# INSTITUT FÜR PLASMAPHYSIK

## GARCHING BEI MÜNCHEN


Solution of an ABEL type integral equation in the presence of noise.


Rudolf Gorenflo and Yehudith Kovetz



IPP/6/29 November 1964



*Die nachstehende Arbeit wurde im Rahmen des Vertrages zwischen dem Institut für Plasmaphysik GmbH und der Europäischen Atomgemeinschaft über die Zusammenarbeit auf dem Gebiete der Plasmaphysik durchgeführt.*

# Contents


page

Abstract 1

A. The case of continuously measured $\mathcal{J}(x)$ 2

1. Introduction 2
2. Smoothing by using GEGENBAUER polynomials 6
3. Avoiding intervals in which $i(r) < 0$ by quadratic programming 8

B. The case where only a few mean values of $\mathcal{J}(x)$ are measured 11

4. The problem of light-pipes 11
5. Approximation by a rational function 15
6. Approximation using GEGENBAUER polynomials 17
7. Reduction to a quadratic programming problem 21
8. Numerical experience 23
9. The case of an arbitrary radius R 26

C. References 28

D. Appendix 29

a) Description of the programs 29

b) Lists of the programs 34

c) Example of an output 49

d) Reconstruction of a given function 50

e) Sketches 51

IPP/6/29 R. Gorenflo
Y. Kovetz

Solution of an ABEL type integral equation in the presence of noise, November, 1964 (in English).

ABSTRACT:

The solution i(r), $0 \leq r \leq 1$, of the integral equation

$$J(x) = 2 \int_{r=|x|}^{1} \frac{i(r)\ r\ dr}{\sqrt{r^2 - x^2}},$$

arising in spectroscopy, can be obtained from $J(x)$ by a half-order differentiation process after substituting $r^2 = 1-s$, $x^2 = 1-t$. Therefore noise in the measured values of $J$ is amplified when computing $i$ by usual numerical methods. In the computation of $i(r)$ two undesirable effects may arise:
(a) lack of smoothness, (b) intervals where $i(r) < 0$, although for physical reasons we should have $i(r) \geq 0$.

(a) and (b) can be avoided as follows:- fitting $J(x)$ as the sum of an optimum number of suitable orthogonal functions, and using the extra information $i(r) \geq 0$ as a restriction, leads by discretization to a quadratic programming problem for the coefficients of the fitting expansion. A modification of this method is applicable when, instead of a continuous $J(x)$ only a few (e.g. 7 or 8) mean values of $J$ (contaminated with noise) are measured over adjacent intervals in $-1 \leq x \leq 1$.

## A. The case of continuously measured $J(x)$

### 1. Introduction

The integral transform

$$(1) \qquad i(r) = -\frac{1}{\pi} \int_{x=r}^{1} \frac{d\,J(x)}{\sqrt{x^2 - r^2}}$$

where $-1 \le x \le 1$, $0 \le r \le 1$, $J(1) = 0$, $J(-x) = J(x)$, solves the ABEL type integral equation

$$(2) \qquad J(x) = 2 \int_{r=|x|}^{1} \frac{i(r)\, r\, dr}{\sqrt{r^2 - x^2}} ,$$

which arises in spectroscopy when, by "side-on" observation of a cylindrical source, a laterally integrated intensity $J(x)$ is measured from which the radial intensity $i(r)$ is to be computed (see for example [1] or [5], where further references are also given). Fig. 1 illustrates the geometry.

Substituting

$$(3) \qquad r^2 = 1 - s, \quad x^2 = 1 - t, \quad i(r) = g(s), \quad J(x) = G(t)$$

in (1) yields

$$(4) \qquad g(s) = \frac{1}{\pi} \int_{t=0}^{s} (s-t)^{-1/2}\, d\,G(t)$$

which, in terms of operational calculus ([4], 290 etc.), is half-order differentiation of $G$. Therefore noise of the measured function $J(x)$ appears amplified in the computed function $i(r)$ . Two unwanted effects may arise when applying usual numerical procedures for computing $i(r)$:

a) lack of smoothness in $i(r)$, that is irregular high-frequency oscillations, superimposed on $i(r)$, which do not have a physical meaning.

b) intervals in which $i(r) < 0$, although for physical reasons we should have $i(r) \ge 0$ everywhere.

In figures 2 and 3 examples of effects (a) and (b) are presented.

A heuristic theory of noise amplification is developed in [5], 22-27. The essential results are as follows:

Inverting (2) by any standard discrete method is a process of linear transformation of a vector $(J_0, J_1, \dots, J_N)$ representing readings of $J$ in equidistant points x, into a vector $(i_0, i_1, \dots, i_N)$. By a suitable idealization the errors of $J$ are

(5) $$\delta_n = \varepsilon_n + \eta_n ,$$

where the strongly correlated $\varepsilon_n$ stand for the systematic error of the continuously working measuring device, whereas the $\eta_n$ are reading and rounding errors of the scanning process giving $(J_0, J_1, \dots, J_N)$ from the measured continuous function $J(x)$. The smooth $\varepsilon_n$ may yield effect (b); the $\eta_n$ are responsible for (a).

Let us consider the $\eta_n$ only (this is justified if the $\varepsilon_n$ are sufficiently small). Because of linearity we can approximately evaluate their influence by applying (1) to

(6) $$J(x) = \begin{cases} \eta_{j-1} , & (j-1)/N \le x < j/N, \quad 1 \le j \le N, \\ 0 , & x = 1, \end{cases}$$

where the $\eta_j$ are uncorrelated random variables with expectation 0 and standard deviation $\sigma$. The worst case is to be expected where $r = 0$. From

(7) $$i(0) = \frac{N}{\pi} \left( \eta_0 - \sum_{j=2}^{N} \frac{\eta_{j-1}}{(j-1)\, j} \right)$$

we conclude that $i(0)$ has expectation $0$ and standard deviation

(7') $CN\sigma$ with $$C = \frac{1}{\pi} \left\{ 1 + \sum_{2}^{N} (j-1)^{-2} j^{-2} \right\}^{1/2} .$$

We have $C \approx 0.36$ for $N \gg 1$. MONTE CARLO calculations confirmed that this is realistic in magnitude.

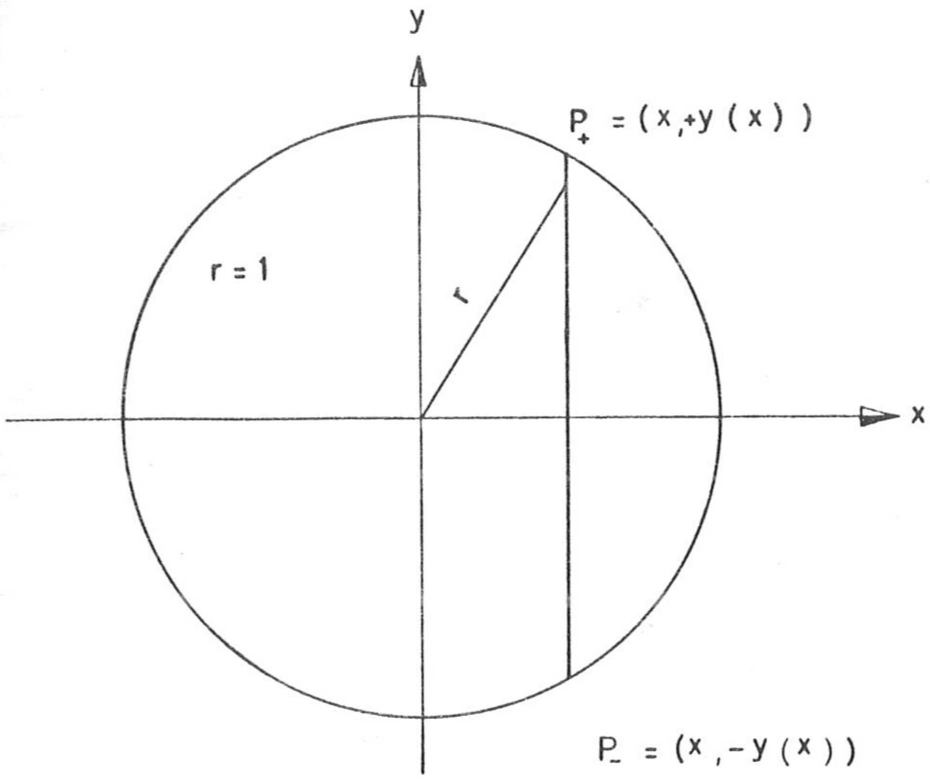


Fig 1

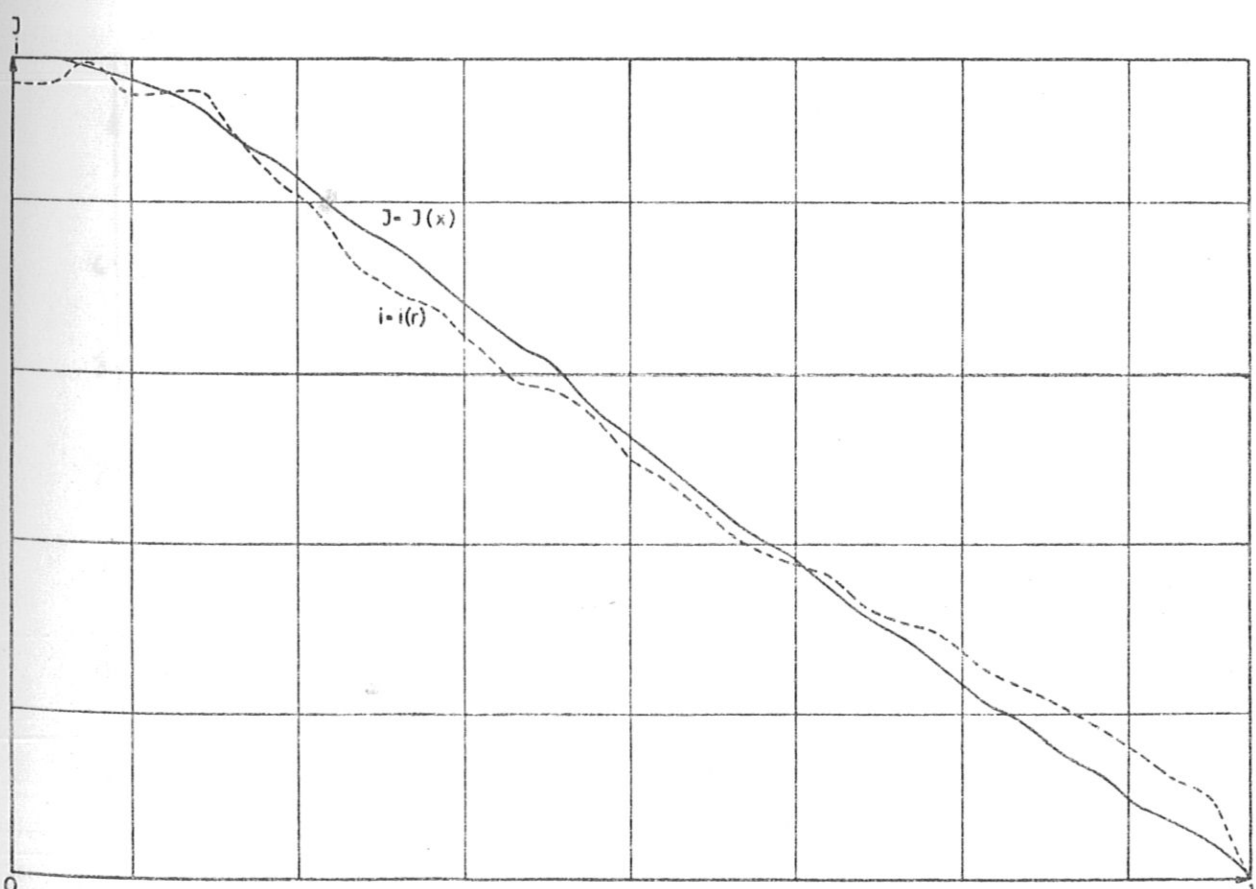


Fig 2

An illustration of effect (a): i(r) not smooth.

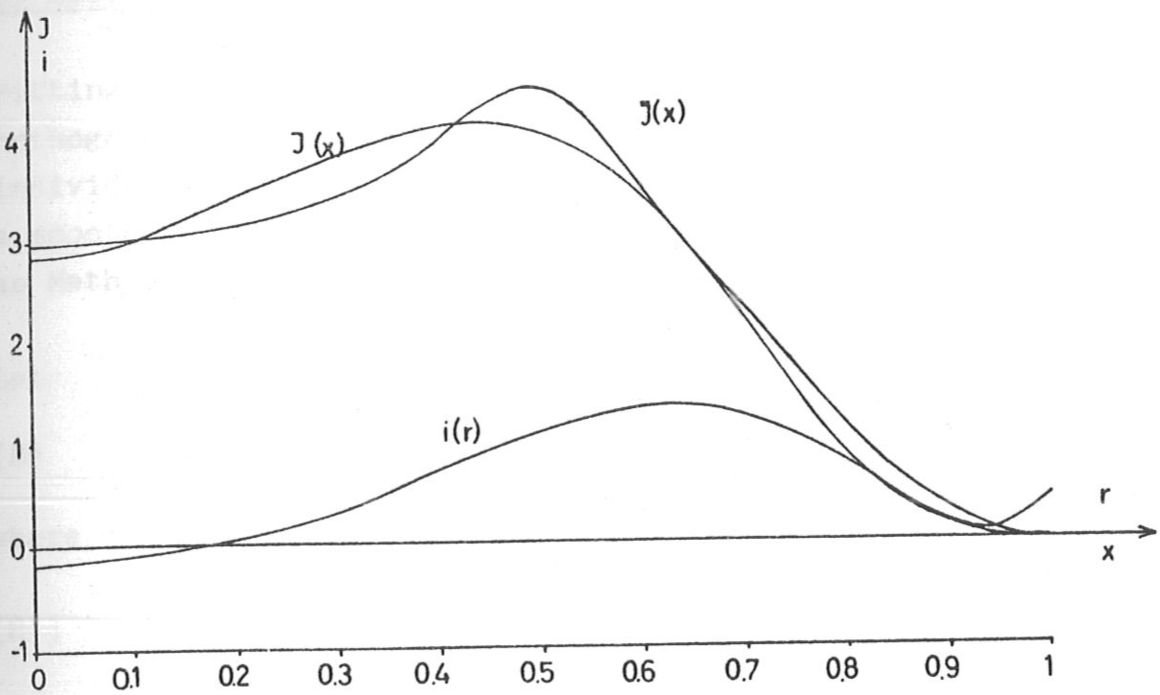


Fig.: 3

An illustration of effect (b): i(r) negative near r = o, $\bar{J}(x)$ = measured function, $J(x)$ = unrestricted least-squares fit to $\bar{J}(x)$, $k=4$.

