EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

A GUIDE TO ANALYTIC EXTRAPOLATIONS

Part II: A program to be used in finding analytic correlations of data, and for the detecting zeros and poles of the scattering amplitude

I. Caprini*), M. Ciulli**), S. Ciulli**), C. Pomponiu*),
I. Sabba-Stefănescu***) and M. Sararu*)

TH-2268
23 March 1977

*) Institute for Atomic Physics, Bucharest, Rumania.

**) Theory Division, CERN, Geneva, Switzerland.

***) Institut für Theoretische Kernphysik der Universität Karlsruhe, Karlsruhe, Germany.

1. INTRODUCTION

In particle physics there often arises the delicate problem of correlating*), in an analytic way, approximative pieces of theoretical and/or experimental information about the scattering amplitude. Indeed, this latter is supposed[1] to be an (unique) analytic function in the s-cut plane. Here s is the usual square of the c.m. energy variable; more complicated analyticity domains, such as Martin's axiomatic one for the partial waves[2], present no special difficulties, only the form of the conformal mapping $\zeta(s)$ (see later) will be affected. This theoretical or experimental information about the scattering amplitude is understood to be given in the form of some complex valued data functions, $d'(s)$, $d''(s)$, ..., each one being defined on some parts Γ_1', Γ_1'', ..., of the cuts Γ, Since these approximate pieces of knowledge**) by no means represent the exact boundary values of some analytic functions, we can view them, merged in a unique data function $d(s)$, defined all over the "physical part" $\Gamma_1 = \Gamma_1' + \Gamma_1'' + \ldots$ of the cuts. In the present form of the program, the cuts Γ are $(-\infty, s_1]$, $[s_2, +\infty)$, and the physical region Γ_1 is supposed to be connected and ranging from s_2 to s_0, but more general situations can easily be rewritten by the reader himself.

The quality of the approximation will be introduced in the form of an error function $\varepsilon\cdot\sigma(s)$:

$$|f(s) - d(s)|_{s\in\Gamma_1} < \varepsilon\,\sigma(s) \tag{1.1}$$

where $f(s)$ stands for the boundary value of the (unknown) analytic scattering amplitude and $\varepsilon\cdot\sigma(s)$ is some prescribed error corridor***) [for technical reasons it is advantageous to normalize $\sigma(s_0) = 1$ (s_0 = the end point of Γ_1); then, ε will represent the error at this end point of Γ_1].

Now, it is well known [see Part 1, henceforth quoted as (1)] that analytic functions $f(s)$, satisfying Eq. (1.1), could nevertheless differ arbitrarily from each other at every point not lying on Γ_1, unless some "stabilizing" condition, for instance in the form

$$|f(s)|_{\Gamma_2} < M\,\rho(s) \tag{1.2}$$

*) Continuity or smoothness arguments are, of course, insufficient when holomorphic functions are required; sum rules might be used, but holomorphy is equivalent only with an infinite number of such sum rules!

**) A typical example is represented by some given choice among the different possible branches of a phase-shift analysis. Another example is provided by the problem of sticking a high-energy (Regge) behaviour to low-energy data. In the references, this latter case is usually treated (incompletely) with finite energy sum rules.

***) In Eq. (1.1) the error corridor has a sharp form. Nevertheless, nothing prevents the repeated use of the above program, for a sequence of values of the normalizing constant ε, corresponding to different confidence levels. The results will then refer to these prescribed levels.

is in force on the remainder Γ_2 of the cuts Γ. Here $\rho(s)$ might be known from theoretical considerations (Martin-type bounds, Froissart asymptotics), but the exact value of the constant M might not be. Again, we shall normalize $\rho(s_0) = 1$.

In any case:

A) if M is also known, Eqs. (1.1) and (1.2) may be used with the programs described in (I), to achieve optimal extrapolations to any interior point of the analyticity domain;

B) if M is not known, a condition of the type (1.2) may nevertheless be successfully used in correlating the different pieces of information contained in the piecewise given, d(s). [If $\rho(s)$ is not known either, Problem B can be stated for every choice of $\rho(s)$. Of course, better results are obtained if $\rho(s)$ is closer to the theoretical one.]

In other words, the compatibility of the condition (1.2) with the Eq. (1.1) is important (see next section) in deciding whether a given d(s) has chances of representing the scattering amplitude correctly, in the sense of (1.1).

2. MATHEMATICAL STATEMENT OF PROBLEM B.

Problem B1: Given d(s) on Γ_1, as well as the function $\sigma(s)$ and $\varepsilon > 0$, consider the class $\mathcal{F}$ of analytic functions in the s-cut plane such that Eqs. (1.1) and (1.2) should be fulfilled. Then, given $\rho(s)$ on Γ_2, for each $f(s) \in \mathcal{F}$ we can define:

$$M_f = \sup_{s \text{ on } \Gamma_2} |f(s)|/\rho(s) \tag{2.1a}$$

as well as[3])

$$M_0 = \inf_{f \in \mathcal{F}} M_f \tag{2.1b}$$

For ε small enough [such that $f \equiv 0$ should be incompatible with (1.1)], we get $M_0 \gneqq 0$.

The program looks for the number M_0, which is clearly a (non-linear) functional of d(s), $\sigma(s)$, $\rho(s)$, and ε.

Applications

(B1a): If some upper value M_{true} for M_f (Eq. 2.1) of the true scattering amplitude is known, we can disprove the analyticity of f(s) or the correctness of the data d(s), if the computed value of M_0 is greater than the above M_{true}. This is, in particular, useful in the problems discussed in the footnote (**) of page 1. Indeed, the price to pay[4,5]) in order to squeeze an analytic function on Γ_1, inside the error channel (1.1) around a "bad" data function d(s), is to have an exploding function on Γ_2, i.e. a very large M_0.

(B1b): In particular[4], if f(s) has a pole at $s = s_{0p}$, let us define the "Blaschke factor" $B(s, s_p)$ to be analytic in the same domain as f(s) and to have a modulus equal to 1, both on Γ_1 and Γ_2 [in order not to change conditions (1.1) and (1.2)] and a simple zero at $s = s_p$. Then, the product $f(s) \cdot B(s, s_p)$ is analytic in the unit disk only if $s_p = s_{0p}$. Computing the $M_0(s_p)$ corresponding to d(s) × × $B(s, s_p)$ for various s_p, we shall get, if ε is small enough, a sharp minimum at $s_p = s_{0p}$. The position of the minimum permits the evaluation of the most probable position of the pole s_{0p}, even if M_{true} is not known (but if there are nevertheless reasons to believe that it is not exceedingly high). The graph M_0 versus s_p is of interest also for greater ε's, when the minimum of $M_0(s_p)$ is flat (and hence the above method fails), since this sort of pattern informs us that analytic functions satisfying Eq. (1.1), and having a pole anywhere in the flat region above M_{true}, can *actually* be constructed (using the methods of Part 1): this would automatically disprove any other pole-search method!

(B1c): The same trick can be used to find a zero s_{0z} of the amplitude, computing the M_0 corresponding to $d(s)/B(s, s_z)$.

Problem B2: Conversely, given d(s), σ(s), ρ(s), and M, the program can compute ε_{00}, the smallest ε for which there still exists a holomorphic function satisfying (1.1) and (1.2). If

$$\varepsilon_f = \sup_{s \in \Gamma_1} |f(s) - d(s)| / \sigma(s) \tag{2.2a}$$

then

$$\varepsilon_{00} = \inf_{f \in \mathcal{F}} \varepsilon_f \tag{2.2b}$$

Applications (B2a), (B2b), (B2c)

The applications are the same as for Applications (B1a, b, c) and are denoted correspondingly: (B2a) for the computation of the ε_{00} for the input function d(s) ; (B2b) for the pole search [i.e. for the ε_{00} corresponding to $d(s) \cdot B(s, s_p)$] while (B2c) corresponds to the zeroes search [computation of ε_{00} for $d(s)/B(s,s_z)$]. Cases (B1a, b, c) and (B2a, b, c) are commuted using different key positions (KEY = 1 or 2 for Problems B1 or B2, KZ = 0, -1, +1 for the subpoints a, b, c), or different entry points [functions EMZERO (..., ..., KZ) and EPSZERO (..., ..., KZ)].

