but it was never part of the actual software.

HAIGH: And V.C. Klema.

MOLER: Virginia Klema was an Argonne employee who had done the first version of stuff. I had mentioned her. She was the one who didn't do the very…. In FORTRAN you could get rid of all the blanks, you didn't need the blanks cause the compiler didn't look at them, and she took advantage of that, for some reason she wanted to have as few characters as possible in her code so it was all squished together with no blanks. It was unreadable by humans but it worked. So she was just got interested in writing programs that machines could run, she wasn't interested in making portable maintainable software. Different objective, but she's listed as one of the authors because she, in a sense, started it all.

HAIGH: In what sense did she start it all?

MOLER: She wrote the first versions of things.

HAIGH: Okay. And whose idea was it embark on the broader EISPACK project?

MOLER: I think probably I'd give credit to Cowell for that, Wayne Cowell, as an administrator at Argonne, he's not one of the technical authors of the thing.

HAIGH: So the date here on the book is 1974.

MOLER: Oh, yeah, but the software was before that. Okay 1974, no that's pretty good, the software was maybe available a couple of years before that. With that many authors on the book we had a hell of time getting it written, getting everybody to agree on it.

HAIGH: Well, was the first edition released while you were still at Michigan?

MOLER: Probably. That would have been 1972, right.

HAIGH: Were earlier versions used internally, to debug them?

MOLER: Not only to debug. An important part of both EISPACK and LINPACK was that we had a rigorous process for what we called certification. We had a number of test sites and we sent these, university and Government lab computer centers, and we sent the codes to these test sites along with test suites, and got back reports from them, and actually had meetings at Argonne with representatives from the test sites. So a big part of the project, was this, well not so much debugging but quality testing and portability testing.

HAIGH: To your knowledge had that been done –

MOLER: Not on this scale, not this rigorously. Elsewhere, you've heard about, let me let you finish your question.

HAIGH: …with other academic or national lab produced software?

MOLER: No, I don't think so.

HAIGH: So were you influenced at all by the emergence of software products on the commercial side, the idea that there could be a piece of software that would be supported and tested and used across a large number of sites?

MOLER: Well, at this point the commercial products came from the computer manufacturers, we didn't yet have NAG [Numerical Algorithms Group] or IMSL [International Mathematical and Statistical Library] libraries, they're coming later –

HAIGH: Yeah, maybe not from the scientific side, but on the data processing side, for example, the file management software, report writers, code library tools, of that era were the beginning of software products.

MOLER: But they weren't portable, they were jut IBM, or just UNIVAC.

HAIGH: Right. So, being in FORTRAN, and being portable, was novel. But it sounds also that the whole idea of expanded, more rigorously developed, more deliberately tested beyond just being portable, that in itself was a new thing –

MOLER: That's right. And the proposals to do the project emphasized that aspect of it.

HAIGH: And who were the proposals submitted to?

MOLER: I don't remember whether NSF was involved in EISPACK, they were certainly involved in LINPACK, and the Department of Energy was involved in both. I didn't have anything to do with the funding or even much to do with writing the proposals, that was done by the people at Argonne, but it was primarily a way to get separate funding for the applied math group at Argonne, which was called MCS, Math and Computer Science, so it was the computer center but it also had this kind of research and development arm as well.

HAIGH: Right. So this would have been the first time that that group had a research mandate in some right?

MOLER: I don't know… one of the first times.

HAIGH: What exactly was your role in the project?

MOLER: It was different on EISPACK than on LINPACK. In LINPACK, which came later, we were writing codes from scratch, and I was the primary author of some of the routines in LINPACK. In EISPACK, we were deliberately not writing new stuff, we were just taking Wilkinson's stuff and translating it to FORTRAN, and checking out, testing it and certifying it. So I didn't write any of EISPACK. Wilkinson did, and then Klema, and Garbow and Boyle translated it to FORTRAN, but then I checked it out. I mean I would get copies of it before anybody else and I would run it. I was at different places, I was at Michigan, and I also spent a sabbatical year at Stanford, and I eventually went to the University of New Mexico, so wherever I was I was working with this software, and then I'd go to Argonne in the summers, sometimes spent the whole summer there.

HAIGH: So just to summarize and make sure that we've got the main points down: EISPACK was important because it was written in FORTRAN, because you took great pains for portability, because it incorporated Wilkinson's new methods –

MOLER: Yes -

HAIGH: And because it was deliberately tested, you know, went through beta stages, with user involvement on a fairly wide scale before it was officially released?

MOLER: Right. And there was a conscious effort to tell the world about this free, portable software.

HAIGH: How was that done?

MOLER: How did we do it? I'm going to give us credit… in a sense this was one of the first, today we'd call it public domain software, but I don't think we used that term back then. I don't know how we advertised it, it was a concern for us, as to how to tell people about it, notices in magazines and press releases. There wasn't any Internet yet.

HAIGH: Were the people you were trying to reach computer center staff or end users in scientific disciplines?

MOLER: Both, both. At a given university or automobile company, or something like that there, there may have been, in the central computer center, someone responsible for math software, and they would acquire this software and make it available on the central mainframe and tell the rest of the university about it. Or, some particular researcher would need the eigenvalue routine, he'd get the software, and then it would become known that so-and-so in the meteorology department has these eigenvalue routines and….

HAIGH: Okay. So imagine that you've got over that first hurdle and there's someone who thinks "I need this EISPACK." How would then get it?

MOLER: You could get it from the Argonne Code Center, so that was one way of doing it. The Argonne Code Center was this organization that was already distributing software to other Department of Energy laboratories, and so on, and they had a certain procedure. If you weren't a the Department of Energy laboratory, you would have to pay a fairly hefty price for the service. We had to make it available through the Code Center because we were part of Argonne, we didn't particularly get along with the Code Center because of their policies. How else did you get it? I'm forgetting, IMSL, when did IMSL come into existence?

HAIGH: I believe in the very early 1970s.

MOLER: Yeah. So I think maybe you could get it from IMSL, separate from their commercial library.

HAIGH: They would just charging for the reproduction?

MOLER: Something like that. And then both IMSL and NAG then took EISPACK, adapted it to their own libraries, changed the name of the routines to fit in with their scheme, changed the calling sequences, and so on, so these became part of the commercial libraries.

HAIGH: So how was the copyright done?

MOLER: I don't think the software was copyrightable. This was a translation project. The original Algol had been published by Springer-Verlag as a book by Wilkinson and Reinsh, called the *Handbook for Linear Algebra*, and we regarded EISPACK as a translation to FORTRAN of Wilkinson's Algol. You'll see that those books there are published by Springer, because they're translations of the Algol, it was a funny attitude. I don't think there's anything on the software about who it belongs to. There's a statement in the manual about that, but not on the software itself.

HAIGH: So if someone in one computer center liked the code they could just copy it for their friend at another computer center

MOLER: And we hoped they'd do that, we wanted that done.

HAIGH: So in that sense it was an ancestor of open source software?

MOLER: Yes, yes.

HAIGH: Software, actually I believe that prior to something like 1976, you actually had to take steps to copyright something, after that, copy right was just created automatically, everything was copyrighted. Before that, I think that if you didn't register it and include the copy right symbol it really did just pass into the public domain.

MOLER: I think that's definitely the case here. I don't think that the source code has anything abut copy right on it, and at the time you didn't copyright software, you put copyright on books.

HAIGH: I think that by that the legal principle had been established that you could if you wanted to. I know during, until the mid 1960s, people believed that it was, that copy right didn't apply to computer software, it was based on a legal precedent that might have played a piano roll –

MOLER: Right –

HAIGH: Because that's what punch cards seemed closest to. Which brings us to the topic, physically they got –

MOLER: Magnetic source code –

HAIGH: Source code on a magnetic tape?

MOLER: Yes, yes.

HAIGH: It didn’t come on punch cards?

MOLER: It was too big for that, I think.

HAIGH: And the first version, do you know about how many functions and how many lines of code?

MOLER: The version of EISPACK available from Netlib today has 79 functions and 11,601 lines of code. These numbers probably haven't changed since the late ' 70s.

MOLER: Yeah. Let’s look that, let’s do that. I was going to show you something here, this is the talk at SIAM (and this is not Power Point this is MATLAB doing the talk). There’s something I wanted to show you here… this is early history, we can go back over this if you want to, we probably should, actually. There’s John Rice, so there was a thing at Argonne, 1970 IMSL started, so there was a thing at Argonne called the National Activity to Test Software, and that was the project that became EISPACK, that was the first name of it, in 1970. IMSL started in 1970, so here’s the Wilkinson and Reinsch book, okay -

HAIGH: Actually, for the transcription can you read the title of the book.

MOLER: Oh, the Wilkinson and Reinsch book is called *The Handbook for Automatic Computation, Volume 2, Linear Algebra*. Numerische Mathematik and its editors had the idea that they were going to publish a whole set of volumes of algorithms, for all kinds of computation, and they had in mind this whole series. That series was the handbook for automatic computation. I can’t imagine what Volume 1 was. Volume 2 is linear algebra, it’s by Wilkinson, J.H. Wilkinson and Christian Reinsch, and that’s the book we’re talking about, there were never any volumes beyond that, nobody else ever wrote the rest of the series. There’s Wilkinson, so 1972 was the first release of EISPACK (it’s an abbreviation for Eigen System Package). There were addons to it, EISPACK 2 in 1976, and EISPACK 3 in 1977, and here’s the part I was making: EISPACK was funded to do research in translation certification and distribution of mathematical software, not in producing the software itself. And this was something we put in the manual, in the documentation, this software is certified, in the sense that reports of poor or incorrect performance would gain immediate attention from the developers, that’s the closest thing we had to a license or the GNU copyleft, or anything like that, we just made this statement.

HAIGH: In this sense certified seems to be a synonym for supported?

MOLER: Yes, that’s right. It doesn’t say there that it was carefully tested and so on.

HAIGH: So in that sense, the onus is placed on the users to do the testing?

MOLER: Well, we’d already done the testing.

HAIGH: So, a hypothetical computer center user hears about the package, thinks it would be a good thing to have, and can either order it from the Argonne Code Center and pay a hefty fee, or purchase it from –

MOLER: IMSL, I think –

HAIGH: Yeah, or they can just go to a friend and run a copy of the tape?

MOLER: Yes.

HAIGH: And then documentation takes the form of this book from Springer?

MOLER: Couple of books, yeah, yes.

HAIGH: Couple of books from Springer, and if they run it and they find something that's wrong with it, they report this to the development team at Argonne?

MOLER: That's correct.

HAIGH: And Argonne fixes it?

MOLER: Yeah, but I'm not sure there was any of that, there weren't, we didn't have any bugs.

HAIGH: Okay. So the statements about fixing poor or incorrect performance, that's conceptually true but in practice the bugs were stamped out in the testing?

MOLER: Right, exactly, yes, yes.

HAIGH: Now how about if at the user site they tweak the source code, make some changes, they think it's better, did that ever happen?

MOLER: I don't know, we never heard about it. They can do what they want with it, but there was certainly no mechanism for them feeding that back.

HAIGH: So everything that was in the second version of EISPACK was contributed from Argonne, there was nothing from the broader community?

MOLER: That's right, that's right, yes. I forget exactly what was the second version and what was the third version, but the time we got to the third version Stewart and I had developed the QZ algorithm for doing generalized eigenvalue problems, eigenvalue problems were the form Ax equals lambda Bx, involving two matrices. Here's where I did write some software. I wrote some FORTRAN software in the EISPACK style for doing this generalized problem, and then Garbow massaged it because I didn't quite have it in the EISPACK style, and that became part of EISPACK two or three, so there were a couple of examples of that that in the later versions of EISPACK were things that didn't come from Wilkinson and Reinsch handbook –

HAIGH: And what did the EISPACK style consist of? Was it things like indentation?

MOLER: Indentation, calling sequence, the names of…. In LINPACK we had a very definite naming convention for the routines so you could look at the name of the routine and if you knew the initials you could figure out what it did. EISPACK that wasn't true, EISPACK was a translation from Wilkinson so we used the names Wilkinson used. Wilkinson's names tell you what the method is rather than what problem it solves. So one of the names is Tred 1, you have no idea what that's for, even if you know it's a matrix eigenvalue routine, you don't know what part of that computation it does.

HAIGH: And at this point, was it an innovation to have standards for naming an indentation on a scientific project or was this kind of thing widely accepted?

MOLER: I'm not sure, we were more careful about it and more fanatical about it than people had been in the past. Certainly NAG and IMSL, well, no, I mean, IBM libraries that had preceded this had standards like that, they had common coding conventions.

HAIGH: Okay. Actually I want something about the period where structured programming becomes a buzzword.

MOLER: Where structured programming –

HAIGH: Stuctured programming, and "GOTO Considered Harmful", it seems to be a period of what people were studying for the first time thinking about code as something that has style, that is read.

[End of Tape 2, Side A] [Start of Tape 2, Side B]

MOLER: Well, EISPACK was a translation from Wilkinson, so it would follow as much as you could in FORTRAN whatever style Wilkinson and his colleagues used. With LINPACK, this business about structured programming, GOTO considered harmful, and so on, was very much in the air, and the LINPACK code was actually produced by another program, a program called TAMPR, which was an acronym for something about transforming programs. We would write FORTRAN in a kind of stylized form, and then TAMPR would actually go do the indenting, and so on, and TAMPR would figure out the scope of loops and it would only use GOTOs and statement labels in a certain stylized way, when they had to be used. So it was structured FORTRAN that TAMPR would produce, as much as that was possible, and that there was this guy Boyle was behind that. There were four versions, single precision, double precision, single precision complex, double precision complex. We only wrote one version and TAMPR would produce the four different types, automatically, and that was a nontrivial transformation because there were certain constructions in the complex codes that were unnecessary in the real codes, so it wasn't jut a text transformation. TAMPR parsed the codes, understood what they were doing, and then wrote new versions in FORTRAN according to its rules for structured FORTRAN.

HAIGH: And that's where a lot of the portability has come from?

MOLER: That's right, yeah.

HAIGH: Do you know of any precedents for that?

MOLER: No, that was brand new stuff. That was Boyle and a guy named Dritz, Ken Dritz at Argonne, who was responsible for that, you won't see Dritz's name as an author of EISPACK, but he was an author of TAMPR.

HAIGH: And do you know if there are publications describing TAMPR?

MOLER: I think there are.

HAIGH: Okay, I'll search for those later. Maybe we should transition formally talking EISPACK to talking about LINPACK, which a lot of your answers(?) are related to -

MOLER: Yes –

HAIGH: LINPACK was produced a couple of years after the first version of EISPACK.

MOLER: Right –

HAIGH: At that time was there a sense that EISPACK had been a big success?

MOLER: Yes, precisely. EISPACK was such a big success but it was only half the story, it was only half of a matrix computation library, we hadn't done the first half, it was the second half. So the project was to go back and do the second half.

HAIGH: And you said that these are routines that computer centers already had?

MOLER: They already had in some form or other, but here the idea was to produce a standard, an international standard. We did have a few innovations in terms of algorithms, and we were also here concerned with performance. That hadn't been such a big concern with EISPACK, but here we were trying to make the routines run as fast as they could, as well as being correct and portable. This is where TAMPR was used, TAMPR was not used in EISPACK.

HAIGH: Okay, so it's portable, and performance is the new thing?

MOLER: Right.

HAIGH: And by and large these are the same methods that computing centers would already have some kind of implementation of?

MOLER: That's right, that's correct.

HAIGH: So how did you go about improving performance?

MOLER: Through the use of the BLAS, Basic Linear Algebra Subroutines. The idea was that the inner loops of the LINPACK routines were done by subroutines, which may or may not be written in FORTRAN. There were FORTRAN versions of them that referenced standard versions, but if for a particular machine you, or the computer

manufacturer, wanted to provide optimized BLAS, they could be used and that would determine the performance of the overall library. FORTRAN compilers got to be good enough that they could produce fast code out of the reference BLAS, but when we first started it that wasn't true.

HAIGH: So what you were doing was really introducing an additional abstraction layer.

MOLER: Exactly.

HAIGH: …compartmentalizing the things that would need to be optimized to a machine.

MOLER: That's right. And that idea has carried over to today in LAPACK, that's a big part of the architecture of LAPACK.

HAIGH: And that was another unprecedented feature, was it?

MOLER: I think so, yes. Well, people had been writing the inner loops in assembly language for a long time –

HAIGH: And originally they'd have written everything in assembly language.

MOLER: That's right, that's right. So the idea here was to have a standardized set of those routines, that did the things that were needed throughout numerical linear algebra. So one of the BLAS is a dot product, the inner product of two vectors.

HAIGH: Alright. And in practical terms this would be some libraries that were linked when you compiled?

MOLER: That's right, yes, that's right.

HAIGH: Okay. So, were there other things that were going to improve performance or is that the main one?

