ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XIV, 2 (1974)

J. PASZKOWSKA (Wrocław)

EVALUATION OF THE ABSORPTION FACTOR

1. Introduction. In structural investigations using Röntgen rays the absorption factor ν is often needed. It is defined in the following way.

The examined crystal V, which is a convex polyhedron, is irradiated by a pencil of rays parallel to a given vector k. The rays refract in the crystal and form then a pencil parallel to a given vector k'.

Let x, y, z be coordinates of the refraction point P, $l_1(x, y, z)$ the length of way of a ray in the crystal before the refraction and $l_2(x, y, z)$ the length of way after the refraction (Fig. 1). Then

$$l(x, y, z) = l_1(x, y, z) + l_2(x, y, z)$$

gives the length of way made by a ray in the crystal.

The absorption factor ν is defined by the formula

$$\nu = \frac{1}{|V|} \iiint_V e^{-\mu l(x,y,z)} dxdydz, \tag{1}$$

where μ is the absorption coefficient and $|V|$ denotes the volume of polyhedron V.

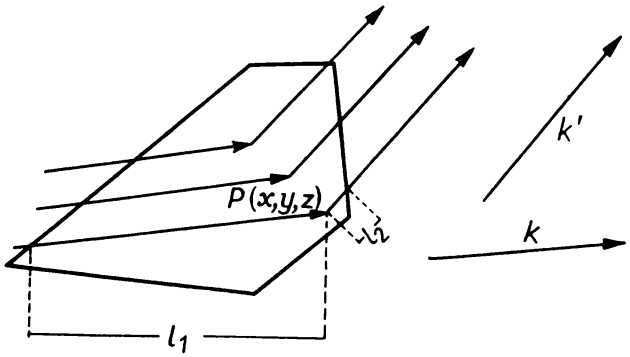

Fig. 1

The paper gives the method of evaluation of the integral (1) and the realization of this method in the form of a procedure in the language ALGOL 60.

2. Outline of the method. Evaluation of the integral (1) consists of the following steps.

A. Linear transformation of variables in the integral (1) (see Section 3), where the vectors k and k' go into the vectors $\varkappa$ and $\varkappa'$, and the polyhedron V goes into the polyhedron W.

Those faces of W through which the rays go in we call *inlet faces*, and those through which the rays go out — *outlet faces*.

B. Division of the polyhedron W into polyhedrons $W_1, W_2, \ldots$ by planes which pass through the edges of inlet faces and are parallel to the vector $\varkappa$.

Every polyhedron obtained in this way contains only one inlet face.

C. Division of every polyhedron W_j $(j = 1, 2, \ldots)$ into polyhedrons $W_{j1}, W_{j2}, \ldots$ by planes which pass through the edges of outlet faces of W and are parallel to the vector $\varkappa'$.

The rays go in every polyhedron W_{jk} through an inlet face of W and go out through an outlet face of W. For every W_{jk} the function in the transformed integral (1) is linear (and depends on equations for the corresponding inlet and outlet faces — see Section 4).

D. Division of every polyhedron W_{jk} into tetrahedrons $W_{jk1}, W_{jk2}, \ldots$ and division of the transformed integral (1) into the sum of integrals over those tetrahedrons.

For any tetrahedron the integral (1) may be expressed by an exact and simple formula (Section 5).

The above-given algorithm is expressed in the language ALGOL 60 in the form of the main procedure *integral* (Section 6) evaluating the absorption factor (1) and in the form of auxiliary procedures *part* (division of W into W_{jk} — Section 7), *plane* (determination of equations for polyhedron faces and volumes of polyhedrons W_{jk} — Section 8) and *sum* (evaluation of difference quotients — Section 8).

3. Transformation of variables. We are given the system $Oxyz$ of coordinates in which unit vectors k and k' have the components (a, b, c) and (a', b', c'), respectively.

Let us define a linear transformation T such that vectors $\varkappa = (1, 0, 0)$ and $\varkappa' = (0, 1, 0)$ imply vectors k and k'. We have

$$\begin{aligned} x &= a\xi + a'\eta + a''\zeta, \\ y &= b\xi + b'\eta + b''\zeta, \\ z &= c\xi + c'\eta + c''\zeta, \end{aligned} \tag{2}$$

where the numbers a'', b'' and c'' are so chosen that

$$\begin{vmatrix} a & a' & a'' \\ b & b' & b'' \\ c & c' & c'' \end{vmatrix} \neq 0. \tag{3}$$

To this end it suffices to assume that a'', b'' and c'' are components of any vector k'' linearly independent of vectors k and k', e.g. k'' is orthogonal to k and k'.

