CHAPTER 1

The Story of HydroLight

I began working on what is now called HydroLight in 1978 when, by almost divine providence, I was given an office next to Rudolph W. Preisendorfer when I arrived at the NOAA Pacific Marine Environmental Laboratory in Seattle, Washington as a new post-doc in late 1977. Oil tankers were entering Puget Sound in large numbers bringing crude oil from Alaska, and the possibility of a spill was on the minds of many. My Ph.D research had been in computational fluid dynamics (my dissertation was a numerical model of Langmuir circulations), and my proposed research project was to develop a numerical hydrodynamic model to predict the movement of spilled oil in Puget Sound. PMEL's charter at that time was to study the "meteorology and oceanography of the Pacific Northwest." My proposed oil spill model would combine both meteorology and oceanography as the winds and currents move the oil from its spill location to the nearest pristine beach. However, in one of my first encounters with government bureaucracy, the day after I arrived at PMEL, I was literally forbidden to work on an oil spill model. NOAA senior managers had gotten wind of my proposed research and decided that oil spill modeling was "fluid dynamics," which has to be done at the NOAA Geophysical Fluid Dynamics Lab in Princeton, New Jersey. My advisor told me to find something else to work on and disappeared.

Preisendorfer heard the that new kid liked to do numerical modeling and was looking for a project. He walked into my office one day in early 1978 and said, "I hear you like numerical modeling and that you're looking for something to work on. You should consider hydrologic optics." I had never heard of either Preisendorfer or hydrologic optics, and little did I realize that I had just changed careers.

Rudy was a pencil-and-paper mathematician who applied his math to what he called "hydrologic optics," which encompassed all of what today is called optical oceanography, ocean-color remote sensing, underwater visibility, and related topics in radiative transfer theory. One of his goals in life was to place hydrologic optics on a firm mathematical foundation. He accomplished this and much more in his magnum opus: the six-volume, 1757-page treatise *Hydrologic Optics* (Preisendorfer, 1976). That work developed the mathematics of invariant imbedding theory as applied to the solution of the scalar radiative transfer equation (RTE). That math is the core of HydroLight.

Anyway, Rudy laid several hundred pages of hand-written notes on my desk and said, "Turn these equations into a computer program." For the next few years I was basically his programmer.

It was a pleasant job. He was a very humble and gentle person who guided me through the process of learning radiative transfer theory, and we worked together on the numerical analysis problems that inevitably arise when continuous mathematics must be discretized for solution on a computer. He took care of my funding by carving out my modest salary from his internal NOAA funding for climate prediction research, which is what he was supposed to be working on as a PMEL scientist. He spent most of his time developing new statistical techniques for climate forecasting. (This was during the early days of recognizing the importance of El Niño/Southern Oscillation for global weather patterns.) I kept a low profile within PMEL and worked mostly on developing the code for solving the RTE. We never received a penny of funding for hydrologic optics research or code development, which in theory we were not doing. After several years of work, we finally had a computer program that could compute underwater radiance distributions with great accuracy and computational speed (relative to other techniques such as Monte Carlo simulation). Just as things were getting scientifically interesting as regards applying the computer model to problems in optical oceanography, Preisendorfer died unexpectedly of a heart attack in 1986 at age 58.

Preisendorfer was the most senior scientist in NOAA who still did science full time, and as a tenured civil servant he was somewhat immune to various political pressures. However, he was never able to get me a civil service appointment at PMEL, and I lived from year to year. The PMEL lab director had no use for either hydrologic optics or me, and he called my computer programming "blue-collar science," which was inherently unworthy of a Ph.D. (To be fair, the lab director foresaw the importance of global climate change and was trying to position PMEL to be at the center of research on that problem. In his view, my office would be better used for someone who wanted to study El Niño.) Without Preisendorfer's protection and funding, my days at PMEL were numbered. I tried to obtain funding from the U.S. Office of Naval Research to continue Preisendorfer's work. One ONR program manager, who admitted that he hadn't even looked at the proposal I sent him, told me, "Who cares what he was working on, he's dead." Another one told me, "Trying to live off of a dead person is a poor way to make a living."