Method of solution

As in Part 1, the problem can be brought into a canonical form, mapping*) the s-cut plane onto the interior of the unit $\zeta(s)$ disk, such that Γ_1 and Γ_2 map onto the right, respectively left, half of the unit circle. Further, the weights $\sigma(\theta)$ and $\rho(\theta)$ can be absorbed into the analytic weight (outer)**) function defined by

$$|C_w(e^{i\theta})|_{\Gamma_1} = 1/\sigma(\theta) \;, \qquad |C_w(e^{i\theta})|_{\Gamma_2} = 1/\rho(\theta) \tag{2.3}$$

computed in (1), so that $d_w(\exp(i\theta)) = d(\exp(i\theta))\ C_w(\exp(i\theta))$, $f_w(\zeta) = f(\zeta)\ C_w(\zeta)$.

The inequalities (1.1) and (1.2) can be brought into the form

$$|f_w(e^{i\theta}) - d_w(e^{i\theta})| < \varepsilon \;, \quad -\pi/2 < \theta < \pi/2 \;, \; (\theta \in \Gamma_1) \tag{2.4}$$

$$|f_w(e^{i\theta})| < M \;, \quad \pi/2 < \theta < 3\pi/2 \;, \; (\theta \in \Gamma_2) . \tag{2.5}$$

Furthermore, by means of a second outer function (see, for instance, Ref. 5):

$$C_0(\zeta,\varepsilon,M) \equiv \exp\left\{-\ln(M/\varepsilon)\cdot\left[0.5 + \frac{i}{\pi}\ln\frac{1+i\zeta}{1-i\zeta}\right]\right\} \;, |C_0|_{\Gamma_1} = 1 \;, |C_0|_{\Gamma_2} = \varepsilon/M \;, \tag{2.6}$$

inequalities (2.4) and (2.5) can be combined into a single one:

$$|\tilde{f}(e^{i\theta}) - \tilde{d}(e^{i\theta})|_{\Gamma_1+\Gamma_2} < \varepsilon \tag{2.7}$$

where

$$\tilde{f}(\zeta) = f_w(\zeta)\, C_0(\zeta) = f(\zeta)\, C_0(\zeta)\, C_w(\zeta)$$
$$\tilde{d}(e^{i\theta}) = \begin{cases} d(e^{i\theta})\, C_0(e^{i\theta})\, C_w(e^{i\theta}) \;, & \text{for } \theta \text{ on } \Gamma_1 \\ 0 \;, & \text{for } \theta \text{ on } \Gamma_2 . \end{cases} \tag{2.8}$$

The point is that inequality (2.7) does not always have a solution, i.e. there does not exist for every ε and $\tilde{d}(\exp(i\theta))$, an analytic $\tilde{f}(\zeta)$ satisfying (2.7)! Indeed, by its very definition (2.8), $\tilde{d}(\exp(i\theta))$ is not an analytic function in the unit disk***), $\tilde{d} \notin \mathcal{F}$, so that there will exist a non-zero "distance" from $\tilde{d}$ to the set $\mathcal{F}$ [see Ref. 6, Eq. (3.1)]:

*) The transformation $\zeta(s) = (\sqrt{1+s'} - \sqrt{1-s'})/(\sqrt{1+s'} + \sqrt{1-s'})$ where $s' = [(s_1+s_2-2s_0)s + s_0(s_1+s_2) - 2s_1s_2]/[(s-s_0)(s_2-s_1)]$ leading from the s-cut plane to the ζ-unit disk, is performed by the function subroutine ZETA1 (S, S0, S1, S2) described in part 1. Thus, the bound functions $\sigma(s)$ and $\rho(s)$ become functions of the angle θ, [$\zeta = \exp(i\theta)$], and henceforth will be denoted by $\sigma(\theta)$ and $\rho(\theta)$.

**) i.e. analytic without zeros.

***) Indeed, if it were analytic, then its property of being zero on a part of the boundary Γ_2 would imply $\tilde{d}(\zeta) \equiv 0$ throughout the unit disk, and hence also on Γ_1.

$$\varepsilon_0\,[d;\varepsilon/M] \equiv \inf_{f\in\mathcal{F}}\Big(\sup_{0<\theta<2\pi}|\tilde{f}(e^{i\theta})-\tilde{d}(e^{i\theta})|\Big) \gneqq 0 \quad . \tag{2.9}$$

This minimal distance depends on the given data function d(s) [ε_0 is a non-linear functional*) of d], as well as on the ratio ε/M, which enters the definition of C_0, see Eq. (2.6) . Indeed, as was shown in Ref. 6 (for a proof see Appendix E of Ref. 5), the value of ε_0 is the norm, (i.e. the modulus of the greatest eigenvalue) of a certain matrix [see Eq. (3.8) of subsection 3.9], built with the negative frequency Fourier coefficients of the weighted data function $\tilde{d}(\exp\,(i\theta))$.

It is obvious that, if for some given function d(exp (iθ)) and ratio M/ε we have

$$\varepsilon_0[d;\varepsilon/M] \gneqq \varepsilon, \tag{2.10}$$

there will *be no* functions $\tilde{f} \in \mathcal{F}$ satisfying (2.7). It can be shown (see Appendix A) that for a fixed data function d(exp (iθ)) , $\varepsilon_0[d,\ \varepsilon/M]$ is a strictly increasing function of ε/M. Decreasing M, ε_0 increases, reaching the value of ε exactly when M falls down to M_0 of Eq. (2.1).

Thus $M_0(\varepsilon) \equiv \inf\,(M_f)$ is the solution of the equation $(M_0 \equiv X)$

$$\varepsilon = \varepsilon_0[d\,;\tfrac{\varepsilon}{X}] \quad , \tag{2.11}$$

for fixed ε.

Conversely, for a given M, we can look for the best approximation of d on Γ_1 by analytic functions from $\mathcal{F}$, i.e. [see Eqs. (2.2)] the smallest ε ($\equiv \varepsilon_{00}$) compatible with the equation $(\varepsilon_{00} \equiv y)$:

$$y = \varepsilon_0[d\,;\,y/M] \quad . \tag{2.12}$$

As already stated, M_0 or ε_{00} gives a "measure" of the non-analyticity of the data function and, as such, it is a very sensitive device for the detection of its singularities (poles).

The subroutine pack EMZERO is designed to solve Eqs. (2.11) and (2.12) for the data function itself (Problems B1a and B2a), as well as for the data function multiplied or divided (see Problems B1b, B2b, B1c, B2c) by the Blaschke factor B_{ζ_0}(exp (iθ)), providing the user with the value of M_0 or ε_{00}. In the ζ-variable, the Blaschke factor B_{ζ_0} takes the simple form

$$B_{\zeta_0}(e^{i\theta}) = \frac{e^{i\theta}-\zeta_0}{1-\zeta_0^{*}e^{i\theta}} \tag{2.13}$$

where $\zeta_0 = \zeta(s_p)$.

*) More precisely, ε_0 depends on the function $d_w(\zeta)$, i.e. both on $d(\zeta)$ and on $C_w(\zeta)$, but for the sake of simplicity we shall drop the index w.

3. DESCRIPTION OF THE SUBROUTINE-PACK EMZERO

3.1 Quantities to be provided by the user

The Subroutine-pack EMZERO is designed to prepare and solve the problem of finding the quantities M_0 and ε_{00} defined in the previous section [Eqs. (2.11) and (2.12)]. To this end, we need the data function d(s) to be given in NPOINT points, situated between S2 and S0, in the form of the array "DATA". Further, we need the error function $\sigma(s)$ at the same points (the array "ERROR"). The supplementary information on the rest of the cuts Γ_2, which we might have in the form of the bounding function $\rho(s)$ [see Eq. (1.2)], can be accommodated by the program defining a function subprogram, FBOUND(S).

Accordingly, the initial values (data) that must be provided by the user are:

S1 = the end of the left-hand cut.

S2 = the beginning of the right-hand cut (where data are given).

S0 = the end point of the region where the data function is known. (This is $\Gamma_1 = \{s | s_2 \leq s \leq s_0\}$.) Actually, S0 is *returned* by ANALYT (S0 ≡ ≡ SEXP(NPOINT)), so that the user could omit its determination.

NPOINT = the number of points where the data function is known.

SEXP(I) = real array, the experimental (real) values of the "energy", s [S2 ≤ ≤ SEXP(I) ≤ S0, where 1 ≤ I ≤ NPOINT].

DATA(I) = complex array, containing the complex values of the data function corresponding to the above energy values [DATA(I) ≡ $d(s_{\mathrm{exp}}$ (I))].

ERROR(I) = the real array of the error values, associated with the data [ERROR(I) = $\varepsilon\cdot\sigma(s_{\mathrm{exp}}$ (I))].