Obviously, the inverse transformation T^{-1} changes vectors k and k' into vectors $\varkappa$ and $\varkappa'$, respectively, which simplifies the evaluation of the length of ray in the crystal and marking inlet and outlet faces. More generally, the transformation T^{-1} carries vectors ck and ck' (c — a constant) into vectors $(c, 0, 0)$ and $(0, c, 0)$, respectively, and, therefore, it does not change their length.

Let us now perform in the integral (1) the change of variables defined by (2) and (3) implying that the jacobian of the transformation is different from zero,

$$(4) \qquad \nu = \frac{1}{|W|} \iiint_W \exp[-\mu l(a\xi + a'\eta + a''\zeta, b\xi + b'\eta + b''\zeta, c\xi + c'\eta + c''\zeta)]\, d\xi\, d\eta\, d\zeta,$$

where W is the polyhedron obtained from the polyhedron V by the transformation T.

In view of the above-remarked property of T^{-1}, the value of function l in the integral (4), being equal to the sum of lengths of vectors parallel to k and k', respectively, is equal to the sum $\lambda(\xi, \eta, \zeta)$ of lengths of two vectors: of a vector parallel to the $O\xi$-axis having its initial point at an inlet face of W and its final point at the point (ξ, η, ζ) and of a vector parallel to the $O\eta$-axis with initial point at the point (ξ, η, ζ) and final point at an outlet face of W.

Inlet and outlet faces of the crystal W can be characterized by simple inequalities. Indeed, let $S_1, S_2, \ldots$ be faces of the polyhedron W. Let

$$(5) \qquad \alpha_k \xi + \beta_k \eta + \gamma_k \zeta + \delta = 0$$

be the equation of the face S_k. This equation may be transformed in such a way that its left-hand side will be negative inside W. The numbers $\alpha_k, \beta_k, \gamma_k$ are then components of the vector ν_k orthogonal to the face S_k and directed towards the outside of W. The face S_k is *inlet* if the scalar product $(\varkappa, \nu_k)$ is negative, i.e. if $\alpha_k < 0$; S_k is an *outlet face* if $(\varkappa', \nu_k) > 0$, i.e. if $\beta_k > 0$.

4. Divisions B and C. Let us now explain why divisions B and C (Section 2) and the decomposition of the integral (4) into integrals over polyhedrons W_{jk} simplify expressions for the last integrals.

Let S_1 be an inlet face defined by the equation

$$a_1 \xi + b_1 \eta + c_1 \zeta + d_1 = 0$$

and S_2 an outlet face for which

$$a_2\xi + b_2\eta + c_2\zeta + d_2 = 0.$$

Let us take a ray, going into the crystal at the point $\Pi_1(\xi_1, \eta, \zeta)$ of the face S_1 and parallel to $O\xi$, which refracts at the point $\Pi(\xi, \eta, \zeta)$

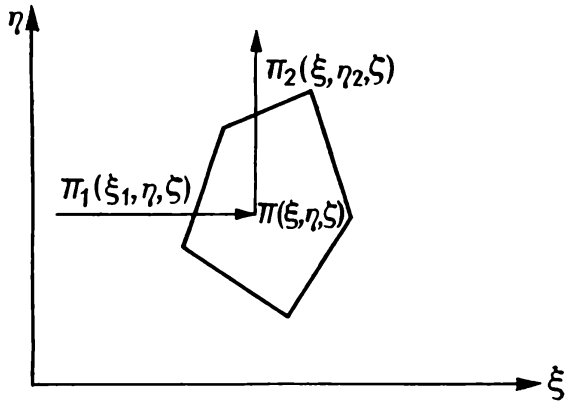

Fig. 2

and goes out at the point $\Pi_2(\xi, \eta_2, \zeta)$ of the face S_2 being parallel to $O\eta$ (Fig. 2). Then

$$\xi_1 = -\frac{1}{a_1}(b_1\eta + c_1\zeta + d_1), \qquad \eta_2 = -\frac{1}{b_2}(a_2\xi + c_2\zeta + d_2).$$