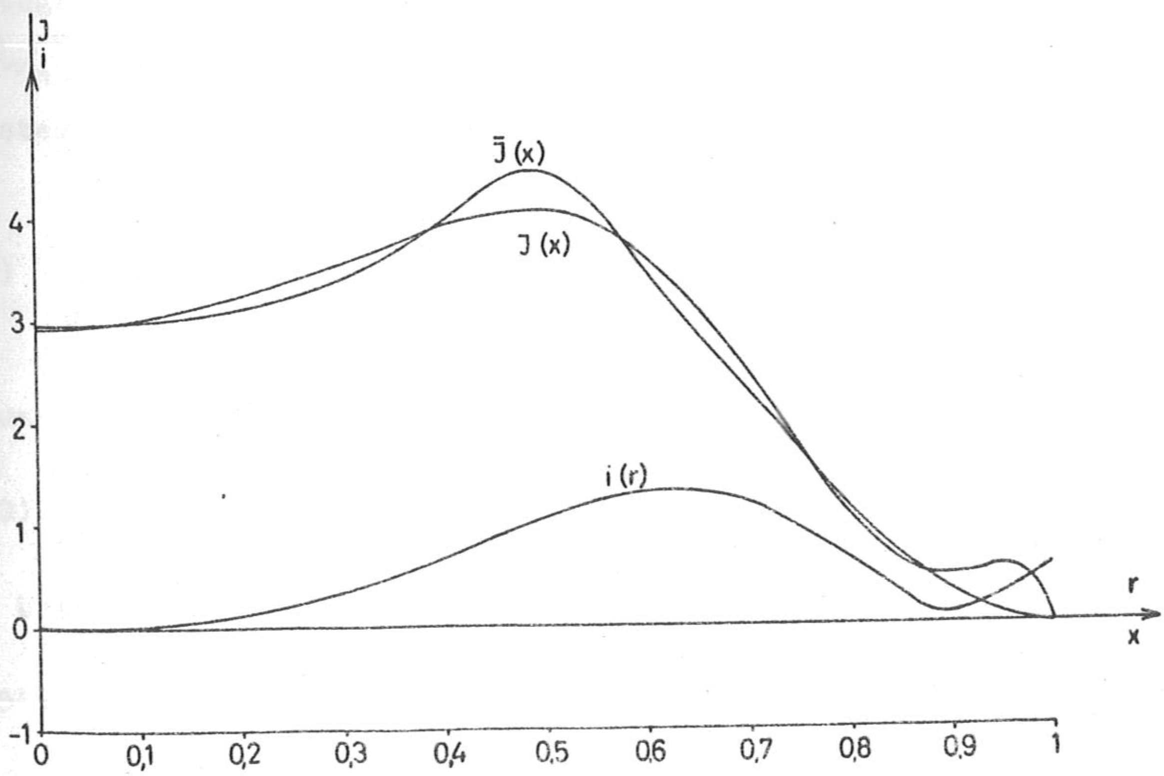


Fig.: 4

$\bar{J}(x)$ as in fig. 3, $J(x)$ = restricted least-squares fit, $i(r) \geq 0$ everywhere, $k=4$.

## 2. Smoothing by using GEGENBAUER polynomials

Fitting $J(x)$ as the sum of an optimum number of suitable orthogonal functions, for which the transforms (1) are individually known, is a natural procedure for obtaining a smooth function $i(r)$. One such procedure is developed as Method $D$ in [5]; numerical experiences are also given.

Let

(8) $$W_n^*(x) = \sqrt{1 - x^2}\; U_n(x),$$

where

(8') $$U_n(x) = \sum_{p=0}^{n} \beta(p,n)\, x^{2p},$$

with coefficients

(8") $$\beta(p,n) = 4^p \binom{-3/2}{n+p} \binom{n+p}{n-p},$$

is (in TRICOMI's notation, [8], 177 etc.) the GEGENBAUER polynomial $C_{2n}^{3/2}(x)$. For even functions $J(x)$, square-integrable in $-1 \leq x \leq 1$, the system $\{W_n^*,\ n = 0, 1, 2, \ldots\}$ is a complete orthogonal system with

(9) $$\int_0^1 W_m^*(x)\, W_n^*(x)\, dx = \begin{cases} 0, & m \neq n \\ \dfrac{(2n+1)(2n+2)}{4n+3}, & m = n. \end{cases}$$

Thus

(10) $$J(x) = \sum_{n=0}^{k} c_n W_n^*(x)$$

is transformed by (1) into

(10') $$i(r) = \frac{1}{2} \sum_{n=0}^{k} c_n P_n^*(r)$$

where

$$(11)\qquad P_n^*(r) = \sum_{p=0}^{n} \gamma(p,n)\, r^{2p} .$$

The $\gamma$ may be computed by solving the triangular systems of linear equations

$$(12)\qquad \sum_{\lambda=p}^{n} b(p,\lambda)\,\gamma(\lambda,n) = \beta(p,n), \quad 0 \le p \le n,\ n \ge 0,$$

where

$$(13)\qquad b(p,n) = \sum_{j=n-p}^{n} \frac{(-1)^{j+p-n}}{2j+1} \binom{n}{j}\binom{j}{n-p} .$$

The numerical procedure consists of fitting to $J(x)$ a least-squares approximation

$$(14)\qquad \sum_{n=0}^{k} c_n W_n^*(x) \approx J(x),$$

where the FOURIER coefficients

$$(14')\qquad c_n = \frac{4n+3}{(2n+1)(2n+2)} \int_0^1 J(x)\, W_n^*(x)\, dx$$

are computed by the SIMPSON integration formula. The measurements are to be taken at $x_j = j/N,\ \ 0 \le j \le N$, and $N$ must be an even integer.

By this method we obtained sufficiently smooth functions $i(r)$ in actual cases, with $20 \le N \le 30,\ \ k \approx 5$. Unfortunately $i(r)$ occasionally had intervals of negativity, especially in such cases where $J(x)$ had a local minimum at $x = 0$ . This was due, either to excessive measurement errors or to errors introduced by truncation of the series, or both. Of course $k$ should not be too large in the presence of noise. If $k$ is too large, the unphysical high-frequency oscillations are fitted too well. However, if $k$ is just sufficiently small that the noise is suppressed it may be so small that the truncation error is unacceptable.

## 3. Avoiding intervals in which $i(r)<0$ by quadratic programming

The idea that the extra information $i(r) \geq 0$ can be used as a restriction on the fit (14) leads to a way out of the dilemma mentioned at the end of § 2. Discretization yields a quadratic programming problem:

Choose $c_0, c_1, \ldots, c_k$ in order to

$$\text{minimize} \sum_{j=0}^{N-1} \left\{ J(j/N) - \sum_{n=0}^{k} c_n W_n^*(j/N) \right\}^2 \tag{15}$$

$$\text{subject to} \sum_{n=0}^{k} c_n P_n^*((j-1)/m) \geq 0, \quad j = 1, 2, \ldots, m.$$

Implementing this idea a double-precision FORTRAN routine was written and successfully applied with values of $N$ between 20 and 30, and $m \approx 20$, $k \approx 5$. Problem (15) was solved by the iterative method of HILDRETH and D'ESOPO ([7], 73-79). Average computer (IBM 7090) running time depends on $N$ and $m$; for $N \approx 25$ and $k \approx 5$ it was approximately 1 minute.

In figure 4 an example is presented.

We give some details of transforming (15) into the standard form of a quadratic programming problem. With

$$(16) \quad J_j = J((j-1)/N), \quad w(n,j) = W_n^*((j-1)/N), \quad 1 \leq j \leq N,$$

we have to minimize

$$(17)\quad D(c) = \sum_{j=1}^{N} \Big( J_j - \sum_{n=0}^{k} w(n,j)\, c_n \Big)^2$$

subject to the restrictions

$$(18)\quad \sum_{n=0}^{k} c_n\, P_n^*(r_\lambda) \geq 0, \quad r_\lambda = (\lambda-1)/m, \quad \lambda = 1, 2, \dots, m.$$

With $0 \leq i \leq k,\ 0 \leq n \leq k,\ 1 \leq \lambda \leq m,$

$$Q_{in} = Q_{ni} = \sum_{j=1}^{N} w(i,j)\ w(n,j),$$

$$b_n = -2 \sum_{j=1}^{N} w(n,j)\, J_j, \quad A_{\lambda n} = -P_n^*(r_\lambda)$$

we get the problem:-

(19) Minimize $D = c'Qc + b'c$
subject to $Ac \leq 0.$

Further details may be found in § 6 and § 7. In these paragraphs we have simply to replace $T_j$ by $J_j$, $\delta(n,j)$ by $w(n,j)$.

The matrix $Q$ is strictly positive definite, when $N \geq k+1$. In order to prove it we have only to prove that it is regular, since by (17)

$$\sum Q_{in}\, c_i c_n = \Big(\sum_{\nu=0}^{k} c_\nu \vec{w}_\nu\Big)^2, \quad \text{where } \vec{w}_\nu = (w(\nu,1), \dots, w(\nu,N)),$$

so that $Q$ is positive semi-definite.

By definition

$$Q = W \cdot W'.$$

We shall prove first that the rank of $W$, $r(W) = k+1$. $W$ is a matrix of $(k+1)$ rows and $N$ columns.

$$w(n,j) = M_j \sum_{p=0}^{n} \beta(p,n)\, s_j^p \qquad \begin{pmatrix} n = 0, 1, \dots, k \\ j = 1, 2, \dots N \end{pmatrix}$$

where $s_j = \left(\frac{j-1}{N}\right)^2$

and $M_j = \sqrt{1 - s_j} \neq 0$.

Let $\lambda_n$ $(n=0, 1, \ldots k)$ be $(k+1)$ constants satisfying

$$\text{(a)} \quad \sum_{n=0}^{k} \lambda_n M_j \sum_{p=0}^{n} \beta(p,n)\, s_j^p = 0 \qquad \text{for } j=1, \ldots, N.$$

Because $M_j \neq 0$ we get, by changing the order of summation,

$$\text{(a')} \quad \sum_{p=0}^{k} \Big( \sum_{n=p}^{k} \lambda_n \beta(p,n) \Big) s_j^p = 0 \qquad (j = 1, 2, \ldots, N).$$

The expression

$$p^*(s) = \sum_{p=0}^{k} \Big( \sum_{n=p}^{k} \lambda_n \beta(p,n) \Big) s^p$$

defines a polynomial of degree $k$, and therefore cannot have more than $k$ zeros, unless $p^*(s) \equiv 0$.
The system (a') expresses the fact that $p^*(s)$ vanishes for $N$ different values of $s$, and therefore it must be identically zero.

This leads us to the triangular system of $k+1$ homogeneous equations, with the $k+1$ unknowns $\lambda_n$,

$$\sum_{n=p}^{k} \lambda_n \beta(p,n) = 0, \quad p = 0, 1, \ldots, k,$$

whose diagonal coefficients are, by (8"), $\beta(n,n) = 4^n \binom{-3/2}{2n} \neq 0$.

Therefore $\lambda_n = 0$ $(n = 0, 1, \ldots k)$ and all the $k+1$ rows of the matrix $W$ are linearly independent, i.e. $r(W) = k+1$.

Now, in order to prove that $Q = WW'$ is regular, we compute its determinant, making use of the algebraic theorem (see e.g. [9], pp. 64-65): –

"If A is matrix of order $(m,n)$, and B is a matrix of order $(n,m)$, and if $m \leq n$, then $\det(AB)$ is equal to the sum of products of the corresponding majors of A and B".

The corresponding majors of $W$ and $W'$ are determinants of order $k+1$, each major of one being formed from the transposed matrix of the major of the other. $det(Q)$ is therefore the sum of $\binom{N}{k+1}$ squares of determinants of order $k+1$, and therefore $det(Q) \geq 0$. As $r(W) = k+1$, $W$ has at least one major which is not zero, and therefore $det(Q) > 0$.

If $N \geq 20$, we should have, of course, $k \lesssim 10$, because otherwise the fit would be insufficiently smooth.

B. The case where only a few mean values of $J(x)$ are measured

4. The problem of light-pipes

In this and the following paragraphs we consider the following problem:

Determine in $r \leq 1$ a radial non-negative intensity $i(r)$ from $N$ "side-on" measurements,

(20) $$T_j = \int_{I_j} J(x)\, dx, \qquad 1 \leq j \leq N.$$

For convenience we put

(21) $$i(r) = 0 \quad \text{in} \quad r > 1, \qquad J(x) = 0 \quad \text{in} \quad |x| \geq 1.$$

The intervals $I_j$ are defined by

(22) $$\begin{cases} I_j = \{x \mid e_j < x < e_{j+1}\}, \\ e_j = \alpha + (j-1)h. \end{cases}$$

$h$ is a positive constant depending on the experimental configuration, $\alpha = e_1$ is the left-hand endpoint of the first interval. $\alpha = e_1$ and $e_{N+1} = \alpha + Nh$ may or may not lie in $-1 < x < 1$, but $e_2$ and $e_N$ must lie in this interval. From these conditions we obtain

$$-1-h < \alpha < 1-(N-1)h \quad \text{and} \quad 0 < h < 2/(N-2).$$

In an actual series of experiments with cylindrical plasma discharges we had $N=7$ or $N=8$. One possible configuration is sketched in fig. 5.

Because of the complicated nature of the experiment and some uncontrollable disturbances $\alpha$ is not exactly known, and the intensity $i(x,y)$ in $x^2+y^2 \leq 1$ is not exactly a radial intensity $i(r)$. However, in order to draw conclusions from the results $T_j$ of the experiment, we assume $i$ to be a function of radius only. As pointed out by BRACEWELL in [2], it is impossible to determine an arbitrary intensity $i(x,y)$ by side-on observation in one direction only.

The difficulty in finding an effective procedure for computing $i(r)$ is due also to the fact that $T_j$ is not exactly equal to $\int_{I_j} J(x)\,dx$, but is disturbed by the "noise" of the measuring apparatus, which is estimated to be as great as 10 to 20% of the maximum of the $T_j$. We suppose the $T_j$ to be disturbed independently of each other.

Two problems arise:

1) Determine $\alpha$, or, equivalently, the centre of the circle in which $i(r)$ exists.

Primarily we have to determine the centre $r=0$ (that is the origin of our coordinate system), but for computational reasons we need the left-hand endpoint $\alpha = -\tilde{c}\,h$ of the first interval. $\tilde{c}$ is the (not necessarily integral) number of intervals $I_j$ to the left side of $x=0$ (fig. 6). For reasons to be explained in § 6 $\tilde{c}$ should not be an integral multiple of 1/2.

In the analysis of actual experimental results it was decided to fix the position of the centre on the basis of physical arguments, because of the lack of a satisfactory automatic method. This results in the introduction of some additional noise.