NOFBD = logical variable. If NOFBD = .TRUE., then no function bound is assumed on the rest of the cuts Γ_2 [i.e. it is supposed that $\rho(s) \equiv 1$]. If NOFBD = .FALSE., then the program assumes that a non-constant function bound exists. If so, the user must supply:

FBOUND(X) = a real-type function subprogram. Here the variable X is also real ($-\infty < X < S_1$ or $S_0 < X < \infty$), with FBOUND(X) ≡ $M\rho(X)$. It has to be declared in an EXTERNAL statement, in the calling program.

All variables discussed above constitute the input arguments for the first subroutine of the pack, ANALYT. This subroutine, the description of which will be given below, prepares the data to be processed by the rest of the program. After this unique call of ANALYT, we can use directly the function subprogram EMZERO (entry point: EPSZERO) which computes M_0 or ε_{00} according to whether EMZERO or EPSZERO was called. The initial constants (M and ε) multiplying the normalized error channel $\sigma(s)$ and the normalized function bound $\rho(s)$ [Eqs. (1.1) and (1.2)], may be changed at will, such that M_0 (or ε_{00}) can be computed for different over-all error widths or Γ_2-bounds. Further, the Problems a, b, and c of Section 2 can be commuted, changing the value of the control parameter KZ. To sum up, when calling EMZERO (or EPSZERO) the user has to provide:

EB = a real constant, representing either the value of EPS ($\equiv \varepsilon$), if M_0 is wanted (EMZERO is called), or BOUND ($\equiv$ M) when one seeks ε_{00} (when EPSZERO is called).

SZERO = means the positions of the variable pole or zero (according to KZ), to be introduced in the data function. In the present form of the program it is real, but, if necessary, it may be declared complex (in the subprogram EMZERO) with no further changes.

KZ = an integer parameter controlling the introduction of a variable zero or a variable pole in the data function in order to detect a pole (KZ = -1) or a zero (KZ = 1) of the scattering amplitude [Problems (B1b), and (B2b) or, respectively, (B1c) and (B2c)]. KZ = 0 leaves the data function unchanged [Problems (B1a) and (B2a)].

In other words, if the user wishes to compute the value of M_0 (defined, according to KZ, as a functional on the data itself or on this latter modified with a Blaschke factor), he should call EMZERO (EPS, SZERO, KZ). For ε_{00}, he should use EPSZERO (BOUND, SZERO, KZ).

Further, *no COMMON-variables have to be determined by the user,* since these latter are meant only as a link between the various subroutines of the present pack.

The EMZERO pack contains the following subprograms:

ANALYT - subroutine
EMZERO - real function (with the ENTRY EPSZERO)
ZETA1 - complex function
THETA1 - real function
STH1 - real function
EXFUN - complex function (with two ENTRY points, CW and CW1)
MOE00 - subroutine
COEF - subroutine
NORM - subroutine.

It is supposed that:

FBOUND - a real function [alias $\rho(s)$] is provided by the user (necessarily if NOFBD = .FALSE.).

As has been already emphasized, in a normal run, *only* the subroutine ANALYT and the real function EMZERO (or EPSZERO) have to be called by the principal program.

3.2 SUBROUTINE ANALYT (S0, S1, S2, NPOINT, SEXP, DATA, ERROR, FBOUND, NOFBD, EPS, BOUND)

The meaning of all arguments but the last two has been explained previously. In fact, the latter are output arguments, and are defined by the subroutine itself after $\sigma(s)$ and $\rho(s)$ [i.e. ERROR(I) and FBOUND(X)] are normalized to 1 at $s = s_0$. So:

$$\text{EPS} \equiv \text{ERROR (NPOINT)} \equiv \varepsilon; \quad \text{BOUND} \equiv \text{FBOUND(S0)} \equiv M$$

(but BOUND ≡ 1, if NOFBD = .TRUE.). Both ε and M (alias EPS and BOUND) may be redefined by the user before the functions EMZERO and EPSZERO are called.

The following *two COMMON-blocks* provide a link between the various subroutines in this pack. As already stated, none of these variables has to be assigned in the main program.

Reading the description of the various variables contained in these COMMON's, we get a first glimpse of the way the program pack is conceived:

COMMON/ANALYT/SS0, SS1, SS2, KFLAG

SS0 = the value of S0,

SS1 = the value of S1,

SS2 = the value of S2,

KFLAG = internal flag, mainly used by the extrapolation subroutines described in Part 1.

COMMON/PWDR/LWRITE, GRAD, NPT, THETA(251),
CARL(251), ERR(251), N, KSWICH, ER

LWRITE = integer parameter, controlling the amount of printed output. It is transmitted to almost every subroutine. If provides an internal knob to be set by the user. We defer its description to every specific subroutine.

GRAD = real parameter (a second internal knob) which controls the subdivision of the integration intervals in some subroutines (in the present pack, only in COEF).

NPT = identical to NPOINT. This argument together with THETA and CARL (see below) is transmitted also to the subroutine COEF which uses the first NPOINT angles (i.e. the THETA's between 0 and $\pi/2$, see below) and the weighted data function to compute negative frequency coefficients.

THETA(251) = real array of angles between 0 and $\pi/2$ (if NOFBD = .TRUE.) or between 0 and π (if NOFBD = .FALSE.). If NOFBD = .TRUE., the first NPOINT values are the images of the experimental energies SEXP(J) by the conformal mapping exp $(i\theta) = \zeta(s)$. If a non-constant function bound $\rho(s)$ is assumed to exist (i.e. if NOFBD = .FALSE.), a number of equally spaced points are added by the subroutine ANALYT between $\pi/2$ and π (J = NPOINT+1, N) in which the values of $\rho(\theta_j)$ are computed and (see below), transmitted to EXFUN.

CARL(251) = complex array, which, at the very beginning, contains the elements of DATA, i.e. the values of the complex data function corresponding to the first NPOINT angles THETA(J). Once EXFUN is called, according to KSWICH:

- if KSWICH = 0, the complex values of the outer function $C_w(\zeta)$ defined by Eq. (2.3) and computed by EXFUN at ζ_j = exp $(i\theta_j)$ [θ_J = THETA(J)] are stored in CARL.
- if KSWICH $\neq$ 0 (the case of the present program), the pre-existing values of CARL(J) are not erased by EXFUN, but are simply multiplied by the above C_w(exp $(i\theta_j)$)'s. Hence, when one leaves ANALYT (and so, also EXFUN), CARL holds the values of the "carlemanized data" C_w(exp $(i\theta)$)$\cdot$d(exp $(i\theta)$); during a call of EMZERO (or EPSZERO) this array might be modified by a Blaschke factor (see Section 3.3), but the initial values are always restored before leaving these subprograms.

THETA and CARL are the main ingredients of the subroutine COEF, which computes the negative Fourier coefficients necessary for the calculation of the functional ε_0 of Eq. (2.9).

ERR(251) = real array containing the values of the error function $\sigma(\theta_i)$ and, if NOFBD = .FALSE. ,also those of $\rho(\theta_j)$, corresponding to the THETA(J) discussed above. According to Eq. (2.3), these values coincide with those of $1/|C_w(\zeta_j)|$ on the unit circle and represents the input of the subprogram function EXFUN. Since in the actual version KSWICH = 2, EXFUN returns here the "weighted errors" $|C_w$(exp $(i\theta_j))|$*ERR(J), this permits ANALYT to check the efficiency of EXFUN; indeed, according to the theory, the latter values have to be close to 1! .

N = the total number of points on the ζ-unit circle. If NOFBD = .TRUE., N = NPOINT, but if NOFBD = .FALSE., N = NPOINT+ *a number* of points between $\pi/2$ and π (added in the subroutine ANALYT, as explained above).

KSWICH = integer control parameter acting on EXFUN, as explained above; in brief:

- if KSWICH = 0, the values C_w(exp $(i\theta_j)$) are stored in the array CARL(J).

- if KSWICH $\neq$ 0, the values $C_w(\exp(i\theta_j))$ multiplied by the already existing CARL(J); putting KSWICH = 2 causes *also* the subroutine EXFUN to return in the array ERR(J) the "weighted errors" $|C_w(\exp(i\theta_j))|$*ERR(J).

ER = the error level in the accuracy test (see the module EXFUN in Part 1, where ER $\equiv$ ERROR).

The above description already makes it clear how ANALYT acts: first, it normalizes the error channel to 1 at the last experimental point S0. There, if NOFBD = .FALSE.,i.e. if a non-constant function bound FBOUND(S) is supposed to exist, this latter is also normalized to 1 at S0.

The normalization constants EPS = ERROR(NPOINT) and BOUND = FBOUND(S0) [corresponding to ε and M of Eqs. (1.1) and (1.2)] are returned and printed out (if NOFBD = .TRUE., BOUND $\equiv$ 1). They may then be changed by the user before calling the function EMZERO and EPSZERO.

