KERNFORSCHUNGSZENTRUM

KARLSRUHE

Februar 1974 KFK 1911

Institut für Angewandte Systemtechnik und Reaktorphysik
Projekt Schneller Brüter

Identification of the LMFBR Dynamic State for Detection of Coolant Boiling

D.M. Wiberg

GESELLSCHAFT
FÜR
KERNFORSCHUNG M.B.H.

KARLSRUHE

Als Manuskript vervielfältigt

Für diesen Bericht behalten wir uns alle Rechte vor

GESELLSCHAFT FÜR KERNFORSCHUNG M.B.H.
KARLSRUHE

KERNFORSCHUNGSZENTRUM KARLSRUHE

KFK 1911

Institut für Angewandte Systemtechnik und Reaktorphysik
Projekt Schneller Brüter

Identification of the LMFBR Dynamic State for Detection of Coolant Boiling

by

D.M. Wiberg+)

+) visiting summer scientist from the University of California, Los Angeles

Gesellschaft für Kernforschung mbH., Karlsruhe

Identification of the LMFBR Dynamic State for Detection of Coolant Boiling

Abstract

Modern identification theory is applied to equations describing an LMFBR to investigate the feasibility of estimating spatial sodium coolant and fuel pin temperatures. It is shown that measurement errors can be reduced and unmeasurable states estimated by a Kalman filter. These state estimates can then be used in conjunction with other indications of sodium boiling to form a scram criterion. Furthermore, the state estimates can be used by the reactor operators to assess reactor safety under many conditions. The need is shown for further study of off-line tuning of the Kalman filter to estimate spatial fuel burn up distribution and to estimate contaminant accumulation in a subassembly.

7. Januar 1974

Identifikation des dynamischen Zustands eines LMFBR zum Nachweis von Kühlmittelsieden

Zusammenfassung

Die moderne Identifizierungstheorie wird auf Gleichungen angewendet, die einen LMFBR beschreiben, um die Möglichkeit zu untersuchen, ortsabhängige Natriumkühlmittel- und Brennstabtemperaturen abzuschätzen. Es wird gezeigt, daß Meßfehler reduziert und nicht-meßbare Zustände mit einem Kalman-Filter abgeschätzt werden können. Diese Zustandsschätzungen können dann in Verbindung mit anderen Hinweisen auf Natriumsieden verwendet werden, um ein "Scram-Kriterium" zu bilden. Die Zustandsschätzungen können ferner von Reaktorbetreibern dazu verwendet werden, die Reaktorsicherheit unter verschiedenen Bedingungen abzuschätzen. Die Notwendigkeit für weitere Studien des "off-line" Abstimmens des Kalman-Filters wird gezeigt, um die ortsabhängige Brennstoffabbrandverteilung und die Häufung von Kontamination in einem Teilbereich des Cores (subassembly) abzuschätzen.

Table of Contents

Introduction

Neutron flux, coolant temperature, and coolant flow are measurable components of the dynamic state of a nuclear reactor. These and other measurable quantities are interpreted by both the reactor operator and the scram system as an indication of the total dynamic state of the reactor and therefore as an indication of the future behavior of the reactor. In most applications the physical cause of the measurements is merely implicitly assumed in influencing the actions of the reactor operators and the scram system, otherwise each measurement can be viewed as a quantity independent of the other measurements. However, when the physical process can be accurately described by a set of mathematical equations, these equations can be used not only to combine all measurements so that each measurement becomes more accurate, but also can be used to give an estimate for all the unmeasurable components in the complete state vector.

This idea of using the state equations to supplement measured data has been used in a number of previous applications. For exemple the period meter uses the measured value of the neutron flux and a mathematical equation, the inhour equation, to estimate a quantity that is not directly measurable, the reactor period. However, the period meter in its present development has not yet taken advantage of the modern theory of identification [1] . Working applications of the modern theory already exist in the aerospace [2 - 4] , chemical [5] , biomedical [6] , etc., industries and have had great success. Studies of applications of the modern theory in the nuclear industry have been made on the Halden reactor [7 - 9] and on rod drop experiments [10, 11] .

The purpose of this report is to propose a particular type of modern identification (Kalman filtering) to estimate the state of a liquid metal fast breeder reactor (LMFBR). Not only will this give a better picture of the reactor state to the operators, but good estimates of coolant and fuel temperature distribution in space can be combined with acoustic and reactivity noise measurements to form a scram signal to prevent coolant boiling. The particular problem of sodium boiling in an LMFBR is perhaps the most compelling reason for the use of a Kalman filter. Thus this report is mainly concerned with the application of a developed theoretical identification procedure, Kalman filtering, to help prevent a problem, coolant boiling, in an LMFBR.

The contents of the remainder of this report are arranged as follows. Firstly the mechanism and dangers associated with the propagation of coolant boiling in an LMFBR are reviewed. Then the state equations of a typical LMFBR are developed. Next a physically-oriented summary of Kalman filter theory is given. Following this is an investigation of the process and measurement noise parameters. Using this data a Kalman filter is applied to the state equations and some numerical results are given. The main body of the report ends with conclusions and suggestions for further work. Finally, an appendix describes some computational problems encountered in the application of the Kalman filter and another appendix describes how the Kalman filter could be implemented on a digital data acquisition system.

Propagation of Coolant Boiling

In a developmental program it is difficult to determine where dangerous situations occur. It is nuclear safety standard practice either to prove that a particular situation cannot lead to danger or to detect and control the situation. It has not yet been proven that coolant boiling causing subsequent fuel rod failures will not propagate in the manner described below, even though the probability of such a situation might be low. Therefore, even though propagation of coolant boiling may be proven innocuous in future LMFBR development, it is the most pressing reason at present to instrument the LMFBR with a Kalman filter.

To describe the propagation of coolant boiling, it is best to start by describing the LMFBR. To be specific assume a type of LMFBR exemplified by the SNR-300 [12 - 14] (see Fig. 1). Assume the core contains 151 subassemblies and that a typical subassembly contains 165 fuel pins that are 6 mm in diameter and .95 meters in length. The fuel pins are parallel to one another in the subassembly and are spaced slightly apart from one another so that the liquid sodium coolant can flow lengthwise along the pins and remove the heat generated by them (see Fig. 2). In normal operation assume the fuel pin center temperature is about 2300^{o}C and the edge temperature is around 700^{o}C. However, the cladding temperature is about 100^{o}C lower. Also take the normal sodium coolant inlet temperature to be 380^{o}C for all subassemblies and the exit temperature to range from 576^{o}C at the core center to 530^{o}C at the core edge. The coolant is pressurized to 2.5 atmospheres so that it boils at about 1000^{o}C.

Assuming these LMFBR parameters, coolant boiling can propagate as follows. Initially the sodium coolant is chemically pure. In reactor operation the sodium picks up contaminants from a number of sources including substances escaping from improperly canned and failed fuel pins, substances leeched from stainless steel tubing and joints, reaction products from the surfaces of structural elements, pieces of structure and instruments that work loose over a period of time, etc. Some of these contaminants could possibly lodge or stick in the small clearance between fuel pins or at spacer grids in a subassembly in such a way as to block more than 60 % of the coolant flow over one particular fuel pin. In that case [15] the heat produced by the pin is sufficient to raise the sodium coolant temperature to boiling. However, note that cladding damage can occur even before the coolant boils, so that it is the cladding temperature that should be kept below 750^{o}C to insure prevention of subassembly damage. Because the sodium vapor cannot carry much heat away from the fuel pin, in a short time the fuel pin melts. There results a fuel-sodium interaction with a number of possible consequences. The worst imaginable is that this interaction eventually expels a large quantity of sodium coolant from that central region of the core in which the reactivity coefficient of sodium is positive. This might immediately cause subassembly damage. A more probable consequence of the fuel-sodium interaction is that it would in effect merely add more contaminent to the sodium coolant. This would make other flow blockages more likely in the future. The action would then be similar to a slowly growing cancer, cutting off more and more flow over a time period that could be months in duration. The probability of a reactor transient would increase in such a situation and the most likely end result of a slowly growing flow blockage would be distortion of the subassemblies and their structural supports. This would necessitate a long and hence costly reactor shut-down.

Sodium boiling over one single fuel pin is extremely difficult to detect because there are 165 pins times 151 subassemblies equals 24,915 pins in the core. It is practically impossible to put at least one thermocouple on each pin. Instead, each subassembly is instrumented with four thermocouples, all reading the bulk sodium temperature at the exit. Logical comparison using a two-out-of-three rule, with the fourth thermocouple as a spare, then gives one subassembly bulk sodium exit temperature signal. Therefore 151 temperature signals emanate from the reactor. It is unlikely that fewer will suffice because each subassembly is somewhat thermally isolated from its neighbors by sodium coolant passing between the subassemblies.

The effect of sodium boiling over a few fuel pins in an otherwise unchanged subassembly is to increase the bulk sodium exit temperature only a few degrees. The exact number of degrees before clad damage should be experimentally verified. One conclusion of this report is that the Kalman filter can estimate this small temperature change, even with very poor thermocouples. However, the small temperature change could also caused by a blockage distributed over a number of coolant channels within the subassembly, and not just affecting a few pins. To prevent this situation from giving a false alarm, a small rise in temperature within a subassembly should be compared with other indications of sodium boiling, such as coolant, neutron, reactivity and/or acoustic noise spectra.

The coolant exit temperature spectrum changes with the imposition of a sudden blockage because the flow becomes more turbulent. This is detectable by a thermocouple with a fast time constant [16] . Also sodium boiling causes the flow to become more turbulent, so that the addition of high frequency components to the coolant noise spectrum is an indication of incipient sodium boiling. However, the fast thermocouples have not been completely tested.

The neutron flux and the reactivity noise spectrum also changes with sodium boiling [17] . The sodium void caused by the sodium vapor replacing liquid sodium changes the Doppler and, to a lesser extent, the absorption coefficients of reactivity. The net change is positive in the center of the core and negative at the edges, so that in these regions there exists a reactivity source or sink that drives the neutron flux and reactivity noise in the case of sodium boiling. However, there exists a region of zero net reactivity coefficient between the center and edge of the core, so that this method must also rely on supplemented indications of sodium boiling to detect boiling in this region.

The acoustic noise spectrum also changes under the sonic noise generated by the collapse of sodium vapor bubbles during boiling [18] . When the spectrum so generated is not masked by the vibration of the operating reactor, sodium boiling can be detected by this means also.

In summary, the reasoning is this. Sodium boiling has not yet been proven innocuous to structural integrity. Until this is done, methods of detection must be employed. No method of detection appears to be 100 % sure, so that reactor shut down must be dictated by an evaluation of the complete reactor state. Use of the reactor equations together with the measurable quantities emanating from the reactor to form a Kalman filter will help evaluate the complete reactor state.

LMFBR State Equations

To obtain an estimation scheme that can be put into practice, a mathematical model for the LMFBR must be found that is a good compromise between simplicity and accuracy. In this preliminary investigation a rather gross approximation is made to obtain a very simple model, which should be improved upon in further studies. All the neutronics is lumped into one equation for the reactor power, and then linearized about the mean value of the operating power which is assumed constant.

$$\Lambda \frac{dP}{dt} = P_o \sum_{i,j=1}^{N,M} \alpha_{ij} T_f^{ij} + S \qquad (1)$$

where $P(t)$ = reactor power deviation from P_o, in mega watts.

P_o = constant mean operating power, 723 MW

Λ = effective neutron lifetime, lumping fast and all delayed neutron lifetimes = 6,7 sec.