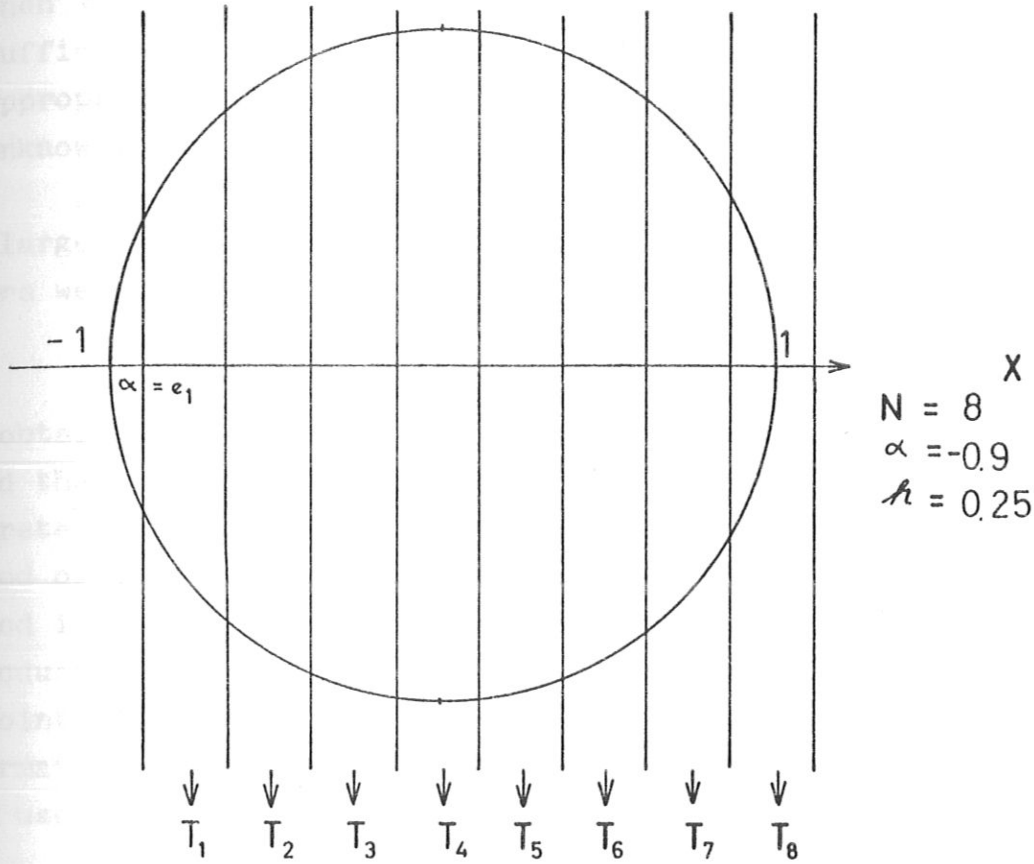
-1
1
X
α = e1
N = 8
α = -0.9
h = 0.25
T1
T2
T3
T4
T5
T6
T7
T8

Fig. 5

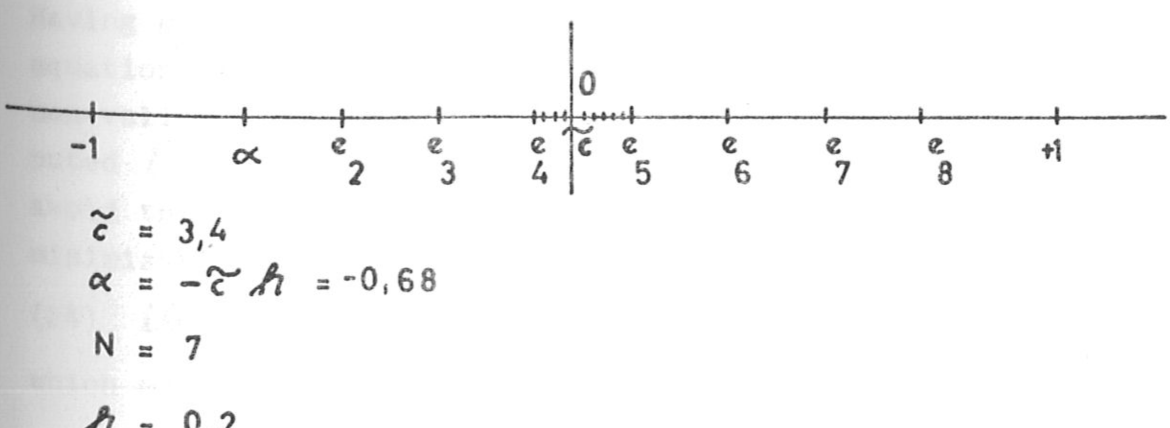
0
-1
α
e 2
e 3
e 4
c̃
e 5
e 6
e 7
e 8
+1
c̃ = 3,4
α = -c̃ h = -0,68
N = 7
h = 0.2

Fig. 6

2) When $\tilde{c}$, and therefore $\alpha$, is given, determine a sufficiently smooth function $J(x)$, lying in an appropriate function space, and approximating the unknown, true function.

For large values of $N$ and sufficiently small measurement errors we could put

$$J\left((e_j + e_{j+1})/2\right) \approx T_j / h$$

and obtain a smooth curve $J(x)$ by interpolation. $i(r)$ could then be computed, either by one of the usual discrete procedures, or by the quadratic programming method of § 3. However, for values of $N \lesssim 20$ this method is not to be recommended, because the error introduced by considering $T_j/h$ as the value of $J$ at the midpoint of the interval $I_j$ is too large. We have the information expressed in (20), and it is wise to make full use of it in the computational procedure.

This naturally leads to the idea of approximating, in an appropriate function space S, by the least-squares principle, which in our case reads as follows:-

$$\text{Minimize} \quad D = \sum_{j=1}^{N} \left( T_j - \int_{I_j} J(x)\, dx \right)^2,$$

(23)

$$\text{where} \quad J \in S.$$

Having computed $J(x)$, we have to solve the integral equation (2) for $i(r)$. Because of the "noise", and the non-validity of our symmetry assumptions, the computed $i(r)$ may turn out to be negative in places. To avoid this difficulty we can impose upon our minimization problem the restriction

(24) $i(r) \geq 0 \qquad \text{in} \quad r \leq 1,$

which may be approximated by

$$i(r_\lambda) \geq 0, \quad \lambda = 0, 1, 2, \ldots, m,$$

where $\{r_\lambda\}$ is a sufficiently dense set in $r \leq 1$.

## 5. Approximation by a rational function

The three main qualitative forms of physically plausible functions $\mathcal{J}(x)$ (arising in an actual series of cylindrical plasma discharges) are sketched in fig. 7. All of them are qualitatively obtainable by the rational function

$$(25) \quad \mathcal{J}(x) = (1-x^2)\left\{ b_1 + \frac{b_2}{1+b_3\,(x^2-b_4)^2} \right\}$$

with suitable coefficients $b_j$. The problem now is to find that point $b = (b_1, b_2, b_3, b_4)$ in the four-dimensional coefficient space, which solves (23). The integrals $\int_{I_j} \mathcal{J}(x)\, dx$ are expressible in terms of elementary functions, although the analytic calculation is tedious and the final expressions are very long.

We used two methods for minimizing $D = D(b)$.

$\alpha$) Direct search (see [6]). This is a method based on the idea of evaluating sequential trial solutions, and comparing each solution with the best one previously obtained, together with a strategy for determining the best direction for the next trial.

$\beta$) SHARE-program SCOOP. This is a similar procedure for minimizing functions of $n$ parameters, where a first "guess" is given.

It turned out that $D(b)$ has many local minima, and therefore it is very important to find a good first guess, i.e. a point which is sufficiently near to the absolute minimum (methods $\alpha$) and $\beta$) yield a local minimum). We decided to try a random search for finding a suitable starting point, but it transpired that this required a very long computer running time (in the order of 30 minutes for a single curve). Furthermore, although the approximation of $\mathcal{J}(x)$ was quite good, $i(r)$ was negative near $r = 0$, in absolute value up to 25% of the maximum

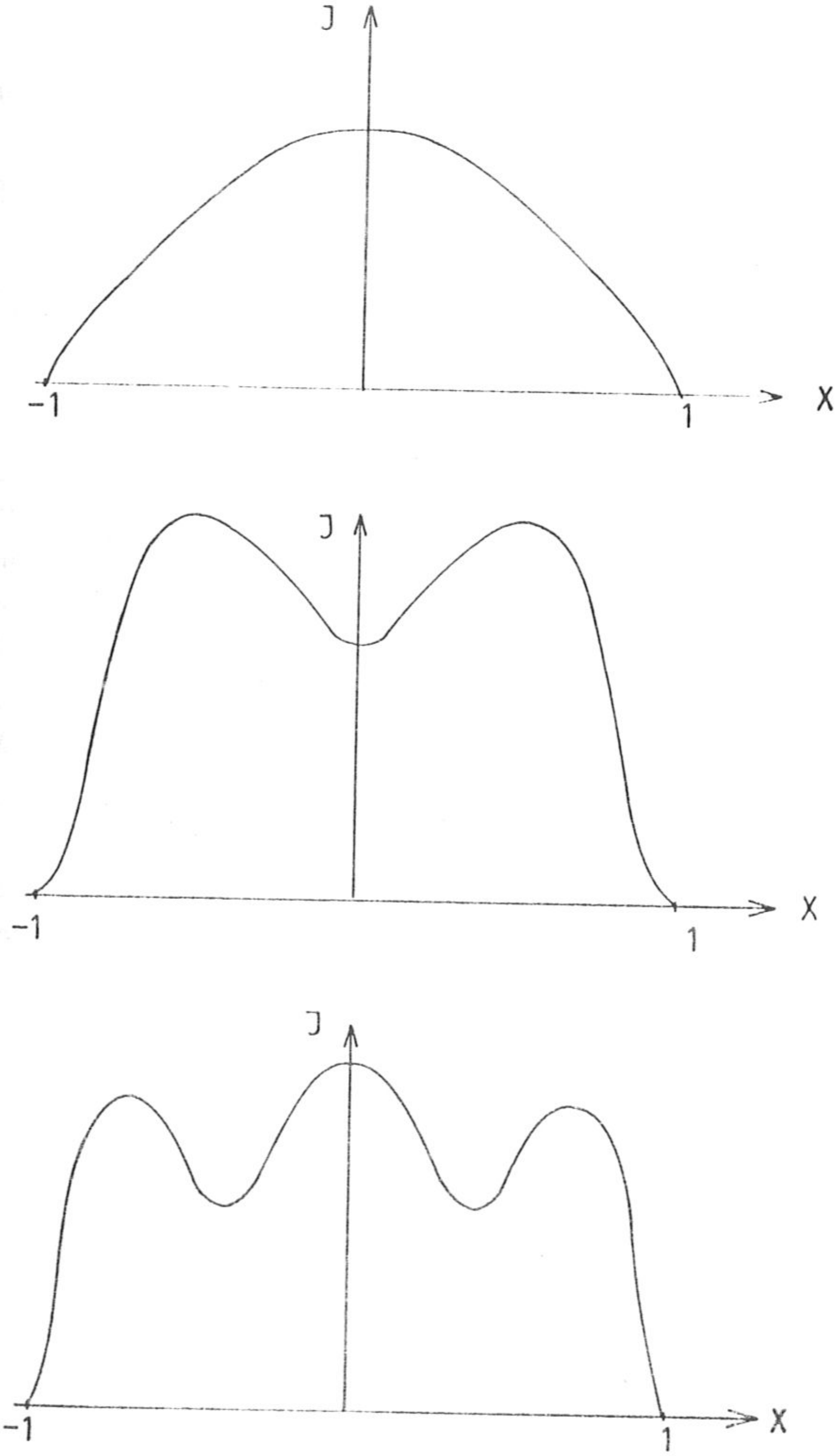


Fig. 7

positive value of $i(r)$ . We do not have an efficient means for avoiding negativity by this method, for the function (25) is nonlinear in $b$ .

We therefore consider these methods to be unsatisfactory.

## 6. Approximation using GEGENBAUER polynomials

We tried to solve (23) by minimizing $D$ using the function

$$(10) \quad J(x) = \sum_{n=0}^{k} e_n W_n^*(x),$$

without the restriction $i(r) \geq 0$ . The main properties of the $W_n^*$ are given in § 2. For $N=8$ we tried the values of $k$ from 3 to 8. There is no point in taking $k \geq N-1$ , for then $D=0$ , the approximation is too good, and for $k \geq N$ even not unique. Another reason why $k$ should be not too large is that $J(x)$ has, as expected for physical reasons, only a few extrema and inflection points; so a too high $k$ would lead to unwanted oscillations - a characteristic of polynomial approximation.

With

$$(26) \quad a_j = \begin{cases} e_j & \text{if} \quad |e_j| \leq 1 \\ -1 & \text{if} \quad e_j < -1 \\ 1 & \text{if} \quad e_j > 1, \end{cases}$$

$$(27) \quad G_p(a,b) = \int_a^b \sqrt{1-x^2}\; x^{2p}\, dx,$$

and

$$(28) \quad \delta(n,j) = \int_{a_j}^{a_{j+1}} W_n^*(x)\, dx = \sum_{p=0}^{n} \beta(p,n)\, G_p(a_j, a_{j+1})$$

(the $\beta$ are defined in § 2, formula (8"))
we have

$$(29) \qquad \int_{I_j} J(x)\,dx = \sum_{n=0}^{k} c_n\, \delta(n,j).$$

The $G_p$ may be recursively computed by the substitution $x = \sin t$ and using formula number 358 on page 321 of [3]. The results are

$$G_0(a,b) = \frac{1}{2}\left\{ \arcsin x \Big|_a^b + \left(x\sqrt{1-x^2}\right)\Big|_a^b \right\}$$

$$(30) \qquad G_p(a,b) = \frac{1}{2p+2}\left\{ \left(x^{2p-1}(1-x^2)^{3/2}\right)\Big|_b^a + (2p-1)\, G_{p-1}(a,b) \right\}$$

for $p \geq 1$.

From (23) and (29) we get

$$(31) \quad D(c) = \sum_{j=1}^{N} \left\{ T_j - \sum_{n=0}^{k} \delta(n,j)\, c_n \right\}^2 .$$

Clearly this quadratic form is positive semi-definite. The minimum of $D$ may be found by solving the system of linear equations

$$\frac{\partial D}{\partial c_i} = 0, \qquad 0 \leq i \leq k.$$

By differentiation and interchange of summation symbols we get

$$(32) \quad \sum_{n=0}^{k} Q(n,i)\, c_n = \sum_{j=1}^{N} \delta(i,j)\, T_j , \quad 0 \leq i \leq k,$$

where

$$(32') \quad Q(n,i) = \sum_{j=1}^{N} \delta(i,j)\, \delta(n,j)$$

are the elements of a symmetric matrix $Q$.

We consider the three cases

(') $k < N-1$

('') $k = N-1$ (number of coefficients $e_j$ equal to number of intervals $I_j$).

(''') $k > N-1$,

and assume $Q$ to be non-singular for $k \leq N-1$. Of course, it will depend on the subdivision $\{e_j\}$ whether this condition is satisfied. Numerical computations showed that usually, but not always, $Q$ is non-singular and therefore, because of (32'), positive definite. Of course, (32') is equivalent to

$$\sum_{n,i} Q(n,i)\, c_n c_i = \sum_j \Big( \sum_i \delta(i,j)\, c_i \Big)^2 .$$

Analytical considerations also indicate that $Q$ will usually be non-singular. Our procedure is a discrete analogue of method $D$ of [5], but with the function system $\{W_n^*\}$, which is the orthogonal system of the continuous limit case as $N \to \infty$. Another argument is that the determinant of $Q$ can be expressed explicitly in terms of elementary functions of $\alpha$. If it does not vanish identically, it can only have isolated zeros.

There are two cases where it is quite evident that $Q$ is singular.

(a) If there is an index $\lambda$ such that $e_\lambda = 0$, then $e_{\lambda-1} = -e_{\lambda+1}$, and by (27) and (28)
$G_p(e_{\lambda-1}, e_\lambda) = G_p(e_\lambda, e_{\lambda+1})$
and
$\delta(n, \lambda-1) = \delta(n, \lambda)$.

(b) If there is an index $\lambda$ such that $(e_\lambda + e_{\lambda+1})/2 = 0$, then $e_\lambda = -e_{\lambda+1}$,
and
$G_p(e_{\lambda-1}, e_\lambda) = G_p(e_{\lambda+1}, e_{\lambda+2})$, $\delta(n, \lambda-1) = \delta(n, \lambda+1)$.

In both cases the matrix $\delta$ has two equal columns. Now, if the number of equal pairs of elements $\delta(n,j)$ is $N_e$ (which is equal to the number of pairs of intervals symmetric with respect to $x = 0$), then, if $k+1 > N - N_e$ the rank of $W$ is less than $k+1$.

Using the theorem mentioned in paragraph 3 ([9], p.64-65), while computing the determinant $\det Q = \det(\delta\delta')$, we find that all the majors are zero, and therefore $\det Q = 0$.

These two cases may be formulated together by saying that when $\tilde{c} = -\alpha/h$ is equal to an integral multiple of 1/2, and when the number of pairs of intervals symmetric with respect to $x=0$ is greater than $N-(k+1)$, then $Q$ is singular.

One might think that this is a special disadvantage of approximating by a system of even orthogonal functions. But if one imagines the $T_j/h$ as a step function to be fitted by a suitably smooth, even function, one sees, that information is lost if $x=0$ is equal to an endpoint or a midpoint of an interval. The reason is that $\mathcal{J}(x)$ is supposed to be an even function, and disregarding the noise, one has less than $N$ values $T$. If the intervals $I_\lambda$ and $I_\mu$ lie symmetrically with respect to $x=0$, then $T_\lambda = T_\mu$.