In other words, ANALYT computes the function $\sigma(\theta)$ [and $\rho(\theta)$] of Eqs. (1.1) and (1.2) in the points $\exp(i\theta_i)$ on the boundary of the unit ζ-disk on which the cut s-plane is mapped. By its very definition (see first footnote of Section 2) $\zeta(s)$ brings the upper lip of the data cut Γ_1 onto $\zeta = \exp(i\theta)$, $0 < \theta < \pi/2$. Hence, if no function bound is assumed (NOFBD = .TRUE.), all the θ's of interest lie between 0 and $\pi/2$ and are the images of the data points SEXP(J) [THETA(J) = = THETA1 {SEXP(J), S0, S1, S2}]. If, on the contrary, $\rho(\theta) \neq 1$ (NOFBD = .FALSE.), some equally spaced THETA(J) lying between $\pi/2$ and π are added to the previous ones, and so their total number N increases from NPOINT to min {251, (3NPOINT-2)}. Further, in order to compute the corresponding values of FBOUND(S), we have to resort to the function STH1 (TH, S0, S1, S2), which is the inverse of the above THETA1 function.

The values of $\sigma(\theta)$ and $\rho(\theta)$ are then stored in the array ERR, while the initial (complex) data are stored in CARL(J); in view of Eq. (2.8), CARL(J) is set to zero for all J's between NPOINT and N. Now, all the input data for EXFUN are ready and the latter can be called on (using a dummy argument). EXFUN returns via COMMON/PWDR/ (KSWICH was set equal to 2!) the weighted data $C_w(\exp(i\theta_I)) \times$ $\times$ CARL(I) in the old array CARL, and the "weighted errors" $|C_w(\exp(i\theta_I))|\cdot$ERR(I) in the array ERR. The fact that these latter have to be closest to 1, enables ANALYT to check the efficiency of the weighting subprogram EXFUN: for LWRITE $\geq$ 3, both the input data and the conformal angles, as well as the deviation from 1 of the "weighted errors", are printed.

Summing up, where ANALYT is left, the situation is the following: the conformal mapping angles and the corresponding data function weighted by $C_w(\exp(i\theta))$

are stored in the arrays THETA and CARL of COMMON/PWDR/; further, the Taylor coefficients of $\ln(C_w(z))$ are stored in the internal memory of EXFUN, permitting quick subsequent computations of $C_w(z)$ or $C_w'(z)$ for every z, when entering CW(Z) or CW1(Z).

3.3 Function subprogram EMZERO (EB, SZERO, KZ)

Entry names:

EMZERO - real type function,
EPSZERO - real type function.

Purpose: to direct the computations leading to the solution of either Eq. (2.11) or Eq. (2.12) depending on the entry name called.

Arguments: Since they have to be produced by the user's program, they were already explained in Section 3.1 To summarize,

EB = real and positive variable representing the input (ε or M).
SZERO = real variable (position of the variable pole/zero).
KZ = integer (Key parameter).

Procedure: Once the function EMZERO or EPSZERO is called, the internal parameter KEY is set equal to 1 or, respectively, to 2. EM should hold the value of ε (if EMZERO is called) or that of M (if EPSZERO is called). As already stated, EMZERO may be used either with the old or with new values of ε (= EPS) and M (= BOUND). These latter quantities are important for MOEOO (see below), since the ratio ε/M defines [see Eq. (2.6)] the second weight $C_o(\zeta, \varepsilon/M)$ which enters [see (2.8)] the $\tilde{d}(\exp(i\theta))$ function,whose coefficients are computed by COEF. Before proceeding to the computation of ε_0, if KZ $\neq$ 0, the $C_w(\exp(i\theta_j))\cdot d(\exp(i\theta_j))$ values contained in COMMON/PWDR/, are modified as follows:

- by multiplication with a Blaschke factor $B_{\zeta_0}(\exp(i\theta))$ [see (2.13)] if a pole is sought (if KZ = -1): see Applications B1b and B2b of Section 2 ;
- by division with the same factor if KZ = 1, and hence a zero is sought; see Applications B1c and B2c .

The value of $\zeta_0 \equiv \zeta(s_{zero})$ is produced upon calling the complex function ZETA1 (SZERO, S0, S1, S2). Then M_0 or ε_{00} is computed by the subroutine MOEOO, according to the value of the parameter KEY:

- if KEY = 1, Eq. (2.11) is solved and EMZERO = $M_0(\zeta_0)$ is computed.
- if KEY = 2, Eq. (2.12) is solved and EPSZERO = $\varepsilon_{00}(\zeta_0)$ is computed.

Before leaving EMZERO, if KZ was different from zero, the zero/pole built at the beginning of the program is destroyed by dividing/multiplying CARL(J) with $B_{\zeta_0}(\exp(i\theta_j))$. So the initial data are restored and COMMON/PWDR/ is ready for further calls of EMZERO/EPSZERO bearing new values of BOUND and EPS. If LWRITE ≥ 1, the value of M_0 or ε_{00} is printed together with (if KZ ≠ 0) the position of the variable pole/zero.

The following subprograms have been discussed in Part 1:

3.4 Complex function ZETA1 (S, S0, S1, S2)

S = complex variable,

S0, S1, S2 = real constants.

3.5 Function THETA1 (RS, S0, S1, S2)

RS = real variable,

S0, S1, S2 = real constants.

3.6 Complex function EXFUN(Z), with ENTRY CW(Z) and CW1(Z)

Z = complex variable,

and they have the preparatory action already described in Section 3.2. They produce respectively the conformal mapping $\zeta(s)$, then they link (both ways) the boundary of the unit disk and the cuts $\Gamma_1+\Gamma_2$, and they compute the $C_w(\zeta)$ weight and its derivative $C'_w(\zeta)$. The two latter functions are quickly calculated for every z, once the Taylor coefficients of $\ln(C_w(\zeta))$ are computed and stored during the first call of EXFUN.

3.7 The subroutine MOE00 (E00, M0, KEY, NFOUR)

3.7.1 The method

The MOE00 subroutine solves Eqs. (2.11) and (2.12):

$$\frac{\varepsilon_c[d\,;\,\varepsilon/x]}{\varepsilon} = 1 \tag{3.1}$$

$$\frac{\varepsilon_0[d\,;\,y/M]}{y} = 1 \tag{3.2}$$

Here ε (respectively M) is the error parameter (or the bound scale factor) entering the C_0-weight function. They are supposed to be known [ε, for Eq. (2.1) and M for Eq. (2.2)], while the solutions x and y are the sought-for values for M_0 and, respectively, ε_{00}. Further, d stands for the data with which MOE00 is presented, contained in COMMON/PWDR/ and supposed to represent the boundary values given on the *unit circle*, the corresponding angles being contained in the same COMMON.

The uniqueness of the solution of Eqs. (3.1) and (3.2) rests upon the important remark that while $\varepsilon_0[d, t]$ strictly increases with $t = \varepsilon/M$, $\varepsilon_0[d, t]/t$ is (see Appendix A) a strictly decreasing function of t. Hence, the points of intersection of the graphs $\varepsilon_0[d, \varepsilon/x]/\varepsilon$ and, respectively, $\varepsilon_0[d, y/M]/y$ with the line $z = 1$, appear as in Fig. 3.1 and, respectively, Fig. 3.2,

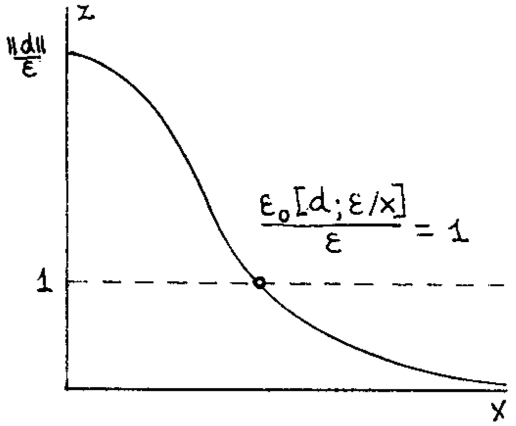

Fig. 3.1

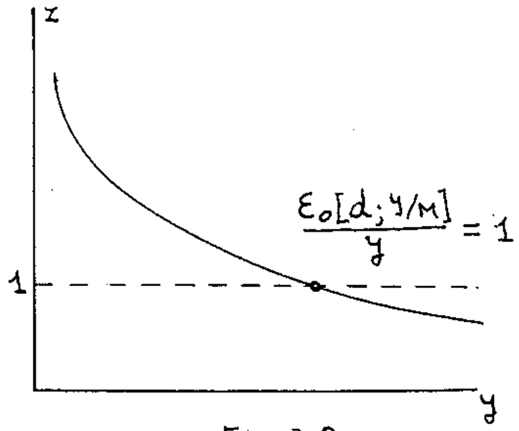

Fig. 3.2

which considerably simplifies the solving procedure.