MOLER: That's the main thing to do performance. Here's another thing on the performance story, let me show you this, this is out of the LINPACK manual. We took these routines, part of the testing process was to take all these routines and send them to our test centers, and this is a list of our test centers. The National Center for Atmospheric Research, Los Alamos; University of Illinois, Urbana; Bell Labs; Yale; there's about a little over 20 Government labs and universities there. We asked them to time this one particular routine, that this is a routine that solves the system of simultaneous linear equations, it's actually back to the same thing we talked about in Forsythe and Moler, and the various places ran them on their machine and returned results to us about how fast they ran, and we published this as an appendix in the back of the LINPACK manual. That became known as the LINPACK Benchmark, and today LINPACK is more famous as a benchmark than it is as a library of subroutines.

HAIGH: Yes, I've noticed that. Every time they report the new super computer on slash dot it's always LINPACK.

MOLER: That's right. Jack Dongarra has continued to manage this, it's now gone from way beyond this list, to a report that's dozens of pages long, listing all the machines. This was a 100 by 100 matrix which is all we could expect the computer centers to do at the time. In fact, this computer at Yale couldn't handle a 100 by 100, it was too big, the biggest it could handle was 75 by 75, and so we multiplied by four-thirds cubed to get Yale's number compatible with the rest of them. This is the original LINPACK Benchmark, that was an appendix in the LINPACK manual, and today the software's changed, LINPACK library has been superseded, but the name LINPACK Benchmark still survives as a way of testing super computers.

HAIGH: Which is a tribute to its portability.

MOLER: Yes. And to the universality of the problem, the fact that the problem itself is a thing you've got to write any way.

HAIGH: Okay. So, performance, portability, BLAS….

MOLER: Here's an interesting list, with the use of Pack as a suffix… so EISPACK, LINPACK, but then they become FUNPACK, SPARSPACK, ROSEPACK, all these other packs, one of the guys, Allen George behind SPARSPACK, actually wrote me a letter and asked me was it okay to call his SPARSPACK, because we'd been calling things PACKS. So, that's another thing we contributed, is the name PACK to the industry.

HAIGH: So, between them, would LINPACK and EISPACK deal with the range of topics that you discussed in the Forsythe –

MOLER: No, no, it's far beyond that. Forsythe and Moler is just about solving systems of linear equations. There's really only one subroutine in that book and one algorithm, well there are two, but this is just, there are in LINPACK, there are 11 or 12 chapters, which are 11 or 12 different problems, and Forsythe and Moler is just one of them.

HAIGH: That's one chapter. So think about the bigger picture implications of that, does that mean that by the 1970s the kinds of students who previously would have read this book and learned how to write their own routine would instead just pick up LINPACK and EISPACK and use one of the subroutines and not need to worry about what was happening under the hood?

MOLER: Well, that's a quandary and something we still debate today: what do you want to teach students, do you want to teach them what's under the hood or not? The trig function signs and co-signs, and so on, we don't teach students how those are computed, though sometimes they ask. I think that we should still teach them how to solve systems of simultaneous linear equations, and, in fact, the exposition of the algorithm for that is a main topic in the book I've just written. But I think you ought to take the attitude that students should understand the ideas behind solving systems of simultaneous linear equations, they're simple enough that you can understand them. The subtleties associated with are complicated algorithms for computing eigenvalues, and so on, most students don't need to know that, then they'll just call it the subroutine.

HAIGH: And did LINPACK do anything to help users decide which method they should use?

MOLER: There are different classes of problems. So, your matrix is either symmetric or it isn't, it's a band matrix or it isn't, and we have one subroutine for each of those classes of problems. So, you could use the general code on a symmetric matrix, and ignore the symmetry, but that would cost you a factor of two in both time and storage, but there aren't multiple methods for the same problem that you have to choose among.

HAIGH: So, this isn't an area where you might have some methods that would give a good approximation, and others that would fail to converge, for a given problem?

MOLER: No, no. That happens in other areas in numerical, quadrature, numerical evaluation of integrals, numerical solution of ordinary differential equations, and so on, you have several methods for solving the same problem, and you have to do some experiments maybe to decide which one is best for your particular problem. For matrix computation that isn't true.

HAIGH: So, everything in LINPACK and EISPACK there's one method and it's going to work, whatever you feed into it?

MOLER: Yes, if your matrix satisfies the criteria for that method.

HAIGH: And that's a cut and dry thing?

MOLER: Yes, that's right. That's oversimplified, so there are routines that will find all the eigenvalues of a matrix, and then there are routines that will find a few eigenvalues of a big matrix, now where does the transition from that one to the other take place, sometimes that's a matter of judgment, but….

HAIGH: Okay. So, whereas your earlier research had been about upper and lower bounds…

MOLER: Oh, its a long way from that.

HAIGH: …in these cases you don't get bounds, you get the real value.

MOLER: Yes, yes, right. Yes, there's no approximations going on here.

HAIGH: That clarifies it for me then. Presumably LINPACK had the same distribution system as EISPACK?

MOLER: Yeah, so when did Netlib show up? Netlib wasn't until 1985, so by the time Netlib appeared and the internet occurred you get all this stuff over the Internet. But before that I think it was the same distribution system.

HAIGH: And do you have a sense of whether in practice most centers would have copied it from another center or would have actually gone to Argonne and paid for a tape?

MOLER: I don't think very many places got it from the Code Center, I think they probably got it from IMSL.

HAIGH: So IMSL took the routines that were there, changed them, renamed them –

MOLER: No, No. IMSL did two things. They did that, IMSL incorporated these things into their own library, but also, as a service to the community, they distributed the original library as well for a small service fee. Thereby competing with themselves.

HAIGH: That's exactly what I was just thinking.

MOLER: IMSL made a good point. They said, "water is free, you pay the water company for the service of bringing it to your house," so LINPACK and EISPACK are not a complete math library, they're only a part. By the time you get differential equations and zero root, polynomial roots and FFTs, and all these other stuff, that's a complete math library, and that's what you get from IMSL and NAG. LINPACK would be one chapter, linear equations would be one chapter, eigenvalues would be another chapter, but then there would be a dozen other chapters, and you could get that whole library in one fell swoop from these organizations.

HAIGH: Right. Do you have a sense of what proportion of computer centers might go the commercial route? Would even a computer center that was using a commercial package also have LINPACK and EISPACK in use for some things?

MOLER: Yeah, probably. So, IMSL and NAG certainly made their companies successful for as long as they were by supplying complete math libraries to the computer centers, both industrial and academic. Even though free products that did a part of the job were available elsewhere.

HAIGH: I see that on your resume there is at least one paper co-authored with Wilkinson. That's 1979.

MOLER: Is that also Dongarra? Oh, no, that's [Alan] Cline, and Stewart, and Wilkinson.

HAIGH: Is that an offshoot of the LINPACK project?

MOLER: Yes. So, we had something that was original in LINPACK was something to estimate the condition number of a matrix, to give you an idea of the accuracy of the result. Stewart and I invented a condition estimator, we wrote it up and it was in an early version of LINPACK. Alan Cline found a counter example that the estimator failed to get right, and Wilkinson provided a fix to the algorithm that got rid by Cline counter example, that was the basis for that four person paper.

HAIGH: So is that true then of any of your other papers from the 70s, that LINPACK and EISPACK are feeding back into your published work?

MOLER: Oh, back into the publications? Well, another one of my most important papers, besides the Fox, Henrici, Moler one, is the paper with Stewart on a generalized matrix eigenvalue problem, that was very much connected with EISPACK.

HAIGH: That's SIAM Journal of Numerical Analysis, 1973.

MOLER: Right. And, most of these published papers are related to that.

HAIGH: OK, so by that point that's been a definite shift from, in your work, from using computers to prove theorems, into publications that are coming from the software?

MOLER: From the software, absolutely. Yes, yes. Almost all, I wouldn't say all of that is.

HAIGH: Okay. And actually the pretext of EISPACK does seem to have produced one conference paper, and that's "A Collaborative Activity to Certify and Disseminate Mathematical Software."

MOLER: Yeah, oh that's right, so –

HAIGH: You got one out of it –

MOLER: Yes.

HAIGH: So back to your career now then. It appears that in 1972 that you moved from Michigan to New Mexico –

MOLER: That's right.

HAIGH: Which would have been a result of looking for a place where you and your wife could work.

MOLER: Exactly.

HAIGH: So how was the transition, was New Mexico a different kind of place from Michigan?

MOLER: Yeah, yeah, that's right. Going from being a small fish in a big pond to being a big fish in a small pond.

HAIGH: What kind of work was going on at New Mexico at that point?

MOLER: Nothing. Well… I went to the math department, there was, you know, a bunch of mathematicians there working on various kind of things. I didn't have much to do with any of it. I was a replacement for Larry Shampine. Larry Shampine had been here in the math department and went to work at Sandia, and I took over his place, so my collaboration with people in New Mexico was more to do with courses and pedagogical things than with research. A colleague there named Stan Steinberg and I wrote most of a textbook that was never published, we never finished it, that was a friendship and a

collaboration. I had some things to do with Los Alamos and Sandia, I spent some time as a consultant there. I was a consultant of the group at Los Alamos that maintained the math library for the computer center at Los Alamos, so that was math software stuff, no publications resulted from that. You won't see many publications coming out of the years in New Mexico.

HAIGH: Because you were working more on software?

MOLER: Well, right, and I wasn't doing research so much any more. I became a department chairman there. There is one interesting thing that won't show up very much on the resume, if at all, I did some consulting for computer manufacturers, a company, does it say I was a consultant for Convex there?

HAIGH: Yes, it dates that 1983 – 1984.

MOLER: So Convex was a computer, building a vector, sort of a cross between a VAX and a Cray, and they wanted consulting on their math library so I worked with them. And then the important thing was IBM. IBM was deciding to build a vector computer, and they got me as a consultant to advise them about the development of a vectorized math library.

HAIGH: So, would that have been "Thomas J. Watson Research Center, 1982-1983"?

MOLER: Yes, and more than that. So that's where I was a consultant, what it doesn't show here is where I was getting research support.

HAIGH: Was that the same project?

MOLER: It went on beyond that. So for several years at New Mexico I was getting research funding not from the government any more but from IBM, to work on math software. Originally for their vector computer, but then for the PC when it first came out.

HAIGH: Actually, while we're talking about consulting positions, the resume also shows you with IMSL as a consultant from 1976 to 1984.

MOLER: Was it that long? Yeah, okay. So when IMSL first started they asked me to be a consultant, to be on their Board, I was listed on their Board for all that time. I did a little bit of consulting with them but not much, and then finally when we started MathWorks I decided it was time to resign my position at IMSL, it was sort of competitive.

HAIGH: So in practice that didn't amount to much?

MOLER: Didn't amount to much.

HAIGH: Alright. So, in New Mexico, did you also have a higher teaching load?

MOLER: No.

HAIGH: Okay. But there was a shift of emphasis from research to administration and departmental things?

MOLER: Yes, that's right, yes.

HAIGH: And, now the other thing your resume shows is that you were a full professor from 1974 to 1982 in the mathematics department, and it also shows you as professor and chairman at the computer science department…

MOLER: I changed jobs –

HAIGH: ...from 1980 to 1982 –

MOLER: Yes

HAIGH: So there's a two year overlap there.

MOLER: I went there in 1972, I left there at the end of 1984, so there was eight years as a professor of math, and four and half years as the chairman of the computer science department.

HAIGH: How did the transition happen, was the computer science department a new thing?

MOLER: No. The computer science department had been there since the very beginning. When we first went there, my then wife became a professor in the computer science department. We got divorced soon after that and, partly because of that, I didn't have anything formal to do with the computer science department. The founding chairman of the computer science department was a guy Don Morrison, and you'll see a couple of papers there with Morrison about interesting little things. When it came time for him to step down as chairman there was a question about who was going to be the next chairman, and it was more complicated than this because we went through a couple of intermediaries, but I essentially replaced Morrison as the chairman of the computer science department. So, I changed jobs, I went from being a math professor to being a computer science professor, and changed from the College of Arts and Sciences to College of Engineering, and changed from having no, you know, no more administrative duties than a standard professor, to being a department chairman.

HAIGH: And how did you find the experience?

MOLER: The first three years was interesting, new challenges, new people, new things to do, new set of colleagues, and then the last couple years I got sort of bored with it, and really didn't do such a good job, and the department administrative assistant really ran the department. The first three years we did lots of recruiting and I spent a lot of time hiring new people, and hired some people that are still there, essentially built the department, the last couple years we didn't have any money to do any hiring so there was less of that activity.

HAIGH: That's interesting. Did the experience in institution building and management relate in any way to the interests that you developed later on in terms of your company?

MOLER: On the contrary. As we talk about the development of MathWorks, it's going to be important part that I didn't have much to do with its development. One of the reasons it's successful is because I haven't anything to do with it. I joke, but it's only half a joke, that my experiences as a department chairman made me quit academia. I was a department chairman and then I got this offer to go to work for Intel, on parallel computers up in Beaverton, and I left in the middle of the year to go that. My future at the University of New Mexico would have been more and more academic administration and I did not look forward to that. I wasn't interested in that, and I wasn't good at it.

HAIGH: But rather than just return to being a professor, you left academia.

MOLER: Yeah. Well, so these were new opportunities too, new challenges, I'd sort of outgrown New Mexico.

HAIGH: Right. And what were you working on at Intel?

MOLER: The first commercial parallel computer, or one of the first ones. It was a entrepreneurial operation at Intel. Intel had seen their employees break off and start new companies, and be successful at it, so they decided they were going to sponsor the creation of a couple semi-autonomous internal companies, and one of those was this operation to build a parallel computer out of Intel chips. I had actually been a consultant to them a year or two before about parallel computing, and so they called me up and said did I want to go to Oregon, and participate in this which was essentially a startup, and I did.

HAIGH: Would that be the operation that later produced the Paragon super computer?

MOLER: I think so. Was that the name of it, what was it, Paragon? Yeah, yeah it is.

HAIGH: Early nineties –

MOLER: Yes, right. So these were parallel computers, they're taking a whole bunch of Intel chips and making single board computers out of them and putting them all in a cabinet, and getting as many in one cabinet as you can jam in there, and then getting them all communicating with each other.

HAIGH: Right. And when you joined in 1985, did they have a prototype working?

MOLER: No, I was one of the first people there to start building up those things.

HAIGH: So your job title was Manager: Applications Research.

MOLER: Yeah.

HAIGH: Was the lack of working hardware a handicap in researching applications?

MOLER: No, we knew exactly what we were building, and we were getting ready for it. So, I was writing programs for this new machine even before I got there, even though it didn't exist. I knew what it was going to be like, and we had simulators for it as well.

HAIGH: At that point, were there already widely published and usable algorithms for massively parallel computers, or were you designing the algorithms as well as the implementations?

MOLER: We were designing the algorithms as well as the implementations. So, the story I like to tell, is this basic algorithm that we started out by talking about from Forsythe and Moler, which became the LINPACK Benchmark, the question is: what's the parallel version of that? And I wrote that parallel version in my head as I drove across the Nevada desert on my way from New Mexico to Oregon. I didn't have a computer and I thought up the algorithm about how to parallelize it, when I was nowhere near a computer. But, by the time I got to my new job I had an idea about what I was going to do.

HAIGH: Did the research community rally around parallel algorithms? Was that a direction that the mainstream of the numerical analysis software community was taking at that point, or did that involve collaborating with a new group of people?

MOLER: Both. Developing parallel algorithms has certainly been, I'm not sure if it's the mainstream, but it's been one aspect of numerical analysis research since that time. Then there are people who work on parallel algorithms for things that aren't numeric and you want to talk to those people as well. My resume shows a couple of conference proceedings about the parallel stuff, "Matrix Computation on Distributed Memory Multi-processors," and this was the first conference on hyper cubes.

HAIGH: 1986.

MOLER: Right.

HAIGH: And then we have, "Parallel Processing and Medium Scale Multi-processors," 1989.

MOLER: Another conference proceedings.

HAIGH: Alright. So what were the main things that you achieved during your time at Intel?

MOLER: I learned about the difficulties of programming parallel computers.

HAIGH: So it was harder than you expected?

MOLER: Harder than anybody expected. We're still facing the same problems today. I'm about to go give a talk and workshop on future super computers, about the possibility of a parallel MATLAB, and it's still the same problems we were facing 20 years ago. How to talk about distributed memory? The programs are so complicated that the mathematics is completely lost in all the parallel management, and the question, is there a reasonable

way to make efficient parallel programs that people can read and understand, and that's something we haven't figured out how to do in 20 years.

HAIGH: So, for this project then the application side was really where the problems were?

MOLER: Yeah.

HAIGH: They could build this computer, and connect these processors together, but the problem was –

MOLER: How to program it, how to program it. Yes, that's right.

HAIGH: And as well as learning the scale of the challenge, did you achieve anything in terms of an algorithm or any kind of solution –

MOLER: One of the things I did do at that time was a lot of lecturing, so I was sort of one of the principle spokesmen for this project, and I gave lots of talks and seminars, and so on, in which I introduced people to parallel computing.