α_{ij} = temperature coefficient of reactivity $^{o}C^{-1}$

$S(t)$ = zero mean neutron noise source, MW

N = number of channels (groups of subassemblies)

M = number of axial zones

$T_f^{ij}(t)$ = fuel temperature deviation in the i<u>th</u> channel and j<u>th</u> axial zone, ^{o}C

The temperature coefficient of reactivity depends on the Doppler node fraction W_D^{ij}. Assuming the overall temperature coefficient of reactivity is -0.005, then

$$\alpha_{ij} = -.005\, W_D^{ij}$$

$$\text{where} \quad \sum_{i,j=1}^{N,M} W_D^{ij} = 1. \tag{2}$$

In the i,j<u>th</u> node the average fuel temperature deviation T_f^{ij} obeyes for i=1, N and j=1, M

$$C_{pf}\, N_P^{ij}\, \pi\, r_o^2\, H\rho_f \frac{dT_f^{ij}}{dt} = W_f^{ij}\, P - 2\,\pi\, r_o\, HN_P^{ij}\, h_T\, (T_f^{ij} - T_c^{ij}) \tag{3}$$

where N_P^{ij} = total number of fuel pins per channel = $m_{ij}\,(\pi\, r_o^2\, H\rho_f)^{-1}$

m^{ij} = fuel mass in the i, j<u>th</u> node in grams

r_o = radius of fuel pin = .6 cm (note a more realistic number is .3 cm)

H = height of axial zone, = 95/M cm

ρ_f = density of fuel pin = 10 g/cm^3

C_{pf} = specific heat of fuel = .3 watt-sec/g ^{o}C

W_f^{ij} = power fraction in node $\sum_{i,j=1}^{N,M} W_f^{ij} = 1$

h_T = heat transfer coefficient = 1.0 watt/cm^2 ^{o}C

$T_c^{ij}(t)$ = sodium coolant temperature deviation in the i,j<u>th</u> node, ^{o}C.

Also in the i,j$\underline{\text{th}}$ node the coolant temperature deviation T_c^{ij} obeys for i=1,N and j=1,M, assuming complete thermal isolation of each subassembly,

$$fN_p \pi r_o^2 \, H\rho_f \, C_{p_c} \frac{dT_c^{ij}}{dt} = 2\pi r_o^2 \, HN_p h_T (T_f^{ij} - T_c^{ij}) - fN_p \pi r_o^2 \rho_f C_{p_c} V(T_c^{ij} - T_c^{ij+1}) \qquad (4)$$

where	f	= coolant / fuel ratio = .127
	V	= coolant flow = 500 cm/sec.
	$T_C^{iM+1}(t)$	= inlet sodium temperature, ^{o}C

Using the numerical values given for an LMFBR typified by the SNR-300 then gives the state equations

$$\frac{dP}{dt} = -.54 \sum_{i,j=1}^{N,M} W_D^{ij} \, T_f^{ij} + .15 \, S \qquad (5)$$

$$\frac{dT_f^{ij}}{dt} = 3.3(W_f/m)^{ij} \, P - 2(T_f^{ij} - T_c^{ij})$$

$$\frac{dT_c^{ij}}{dt} = 4(T_f^{ij} - T_c^{ij}) - 0.5 \, M(T_c^{ij} - T_c^{ij+1})$$

For N = 3 and M = 3 the reactor nodes are as pictured in Fig. 3 and the values of W_D^{ij}, W_f^{ij}, and m^{ij} are given in Table 1.

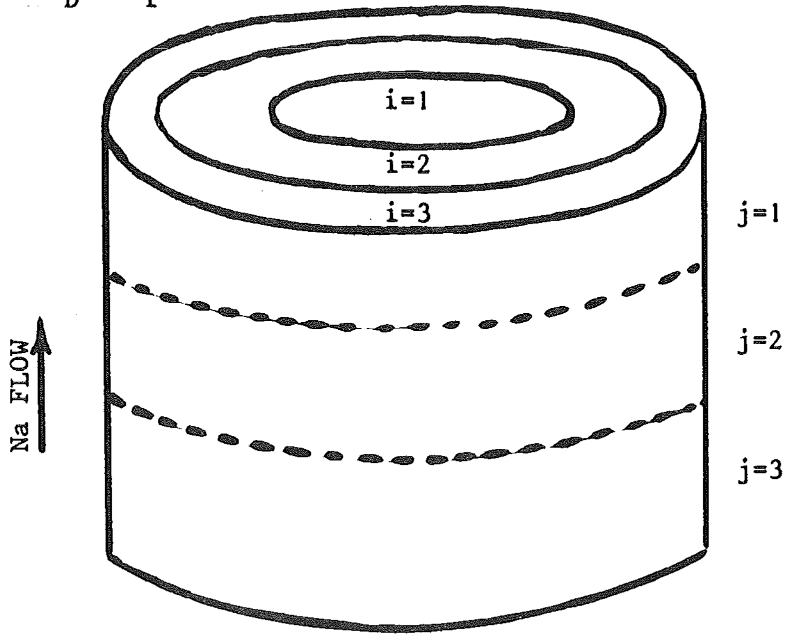


Fig. 3 LMFBR with 9 nodes.

Table 1. Coefficients of a 9 node LMFBR model.

		Doppler coefficient	Power fraction	fuel mass x 10^6g
i	j	W_D^{ij}	W_f^{ij}	m_{ij}
1	1	0.065	.032	.417
1	2	.167	.105	.380
1	3	.031	.025	.417
2	1	.135	.063	.925
2	2	.301	.195	.840
2	3	.040	.043	.925
3	1	.105	.119	9.145
3	2	.145	.346	8.292
3	3	.011	.072	9.145

The given numerical data is sufficient to calculate the dynamic behavior of a reactor model with N and M = 3 or less. Data is available [19] for N = 10 and M = 11. Because the 151 subassemblies are arranged in concentric annuli, the cyclindrical symmetry necessitates the computation of at most N = 10 channels. This simplification makes the problem computationally feasible.

However, rather than dealing with a large dimensional state vector, for simplicity of further exposition choose N and M unity. This lumps the reactor into one node and sacrifices accuracy for clarity. Then the equations (5) can be put into the vector matrix (state space) form

$$\frac{d}{dt}\begin{pmatrix} P \\ T_f \\ T_c \end{pmatrix} = \begin{pmatrix} 0 & -.54 & 0 \\ .11 & -2 & 2 \\ 0 & 4 & -4.5 \end{pmatrix} \begin{pmatrix} P \\ T_f \\ T_c \end{pmatrix} + \begin{pmatrix} .15 & 0 \\ 0 & 0 \\ 0 & .5 \end{pmatrix} \begin{pmatrix} S \\ T_o \end{pmatrix} \qquad (6)$$

where $T_o = T_c^{12}$ = inlet sodium temperature. This is the assumed state space equation for the operating LMFBR dynamics.

LMFBR Noise Properties

The only external measurements considered here are the 161 temperature signals indicating the bulk exit temperature of each subassembly and also the signals from the neutron flux meters. Because spatial flux effects are negligible in the SNR-300, these flux meter signals can be logically combined to give one signal proportional to total power. In this section the noise properties of these temperature and power signals are investigated.

The bulk exit sodium temperature of a subassembly of the KNK reactor was measured at zero reactor power [16] . At zero power most of the measurement error is due to the turbulence in the sodium flow, because the thermocouple measures local rather than bulk exit temperature. From the graph of the temperature measurement at zero power (Fig. 4 a) there appears a sine wave of 0.01^{o}C amplitude at 0.1 Hz plus a hash of 0.005^{o}C amplitude at frequencies greater than 5 Hz. For simplicity of modelling this was taken to be a white noise of autocorrelation .003 $\delta(t-\tau)$ $^{o}C^{2}$sec (Fig. 4 b). This represents about the best that the thermocouples can measure. As the reactor comes to full power, it is probable that more turbulence, cable pick-up, vibration, etc., will corrupt the measurements. Taking the industrial thermocouple standard of ± 1.5 ^{o}C as the worst case, then the temperature measurement white noise autocorrelation is at most 1.0 $\delta(t-\tau)$ $^{o}C^{2}$sec.

The neutronic noise properties can be found from standard derivations [20] to give

$$< P_m^2 > \;=\; \frac{E_f P_o}{W} + E_f P_o D \;|H(j\omega)|^2 \qquad (7)$$

where $< P_m^2 >$ = auto power spectral density, $(MW)^2$ sec
E_f = energy per fission = 3.2×10^{-17} MW sec
W = detector efficiency = 10^{-9}
D = Diven factor $\frac{\overline{\nu(\nu-1)}}{\nu^2}$ = .8
$H(j\omega)$ = reactivity transfer function

Let $H_s(j\omega)$ be the transfer function from S to P in equation (6). Then

$$\frac{H(j\omega)}{H_{s(j\omega)}} = \frac{\Lambda}{\ell} \qquad (8)$$

where ℓ = prompt neutron lifetime = 4.6×10^{-7} sec.

As an interesting aside, note that because the noise amplitude depends on the prompt neutron lifetime, fast reactors are much noisier than thermal reactors. Using numerical values in (7) and (8) gives

$$< P_m^2 > \;=\; 2.3 \times 10^{-5} + 3.9\, |H_s(j\omega)|^2 \qquad (9)$$

Therefore the white noise S has autocorrelation $3.9\ \delta(t-\tau)\ (\mathrm{MW})^2\ \mathrm{sec.}^{-1}$ and the neutron flux measurement noise has autocorrelation 2.3×10^{-5} $\delta(t-\tau)\ (\mathrm{MW})^2\mathrm{sec.}$

Substituting the measured value T_{om} of the inlet sodium temperature T_o into the state equations (6) gives the stochastic description of the LMFBR as

$$\frac{d}{dt}\begin{pmatrix} P \\ T_f \\ T_c \end{pmatrix} = \begin{pmatrix} 0 & -.54 & 0 \\ .11 & -2 & 2 \\ 0 & -4 & -4.5 \end{pmatrix}\begin{pmatrix} P \\ T_f \\ T_c \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ .5 \end{pmatrix} T_{om} + \begin{pmatrix} .15 & 0 \\ 0 & 0 \\ 0 & .5 \end{pmatrix}\begin{pmatrix} S \\ m_{T_o} \end{pmatrix}$$

$$\begin{pmatrix} P_m \\ T_{cm} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} P \\ T_c \end{pmatrix} + \begin{pmatrix} m_P \\ m_{T_c} \end{pmatrix} \qquad (10)$$

where S, m_{T_o}, m_p and m_{T_c} are independent zero mean white noises with

$$< S^2 > \;=\; 3.9 \quad (\mathrm{MW})^2\ \mathrm{sec}^{-1}.$$

$$< m_{T_o}^2 > \;=\; .003 \text{ to } 1.0 \quad (^{\circ}\mathrm{C})^2\ \mathrm{sec.}$$

$$< m_p^2 > \;=\; 2.3 \times 10^{-5} \quad (\mathrm{MW})^2\ \mathrm{sec.}$$

$$< m_{T_c}^2 > \;=\; .003 \text{ to } 1.0 \quad (^{\circ}\mathrm{C})^2\ \mathrm{sec.}$$

Summary of Kalman Filtering

A Kalman filter estimates the state vector of a linear dynamic process in an optimal manner [21] . Assume that the physical system obeys the formal dynamical equations

$$\begin{aligned} \frac{d\vec{x}}{dt} &= A\,\vec{x} + B\,\vec{u} + G\,\vec{w} \\ \vec{y} &= C\,\vec{x} + \vec{v} \end{aligned} \qquad (12)$$

where $\vec{x}$ is the n-dimensional state vector, $\vec{u}$ is a p-dimensional known input, $\vec{w}$ is an ℓ-dimensional zero mean white noise, $\vec{v}$ is an m-dimensional zero mean white noise, $\vec{y}$ is an m-dimensional measurement vector and A, B, C and G are compatible matrices. The stochastic LMFBR equation (10) is in the form (12). Furthermore the white noises have spectral density

$$<\vec{w}\vec{w}^{\dagger}> \;=\; Q \qquad\qquad <\vec{v}\vec{v}^{\dagger}> \;=\; R \qquad (13)$$

where Q is an $\ell x \ell$ symmetric nonnegative definite matrix, R is an m x m symmetric positive definite matrix, and the superscript † denotes transpose. This corresponds to equation (11) for the LMFBR.