The program works with either the quantity $\varepsilon_0[d, \varepsilon/x]/\varepsilon$, or with the quantity $\varepsilon_0[d, y/M]/y$, according to the value of the control parameter KEY set by the subprogram EMZERO. The resulting equation is solved by the chord method. To get a first coarse approximation, a logarithmical interpolation is done for $\ln(\varepsilon_0[d, \varepsilon/x]/\varepsilon)$ as a function of ln x and for $\ln(\varepsilon_0[d, y/M]/y)$ as a function of ln y. When a value sufficiently close to 1 has been obtained, a second, finer, linear (not logarithmic) interpolation is performed: more (NFOUR) Fourier coefficients are taken into account, and, on the other hand, the integration intervals (controlled by the parameter GRAD) are finer. The computation stops when a preassigned error value is reached.

3.7.2 Description of the module MOE00 (E00, M0, KEY, NFOUR)

E00 = real variable; it represents an *input* variable (= ε) if KEY = 1, or an *output* variable (= ε_{00}) if KEY = 2.

M0 = real variable; *input* (= M) if KEY = 2 or *output* (= M_0) if KEY = 1.

KEY = integer variable, used to solve either the equation $\varepsilon = \varepsilon_0[d, \varepsilon/x]$ for given ε, finding in this way $M_0(\varepsilon)$ (KEY = 1), or the equation $y = \varepsilon_0(h, y/M)$ for given M, finding thus $\varepsilon_{00}(M)$ (KEY = 2).

NFOUR = the maximum number of Fourier coefficients taken into account in NORM.

The MOE00 subprogram calls:

- the SUBROUTINE COEF that computes the N^{th} negative frequency Fourier coefficient of the weighted data function $\tilde{d}(\exp(i\theta))$ defined in Eq. (2.8).

- the SUBROUTINE NORM that computes the quantity $\varepsilon_0[d, \varepsilon/M]$ which enters Eqs. (3.1) and (3.2) using the Fourier coefficients produced by COEF.

  Quantities in common block:

COMMON/CARL/LWRITE, GRAD

- if LWRITE ≤ 0 nothing is printed;
- if LWRITE ≥ 1 the final result is printed;
- if LWRITE ≥ 2 also the intermediate approximations are displayed.

GRAD = real parameter transmitted to the SUBROUTINE COEF, controlling the fineness of the integration intervals.

3.8 Subroutine COEF (EML, NC, SUM)

Purpose: To compute the Fourier coefficients [of index (-NC)] of *the product* of some given function d(exp (iθ)) with the outer weight function C_o(exp (iθ), ε/M) defined by

$$|C_0(e^{i\theta}, \varepsilon/M)| = 1 \quad \text{for } -\alpha < \theta < \alpha \tag{3.3}$$
$$|C_0(e^{i\theta}, \varepsilon/M)| = \varepsilon/M, \text{ for } -\pi < \theta < -\alpha \text{ and } \alpha < \theta < \pi.$$

Here α ≡ THETA(NPT), (see below).

The function d(exp (iθ)) is supposed to be of real type

$$d(e^{i\theta}) = d^*(e^{-i\theta})$$

and to vanish identically between α and π (i.e. on Γ_2); owing to these two last conditions, the integration is performed only between 0 and α. Both the values of d(exp (iθ_J)) and the angles θ_J (J = 1, NPT) are transmitted to COEF via COMMON/PWDR/.

Arguments:

EML = real variable transmitting the value of the ratio ε/M which, along with the end point of Γ_1 (= α), defines C_o(exp (iθ), ε/M); namely

$$\text{EML} \equiv \frac{-1}{2\pi} \ln (\varepsilon/M) \quad .$$

NC = integer parameter, equal to minus the index of the Fourier coefficient c_{-NC}, to be computed.

SUM = real output variable, equals c_{-NC}.

Common blocks:

COMMON/PWDR/LWRITE, GRAD, NPT, THETA(251), DATA(251)

LWRITE = dummy argument.

GRAD = real argument (in degrees); if the difference between two subsequent angles exceeds 1.9 × GRAD degrees, this interval is automatically subdivided.

NPT = number of THETA(J) points where d(exp (iθ_J)] is given.

THETA = real array containing the angles where d(exp (iθ_J)) is given.

DATA = complex array holding the corresponding values of d(exp (iθ_J)).

Procedure: Since, beside exp (i•NC•θ), also the function C_o(exp (iθ), ε/M) has a rapidly varying phase (infinite rapidly varying at the end point of Γ_1), the conventional Fourier subroutines cannot be used. Indeed, *around* θ = α [see Eq. (2.6), which is valid for α = π/2, as is the case in the present paper] the function C_o(exp (iθ), ε/M) behaves like exp [-i•EML•ln (α-θ)] multiplied by a slowly varying part [S(θ, EML)]:

$$C_o(e^{i\theta}, \varepsilon/M) = X^{-iE} \cdot S(X,E) \qquad (3.4)$$

where: X = α - θ, E ≡ EML ≡ (-1/2π) ln (ε/M).

Reckoning all that, the program approximates the remainder of the integrand in intervals (X_k, X_{k+1}), around X = 0, by the parabolas:

$$e^{i\cdot NC\cdot\theta} S(X,E)\, d(e^{i\theta}) \approx \sum_{J=0}^{2} W_J X^J , \quad X_k < X < X_{k+1} \qquad (3.5)$$

Combining (3.4) and (3.5), we get:

$$\int_{X_k}^{X_{k+1}} dX\, C_0(e^{i\theta}, \varepsilon/M)\, d(e^{i\theta})\, e^{i\cdot NC\cdot\theta} \approx \sum_{J=0}^{2} W_J \left(\frac{X_{k+1}^{J+1-iE} - X^{J+1-iE}}{J+1-iE}\right) \qquad (3.6)$$

Once the point X_{k_0} is reached, beyond which the phase of C_o varies more slowly (actually much slower) than that of exp (i•NC•θ), the usual Filon method is used: C_o•d, rather than exp (i•NC•θ)•S•d, is now approximated by (3.5)-type parabolas, the rest being written as weighted sums of the integrals:

$$\int e^{-i\cdot NC\cdot X} X^J\, dX \qquad (J = 0, 1, 2)\,. \qquad (3.7)$$

As in the case of the $\int X^J \cdot X^{-iE}$ dX of Eq. (3.6), the integrals (3.7) can then be written in a closed form.

The importance of subdividing the integration intervals into finer ones (see the explanation of GRAD), is now clear; while nothing can be gained concerning the precision of d(exp (iθ)) [whose values are known only for the angles θ = THETA(I), (I = 1, NPT)], since exp (i•NC•θ), S(X, E) or C_o(exp (iθ), ε/M), entering the coefficients W_J [appearing in Eqs. (3.5) or (3.7)] are well known and calculable functions, the over-all precision may be considerably enhanced. This is especially the case when d(exp (iθ)) itself is a slowly varying function.

3.9 Subroutine NORM (NN, C, ERROR, EPSIL)

Purpose: It computes the Hermitian norm ε_0 of the Hankel matrix:

$$\hat{C} = \begin{pmatrix} c_{-1} & c_{-2} & c_{-3} & \cdot & \cdot & c_{-n} \\ c_{-2} & c_{-3} & \cdot & \cdot & c_{-n} & 0 \\ c_{-3} & \cdot & \cdot & c_{-n} & 0 & \cdot \\ \cdot & \cdot & c_{-n} & 0 & & \cdot \\ \cdot & c_{-n} & 0 & & & 0 \\ c_{-n} & 0 & \cdot & \cdot & 0 & 0 \end{pmatrix} \tag{3.8}$$

by an iteration procedure. The process is stopped when a pre-assigned error level ERROR, is reached.

Arguments:

Input: NN = the dimension of the matrix $\hat{C}$.

C = real array, holding the NN coefficients c_{-K} used to build the matrix $\hat{C}$.

ERROR = the pre-assigned error level.

Output: EPSIL = the (approximate) Hermitian norm, ε_0.