I then sent letters to several senior scientists who had worked with Rudy and knew the importance of his work. I asked for suggestions on how to find funding to complete our various half-finished projects. Fortunately—another almost divine intervention—Ray Smith knew that ONR was in the process of establishing a new program in Ocean Optics, to be run by Richard Spinrad. Smith realized that a lot of unpublished work by the preeminent theoretician of the field was about to disappear along with me. To make a long story short, Smith called Spinrad who called ONR and told them to give me a year of funding. I was two weeks away from unemployment when I got a phone call from the ONR Ocean Biology program saying they would give me a year of funding. The old-boys' network had saved the day! Rudy had been working on another book, which at his death was just a six-inch-high stack of handwritten notes. I used my year of ONR funding to compile those notes and get published Rudy's text *Principal Component Analysis in Meteorology and Oceanography* (Preisendorfer, 1988). I also published several papers that we had underway. Finally, I wrote a long technical memo (Mobley and Preisendorfer, 1988) that documented the mathematical algorithms used in HydroLight.

After that year of ONR funding I found a temporary job for 18 months as a sabbatical replacement physics professor at a small liberal arts college. During that time I published a paper (Mobley, 1989) describing HydroLight (still unnamed), but did no further work on the code. However, Spin-

rad was moving up a notch at ONR and encouraged me to apply for his soon-to-be-vacated job as the Ocean Optics Program Manager. I was selected and became the second manager of that program beginning in 1989. While at ONR I had no time for further scientific work with HydroLight, but I did continue desultory writing on what would eventually become *Light and Water* (Mobley, 1994). Rudy and I had started that book, and his intention was that it would be, in effect, a seventh volume of *Hydrologic Optics* on numerical techniques. I expanded the subject matter to make it of more general interest to optical oceanographers.

My years at ONR were a combination of learning a lot of interesting science done by the people ONR funds, pointless paperwork, and really nasty behind-the-scenes politics. As President Truman once said, "If you want a friend in D.C., get a dog." ONR was, shall I say, an education into how science really works. Seeing science from the funding-agency side is an experience every young scientist should have. I hated to leave the science part of ONR and my tenured, hard-money civil service job (my first real job ever), but a couple of years of the politics was enough for me. I'm not a schemer or someone who "enjoys a good fight," and I have no doubt that I would have died from the stress within ten years. I left ONR in 1992 and went to JPL for two years, where I had worked out a deal to allow me to finish *Light and Water*.

The original version 1 of HydroLight that I developed at PMEL was run on a CDC 6600 mainframe computer. While at JPL I rewrote the code as needed to run it on a Unix workstation (version 2). It was all incredibly primitive by today's standards, with everything done via the Unix command line, but I was able to begin doing science with the code and publishing papers that used HydroLight.

As people began to see papers using HydroLight results, they asked for the code. At that time, in the mid 1990s, I gave out the code for free, with the request to be included as a co-author on papers that used the code. One of those users was Curtiss Davis, then at the U.S. Naval Research Lab. He was able to get me some NRL funding to add inelastic scattering to the code (Raman scattering by water and chlorophyll and CDOM fluorescence), which he needed for his research purposes. In the fall of 1996 I landed back in Seattle at Sequoia Scientific, Inc., a small instrument-development and contract-research company. Once there, I developed a version (3) to run on personal computers (PCs at that time were running Microsoft Windows 3.1). The user interface was just a question-and-answer program written in Fortran. However, the ONR Ocean Optics program, then headed by Steve Ackleson, gave me the funding to develop a graphical user interface for PCs. These two small contracts for the addition of inelastic scatter and the development of a GUI for Windows are the only funding I have ever received for the development of HydroLight. All other development was either piggybacked on Preisendorfer's climate research or, later on, was paid for by revenue from the commercial version.

As the code became ever more popular, users asked for more and more new features, and more of my time was required for new development and user support. Of course, I think it would be nice if some funding agency simply gave me $100K a year to pay my time for training, user support, and the development of new versions, in which case I could put HydroLight on the web and give it away for free. However, that's not the way the system works. Since I received no funding for such work, my only choice was to commercialize the software and let it pay its own way. This is a common procedure in U.S. science; e.g., a scientist develops an instrument for his or her own research and then commericializes it to support making it widely available.

I needed a good name for the commercial version of the code, and I came up with "Hydrolight." The name "Hydrolight" was already trademarked in the U.S. by a company that makes hydrogen-powered lights. During the review of a paper, the Optical Society of America requested that I decide on an "official" spelling for the code (some users were calling it Hydro-light, hydrolight, etc.). I then decided to capitalize the L and call it HydroLight. I repackaged everything to make it more user friendly, wrote a Users Guide and a Technical Documentation, and began licensing the PC version of the code as HydroLight version 4.0 in late 1998. At that time PCs were taking over the science world and there was decreasing interest in the Unix version, which I quit using myself once I got to Sequoia.