HAIGH: So you were what they later called an evangelist?

MOLER: Yes.

HAIGH: And how did you manage a job of this kind, did you have a significant size team?

MOLER: Yeah, I did. I was in charge of a group of six or eight people there, yes, and we were writing the math library and writing demos, training people, and promoting this thing.

HAIGH: Did you enjoy the role of a hands-on manager?

MOLER: I didn't do much with it. These were people we'd hired that were academic types, like myself, and it was a little bit like being a department chairman, where these guys are off doing their own thing.

HAIGH: So, it was much more in the style of an academic research lab than a development effort?

MOLER: It shouldn't have been. So this was a failure on my part to do more managing. Another thing a manager does is promote the group, or explain the group, within the organization. There's all these other people in the organization, you know, saying, "What are those math guys doing? What are they doing for us, we all have these difficult jobs, those guys just seem to be running around to conferences, you know, writing funny little programs," and I was not good at explaining the relevance of that to everybody else.

HAIGH: Right. Acting as a buffer between the lab and the rest of the organization.

MOLER: So you've got to be an evangelist, not only to the rest of the world, but within the organization.

HAIGH: Were any of the people that came to work at the lab people you had worked with before in the academic world?

MOLER: A couple of them were people I knew from the academic world. I hadn't worked with them before, but I hired a couple people like that.

HAIGH: And you approached that the same way that you would an academic hire?

MOLER: To some extent yeah, yeah, right. One of the best guys I hired was somebody I didn't know before, but he was an artificial intelligence guy from Portland State, or something like that. He was really anxious to get out of Portland State, and happy to come to work for us, and he turned out to be a bright innovative guy, and he's still an important guy at Intel today. There were two like that.

HAIGH: Okay, do you want to name them?

MOLER: Joe Brandenburg and David Scott.

HAIGH: So you were there for two years –

MOLER: So, one of those joint papers on the proceedings is with David Scott.

HAIGH: Okay. I'll make sure the resume goes to the transcriber and accompanies this into the archive. Now why did you leave in 1987 and go to Ardent Computers?

MOLER: It was a more exciting, and possibly higher paying, job. So, after two and a half years at Intel, the machine was not a big success, and we weren't selling very many of them, the entrepreneurial experience within Intel I'd say was a failure. It wasn't a startup, Intel was still in there. I would never consider working for a big organization, like Intel itself, and they were still in there meddling with this thing. And then these guys in Silicon Valley were doing a real startup and they seemed to understand the industry far better than the Intel guys had, they'd had experience with startups before, there were much brighter people working at this Ardent organization than the guys at Intel.

HAIGH: Were you on the research side or the management?

MOLER: Both.

[End of Tape 2, Side B] [Start of Tape 3, Side A]

MOLER: If the staff that was assembled at Ardent had been an academic computer engineering department at a university someplace, that would have been one of the top departments in the world, and that wasn't true of the staff at Intel.

HAIGH: Was your role roughly the same as it had been at Intel?

MOLER: Roughly the same thing. I was in charge of a group that was building a math library, building demos. I did some proselytizing though I wasn't the only one here. Hired some people. I did benchmarks. a lot of people before they bought this machine wanted to have us run codes on their machine to see how fast it was on their particular application, and my group did that.

HAIGH: And at the time you joined in 1987, did they have a working computer?

MOLER: No, no.

HAIGH: But that had arrived by 1989?

MOLER: Yes.

HAIGH: Were you their first head of scientific software?

MOLER: Yes, right.

HAIGH: So again this was something you were building from the ground up?

MOLER: That's correct, yes.

HAIGH: Anyone there that you think is particularly important?

MOLER: Several of the people there I still have contact with them, one is Steve C. Johnson, who was one of the original Unix guys at Bell labs, wrote the portable C compiler, and YACC, and LAX, they hired him to be in charge of software at Ardent, and he became a good friend, and he now works at MathWorks. He's now in charge of our compiler effort at MathWorks.

HAIGH: And in the startup environment, did your group work closely with the people doing the hardware and operating systems?

MOLER: Absolutely, there were just a few of us in each group and we had a lot of contact with those guys. The LINPACK Benchmark was one of the ways we had of measuring the speed of this machine. In fact, the stated goal of this machine, in our original advertising and prospectus, and so on, was to make a machine that would run the LINPACK Benchmark at six megaflops, that's trivial today, but that was a big thing back then, to hit six megaflops.

HAIGH: And it succeeded?

MOLER: Yes. I remember very well when we got the first chip back and we put it in the machine and really timed it, I mean we'd been doing all kinds of simulations and extrapolations and we timed it and it was six point three megaflops, and there was a big party when we hit that.

HAIGH: And what was the culture like in general, what was it like to work there?

MOLER: It was a Silicon Valley startup. I mean it was very competitive, lots of egos involved, we had, there was an East coast company called Stellar which was building a similar machine and had a similar amount of talent, and we eventually killed each other. The company was a failure, we didn't sell enough machines, we went belly up, we went through more venture capital than any other computer firm in history before the dotcom fiasco. We went through eight hundred million dollars in venture capital.

HAIGH: That's only slightly less than Webvan.

MOLER: [Laughter] And this was long before that, yes. I mean it was an intense experience, we worked hard. The east coast company was started by a man named Bill Poduska. He's a famous entrepreneur in Massachusetts, he started Prime computer and Apollo computer, he was part of the Massachusetts' miracle, he was sponsor of the Boston ballet. The Boston Globe did a story comparing him with, his company, with the company, with his competitor on the west coast, which was our company. Our company was founded by another successful entrepreneur named Allen Michaels, and the Boston Globe said that we were a Silicon Valley bootcamp. That was an intense experience. There was also a relation with MathWorks, so MathWorks had started already. I wasn't working for MathWorks, I was off on these fantasies with Ardent and Intel, but we had a connection between Ardent and MathWorks. That is, the Ardent machine came with MATLAB on it, there was an agreement between MathWorks and Ardent and we worked with MATLAB on the Ardent machine. When we forged the agreement we had a meeting and I gave the Ardent guys my MathWorks card and the MathWorks guys my Ardent card, cause I was already part of both companies.

HAIGH: I think I saw the Ardent computer described as a desktop super computer.

MOLER: It wasn't desktop, it was a personal super computer.

HAIGH: Personal?

MOLER: Yeah.

HAIGH: So, how is that different from high-end work station?

MOLER: It's more expensive. [Laughter] I mean today it would be called a high end work station. So, this was an attempt in the late eighties to build what today would be called a high end work station.

HAIGH: But instead of something like Sun, where they start off very humble with the commodity hardware and build up, this was a whole new everything from the ground up, best machine possible?

MOLER: Absolutely, absolutely, yes. It took longer than we wanted to build it, it took even longer than that to write the software, the machine ended up costing twice as much as the original prospectus, it had to run on 220 volts. It was the size of a small refrigerator, it was noisy, so you wouldn't actually want it in your office. So, the idea was to make a powerful work station for scientists and engineers that did both graphics and

Cray level arithmetic, Cray level vector computing, and we got part way there but didn't achieve the whole thing.

HAIGH: Was there anything about the architecture that was particularly relevant from a scientific software viewpoint, that made it challenging or exciting?

MOLER: Yeah. It was both vector and parallel. There were four processors and each one was a vector processor, so we had to worry about both vectorizing and parellelizing algorithms.

HAIGH: And did you come up with any major innovations to make that work or was it largely unsolved?

MOLER: I wouldn't call them major, but I made some innovations in vectorization and parallelization that are still…. We are building a parallel MATLAB, and the stuff I learned at Ardent about that kind of computing is having an influence on the thing today. I also learned about computer graphics, I didn't make any innovations or any contributions there, but this was the first machine I worked on, one of the first machines anybody worked on, that had really fancy graphics. So, color tables, and animation, and perspective and lighting, and all that stuff, we were doing on that machine.

HAIGH: So, did that lead to any novel work in terms of user interface or was it just in terms of rendering the output?

MOLER: Yes, we had to worry about user interface. I'd worked on machines with graphics user interfaces, I'd worked on the Sun since it first came out, but this was the first time that I myself had to write any software that had a graphical user interface, or interactive graphics.

HAIGH: So, did that experience feed back into MATLAB?

MOLER: Yeah, it certainly did, yes.

HAIGH: So, we'll follow up on that in tomorrow's interview.

MOLER: Right, yeah.

HAIGH: And then in 1989 you left Ardent?

MOLER: Ardent folded, imploded. It temporarily joined forces with its East Coast competitor named Stellar, the combined company was called Stardent. It didn't last very long. Almost everybody left immediately, I did. MathWorks was already well established by then, and so I went to work for my own company.

HAIGH: So, that was just your cue to go and work full time for MathWorks?

MOLER: Exactly.

HAIGH: Now the other main area that we need to talk about is professional activities: SIAM, the Springer Lecture Notes, and SIGNUM. So, if you want to and you've got enough energy we can talk about that now.

MOLER: Sure, alright. If you do.

HAIGH: So, looking at this now it's clear that you have done quite a lot of work as an editor in different fields.

MOLER: Yes.

HAIGH: It seems the very first thing, chronologically that you did with that was editor at the ACM SIGNUM newsletter, so if you could talk a bit about SIGNUM, what it was at that point, who was involved, how it changed your career?

MOLER: So, ACM has these special interest groups… I've had little to do with ACM, this thing really soured me on ACM. Forsythe had been involved with ACM, Forsythe had been president of ACM. I'd had a little bit to do with ACM before that, they had a special interest group on numerical analysis, a guy named Joe Traub, who was originally at Bell Labs, now a professor at Columbia –

HAIGH: I'm interviewing him next week –

MOLER: Yeah, Traub was one of the originators of SIGNUM, he edited the first couple of newsletters, and then he asked me to take it over. This was a pretty amateur operation, there wasn't much to it, and the ACM Headquarters kept second guessing me. They had policies that they wanted their newsletters to follow, and they kept changing my things, and writing me letters bawling me out for stuff, and I… I gave it up after a while.

HAIGH: So then the newsletter itself wasn't particularly important. How about SIGNUM as a community, was that something that a large number of numerical people were involved with?

MOLER: No. I don't think it's ever been viable. I'm not sure that it's ever done anything. That community's home is in SIAM not ACM, and the numerical analysis group within ACM has always been an anomaly.

HAIGH: You were involved in the early days, and then you were on the SIGNUM Board of Directors from 1971 to 1974?

MOLER: Yeah, same comments.

HAIGH: So, at which point you basically realized that wasn't where the direction was, and it wasn't going to be there?

MOLER: That's right.

HAIGH: Alright. Now then, looking, co-chair ACM lecture series, but the next interesting thing –

MOLER: The real turning point here, was one night, late at night, Ed Block, the guy who started SIAM and ran it for years, called me up and said did I want to be an editor of SIAM Journal of Numerical Analysis?

HAIGH: Yeah, and that's 1969?

MOLER: Yeah. And that was the beginning of my association with SIAM, and it's been going on in various forms ever since. The managing editor of SIAM Numerical Analysis at the time was Tom Hull at Toronto –

HAIGH: Is a managing editor like an editor in chief?

MOLER: Editor in Chief, yes, yes. Also Alston Householder, who you heard me mention earlier, was an editor, and so the paper that I handled went through Hull and Householder originally. Later on, I became Editor in Chief myself.

HAIGH: Yeah, we have 1973 to 1976 as EIC. So, was the SIAM Journal of Numerical Analysis the major journal in the field?

MOLER: No, it wasn't the major journal, it was one of several.

HAIGH: And what would the others have been?

MOLER: Springer published this journal called *Numerische Mathematik*, still does. The American Math Society published something called *Mathematical Tables and Aids to Computation* –

HAIGH: I believe they renamed that, but I can't remember what to.

MOLER: Yes. Those I'd say were the three journals of numerical analysis at the time.

HAIGH: Was there anything distinctive about the SIAM journal?

MOLER: It wasn't German, and it wasn't American Math Society. [Laughter]

HAIGH: Okay, not being German was fairly straight forward. But not being American Math Society, what did that mean in terms of difference in emphasis or editorial policy?

MOLER: The MTAC was always, I think was kind of an oddball out at AMS, even. The editors who ran MTAC, and so on, weren't main line AMS guys, there wasn't like Transactions or Proceedings of the society. The American Math Society really doesn't have any real specialized journals in particular fields, so this is a little bit odd for them. Whereas, SIAM, that's its main business, and numerical analysis is the most important discipline within the SIAM community, I mean far more, if you ask people, ask SIAM members what their interests are, numerical analysis and scientific computing would be

far and away the dominant ones. SIAM has since started a second journal called *The SIAM Journal on Scientific Computing* and I haven't had any editorial functions on that.

HAIGH: Oh, your resumé lists you as an editor from 1979 to 1982.

MOLER: Does it! Well, maybe so, maybe so.

HAIGH: But nothing particularly memorable?

MOLER: Okay, alright. No, nothing particularly memorable.

HAIGH: So, you were called up, you were asked to be editor, you accepted. Did that mean getting involved with a different community of people or were the other editors people that you knew already?

MOLER: Yeah, the editors were all people that I knew already, but you know I got more involved with SIAM. I later came onto the SIAM council. I went to the SIAM office a couple of times, seeing how SIAM worked, seeing how publishing worked, was something new.

HAIGH: And what other kind, these other roles within SIAM did those make you feel part of a different kind of group, did you begin to identify –

MOLER: Yeah, well it solidified the relationship with that community. There's really been two phases to my SIAM activities. There's this early part where I was editor of SIAM Numerical Analysis and I was on the SIAM council, and then there was a period when I didn't have much to do with SIAM for a while, and –

HAIGH: That would be late seventies and eighties?

MOLER: Yeah. And then I got back into SIAM and I became the SIAM vice president at large. When was that?

HAIGH: 1994 to 1996 –

MOLER: 1994 to 1996. So then I was SIAM vice president, and now I'm on the SIAM Board of Trustees, and I've been for the last several years.

HAIGH: And what drew you back into SIAM?

MOLER: Getting elected vice president. I've always gone to SIAM meetings, and so on, but I haven't been involved with SIAM administration, you know, policy making, between the two chunks of close affiliation with SIAM.

HAIGH: And prior to 1969 had you been an active member of SIAM?

MOLER: What's an active member? Go to meetings, yeah -

HAIGH: Go to the annual meeting.

MOLER: Sure, sure, yes.

HAIGH: So, what was SIAM doing right that SIGNUM wasn't?

MOLER: Well lot's of things. SIAM's small, more personal. ACM was like Intel, this big bureaucratic organization. This is a long story, but SIGNUM had local chapters and we would report on the local chapters. They would meet once a month, or something like that, and hear a speaker and we'd list that in the newsletter. So, I wrote that the Los Angeles SIGNUM Chapter had a meeting and Hanson was the speaker. Well, I got bawled out about this because it was not the Los Angeles chapter of SIGNUM, it was the Los Angeles… it had to be worded a different way, and I can't even remember the words, and it had to do with legal responsibility. If the Los Angeles chapter owed some people some money, would the creditor go to SIGNUM or to ACM to collect, and the name of that organization reflected that distinction, and I can't even remember the words, it was a difference between ACM's Los Angeles SIGNUM or Los Angeles SIGNUM ACM, something like that. The fact that grown people spent time writing letters around, worrying about this kind of thing, I found immensely off putting.

HAIGH: So, in reality you don't find that was a substantial risk?

MOLER: No, no.

HAIGH: So, you think that's a culture of centralized control there that's counterproductive?

MOLER: It's busy work, this wasn't even the ACM officers, it was one of the, what we call in SIAM, volunteers. It was an ACM officer but not an ACM employee who wrote me that letter.

HAIGH: Right.

MOLER: Anyway, I don't need to go on and on about ACM, but everything I've had had to do with them has been very off putting.

HAIGH: Okay, SIAM by contrast would delegate more power, be less bureaucratic?

MOLER: Be less bureaucratic, yeah. Smaller and more personal are the important parts of it. Also the kind of things I'm interested in, applied mathematics, numerical analysis and scientific computing permeate, are important, throughout SIAM. They are SIAM's most important activity. At the ACM that's not true. Scientific technical computing, is a very small part of their interests.

HAIGH: And that's something that's changed over time, I think. I mean Householder was an early president of the association.

MOLER: Oh yeah, long time ago that was true, but it hasn't been true for –

HAIGH: So by the sixties and seventies, it was already quite marginal within ACM?

MOLER: Right. Yes.

HAIGH: Alright, during the time –

MOLER: When the first issue of the *ACM Survey* journal came out somebody reviewed it and said he was glad to see there weren't any summation signs in the journal. [Laughter]

HAIGH: Are there any contributions or innovations that you think were particularly notable during your time editing the SIAM Journal on Numerical Analysis?

MOLER: The SIAM Journal on Numerical Analysis has changed, its become quite theoretical. I can't understand most of the papers that are published in it today. For my own personal interest it's been replaced by SISC, the *SIAM Journal on Scientific Computing*. If I were a managing editor today it would be of SISC not SINUM.