Let us return to the three cases ('), (") and (''').

Assuming $Q$ to be regular, there is a unique solution in cases (') and ("). However, in case (") this solution is uninteresting, since it gives $D=0$. To show this, it is sufficient to see that if $Q$ is regular, so also is the matrix $\delta$, since $\delta$ is quadratic, and $Q = \delta\delta'$. Therefore there exists a unique solution to the system of linear equations

$$T_j - \sum_{n=0}^{k} \delta(n,j)c_n = 0, \quad j = 1, 2, \ldots, k+1$$

and by (31), $D = 0$.

In case (''') $Q$ is singular. Proof: The quadratic form

$$A = \sum_{\mu,\lambda=0}^{k} Q(\mu,\lambda)c_\mu c_\lambda$$

can be expressed as

$$A = \left(\sum_{\nu=0}^{k} \vec{\delta}_\nu c_\nu\right)^2$$

with the vectors $\vec{\delta}_i = (\delta(i,1), \delta(i,2), \ldots, \delta(i,N))$.

The system
$$\sum_{\nu=0}^{k} \vec{\delta}_\nu c_\nu = 0$$
does not have a unique solution. The $\vec{\delta}_\nu$ are $N$- dimensional vectors, and so more than $N$ of them are always linearly dependent.

A double-precision FORTRAN-routine was written for case (') incidentally also applicable in case ("). Test runs show that in actual cases either the fit is not sufficiently good (if $k$ is too small), or that $J(x)$ has large oscillations (if $k$ is too large), even into negative values of $J(x)$, which is physically meaningless. In either case $i(r)$ may be significantly negative near $r=0$. It is possible, as also shown by test runs, to reproduce a polynomial $i(r)$ of degree $k$, given in advance, from which the values $T_j$ are exactly calculated, but the method is very sensitive to superposition of noise on the $T_j$.

## 7. Reduction to a quadratic programming problem

We decided to take into account the information $i(r) \geq 0$. Then we have to find a vector
$$c' = (c_0, c_1, \ldots, c_k)$$
(the prime denotes a row-vector, $c$ denotes the corresponding column-vector), so that the quadratic form

$$(31) \qquad D(c) = \sum_{j=1}^{N} \left\{ T_j - \sum_{n=0}^{k} \delta(n,j)\, c_n \right\}^2$$

is to be minimized under the restrictions
$$i(r_\lambda) = \sum_{n=0}^{k} c_n P_n^*(r_\lambda) \geq 0, \quad \lambda = 1, 2, \ldots, m.$$

Here $k \leq N-1$, $\delta(n,j)$, $P_n^*(r_\lambda)$ are known constants, $P_n^*(r)$ is defined in § 2, formula (11). $m$, the number of restrictions, is chosen so as to give a sufficiently dense set of discrete values $r_\lambda$. After some trial runs we decided to take $k=5$ and $m=20$, $r_\lambda = (\lambda-1)/m$.

In order to transform the problem into the standard form of quadratic programming we compute

$$D(c) = \sum_{j=1}^{N} \left\{ \sum_{n=0}^{k} \delta(n,j)\, c_n \right\}^2$$
$$- 2 \sum_{n=0}^{k} \left\{ \sum_{j=1}^{N} \delta(n,j)\, T_j \right\} c_n + \sum_{j=1}^{N} T_j^2 .$$

Furthermore, if we set

(32") $$Q_{ni} = Q_{in} = Q(n,i),$$

(33) $$b_n = -2 \sum_{j=1}^{N} \delta(n,j) T_j , \quad A_{in} = -P_n^*(r_i),$$

we can state our problem in matrix form:-

(34)
By suitably choosing $c$
minimize $D(c) = c' Q c + b' c$
subject to the restriction $A c \leq 0$.

As pointed out in § 6, we assume $Q$ to be a positive definite $(k+1, \ k+1)$ matrix, ( $\tau = -\alpha/h$ is not an integral multiple of 1/2).

To solve this problem we used the iterative method of HILDRETH and D'ESOPO ([7], 73-79). This method yields a sequence $c^{(j)}$ converging to the unique solution $c$ if two conditions are fulfilled:

(.) the matrix $Q$ must be strictly positive definite,
(..) there must exist a vector $c$ such that $A_i c \leq -\varepsilon$ for $1 \leq i \leq m$ and some $\varepsilon > 0$. Here $A_i = (A_{i0}, A_{i1}, \dots, A_{ik})$.

(.) is assumed to be satisfied if $-\alpha/h \neq j/2$ for all integers $j$. (..) is satisfied if $c' = (-1, 0, 0, \dots, 0)$. For then $A_i c = -\sum_{n=0}^{k} c_n P_n^*(r_i) = -P_0^*(r_i) = -1 < 0$.

As a stopping criterion for the double-precision FORTRAN-program we used:- "Stop iterating if

$$|c_i^{(p+1)} - c_i^{(p)}| / |c_i^{(p)}| < 10^{-4} \quad \text{and} \quad c_i^{(p)} \neq 0$$

for five values $p$ in succession or after at most $L$ iterations with $L = 200$."

As a measure of the quality of approximation the quantity

$$d_{rel} = \frac{\sqrt{D}}{\sum_{j=1}^{N} T_j}$$

was computed.

## 8. Numerical experience

We give statistics for the computing times of several runs. The program was organized so that in one computer run an arbitrary number of functions $i(r)$ could be computed, after reading the corresponding sets of data $T_j$

| Number of functions $i(r)$ computed | running time in minutes | average running time for one function in minutes |
|---|---|---|
| 56 | 43 | 0.77 |
| 86 | 60 | 0.70 |
| 59 | 63 | 1.07 |
| 66 | 56 | 0.85 |
| 56 | 70 | 1.27 |
| 123 | 120 | 0.98 |
| 446 | 412 | 0.92 |

The average number of iterations is another important statistic. For the 56 functions computed in 43 minutes the total number of iterations was 1754, and we have

$1754/56 \approx 31.4$ as an average number of iterations.

The quality of the approximation is illustrated by some sketches *), in which are plotted the $T_j/h$ and the functions $J(x)$ and $i(r)$. If the measurements were accurate, in the intervals $(e_j, e_{j+1})$ the curve $y = J(x)$ should intersect the horizontals $y = T_j/h$. Of course, this could also be achieved with inaccurate $T_j$ by choosing $h$ sufficiently high, but then $J(x)$ and $i(r)$ would be physically meaningless, strongly oscillating functions. In case of measurement errors it is not wise to attempt too close an approximation. $d_{rel}$ was usually about $10^{-2}$.

On the other hand, it must be admitted that polynomial, least-squares approximations of sufficiently low degree yield functions which do not have too many extrema and inflection points, but at the same time are not quite satisfactory. This happened especially often in cases where $J(x)$ had a local minimum at $x = 0$. Sometimes $J(x)$ had a local maximum very near to $x = 1$, which was considered to be physically meaningless. The reason for it lies in the wild behaviour of the functions $W_n^*(x)$ at $x \approx 1$, which is illustrated in ([5], p. 19)

This computational effect was considered tolerable, because the more interesting region is near $x = 0$. Further details are mentioned in the comments under the sketches (see appendix).

We tried also to reconstruct an analytically given function $i(r)$ from the values $T_j$ computed for a given subdivision $(e_j, e_{j+1})$, $j = 1, 2, \ldots, N$ of $-1 \leq x \leq 1$. We chose an even function $i(r)$, which has a deep local minimum at $r = 0$, and one maximum between 0 and 1.

---

*) see appendix, p. 51

We took

$$i(r) = \frac{1}{2}\,(1-r^2)(r^2+\beta),$$

which is $\geq 0$ in $0 \leq r \leq 1$, if $\beta \geq 0$.

If $0 \leq \beta < 1$, then $i(r)$ has a maximal value $i_{max} = \frac{1}{8}\,(1+\beta)^2$ at $r = r_{max} = \sqrt{(1-\beta)/2}$, and by varying $\beta$ one can alter the ratio

$$\frac{i_{max}}{i(0)} = \frac{\frac{1}{8}\,(1+\beta)^2}{\beta/2}.$$

We choose $\beta = 0.1$, which yields $i(0) = 0.05$,

$$i_{max} = 0.15125,\quad r_{max} = \sqrt{0.45} \approx 0.67,\quad \frac{i_{max}}{i(0)} = 3.025.$$

Expanding $i(r) = \frac{1}{2}\sum c_j P_j^*(r)$ yields $c_j = 0$ for $j \geq 3$,

$c_0 \approx 0.22095$, $c_1 \approx -0.002963$, $c_2 \approx -0.013545$,

from which $J(x)$ and the $T_j$ can be computed.
The $T_j$ were computed for $\alpha = -0.827$, $h = 0.23$, $N = 7$.

The lists, in appendix p. 50 , give the exactly computed values of $J(x)$ and $i(r)$ (computed from the given coefficients $c_j$ ), and then the values of $J(x)$ and $i(r)$, computed from the values $T_j$. Subsequently the influence of inaccurate estimated $\alpha$, and then the influence of disturbed $T_j$ are investigated. The values of $T_j$ are disturbed by adding normally distributed random numbers with mean 0 and standard deviation equal to $\frac{1}{10} \max\limits_{j=1}^{N} T_j$.

The authors would like to express the opinion, that studies in curve-fitting according to ideas proposed by HOOKE and JEEVES ( [6], 222-224) should be undertaken. That is, one should have an alternative to polynomial approximation (or more generally: to approximation by special systems of smooth orthogonal functions) and instead, fit to given values $\overline{y}_n$ (which are supposed to belong to a sufficiently dense set of equidistant $x_n$)

a discrete set of points $(x_n, y_n)$. In determining $y_n$, the number of maxima, minima and inflection points of the smooth curve $y = y(x)$, which we suppose to represent the proper physical process, should be used, in the form of linear difference inequality restrictions on the convexity and concavity of the fitted function. Under these restrictions a convenient deviation function $D(y_1, \ldots, y_N; \overline{Y}_1, \ldots, \overline{Y}_N; x_1^*, \ldots, x_M^*)$ is to be minimized, where the $x_j^*$ are the special abscissae (of inflection points and extrema), and the $y_j$ and $x_\lambda^*$ are to be determined.

By this method it should be possible to avoid the previously mentioned disadvantages of approximation by a finite sum of special functions.

## 9. The case of an arbitrary radius $R$

Up to now we assumed the radius of the configuration to be equal to 1. For convenience we shall now give the essential transformation formulae for the case of an arbitrary, positive radius $R$. For this purpose let us assume that for the case $R = 1$ we use the variables $\xi$ and $\varrho$, $-1 \leq \xi \leq 1$, $0 \leq \varrho \leq 1$, and the intensities $J(\xi)$ and $i(\varrho)$.

Then

$$(35) \qquad T_j \approx \int_{e_j}^{e_{j+1}} J(\xi)\,d\xi \approx J(\overline{\xi}_j)\,h,$$

$$(36) \qquad J(\xi) = 2 \int_{\varrho=|\xi|}^{1} \frac{i(\varrho)\,\varrho\,d\varrho}{\sqrt{\varrho^2 - \xi^2}}.$$

For the case of an arbitrary $R > 0$ we use the variables $x$ and $r$, $-R \leq x \leq R$, $0 \leq r \leq R$, $J^*(x)$, $i^*(r)$.

a discrete set of points $(x_n, y_n)$. In determining $y_n$, the number of maxima, minima and inflection points of the smooth curve $y = y(x)$, which we suppose to represent the proper physical process, should be used, in the form of linear difference inequality restrictions on the convexity and concavity of the fitted function. Under these restrictions a convenient deviation function $D(y_1, \ldots, y_N; \overline{Y_1}, \ldots, \overline{Y_N}; x_1^*, \ldots, x_M^*)$ is to be minimized, where the $x_j^*$ are the special abscissae (of inflection points and extrema), and the $y_j$ and $x_\lambda^*$ are to be determined.

By this method it should be possible to avoid the previously mentioned disadvantages of approximation by a finite sum of special functions.

## 9. The case of an arbitrary radius $R$

Up to now we assumed the radius of the configuration to be equal to 1. For convenience we shall now give the essential transformation formulae for the case of an arbitrary, positive radius $R$. For this purpose let us assume that for the case $R = 1$ we use the variables $\xi$ and $\varrho$, $-1 \leq \xi \leq 1$, $0 \leq \varrho \leq 1$, and the intensities $J(\xi)$ and $i(\varrho)$.

Then

(35) $$T_j \approx \int_{e_j}^{e_{j+1}} J(\xi)\,d\xi \approx J(\overline{\xi}_j)\,h,$$

(36) $$J(\xi) = 2\int_{\varrho=|\xi|}^{1} \frac{i(\varrho)\,\varrho\,d\varrho}{\sqrt{\varrho^2 - \xi^2}}.$$

For the case of an arbitrary $R > 0$ we use the variables $x$ and $r$, $-R \leq x \leq R$, $0 \leq r \leq R$, $J^*(x)$, $i^*(r)$.

Then

$$(35') \quad T_j \approx \int_{e_j^*}^{e_{j+1}^*} J^*(x)\, dx \approx J^*(\overline{x_j})\, h^*,$$

$$(36') \quad J^*(x) = 2 \int_{r=|x|}^{R} \frac{i^*(r)\, r\, dr}{\sqrt{r^2 - x^2}} .$$

(The meaning of $e_j^*$, $\overline{\xi_j}$, $\overline{x_j}$, $h^*$ is obvious).

Then $\xi = x/R, \quad \rho = r/R, \quad h = h^*/R .$

By comparing (35) and (35') we obtain

$$(37) \quad J^*(x) = R^{-1}\, J(\xi) .$$

From (36) we obtain, by changing the variables,

$$J(\xi) = 2 \int_{r=|x|}^{R} \frac{i(r/R)\,(r/R)\,(dr)/R}{R^{-1}\sqrt{r^2 - x^2}} = 2 \int_{r=|x|}^{R} \frac{R^{-1}\, i(r/R)\, r\, dr}{\sqrt{r^2 - x^2}}$$

which, by (37), $= R\, J^*(x)$.

Therefore (36') yields

$$(38) \quad i^*(r) = \frac{i(\rho)}{R^2} = \frac{i(r/R)}{R^2} .$$

Acknowledgements

The authors are indebted to a number of colleagues at the Institut für Plasmaphysik for helpful discussions, especially to Dr. A. Eberhagen, the evaluation of whose experimental results gave rise to the above work, and to F.M. Larkin from Culham Laboratory, who assisted us in styling the English text of this report.

## C. References

[1] W.L. BARR, Method for computing the radial distribution of emitters in a cylindrical source. Journal of the Optical Society of America 52, 885-888 (1962).

[2] R.N. BRACEWELL, Strip integration in radio astronomy. Australian J. Phys. 9 (1956), 198-217.

[3] I.N. BRONSTEIN / K.A. SEMENDJAJEW, Taschenbuch der Mathematik, Leipzig 1960.

[4] G. DOETSCH, Einführung in Theorie und Anwendung der LAPLACE-Transformation. Birkhäuser-Verlag Basel / Stuttgart 1958.

[5] R. GORENFLO, Numerische Methoden zur Lösung einer ABELschen Integralgleichung. IPP/6/19. Institut für Plasmaphysik, 8046 Garching bei München (Germany), Mai 1964.

[6] R. HOOKE / T.A. JEEVES, "Direct search" solution of numerical and statistical problems. Journal ACM 8 (1961), 212-229.

[7] H.P. KÜNZI / W. KRELLE, Nichtlineare Programmierung. Springer-Verlag, Berlin / Göttingen / Heidelberg 1962.