Procedure: The Hermitian norm ε_0 of the matrix $\hat{C}$ is defined as the square root of the largest eigenvalue λ_0 of the Hermitian matrix $A \equiv \hat{C}^+\hat{C}$. The algorithm consists of a successive squaring of the matrix $A \equiv \hat{C}^+\hat{C}$ and taking the traces of the resulting matrices. Indeed, since by construction A is an Hermitian matrix, it can be diagonalized:

$$A_d = U A U^{-1} \tag{3.9}$$

where

$$A_d = \begin{pmatrix} \lambda_0 & 0 & & & \\ 0 & \lambda_1 & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \lambda_{NN-1} \end{pmatrix} \tag{3.10}$$

Hence $\mathrm{Tr}(A_d)^{2^n} = (\lambda_0)^{2^n} + (\lambda_1)^{2^n} + \ldots + (\lambda_{NN-1})^{2^n}$. (3.11)

As the trace of a matrix is an invariant under unitary transformations, the above values of the traces stay true for the original matrix too. Thus

$$\mathrm{Tr}\, A = \lambda_0 + \lambda_1 + \ldots + \lambda_{NN-1} \tag{3.12}$$

$$\mathrm{Tr}\,(A^{2^n}) = \lambda_0^{2^n} + \lambda_1^{2^n} + \ldots + \lambda_{NN-1}^{2^n} \tag{3.13}$$

Further, it can be shown that the quantity

$$\bar{\lambda}_0^{(n)} \equiv \left(\mathrm{Tr}\, A^{2^n} \right)^{1/2^n} \tag{3.14}$$

tends to the largest eigenvalue of $A = \hat{C}\hat{C}^+$ (denoted here by λ_0, $\lambda_0 \equiv \varepsilon_0^2$) when $n \to \infty$.

Proof: Consider the relative error

$$\eta_n \equiv \frac{\bar{\lambda}_0^{(n)} - \lambda_0}{\lambda_0} \; . \tag{3.15}$$

Using the definition of $\bar{\lambda}_0^{(n)}$ we get

$$\eta_n = \left[1 + \sum_{i=1}^{NN-1} \left(\frac{\lambda_i}{\lambda_0} \right)^{2^n} \right]^{1/2^n} - 1 \; .$$

Let us set

$$S \equiv 1 + \sum_{i=1}^{NN-1} \left(\frac{\lambda_i}{\lambda_0} \right)^{2^n}$$

Then

$$\eta_n = S^{1/2^n} - 1 = \frac{S - 1}{1 + S^{1/2^n} + (S^{1/2^n})^2 + \ldots + (S^{1/2^n})^{2^n-1}}$$

and we obtain

$$\eta_n = \frac{\sum_{i=1}^{NN-1} (\lambda_i/\lambda_0)^{2^n}}{1 + S^{1/2^n} + (S^{1/2^n})^2 + \ldots + (S^{1/2^n})^{2^n-1}} \; .$$

It is easy to see that S obeys the bounds

$$1 < S < NN$$

which gives the following bound for the relative error η_n:

$$\eta_n < \frac{NN-1}{2^n} \xrightarrow[n \to \infty]{} 0 \; . \tag{3.16}$$

This ends the proof.

Assuming λ_0 strictly greater than the remaining eigenvalues (in fact the subroutine works on this hypothesis), the bound (3.16) can be substantially improved.

We define a quantity*)

$$\Delta \equiv 1 - \frac{\mathrm{Tr}\,(A^{2^n})}{[\mathrm{Tr}(A^{2^{n-1}})]^2} \tag{3.17}$$

After some straightforward algebraic manipulations, Δ can be cast into the form:

$$\Delta = \frac{2\lambda_0^{2^{n-1}}\,(\lambda_1^{2^{n-1}} + \ldots + \lambda_{NN-1}^{2^{n-1}}) + Q^2}{(\lambda_0^{2^{n-1}} + \quad \ldots \quad + \lambda_{NN-1}^{2^{n-1}})^2}$$

where Q^2 is a positive quantity.

Hence

$$\Delta > \frac{2\lambda_0^{2^{n-1}}\,(\lambda_1^{2^{n-1}} + \ldots + \lambda_{NN-1}^{2^{n-1}})}{(\lambda_0^{2^{n-1}} + \ldots + \lambda_{NN-1}^{2^{n-1}})} = \frac{2\lambda_0^{2^{n-1}}\,(\mathrm{Tr}A^{2^{n-1}} - \lambda_0^{2^{n-1}})}{(\mathrm{Tr}\,A^{2^{n-1}})^2} ,$$

$$\Delta > \frac{2\,\lambda_0^{2^{n-1}}}{\mathrm{Tr}\,A^{2^{n-1}}} - 2\left(\frac{\lambda_0^{2^{n-1}}}{\mathrm{Tr}\,A^{2^{n-1}}}\right)^2$$

The quantity

$$u \equiv \frac{\lambda_0^{2^{n-1}}}{\mathrm{Tr}\,A^{2^{n-1}}}$$

is related to the relative error η_{n-1} by the obvious relation

$$u^{-1} \equiv 1 + \sum_{i=1}^{NN-1} (\lambda_i/\lambda_0)^{2^{n-1}} = (1 + \eta_{n-1})^{2^{n-1}} \quad (> 1) .$$

We have thus got the inequality

$$\Delta > 2u - 2u^2 .$$

The two solutions of the equation $\Delta = 2u - 2u^2$ are:

$$u_1 = \frac{1 + \sqrt{1-2\Delta}}{2}$$

$$u_2 = \frac{1 - \sqrt{1-2\Delta}}{2} .$$

*) Δ is supposed to go to zero as n grows. It is here that the hypothesis $\lambda_0 > \lambda_i$ ($i = 1 \ldots NN-1$) is essential. If $\lambda_0 = \lambda_1$ then $\Delta \xrightarrow[n\to\infty]{} 1/2$, and if $\lambda_0 = \lambda_1 = \lambda_2$ then $\Delta \xrightarrow[n\to\infty]{} 2/3$, etc.

To obey the inequality $\Delta > 2u - 2u^2$, u will have to satisfy either $0 < u < u_2$, or $u_1 < u < 1$. Now, the first condition is violated for n large enough*). It remains the second condition which means that η_{n-1} will satisfy:

$$\frac{1+\sqrt{1-2\Delta}}{2} < \frac{1}{(1+\eta_{n-1})^{2^{n-1}}} < 1$$

or

$$1 < 1 + \eta_{n-1} < \left(\frac{2}{1+\sqrt{1-2\Delta}}\right)^{\frac{1}{2^{n-1}}}$$

We get the bounds for η_{n-1}:

$$0 < \eta_{n-1} < \left(\frac{2}{1+\sqrt{1-2\Delta}}\right)^{\frac{1}{2^{n-1}}} - 1 . \qquad (3.18)$$

The corresponding bound for η_n can be found from the equations

$$\lambda_0(1+\eta_n) = \left(\mathrm{Tr}\, A^{2^n}\right)^{1/2^n}$$
$$\lambda_0(1+\eta_{n-1}) = \left(\mathrm{Tr}\, A^{2^{n-1}}\right)^{1/2^{n-1}}$$

Together with the definition of Δ, we obtain the recurrence relation for η_n:

$$1+\eta_n = (1+\eta_{n-1})(1-\Delta)^{1/2^n}. \qquad (3.19)$$

Thus the upper bound for η_n becomes

$$0 < \eta_n < \left[\frac{4(1-\Delta)}{(1+\sqrt{1-2\Delta})^2}\right]^{1/2^n} - 1 . \qquad (3.20)$$

*) Indeed for n large, $\Delta \to 0$, so that $u_2 \to 0$, while u goes to 1 (because $\eta_{n-1} \xrightarrow[n\to\infty]{} 0$, as was shown above).

3.9.1 Programming considerations

The program computes the traces of the consecutively squared matrices $A = C^+C$. To avoid working with too large numbers (danger of overflow), renormalization is performed each time, dividing the matrix by its trace.

Thus a sequence of matrices A_i (i = 1, 2, ...) is constructed, by the recurrence relation

$$\begin{aligned} A_0 &\equiv A \\ A_n &= \frac{A_{n-1}^2}{\mathrm{Tr}(A_{n-1}^2)} \equiv \frac{A_0^{2^n}}{\mathrm{Tr}(A_0^{2^n})}, \quad n=1,2,\ldots \end{aligned} \tag{3.21}$$

Also, we denote

$$\begin{aligned} Sp(n) &\equiv \mathrm{Tr}\, A_{n-1}^2 \quad , \; n \geq 2, \\ Sp(1) &\equiv \frac{\mathrm{Tr}\, A^2}{(\mathrm{Tr}\, A)^2} \quad . \end{aligned} \tag{3.22}$$

With these notations it can be easily checked that the quantity Δ given by (3.17) can be expressed as

$$\Delta = 1 - Sp(n) \tag{3.23}$$

The approximate largest eigenvalue given by (3.14) can be expressed by means of the normalized quantities Sp(n).