HAIGH: And was that process already at work during the 1970s, or did it happen afterwards?

MOLER: I think it happened afterwards.

HAIGH: So, was there anything you did do as managing editor that put a stamp on the Journal, anything that was controversial?

MOLER: No, I don't think so. That was a long time ago. No, I don't have anything.

HAIGH: Okay. How about SIAM in more general, in your time on the National Council or organizing national meetings, was –

MOLER: The big thing that's faced SIAM in the last several years, and which I've been involved in discussions of, is the impact of the Internet in electronic publishing. So, you know, publishing journals is how SIAM makes its money, and if we start giving them away on the Internet how are we going to exist? Libraries are facing the same dilemma, librarians are scared of the Internet too. How do scientific journals and librarians deal with the Internet? And so this has been something there's been a lot of discussion about in SIAM and I think we've done a good job of handling it.

HAIGH: Any other conflicts or disagreements that you remember over the years?

MOLER: Yeah, there's some we can talk about after you turn off the tape. [Laughter]

HAIGH: Obviously there are some things that just come down to personalities. Is there anything you want to go on the record with, that perhaps reflects different visions of what numerical analysis should be, or what the role of an academic or practitioner should be?

MOLER: Right. Is there a new discipline? So we're now talking about computational science, or computational science and engineering, as a discipline that's different from numerical analysis, and different from mathematics, and different from computer science.

What is that discipline and how should ACM, how should SIAM deal with it, how should universities deal with it?

HAIGH: Yeah.

MOLER: So in scientific and technical computing, there's a body of knowledge that has to do with numerical analysis, and scientific programming, and computer graphics and scientific database management, that permeates a bunch of disciplines whether it's being done by electrical engineers or petroleum engineers or, you know, the Woods Hole Oceanographic Institute, these guys a;; have a common set of things they're practicing. It's part mathematics, part computer science, part computer engineering, and there should be a place in universities to study that, and a professional organization to promote that. SIAM is the obvious place to do that, except SIAM has been slow to take that up. There are still strong connections to traditional applied mathematics at SIAM, and it's reluctant, it doesn't understand this. Other organizations… ACM has dabbled in this, IEEE, their magazine's called *Computational Science and Engineering* that I'm an editor of, that's not a SIAM journal, does it say that there?

HAIGH: Yes, it says IEEE/AIP.

MOLER: Right. So there was an American Institute of Physics magazine, these are not scientific journals but they're sort of magazines that have review articles, and opinion columns, and that sort of stuff.

HAIGH: So they're aimed at practitioners?

MOLER: Yes. And American Institute of Physics had one called, I don't know, something to do with computational physics, the IEEE Computer Society had one, and they were both sort of just of muddling along, not terribly successful, and they came to SIAM, and said "SIAM do you want to take this over?", and SIAM decided not to do it. SIAM doesn't have a popular magazine, but SIAM was hard at work figuring about reformatting *SIAM Review* and wasn't sure they wanted to take this on. So, the jury's still out on that, they combined these two magazines. I'm on the editorial board but haven't been very active in it, I don't know whether the thing's going to succeed or not. But this has become the place where there's discussion about this discipline called computational science. ACM now has a special interest group with that name (I keep saying ACM, we've talked about ACM so much). SIAM has a special interest group on computational science and has a couple of very successful conferences on it, but there's no publication, SIAM publication, with that focus.

HAIGH: Right.

MOLER: Except all of SIAM publications in a sense.

HAIGH: Yeah, I think in some of those areas where the question would be is it one big thing with all these different parts -

MOLER: Right -

HAIGH: Or is it a bunch of separate things that just have some aspects in common?

MOLER: And universities are dealing with this, do you have a program in this, do you have a department in this, what should you do? And that's a big topic here at Santa Barbara.

HAIGH: And you would come down more on the connectedness side?

MOLER: I think that's right. So, the dean at Stanford, I had lunch with him about this topic, and he says "Really we don't need to have a separate department or curriculum in this we already do it. Mechanical engineering is doing it, electrical engineering is doing it, they're all doing computational science," and I said "Yeah, that's just the point. They're all doing it but they're not contributing to the overall fabric. What if we didn't have a computer science department today and you said, look everybody's computing we don't need a computer science department. Or, what if we didn't have a math department and you said everybody's using mathematics we don't need a separate department for that, and there is a common body of knowledge that isn't covered by computer science, isn't covered by mathematics, isn't covered by any particular application discipline that students should be studying and people should be doing research in that."

HAIGH: Do you think you're role in producing software that's been used in so many different areas has made you more aware of this?

MOLER: Oh, I think so, yeah, absolutely. You know, as I promote this I can be accused of promoting MATLAB, because MATLAB would be at the center of this, but it's not the same thing.

HAIGH: And the other thing that stands out on here, in terms of other professional activities, is 1975 to 1984 as editor of the Springer Lecture Notes in Computer Science.

MOLER: That was a long time, and I didn't do anything there.

HAIGH: For that series, would there be a large number of editors?

MOLER: Yes, and the Springer person was somebody I knew, two of them actually, they kept me on the masthead there for a long time. But I never did anything on it.

HAIGH: Not much actual editing, alright. Nothing else that we need to talk about then?

MOLER: Professional activities. There's something that isn't on here…. oh some of these things down here.

HAIGH: Boards and advisory committees on your resume.

MOLER: Yeah, yeah.

HAIGH: Board of Directors, MathWorks, we'll come to that. Trustee of SIAM, we've kind of covered that. Some more recent advisory committee work. These other things are all 1998 to present…

MOLER: Yeah.

HAIGH: Any you think are important to mention?

MOLER: Well, this thing called the, I should just, this is an interesting activity that I can't say much…. naw let's forget about it, let's not talk about that. Yeah, some of that's brand new stuff that I haven't even met with these people on, so, yeah, that's good enough.

HAIGH: Maybe let's just say a word about that in general. So, the resume shows a number of centers and initiatives that have asked you to join their advisory committees, is that because of your prominence with MATLAB?

MOLER: I think so, yes.

HAIGH: Who would the other people on the advisory committees be?

MOLER: It depends upon the committee. In some cases it's because friends of mine have risen to the administrative positions now where they're in charge of this thing. So, for instance, at Cornell, John Swanson started a company called ANSYS that does mechanical design software, finite element software. He's an alumni of Cornell, he gave Cornell a bunch of money to start up a center to do more with the use of computing and engineering education. The head of that center is Sid Leibovich who was my roommate at Caltech, and so he has me on their advisory committee. So, once a year we go to Cornell and hear talks about what's going on in the center and react to them, it's very interesting.

HAIGH: Sure. And with that kind of position what do you think the ratio is in terms, of bringing legitimacy and respectability to something, versus the institution actually changing as a result of what you say?

MOLER: The trouble with some of the other advisory committees… so I was on the advisory committee for the University of New Mexico, College of Engineering, and University of New Mexico, Computer Science Department, and also at WPI. I've quit those things. We'd come there and we'd hear about all the problems they're having, particularly the University of New Mexico, they don't have any money, they don't get any respect from the State, we write a letter saying, you know, these are good people, treat them better, and nothing ever comes of it. It's enormously frustrating, so I don't think some of those things have any impact. Some of these newer ones, it's too early to tell.

HAIGH: Oh, the other thing we didn't mention is your Ph.D. students. You mentioned a couple. Is there anyone else?

MOLER: So, I'm very proud of this list, this has been one of my important contributions, this list of Ph.D. students. They're in three groups, math students from the University of Michigan from my first days there, a couple of people from Stanford from when I've been there on sabbaticals, and then a group of students from the University of Mexico in both math and computer science. And some of the early ones are now in pretty prominent positions and are pretty well known. Norm Schryer was my first student from Michigan, he's had a career at Bell Labs, where he's done lots of things. Allen Klein is an influential professor at the University of Texas, very involved in campus politics at Texas. Charles Crawford went to Canada. David Kammler is a professor at Southern Illinois University and has written an important textbook on Fourier series. Stan Eisenstat is a professor at Yale, Charlie Van Loan is a professor at Cornell, both well known. Linda Kaufman went to Bell Labs. One of these guys, James Sanderson from New Mexico, just got written up in the Smithsonian magazine, he has became a naturalist and a preservationist, he straps cameras to trees and snaps pictures of animals in the night that are thought to be extinct, though that doesn't have anything to do with what we're talking about. Then Jack Dongarra is on here. Dongarra may be my most well known student, he's certainly made a name for himself in super computing.

HAIGH: Yes, he'll be receiving his own interview.

MOLER: Right. So, that's an important list of people. That was an important part of my life. Of course I haven't done anything like that for 20 years.

HAIGH: Do you miss it?

MOLER: A little bit. In my, being back here at UC Santa Barbara I may have a chance to do some more of that.

HAIGH: And would you say there's anything distinctive about your approach to being a dissertation advisor. Did you model yourself on Forsythe?

MOLER: I don't know, I hadn't thought about that. I'm not sure I modeled myself on Forsythe. A lot of people find my personality pretty strong, I'm pretty assertive and I have strong opinions, and I probably let those… I'm more intimidating to students than Forsythe ever was.

HAIGH: In terms of arguing with them, telling them what to do?

MOLER: Yes, and this also happens with employees at MathWorks where I'm Chairman and Chief Scientist, and that's pretty formidable to new people. Once you get to know me I'm a real teddy bear, but your first couple of meetings with me might be kind of scary. I'm back into teaching here at Santa Barbara, and I'm going to do more of that here. I haven't yet got into this level of thing of advising any graduate students though.

HAIGH: So what made you want to come back and teach again?

MOLER: Well, part of it is living here in Santa Barbara. Patsy lived in Santa Barbara for a long time before we got married, and we wanted to get out of New England winters and

so on. I've never had much to do with the day to day management of MathWorks but I'd always had a lot to do with the software development, and now I have less to do with that.

HAIGH: I think many researchers wouldn't be too sad if they never taught a class again.

MOLER: Ahr… Oh, I like teaching. Yeah, I always have, I enjoy doing it.

HAIGH: Are you teaching graduate students here?

MOLER: No, this is an undergraduate course in numerical methods for computer science majors. It's a real challenge because they don't know very much mathematics.

HAIGH: So you hope that one or two of them will be filled with that same experience you had in taking Todd's course?

MOLER: That would be great, that would be great. One or two would be great, yes, that could happen. Thanks… that's a good summary.

HAIGH: Great. Well, on that thoughtful note let's finish the first interview.

MOLER: Nice point.

HAIGH: We've gone though the whole thing in four hours here, which is obviously a lot less time than it took to live it, so maybe we're seeing the symmetry more neatly here.

[End of Tape 3, Side 1] [Start of Tape 3, Side 2]

HAIGH: This is the beginning of the second session held on March 9, 2004.

Something I realized we didn't talk about in the last session was either of the books after your first 1967 publication with Forsythe.

MOLER: After I left Stanford, Forsythe designed and taught a course on numerical computing using software. The idea here was that, rather than the numerical analysis course, where you analyzed and derived algorithms and wrote programs (and you may or may not do any computing, write programs from scratch), this was a course in which you would take software that was already written and learn how it worked and learned how to use it, use it intelligently. It was a course for not for numerical analysis specialists, but as a kind of a service course for students and elsewhere in the university. Forsythe taught this course a couple times at Stanford using some Fortran subroutines. Then a graduate student of his named Mike Malcolm taught the course a couple times, and together they wrote some notes, which I guess were the draft of a textbook for that course.

Forsythe suddenly died of stomach cancer, and when he knew he was sick he asked me to help Malcolm finish the book, so we did. I worked on it some and we submitted the manuscript to Prentice-Hall, and Prentice-Hall sent it out to some reviewers and they said, "this book reads like it was written by three different authors who never talked to

each other." And that was true, the different chapters were written by different people, and there was no integration. We defined the norm of a matrix three different times in three different places. So we put the project on the shelf for a while, and both Malcolm and I got divorced and remarried, moved, and eventually we got back to it. I rewrote the thing pretty much from scratch. We changed the software, and then we published it with Forsythe as an author even though it was three or four years after his death, and this book was called *Computer Methods for Mathematical Computation*. Prentice-Hall, 1977. And it covered the standard topics: matrix computation, finding roots of poly, finding roots of equations, numerical evaluation integrals, numerical solution of ordinary differential equations, random numbers, least squares.

That book came with about 12 or 15 Fortran subroutines, most written specially for that book. The subroutines were printed in the book and the subroutines were intended to be read by humans as well as used on the computer. That book became quite popular and widely used for a long time, both as a textbook for this course and as a source for those subroutines. I used to distribute the subroutines. It was a box of punch cards, there are 2000 cards in a box, and I could get the entire library in one box, and I would send people a box of punch cards, or I'd send them a small magnetic tape. In fact, I set up a chain letter thing where I'd address it to four or five people, and mail it to the first one and then say when you've made your copy please mail it on to the second one, that worked quite well. We distributed that that way for quite a while, when Netlib started up these subroutines were put into Netlib.

Ten years later, David Kahaner and Steve Nash said they wanted to rewrite the book, modernize the software, put more mathematics into the discussion, and they did that, they rewrote the book, and they used some of the material I'd written, and some of the homework problems. I didn't contribute much new to the new book, and that book became Kahaner, Moler and Nash, *Numerical Methods for Software*. Here instead of using the simple readable software Kahaner wanted to get serious software and so he got Fortran subroutines that were being used at Los Alamos, and he was at the National Bureau of Standards, there was a project called SLATEC, which was the math software project involved with the National Labs, and so they SLATEC software in this book. The software wasn't printed in the book, it wasn't intended to be read by the students, it just became black boxes, and that was a mistake. I wasn't happy with that approach to it, it was a different attitude towards the software. That book was still pretty popular. I don't think it was a popular as the original Forsythe, Malcolm and Moler, but it was fairly widely used. Both of the books have been replaced in modern times by Numerical Recipes, that's not our book, that's another book that replaced it. The book I've just finished is now a MATLAB version of Forsythe, Malcolm and Moler, over 20 years later.

HAIGH: And with the MATLAB version, have you found some way of giving those students an idea of what's going on inside?

MOLER: Absolutely. They have to, the programs are printed in the book. They're MATLAB programs so they're much shorter than the Fortran program, many of them will fit on one page, and the homework consists of not only using these programs but also

modifying them, adding new things to it, changing them in some way so that you really have to read them and understand them.

HAIGH: Back to the 1977 book. At that point was this a novel idea to provide a book together with a –

MOLER: Collection of software –

HAIGH: Collection of high quality mathematical routines?

MOLER: I think it was yeah, I can't think of anything that was comparable at the time, yeah it probably was.

HAIGH: And this book again would have been used for a wide range of scientific and engineering disciplines?

MOLER: That was the intention, yes.

HAIGH: Did you find that some areas picked it up more than others?

MOLER: I don't know, I'm not sure who all used it. As I say, it was widely used and in computer science departments, in Math departments, but then in various engineering departments and physics departments. A couple of the routines from book actually found their way into MATLAB eventually, so pieces of that were predecessors to things in MATLAB.

HAIGH: Alright. That probably gives us a good lead into a discussion of MATLAB then. So where did MATLAB originally come from?

MOLER: So we talked yesterday about LINPACK and EISPACK and the projects at Argonne, while I was working on LINPACK, it started with LINPACK, I was also teaching in New Mexico and I wanted to have my students have access to LINPACK and EISPACK without writing Fortran subroutines, which they didn't know how to do. I wanted them to have easier access, so I wrote the first version of MATLAB while I was a professor at the University of New Mexico, as a kind of a matrix calculator. It was not very sophisticated, you couldn't add functions to it, it didn't have any graphics. It was simply a calculator for matrices, you could invert matrices and compute determinants, and compute their eigenvalues, and that was about all.

HAIGH: And what year was this?

MOLER: This was in the late 70's. It evolved over several years, I suppose it probably started maybe 1976, or 1977, and it went on for, until MathWorks itself was founded in 1984.

HAIGH: And through that period was the package only used in your classes, or did it begin to spread?

MOLER: It began to spread. We sent out from the University of New Mexico where I was. Much of it was in the time when I was chairman of the computer science department and a little numerical software research group in New Mexico. We probably sent out two or three hundred copies of it, we'd make magnetic tapes, we actually charged people a $75.00 service fee for making the tape, and that money went into the fund of our research group. A student there named Karen George worked for me as a research assistant, and she made the tapes, and part of her salary was paid by the $75.00 fee, so Karen George is the first person in history to make any money off of MATLAB.

HAIGH: But all the actual code was written by you?

MOLER: Yes, yes.

HAIGH: And did it work interactively?

MOLER: Yes it did, yes. It was a calculator, you sat at a terminal of some kind and typed in statements and had it do the arithmetic and print back the results. That's the only way it worked, there wasn't any real facility for programming it. There were time-sharing systems at the time, the University of New Mexico had a General Electric time-sharing system on its central computer and we used that. There were commercial time-sharing systems that people subscribed, to and we used that. I worked on it at Argonne in the summers, and on the IBM mainframes at Argonne they had a time-sharing system (it was called TSO, I think for time-sharing option) and MATLAB worked on that.