The Kalman filter computes $\hat{\vec{x}}$ (t), the conditional mean of $\vec{x}$ (t) given the measurement time history $\vec{y}$ (τ) for $t_o \leq \tau \leq t$. The vector $\hat{x}$ (t) is the optimal estimate for $\vec{x}$ (t) in the sense that it minimizes any convex function of the error $\tilde{x}$ (t) = $\vec{x}$ (t) - $\hat{x}$ (t). To compute $\hat{x}$ (t), an analog or digital computer finds the solution to

$$\frac{d\hat{x}}{dt} = A\,\hat{x} + B\,\vec{u} + K\,(\vec{y} - C\,\hat{x}) \qquad (14)$$

where K is a precomputed gain matrix.

$$K = PC^{\dagger}R^{-1} \qquad (15)$$

where P is the variance of $\tilde{x}$ that obeys the matrix Riccati equation

$$\frac{dP}{dt} = AP + PA^{\dagger} + GQG^{\dagger} - PC^{\dagger} R^{-1} CP \qquad (16)$$

In the case of the LMFBR only the steady state solution P_{ss} to this equation is sought, i.e.

$$0 = AP_{ss} + P_{ss}A^{\dagger} + GQG^{\dagger} - P_{ss}C^{\dagger} R^{-1} CP_{ss} \qquad (17)$$

Only the steady state solution is needed because the state equation (10) is time-invariant over the time intervals considered, because the initial conditions are of no consequence, and mainly because only steady state solutions have the desired numerical accuracy when dealing with so many state variables as necessitated by the general model of equation (5).

Because P_{ss} is the error variance, it is a measure of how good an estimate of the state is obtained by the Kalman filter. Furthermore the type, number, and position of sensors can be optimized by computing a corresponding P_{ss} for each type, number, and position and then examining the effects. Thus P_{ss} becomes an indication of the feasibility of using a Kalman filter for LMFBR coolant boiling detection.

Numerical Procedures and Results

The solution of the steady state Riccati equation (17) was found for the three-dimensional model (10) and (11). The results are given in Table 2. The numerical values contained therein were found using the subroutine RIC SS6 obtaines from Macdonnell-Douglas Corporation Western Division, Huntington Beach, Calif. A listing of the routine is given in Appendix I.

The RIC SS6 program was tested with both a 2 and an 8 dimensional test subroutine and was found to be accurate to 0.5%. Unfortunately this accuracy is problem dependent, and the routine did not work for LMFBR models of more than 3 state variables. Because an accurate model must have more than 30 state variables, the numerical procedures need more refinement. A more detailed description of the problems and attempted solutions is given in Appendix I.

Table 2. Results

3 State Variable Model (P in MW, T_f in ^oC, T_c in ^oC)

Good Thermocouples and Flux Meter

TC read ± .05 C *   Flux Meter Efficiency $W = 10^{-9}$

Standard deviations:

P = 0.037 MW

T_f = 0.013 ^oC

T_c = 0.019 ^oC

$$P_{ss} = 10^{-4} \times \begin{pmatrix} 14.1 & .05 & -.04 \\ .05 & 1.7 & 1.7 \\ -.04 & 1.7 & 3.6 \end{pmatrix}$$

Bad Thermocouples and Flux Meter

TC read ± 1 ^oC * $W = 10^{-9}$

Standard deviations:

P = 0.037 MW

T_f = 0.16 ^oC

T_c = 0.32 ^oC

$$P_{ss} = 10^{-4} \times \begin{pmatrix} 14.4 & -7.3 & -9.0 \\ -7.3 & 261 & 335 \\ -9.0 & 335 & 1017 \end{pmatrix}$$

Bad Thermocouples and No Flux Meter

TC read ± 1^oC * $W = 0$

Standard deviations:

P = 0.76 MW

T_f = 0.27 ^oC

T_c = 0.35 ^oC

$$P_{ss} = 10^{-4} \times \begin{pmatrix} 5287 & 363 & -72 \\ 363 & 732 & 656 \\ -72 & 656 & 1253 \end{pmatrix}$$

<u>Good Thermocouples and No Flux Meter</u>

TC read ± .05°C [*] $\qquad W = 0$

Standard deviations

$P = 0.26$ MW

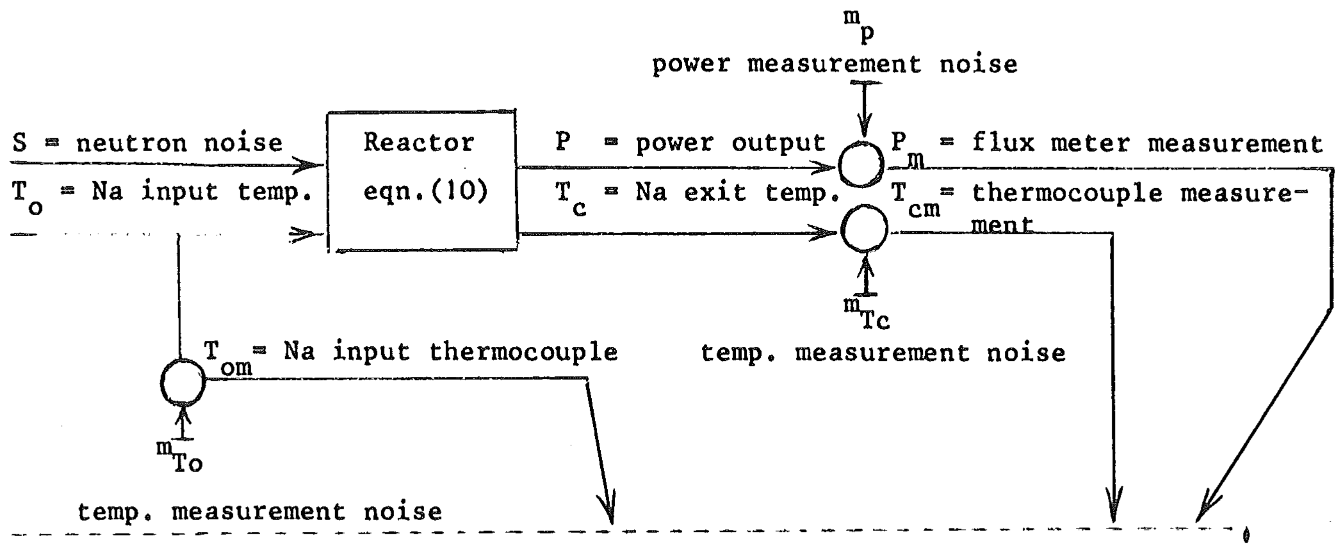
$T_f = 0.063$ °C

$T_c = 0.044$ °C

$$P_{ss} = 10^{-4} \times \begin{pmatrix} 676 & 134 & 68 \\ 134 & 40 & 25 \\ 68 & 25 & 19 \end{pmatrix}$$

[*] Really $< m_T^2 > = .003$ and 1.0 respectively.

The Kalman filter corresponding to the three dimensional model (10) with good thermocouples and a flux meter is pictured in Fig. 5.



$$\frac{d}{dt}\begin{pmatrix} \hat{P} \\ \hat{T}_f \\ \hat{T}_c \end{pmatrix} = \begin{pmatrix} 0 & -.54 & 0 \\ .11 & -2 & 2 \\ 0 & 4 & -4.5 \end{pmatrix} \begin{pmatrix} \hat{P} \\ \hat{T}_f \\ \hat{T}_c \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} T_{om} + \begin{pmatrix} 61.3 & .0013 \\ .22 & .056 \\ -.17 & .119 \end{pmatrix} \begin{pmatrix} P_m - \hat{P} \\ T_{cm} - \hat{T}_c \end{pmatrix}$$

COMPUTER

Fig. 5 Mechanization of the Kalman filter.

The equations pictured within the dotted lines are the Kalman filter, whose solutions $\hat{P}$, $\hat{T}_f$, and $\hat{T}_c$ are available to the reactor operators and the scram system in general.

Table 2 shows that the Kalman filter can estimate the unmeasurable components of the state vector with approximately the same variance as the measured components, and significantly decreases the variance of the measured components beyond the accuracy of the measuring instruments. This can be done for all ranges of instrument accuracy, even for commercial as opposed to laboratory instruments. Therefore it is reasonable to proceed with further experiments to ascertain whether incipient sodium boiling can be indicated.

Because no results are available yet for higher dimensional LMFBR models, physical reasoning must be used to predict what will happen as model dimension increases. Adding more channels, i.e. $N=151$, makes the per channel contribution of the flux meter negligible. Therefore results for the temperature accuracies should tend to those of the bottom half of Table 2. Adding more axial zones, $M=11$, increases the sensitivity to changes in parameters of the state equation, and decreases the variance from the physical model. This relationship must be explored via experiment with reasonable physical models. Of course, more results are expected soon, when the numerical difficulties of higher dimension have been cleared up. With increased dimension, the spatial resolution will be much finer. Then the reactor operators can tell where boiling occurred in the event of a scram. Thus the difficulty can quickly be located and the system brought back on to line faster.

The mechanization in Fig. 5 shows how the reactor state can be displayed to the reactor operators. Given some unforseen occurrence, this picture can help the operator assess the danger and the measures that must be taken to combat it. Thus a Kalman filter is useful for other occurrences than sodium coolant boiling. From an overall point of view, this might be the most compelling reason for the installation of a Kalman filter on an operating LMFBR.

The numerical values shown within the dotted lines of Fig. 5 are the parametric values of the reactor equations. Due to fuel burn up, changes in the heat transfer coefficient with scale build-up, etc., these parameters are not constant as assumed. They vary slowly, over a period of days. Perhaps once a day the Kalman filter needs to be "tuned" to obtain new parameter values. The accuracies indicated by Table 2 are so good that it seems probable methods such as those used by Olsson, et al., on the Halden reactor [7] will give new parameter values to a high degree of accuracy. This can be done off-line on a large digital computer to obtain the most accuracy possible. Having estimates for these parameters on a daily basis will then enable reactor operators to evaluate fuel burn-up, crud build-up, etc. Thus further efficiencies can be made by adjusting the loading schedules, etc. according to fuel burn up. Slow acting dangers such as crud build-up can also be assessed without inspecting reactor core components.

Thus Table 2 and Fig. 5 indicate the following reasons a Kalman filter should be installed on an LMFBR.

1. a Kalman filter indicates incipient sodium boiling
2. a Kalman filter tells where sodium boiling has occured in the event of a scram
3. "tuning" of a Kalman filter indicates crud build-up and fuel burn-up distribution
4. a Kalman filter gives the operators a picture of overall reactor dynamic operation.

It must be remembered that the results of Table 2 were obtained for a very simple mathematical model. Thus P_{ss} represents the error variance only between the mathematical model and the Kalman filter. There is an additional error between the physical system and the mathematical model. Therefore the results of Table 2 must be smaller than for a Kalman filter applied to the physical LMFBR system. It is mainly this fact that necessitates further experiments using hardware as close as possible to that encountered in practice. It is by no means claimed that one should use a Kalman filter on an LMFBR. The claim is that further investigation appears justified.

Future Work

Given the encouraging nature of these preliminary results, a four pronged attack seems justified on the development of a Kalman filter for application to LMFBRs. The first prong is the use of preliminary and subsequently developed Kalman filters directly on existing LMFBRs, such as EBR-2 and KNK. The second prong is the development of a Kalman filter for a single subassembly to be tested on a sodium loop such as at UCLA, Santa Suzanna, and Karlsruhe. The third prong is computer simulation, and the fourth is further theoretical development. To speed development, all four prongs should be pressed forward simultaneously.