[8] F.G. TRICOMI, Vorlesungen über Orthogonalreihen. Springer-Verlag Berlin / Göttingen / Heidelberg 1955.

[9] F.E. HOHN, Elementary Matrix Algebra. Macmillan Company, New York 1964.

## D. Appendix

### a) Description of the programs

We present here the listings of the two FORTRAN programs which implement the above methods. In order to use the programs they may have to be adjusted to suit individual requirements.

A short description of the programs is given and, together with the comment cards in the programs themselves, this should be enough to indicate how any necessary changes may be effected.

Because of limited programming time, program efficiency and elegance have, in places, been disregarded in the interests of simplicity. The programs are given as they have been run, together with an example of a printed output (of program 1) and some curves obtained from the output of program 2.

The programs consist of the following routines:

| | program 1 | program 2 |
|---|---|---|
| standard type | INVERS | INVERS |
| | LGLSY | LGLSY |
| | BKOEFF | BKOEFF |
| | BETAKO | BETAKO |
| | GAMMAK | GAMMAK |
| similar type | MATCAL | MATCAL |
| | QUADRA | QUADRA |
| special type | MAIN routine | MAIN routine |
| | COMPUT | INOUT |
| | | COMPUT |
| | | GINDE |
| | | DEFIN |

Subroutines INVERS and LGLSY in the two programs differ in COMMON statements only.

MATCAL and QUADRA have the same function in both programs, but are a little different in each of them.

The routines BKOEFF, BETAKO, GAMMAK, which compute the coefficients $b(p,n)$, $\beta(p,n)$, $\gamma(p,n)$ are given in [5].

Description of program 1 - (continuous case)

The program uses a quadratic-programming method to approximate a continuous function with Gegenbauer polynomials of degrees from 4 to 7 successively.

In order to run the program it is necessary to give the following data deck:

| Items on card | Format | explanation |
|---|---|---|
| NT, RGROSS | I10,E14.4 | NT = Number of points including 0.<br>RGROSS = scaling factor |
| $T_1, \ldots, T_6$<br>$T_7, \ldots, T_{NT}$ | 6E12.4 | $T_i$ = The measured values, $i=1, \ldots, NT$, of $J(x)$. |

These data can be repeated as often as new measurements have been made.

Description of program 2 - (discrete case)

This program solves the above problem with polynomials of degree 5 only.

The results are written on tape as well as on off-line printer. The results on tape can then be used for further computations, or for graphical output. Channel 3 is used in this program.

The input data is divided into three classes:

1) Data for one machine run.
2) Data for a whole set of measurements, which consists of single measurements at given times.
   (A measurement consists of N measured values which are to be fitted by one curve)
3) Data for a single measurement.

Class 1 consists of a single card, class 2 consists of a single card which is repeated every time a new set is started , and class 2 consists of 2 cards for every measurement. (see data sheet).

The program assumes that for each measurement there are 8 measured points, the first of which can be excluded in case it was wrongly measured. In that case, NST should be 7. (see data sheet). This limitation to 8 points can easily be removed by changing only the READ instructions in subroutine INOUT. In any case, NST cannot exceed the value of 20, because of DIMENSION statements. The program checks then if one of the intervals lies outside the segment $\langle -1, 1 \rangle$, and decides how many points will enter the computation (N8 - see glossary).

The comment cards in the various routines, in addition to the data sheet and the following glossary should be sufficient to explain the function and the use of each of them.

## DATA - sheet

| class | Items on card | Format | Explanation and comments |
|---|---|---|---|
| 1 | MISPAR | I 12 | Mispar = running number of set of measurements on tape. It is equal to the number of sets which are already on tape. |
| 2 | NOB, HROH, DELAMB, EK, WELL, NST, NZAHL | I4,4F8.3, 2I3 | NOB = Identification number of set. HROH = Original length of interval. DELAMB, EK, WELL = data irrelevant to this program, but which may be needed for further computations. NST = number of measured points. Usually NST = 8. If the first point was wrongly measured, NST = 7. |
| 3 | 1)NOB,WALZT, $T_1, T_2, T_3, T_4$, 2)NOB, $T_5, \ldots, T_8$, FC | I4,5E12.4 | WALZT = time $T_i$ = the measured values $(i = 1,2,\ldots 8)$ FC = centre = $\tilde{c}$. |

## GLOSSARY

| Routine | FORTRAN Variable | Term in theoretical part | Explanation and comments |
|---|---|---|---|
| MAIN | N8 | N | Number of measured values which are to enter computation (see NST in data sheet!) |
| | AL(I) | $e_i$ | end points of computed intervals |
| | DELTA(J,N) | $\delta(n,j)$ | $\delta_{n,j} = \sum_{\rho=0}^{n} \beta(\rho,n) \int_{I_j} x^{2\rho} \sqrt{1-x^2}\, dx$ |
| | R(I) | $r_\lambda$ | points in which the restrictions are computed $(r_\lambda = \lambda/m,\ \lambda = 0, 1, \ldots, m-1)$ |
| | P(I,N) | $A_{i,n}$ | coefficients of restrictions |
| | E(I) | $b_i$ | coefficients of the linear part of D |
| | QREC | Q | The positive definite matrix |
| | Q | $Q^{-1}$ | |
| | V,G,U,W | | Matrices and vectors needed for solving the quadratic-programming problem (see [7]) |
| | NIT | | number of iterations, not exceeding 200 |
| | D,DREL | $D, D_{rel}$ | Absolute and relative deviation function |

| Routine | FORTRAN Variable | Term in theoretical part | Explanation and comments |
|---|---|---|---|
| INOUT | RGROSS | R | Radius of Vessel in mm. This is a constant in the program. |
| COMPUT | F,RINT | J(x),i(r) | |

## b) Lists of the Programs

Program 1

```
                 MAIN PROGRAM TO QUADRATIC PROGRAMMING
C                         CONTINUOUS CASE
D        DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D        DIMENSION QREC(11,11),Q(11,11)
D         DIMENSION RSL(11),FMATEL(11,11)
D        DIMENSION P(20,20),E(20),T(40),X(40)
D        DIMENSION WE(40,11)
         COMMON B,BETA,NK,GAMMA,K
         COMMON NT
         COMMON QREC,Q
         COMMON RSL,FMATEL
         COMMON P,E,T,X
         COMMON WE,RGROSS
D        DIMENSION CM(11,11) ,C(11)
C        I N P U T.
       3 READ INPUT TAPE 12,100,NT,RGROSS
         READ INPUT TAPE 12,101,(T(J),J=1,NT)
         DO 1 J=1,NT
D      1    X(J)=FLOATF(J-1)/FLOATF(NT-1)

C        O U T P U T   FOR INPUT.

         WRITE OUTPUT TAPE 3,102,NT,RGROSS

            NK=11

D        CALL BKOEFF
D        CALL BETAKO
D        CALL GAMMAK

            K = 5

D      4 CALL MATCAL
         WRITE OUTPUT TAPE 3,160
D           CALL INVERS
         DO 93 I=1,K
         DO 93 J=1,K
D           CM(I,J)=0.
         DO 93 L=1,K
D     93    CM(I,J)=CM(I,J)+QREC(I,L)*Q(L,J)
         DO 33 I=1,K
      33 WRITE OUTPUT TAPE 3,104,(CM(I,J),J=1,K)
D        CALL QUADRA(C)
         CALL COMPUT(C)
         IF(K-7)2,2,3
       2    K=K+1
         GO TO 4
     100 FORMAT(I10,E14.4)
     101 FORMAT(6E12.4)
     102 FORMAT(1H1,10X,4HNT =I4/1H ,5X,HRGROSS =F6.2)
     104 FORMAT(1H ,6E2 .4)
     160 FORMAT(28HLCHECKING THE INVERSL MATRIX//)
         END(1,0,0,0,0,0,1,0,0,1,0,0,0,.,.)
```

SUBROUTINE MATCAL

```
      SUBROUTINE MATCAL
C                      COMPUTES MATRICES NECESSARY TO SOLVE QUADRATIC-PROGR.PROBLEM
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION QREC(11,11),Q(11,11)
D      DIMENSION RSL(11),FMATEL(11,11)
D     DIMENSION P(20,20),E(20),T(40),X(40)
D     DIMENSION WE(40,11)
      COMMON B,BETA,NK,GAMMA,K
      COMMON NT
      COMMON QREC,Q
      COMMON RSL,FMATEL
      COMMON P,E,T,X
      COMMON WE
D     DIMENSION R(20)
C     CALCULTATION OF WE.
      DO 3 J=1,NT
      DO 3 N=1,K
D        WE(J,N)=0.
      DO 2 L=1,N
D   2    WE(J,N)=WE(J,N)+BETA(L,N)*X(J)**(2*L-2)
D   3    WE(J,N)=WE(J,N)*SQRTF(1.-X(J)**2)
      DO 4 I=1,20
D        WW=I -1
C     CALCULATION OF MATRIXP-RESTRICIIONS.
D   4    R(I)=WW*0.05
      DO 5 I=1,20
      DO 5 N=1,K
D        P(I,N)=0.
      DO 5 L=1,N
D   5    P(I,N)=P(I,N)-GAMMA(L,N)*R(I)**(2* L-2)
C     CALCULATION OF VECTOR E.
      DO 7 N=1,K
D        E(N)=0.
      DO 6 J=1,NT
D   6    E(N)=E(N)-WE(J,N)*T(J)
D   7   E(N)=2.*E(N)
C     CALCULTAION OF MATRIX QREC.
      DO 8 M=1,K
      DO 8 L=1,K
D        QREC(M,L)=0.
      DO 8 J=1,NT
D   8    QREC(M,L)=QREC(M,L)+WE(J,M)*WE(J,L)
      RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

SUBROUTINE COMPUT(C)

```
      SUBROUTINE COMPUT(C)
C                       COMPUTES AND PRINTS RESULTING FUNCTION AND INTENSITIES
      DIMENSION S(40),RE(40),RE2(40)
D     DIMENSION WE(40,11)
D     DIMENSION V(20),PP(11),F(40),RINT(40)
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION QREC(11,11),Q(11,11)
D      DIMENSION RSL(11),FMATEL(11,11)
D     DIMENSION P(20,20),E(20),T(40),X(40)
      COMMON B,BETA,NK,GAMMA,K
      COMMON NT
      COMMON QREC,Q
      COMMON RSL,FMATEL
      COMMON P,E,T,X
      COMMON WE,RGROSS
D     DIMENSION C(11)
      WRITE OUTPUT TAPE 3,100,K
      DO 52 J = 1,NT
         S(J) = 0.
      DO 52 N = 1,K
   52    S(J) = S(J) + C(N) * WE(J,N)
         D = 0.
      DO 54 J = 1,NT
   54    D = D + (T(J) - S(J))**2
      WRITE OUTPUT TAPE 3,700,D
  700 FORMAT( 6H   D   =E12.5)
      WRITE OUTPUT TAPE 3,102
      DO 5 I = 1,NT
         J = I - 1
      DO 2 N = 1,K
         PP(N) = 0.
      DO 2 L = 1,N
         EX = X(I)**(2*L - 2)
    2    PP(N) = PP(N) + GAMMA(L,N) * EX
         F(I) = 0.
         RINT(I) = 0.
      DO 4 N = 1,K
         F(I) = F(I) + C(N) * WE(I,N)
    4    RINT(I) = RINT(I) + C(N) * PP(N)
         RINT(I) = RINT(I) /(2.*RGROSS)
         RE(I) = F(I) - T(I)
      WRITE OUTPUT TAPE 3,101, J,X(I),T(I),F(I),RINT(I),RE(I)
    5 CONTINUE
      RETURN
  100 FORMAT(1H1,10X,16HVALUES FOR  K  =I3/////)
  101 FORMAT(1H I4,F8.2,14X,5F16.4)
  102 FORMAT(1H ,7X6HX OR R,23X4HT(X),10X9HCAPITAL I,9X7HSMALL I,9X10HDI
     1FFERENCE///)
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

Program 2

```
                     SOLVES THE APPROXIMATION PROBLEM BY CONVERTING IT
C                    INTO A QUADRATIC-PROGRAMING PROBLEM
C                    DISCRETE CASE.
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION T(20),ALPHA(1),H(1)
D     DIMENSION QREC(11,11),Q(11,11)
D     DIMENSION RSL(11),FMATEL(11,11),DELTG(1)
      COMMON B,BETA,NK,GAMMA,K
      COMMON T,ALPHA,N8
      COMMON QREC,Q
      COMMON RSL,FMATEL,DELTG
      COMMON H,WALZT,NZAHL
      COMMON MISPAR,RGROSS,FC,NINT
D     DIMENSION P(20,20),E(11),DELTA(11,11)
      COMMON P,E,DELTA
D     DIMENSION CM(11,11),C(11),AL(21),ALI(21)
      COMMON AL



      READ INPUT TAPE 12,5111,MISPAR

      REWIND 13
         NK = 11

D     CALL BKOEFF
D     CALL BETAKO
D     CALL GAMMAK

      IF(MISPAR - 1) 8,509,509
  509 DO 510 I = 1,MISPAR
  510 CALL SKPFIL(13)
C                    READ AND WRITE DATA FOR A SET OF MEASUREMENTS.
    8 CALL INOUT(1)
         NL = 0
C                    READ AND WRITE DATA FOR SINGLE MEASUREMENTS.
   10 CALL INOUT(2)
         MI = 0
         NL = NL + 1
C                    TRANSFORM THE ORIGINAL INTERVAL INTO ONES INCLUDED
C                    IN (-1,1)
D  59 IF(ABSF(ALPHA)-1.) 1,1,2
D   1 AL(1) = ALPHA
      GO TO 5
D   2 IF(ALPHA) 4,1,3
D   3 AL(1) = 1.
      GO TO 5
D   4 AL(1) = -1.
    5 DO 15 L = 1,N8
      J = L+1
D     ALI(J) = FLOATF(L)*H + ALPHA
D     IF(ABSF(ALI(J)) - 1.) 11,11,12
D  11 AL(J) = ALI(J)
      GO TO 15
D  12 IF(ALI(J)) 14,11,13
D  13 AL(J) = 1.
```

SOLVES THE APPROXIMATION PROBLEM BY CONVERTING IT

```
      GO TO 15
D  14 AL(J) = -1.
   15 CONTINUE
         NNN  = N8+1
      WRITE OUTPUT TAPE 3,150,(AL(J),J=1,NNN)

       K  = 6

C                    COMPUTE MATRICES AND VECTORS
D      CALL MATCAL

      WRITE OUTPUT TAPE 3,138,DETLG
C                    CHECK QREC FOR SINGUALRITY.
        DO 93  I  = 1,K
         DO 93  J  = 1,K
D     CM(I,J) = 0.
      DO 93  L  = 1,K
D  93 CM(I,J) = CM(I,J) + QREC(I,L)*Q(L,J)

C                    IS LOG OF DETERMINANT TOO SMALL.
      IF(DETLG + 9.)87,87,88
C                    IT IS.CHANGE THE CENTRE.
   87    MI  = MI  + 1
D        FMI  = MI
             FC  = FC   + (-1.)**MI*FMI*0.05
      CALL INOUT(3)
      GO TO 59

C                    PRINT THE UNIT MATRIX QREC*QREC**(-1)
   88 DO 33 I  = 1,K
   33 WRITE OUTPUT TAPE 3,161,(CM(I,J),J=1,K)

C                    SOLVE THE QUDRATIC-PROGRAMING PROBLEM
D     CALL QUADRA(C)
C                    COMPUTE FUNCTION VALUES AND INTENSITIES
       CALL COMPUT(C)
  402 IF(NL - NZAHL)10,7,7
    7 ENDFILE 13
       GO TO 8
 1000 FORMAT(1HJ,3(E20.5))
  139 FORMAT(1HJ,(8(E12.4,4X)))
  190 FORMAT(I6,F4.1)
  150 FORMAT(18HJTHE INTERVALS ARE/1HJ,9E13.4///)
  160 FORMAT(28HLCHECKING THE INVERSE MATRIX//)
  161 FORMAT(1H 6E20.5)
  170 FORMAT(12HL0 U T P U T//)
  137 FORMAT(32H SINGULAR MATRIX, NO COMPUTATION///////)
  138 FORMAT(21H LOG10 OF DETERMINANT E12.3)
 5111 FORMAT(I12)
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

