In 1956, with a new degree in math from UC Berkeley, I started working in the preliminary design section of the North American Aviation engineering department in a group that calculated the points for chebyshev polynomials in order to fit curves going thru sparse data points. The curves, drawn by engineers, represented data for proposed jet engines to be used in an IBM 704 computer program for the optimum design of military aircraft. The mission for an aircraft would be simulated in the computer. For each flight, the geometry of the airplane would be altered until the best design was found. The simulated flights would also involve various jet engines, with data supplied by the engine companies. A curve could represent fuel consumption versus speed of the aircraft at a particular altitude. I designed a nomogram that easily gave the chebyshev points without the need of calculations. For this, I was made head of the smaller curve fitting group, and allowed to attend an IBM 704 assembly language programming course.

I noticed that the engineers, who drew these engine data curves, used whatever French curves they pulled from a desk drawer, to draw the smooth curves, rather than having any real knowledge of the shape of the curves. I then devised a method of interpolating points of a polynomial, that would avoid the need of drawing the curves and calculating the chebychev polynomials. It consisted of adding an extra x-coordinate and finding the y-coordinate so that the curve would have a minimum arc length. When there were only three data points, for which a parabola thru them would give a gross error, I would find a cubic equation, that gave a smooth curve thru the three data points plus the extra point for the given x-coordinate. I then wrote the interpolation program, for not only the curve that represented a single flight elevation, but was extended for a family of curves that represented a flight path thru different altitudes. As a result of using my interpolation program, the group of people involved in drawing the curves and fitting them to interpolating polynomials, were given other jobs.

Afterwards, I joined a small programming research group within the company computer department. The head of the group had been instrumental in applying IBM card processing equipment with Hollerith cards to solve engineering problems. This was before the advent of the IBM computers. While a member of this group, I designed and wrote an interpretive programming system, called TabForm, that enabled engineers to use the IBM 704 computer to work out calculations previously done on their desk calculators with the data and operations written in rows and columns on tabular forms. This was before the introduction of programming languages such as Fortran.

Shortly afterwards, our group was disbanned in order that we would become involved with the introduction of newer IBM computers and systems. Myself, with two collegues, decided instead to transfer to Autonetics, a division of North American Aviation, that designed and built military computers. We became members of a logical design group, that also wrote the programs for these computers. I worked on the software for the Verdan computer, which was a combined digital computer and a digital differential analyzer. It was to be used for navigating naval ships, including submarines. I wrote the programs needed for production testing in the factory and subsequent maintenance of the computer. I also taught a class of Navy tecnicians in the programming of the Verdan. I then got a leave of absence from Autonetics in order to attend a school in France.

After returning from France, I rejoined Autonetics for a short time before being hired by California Computer Products (CalComp) to write the program for the Fadac artillery computer. CalComp was formed by former engineers from Autonetics. The Fadac computer was designed and built by Autonetics. After finishing the Fadac program, I returned to France, where I had a job as a programmer at the CNRS Institut Blaise Pascal in Paris.