Operating experience on an LMFBR is of prime importance, and should be obtained as early as possible. It is in actual application where the main difficulties can be found and worked upon. Because hardware takes a long time to set up, this prong of the attack has the longest lead time and should be worked on first to speed development.

Sodium loop testing is also of importance. The stochastic modellling of thermocouple accuracy and of the coolant turbulence should be based on further experiments done in a sodium loop. An investigation of the thermal coupling between subassemblies can best be done experimentally on a sodium loop. But perhaps most important is the construction of a Kalman filter to estimate the temperature in one subassembly immersed in a sodium loop. A number of thermocouples distributed throughout the test subassembly can check the accuracy of the states estimated by the Kalman filter that uses only the exit temperature measurement.

Computer simulation of the Kalman filter can give insight into the effect of changes in parameters. A Kalman filter derived using the linear deviation equation (5) should be used to estimate the state of a nonlinear model. Perhaps some nonlinearities need to be incorporated into the Kalman filter itself, and computer simulation should determine the answer. In fact, a boiling occurrence (where the state equations are different) can be simulated and the response of the Kalman filter observed. Other effects than boiling can be simulated, such as reactivity insertion, to see if the Kalman filter will indicate danger. Also the quantization can be determined that is necessary to adapt a Kalman filter to a data acquisition and safety control system such as MISS (see Appendix II). Finally, the sensitivity of the Kalman filter must be determinded by computer simulation, i.e. the effect of mismodelling the physical reactor by the assumed mathematical equations.

A very large area for study, mainly on a computer, is to determine a good method for "tuning" the paramters of the Kalman filter. Tuning, perhaps daily, will minimize sensitivity effects and provide burn up and crud accumulation data. There exists much literature [22-25] in the tuning area of identification, and a number of methods should be compared for application to an LMFBR.

One other reason for the development of a Kalman filter for an LMFBR is the stimulus it will give to theoretical matters. In Appendix I the need is demonstrated for a better numerical method to determine the eigenvalues of a nonsymmetric matrix. The application of modern identification techniques (such as [26]) in the tuning problem will lead to their refinement. But most welcome will be further progress in the stochastic modelling of a nuclear reactor. The assumption of a Wiener process to drive the formal stochastic equation (1) is dissatisfying. The application of a Kalman filter to a branching process [27] must be put on firm theoretical ground. Indeed, there appears to be no analytical stochastic model of a nuclear reactor that reduces to a Markov process with space dependence, which is needed for theoretical development of the identification theory. This has implications for the better theoretical understanding of stochastic processes in distributed systems, i.e. the basic processes of nature.

Appendix I

Computational Problems

The solution P_{ss} to the matrix Riccati equation (16) was sought using the subroutine RICSS6. It can be shown [28] that $P_{ss} = GF^{-1}$, where

$$\begin{pmatrix} A & -C^{\dagger}R^{-1}C \\ -GQG^{\dagger} & -A^{\dagger} \end{pmatrix} \begin{pmatrix} F \\ G \end{pmatrix} = \Lambda \begin{pmatrix} F \\ G \end{pmatrix}$$

where Λ is the n x n matrix of stable eigenvalues. The solution thus depends on the accuracy to which the eigenvalues can be found. For the 2x2 and 8x8 test cases, the eigenvalues were found to five significant digits. Unfortunately, for the 7x7 LMFBR model with N=1 and M=3, the eigenvalues could be found to only one significant digit. Changing from CHARD subroutine to an eigenvalue routine from the IBM scientific subroutine package called ATEIG, in conjunction with HSBG, yielded an increase in accuracy of only a factor of two. The subsequent calculations were therefore inaccurate and not kept.

It is possible the trouble stems from the almost singularity of the 2x2 diagonal blocks of A. An attempt to gain computational accuracy will be made by reducing the matrix to tridiagonal form. Then in double precision the following algorithm to compute the principle minors can be used:

$$\begin{aligned}
P\ (0) &= 1 \\
P\ (1) &= a_{11} - \lambda \\
P\ (2) &= \lambda^2 - (a_{11}+a_{12})\ \lambda + a_{22}\ (a_{11}-a_{12}) - a_{12}\ (a_{21}-a_{22}) \\
P\ (i) &= \left[\lambda^2 - (a_{ii}+a_{i-1,i-1})\lambda + a_{ii}\ (a_{i-1,i-1}-a_{i-1,i}) - a_{i-1,i}(a_{i,i-1}-a_{ii})\right] P(i-2) - \\
&\quad a_{i-2,i-1}\ a_{i-1,i-2}\ (a_{ii}-\lambda)\ P(i-3)\ \ i=3,4,..
\end{aligned}$$

From P (n) is obtained the characteristic polynomial, which is factored to obtain the eigenvalues.

Appendix II

Computer Implementation

The Kalman filter can be implemented on either an analog or a digital computer. To achieve the most accuracy an analog computer could be used. Then the partial differential equations inherent in reactor models can be most accurately simulated. However, the cost for this separate system would be in excess of \$ 100,000. This is not much when compared with possible savings of shut down time and ruined equipment. It is much when compared with the possibility of implementing the Kalman filter on existing digital computers at practically no cost.

The SNR-300 has two computers, called the safety computer and the process computer. The safety computer is a simple, redundant computer dealing directly with signals from the reactor to form scram criteria. It should not be touched. The process computer is a time sharing computer taking accurate signals from the reactor to perform such tasks as burn-up calculations, etc. It appears that the process computer is quite suitable for implementation of the Kalman filter. It could send signals to the safety computer if desired to form scram criteria with other detection methods. There appears to be only a small loss in accuracy from the analog computer because a fine spatial mesh can be used.

Thus it is proposed to do the experimental investigations leading to the development of the Kalman filter using analog computers. This retains the accuracy. However, final development will be on the digital process computer that would be part of the reactor system even if the Kalman filter were not present.

References

1. D.M. Wiberg, "Period meter design via identification techniques", to appear

2. Hughes Aircraft report, Kalman filtering of missiles.

3. G. Lind, UCLA M.S. comprehensive exam, June 1973.

4. D.R. Vaughan and T. Blackburn, Saturn autopilot design, AIAA Journal.

5. J. Seinfeld, Work on smog control, California Institute of Technology

6. G. Swanson, Ph.D.-Thesis, Stanford, 1972.

7. G. Olsson, "Maximum Likelihood Identification of Some Loops of the Halden Boiling Water Reactor", Halden Project Report, 1973.

8. G. Olsson, "Modelling and Identification of Nuclear Power Reactor Dynamics from Multivariable Experiments", 2nd Proc. IFAC Congress on Identification, The Hague, June, 1973.

9. I. Gustavsson, "Maximum Likelihood Identification of the Agesta Reactor and Comparison with Results of Spectral Analysis", Div. of Auto. Control, Lund Report 6903, 1969.

10. Venerus, Bullock "Estimation of the Dynamic Reactivity Using Digital Kalman Filtering", NSE 40, 199-205 (1970).

11. L.J. Habegger, and R.E. Bailey, "Minimum Variance Estimation of Parameters and States in Nuclear Power Systems", Proc. 4th IFAC Congress, Warsaw (1969), paper 12.2, also ANS Transactions Vol.II,1, Toronto, June 10-13, 1968, p. 237.

12. W. Häfele, G. Kessler, "SNR: The German-Benelux Fast Breeder", Nuclear News, March 1972.

13. F.R. Farmer, An appreciation of fast reactor safety, 1970.

14. K. Gast: Die Ausbreitung örtlicher Störungen im Kern schneller natriumgekühlter Reaktoren und ihre Bedeutung für die Reaktorsicherheit, KFK-1380, Mai 1971.

15. D. Kirsch: Untersuchung zur Strömungs- und Temperaturverteilung im Bereich lokaler Kühlkanal in Stabbündel-Brennelementen, KFK-1794, Feb. 1973.

16. L.Krebs, G. Weinkötz, "Measurements of Temperature Fluctuations" English Review Meeting, Karlsruhe, 14 June, 1973.

17. M. Edelmann, J. Ehrhardt, H. Massier, K. Vogel, "Experiments for Development of Methods and Systems to Detect Sodium Boiling in an LMFBR". IAEA Symposium on Nuclear Power Plant Control and Instrumentation, Prague, 22-26 January, 1973.

18. R.D. Smith, "Protective Instrumentation for Fast Reactors", Conf. on the Engineering of Fast Reactors for Safe and Reliable Operation, Oct. 1972, Karlsruhe.

19. G. Heusener, G. Kessler, F. Dunn, J. Jachson, G. Fischer, et.al. "Analysis of Hypothetical Accidents for SNR-300", KFK-report to be published.

20. H. Borgwaldt, "Neutron Noise in a Reactor with an External Control Loop", Nukleonik, 11. Bd., Heft 2, 1968 pp. 76-84.

21. K.J. Åström, "Introduction to Stochastic Control Theory", Academic Press, New York, 1970.

22. K.J. Åström and P. Eykhoff, "System Identification, a Survey", Automatica, Vol. 7, pp. 123-162, Pergamon Press 1971.

23. A.V. Balakrishnan and V. Peterka, "Identification in automatic control systems", Proc. 4th IFAC Congress, Warsaw, survey paper, also Automatica, 5, 817-829, 1969.

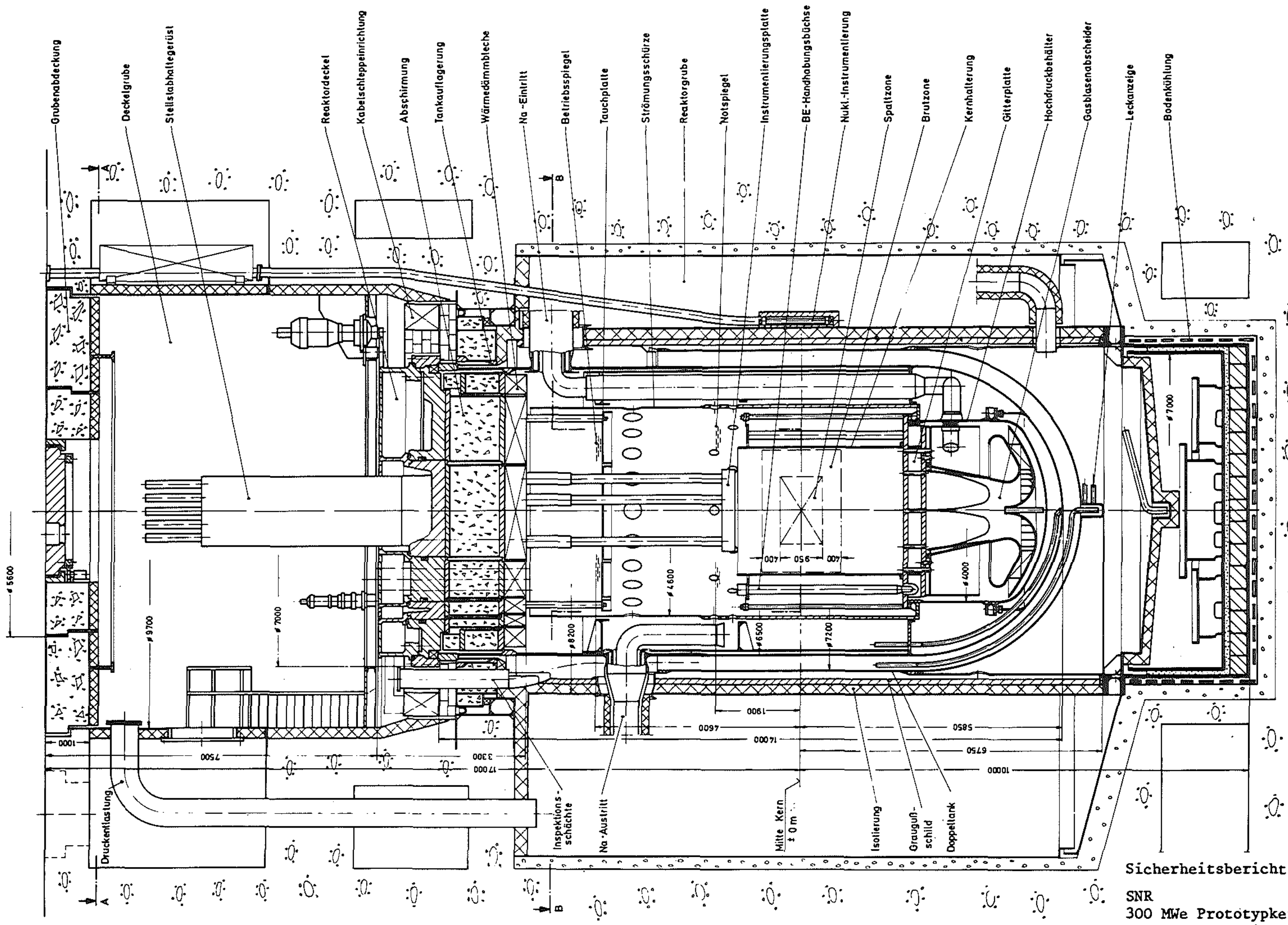
24. P. Eykhoff, "Process Parameter and State Estimation", Automatica, Vol. 4, pp. 205 - 233, 1968.