HAIGH: Was that an IBM product?

MOLER: It was an IBM product, an IBM operating system.

HAIGH: So was the code portable?

MOLER: The code was in Fortran, and yes, very definitely intended to be portable, and it ran on a wide range of machines. There were concerns about such things as character sets because not all Fortrans, this was before the ASCII standard characters, so we had different ways of representing characters on different machines. This was when people first began upper and lower case text, and originally the MATLAB was case insensitive, because some machines had upper and lower case, and others didn't.

HAIGH: And through this period were the underlying routines all from LINPACK and EISPACK?

MOLER: Yes, they were from LINPACK and EISPAK. It was only a small subset of the routines of LINPACK and EISPACK. LINPACK and EISPACK were big packages by the standards of that time and MATLAB didn't use them all. For example, if you solve a system of linear equations you can take advantage of symmetry, and LINPACK had a symmetric linear equation solver. MATLAB didn't use it, MATLAB ignored the symmetry and just used one subroutine for all kinds of matrices. It was never important that MATLAB execute fast. The run time efficiency of MATLAB, until recently, has

never been a top priority, it was ease of use, availability of different features, convenience, that overshadowed trying to make it fast.

HAIGH: Presumably since the base of LINPACK and EISPACK were portable, that made portability of MATLAB easier then?

MOLER: Well yes, but the numeric part of MATLAB was the easy part. The original thing about MATLAB was the language parser, and that's something you don't ordinarily write in Fortran, and that wasn't easy to do and easy to make it portable. There's a book by Niklaus Wirth, (he was later the inventor of Pascal, but before Pascal he'd written a textbook called *Algorithms Plus Data Structures Equals Programs*, it was published by Prentice-Hall), and that talked about how to write a parser and how to write a small compiler. He invented a simple language called PL0, and then he did a recursive dissent parser for PL0. So, I read that book and learned that technology, and MATLAB was PL0, implemented in Fortran, with matrices as the only data type. Fortran wasn't recursive, I had to make my own stack to handle the recursion within Fortran, so I did that in a portable Fortran and it was tricky.

HAIGH: Were there any other ideas around at the time that were influential on the MATLAB?

MOLER: Yeah, there are. So, Ken Iverson's APL language was already in existence. When time sharing first started and people first began to interact with computers at a terminal rather than submit cards to the computer center, BASIC and APL were the two ways they did interactive mathematics on computers in those days. APL is a bizarre language: an exotic character set, lots of strange constructions, but it got a bunch of fans, there were APL groupies around at the time, still are today. It only ran on IBM mainframe. You can think of the early MATLAB as a portable APL with a civilized character set, a standard character set. There was also a program developed at Argonne, called Speakeasy, a wonderful name for a program. The principle author or Speakeasy was Stan Cohen, a physicist at Argonne, and that had a lot of influence on MATLAB. It was an interactive math program in which the matrices and vectors played a prominent role. You could sit at a terminal at Argonne tied up that IBM mainframe and do interactively do calculations in Speakeasy. I think it even had some graphics, I'm not sure. Cohen had a small group there devoted to Speakeasy and he used to have students come in the summer and they'd work on Speakeasy. Argonne was interested in spin offs and technology transfer and so they actually asked Cohen to leave Argonne and start his own company, he didn't particularly want to but he did. He went to downtown Chicago and started the Speakeasy Company, and I think it's still in existence today. It was never widely used but it was a very interesting idea and an important influence on MATLAB. So APL, Speakeasy, LINPACK, EISPACK, and PL0 were the predecessors to MATLAB.

HAIGH: So why was MATLAB so much more successful than Speakeasy? Was it portability?

MOLER: May have been portability. Speakeasy initially only ran on IBM machines, IBM time sharing systems… good question. MATLAB became popular within my crowd, which was the numerical analysis, numerical linear algebra crowd, and in 1979 to 1983 that's where it was used. It became popular within that crowd but that's not a big group, certainly not a basis for a commercial product. Speakeasy had a different clientele so it was done by a physicist, and Cohen never promoted Speakeasy very much. I don't know, I don't know. Interesting question. I don't have a clear answer to that.

HAIGH: Was it very difficult to write portable interactive code in Fortran?

MOLER: Was it difficult.

HAIGH: Yes, did Fortran have terminal capabilities?

MOLER: No, Fortran didn't have capabilities for that. But there wasn't much to this, I mean it was only a question of being able to read characters from a terminal. The time sharing systems had some way that Fortran could get its input from a terminal instead of from a file system, and could send it's output back to a terminal. So, that was how you got the characters in and the characters back out. Then there was this parsing stuff. Character manipulation in Fortran, that's not natural, but I knew how to do it. The programs still works today, I could show you the original, original MATLAB here. I'm not sure how we would put this on the tape, but you want to take a look at it? I've actually been thinking a source code for this original MATLAB on our web site at MathWorks as sort of a historical artifact, people are interested in it. Whenever I give a talk about MATLAB, that has a historical flavor, I show off the original MATLAB and people enjoy seeing that.

HAIGH: What would be cool is a Java emulator running the software on a virtual machine, inside a web browser. I've seen this done to run software for old micros. So, if something like that existed for a minicomputer platform that MATLAB ran on you, might be able to have it appear interactively running inside a web browser, using the original executable code.

MOLER: So yes, that's the question, how to do this over the web? That's a very good idea. And then you have to hook the Java up to Fortran.

HAIGH: The Java is emulating another machine, so if anyone has written a Java based application that would run inside a browser to emulate a platform that MATLAB ran on…

MOLER: Well MATLAB itself will run on machines today, if you have a Fortran compiler for your PC. There's a Fortran compiler in Windows that will compile the original Fortran MATLAB, and you can run it.

HAIGH: That's true. Also, I understand that it's theoretically possible to produce compilers for other languages, such as Fortran, that directly target the Java Virtual Machine, but I'm not aware that anyone's done it.

MOLER: That isn't what I'd do. You'd want a Java program managing the window and collecting the input text, and then hooking up with the Fortran program to do the calculation. You don't want to emulate any machine, you've got a machine. Just run it on that machine. Yeah, okay, I'll think about that. Good idea.

HAIGH: Returning to the topic…. Once you've written interactive Fortran code, it seems that wasn't too much of a problem confining it to run on a wide range of time sharing operators?

MOLER: That's right.

HAIGH: Did commercial time sharing services add MATLAB to their libraries during this period?

MOLER: No, they didn't.

HAIGH: Was that because they didn't want to, or because you wouldn't have let them?

MOLER: That's because they probably didn't know about it. This version wasn't that useful to a general audience. It would be useful in teaching in numerical linear algebra course and that's it.

HAIGH: So, at this point, all of those two to three hundred copies that were dispatched would be going to teaching courses in this area?

MOLER: That's correct, yeah.

HAIGH: Was there a formal version number system at this point?

MOLER: No, there wasn't a formal version numbering, there were dates on it. So when you ran MATLAB it would say version of March 3, 1980.

HAIGH: And roughly was proportion of your time did you spend working on MATLAB?

MOLER: Oh this was very much a hobby, a side project. There was never any support of any kind. We had research grants for LINPACK and that didn't cover this. I had this IBM research grant that I mentioned, that didn't cover it, so there were never any proposals written to develop MATLAB. It was a hobby.

HAIGH: Do you think anyone would have been interested in funding it?

MOLER: There was some interest in it, yes. About probably 1983, I actually talked to NAG about it. There was some interest at NAG about possibly taking it on as a commercial product, but that's about when John Little entered the picture and MATLAB became a commercial product on its own.

HAIGH: I see that you got one conference paper out of it, at the 1980 National Computer Conference. Can you remember anything about how that was received?

MOLER: I there was an announcement at that computer conference that I was talking about matrix computation, and people showed in the room thinking that I was talking about a management system where you had matrix management –

HAIGH: Oh, matrix management, like DEC.

MOLER: Matrix management, yes [Laughter]. So, that shows what the National Computer Conferences had become by then. There was another paper there, there's an earlier paper at a conference in Mexico, "Demonstration of a Matrix Laboratory," *Proceeding of the Cocoyoc Conference on Numerical Analysis.*

HAIGH: But that's three years later in 1983.

MOLER: Oh, it is, okay.

HAIGH: So this, this presumably, this conference would have been when MATLAB was unveiled to the public?

MOLER: I think that's right. As we have tried to establish, you know, intellectual property questions about MATLAB, we've gone back to that 1980 paper as being the first published paper on MATLAB. There was also a technical report out of the University of New Mexico, two of the them actually, one the users guide for MATLAB, and the other the installation guide for MATLAB, and those aren't on my resume because they weren't refereed publications. Is there a later section where they mention that? No… this is my official resume and it doesn't have all the technical reports I wrote.

HAIGH: Well those technical reports might be worth preserving.

MOLER: Yes, they are, yes. We have copies of those and I have them and then MathWorks has them as part of its history of MATLAB collection, such as it is.

HAIGH: So at the conference there were people who turned up and were disappointed that you weren't talking about matrix management. Presumably people teaching the relevant courses, more part of the community that already knew about MATLAB. Were there any cases of people who saw the package for the first time and said, "Oh, an interactive teaching aid, this is something important, this gives me an idea, why didn't you do this years ago." Anything like that?

MOLER: Well, I wouldn't put it that way. It's about time, I think, that we got to the year I spent at Stanford, and the interaction that led to the start of the commercial version. You want to get on to that?

HAIGH: Sure. What is that?

MOLER: Okay. So, that was the academic year 1979-1980, and I spent that year at Stanford. I taught a course at Stanford, Math 237, a graduate course in numerical analysis, and it was a course on matrix computation. There were probably 25-30 students in the class, half the students were graduate students in computer science, numerical

analysis math. I used MATLAB in this course, the course wasn't about MATLAB but MATLAB was a tool that was used in the course. The computer science students and numerical students were not very impressed. This was not a sophisticated language, they were taking courses from Don Knuth and John McCarthy, and so on, who did real computer science. There was not much mathematics going on here, these algorithms were old, traditional algorithms, there was no convergence theory, no partial differential equations, this was not the usual fare for the graduate course in matrix computation. The other half of the students in that class were students that had come from various departments in engineering, electrical engineering, mechanical engineering, nuclear engineering, they knew about matrices, they'd used matrices a lot in their theoretical work, many of their textbooks described things in terms of matrices. They had done computation with matrices using Fortran. Some of them knew about LINPACK and EISPACK, when they saw MATLAB they thought this was terrific. They were doing calculations in subject areas that I didn't know anything about, particularly control theory and signal processing. The language of automatic control is expressed in terms of matrices, or what in electrical engineering is called system theory, and it's about both control systems and electrical system. This is all in terms of matrices, the stability of these systems depends upon eigenvalues of matrices, and in designing a feedback control system you use matrix analysis to adjust the gains and the delay times, and so on.

So these students, particularly in control theory at Stanford in electrical engineering, took up MATLAB. There were two companies in Palo Alto that had spun off the Stanford electrical engineering department. One was a company called ISI, Integrated Systems, Inc., and the other was a company called SCT, System Control something. MATLAB got to both those companies, either because employees of those companies took my class or because students who took my class soon went to work for those companies. They started using MATLAB in their work at those companies, and they both developed commercial products based on top of Fortran MATLAB. The first one was developed at ISI, and the product was called Matrix-X. They asked me could they use MATLAB in this product, and I said yes, so they took that Fortran distribution of MATLAB I had, they added a great deal to it and made a commercial product out of it.

HAIGH: And the commercial product was another interactive matrix calculator?

MOLER: Another interactive matrix calculator, but now with graphics, and particularly with functions to do control theory. I didn't know anything about control theory, but the commercial products added control theory capability to the functions that were already in MATLAB. Across town in Palo Alto, at Systems Control, they also built a commercial product. They were about six months behind ISI, ISI was their competitor, they built a system called Control C. The primary engineer behind Control C was a Systems Control employee named John Little, he hadn't taken my course at Stanford but colleagues of his had, and I'm trying to remember the name of the guy. There was a graduate student at Stanford who took my course at Stanford, and he either already worked for System Control or went to work there soon after. He took MATLAB over to System Control, and Little says he saw it and its changed his life that he never wrote another Fortran program again, he began using MATLAB immediately.

HAIGH: Did MATLAB have its programming language by this point?

MOLER: No, no, this was the simple calculator. You could add functions to it by writing the functions in Fortran and adding them to the system. So that was what was originally by both these companies, they took my Fortran and added additional function to it by writing Fortran programs to solve these control problems.

HAIGH: And at this point there were never any cases where people gave you feedback or new pieces of code that you incorporated into MATLAB?

MOLER: At SCT Little's had a colleague who was named Steve Bangert. Little and Bangert found some errors, in my Fortran parser and they told me about them and I changed it as a result. It wasn't new functionality it was just fixing something that was already there.

These two products, Matrix-X and Control C, were commercial products that these two companies sold, I don't know how many they sold or how much money they made off it in the first two years, but they became quite successful in their own way, people talked about them at conferences and so on. In 1983, I think it was, there was a conference in Boulder, Colorado, organized by Lloyd Fosdick on mathematical software for small machines. The PC had just come out and people were trying to use the PC for technical computing, it was hard because the PC was small and slow, and didn't have much memory. I went to Boulder with a couple of my graduate students from New Mexico. We hauled a couple of IBM PC/XT's with us to show off MATLAB at this conference. John Little came from California to the conference specifically to see me and to ask me about commercializing MATLAB. He appreciated the possibility of the PC as a serious technical tool, he wanted to start a company based on MATLAB, and he was a control engineer. He's a systems engineer. He has undergraduate form MIT, and has a masters degree from Stanford in electrical engineering with a emphasis in control theory, and he was working for System Control as a control engineer on some Air Force contracts.

HAIGH: How old would he have been in 1983?

MOLER: In his twenties… we can look that up.

HAIGH: Quite young.

MOLER: Quite young. He went into his boss at System Control, and said he wanted to quit and start a company, and his boss said could he come with him. System Control was quite an incubator for little companies. Little quit his job at System Control, and went off, he was renting an A-frame, a small cabin, in the hills up behind Stanford. With his own money he went out and bought a Compaq portable, it was the size and weight of a sewing machine, a little green screen on it. Initially his machine didn't even have a hard disk, he had to swap floppies. He bought this machine with his own money, and started building the commercial MATLAB.

He took the design I had, but he threw away all my code. The Fortran was gone, and he programmed in C, which was kind of a gutsy thing to do at the time because C was a

fairly new language, it had never been used for large scale technical programs. The floating point arithmetic capabilities in C didn't work very well at the beginning, but Little used it. He took the LINPACK routines and EISPACK routines that I had written, that I had used, and carefully rewrote them in C by hand, being sure to preserve the numerical properties of the original Fortran, because he didn't want to change those algorithms at all.

Steve Bangert had a wife and kids, so he couldn't quit his job at SCT, but Little asked him to join him, and Bangert worked on it part time on nights and weekends. Bangert wrote the language processing part, and Little wrote the mathematical part, and they reimplemented MATLAB. They also added M-files and toolboxes, and they added more control structure so at this point it became a programming language, rather than just a calculator.

[End of Tape 3, Side A] [Start of Tape 3, Side B]

MOLER: So, in 1983 and 1984, when Little and Bangert were creating the new commercial MATLAB I was on the sidelines. I talked to them, I knew what they were doing, I encouraged them but I didn't really participate very much, well maybe I began to, so as soon as we had M-files I began to write some M-file functions that extended MATLAB, it would be interesting to see when I did the first one. I can't remember.

HAIGH: So, what exactly was an M-file?

MOLER: An M-file is a program written in MATLAB's own language, so –

HAIGH: Does the language have a name?

MOLER: The language is M, for MATLAB.

HAIGH: And what was distinctive about the language?

MOLER: What was distinctive about the language is that it was based on matrices. There's a joke that MATLAB is strongly typed, it has only one type. It's not true today, but for years in MATLAB the only data structure in MATLAB in the matrix, even scalars were one by one matrices. If you wrote a FOR loop in MATLAB, FOR I=1 TO N, I and N are matrices, they're 1x1, and they have integer values, and there's no imaginary part but they're still matrices. There are no type declarations in MATLAB so everything's a matrix, it is created on the fly. So that was what was special about MATLAB. MATLAB stands for MATrix LABoratory, it was all based on matrices. Things that if you're programming in Fortran, or C, or Java require doubly or triply nested loops, become single statements in MATLAB, and the notation is much loser to the language of linear algebra, the language of matrices that is used in textbooks on control theory, for example.

HAIGH: Have the notations been similar when used in batch mode as a calculator?

MOLER: Yes. The notation was the same.

HAIGH: So, aside from letting users write functions, and work in a macro kind of way to string things together, were there other important things that the program language brought to the overall product?