To this end we shall apply successively the recurrence relation (3.21) starting from the last step:

$$\begin{aligned} A_n = \frac{1}{\mathrm{Tr}\, A_{n-1}^2} A_{n-1}^2 &= \frac{1}{(\mathrm{Tr}\, A_{n-1}^2)(\mathrm{Tr}\, A_{n-2}^2)^2} A_{n-2}^4 = \ldots \\ &= \frac{1}{(\mathrm{Tr}\, A_{n-1}^2)(\mathrm{Tr}\, A_{n-2})^2 \ldots (\mathrm{Tr}\, A_0)^{2^n-1}} A_0^{2^n} \end{aligned}$$

If we take the trace of both sides (first and last ones) noticing that $\mathrm{Tr}\, A_n = 1$ (for $n \neq 0$) by definition (3.21), we get

$$(\mathrm{Tr}\, A_0^2)^{2^n-1} \cdot (\mathrm{Tr}\, A_1^2)^{2^n-2} \ldots (\mathrm{Tr}\, A_{n-2}^2)^2 (\mathrm{Tr}\, A_{n-1}^2) = \mathrm{Tr}\, A_0^{2^n} \equiv (\bar{\lambda}_0^{(n)})^{2^n}.$$

Thus the approximation $\bar{\lambda}_0^{(n)}$ [see (3.14)] of the largest eigenvalue λ_0 has the expression

$$\bar{\lambda}_0^{(n)} = \mathrm{Tr} A \cdot Sp(1)^{1/2} \cdot Sp(2)^{1/2^2} \ldots Sp(n)^{1/2^n} . \tag{3.24}$$

The relative error associated with this value is bounded from above by

$$\eta_n^{(max)} = \left[\frac{4(1-\Delta)}{(1+\sqrt{1-2\Delta})^2} \right]^{1/2^n} - 1 \tag{3.25}$$

[with Δ given by (3.23)].

When n is large, the two terms of the difference (3.24) are very close to each other. Therefore, to avoid loss of precision by round-off errors, we can use an equivalent form of (3.25):

Denoting

$$Y \equiv \frac{4(1-\Delta)}{(1+\sqrt{1-2\Delta})^2} \quad , \quad YY \equiv Y^{1/2^n} , \tag{3.26}$$

we have the equalities

$$YY - 1 = Y^{1/2^n} - 1 = \frac{Y - 1}{1 + YY + YY^2 + \ldots + YY^{2^n-1}} \tag{3.27}$$

The numerator is

$$Y - 1 = \frac{4\Delta^2}{(1+\sqrt{1-2\Delta})^4} \tag{3.28}$$

The harmful compensation has thus been done explicitly and we are left with a formula suitable for a computer:

$$\eta_n^{(max)} = \frac{4\Delta^2}{(1+\sqrt{1-2\Delta})^4 \, (1+YY+\ldots+YY^{2^n-1})} \tag{3.29}$$

The program works iteratively. When the bound for η_n given by (3.29) gets under the pre-assigned error level ERROR, the program terminates and returns the value $\bar{\lambda}_0^{(n)}$ given by (3.24) in the variable EPSIL. If after 11 iterations (A ≃ 1000) the given ERROR level cannot be reached, a message is printed "pathological case" and the program terminates. From the analysis made above, it is clear that such a situation can occur if two or more eigenvalues of the largest modulus are equal. Still, as was shown at the beginning of this description, the $\bar{\lambda}_0^{(n)}$ given by (3.24) offers an approximation to this common value. A relative error bound is given by $N/2^n$ [cf. (3.16)], which is clearly not as good as (3.29).

4. OUTLOOK

Part 1 deals with the analytic extrapolation of some data function [DATA(J), given in the points $S_{exp}(J)$ lying between S2 and S0] to the interior points of the s-plane, cut along $(-\infty$, S1] and [S2, $\infty)$; Part 2 deals with the computation of the constants M_0 and ε_{00}. As explained in Section 2, the latter are two important functionals [see Eqs. (1.1) and (1.2)] in measuring "the degree of analyticity" of the data function we are presented with (see Problems B1a/B2a). They could be used as a sensitive detecting device in correlating, in an analytic way, low- and high-energy data, in detecting and finding the position of poles (B1b, B2b), and zeros (B1c, B2c) of the scattering amplitude or of any other meromorphic function. The complex functions (entries): CYEX, PNEX, CAEX, EXMO, EXEOO, and EXEX defined in Section 3 of Part 1 compute, respectively, the Cauchy-weighted extrapolation (dispersion integral), the Poisson-weighted dispersion integral, the centre of all the analytic extrapolations, the extremal holomorphic function corresponding to M_0, the extremal holomorphic function corresponding to ε_{00}, as well as the values of some other extremal extrapolations defined there. The real functions EMZERO or EPSZERO, defined in Part 2, return the values of the functionals M_0 and ε_{00} computed for the input data itself (if KZ is set to 0) or (if KZ = -1/+1) divided by a Blaschke factor having a pole/zero at S = SZERO.

As repeatedly stated, in a normal run the user may call only the above functions (entries) together with the subroutine ANALYT. The subroutine ANALYT (SO, S1, S2, NPOINT, SEXP, DATA, ERROR, FBOUND, NOFBND, EPS, BOUND) has a preparatory mission and has to be called only once, at the beginning of the program (the input variables were underlined; further, no common arguments are to be prepared by the user). The normalization constants EPS and BOUND are returned by ANALYT, but they may be redefined, at will, before using CYEX (X, EPS, BOUND) or EMZERO (EPS, SZERO, KZ) and EPSZERO (BOUND, SZERO, KZ). However, if there is a need to save computer time, the remark 3.1.1a of Section 3, Part 1, should be observed.

APPENDIX A

The fact that $\varepsilon_0[d, \varepsilon/x]$ and $(1/y)\varepsilon_0[d, y/M]$ are (strictly) decreasing functions of x, respectively of y, is of great importance in solving Eqs. (3.1) and (3.2) (see Figs. 3.1 and 3.2). Before proceeding to the corresponding proofs, we shall remind the reader that ε_0 denotes the least deviation of an analytic function $\tilde{f}(\zeta)$ (in the unit disk) from the (non-analytic!) boundary values

$$\tilde{d}(e^{i\theta};\varepsilon/M) \equiv d(e^{i\theta})\cdot C_0(e^{i\theta};\varepsilon/M) \ , \tag{A.1}$$

where $d(\exp(i\theta))$ is the "data function" (i.e. the measured amplitude on Γ_1, and zero, by definition, on Γ_2, but this latter fact is immaterial for the proofs which follow), while C_o is the weight function defined in (2.6):

$$|C_0(e^{i\theta};\varepsilon/M)| = \begin{cases} 1, & \text{on } \Gamma_1, \\ \varepsilon/M, & \text{on } \Gamma_2 . \end{cases} \tag{A.2}$$

Since by "deviation" we mean the largest value of the modulus of the difference between the two functions of interest (i.e. we use the L^∞ norm, $\|\tilde{f} - \tilde{d}\|_{L^\infty} = \sup_{0\le\theta\le 2\pi} |\tilde{f}(\exp(i\theta)) - \tilde{d}(\exp(i\theta), \varepsilon/M)|$), $\varepsilon_0[d, \varepsilon/M]$ is defined by

$$\varepsilon_0[d;\varepsilon/M] \equiv \inf_{\tilde{f}\in\mathcal{F}} \sup_{0<\theta<2\pi} |\tilde{f}(e^{i\theta}) - \tilde{d}(e^{i\theta};\varepsilon/M)| \ , \tag{A.3}$$

the infinuum being taken over all functions $f(\zeta)$ analytic in $|\zeta| < 1$. It is clear that ε_0 associates a *number* to every data function $d(\exp(i\theta))$ (and to every ratio ε/M), i.e. is a functional on the (L^∞) functions defined on the boundary $\zeta = \exp(i\theta)$ of the unit disk.