```
      SUBROUTINE INOUT(I)

      SUBROUTINE INOUT(I)
C                    READS AND WRITES DATA
C                    DEFINES ALPHA AND N8.
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION T(20),ALPHA(1),H(1)
D     DIMENSION QREC(11,11),Q(11,11)
D     DIMENSION RSL(11),FMATEL(11,11),DELTG(1)
      COMMON B,BETA,NK,GAMMA,K
      COMMON T,ALPHA,N8
      COMMON QREC,Q
      COMMON RSL,FMATEL,DELTG
      COMMON H,WALZT,NZAHL
      COMMON MISPAR,RGROSS,FC,NINT
      GO TO (7,8,20 ),I
    7 READ INPUT TAPE 12,103,NOB,HROH,DELAMB,EK,WELL,NST,NZAHL
       WRITE OUTPUT TAPE 3,100
      WRITE OUTPUT TAPE 3,1000,NOB,DELAMB,EK,WELL,NZAHL,NST
         NINT = 20
        RGROSS = 23.
   10    MISPAR=MISPAR+1
      WRITE OUTPUT TAPE 3,106,MISPAR
  106 FORMAT(1HL,I5,16H SCHUSSE ON TAPE/)
      WRITE TAPE 13,NOB,WELL,NZAHL,NINT
       GO TO 25
    8 IF(NST-8) 45,46,46
   45 READ INPUT TAPE 12,105,NOB,WALZT,DUMMY,(T(I),I=1,3),NOB,(T(I),I=
     14,7),FC
      GO TO 19
   46 READ INPUT TAPE 12,105,NOB,WALZT,(T(I),I=1,4),NOB,(T(I),I=5,8),FC
   19 IF(HROH - 4.5) 11,11,12
   11 IF(FC - 4.) 13,13,14
   13    FNST = NST
         RGROSS = (FNST - FC) * HROH
      GO TO 12
   14      RGROSS = FC * HROH
   12    H = HROH/RGROSS
      WRITE OUTPUT TAPE 3,102,HROH,RGROSS,H
   20    ALPHA = -FC*H
D        ALPHA = ALPHA
      WRITE OUTPUT TAPE 3,104,FC,ALPHA
         ST = NST
         E1 = ALPHA + (ST - 0.5) * H
      IF(E1 - 1.) 22,22,23
   22     N8 = NST
      GO TO 24
   23    N8 = NST - 1
   24 WRITE OUTPUT TAPE 3,130,WALZT,N8
      WRITE OUTPUT TAPE 3,82
      WRITE OUTPUT TAPE 3,139,(T(L),L=1,N8)
   25 RETURN
  100 FORMAT(19H1I N P U T  D A T A/)
  102 FORMAT(1H1,20X,6HHROH =E12.4/1H ,18X,8HRGROSS =E12.4/1H 23X,3HH =E
     112.4)
  105 FORMAT(I4,5E12.4)
  103 FORMAT(I4,4F8.3,2I3)
  139 FORMAT(1HJ,(8(E12.4,4X)))
```

```
      SUBROUTINE INOUT(I)

  82 FORMAT(1HJ,24H THE MEASURED VALUES ARE)
 104 FORMAT(1H ,22X,4HFC =E12.4/1H ,19X,7HALPHA =E12.4///)
 130 FORMAT(1H ,30X,5HTIME  E12.4,3X,9HMICROSEC.,10X,I2,8H  LIGHTS/)
1000 FORMAT(1HJ,20X,10HSCHUSS NR.I5//1H ,18X,8HDELAMB = E12.4/1H ,22X,
    14HEK =E12.4/1H ,20X,6HWELL = E12.4//1H ,22X,I4,11H ZEITPUNKTE/
    21H ,22X,I4,16H ORIGINAL LIGHTS//)
     END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

```
      SUBROUTINE COMPUT (C)

      SUBROUTINE COMPUT (C)
C                    COMPUTES FUNCTION VALUES AND INTENSITIES
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION T(20),ALPHA(1),H(1)
D     DIMENSION QREC(11,11),Q(11,11)
D     DIMENSION RSL(11),FMATEL(11,11),DELTG(1)
      COMMON B,BETA,NK,GAMMA,K
      COMMON T,ALPHA,N8
      COMMON QREC,Q
      COMMON RSL,FMATEL,DELTG
      COMMON H,WALZT,NZAHL
      COMMON MISPAR,RGROSS,FC,NINT
       DIMENSION AL(21)
D     DIMENSION P(11),C(11),V(11),X(101),F(101),RINT(101),XX(21)
         FNINT = NINT
        R = RGROSS / 10.
        NNN = N8 + 1
        HROH = H * R
C                    TRANSFORM BACK TO ORIGINAL INTERVALS
        AL(1) = ALPHA * R
      DO 6 I = 2,NNN
    6   AL(I) = AL(I-1) + HROH
      WRITE OUTPUT TAPE 3,102
      WRITE OUTPUT TAPE 3,103,(AL(I),I=1,NNN)
      DO 5 I = 1,NINT
D     X(I) = FLOATF(I-1)/FNINT
D       XX(I) = X(I) *R
      DO 2 N = 1,K
D     V(N) = 0.
D   1 P(N) = 0.
      DO 2 L = 1,N
D     V(N) = V(N) + BETA(L,N)* X(I)**(2*L-2)
D   2 P(N) = P(N) + GAMMA(L,N) * X(I)**(2*L-2)
D     F(I) = 0.
D   3 RINT(I) = 0.
      DO 4 N = 1,K
D       F(I) = F(I) + V(N) * C(N)
D   4    RINT(I) = RINT(I) + C(N) * P(N)
C                    TRANSFORM BACK TO ORIGINAL VALUES OF FUNCTION AND
C                    INTENSITIES
D       RINT(I) = RINT(I)* H /(2.*R)
D       F(I) = F(I) * SQRTF(1. - X(I) ** 2) * H
    5 CONTINUE
      WRITE OUTPUT TAPE 3,100
      WRITE OUTPUT TAPE 3,101,(XX(I),F(I),RINT(I),I = 1,NINT)
      WRITE TAPE 13,WALZT, N8,    (XX(I),RINT(I),I=1,NINT)
      WRITE TAPE 13,(AL(I),I=1,NNN),(T(I),I=1,N8)
      WRITE TAPE 13,(F(I),I = 1,NINT)
      RETURN
  100 FORMAT(1HJ 2H X,16X,1HF,18X,4HRINT//)
  101 FORMAT(1H ,F4.2,E25.8,E25.8)
  102 FORMAT(18HJTHE INTERVALS ARE/)
  103 FORMAT(1H ,9E14.4/1H ,8E15.4)
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

SUBROUTINE MATCAL

```
      SUBROUTINE MATCAL
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION T(20),ALPHA(1),H(1)
D     DIMENSION QREC(11,11),Q(11,11)
D     DIMENSION RSL(11),FMATEL(11,11),DELTG(1)
      COMMON B,BETA,NK,GAMMA,K
      COMMON T,ALPHA,N8
      COMMON QREC,Q
      COMMON RSL,FMATEL,DELTG
      COMMON H,WALZT,NZAHL
      COMMON MISPAR,RGROSS,FC,NINT
D     DIMENSION P(20,20),E(11),DELTA(11,11)
      COMMON P,E,DELTA
D       DIMENSION AL(21)
      COMMON AL
D     DIMENSION R(20)
      DO 31 J=1,N8
      DO 31 N = 1,K
D     DELTA(J,N) = 0.
      DO 31 L = 1,N
      J = J
D     F = AL(J)
D       B = AL(J+1)
       L = L
D  31 DELTA(J,N) = DELTA(J,N) + BETA(L,N)*DEFIN(L,F,B)


   60 DO 61 I=1,20
D     WW = I-1
D     RW = 0.05
D  61  R(I) = WW*RW
      DO 63 I=1,20
      DO 63 N=1,K
D  62 P(I,N)=0.
      DO 63 L=1,N
D  63 P(I,N) = P(I,N) - GAMMA(L,N)*R(I)**(2*L-2)
      DO 65 N=1,K
D  64 E(N) = 0.
      DO 65 J=1,N8
D  65 E(N)=E(N)-(DELTA(J,N)*T(J))*2.

      DO 94 M=1,K
      DO 94 L = 1,K
D     QREC(M,L) = 0.
      DO 94 J = 1,N8
D  94 QREC(M,L) = QREC(M,L) + DELTA(J,M) * DELTA(J,L)

C                     CALCULATE QREC**-1.
D     CALL INVERS

      RETURN
      END(1, , , , , ,1, , ,1, , , , , )
```

FUNCTION GINDE(L,X)

```
      FUNCTION GINDE(L,X)
C                       COMPUTES THE INDEFINITE INTEGRALS
D     DIMENSION G(11)
D        Z = 1. - X**2
      IF(Z) 2,1,2
D   2    Y = X / SQRTF(Z)
      IF DIVIDE CHECK 1,6
    6 IF QUOTIENT OVERFLOW 1 , 3
    1 IF(X) 7,7,10
D   7    Y = -1.E38
       GO TO 3
D  10    Y = 1. E 38
D   3 G(1) = 0.5*(ATANF(Y)+X*SQRTF(Z))
      IF(L-1)4,4,5
D   4 GINDE=G(1)
      GO TO9
    5 DO 8 I=2,L
        I = I
D     W = FLOATF(I)
D   8 G(I)=1./(2.*W)*(-(X**(2*I-3)*Z*SQRTF(Z))+(2.*W-3.)*G(I-1))
D     GINDE = G(L)
    9 RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

FUNCTION DEFIN(L,A,B)

```
      FUNCTION DEFIN(L,A,B)
C                       COMPUTES THE DEFINITE INTEGRALS
D   1 G1 = GINDE(L,A)
D   2 G2 = GINDE(L,B)
D     DEFIN = G2-G1
D     RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

General routines

SUBROUTINE QUADRA(C)

```
      SUBROUTINE QUADRA(C)
C                     SOLVES THE QUADRATIC-PROGRAMMING PROBLEM
D     DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
D     DIMENSION QREC(11,11),Q(11,11)
D      DIMENSION RSL(11),FMATEL(11,11)
D     DIMENSION P(20,20),E(20),T(40),X(40)
D     DIMENSION WE(40,11)
      COMMON B,BETA,NK,GAMMA,K
      COMMON NT
      COMMON QREC,Q
      COMMON RSL,FMATEL
      COMMON P,E,T,X
      COMMON WE
D       DIMENSION C(11),CC(11)
D     DIMENSION U(20),UPSI(20),R(20),V(20),G(20,20),W(20)
D     DIMENSION A(20),Z(20)
      DO 500 I = 1,K
D 500 UPSI(I) = 0.
      DO 68 I = 1,20
D  85 V(I) = 0.
      DO 68 M = 1,K
      DO 68 N = 1,K
D  68 V(I) = V(I) + P(I,N) * Q(N,M) * E(M) * 0.5

      DO 70 I = 1,20
-     DO 70 J = 1,20
D     G(I,J)=0.
      DO 70 L = 1,K
      DO 70 N = 1,K
D  70 G(I,J) = G(I,J) + 0.25 * P(I,N) * Q(N,L) * P(J,L)

      DO 71 I=1,20
D  71 U(I) = 0.
          NIT = 0.

   76 DO 6 I = 1,20
D   6 W(I) = 0.
      DO 58 J = 2,20
D  58 W(1) = G(1,J)*U(J) + W(1)
      W(1) = -1./G(1,1)*(W(1) + V(1)*0.5)
      IF DIVIDE CHECK 1,2
    2 IF QUOTIENT OVERFLOW 1,3
D   1    W(1) = -1.E37 * (W(1) + V(1) * 0.5)
D   3 IF(W(1))66,66,67
D 66  U(1) = 0.
      GO TO 55
D  67 U(1) = W(1)
   55 DO 73 I = 2,20
D      A(I) = 0.
D       Z(I) = 0.
      L = I-1
      DO 56 J = 1,L
D  56 A(I) = G(I,J)*U(J) +A(I)
       M = I + 1
       DO 57 J = M,20
D  57 Z(I) = G(I,J)*U(J) + Z(I)
```

```
      SUBROUTINE QUADRA(C)

D      W(I) = -1./G(I,I)*(A(I) + Z(I) + 0.5*V(I))
      IF DIVIDE CHECK 4,5
    5 IF QUOTIENT OVERFLOW 4,10
D   4    W(I) = -1.E37 * (A(I) + Z(I) + 0.5*V(I))
D  10 IF(W(I)) 91,91,74
D  91 U(I) = 0.
      GO TO 73
D  74 U(I) = W(I)
   73 CONTINUE

      DO 40 J = 1,19
D  40 W(20) = W(20) + G(20,J) * U(J)
D     W(20) = -1./G(20,20)*(W(20) + V(20)*0.5)
      IF DIVIDE CHECK 7,8
    8 IF QUOTIENT OVERFLOW 7,9
D   7 W(20) = -1.E37 * (W(20) + V(20)*0.5)
D   9  IF(W(20)) 41,41,42
D  41 U(20) = 0.
      GO TO 43
D  42  U(20) = W(20)
   43 DO 25 L=1,K
D     CC(L) = 0.
      DO 25 N = 1,20
D  25 CC(L) = CC(L)+P(N,L)*U(N)
      DO 84 I = 1,K
D     C(I) = 0.
      DO 84 L = 1,K
D  84 C(I) = C(I)-Q(I,L)*(CC(L)+E(L))*0.5
         NIT = NIT + 1
      DO 501 I = 1,K
D           UPI = ABSF(UPSI(I) - C(I))
D           UP  = UPI /ABSF( UPSI(I))
      IF DIVIDE CHECK 1001,1002
 1002 IF QUOTIENT OVERFLOW 503,1003
 1001  IF(UPI - 0.0001) 501,501,503
 1003 IF(UP - 0.0001) 501,501,503
  501 CONTINUE
      GO TO 90
  503 DO 502 I = 1,K
D 502 UPSI(I) = C(I)
      IF(NIT - 200) 76,89,89
   90 IF(NN-4) 83,83,80
   83 NN = NN + 1
      DO 504 I = 1,K
D 504 UPSI(I) = C(I)
      IF(NIT - 200) 76,89,89
   89 WRITE OUTPUT TAPE 3,133
   80 WRITE OUTPUT TAPE 3,145,NIT
  133 FORMAT(33HLNUMBER OF ITERATIONS EXCEEDS 200/15H NO CONVERGENCE)
  104 FORMAT(1H ,6E20.4)
  145 FORMAT(1HL,I5,10HITERATIONS/)
      RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