MOLER: No. Well, the programming language consisted of taking the calculator language and putting it in a file and adding parameters to it so it became a function. I don't know if there were any additions to the language really, because I already had FOR statements, and WHILE statements, but then once you have this then you immediately start writing programs in this language, functions in this language, that then extend it's capability, particularly in the areas of control theory and signal processing. A collection of programs for a particular application area is called the toolbox, and the first toolboxes were in control theory and signal processing. Little actually wrote those, himself, in the first versions.

HAIGH: So, right from the beginning MATLAB would have shipped with a collection of toolboxes?

MOLER: Yes. There's the MATLAB toolbox, which extends the mathematical capability of MATLAB itself, and then there are application toolboxes, like control and signal processing, which you pay extra for, that extend in that particular specialized area. The numerical solution of ordinary differential equations is another important mathematical tool in MATLAB, and I wrote the first ordinary differential equation solvers, in M, in the very early days of the commercial MATLAB.

HAIGH: So, this fairly small technological development had an enormous impact on the way that the program would be used?

MOLER: Yes, that's right. The commercial debut of MATLAB was at a control conference, I think it's the ACC, the Automatic Control Conference (there are two annual control conferences, one occurs in June and one occurs in December, and this was the December one). It was in 1984, in December, in Las Vegas, Nevada, and we debuted the commercial MATLAB there. Coincidentally I was on my way from New Mexico where I had just resigned as department chairman, to Oregon where I was going to work for Intel, and I drove through Las Vegas on my way, with a computer in my car. Little flew from California, and the two of us showed off MATLAB at the Control Conference in Las Vegas just before Christmas in 1984.

HAIGH: And where had the money come from to underwrite the development and launch of the program?

MOLER: It was all Little. I mean everybody was on their own. Little quite his job, didn't have any salary for a year and a half, and was living off his own reserves. Bangert kept his job at SCT until MathWorks began to make some money, and then he went to work for MathWorks. Jack's girlfriend at the time, named Nancy Whittenberg, became our business manager initially. She didn't have any salary until there was some money to pay for her. There was never any outside investment.

HAIGH: Do you know if the company had been incorporated prior to the product introduction?

MOLER: Yes, it had. The company was incorporated in California in 1984.

HAIGH: And did you have any equity in the original company?

MOLER: Yes. The primary owners of the original company were John Little, me, and Steve Bangert. There were a couple of other guys who were sort of early advisors who had small pieces of the company originally, but they're out of the picture now. There was a guy who had helped with the graphics who was a rather small fractional owner at the beginning.

HAIGH: When the product was introduced at the conference was it finished and available for purchase?

MOLER: Was it finished? It's never finished. It was available for purchase, that was Version 1, maybe it was Version 0.9. Maybe it was almost finished, but yes, we were ready to go with the first version. We can check on this. The actual first copy of the commercial MATLAB was sold to Nick Trefethen at MIT. He was in the numerical analysis community, he wasn't a control guy, but he bought the first version of the commercial MATLAB for use in his class at MIT, in the way I'd been using it for years before that. His purchase order, we framed it and it's on the wall back in Natick. I think it's dated February, 1985, it has to be, it couldn't have been February, 1984.

HAIGH: Did he purchase one copy?

MOLER: No, I think he bought ten copies.

HAIGH: So, it was clear to everyone that with the commercial version, unlike the earlier one, you couldn't just copy it and give it to your friends.

MOLER: Yes, yes.

HAIGH: And did you take a booth at the conference, was it a paper presentation?

MOLER: No. It was booth. Well, it was sort of a booth, there were exhibits at the conference but it was very small scale, hardly anything at all tabletop exhibits. I think we had a card table, or a portable table. Little made a portable sign by taking PVC pipe and gluing it together, and he had a kind of banner printed up. He brought his portable COMPAQ from California, I brought my PC/XT from New Mexico. We had just moved out of our house in New Mexico, and the movers wouldn't take like plants with them on the moving truck, so I had a potted palm in my car that I was taking from New Mexico to Oregon, and I brought it in and put it alongside our table…

HAIGH: Thus adding sophistication.

MOLER: That was our first booth.

HAIGH: And at this point there was essentially no sales or marketing function at all?

MOLER: That's correct. Little had anticipated that and there was a kind of, there was a company that had been selling Control C, and was continuing to sell Control C. There was a product for simulation called ACSL, and it was written by two guys named Mitchell and Gauthier. They started a company called MGA for Mitchell Gauthier Associates. They sold ACSL, they sold a couple of other things and Little set up an arrangement with them to sell MATLAB. That went on for two or three years probably, so for the first couple years of MathWorks we didn't have our own sales organization, it was done by a group at MGA. It soon got to the point where MATLAB was that company's biggest product, even outselling their own product ACSL, and within the company the various sales group began to compete with each other so we actually acquired the MATLAB portion of that company to become our own sales force.

HAIGH: Do you remember what year that would have been?

MOLER: That was, the arrangement with the other company went on for three or four years probably, so that, it was probably 1988 that we did that.

HAIGH: How about physical production of the copies?

MOLER: MATLAB was a small program to begin with. Little was careful that MATLAB fit on the PC at the time, and the first PC's had only half a megabyte of memory, and MATLAB would run in that much memory.

HAIGH: That was the maximum I think, the minimum on the motherboard was something like 16 or 32K –

MOLER: Yes, but you probably had to get the full memory to run MATLAB.

HAIGH: Would it fit on one floppy?

MOLER: It may have fit on one floppy, certainly the distribution media for PC's, initially, was floppies, and it was probably two or three with some of the M-files with it. The initial shipments were from California, and I don't know how Little made media for other machines, because he sold copies for the VAX and the Sun.

HAIGH: That's important. Were the other ports made available simultaneously with the PC version?

MOLER: Nearly simultaneously. The PC version was first… no it must have been later. As soon as we got some money, and that money must have come from sales of PC versions, we bought a Sun workstation, and Bangert put it in his home in Cupertino, and that became the central machine of MathWorks, a Sun workstation in Bangert's bedroom in Cupertino. We got to the point where we could make tapes but I'm not sure if that was in California or not. Little soon moved the company to Massachusetts, and we'll talk about that, but by the time we had offices in Massachusetts we had a tape drive, we could

make tapes. The initial copies of MATLAB were just little reproducing discs on his COMPAQ.

HAIGH: Did it come with a printed manual?

MOLER: Yes. That was one of my early contributions. I had written the original manual, and we adapted the original manual to the commercial version. The original manual, the text processing system was something called TROFF, which is the Unix text processing system, and one of the first things I did when I got to Oregon, was to find someone who could TROFF and produce printed manuals from it. It turned out to be a guy I knew back in New Mexico, named Walt Brainerd. Brainerd worked at Los Alamos, but he had a little company in the basement of his house where he had a small PDP machine, and some kind of type setting equipment, and he took our TROFF and produced the first manuals. Looked pretty good. It was typeset and most other beginning software then was not documented that well.

HAIGH: So that made it look more professional.

MOLER: Yes, it did.

HAIGH: And how about service and support?

MOLER: Well, the same story. MathWorks started in 1984 with one employee, John Little. It doubled every year for the next seven years, so by 1991 we had 2 to the seventh people, 128 people. So, first of all there was one, two, then three. In the early days everybody did everything. MathWorks was only in California for about a year and then Little and Nancy moved back to Massachusetts, bought a home in Sherborn, Massachusetts, and moved back there. There wasn't any MathWorks office in California, it was just Bangert and Little in their homes. So Little and fiancée moved back to Massachusetts, and that meant now MathWorks was in Massachusetts. Little's family had a summer home on Nantucket, and during the summers they'd go over there and they'd have the telephone calls forwarded to Nantucket, or forwarded to Cupertino, so you could call the MathWorks headquarters, and you may end up getting to Bangert in California or to Jack and Nancy out on Nantucket, if that's where they were.

HAIGH: And that would be the situation for the first two years?

MOLER: First couple of years, yeah.

HAIGH: And how did the product sell after it was first introduced?

MOLER: It was slow at first, one or two copies a month something like that… we ought to see if we can find out the statistics. This is now the twentieth anniversary year of MathWorks, and we're going to have a celebration in May, the entire company is going to Disneyworld in Florida, we're shipping a thousand people to Florida. People are putting together some historical kind of information and we can get more firm information on that. Since I was never involved with the commercial aspects, the beginning here, I've never had first hand information about that.

HAIGH: But you do know that it was at least a couple of years before it became apparent that this was going to grow into a successful company that would support everybody full-time.

MOLER: That's right. Well, you can roughly figure out what the income of the company was by the number of employees, because we hired as many people as we could afford.

HAIGH: So, in these very early days people were working from home without pay, through the period of initial commercialization. After the product started being sold were people drawing salaries?

MOLER: Yes, yes.

HAIGH: Where did the money come from?

MOLER: From the sales.

HAIGH: So if you only sold a couple of copies in a month, how expensive was a copy?

MOLER: Couple of thousand dollars. There wasn't much money there. As I said, Little was the only, he was the only employee and he wasn't even paid for the first year of the company or something like that.

HAIGH: Did sales just pick up gradually or was there some kind of breakthrough moment?

MOLER: No, I don't think there was any breakthrough moment, not that I know of.

HAIGH: Where were the early sales going to?

MOLER: MATLAB has always had a broad base of sales, academic sales are always an important part but not a big money maker, we've always had very, very significant academic discounts and that's been important because students have learned about MATLAB in school and then gone off to companies and wanted to use it there. But it was also sold to research engineers in automobile companies, in aerospace companies, Wall Street firms, research institutions like Woods Hole and Los Alamos. Those were the early adopters, with just handfuls of sales at these places initially.

HAIGH: And how did that change over time?

MOLER: It just increased and got more board based. It's only within the last few years that we've had industrial site licenses, so we now have something called the… it's an acronym for "how many engineers do you have working at this company?" So we'll go to a company like Lockheed or Ford and they'll have enough uses of MATLAB that they'll say, "alright we want to get it for everybody, we want a price that covers everybody in this organization." It's only been within the last year or two that we've been able to sell it on that basis.

HAIGH: So, in the early days then one person would have found the product, liked it, told their friends, is that how –

MOLER: Yes, that's right. Word of mouth was important, students coming in that used it in universities were important.

HAIGH: Can you remember what the first things were that MATLAB did to work with its user community, or to encourage its development?

MOLER: We've always been a user of the Internet. I think in terms of dotcoms registered, we're number 73 or something like that.

HAIGH: Would that be the late 80's?

MOLER: Oh, no, not late 80's, early 80's. That was soon after we got started. At the time there was about five backbone machines on the internet that talked to each other, and then that got connected to nearby smaller facilities. So there was a backbone machine that Apple computer in Cupertino, which is a few blocks from Bangert's house, so originally e-mail to MathWorks.com went through Apple in Cupertino to Bangert's place, and that was, that was the middle 80's, I guess.

HAIGH: How about the USENET newsgroup?

MOLER: I think we just had the tenth anniversary of that. That was actually started by a graduate student at UCLA. We didn't start it, but we've always participated in it as individuals. It's not moderated, it's not controlled by MathWorks, but we support it strongly.

HAIGH: And the printed newsletter?

MOLER: Yes, we've had a printed newsletter for… well we can go look that up, we were looking at it last night, because I've always written a column in it called Cleve's Corner. Does that have the dates on Cleve's Corner?

HAIGH: Your first Cleve's Corner listed on this resume is Winter, 1990.

MOLER: That's probably right. I remember an earlier, simpler one than that. I mean, the first one was about floating point arithmetic and it isn't listed there.

HAIGH: So, through this very early period, before you joined the company full time in about 1989 then, these deliberate efforts to work the user community hadn't yet begun? The users would just buy the product and they would tell their friends and they would sell it on?

MOLER: Well, or we ran ads, we had sales people calling.

HAIGH: In what kinds of places would you have advertised?

MOLER: In control magazines, and signal processing magazines. The *IEEE Spectrum* has always been a place where we've gotten a lot of leads.

HAIGH: And was that because that was the area that the original toolboxes were strongest in?

MOLER: Yes, Yes.

HAIGH: And that in turn was the result of Little's background?

MOLER: Correct. And that's why the commercial MATLAB had been built, was to serve that community. That was always anticipated as where we were going.

HAIGH: So when the company did start to hire people, what were those people doing?

MOLER: Employee #1 was Little, Employee #2 was Bangert, and Employee #3 was Nancy Whittenberg, who was our business manager. Employee #4 was Loren Shure. Little had actually known her as an undergraduate at MIT, she had gone on to UC San Diego and gotten a Ph.D. in geophysics and she was hired as a kind of general purpose mathematician programmer. She wrote the first signal processing toolbox. When it came time to hire a marketing guy, that was when I was at Ardent, so it must have been 1987, there weren't enough people in Massachusetts to interview him, so we flew him to California for me and Bangert to talk to him. We hired a woman to handle the mail and the orders and make discs, kind of an office manager.

HAIGH: And by that point you had a physical office to manage?

MOLER: Yes. Little and Nancy moved back to Sherborn. When they hired Loren she actually spent her first month working in Little's dining room in Sherborn, but then they rented some office space in Sherborn, Massachusetts, 20 North Main, and they were in that office space until there were maybe eight or ten of them. And then the company moved to South Natick, Massachusetts, Elliot Street, in a lovely old building with nice grounds around it, and they were there for several years. The first time I went back to Massachusetts and visited them was in that building. I never visited them when they were in the Sherborn offices on North Main. There was maybe a dozen people working there, something like that, so by my rule of doubling every year that must have been 1988 maybe, 1987, and I remember walking in this place and there was a conference room with a conference table and there was a supply room with pads of paper and staplers and so on, and I said, "My God this is getting to be a real company!"

HAIGH: And when was the second version of MATLAB produced, and what did it do?

MOLER: MATLAB 1 was in 1984, MATLAB 2 was in 1985, MATLAB 3 was in 1987. I have in my mind the toolboxes were introduced with MATLAB 3, but I can't believe it took us three years to get to that point. I think MATLAB 2 was when toolboxes were introduced, in 1985, and MATLAB 3, in 1987, had probably, was significant graphics, black and white graphics, the PC's began to have graphic cards on them, there was the

Hercules graphics card. Five years later, in 1992, that's when we introduced color graphics with MATLAB 4. I was working for the company by then.

HAIGH: So, thinking back to what you've been saying. MATLAB 1 would have been at the point when sales were very low, MATLAB 2 would have been the first version to sell in appreciable numbers.

MOLER: Maybe, maybe. MATLAB 2 came right after MATLAB 1, there probably wasn't much difference there.

HAIGH: Let me think of the timing on this. Would MATLAB 3 have been about the time that you came to work for the company full time?

MOLER: Yes… no.

HAIGH: Sorry, 1989 is that date. So, can you give a snapshot of what the company was like at the point at which you came to work for it full time?

MOLER: I came to work for it full time in 1989, when Ardent failed. I stayed in California and worked out of my home in California for a couple of years before I moved back to Massachusetts, there were about 30 people working for the company in 1989, our offices were in South Natick. By that time we had hired Jean O'Keefe as our business manager, and she continues to be our CFO and Senior Vice President today. She's another very important figure in the history of MathWorks. She was already on board in 1989. The entire business side, personnel side, human relations, negotiating over buildings and space, has all been Jean O'Keefe, she's Little's right hand man when it comes to running the company.

HAIGH: Little was CEO?

MOLER: Yes, Little always been CEO.

HAIGH: And what was Bangert's job title?

MOLER: Bangert was sort of chief programmer, I'm not sure he ever had any title. He was possibly secretary, there is an official Board of Directors, that never meets. I'm Chairman of the Board of Directors, which is kind of an honorary title, and Bangert was secretary of the Board of Directors for a while. Bangert's now retired and is not active in the company. He never came to Massachusetts, he worked out of his home in California for a long time.

HAIGH: And how many other people were working on product development by this point?

MOLER: By 1989 with 30 people. It's always been true that about one-third of the company works on product development, maybe a little over that, but still true today, so with that rule there were ten or 12 people working on product development.

HAIGH: Do you have a sense of what they would have been doing?

MOLER: Some were working, the addition of the graphics was an important thing, there were people working on graphics. We came up with new versions of the control toolbox and the signal processing toolbox. We added other toolboxes by outside authors, so there was a spline toolbox written by Carl de Boor at the University of Wisconsin, and a system identification toolbox written by Lennart Ljung. He just got elected a foreign associate to the National Academy of Engineering this year, he wrote the system identification toolbox back about this time, wrote a text book that went with it, and that's been a very important product for us and he received, both Ljung and de Boor receive royalties.

HAIGH: And to produce the toolbox you just use the built in MATLAB tools, you don't have to change anything in the core code?

MOLER: That's right, that's right. These toolboxes are written in M. By 1989, let's see, we hired an Englishman by the name of Andy Grace, who'd written a thesis using MATLAB about optimization in control, and so Andy took his thesis and it turned into our first optimization toolbox around this time. And then shortly after that, or near this time, Andy Grace and a guy by the name of Joe Hicklin started work on something called Simulink, and that was our second major product. That's a completely different thing than MATLAB, serving a completely different community, and that was going on in 1989. We don't need to go into Simulink, but the story here is that in 1989 we were already reaching out in different directions, tackling different fields.