In the following we shall be interested in evaluating ε_0 for different values of the ratio ε/M; this amounts to working with new weighted data functions, defined by

$$\tilde{d}(e^{i\theta};\varepsilon''/M') \equiv d(e^{i\theta})\, C_0(e^{i\theta};\varepsilon''/M') \tag{A.4}$$

$$|C_0(e^{i\theta};\varepsilon''/M')| = \begin{cases} 1 & \text{on } \Gamma_1 \\ \varepsilon''/M' & \text{on } \Gamma_2' \end{cases} \tag{A.5}$$

instead of those of Eqs. (A.1) and (A.2). Then, in analogy with (A.3), we shall write:

$$\varepsilon_0[d;\varepsilon''/M'] \equiv \inf_{\tilde{f}} \sup_{0<\theta<2\pi} |\tilde{f}(e^{i\theta}) - \tilde{d}(e^{i\theta};\varepsilon''/M')| \ . \tag{A.6}$$

1. Proof that $\varepsilon_0[d, \varepsilon/M]$ is strictly decreasing with M

Let $M < M'$, and let $\tilde{f}_M(\zeta)$ be that holomorphic function which departs least from the weighted data function (A.1). (The function $\tilde{f}_M$ really exists[3]) and, moreover (see for instance Ref. 5), the difference $\tilde{f}_M(\exp(i\theta)) - \tilde{d}(\exp(i\theta), \varepsilon/M)$ has constant modulus all over $0 \leq \theta \leq 2\pi$.) Then, since [see (A.2) and (A.5)]

$$\left| C_0(e^{i\theta}; \varepsilon/M')/C_0(e^{i\theta}; \varepsilon/M) \right| = \begin{cases} 1, & \text{on } \Gamma_1 \\ M/M' < 1, & \text{on } \Gamma_2 , \end{cases} \tag{A.7}$$

we have

$$\begin{aligned} \varepsilon_0[d; \varepsilon/M] &= \sup_\theta \left| \tilde{f}_M(e^{i\theta}) - \tilde{d}(e^{i\theta}; \varepsilon/M) \right| \\ &\geq \sup_\theta \left| \frac{C_0(e^{i\theta}; \varepsilon/M')}{C_0(e^{i\theta}; \varepsilon/M)} \right| \cdot \left| \tilde{f}_M(e^{i\theta}) - \tilde{d}(e^{i\theta}; \varepsilon/M) \right| = \\ &= \sup_\theta \left| \tilde{f}_M(e^{i\theta}) \frac{C_0(e^{i\theta}; \varepsilon/M')}{C_0(e^{i\theta}; \varepsilon/M)} - \tilde{d}(e^{i\theta}; \varepsilon/M') \right| , \end{aligned} \tag{A.8}$$

where the last equality involving $\tilde{d}(\exp(i\theta), \varepsilon/M')$ follows from the use of definition (A.4).

Now, $\tilde{f}_M(\exp(i\theta)) \cdot C_0(\exp(i\theta), \varepsilon/M')/C_0(\exp(i\theta), \varepsilon/M)$ being also the boundary values of an analytic function in the unit disk, the last expression of (A.8) is surely greater than or at most equal to *the infimum*:

$$\inf_{\tilde{f}'} \sup_\theta \left| \tilde{f}'(e^{i\theta}) - \tilde{d}(e^{i\theta}; \varepsilon/M') \right| \equiv \varepsilon_0[d; \varepsilon/M'] . \tag{A.9}$$

More exactly, since the function $\tilde{f}_{M'}$ which realizes the infimum (A.9), is such that $\tilde{f}_{M'} - \tilde{d}(\exp(i\theta), \varepsilon/M')$ has constant modulus along all $\Gamma_1 + \Gamma_2$, and since [owing to the similar property of $\tilde{f}_M$ combined with *the strict* inequality (A.7)] the last expression of (A.8) has *not* this property,

$$\varepsilon_0[d; \varepsilon/M] \gneq \varepsilon_0[d; \varepsilon/M'] , \text{ for } M < M', \tag{A.10}$$

which completes the proof of the monotony of ε_0 on M.

2. Proof that $(1/\varepsilon)\varepsilon_0[d, \varepsilon/M]$ is strictly decreasing with ε

Since ε_0 depends on M or ε only through the ratio ε/M, the previous proof is equivalent to the statement that, for constant M's, $\varepsilon_0[d, \varepsilon/M]$ monotonically *increases* with ε. Nevertheless, as will be proven here, this increase is slow enough, such that $(1/\varepsilon)\varepsilon_0[d, \varepsilon/M]$ is a *decreasing* quantity.

To this end we shall define a problem which is in some respects dual to the initial one, in which the role of M and ε are inverted. Define first the weight $\overline{C_o}$ by

$$|\bar{C}_o(e^{i\theta}; M/\varepsilon)| = \begin{cases} M/\varepsilon & \text{on } \Gamma_1 , \\ 1 & \text{on } \Gamma_2 , \end{cases} \tag{A.11}$$

as well as the weighted data function

$$\tilde{d}'(e^{i\theta}; M/\varepsilon) = d(e^{i\theta})\, \bar{C}_o(e^{i\theta}; M/\varepsilon) .$$

Beside the reversal of the roles of M and ε, the unique change made is the inversion of Γ₁ and Γ₂ ($|\overline{C_o}|$ equals 1 on Γ₂, rather than on Γ₁), which amounts to the change of variable θ → θ − π. But the infimum problem is invariant under this translation, so that

$$\begin{aligned} &\inf_{\tilde{f}} \sup_{-\pi<\theta<\pi} |\tilde{f}(e^{i\theta}) - \tilde{d}'(e^{i\theta}; M/\varepsilon)| \\ &= \inf_{\tilde{f}'} \sup_{0<\theta<2\pi} |\tilde{f}'(e^{i\theta}) - \tilde{d}'(e^{i(\theta-\pi)}; M/\varepsilon)| \\ &\equiv \varepsilon_o[\bar{d}; M/\varepsilon] , \quad \text{where } \bar{d}(e^{i\theta}) \equiv d(e^{i(\theta-\pi)}) . \end{aligned} \tag{A.12}$$

On the other hand, comparing Eq. (A.11) with Eq. (A.2) and remembering that an outer function is, up to a constant phase, completely determined by the boundary values of its modulus, we have

$$\bar{C}_o(e^{i\theta}; M/\varepsilon) = \frac{M}{\varepsilon} C_o(e^{i\theta}; \varepsilon/M) , \tag{A.13}$$

up to a constant phase.

Hence (up to a phase)

$$\tilde{d}'(e^{i\theta}; M/\varepsilon) = \frac{M}{\varepsilon} d(e^{i\theta})\, C_o(e^{i\theta}; \varepsilon/M) , \tag{A.14}$$

so that the left-hand side of (A.12) is nothing but the usual ε_0 for the "data function"

$$d'(e^{i\theta}) \equiv \frac{M}{\varepsilon} d(e^{i\theta}) \tag{A.15}$$

rather than d(exp (iθ)). Hence

$$\varepsilon_o[d'; \varepsilon/M] = \varepsilon_o[\bar{d}; M/\varepsilon] . \tag{A.16}$$

Now, since ε_0 has the properties of a distance, we have

$$\varepsilon_0[\lambda d ; \varepsilon/M] = \lambda\, \varepsilon_0[d ; \varepsilon/M] ,$$

so that

$$\begin{aligned} \varepsilon_0[d' ; \varepsilon/M] &\equiv \varepsilon_0[\tfrac{M}{\varepsilon} d ; \varepsilon/M] \\ &= \tfrac{M}{\varepsilon}\, \varepsilon_0[d ; \varepsilon/M] . \end{aligned} \tag{A.17}$$

Equation (A.16) then reads:

$$\frac{M}{\varepsilon} \cdot \varepsilon_0[d ; \varepsilon/M] = \varepsilon_0[\bar{d} ; M/\varepsilon] \tag{A.18}$$

This completes the proof; indeed, since the actual form $\bar{d}$ of the data function is immaterial for the proof of the monotony of $\varepsilon_0[\bar{d}, t]$ with respect to $t = M/\varepsilon$, this means that the right-hand side of (A.18) decreases when ε increases. In other words, $(1/\varepsilon)\varepsilon_0[d, \varepsilon/M]$ is a strictly decreasing function of ε, at fixed M.

REFERENCES

1) A. Martin, Scattering theory, unitarity, analyticity, and crossing, Lecture Notes in Physics (Springer Verlag, Berlin, Heidelberg, New York, 1969), Vol. 3.

2) A. Martin, ibid.
G. Auberson and N.N. Khuri, Phys. Rev. D 6, 2953 (1972).
G. Mahoux, S.M. Roy and G. Wanders, Nuclear Phys. 70B, 297 (1974).
G. Auberson and L. Epele, Nuovo Cimento 25A, 453 (1975).

3) S. Ciulli and G. Nenciu, Nuovo Cimento 8, 735 (1972).

4) I. Caprini, S. Ciulli, C. Pomponiu and I. Sabba Stefănescu, Phys. Rev. D 5, 1658 (1972).

5) S. Ciulli, C. Pomponiu and I. Sabba Stefănescu, Phys. Reports 17C, 133 (1975).

6) S. Ciulli and G. Nenciu, J. Math. Phys. 14, 1675 (1973).