I had to make a number of decisions for the commercial product. I decided that I would continue to provide users with the source code because some users of the previous Unix version had been modifying the code for their own purposes (creating new IOP models, reformatting the output for their needs, etc.). It was unheard of in those days for a company to release source code, and of course the "intellectual property" lawyer I consulted told me that I must carefully guard the source code. However, I decided to rely on the honesty of the licensed users not to pirate the code, and I have been impressed with peoples' care and integrity in safeguarding the code in accordance with the license agreement. (I am always happy to listen to lawyers, but I often ignore their advice.)

Since I was giving out the source code, I also decided to provide a compiler. That way I would need to support only one compiler (at that time, even Microsoft had a Fortran compiler). I chose the Lahey compiler, which I had been using myself for the PC version and which had performed very well for me.

I of course had to decide on the cost of a license. However, I had given the license fee no thought at all, and I certainly hadn't done any "market research," when one day I got a phone call from a consulting company that had heard that I was going to commercialize HydroLight and wanted to know the cost of a license. In a split second I just came up with $10,000. The guy on the phone didn't hesitate a bit at this price, and I made my first sale. I later lowered the license fee to $9,995 at the request of a U.S. Navy scientist who wanted to license the code. He would have to do extra paperwork for anything costing $10,000 or more, but he could do a "sole source" procurement for something that cost less than $10,000. I have not raised the price since the first copy was licensed.

I also decided to charge once for a license and then to provide user support and future upgrades at no additional charge. I personally resent companies that charge exorbitant annual "maintenance" fees for their software, and that stop supporting older versions of their software in order to force users to by a newer version every few years. By giving out free upgrades I would have to support only the current version since no one would have any reason to run older versions. Some users have now been receiving free upgrades to HydroLight since they licensed version 4.0 in 1998. It has turned out that this policy has been the best advertising I could have done for HydroLight. The code does indeed need to pay its own way, but I'm equally interested in seeing it used for science. Although a couple of people—who have hard-money faculty positions—have criticized me for charging for the software, most users (and funding agencies) understand the economics. Those who do not are welcome to write their own code, starting with the equations in *Light and Water*.

HydroLight started out as nothing more than a post-doctoral research project. I never dreamed back in 1978 that I would have a career in optical oceanography tied closely with that code. Even when I commercialized the code in 1998, I expected that everyone who wanted it would get it the

first year, and that would be the end of it. I am astounded that the product now has over 200 licensed users in 30 countries on every continent except Antarctica. Hundreds of papers have now been published using HydroLight results. I never cease to be amazed at the applications users have found for HydroLight. I'm sure that Preisendorfer would be speechless at the computational capabilities of the code on today's computers, but I don't think he would be at all surprised to learn that his original mathematics has not needed any improvement at all.

The most recent version 5.2 still has code inside that was written when I was a post-doc. It may have been good code when written, but it's ugly, difficult to understand, hard to maintain, violates modern standards for good programming, and I'm embarrassed to have anyone see it. I've been telling myself for 15 years that I'll rewrite everything from scratch as soon as I retire. That time has now come.

1.1 Computer History

The original version of HydroLight was written in Fortran 66 for a CDC 6600 computer, which was available to Presendorfer and me at PMEL. The room-sized computer itself was located at the NOAA Environmental Research Laboratory Headquarters in Boulder, Colorado. There was a dedicated phone line for communication between that location and PMEL in Seattle. Input was via card decks read in at PMEL, with output via a 600 line-per-minute printer at PMEL. With luck, I could submit a run at the end of the day and the printout would be awaiting me the next morning.

The CDC 6600 is now considered to have been the first "supercomputer" because of its architecture. It cost $7 million. It had 64 kilobytes of magnetic-core memory (that's right, kids, 64 Kbytes of RAM, created by wires threaded though small ferromagnetic rings). The equivalent of hard-drive memory was 7-track tape drives. It could crunch numbers at at awesome 0.5 million floating point operations per second (0.5 MFLOPS). To put this into perspective, current (2016) fast PC chips run in the 100 giga FLOPS range, roughly 200,000 times faster that the 6600. My desktop PC has 12 Gbytes of RAM, or 187,500 times more "core" memory, and I have a couple of tera bytes of hard drive. Current supercomputers have speeds over one peta FLOPS, or two billion times faster that the 6600, and they have 12 Tbytes or more of RAM, over 100 million times greater.