SUBROUTINE LGLSY

```
      SUBROUTINE LGLSY
C                     SOLVES A SYSTEM OF LINEAR EQUATIONS.
C                      PROGRAMMED BY W.LUENOW.
C
D       DIMENSION DETLG(1),FMATEL(11,11),RSL(11)
D     DIMENSION T(20),ALPHA(1)
D      DIMENSION DUM(11,11),DUM1(11,11)
D      DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
      DIMENSION LIST(11)
      COMMON B,BETA,NK,GAMMA,JM
       COMMON T,ALPHA,N8
      COMMON DUM,DUM1
      COMMON RSL,FMATEL
      COMMON DETLG
       DO 1 J = 1,JM
    1 LIST(J) = 0
C
C
      DO 2 J=1,JM
D     GE = 0.0
C
C
      DO 3 K=1,JM
C
      DO 5 L=1,JM
      IF(K-LIST(L)) 5,3,5
    5 CONTINUE
C
D     IF(ABSF(FMATEL(J,K))-GE) 3,4,4
D   4 GE = ABSF(FMATEL(J,K))
      LIST(J) = K
    3 CONTINUE
C
C
D     IF(GE-1.0E-35) 1000,1001,1001
 1000 WRITE OUTPUT TAPE  3,   1100
 1100 FORMAT (49H1ABSF DES GROESSTEN ELEMENTES KLEINER ALS 1.0E-35)
      CALL EXIT
C
C
 1001 IF(J-1) 1002,1002,1003
D1002 DETLG = 0.43429*LOGF(GE)
      GO TO 10
C
D1003 DETLG = DETLG + 0.43429*LOGF(GE)
C
C
   10       M = LIST(J)
      JP = J+1
      IF(JP-JM) 11,11,800
C
   11 DO 12 K=JP,JM
D     FA = FMATEL(K,M)/FMATEL(J,M)
C
      IF DIVIDE CHECK  5101,5150
 5150 IF QUOTIENT OVERFLOW  5101,5100
```

```
      SUBROUTINE LGLSY

D5101 IF(FMATEL(K,M)) 5160,5160,5161
D5160 B1 = -1.
      GO TO 5170
D5161 B1 = +1.
C
D5170 IF(FMATEL(J,M)) 5180,5180,5181
D5180 B2 = -1.
      GO TO 5190
D5181 B2 = +1.
C
D5190 FA = 1.0E+37*B1*B2
C
 5100 DO 13 L=1,JM
C
      DO 14 L2=1,JM
      IF(L-LIST(L2)) 14,13,14
   14 CONTINUE
C
D     FMATEL(K,L) = FMATEL(K,L) - FA*FMATEL(J,L)
   13 CONTINUE
C
D     RSL(K) = RSL(K) - FA*RSL(J)
   12 CONTINUE
    2 CONTINUE
C
C
  800 DO 15 J=1,JM
      JR = JM-J+1
      M = LIST(JR)
D     FA = RSL(JR)
      KL = JR+1
      IF(KL-JM) 32,32,16
C
   32 DO 20 K=KL,JM
      M1 = LIST(K)
D  20 FA = FA - FMATEL(JR,M1)*RSL(K)
C
D  16 RSL(JR) = FA/FMATEL(JR,M)
C
      IF DIVIDE CHECK  5200,5250
 5250 IF QUOTIENT OVERFLOW  5200,15
D5200 IF(FA) 5201,5201,5202
D5201 B1 = -1.
      GO TO 5210
D5202 B1 = +1.
C
D5210 IF(FMATEL(JR,M)) 5220,5220,5221
D5220 B2 = -1.
      GO TO 5230
D5221 B2 = +1.
C
D5230 RSL(JR) = 1.0E+37*B1*B2
C
   15 CONTINUE
C
C
```

```
      SUBROUTINE LGLSY

      DO 50 J=1,JM
   34 IF(J-LIST(J)) 33,50,33
   33 M = LIST(J)
D     FA = RSL(M)
D     RSL(M) = RSL(J)
D     RSL(J) = FA
      JA = LIST(M)
      LIST(M) = LIST(J)
      LIST(J) = JA
      GO TO 34
   50 CONTINUE
C
      RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