HAIGH: Right. And presumably for the people who were doing the development, as the focus was on toolboxes that you would look for people who, as in the case you described, had expertise in using and extending stuff in the M language, rather than someone who's a great C hacker?

MOLER: We had both, we had both. There was certainly work being done on the internal structures of MATLAB, the MATLAB language, the graphics, those aren't C programmers, and they don't even necessarily know how to use MATLAB, or know anything about the application areas that it's used for. In other cases the people are working in M and they're specialists in the application area, signal processing, bioinformatics, whatever the applications are, and they may have never studied computer science.

HAIGH: Would most of those people have Ph.D's?

MOLER: It's not most, but it's close to most. We have probably 350 developers and there's well over a hundred Ph.D's. I'm not sure who has a Ph.D., it doesn't make much difference at MathWorks as to whether you have a Ph.D. or not.

HAIGH: Now it seems that there was quite a long gap in releases, between 1987 with Version 3, and 1992 with Version 4.

MOLER: Yes.

HAIGH: Why was that?

MOLER: Because of Rich Haney. We wanted to get serious and professional about the graphics, and we went through two versions of the graphics. Probably in 1990 we were almost ready to release MATLAB 4, with color graphics, in fact Jim Tung, the marketing guy, and I, went on a press tour and showed off what MATLAB, what the next release of MATLAB was going to be, and showed off this color graphics we were doing, and it was all fake. We didn't really have it working. We jumped the gun, and we decided at the very last minute not to release that because we felt that it didn't have any future. It was just a dead end version of doing things, and we redid all the graphics from the ground up in an object oriented way that remains today, and that took us another two years.

[End of Tape 4, Side A] [Start of Tape 4, Side B]

HAIGH: Were there interim releases between the major upgrades?

MOLER: Yes. So 3.5 was our major version for quite a while, and then there are even letters that go on that, 3.5A, 3.5B. On the IBM PC there was an embarrassing bug about what happened if their was a directory on your path that was empty, and MATLAB would try and look for the files in this directory and not find any and crash, and we came out with several letters. 3.5G, H and I were all trying to fix that bug.

HAIGH: Would customers receive free technical support after they purchased the product?

MOLER: For a while there was a difference between Unix… between Pro MATLAB and PC MATLAB. So on the IBM PC (and eventually on the Macintosh, the Macintosh version came out later) you bought the product and you had technical support, you could call us or e-mail us with questions but you didn't get updates. If you wanted the next version you had to buy it, there was a reduced price for the next version but you didn't get it automatically. On Unix machines, from the very beginning, there was actually a maintenance fee and you automatically got updates whether you wanted them or not. It wasn't too long ago that we stopped making the distinction between Unix and PC as far as our pricing and maintenance. The Unix versions, the workstation versions, were more expensive than the PC version because the machines were more powerful.

HAIGH: The capabilities of the code itself were exactly the same?

MOLER: Yes, yes.

HAIGH: Presumably that must have become much more of a challenge when moved to graphics intensive version?

MOLER: Well that's right. So there are machine dependent parts, operating system dependant parts, that do the graphics and, you know, over the years, well it's still true today, there's Windows, there's Unix with X Windows, and there's the Mac, and their graphics are different. The capabilities the user of MATLAB sees are the same, that's one

of its attractions. You can go from the PC to the Mac to the Unix work stations and see the same system, but the details of the implementation for the graphics are different.

HAIGH: And did you achieve that by rewriting the code, or by introducing an extra level of abstraction to aid portability?

MOLER: Both. The code is common as far as we can go. So at the top level the user sees something called "handle graphics," which is completely machine independent. In the internals of MATLAB, the part that's doing the scaling and processing the parameters and computing the contours and all that, is machine independent. It's written in C but it's the same on all machines, and only when you get down to actually calling on the operating system to draw lines on the screen does it becomes machine dependent. We had hoped that Java was going to be the answer to it. We were going to redo everything to make use of Java and that was going to get rid of our machine dependence, but that hasn't worked out.

HAIGH: When did graphical user interface capabilities first appear?

MOLER: Well, I don't know, I forget. Probably with MATLAB 4, 1992, probably before that, I don't remember.

HAIGH: Would there have been a text base menuing environment, between the straight DOS command line and real windows?

MOLER: It isn't the DOS command line, it's the MATLAB command line.

HAIGH: CLI.

MOLER: Yes. What is CLI… Command Line Interface. So, there's never been a menuing type system of any significance, it's always been command line or toolboxes or GUIs. The command line interface is important today, that's what I use when I'm interacting with MATLAB, is the command line a lot. I use it in class with the command line, but then I also write GUIs for the students to use to demonstrate algorithms and properties.

HAIGH: Was a graphical user interface something that customers were demanding?

MOLER: Yeah, customers were demanding, our people were demanding this is what a modern…. I'm not sure if our customers demanded a graphical user interface as much as our people though it was a good idea. We have experts in the application areas in computer science, in graphics and we know what the customers need, we're driven more by our own ideas than we are by responding to customer's request. They look to us to tell them what they need.

HAIGH: Okay. Let's rewind slightly chronologically back 1989.

MOLER: Right.

HAIGH: When you joined the company full time what was your job title?

MOLER: I don't know. From the very beginning, from the establishment of the company, I've had the title of Chairman of the Board, if wanted to use, I've never put it on my business card, I guess probably I called myself Chief Scientist, so I've said Chairman of the Board and Chief Scientist for as long as I can remember.

HAIGH: And what were your main responsibilities then?

MOLER: It's always been whatever I wanted to do or whatever I wanted to take on. I add new mathematical capabilities to MATLAB. One of the first things I did while I was working for the company but while I was still in California, was add sparse matrix capability to MATLAB. This is something that we decided we really needed that would extend its capabilities in important areas, and John Gilbert here, who we had lunch with today, and I worked in Menlo Park, in California, in my home, and added sparse matrix capability.

HAIGH: And you were personally rewriting the core C code?

MOLER: No, oh yes, yes. No this wasn't rewriting core C code, this was adding another section of core C code, because it was adding something to the system it wasn't replacing something that was already there.

HAIGH: Right. So, would the user have to explicitly say, "treat this as a sparse matrix," and then it would just use a separate set of subroutines?

MOLER: Yes, that's correct. You know, I added special functions, gamma functions and beta functions to MATLAB, the ordinary differential equation solvers were my area of responsibility for a while until we actually hired people to work on that. For symbolic computation for working with formulas and doing calculus, we set up a partnership with Maple, and we actually use the Maple symbolic engine to do symbolic computations. I wrote the first version of the interface between MATLAB and Maple. I wrote the manual for that. I've written a lot of the documentation, an introductory user's guide I wrote.

HAIGH: How would you compare the satisfactions that you get from programming with the other kinds of work that you've done?

MOLER: I love to program, I'm a real hacker, I get a lot of satisfaction out of it. I will stay awake all night tackling some programming project and I won't do that with, well with anything else. It's very satisfying. It's puzzle solving, and I like to do that.

HAIGH: Is there any one piece of code that you're particularly proud of?

MOLER: Oh…. In this textbook I've written, there's a lot of code associated with the textbook, and some of them are expository M-files. They're programs that are supposed to be read by students to understand how the algorithms work, and so the most important concern is making them readable and understandable by students, and I'm proud of what

I've achieved there. It's not a usual criterion for programming, to make it readable, but that's what I've done there.

HAIGH: As the scope of the product grew, did that mean that you had to change the way you worked?

MOLER: A little bit. As the scope of the product has grown, and MATLAB is now used in areas and used in things that I know nothing about, in engineering design and circuit design and the automobile industry and so on, and the whole Simulink thing, that's a whole side of our business that I know very little about and have very little to do with. I'm still involved indirectly because those products call upon the mathematics in mainline MATLAB, so I'm responsible for the core that these guys are built on top of, but I have nothing to do with the superstructure.

HAIGH: Has the percentage of developmental devoted to the superstructure increased over time?

MOLER: Oh yes, that's right it has.

HAIGH: So, was there any particular development that played a big part in moving use beyond control engineering?

MOLER: Well it's our understanding of other fields, and our acceptance in other fields. One very important event was when a research group from Mercedes Benz (it was not yet Daimler Chrysler, but it was the Mercedes people from Stuttgart), visited us in Natick. To back up, I mentioned Matrix X as this product that was built on the original Fortran MATLAB by ISI, they were our competitors in the control business and in the simulation business for a number of years, and they were very successful at that. They had large scale contracts with automobile manufacturers in the U.S. and in Germany and in Japan, and their product was more expensive, it didn't run on PC's, it only ran on work stations, and we were always playing catch up to those guys, intense competition. An important event was when a research group from Mercedes visited us, and then visited ISI, and decided to use MATLAB and Simulink for the next project, rather than the competitors. And that happened a couple of times. It happened at Toyota in Japan, I wasn't involved in that one, but the Mercedes one I was involved in.

HAIGH: So that represents a shift in the user base proportionately away from classrooms and more into heavy duty industrial use.

MOLER: That's correct, yes, yes. It also happens in academia, recently I was involved in getting the Air Force Academy in Colorado Springs, to adopt MATLAB for all cadets at the Air Force Academy. So beginning this year when students come to the Air Force Academy they get a PC and MATLAB is installed on that PC for all students. I was involved in talking to the Air Force about how they could use MATLAB in their classroom.

HAIGH: So, was control engineering the first big application area embracing MATLAB?

MOLER: Yes.

HAIGH: Would there be any area you could point to as the second area?

MOLER: Signal processing.

HAIGH: And anything after that?

MOLER: Image processing, which is two-dimensional signal processing. Optimal control, the control problem is to choose the parameters in your control circuit to minimize response time and maximize stability, and originally you do that by hand, by trial and error, but then it can be programmed and then that becomes optimal control, so we expanded in that way. We realized that MATLAB was being widely used on Wall Street in financial firms for modeling their financial instruments, and so we have introduced a line of products specifically for the financial industry, there's a financial toolbox and a financial time series toolbox and things like that. Our latest effort is in bioinformatics and biosimulation, the human genome project and so on. MATLAB is used in that and we now have toolboxes in that area.

HAIGH: And were all those things achieved purely by adding toolboxes, or did you have to re-implement some of the core areas?

MOLER: We have reimplemented some of the core areas, so we have extended the MATLAB language. I guess it was with MATLAB 5 that we introduced an object oriented approach, where you could create new data types and overload operators associated with those data types, and create your own structures and methods on things that MATLAB had never imagined. The connection to Maple and for symbolic objects is done that way. So the core MATLAB was extended to provide for objects and then in the MATLAB M language I can now write the connection to Maple without making any additional changes to core language.

HAIGH: Sounds like you've been motivated by a desire to increase flexibility in applications in specific areas but it's not closely tied…

MOLER: That's correct, and it's also motivated by progress in computer science, this is what the world was doing with C++, and Java, and so on, and we make those kind of changes to MATLAB.

HAIGH: So, you mentioned that in the case of Wall Street they'd actually taken up the product and begun to use it, and then you followed that with a toolbox that was designed to support them. Are there other examples where…

MOLER: The bio thing is another example of that, people in pharmaceutical companies and the genetics companies, and so on, have been using MATLAB for a long time without our particularly paying attention to them except as regular customers, and now we've added new capabilities aimed at their area.

HAIGH: Have there been any kind of formal programs to get feedback from users and to discover new opportunities?

MOLER: Yes, yes there are. We had a couple of MATLAB user conferences years ago and then we decided that was not effective because the people that came to the MATLAB users conference were already MATLAB groupies who were already using MATLAB. We wanted to meet new people, and the users conferences weren't doing that. We now have more specialized conferences, so we'll go to Detroit and hold a workshop on the use of MATLAB in automotive, and we have set up advisory councils of representatives of some of our most important customers, these are big time MATLAB users that influence other users at their company. In Detroit, we have an automotive advisory council that meets there and we meet with them. And then we have an aerospace advisory council that's met in Washington, and in Natick. There's people from Lockheed, and Government agencies, and so on, that are a part of that. These are people who use MATLAB as their main tool in their professional lives, and they influence many other people at their company, and they're sort of internal consultants and evangelists for MATLAB at the places they work.

HAIGH: Have you ever found cases of users becoming fanatical about pushing one particular idea?

MOLER: Yes there are some weird MATLAB customers… these are real…. Listen, we don't need to get involved in stories about that.

HAIGH: Alright. Getting back to the issue of competitors then, you've mentioned that in the early days Matrix X was in some ways more polished and powerful.

MOLER: Yes.

HAIGH: What have other important competitors been?

MOLER: Well, what's interesting about MATLAB is that it's used in many different fields and it has different competitors in different fields. So, in the mathematics business, which is my business, Mathematica is a competitor of ours. In the control business Matrix X was, and a product called D Space is now. Both of those are competitors of ours, but they're not competitors of each other. Mathematica and Matrix X never had anything to do with each other. We're very much involved now in electronic circuit design and there's a company called Synopsis, for example, that's a big player there. In the image processing business there's a language called IDL, and the company was called RSI, I think, Research Systems. A couple of years ago they were bought by Kodak, and they were interested in processing satellite images in aerial photographs and so on. Our biggest competitor is C… our biggest competitor is people writing their own programs, when they could be using stuff that we've already written.

HAIGH: And how to you go about persuading them that this isn't something they should do?

MOLER: It's our job. As I say it's word of mouth, our sales people are very low pressure, they'll call these guys, they'll talk to them, they'll call them several times, send them demo copies, send them free copies. Sometimes it's a long process to sell MATLAB to some of these people who don't want to change they way they've been computing for a long time.

HAIGH: Do you think that's because they find working at a lower level more enjoyable?

MOLER: No, I don't think it's more enjoyable, it's just habit and inertia. I do the same thing, with some of the tools I use on my computer. I'm very old fashioned, very out date, with my editor and my mail system, and so on, but it's too late for me to change.

HAIGH: How would you distinguish MATLAB from Mathematica?

MOLER: Mathematica is primarily a symbolic computation system, it manipulates formulas, it knows calculus, the internal data structures are lists. MATLAB does not know anything about calculus, it doesn't know what the derivative of sine X is, it can't even pose that question. In order to do that, it uses our connection to Maple, and one of the reasons we did the connection to Maple was to compete with Mathematica. Mathematica is one man's vision of how computing ought to be done, that's Steven Wolfram, one of the smartest guys I know, and he's designed this system almost completely by himself. MATLAB is a bit of a hodgepodge, particularly with respect to our toolboxes. Different toolboxes are written by different authors and they have quite different styles, and it's really quite an eclectic collection of stuff. Also, our company is three or four times as big as Wolfram's, we're a much bigger operation than Mathematica. These kinds of things we've been talking about with engineering design, the automobile industry, the aerospace industry, Mathematica has very little presence there, and that's where we make most of our money.

HAIGH: Is that because those problems require numerical methods?

MOLER: Well, it's because the symbolic computation doesn't work there. Mathematica has numerical capability as well but it's not nearly as powerful as MATLAB's, and then this whole history we have about being active in that industry is really important.

HAIGH: So, Mathematica would be used more in classrooms and by research mathematicians?

MOLER: Yes, and by physicists. Wolfram is a physicist, and physicists use it.

HAIGH: Is that because they have different kinds of problems from engineers?

MOLER: In part, it's also just a cultural thing, they know other physicists that are already using it. The relativity business, there's a little niche market, Mathematica has a lock on the relativity business. We have nothing to offer them, don't want to.

HAIGH: And how about Maple, you've mentioned that. Is that a competitor more than a partner?

MOLER: Well it's both… and it's a tenuous relationship. Maple is a Canadian product developed by a couple of professors at the University of Waterloo, it's about as old as MATLAB, some of its earlier history is similar to MATLAB's, but it's still very small, there's only, there's under a hundred people working for the Maple company, they probably do a tenth of the business that we do. They have a very mathematically sophisticated system for doing this kind of symbolic manipulation, their system is more powerful mathematically than Maple, than Mathematica's. Mathematica has other things that are better than Maple in terms of user interface and so on, but Mathematica and Maple compete with each other strongly, and primarily in the academic markets. The Maple company would like to be like MathWorks, but they don't have the products or the connections to achieve that.

HAIGH: And how about the strengths and weaknesses at MATLAB when it's competing in these other areas, like circuit design and image processing? What do you get from being a much broader based kind of product, and what are the limitations of that?

MOLER: You get programmability, you get flexibility. If you have a canned system, a GUI based system for doing anything, image processing, circuit design, Wall Street, it'll do what it does and that's all. If the designers of that system didn't think of your particular aspect of the problem, you can't change it. MATLAB can start off with a GUI based toolbox that does the basic problems in a field, and then customers can extend it themselves, by writing programs in the MATLAB language.

HAIGH: And are customers able to see the code inside the toolboxes?