25. M. Cuenod and A.P. Sage, "Comparison of Some Methods Used for Process Identification", Automatica, Vol. 4, pp. 235-269, 1968.

26. A.V. Balakrishnan Springer Verlag red series, approx 86 (his second book in this series).

27. T.E. Harris, The Theory of Branching Processes, Springer, Berlin, 1963.

28. D.M. Wiberg, State Space and Linear Systems, Schaum's Outline Div. McGraw-Hill, 1971.



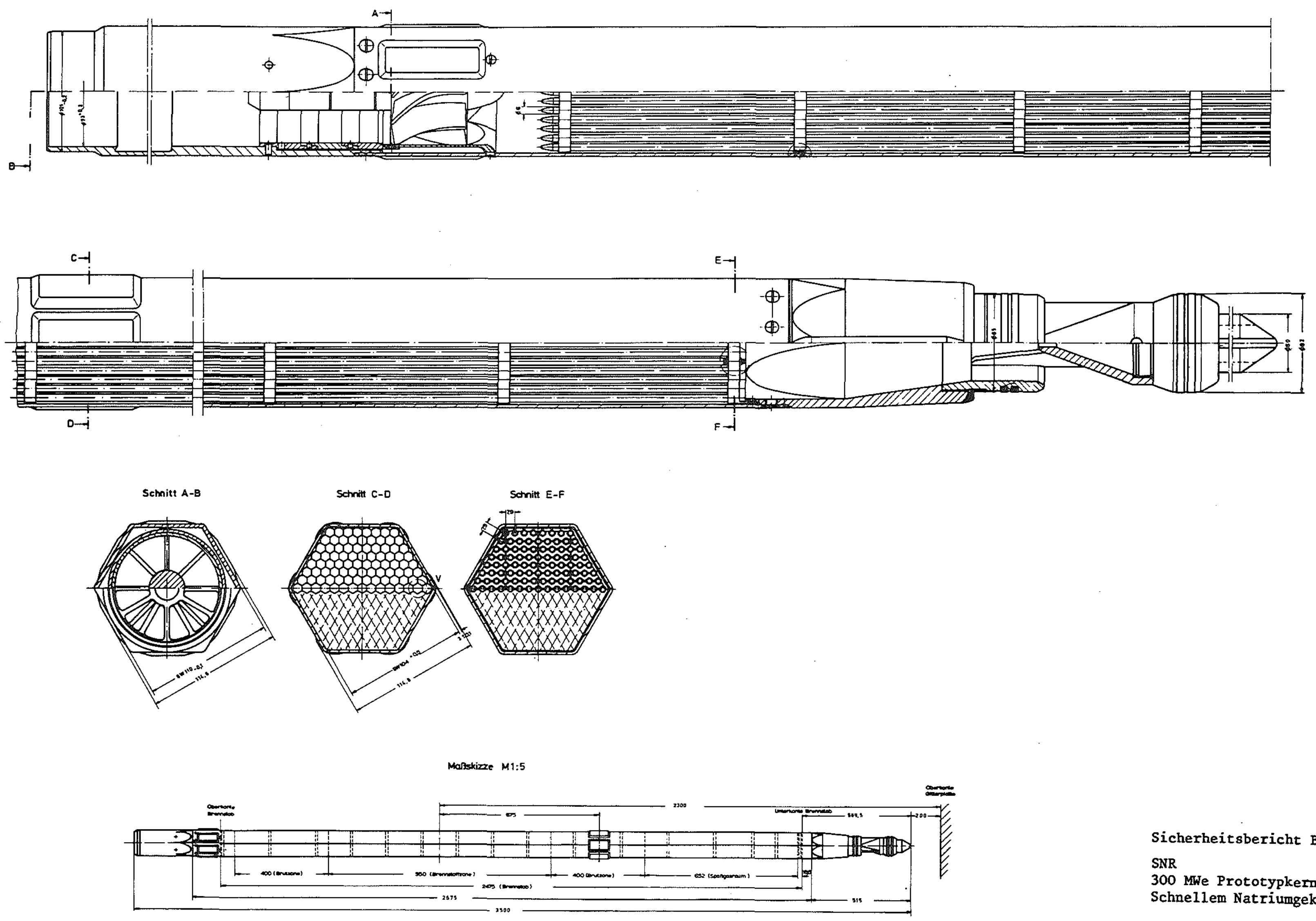
Sicherheitsbericht Band 3, INTAT-84a

SNR
300 MWe Prototypkernkraftwerk mit
Schnellem Natriumgekühltem Reaktor

300 MWe PROTOTYPKERNKRAFTWERK SNR

REAKTOR LÄNGSSCHNITT

Fig. 1



Sicherheitsbericht Band 3, INTAT-84a

SNR
300 MWe Prototypkernkraftwerk mit
Schnellem Natriumgekühltem Reaktor

300MWe PROTOTYPKERNKRAFTWERK SNR

BRENNELEMENT I

Fig. 2

```
      DIMENSION A(3,3),G(3,3),Q(3,3),P(3,3)
      COMMON/WORK/DUMMY(15450)
      ND = 3
      DO 348 I=1,ND
      DO 348 J=1,ND
      G(I,J)=0
      Q(I,J)=0
  348 A(I,J) = 0
      A(1,2) = 0.5
      A(2,1) = -2
      A(2,2) = -2.
      A(2,3) =4.
      A(3,2) = 2.
      A(3,3) = -6.
C
C     GOOD THERMOC AND FLUX METER
C
      Q(1,1) = 0.1
      Q(3,3) = 0.003
      G(1,1) = 50000
      G(3,3) = 300
      CALL RICSS6(A,G,Q,P,ND,ND)
      CALL PRMAT(P,ND,ND,ND,ND)
      TRP=0.0
      DO 345 L=1,ND
  345 TRP=TRP+P(L,L)
  344 FORMAT(E16.8)
      WRITE(6,344) TRP
C
C     BAD THERMOC AND FLUX METER
C
      Q(3,3) = 1
      G(3,3) = 1
      CALL RICSS6(A,G,Q,P,ND,ND)
      CALL PRMAT(P,ND,ND,ND,ND)
C
C     BAD THERMOC AND NO FLUX METER
C
      G(1,1) = 0
      CALL RICSS6(A,G,Q,P,ND,ND)
      CALL PRMAT(P,ND,ND,ND,ND)
C
C     GOOD THERMOC AND NO FLUX METER
C
      Q(3,3) = 0.003
      G(3,3) = 300
      CALL RICSS6(A,G,Q,P,ND,ND)
      CALL PRMAT(P,ND,ND,ND,ND)
      STOP
      END
      SUBROUTINE RICSS6 (A,G,Q,P,NS,NAR)
      DIMENSION  RTM(25,3), ITM(25,3), SCL(25)
      DIMENSION  SSS(5000), SR(50,50), SI(50,50),
     1                 SRS(2500),SIS(2500),AR(25,25),AI(25,25),
     2                 ALPHA(25,25),BETA(25,25),GAM(25,25),DELTA(25,25),
     3                 SRR(25,25),SII(25,25),AA(50,50),VR(50),VI(50)
      DIMENSION A(NAR,NAR),G(NAR,NAR),Q(NAR,NAR),P(NAR,NAR)
      DIMENSION RR(50),RI(50),H(50,50)
      DIMENSION IANA(50)
      COMMON /WORK/SSS,H
      EQUIVALENCE (AA,SRS,SSS),(SIS,SSS(2501))
      EQUIVALENCE (SRS,SR),(SIS,SI),(SRS(1251),ALPHA),(SRS(1876),BETA),
     1            (SIS(1251),GAM),(SIS(1876),DELTA),
     2            (SRS(626),SRR),(SIS(626),SII),
     3            (SRS,AR),(SIS,AI)
      EQUIVALENCE  (H,RTM), (H(1,3),ITM), (H(1,5),SCL)
C
C     HAMILTONIAN COMPUTATION (H)
C
      NSTP = 50
      DO 10I=1,NS
      I2=NS+I
      DO 10 J=1,NS
      J2=NS+J
      H(I,J)   =-A(I,J)
      H(I2,J)  = Q(I,J)
      H(I,J2)  = G(I,J)
   10 H(I2,J2) = A(J,I)
      WRITE (6,1010)
 1010 FORMAT(1H0,25X,18HHAMILTONIAN MATRIX//)
C
C     DETERMINATION OF EIGENVALUES AND EIGENVECTORS OF H
C
      N2=2*NS
      CALL PRMAT(H,N2,N2,50,50)
C
C     SELECTION OF UNSTABLE EIGENVALUES AND THEIR VECTORS
C
      DO 20 I =1,N2
      DO 20J = 1,N2
   20 AA(I,J) = H(I,J)
      CALL CHARD (AA,N2,RR,RI,0.0,1,NSTP)
      KREG = 0
      NOIMAG = 0
      NPASS = 0
      DO 50 L = 1,N2
      IF (RR(L)) 50,25,30
   25 WRITE (6,180)
  180 FOPMAT (1H0,71HTHE HAMILTONIAN HAS AN IMAGINAPY ROOT, NO STEADY-ST
     *ATE SOLUTION EXISTS  )
      RETURN
   30 CONTINUE
      IF (RI(L).GT.0.0) NOIMAG = 1
      IF (RI(L).LT.0.0) GO TO 50
      KREG = KREG + 1
      ROOTR = RR(L)
      ROOTI = RI(L)
      CALL IGVEC5( H,ROOTR,ROOTI,N2,NSTP,VR,VI,NPASS)
      DO 32 I=1,N2
   32 SR(I,KREG) = VR(I)
      IF(ABS(RI(L)/RR(L)).LT.0.000001) GO TO 50
      KREG = KREG + 1
      DO 38 I=1,N2
   38 SR(I,KREG) = VI(I)
   50 CONTINUE
      IF(KREG.EQ.NS) GO TO 60
      WRITE (6,190)KREG,NS
  190 FORMAT (1H0,27HTHE RICS2 SUBROUTINE FOUND ,I3,38H STABLE ROOTS INS
     *TEAD OF THE REQUIRED ,I3 )
      RETURN
   60 CONTINUE
C
C     COMPUTATION OF P FROM T11 AND T21
C
      DO 70 I=1,NS
      DO 70 J=1,NS
      J2 = NS + J
      AR(I,J) = SR(J,I)
   70 AI(I,J) = SR(J2,I)
      CALL SID (AR,NS,25,25,AI,NS,25,SIG,IER,RTM,ITM,SCL)
      DO95I=1,NS
      DO95J=1,NS
   95 P(I,J) = AI(I,J)
C
C     CHECK SOLUTION
C
      DO100I=1,NS
      DO100J=1,NS
      H(I,J)=0.0
      DO100K=1,NS
  100 H(I,J)=H(I,J)+P(I,K)*G(K,J)
      DO110I=1,NS
      DO110J=1,NS
      SR(I,J)=Q(I,J)
      DO110K=1,NS
  110 SR(I,J)=SR(I,J)+P(I,K)*A(K,J)+A(K,I)*P(K,J)-H(I,K)*P(K,J)
      WRITE(6,120)
  120 FORMAT(1H0,28HRICSS2 CHECK SOLTUICN, P-DOT )
      CALL PRMAT (SR, NS, NS, NSTP, NSTP)
      RETURN
      END
      SUBROUTINE IGVEC5 (H,RR,RI,N,NDIM,VR,VI,NPASS)
C
C     THIS IS A GENERAL EIGENVECTOR SOLVER WHERE H IS A REAL MATRIX AND