```
      SUBROUTINE INVERS

      SUBROUTINE INVERS
C                   COMPUTES THE INVERSE OF A GIVEN MATRIX
D     DIMENSION C(11,11),CM1(11,11)
D      DIMENSION RSL(11),FMATEL(11,11)
D     DIMENSION I(20),ALPHA(1)
D      DIMENSION B(11,11),BETA(11,11),GAMMA(11,11)
      COMMON B,BETA,NK,GAMMA,K
       COMMON I,ALPHA,N8
      COMMON C,CM1
       COMMON RSL,FMATEL
      DO 5 N = 1,K
      DO 1 I = 1,K
      DO 1 J = 1,K
D   1 FMATEL(I,J) = C(I,J)
       DO 2 I = 1,K
      IF(N-I) 4,3,4
D   3 RSL(I) = 1.
      GO TO 2
D   4 RSL(I) = 0.
    2 CONTINUE
D     CALL LGLSY
      DO 5 I = 1,K
D     CM1 (I,N) = RSL(I)
    5 CONTINUE
D       RETURN
      END(1,0,0,0,0,0,1,0,0,1,0,0,0,0,0)
```

c) Example of an Output

VALUES FOR K = 5

D = 0.86775E 02

| | X OR R | T(X) | CAPITAL I | SMALL I | DIFFERENCE |
|---|---|---|---|---|---|
| 0 | -0. | 29.8000 | 29.5995 | -0. | -0.2005 |
| 1 | 0.04 | 29.9000 | 29.7110 | 0.0046 | -0.1890 |
| 2 | 0.08 | 30.0000 | 30.0469 | 0.0190 | 0.0469 |
| 3 | 0.12 | 30.4000 | 30.6095 | 0.0444 | 0.2095 |
| 4 | 0.16 | 30.7000 | 31.3993 | 0.0827 | 0.6993 |
| 5 | 0.20 | 31.3000 | 32.4091 | 0.1360 | 1.1091 |
| 6 | 0.24 | 32.2000 | 33.6182 | 0.2065 | 1.4182 |
| 7 | 0.28 | 33.2000 | 34.9852 | 0.2957 | 1.7852 |
| 8 | 0.32 | 34.7000 | 36.4415 | 0.4040 | 1.7415 |
| 9 | 0.36 | 36.8000 | 37.8872 | 0.5301 | 1.0872 |
| 10 | 0.40 | 39.5000 | 39.1888 | 0.6708 | -0.3112 |
| 11 | 0.44 | 42.4000 | 40.1818 | 0.8203 | -2.2182 |
| 12 | 0.48 | 44.8000 | 40.6785 | 0.9702 | -4.1215 |
| 13 | 0.52 | 44.7000 | 40.4817 | 1.1095 | -4.2183 |
| 14 | 0.56 | 40.7000 | 39.4054 | 1.2254 | -1.2946 |
| 15 | 0.60 | 36.4000 | 37.3031 | 1.3032 | 0.9031 |
| 16 | 0.64 | 32.0000 | 34.1012 | 1.3288 | 2.1012 |
| 17 | 0.68 | 27.5000 | 29.8379 | 1.2897 | 2.3379 |
| 18 | 0.72 | 23.0000 | 24.7020 | 1.1783 | 1.7020 |
| 19 | 0.76 | 18.5000 | 19.0657 | 0.9948 | 0.5657 |
| 20 | 0.80 | 13.8000 | 13.4993 | 0.7525 | -0.3007 |
| 21 | 0.84 | 9.7000 | 8.7475 | 0.4827 | -0.9525 |
| 22 | 0.88 | 6.3000 | 5.6253 | 0.2420 | -0.6747 |
| 23 | 0.92 | 3.6000 | 4.7355 | 0.1201 | 1.1355 |
| 24 | 0.96 | 1.5000 | 5.6704 | 0.2502 | 4.1704 |
| 25 | 1.00 | 0. | -0. | 0.8199 | -0. |

## d) Reconstruction of a given Function

F AND I ARE THE GIVEN FUNCTIONS
FS AND IS ARE THE RECONSTRUCTED FUNCTIONS
F1 AND I1 ARE THE FUNCTIONS COMPUTED AFTER THE CENTRE HAS BEEN SHIFTED BY 0.013 TO THE RIGHT
F2 AND I2 ARE THE FUNCTIONS COMPUTED FROM DISTURBED VALUES T(J)

| X | F(X) | FS(X) | F1(X) | F2(X) |
|---|---|---|---|---|
| 0.05 | 0.20000000 | 0.20000000 | 0.20015321 | 0.20722602 |
| 0.10 | 0.20057879 | 0.20057879 | 0.20087960 | 0.20416192 |
| 0.15 | 0.20226105 | 0.20226105 | 0.20288976 | 0.19604078 |
| 0.20 | 0.20488539 | 0.20488539 | 0.20573109 | 0.18587842 |
| 0.25 | 0.20818703 | 0.20818703 | 0.20881410 | 0.17775276 |
| 0.30 | 0.21180376 | 0.21180376 | 0.21160392 | 0.17562103 |
| 0.35 | 0.21528500 | 0.21528500 | 0.21378886 | 0.19203066 |
| 0.40 | 0.21810342 | 0.21810342 | 0.21535192 | 0.19704839 |
| 0.45 | 0.21967035 | 0.21967035 | 0.21648601 | 0.21772173 |
| 0.50 | 0.21935460 | 0.21935461 | 0.21732882 | 0.23832522 |
| 0.55 | 0.21650635 | 0.21650635 | 0.21755257 | 0.25151677 |
| 0.60 | 0.21048655 | 0.21048654 | 0.21592062 | 0.25033666 |
| 0.65 | 0.20070399 | 0.20070399 | 0.21000930 | 0.23073363 |
| 0.70 | 0.18666264 | 0.18666264 | 0.19637251 | 0.19401680 |
| 0.75 | 0.16802353 | 0.16802353 | 0.17146560 | 0.14834788 |
| 0.80 | 0.14468952 | 0.14468955 | 0.13359835 | 0.10816022 |
| 0.85 | 0.11692800 | 0.11692799 | 0.08590218 | 0.09031732 |
| 0.90 | 0.08556531 | 0.08556516 | 0.03964715 | 0.16042360 |
| 0.95 | 0.05234165 | 0.05234121 | 0.01531593 | 0.14831163 |
| 1.00 | 0.02074276 | 0.02074187 | 0.03271820 | 0.17471065 |

| R | I(R) | IS(R) | I1(R) | I2(R) |
|---|---|---|---|---|
| 0.05 | 0.04999999 | 0.05000003 | 0.04948173 | 0.15745167 |
| 0.10 | 0.05112187 | 0.05112191 | 0.05135211 | 0.14777616 |
| 0.15 | 0.05444999 | 0.05445004 | 0.05660973 | 0.12100688 |
| 0.20 | 0.05987187 | 0.05987190 | 0.06429449 | 0.08347224 |
| 0.25 | 0.06719999 | 0.06720003 | 0.07311862 | 0.04430833 |
| 0.30 | 0.07617188 | 0.07617190 | 0.08186080 | 0.01356678 |
| 0.35 | 0.08644999 | 0.08645003 | 0.08977760 | -0.00000419 |
| 0.40 | 0.09762187 | 0.09762192 | 0.09691401 | 0.00887304 |
| 0.45 | 0.10919999 | 0.10920005 | 0.10419234 | 0.04035231 |
| 0.50 | 0.12062187 | 0.12062193 | 0.11317902 | 0.08864064 |
| 0.55 | 0.13124999 | 0.13125005 | 0.12547867 | 0.14253467 |
| 0.60 | 0.14037187 | 0.14037193 | 0.14178772 | 0.18735798 |
| 0.65 | 0.14720000 | 0.14720005 | 0.16076186 | 0.20839927 |
| 0.70 | 0.15087187 | 0.15087193 | 0.17801835 | 0.19549729 |
| 0.75 | 0.15045000 | 0.15045010 | 0.18580905 | 0.14803383 |
| 0.80 | 0.14492187 | 0.14492209 | 0.17417128 | 0.07909641 |
| 0.85 | 0.13320000 | 0.13320035 | 0.13469267 | 0.01677017 |
| 0.90 | 0.11412187 | 0.11412232 | 0.06842185 | -0. |
| 0.95 | 0.08644999 | 0.08645026 | -0.00007860 | 0.05516473 |
| 1.00 | 0.04887187 | 0.04887119 | 0. | 0.21867814 |

e) Sketches

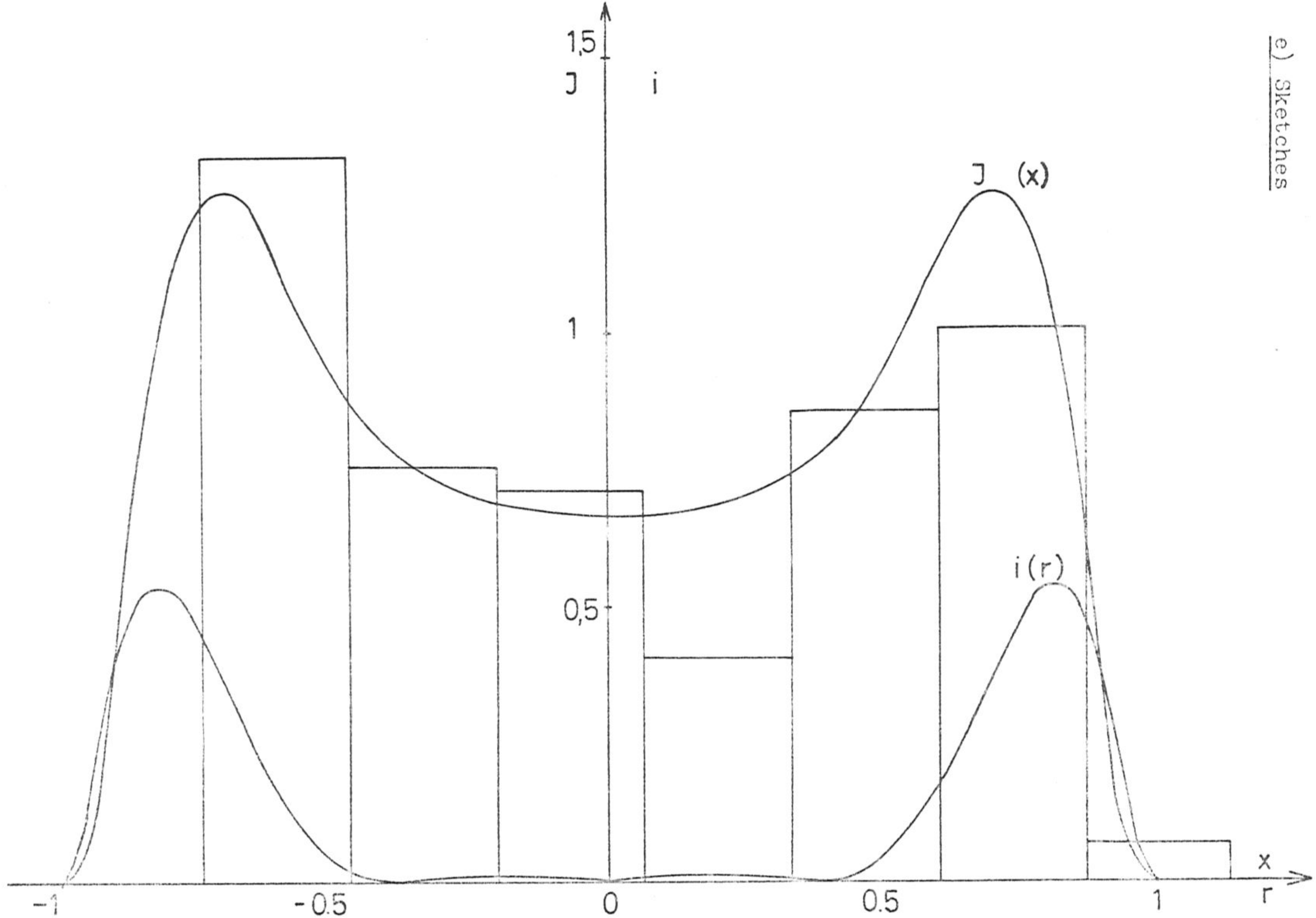


Fig.: 8

$J(x)$ = the fit to the values $T_j/h$ of the step-function.
$k = 5$ as in most of the cases which have been measured.

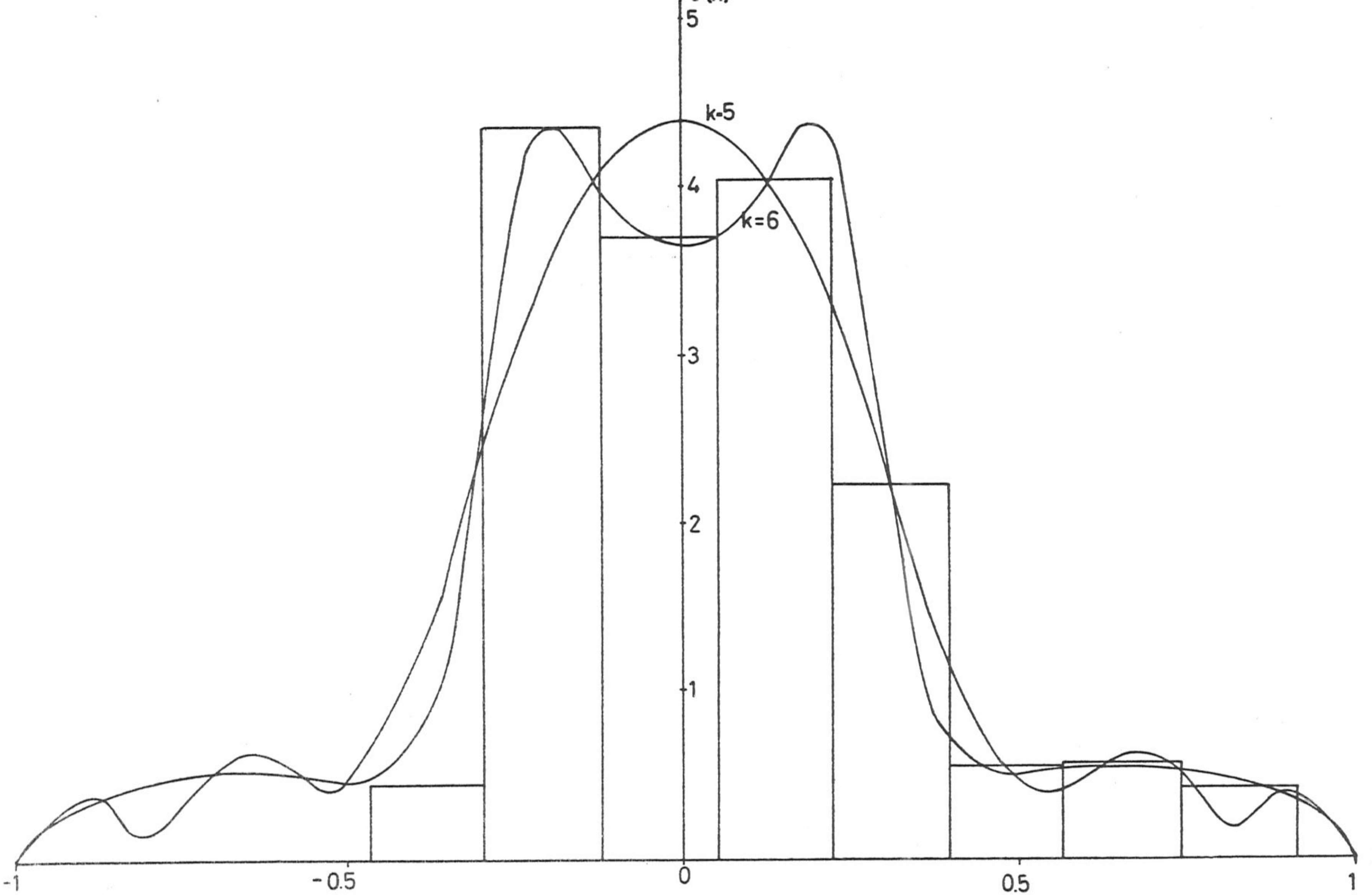


Fig.: 9

k = 5 is not large enough in order to approximate $\mathcal{J}(x)$; k = 6 is much better in this case.

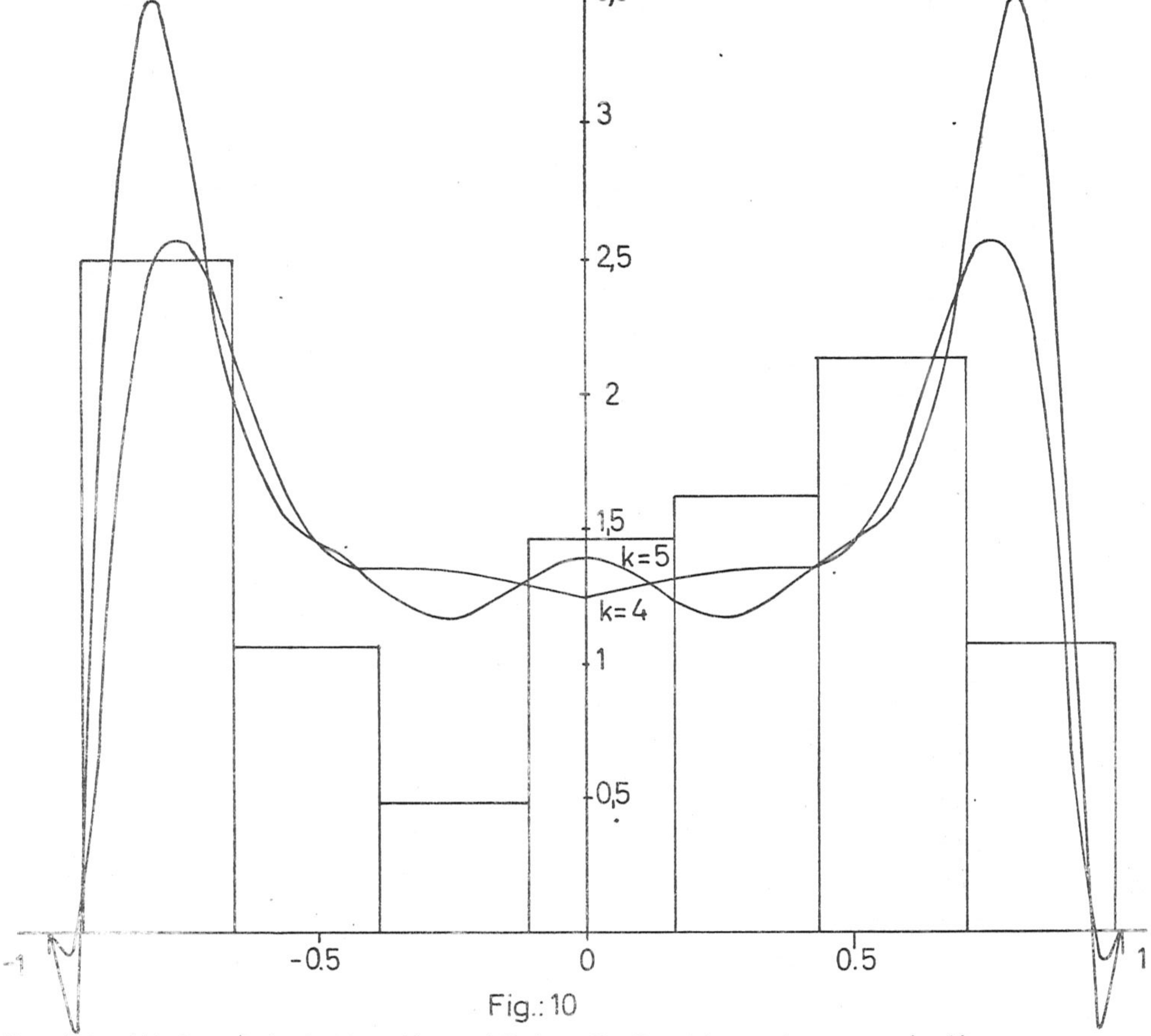


Fig.: 10

The fit with k = 4 is better than with k = 5. Negative values are in the neighbourhood of $|x| = 1$.

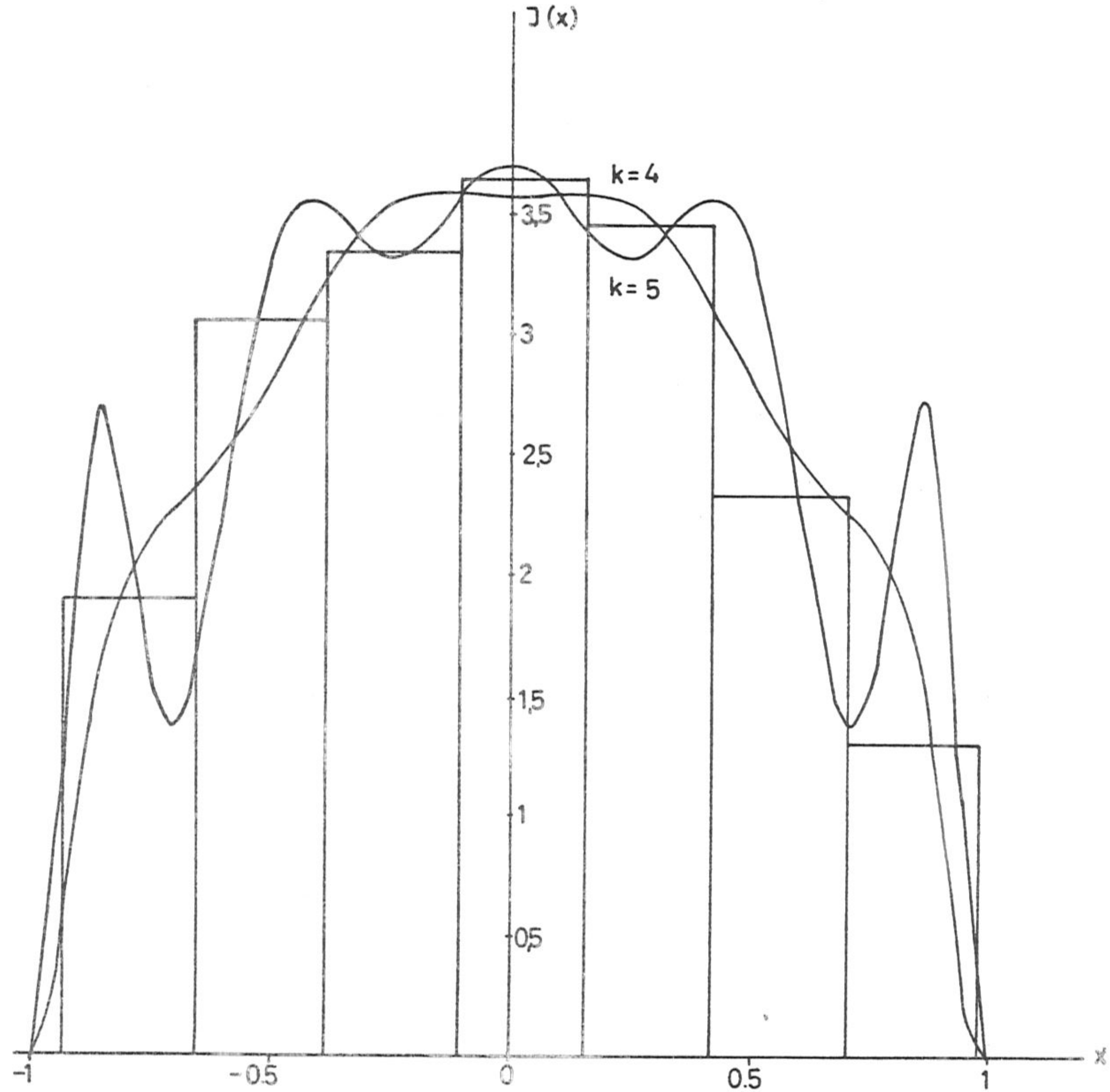


Fig.: 11

The fit with $k = 5$ is too close. $k = 4$ gives a more reasonable result.

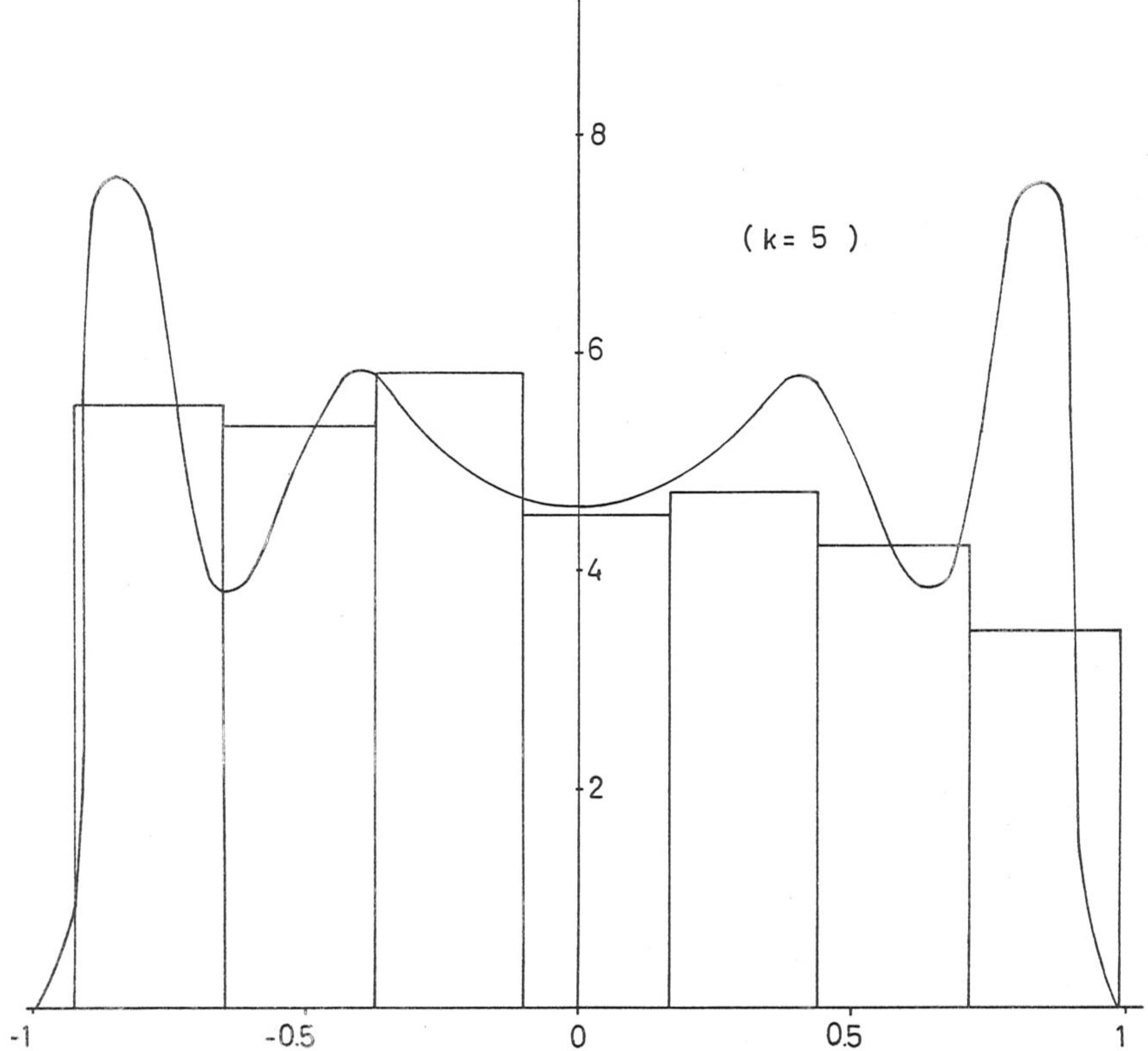


Fig.: 12

An example for the possible behaviour of $J(x)$ in the neighbourhood of $|x| = 1$.