MOLER: Yes, yes. The toolbox code is shipped with MATLAB and you can see it and read it and modify it.

HAIGH: But I assume right from the beginning that they don't get the C code?

MOLER: That's correct, that's correct.

HAIGH: Not even the first version.

MOLER: So, half of MATLAB is written in MATLAB and that's shipped with the product. So half of MATLAB is open source.

HAIGH: And do you think the MATLAB approach brings with it any weaknesses when computing against these more specialized applications?

MOLER: Yes: it can be slow. So because of the math interpretative nature of the MATLAB. So now we switched from, I'm switched from thinking about user interface and GUIs, and so on, to talking about large scale high performance computation and there MATLAB can be much slower than something written in C or Fortran. That's one of the things we're trying to address, but it's going to be a long battle.

HAIGH: So this system is still interpreted?

MOLER: Yes. When you get down to doing a matrix computation of a differential equation, those libraries are written in C or Fortran, and highly optimized. So if you want to invert a matrix, or compute eigenvalues, the MATLAB program that does it is faster than anything you'll be able to write yourself. But if you were to write something like that in M, then it would be slower by orders of magnitude.

HAIGH: So are there an important class of real world applications to date that with modern computer hardware would run too slowly on MATLAB and would be doable with other methods?

MOLER: Yes. We have never tried to really be part of real high performance scientific computing, the kind of stuff that's going on at the Government labs in the weapons programs, for example. We don't come near touching that. We don't a have a parallel version of MATLAB, that's something we're talking about. I've written papers in the past. One of the Cleve's Corners is about why there isn't a parallel MATLAB, that's an old thing that we're rethinking now, and the future of high performance computing is certainly in terms of parallel computing, so that's something that we're going to do more than we've done in the past.

HAIGH: Okay. I think we should probably move now then to consider the last ten years. You've touched on that already. How has your personal role changed over time?

MOLER: I've been with MathWorks now, well in a sense I've been with MathWorks since its beginning, but I've worked for them for 15 years now. I'm going to be 65 years old in August. I'm not exactly going to retire, but I'm out of the main line software development. I've never been involved… well, for a while I sort of was somewhat involved in the management of the company. There's an executive management team that meets weekly, and I used to attend those meetings, but I was too disruptive. When it came time to deciding personnel issues, distributorship issues, financial issues, I didn't really have any expertise in those and didn't contribute much to the discussion, but because I was such a demonstrative guy and because I was one of the owners of the company, if I spoke up at those meetings I was disruptive. So I stopped participating in the management side of the company, and now I find I'm also doing that with the software development side of the company. We have good young people, who I helped hire, that are working on the mathematical part of MATLAB, the matrix guts of MATLAB. I don't always agree with what they're doing, but I'm going to let them do it. It's time they have responsibility for that. I give them advice when they ask for it, and they may or may not take my advice.

HAIGH: With the company growing so fast, were there particular challenges in instilling a corporate culture and keeping people working productively together?

MOLER: Sure, that's always a big concern, and we have people in our human resources group, our personnel department, whose job it is to worry about those kind of things. We have team building exercises, groups go off and climb on ropes strung between trees together, go on outings together go camping together, and those are all intended to team building within the group. With a thousand people in the company it's hard to have that

feeling throughout the whole company. We have two big company social events every year, a summer outing and a Christmas party, that involve our employee's families as well as the whole company. In May we're going to have this expedition to Florida, to celebrate our twentieth anniversary. This is not only going to involve everybody from Natick, but we have recently acquired our European distributors and started a subsidiary in Korea, so we have a couple hundred employees offshore, and we're going to bring all of them to Disneyworld along with everybody from Natick, and that will be the first time we've had all these people together, and that's intended, you know, to build a company wide team.

HAIGH: Do you think that the same kinds of people would come to work at MATLAB today that would have been hired ten years ago, or do you think that's changed as the company's grown larger.

MOLER: That's changed. For some people at MathWorks it's just a job. They've worked at other companies, they'll work at other companies in the future, it's a good programming job, and they'll be there for a while. Some of the old timers have been there for a long time, this has been their life, they have a different loyalty, different feeling towards MATLAB the product, and towards the company, than some of the more recent people.

HAIGH: Is the company still privately held?

MOLER: Yes.

HAIGH: Was that a deliberate decision?

MOLER: Absolutely. If we were to go public it would change the company culture a lot. We don't have an outside Board of Directors, we don't have anybody telling us what to do, we don't have to worry about the stock price. You look at publicly held companies and every quarter they have to predict how much money they're going to make that quarter, and if they're below the prediction then their stockholders say they're in trouble. If they're above the prediction, the stockholders say they don't understand the industry. So, we don't have to do that, we're not beholden to quarterly sales numbers and that's a good thing. We have a very aggressive profit sharing plan, where anybody who's been with the company six months shares in profits every quarter, and that can be a significant addition to your official salary. So, that's our substitute for going public and having some of our employees… if we went public some of our employees, not all of them, would have stock options that would get them some money, but we're not going to do that.

HAIGH: And how strong would you say the company's ties remain with academic culture and values?

MOLER: Pretty strong, very strong. You know, I'm a part time professor, I visit academic institutions a lot, I'm still active in that community, and we have other people at the company who have been professors. The head of our image processing group is a former professor at the University of Illinois, for example. We have a couple of people

who still have academic ties to schools in Europe and in England, we have a fairly strong ties with Cambridge University in England.

[End of Tape 4, Side B] [Start of Tape 5, Side A]

HAIGH: One thing I realized I didn't mention earlier with competitors: has there been anything like an open source competitor to MATLAB?

MOLER: Yes, there are. There are several MATLAB clones, small versions of MATLAB that have open source. One of them is called Octave, developed by a guy that used to be in Texas and now is in Wisconsin, his name's Eaton, or something like that, I think it's John Eaton. That's a public domain, open source, free MATLAB derivative. It's not exactly the same as MATLAB, it doesn't have the graphics capability of MATLAB, it doesn't have all the toolboxes, doesn't have any of the toolboxes I think, but as far as the base MATLAB that's a free version of it. There's a very inexpensive version, just a hundred dollars or something like that, called O-Matrix, by a couple of guys at the University of Washington. There is an official French government sponsored version called Scilab, out of INRIA the French research organization, and they also have a free version of something like Simulink as well. I'm not very worried about those, I think they get us as many customers as they take away. I think if people try the free version, and get used to this kind of computing, that they'll buy the real thing eventually. There's a small version that works on Palms as well, from a guy in France or Switzerland.

HAIGH: So there's nothing that you've had to do so far to strengthen your position against these, or to make any changes?

MOLER: No, no.

HAIGH: Alright. How about, briefly, any key developments to MATLAB as product since the 1992 version which added graphics? Anything you thinks stands out?

MOLER: Oh, sure. Well, to the core MATLAB adding objects was a big event.

HAIGH: Right.

MOLER: Hooking up to Java was an important event, so you can write Java methods and use them directly from MATLAB. We make use of that, some of the auxiliary tools are on MATLAB, like the editor and the debugger and the connections to database are done in Java. And then outside of the core MATLAB, the introduction of the additional toolboxes are all important developments, and a user of the MATLAB product may not be aware, and certainly doesn't care, whether the tools he's using are written in M or written in C. So the big effort in image processing, for example, all of that is in M, but that's an important addition to MATLAB. And then, for the company, the whole Simulink code generation, real-time workshop, that whole thing is a big development for the company, but it's not main line MATLAB.

HAIGH: Would you like to talk a little about that?

MOLER: So, computer aided design, whether or not there's much mathematics in it, is involved in the design of all kinds of products. One of our demos involves the lifting mechanism for the window in your car. If you have a high end car… if you have electric windows on your car and you push the button the window goes up, but you have to hold the button down. If you take your finger off the button the window stops, high end cars you can click the button and let go and the window will go up, well that's dangerous because your hand or your head could be in the way so something has to detect that and stop the window. That's done by measuring the current that's going into the motor that's lifting the window, and there's actually a Motorola chip that controls this whole mechanism. When you push the button to make the window go up it sends a signal to a chip that starts this motor and lifts the window. That whole system is designed with MATLAB and Simulink, and the designer of that system will model it in Simulink, and then when he's happy with his design he pulls down on a menu item and MATLAB will actually write the code that goes into the Motorola chip that's eventually in your car. There are hundreds of examples of that, so Simulink doesn't run in your car or in your cell phone or in your disc drive, but control code generated by MATLAB does. And that's a long way from the matrix laboratory that I was playing with 25 years ago.

HAIGH: So, the MATLAB/Simulink combination would be used primarily for that method of embedded application?

MOLER: For generating the code for the embedded application, yes. For designing the system and then generating the code.

HAIGH: And when did that capability first become available?

MOLER: Several years ago, it's been a constantly improving thing. One of the key criteria is that the embedded code should be small, should be compact, because there's not much memory in your cell phone or your car door. So one of the goals of this group is to have MATLAB generate compact code and that's always a moving target.

HAIGH: So that's the second major product for the company?

MOLER: The Simulink thing, and its add-on code generation.

HAIGH: Had people ever suggested earlier that you should bring out a larger number of products? Were there pressures internally?

MOLER: That's always been our goal.

HAIGH: But you've achieved it expanding MATLAB through toolboxes.

MOLER: Yes, but the Simulink product was also a natural to do because this block diagram language is the language that's used by system designers, control systems, electronic systems. If you look in textbooks on these subjects you won't see differential equations, you'll see block diagrams, and that's the Simulink language.

HAIGH: But it does seem that MathWorks is quite distinctive in having been successful, grown rapidly, remained independent, and until recently stuck basically with one product, as compared to many software companies that have tried to diversify and bring out less closely related products in different areas. Was that a deliberate policy of sticking to what you were good at?

MOLRER: Well I think so. I wouldn't say that it's one product, it hasn't been for a long time. So, Simulink has been there for a long time and people using Simulink might not even know they're using MATLAB, they think they're just using Simulink, so it's been at least two products for a long time. And then if you look at our price sheet there are 60 or 70 products on there, with all the toolboxes.

HAIGH: Right. Product is not the word then, but for a long time only one product family.

MOLER: It's one product family, yes. Well, a lot of other companies that I know about have one product, or one product family. The finite element guys, NASTRAN, ANSYS, those are one family of products. Parametric Technology is another company nearby us, that we're sometimes compared with, though they're actually bigger than we are, that only has really one product family.

HAIGH: So, in the scientific software area this is more common?

MOLER: Yeah, I think that's right.

HAIGH: If you think of something like the mainframe packaged business and system software industry, that pretty much consolidated that down to Computer Associates.

MOLER: Yes.

HAIGH: Do you think there's a deeper reason why that didn't happen in scientific software?

MOLER: Well how did it happen? Computer Associates is what it is because it kept acquiring companies.

HAIGH: Right. But something must have been different about the scientific ecosystem, so that companies could remain independent…

MOLER: Didn't have to be acquired –

HAIGH: …grow organically, stick with single product families, run along perfectly well.

MOLER: Yeah, that's right. Yes. [long pause] I can't think, there's not a technical equivalent to Computer Associates, is there –

HAIGH: Not that I'm aware of.

MOLER: No, I don't think so. Thank God. [laughs]

HAIGH: So, the range of platform support. So, you said from its very early stage it worked on DEC, and it worked on Sun, and obviously on PC's.

MOLER: Yes.

HAIGH: Macintosh was added relatively soon?

MOLER: Macintosh took us a while to add. We had it for a long time, then we dropped it when it looked like the Mac was dying, and when it was still its own operating system, and was odd man out in our development operation. Then when Mac didn't die, and when they became a sort of Unix workstation, we went back, we were welcomed with open arms in the Mac community.

HAIGH: Any other platforms that proved to be particularly important?

MOLER: We've always been on most viable platforms. At our height, when there were lots of Unix work stations, we had over a dozen platforms with Apollo and Convex and those kind of things. We had a Cray version at one time.

HAIGH: Did that sell?

MOLER: We sold a few of those. But right now it's down to six or seven: PC, Mac, Sun, Hewlett-Packard, Silicon Graphics, the DEC Alpha. We have just announced our latest product, which is MATLAB on the Itanium running Linux, 64 bit, so that's a new platform for us. I don't think that has a future, I think Intel is going to give up on the Itanium.

HAIGH: Do you think you'll produce something for the 64 bit AMD?

MOLER: Yes, I think we will. Because Intel has just announced that they're coming out with their own 64 bit Pentium family processor.

HAIGH: More generally, what do you think the key challenges and opportunities would be in the future?

MOLER: My immediate concern, what I'm thinking about, is parallel MATLAB, and some kind of Beowulf cluster kind of MATLAB capability. We've just scratched the surface in the bio simulation, bio informatics business. The big competitor there is public domain software. That's a new field, and you can't say it's a strong tradition, but there's a very strong movement for public domain, open source software in that business. And then as I say, people in all kinds of areas that are still writing their own C, or their own Fortran, we want to make MATLAB execute as fast as programs written in C, and we have recently hired a group of top flight computer scientists and compiler guys to work on that, but it's a big job and it's going to take a while to do that.

HAIGH: That would be a just-in-time compiler kind of approach?

MOLER: Yes, that's right, that's right. With MATLAB 6.5 we had the beginnings of a JIT, and that development is very actively going on at MathWorks now.

HAIGH: Do you think that the kinds of ties that you had originally with the broader numerical analysis community are going to be equally important for the company going forwarded?

MOLER: I'm hesitating because I think they're less so. The kind of work that I know about, that's going on in the numerical analysis community, isn't going to find its way in MATLAB, I don't think. I almost want to say that the math part of MATLAB is finished.

HAIGH: Why is that?

MOLER: Because we have the basic capabilities we need, and with our matrix computation and our differential equation computation, and we don't need to embellish it any more. We've got to do parallel versions of those things, so there's a lot of work to be done there and some of my friends in the numerical analysis community, like John Gilbert, we're going to be working with them over the next few years about parallel versions of these basic algorithms. But… I don't see what else we need there. The general area of large scale partial differential equations, computational fluid dynamics, computational chemistry, quantum chemistry, those are serious areas in technical computing where there are both commercial and public domain software, and we don't touch them. The most widely used program on super computers is a commercial program called Gaussian, which does quantum chemistry, and we don't have anything like that, and that's a different kind of mathematics than what we do. But it's not a big enough market and those guys already have it, we don't have any connections to that community, you know I don't see that there's any reason for us to go there. [1]

[1] On March 8, 2006 Moler added the following comment by email after reviewing the transcript: "I've been thinking about my comment that nothing much more needs to be done with the mathematical features in MATLAB. I'd like to revise that comment. It doesn't sound right and it isn't true. It's true that the traditional features that I've worked on, like basic matrix computation, are mature. But there is a great deal of mathematical development in progress and planned for the future. As a company, we are broadening our mathematical interests and expertise. Current work includes:

HAIGH: So, as far as this broad audience of practicing scientists and engineers and financial analysts, goes, the mathematics they need is all there already?

MOLER: Yeah, except in some specialized areas, yes. [Pauses] Let's come back in a few years and see if I'm right. [Laughter]

HAIGH: Well, that's an interesting note to finish on. So, if you were today in the same kind of position you were back the 1970's, what kind of opportunities or interesting areas do you think you might be drawn?

MOLER: I don't understand, what are you getting at?

HAIGH: Well, back in the 70's when you were working on LINPACK and EISPACK, and what became the first creating of MATLAB, those was clearly an area where there was a lot of work to be done, those tools were clearly very much needed –

MOLER: Oh, are there any comparable areas today?

HAIGH: Yes. In the case of EISPACK you were taking the cutting edge work, bringing it to the masses.

MOLER: Well, in mathematics, in mathematical computing, I don't know of any or else I'd be putting them in MATLAB. So… some of these things other people can put into MATLAB, so there's the Fast Multipole algorithm, and the multi grid algorithm, for example, in certain kinds of matrix computation, and we don't have those in MATLAB. But other people have written those in MATLAB, and we have something called MATLAB Central, where you can go to our web site and find thousands of MATLAB programs that other people have written. I don't know whether there's a Multipole Algorithm or multi grid program there, but there might be. But that's not mainstream, that's not that important. It's pretty hard today to crack the barrier of, saying "this is of wide enough importance that we ought to make it part of mainstream MATLAB." It's reached a certain critical mass and it shouldn't get any bigger.

New optimization techniques
Genetic algorithms
Mixed integer/continuous linear and nonlinear programming
More powerful statistical computations
Bioinformatics
Biosimulation
Control in the presence of uncertainty (robust control)
Image morphing
Filter design

We may not be inventing new mathematics or new mathematical algorithms, but we are making mathematical thinking and mathematical techniques accessible to scientists and engineers who might not encounter them otherwise."

HAIGH: Well, unless you have any other thoughts –

MOLER: No, this has been a fascinating time. I've really enjoyed thinking about these things, and it'll be very interesting to come back in a few years and listen to this.

HAIGH: Thank you very much for taking the time.

MOLER: Well, thank you Tom.