C     THE EIGENVALUE IS REAL OR COMPLEX. IT SOLVES FOR V IN
C     HV = LV WHERE L IS THE INPUT EIGENVALUE
C
C                                       INPUTS
C         H       -  N ORDER SINGE PREC. MATRIX DIMENSIONED NDIM X NDIM
C         RR,RI   -  REAL AND IMAG. PARTS OF EIGENVALUES. DOUBLE PREC.
C         N       -  ORDER OF MATRIX AND VECTORS
C         NDIM    -  FIXED DIMENSION LIMITS OF H,VR,AND VI.
C                                       OUTPUTS
C         VR,VI   -  REAL AND IMAG. EIGENVECTOR ARRAYS - DOUBLE PREC.
C                                      IN - OUT
C         NPASS   -  PASS FLAG IF SET 0 ON INPUT WILL ALWAYS CALCULATE
C                    H**2 FOR COMPLEX ROOT.  SET  BY IGVEC5 TO 1 IF
C                    A**2 CALCULATED.
C
      DIMENSION H(NDIM,NDIM)
      DIMENSION A(50,50), VR(NDIM), VI(NDIM), NCOL(50)
      COMMON/WORK/DUMMY(12900),NCOL,A
      T = RI
      IF (T.EQ.0.0) GO TO 35
      IF (NPASS.GT.0) GO TO 10
C
C                A = H**2
C
      CALL MATMSP (H, H, A, N, NDIM, NDIM, 50)
      REWIND 4
      WRITE (4) A
      REWIND 4
      NPASS = 1
      GO TO 15
   10 CONTINUE
      READ (4)  A
      REWIND 4
   15 CONTINUE
C
C                A =  H**2 - 2.0*RR*H(I,J) + RR**2 + RI**2  FOR RI NOT 0.
C
      DO 20 I = 1,N
      DO 20 J = 1,N
   20  A(I,J) =  A(I,J) - 2.0*RR*H(I,J)
      DO 30 I = 1,N
   30  A(I,I) = A(I,I) + RR*RR + RI*RI
      GO TO 55
   35 CONTINUE
C
C                A = H - RR    FOR  RI = 0.0
C
      DO 40 I = 1,N
      DO 40 J = 1,N
   40  A(I,J) = H(I,J)
      DO 50 I = 1,N
   50  A(I,I) =  A(I,I) - RR
C
   55 CONTINUE
C
```

```
C        NORMALIZE MATRIX BY MAKING MAX. ELEMENT  1.0
C
      BIG = 10.0E-25
      IBIG = 0
      JBIG = 0
      DO 70 I = 1,N
      DO 70 J = 1,N
      X   = ABS(A(I,J))
      IF (X.LT.BIG) GO TO 70
      BIG = X
      IBIG = I
      JBIG = J
   70 CONTINUE
      TEMP = A(IBIG,JBIG)
      DTEMP = 1.0 / TEMP
      DO 80 I = 1,N
      DO 80 J = 1,N
   80 A(I,J) = A(I,J) * DTEMP
      DO 90 I = 1,N
   90 NCOL(I) = I
C
C             SOLVE FOR X USING CROUT METHOD MAXIMIZING ALONG DIAGONAL
C             STOPPING WHEN DIAGONAL ELEMENTS REMAINING BECOME SMALL
      N1 = N - 1
      ICOLX = 0
      DO 200 K = 1,N
      K1 = K - 1
      BIG = 10.0E-26
      IBIG = 0
      DO 115 I = K,N
      VI(I) = A(I,I)
      IF (K.EQ.1) GO TO 115
      DO 110 L = 1,K1
  110 VI(I) = VI(I) - A(I,L)*A(L,I)
  115 CONTINUE
      IF (K.EQ.N) GO TO 220
      IF(T.EQ.0.0) GO TO 118
      IF (K.LT.N1)  GO TO 118
      X = VI(N)- VI(N1)
      X = ABS(X)
      IF (X.LT.10.E-30) GO TO 185
  118 CONTINUE
      DO 120 I = K,N
      X   = ABS(VI(I))
      IF (X.LT.BIG) GO TO 120
      IBIG = I
      BIG = X
  120 CONTINUE
      IF (IBIG.EQ.0) GO TO 185
      IF (IBIG.EQ.K) GO TO 140
C
C        MAKE SIMILABITY TRANSFORMATION BY INTERCHANGING
C        COLS K AND IBIG  AND ROWS  K AND IBIG
C
      I = NCOL(IBIG)
      NCOL(IBIG) = NCOL(K)
      NCOL(K) = I
      DO 125 I = 1,N
```

```
            TEMP = A(I,K)
            A(I,K) = A(I,IBIG)
        125 A(I,IBIG) = TEMP
            DO 130 J = 1,N
            TEMP = A(K,J)
            A(K,J) = A(IBIG,J)
        130 A(IBIG,J) = TEMP
        140 CONTINUE
            IF (K.EQ.1) GO TO 165
            DO 150 I = K,N
            DO 145 L = 1,K1
        145 A(I,K) = A(I,K) - A(I,L)*A(L,K)
        150 CONTINUE
        165 CONTINUE
            KP1 = K + 1
            DO 180 J = KP1,N
            IF (K.EQ.1) GO TO 175
            DO 170 L = 1,K1
        170 A(K,J) = A(K,J) - A(K,L)*A(L,J)
        175 A(K,J) = A(K,J)/A(K,K)
        180 CONTINUE
            GO TO 190
        185 CONTINUE
            ICOLX = K
            GO TO 210
        190 CONTINUE
        200 CONTINUE
        210 CONTINUE
            IF (ICOLX.LT.N1) GO TO 900
            IF (ICOLX.GT.N1) GO TO 220
            A(N1,N1) = VI(N1)
            N2 = N-2
            DO 215 I = 1,N2
            A(N,N1) = A(N,N1) - A(N,I)*A(I,N1)
        215 A(N1,N) = A(N1,N) - A(N1,I)*A(I,N)
        220 CONTINUE
            A(N,N) = VI(N)
            X   = ABS(VI(N))
            IF (X.LT.10.0E-12) GO TO 230
            WRITE(6,1225) X,RR,RI
       1225 FORMAT(1H0,//20X,18HIGVEC5 - X,RR,RI  ,3E18.8)
        230 CONTINUE
            VI(N) = 1.0
            DO 250 K = 1,N1
            I = N - K
            VI(I) = 0.0
            I1 = I + 1
            DO 240 L = I1,N
        240 VI(I) = VI(I) - A(I,L)*VI(L)
        250 CONTINUE
            DO 260 I = 1,N
            J = NCOL(I)
        260 VR(J) = VI(I)
      C                 CALCULATE VI  = - 1/RI*(H - RR*I)*VR
            IF (T.EQ.0) GO TO 335
            DO 300 I = 1,N
            VI(I) = - RR*VR(I)
            DO 290 J = 1,N
```