The 6600 was not free to us. It cost 11 cents per second of execution time, and Preisendorfer had to bring that money in via proposals. I well remember one run that cost almost $1,100 (half a month of my salary!). To minimize the financial loss if an error was made in the input or if a change to the code didn't work, the code was partitioned into five separate programs, which were executed sequentially. Then, for example, if an error was made when running the fourth program, it was necessary only to rerun that program using the intermediate output saved to tape from a successful run of the third program. A full, start-to-finish HydroLight run could therefore take as long as five days if the runs were made overnight.

Fortran 66 allowed for arrays to have a maximum rank of three, i.e., three dimensions indexed as array(i,j,k). However, HydroLight needed 4-dimensional arrays, in particular to describe scattering from an initial direction (θ', ϕ') to a final direction (θ, ϕ). To circumvent the limits on array dimensions, 4-D arrays were stored as 1-D arrays, with the index (storage location) of the 1-

D array being explicitly computed from the 4 indices, and vice versa. There was thus a lot of indexing calculation overhead done in some array manipulations. My Unix workstation at JPL had a Fortran 77 compiler. Fortran 77 allowed arrays of rank 7, so I was able to rewrite the code to eliminate the cumbersome 1-D array storage of inherently 4-D arrays.

Because of the small amount of core memory on the 6600, it was necessary to store large arrays on the 7-track tape drives. The arrays were then read piecemeal into core memory for processing. To multiply two large matrices, for example, the i^{th} row of the left matrix and the j^{th} column of the right matrix would be read into core as two 1-D arrays, the row times column product would be computed as the dot product of the 1-D arrays (via a DO-loop over the array elements), and the result would be stored in the (i, j) element of the product matrix. The now obsolete PAUSE command in Fortran allowed the program to suspend execution and issue a console command to the computer operator to mount the desired 7-track tape. After finding and mounting the tape, the operator hit a key to resume execution. The advent of virtual memory in the late 1980s made array size almost irrelevant and enabled important simplifications of the math code.

I am sometimes asked by people 40 years my junior why I wrote HydroLight in Fortran, and not in MATLAB, Python, Java, C++, whatever, which they have just learned in school. One answer is that MATLAB/Python/Java/C++/whatever had not yet been invented in 1978. Another answer is that I've have 50 years of scientific programming experience with Fortran. I wrote my first computer program using Fortran IID (Fortran II for a computer with a disk drive) in 1964 on an IBM 1620 computer (Google that one if you want to see a primitive computer). I was then a junior in high school and a professor at the local university taught an evening programming class open to high school students. But the real reason for sticking with Fortran is that Fortran always has been, is now, and probably always will be the best language for serious number crunching, which is what HydroLight does. Many of the features of other languages are irrelevant to the numerical calculations in HydroLight, and those languages often lack the features needed for scientific computations. (Google "Fortran vs C", for example, and you'll see what I mean.) HydroLight and I are not dinosaurs "stuck" in the Fortran world; we remain there because it's still the best language for scientific computing. That's why big numerical codes such as general circulation models are usually still in Fortran. I once had a programmer who wanted to use MATLAB for a programming task I had. I said OK, but I also programmed the same problem in Fortran for comparison. The answers were the same, but the Fortran code ran over 100 times faster than MATLAB. Enough said.

References

C. D. Mobley. A numerical model for the computation of radiance distributions in natural waters with wind-roughened surfaces. *Limnol. Oceanogr.*, 34:1473–1483, 1989.

C. D. Mobley. *Light and Water: Radiative Transfer in Natural Waters*. Academic Press, 1994. Available online at `www.oceanopticsbook.info/view/introduction/level_2/text_books_relevant_to_ocean_optics`.

C. D. Mobley and R. W. Preisendorfer. A numerical model for the computation of radiance distributions in natural waters with wind-roughened surfaces. Technical report, U.S. Department of Commerce, NOAA Pacific Marine Environmental Lab, 1988. Tech. Memo. ERL PMEL-63.

R. W. Preisendorfer. *Hydrologic Optics*. U.S. Department of Commerce, NOAA Pacific Marine Envoronmental Laboratory, 1976. In 6 volumes. Available online at `www.oceanopticsbook.info/view/introduction/level_2/text_books_relevant_to_ocean_optics`.

R. W. Preisendorfer. *Principal Component Analysis in Meteorology and Oceanography*. Elsevier Science Publishers, 1988. Posthumously compiled by C. D. Mobley.