6a

```
  290 VI(I) = VI(I) + H(I,J)*VR(J)
  300 VI(I) = - VI(I)/RI
      RETURN
  335 CONTINUE
      DO 350 I = 1,N
  350 VI(I) = 0.0
      RETURN
  900 CONTINUE
      WRITE (6,1900) ICOLX,(VI(I),I=1,N)
 1900 FORMAT(1H0,//20X,22HIGVEC5 ERROR - ICOLX =,I10/(6X,10E12.4))
      RETURN
      END
      SUBROUTINE MATMSP (A, B, C, N, NA, NB, NC)
      DIMENSION  A(NA,NA), B(NB,NB), C(NC,NC)
      DO 20 I = 1,N
      DO 20 J = 1,N
      SUM = 0.0
      DO 10 K = 1,N
      SUM = SUM + A(I,K)*B(K,J)
   10 CONTINUE
      C(I,J) = SUM
   20 CONTINUE
      RETURN
      END
```

```
          SUBROUTINE PRMAT (ARAY,MS,NS,MAR,NAR)
          DIMENSION ARAY(MAR,NAR)
      500 FORMAT (1H ,E12.5,6(4X,E12.5))
      501 FORMAT (1H0,E12.5,6(4X,E12.5))
          DO10J2=1,NS,7
          WRITE(6,501)
          NJ=NS-J2
          IF(NJ.LT.7) GO TO  3
          J3=J2+6
          GOTO 5
        3 J3=NS
        5 DO10I=1,MS
       10 WRITE(6,500)(ARAY(I,J),J=J2,J3)
          RETURN
          END
      SUBROUTINE SID (A, N, NDROW, NDCOLA, B, M, NDCOLB, SIGDIG, IERROR,
     *   PIVOT, INDEX, SCALEB )
C
C     SID - A SINGLE PRECISION SIMULTANEOUS EQUATION SOLVER, INVERSE
C                      FINDER, AND DETERMINANT SUBROUTINE
C
      DIMENSION A(NDROW,NDCOLA), B(NDROW,NDCOLB), PIVOT(N,3),
     *   SCALEB(M), INDEX(N,3)
      DOUBLE PRECISION  DBIGP2
      DATA         DBIGP2
     *  /  7378697629483829.D4          /
C
      EPS = 1.E-3
  712 EPS = EPS/2.
      EPSP15= EPS + 1.5
      IF (EPSP15 .NE. 1.5)  GO TO 712
      SIGMCH = ALOG10(1.522/EPS)
      BIGPW2 = DBIGP2
      PIVOT(1,1) = 0.
C
C     SCALE ROWS
C
      DO 38 I=1,N
      ROWMX = 0.
      DO 28 J=1,N
      IF ((ABS(A(I,J))) .GT. ROWMX)  ROWMX =  ABS(A(I,J))
   28 CONTINUE
      IF ( ROWMX)  29, 750, 29
   29 CONTINUE
      ROWMXI = 1. / ROWMX
      DO 32 J=1,N
      AIJ = A(I,J)
      A(I,J) = (A(I,J) * ROWMXI ) * BIGPW2
      IF (A(I,J) .EQ. 0.)  A(I,J) = (AIJ * BIGPW2) * ROWMXI
   32 CONTINUE
      IF (M) 34, 38, 34
   34 DO 36 J=1,M
      BIJ = B(I,J)
      B(I,J) = (B(I,J) * ROWMXI ) * BIGPW2
      IF (B(I,J) .EQ. 0.)  B(I,J) = (BIJ * BIGPW2) * ROWMXI
   36 CONTINUE
   38 PIVOT(I,2) = ROWMXI
C
C     SCALE COLUMNS
C
      DO 10 J=1,N
      COLMX = 0.
      DO 4 I=1,N
      IF (ABS(A(I,J)).GT. COLMX) COLMX = ABS(A(I,J))
    4 CONTINUE
      IF ( COLMX )  5, 750, 5
    5 CONTINUE
      COLMXI = 1./COLMX
      DO 8 I=1,N
      AIJ = A(I,J)
      A(I,J) = (A(I,J) * COLMXI)* BIGPW2
      IF (A(I,J) .EQ. 0.)  A(I,J) = (AIJ * BIGPW2) * COLMXI
    8 CONTINUE
   10 PIVOT(J,3) = BIGPW2 * COLMXI
      IF (M) 14,24,14
   14 DO 22 J=1,M
      COLMX = 0.
      DO 16 I=1,N
      IF (ABS(B(I,J)).GT. COLMX )  COLMX = ABS(B(I,J))
   16 CONTINUE
      IF (COLMX )  17, 22, 17
   17 CONTINUE
      SCALEB(J) = COLMX / BIGPW2
      COLMXI = 1./COLMX
      DO 20 I=1,N
      BIJ = B(I,J)
      B(I,J) = (B(I,J) * COLMXI) * BIGPW2
      IF (B(I,J) .EQ. 0.)  B(I,J) = (BIJ * BIGPW2) * COLMXI
   20 CONTINUE
   22 CONTINUE
   24 CONTINUE
C
C     INITIALIZATION
C
      PMONE=1.
      DO 42 J=1,N
      PIVOT(J,1) = 0.
   42 INDEX(J,3) =0
      DO 550 I=1,N
C
C     SEARCH FOR PIVOT ELEMENT
C
      ABPIVI=0.
   45 DO 105 J=1,N
   50 IF (INDEX(J,3)-1) 60,105,60
   60 DO 100 K=1,N
   70 IF (INDEX(K,3)-1) 80,100,80
   80 IF (ABS(A(J,K)) - ABPIVI) 100,100,85
   85 IROW=J
   90 ICOLUM=K
      ABPIVI=ABS(A(J,K))
  100 CONTINUE
  105 CONTINUE
      IF (I-1) 115,120,115
  115 IF ( ABPIVI .GE. PIVMIN ) GO TO 123
  120 PIVMIN=ABPIVI
      IF (ABPIVI) 123,750,123
  123 CONTINUE
      INDEX(ICOLUM,3)=1
      PIVOTI=A(IROW,ICOLUM)
      PIVOT(I,1) = PIVOTI
C
C     INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL
C
  130 IF (IROW-ICOLUM) 140, 260, 140
  140 PMONE=-PMONE
  150 DO 200 L=1,N
  160 SWAP=A(IROW,L)
  170 A(IROW,L)=A(ICOLUM,L)
  200 A(ICOLUM,L)=SWAP
  205 IF (M) 260, 260, 210
  210 DO 250 L=1, M
  220 SWAP = B(IROW,L)
  230 B(IROW,L) = B(ICOLUM,L)
  250 B(ICOLUM,L) = SWAP
  260 INDEX(I,1)=IROW
  270 INDEX(I,2)=ICOLUM
C
C     DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
      PIVINV=1.0/PIVOTI
  330 A(ICOLUM,ICOLUM) = BIGPW2
  340 DO 350 L=1,N
  350 A(ICOLUM,L)= A(ICOLUM,L)*PIVINV
  355 IF (M) 380, 380, 360
  360 DO 370  L=1,M
  370 B(ICOLUM,L) = B(ICOLUM,L)*PIVINV
C
C     REDUCE NON-PIVOT ROWS
C
  380 DO 550 L1=1,N
  390 IF(L1-ICOLUM) 400, 550, 400
  400 T=A(L1,ICOLUM)
      IF (T) 420,550,420
  420 A(L1,ICOLUM)=0.0
  430 DO 450 L=1,N
  450 A(L1,L)=A(L1,L)-A(ICOLUM,L)*T
  455 IF(M) 550, 550, 460
  460 DO 500 L=1,M
  500 B(L1,L) = B(L1,L) - B(ICOLUM,L)*T
  550 CONTINUE
C
C     INTERCHANGE COLUMNS
C
  600 DO 710 I=1,N
  610 L=N+1-I
  620 IF (INDEX(L,1)-INDEX(L,2)) 630, 710, 630
  630 JROW=INDEX(L,1)
  640 JCOLUM=INDEX(L,2)
  650 DO 705 K=1,N
  660 SWAP=A(K,JROW)
  670 A(K,JROW)=A(K,JCOLUM)
  700 A(K,JCOLUM)=SWAP
  705 CONTINUE
  710 CONTINUE
C
      PIVOT(1,1) = PIVOT(1,1) * PMONE
C
      SIGDIG = SIGMCH - ALOG10(BIGPW2/PIVMIN)
      IF (SIGDIG .LT. .85)  SIGDIG = 0.
C
C     UNSCALE INVERSE AND SOLUTION(S)
C
      DO 720 J=1,N
      ROWMXI = PIVOT(J,2)
      DO 720 I=1,N
      IF (ROWMXI .LT. 1.) GO TO 715
      A(I,J) = (A(I,J) * PIVOT(I,3)) * ROWMXI
      GO TO 720
  715 A(I,J) = A(I,J) * (PIVOT(I,3) * ROWMXI)
  720 CONTINUE
      IF (M) 725, 735, 725
  725 DO 730 J=1,M
      ROWMXI = SCALEB(J)
      DO 730 I=1,N
      IF (ROWMXI .LT. 1.) GO TO 728
      B(I,J) = (B(I,J) * PIVOT(I,3)) * ROWMXI
      GO TO 730
  728 B(I,J) = B(I,J) * (PIVOT(I,3) * ROWMXI)
  730 CONTINUE
  735 CONTINUE
C
      IERROR = 1
      RETURN
  750 IERROR = -1
      SIGDIG = 0.
      RETURN
      END
```

```
FORTRAN IV G1  RELEASE 1.1           POLYEV           DATE = 73219          08/52/18           PAGE 0001

      SUBROUTINE POLYEV
      COMMON /COEFER/PR(51),M,X,Y,AP,RP,RI,IERRR
      U = PR(1)
      V   = 0.0
      DO 20 I =2,M
      US = U
      U = X*U -Y*V +PR(I)
   20 V = X*V + Y*US
      AP  = ABS(U) + ABS(V)
      RETURN
      END
```

```
               SUBROUTINE LEMBRT
      C        THIS ROUTINE SYSTEMATICALLY FINDS A ROOT OF A POLYNOMIAL
      C        USING A SIMPLE CAGING SCHEME BASED ON D-ALEMBERTS LEMMA
               COMMON /COEFER/PR(51),M,X,Y,AP,RR,RI,IERRR
               DIMENSION NFLAG(5),U(5),V(5),P(5)
               EQUIVALENCE (P,P1),(P(2),P2),(P(3),P3),(P(4),P4),(P(5),P5)
               L = 1
               RR = 0.0
               RI = 0.0
               SIGN = 1.0
               IFLAG = 0
               JFLAG = 0
               KFLAG = 0
               DEL = 0.5
               DDEL = 8.0
               DO  5 I = 1,5
             5 NFLAG(I)=0
            10 IF (IFLAG.LT.5) GO TO 25
               IF (JFLAG.GT.0) GO TO 25
            20 IFLAG = 0
               IF (KFLAG.LT.3) GO TO 22
               RR = RR + SIGN/19.0
               RI = RI + SIGN/13.0
               SIGN = -2.0*SIGN
            21 CONTINUE
               IF (ABS(RI).LE.1.0) GO TO 22
               SIGN = SIGN/97.0
               RR = SIGN/3.0
               RI = -SIGN
               GO TO 21
            22 KFLAG = KFLAG + 1
               DEL = DDEL*DEL
               DDEL = DDEL + 1.3
            24 NFLAG(L) = 0
               GO TO 30
            25 IFLAG = IFLAG + 1
            30 CONTINUE
               DO 40 I = 1,5
               IF (NFLAG(I).NE.0) GO TO 38
               X  = RR
               Y  = RI
               IF (I.EQ.1) GO TO 35
               IF (I.EQ.2) X = X + DEL
               IF (I.EQ.3) X = X - DEL
               IF (I.EQ.4) Y = Y + DEL
               IF (I.EQ.5) Y = Y - DEL
            35 U(I) = X
               V(I) = Y
               CALL POLYEV
               P(I) = AP
            38 NFLAG(I) = 0
            40 CONTINUE
               IF (JFLAG.GT.27) GO TO 60
               DO 45 I = 1,5
               IF (P(I).GT. 1.0E-07) GO TO 48
            45 CONTINUE
               GO TO 60
            48 DIF1 = AMAX1(P1,P2,P3,P4,P5)
      DIF2 = AMIN1(P1,P2,P3,P4,P5)
      DIF = DIF1 - DIF2
      IF (( DIF.GE.1.0).AND.(P1.LT.1.0)) GO TO 55
      IF (P1.EQ.0.0) GO TO 60
      DIF = DIF/P1
      IF (DIF.LT.0.001)GO TO 20
   55 CONTINUE
   60 CONTINUE
      DO 70 J = 1,5
      I = J
      IF (P(J).EQ.0.0) GO TO 100
   70 CONTINUE
      DIF2 = AMIN1(P2,P3,P4,P5)
      IF (P1.GT.DIF2) GO TO 80
      IF (DEL.LT.10.0E-30) RETURN
      DEL = 0.5*DEL
      XX = RR + DEL
      YY = RI + DEL
      IF ((XX.EQ.RR).AND.(YY.EQ.RI)) RETURN
      IF ((XX.EQ.RR).AND.(RI.EQ.0.0)) RETURN
      IF ((RR.EQ.0.0).AND.(RI.EQ.YY)) RETURN
      IF ( JFLAG.GT.100)   GO TO 220
      JFLAG = JFLAG + 1
      NFLAG(1) = 1
      GO TO 30
   80 AMINY = P2
      N = 2
      DO 85 I=3,5
      IF (P(I).GT.AMINY) GO TO 85
      N = I
      AMINY = P(I)
   85 CONTINUE
      L = 3
      IF (N.EQ.3)  L = 2
      IF (N.EQ.4)  L = 5
      IF (N.EQ.5)  L = 4
      NFLAG(1) = 1
      NFLAG(L) = 1
      U(L) = U(1)
      U(1) = U(N)
      V(L) = V(1)
      V(1) = V(N)
      P(L) = P1
      P1   = P(N)
      RR = U(1)
      RI = V(1)
      GO TO 10
  100 RR = U(I)
      RI = V(I)
      RETURN
  220 IERRR= 2
      RETURN
      END
```

```
FORTRAN IV G1  RELEASE 1.1           VERIFY           DATE = 73219         08/52/18            PAGE 0001

                SUBROUTINE VERIFY(ROOTR,ROOTI,L,ASUB,APRIME,COLVEC     )
      C
      C
      C         THIS SUBROUTINE VERIFIES THE ROOTS OF  AN NTH DEGREE POLYNOMIAL.
      C         THE METHOD USES THE WARRING FORMULAE TO REPRODUCE THE COEFFICIENTS
      C         OF THE POLYNOMIAL. THIS SUBROUTINE ASSUMES ONLY REAL COEFFICIENTS.
      C
      C         THE POLYNOMIAL WILL BE NORMALIZED
      C
                REAL  NEG
                COMPLEX SUM,COLVEC,APRIME
                DIMENSIONROOTI(1),COLVEC(1),APRIME(1),ASUB(1),ROOTR(1)
      C
      C
      C
      C
      C
      C
                DO 3333 IAPAR=1,1
                IF (IAPAR .EQ. 2)  GO TO 3333
                LL=1
                SUM = (0.0,0.0)
      C
                ONE =  1.0
                NEG = -1.0
                ASUB(LL)=ONE
                IF(L.EQ.0)GO TO 99
                LL=LL+1
                DO 20 I = 1,L
                APRIME(I) = CMPLX(ROOTR(I), ROOTI(I))
                SUM  = APRIME(I) + SUM
             20 CONTINUE
      C
      C
      C                   STORE COFFICIENT WITH SIGN
      C
      C
             21 ONE=NEG*ONE
                ASUB(LL) = ONE * REAL(SUM)
                IF(LL.GT.L)GO TO 99
                SUM = (0.0,0.0)
                LL=LL+1
                IF(LL .EQ. 0)GO TO 99
      C
      C
      C                   FORM COLUMN VECTOR
      C
      C
                DO 30 I = 1,L
                COLVEC(I) = APRIME(I)
                APRIME(I) = (0.0,0.0)
             30 CONTINUE
      C
      C
      C                   CALCULATE NEW COLUMN VECTOR
      C
      C
                DO 41 I = 1,L
```

```
            DO 40 K = 1,L
            IF(K.GE.I)GO TO 40
            APRIME(I) = CMPLX(ROOTR(I), ROOTI(I)) * COLVEC(K) + APRIME(I)
       40 CONTINUE
            SUM = APRIME(I) + SUM
       41 CONTINUE
            GO TO 21
       99 RETURN
     3333 CONTINUE
            RETURN
            END
```

```
      SUBROUTINE CHARD(A,N,RR,RI,CRIT,IPPNT,NVAR)
C                    WHERE -
C              A IS A DOUBLE PREC. NVAR BY NVAR DIMENSIONED MATRIX
C              N IS ORDER OF MATRIX USED
C              RR,RI  STORAGE ARRAYS FOR NROOTS
C              CRIT IS DIVISOR CRITERIA (NORMALLY 0)
C              IPPNT  -  IF NOT ZERO ROUTINE PRINTS ROOTS
C                          PLUS POLY COEFF. INPUT TO RF AND AS
C                          COMPUTED BY VERIFY  - THERE MAY BE ONE OR MORE
C                          POLYS
C              NVAR IS DIM OF MATRIX(MAX)
C     CHARD VERSION OF MARCH 8,1967 - J.C. BIDWELL
C     THIS SUBROUTINE COMPUTES THE EIGENVALUES OF A REAL MATRIX
C     SYMMETRIC OR NONSYMMETRIC
C     THE INPUT MATRIX IS TRANSFORMED BY SIMILARITY TRANSFORMATIONS
C     INTO ONE OF THE FROBENIOUS FORMS WHERE ROW 1 CONTAINS ALL BUT
C     THE LEADING COEFFICIENT OF THE CHARACTERISTIC EQUATION -THE
C     LEADING COEFF. IS OF COURSE  1.0
C     ACCURACY IS INCREASED BY MAXIMIZING DIVISOR BY INTERCHANGING
C     ROWS AND COLS.
C     THE ROOTS OF THE CHARACTERISTIC EQ. ARE SOLVED USING
C     A D-ALEMBERT LEMMA TECHNIQUE
C     WHERE ALL VALUES IN A ROW TO THE LEFT OF THE DIAGONAL ARE LESS
C     THAN INPUT CRITERIOR (CRIT) PROGRAM SUBDIVIDES PROBLEM USING
C     RF  TO OPERATE ON TWO OR MORE LOWER ORDER POLYNOMIALS.
C     CHARD USES POLYRF,LEMBRT,POLYEV ROUTINES AND COEFER COMMON DATA
C     CHARD USES VERIFY ROUTINE
      COMPLEX  C, D
      DIMENSION A(NVAR,NVAR),RR(1),RI(1)
      DIMENSION XX(50),YY(50)
      DIMENSION  ROOTR(50), ROOTI(50), B(51), C(52), D(52),
     1           COEP(51), ROW(50), COL(50)
      CALL OVERFL(JACK)
C     THE CODING USING THE 3000 NUMBERS HAVE TO DO WITH A CUSTOM
C     MATRIX NORMALIZATION FOR A SPECIAL CLASS OF PROBLEMS
C     IF N GE 20 DIVIDE ALL MATRIX ELEMENTS BY 10.0
      DO 3050 I = 1,N
      DO 3050 J = 1,N
 3050 A(I,J) = A(I,J)/10.0
 3100 CONTINUE
      JACK=0
      M=N
      NR=0
    1 L=M
    2 K=L-1
      BIG=CRIT
      JJ=0
C     FIND LARGEST ROW ELEMENT TO LEFT OF DIAGONAL
      DO 10 J=1,K
      AA  = ABS(A(L,J))
      IF (AA.LE.BIG) GO TO 10
      BIG = AA
      JJ=J
   10 CONTINUE
C     IF ALL ELEMENTS LEFT OF DIAGONAL ARE LE CRITERIA GO TO COMPUTE
C     EIGENVALUES OF REDUCED MATRIX
      IF (JJ.EQ.0) GO TO 70
C     SHIFT ROWS AND COLS IF NECESSARY
```

```
      IF (JJ.EQ.K) GO TO 40
      DO 20 J=1,M
      X = A(JJ,J)
      A(JJ,J) = A(K,J)
   20 A(K,J)=X
      DO 30 I=1,L
      X= A(I,JJ)
      A(I,JJ)=A(I,K)
   30 A(I,K)=X
   40 CONTINUE
C     MAKE SIMILARITY TRANSFORMATION ON MATRIX
      DI  = 1.0 / A(L,K)
C     ROW IN EFFECT IS THE LEFT OR INVERSE SIMILARITY MATRIX
C     COL IN EFFECT IS THE RIGHT           SIMILARITY MATRIX
      DO 42 J=1,M
      ROW(J)=A(L,J)
   42 COL(J) = - ROW(J) * DI
      COL(K) = DI
C                 (ROW + I) * A   WHERE ROW IS KTH ROW,I THE IDENTITY MA
      DO 50 J=1,M
      SUM = 0.0
      DO 45 I=1,M
   45 SUM=SUM+A(I,J)*ROW(I)
   50 A(K,J)=SUM
C                 A * (COL + I)   WHERE COL IS KTH ROW,I THE IDENTITY MA
C     FIRST K ROWS LESS KTH  COL.
      DO 60 I=1,K
      DO 60 J=1,M
      IF (J.EQ.K) GO TO 60
      A(I,J)=A(I,J)+A(I,K)*COL(J)
   60 CONTINUE
C     LTH  ROW
      DO 65 J=1,M
   65 A(L,J) = 0.0
C     KTH COL
      A(L,K) = 1.0
      DO 68 I=1,K
   68 A(I,K)=A(I,K)*COL(K)
      L=L-1
      IF (L.EQ.1) GO TO 70
      GO TO 2
C     SET UP TO COMPUTE ROOTS OF REDUCED OR FULL MATRIX
   70 CONTINUE
      IF (L.EQ.M) GO TO 200
      COEP(1) = 1.0
      J=1
      DO 80 I=L,M
      J=J+1
      COEP(J)=-A(L,I)
   80 CONTINUE
C     J BECOMES DEGREE OF POLYNOMIAL
      J=J-1
      CALL OVERFL(JACK)
      IF (JACK.EQ.1) WRITE (6,1082)
 1082 FORMAT(1H0,15X,17HOVERFLOW IN CHARD)
      CALL POLYRF(COEP,J,XX,YY,IERR)
      IF (IERR.NE.0) WRITE (6,1085) IERR
 1085 FORMAT(1H0,10X,13HPOLRF  IERR =,I8)
```

```
C     STORE J ROOTS
      DO 90 I=1,J
      NR=NR+1
      ROOTR(I) = XX(I)
      ROOTI(I) = YY(I)
      RR(NR)=XX(I)
   90 RI(NR)=YY(I)
      IF (IPRNT.EQ.0) GO TO 100
C     PRINT COEFF FOR J ROOTS
      CALL VERIFY(ROOTR,ROOTI,J,B,C,D)
      WRITE (6,1092) J
 1092 FORMAT(1H0,15X,23HPOLYNOMIAL COEFFICIENTS//21X,I4,3X,5HROOTS//
     118X,5HINPUT,13X,6HOUTPUT)
      JJ=J+1
      DO 95 I=1,JJ
   95 WRITE (6,1095) COEP(I),B(I)
 1095 FORMAT(1H ,12X,E15.7,4X,E15.7)
  100 CONTINUE
      IF (NR.GE.N) GO TO 500
      M=N-NR
      IF (M.EQ.1) GO TO 220
      GO TO 1
C     ONE EIGENVALUE IS A DIAGONAL ELEMENT
  200 NR=NR+1
      RR(NR)=A(L,L)
      RI(NR)=0.0
  210 IF (IPRNT.EQ.2) WRITE (6,1210) NR,RR(NR)
 1210 FORMAT (1H0,10X,9HREAL ROOT,I6,4X,E15.8)
      IF(NR.EQ.N) GO TO 500
      IF(L.EQ.2) GO TO 220
      M=N-NR
      GO TO 1
  220 NR=NR+1
      RR(NR)=A(1,1)
      RI(NR)=0.0
      GO TO 210
C     PRINT OUT N ROOTS IF CALLED FOR
  500 CALL OVERFL(JACK)
      IF (JACK.EQ.1) WRITE (6,1510)
 1510 FORMAT(1H0,15X,15HOVERFLOW IN RF   )
C     IF N GE 20 MULT. ALL ROOTS BY 4.0
      DO 3150 I = 1,N
      RR(I) =10.0*RR(I)
 3150 RI(I) =10.0*RI(I)
 3200 CONTINUE
      IF (IPRNT.EQ.0) RETURN
      WRITE (6,1520)
 1520 FORMAT(1H0,25X,5HROOTS//10X,4HREAL,12X,4HIMAG)
      DO 540 I=1,N
  540 WRITE (6,1540) I,RR(I),RI(I)
 1540 FORMAT(1H ,2X,I4,4X,E15.8,4X,E15.8)
      RETURN
      END
      SUBROUTINE POLYRF(P,N,X,Y,IERR)
C     VERSION OF JUNE 28,1967  BY JC BIDWELL - A1,A2 - PARTS UNKNOWN
      DIMENSION  X(1), Y(1), P(1)
      COMMON /COEFER/PR(51),M,A,E,AP,RR,RI,IERRR
      IF ((N.LT.1).OR.(N.GT.50)) GO TO 200
      IERR = 0
      IERRR= 0
      J = 1
      M = N+1
      DO 10 I = 1,M
      PR(I) = P(I)
   10 CONTINUE
      IF (N.EQ.1) GO TO 100
   15 CONTINUE
      CALL LEMBRT
      IF (RI.EQ.0.0) GO TO 40
      IF (RR.EQ.0.0) GO TO 16
      TEST = RI/RR
      IF (ABS(TEST).LT.0.000001) GO TO 40
   16 CONTINUE
      X(J) = RR
      Y(J) = RI
      IF (IERRR.NE.0) GO TO 220
      IF(J.EQ.N) RETURN
      J = J + 1
      X(J) = RR
      Y(J) = -RI
      IF(J.EQ.N) RETURN
      J = J + 1
      M = M - 2
      X2 = 2.0*RR
      XY =  -(RR*RR + RI*RI)
      DO 20 I = 2,M
      PR(I) = PR(I) + X2*PR(I-1)
      PR(I+1) = PR(I+1) + XY*PR(I-1)
   20 CONTINUE
      IF (M.EQ.2) GO TO 100
      GO TO 15
   40 CONTINUE
      X(J) = RR
      Y(J) = 0.0
      IF (IERRR.NE.0) GO TO 220
      IF (J.EQ.N) RETURN
      J = J + 1
      M = M - 1
      DO 50 I = 2,M
   50 PR(I) = PR(I) + RR*PR(I-1)
      IF (M.EQ.2) GO TO 100
      GO TO 15
  100 D = PR(1)
      X(J) = -PR(2)/D
      Y(J) = 0.0
      RETURN
  200 WRITE (6,1200) N
 1200 FORMAT(1H0,10X, 3HN =,I4,21HOUTSIDE LIMITS POLYRF)
      IERR=1
      RETURN
  220 IERR = IERRR
```

```
FORTRAN IV G1  RELEASE 1.1          POLYRF          DATE = 73219          08/52/18          PAGE 0002

 0058                RETURN
 0059                END
```