The length $\lambda(\xi, \eta, \zeta)$ of the ray way in the crystal is equal to

$$(\xi - \xi_1) + (\eta_2 - \eta) = \frac{1}{a_1}(a_1\xi + b_1\eta + c_1\zeta + d_1) - \frac{1}{b_2}(a_2\xi + b_2\eta + c_2\zeta + d_2)$$

and, therefore, it is a linear function of coordinates of the point Π, which obviously depends on the equations of faces S_1 and S_2. Therefore the evaluation of integral (1), after having performed the change of variables (Section 3) and divisions B and C of the polyhedron W (Section 2), will be reduced to evaluation of many integrals of the form

$$\cdot \iiint_U e^{\alpha\xi+\beta\eta+\gamma\zeta+\delta}\, d\xi d\eta d\zeta, \tag{6}$$

where U is a polyhedron. In particular, if U is a tetrahedron, the integral (6) is expressed by a very simple formula which will be given in the next section. That is why we divide each of polyhedrons W_{jk} into tetrahedrons W_{jkl} in the step D (Section 2).

5. Integration of the function $\exp(\alpha\xi + \beta\eta + \gamma\zeta + \delta)$ **over a tetrahedron.** We assume the following notation:

1° C denotes a tetrahedron with vertices $\Pi_k(\xi_k, \eta_k, \zeta_k)$, $k = 1, 2, 3, 4$;

2° $w_k = \alpha\xi_k + \beta\eta_k + \gamma\zeta_k + \delta$ $(k = 1, 2, 3, 4)$;

3° $\exp(w_1, w_2, w_3, w_4)$ denotes the difference quotient of the exponential function at points w_1, w_2, w_3, w_4.

We prove that

$$\iiint_C e^{\alpha\xi+\beta\eta+\gamma\zeta+\delta}\,d\xi d\eta d\zeta = 6\,|C|\exp(w_1, w_2, w_3, w_4). \tag{7}$$

Indeed, let (ξ, η, ζ) be the barycentric coordinates of a point of the tetrahedron, i.e. the numbers p_1, p_2, p_3, p_4 such that $p_1, p_2, p_3, p_4 > 0$, $p_1+p_2+p_3+p_4 = 1$ and

$$\begin{aligned}\xi &= p_1\xi_1 + p_2\xi_2 + p_3\xi_3 + p_4\xi_4,\\ \eta &= p_1\eta_1 + p_2\eta_2 + p_3\eta_3 + p_4\eta_4,\\ \zeta &= p_1\zeta_1 + p_2\zeta_2 + p_3\zeta_3 + p_4\zeta_4.\end{aligned}$$

The elimination of p_4 yields:

$$\begin{aligned}\xi &= (\xi_1-\xi_4)p_1 + (\xi_2-\xi_4)p_2 + (\xi_3-\xi_4)p_3 + \xi_4,\\ \eta &= (\eta_1-\eta_4)p_1 + (\eta_2-\eta_4)p_2 + (\eta_3-\eta_4)p_3 + \eta_4,\\ \zeta &= (\zeta_1-\zeta_4)p_1 + (\zeta_2-\zeta_4)p_2 + (\zeta_3-\zeta_4)p_3 + \zeta_4,\end{aligned}$$

$$\left|\frac{D(\xi, \eta, \zeta)}{D(p_1, p_2, p_3)}\right| = \begin{vmatrix}\xi_1-\xi_4 & \xi_2-\xi_4 & \xi_3-\xi_4\\ \eta_1-\eta_4 & \eta_2-\eta_4 & \eta_3-\eta_4\\ \zeta_1-\zeta_4 & \zeta_2-\zeta_4 & \zeta_3-\zeta_4\end{vmatrix} = 6\,|C|,$$

$$\begin{aligned}&\iiint_C e^{\alpha\xi+\beta\eta+\gamma\zeta+\delta}\,d\xi d\eta d\zeta\\ &= 6\,|C|\int_0^1 dp_1\int_0^{1-p_1} dp_2\int_0^{1-p_1-p_2}\exp\big[\alpha\big((\xi_1-\xi_4)p_1 + (\xi_2-\xi_4)p_4 +\\ &\quad + (\xi_3-\xi_4)p_3 + \xi_4\big) + \beta\big((\eta_1-\eta_4)p_1 + (\eta_2-\eta_4)p_2 + (\eta_3-\eta_4)p_3 + \eta_4\big) +\\ &\qquad + \gamma\big((\zeta_1-\zeta_4)p_1 + (\zeta_2-\zeta_4)p_2 + (\zeta_3-\zeta_4)p_3 + \zeta_4\big) + \delta\big]\,dp_3\\ &= 6\,|C|\int_0^1 dp_1\int_0^{1-p_1} dp_2\int_0^{1-p_1-p_2}\exp\big[(w_1-w_4)p_1 + (w_2-w_4)p_2 +\\ &\qquad + (w_3-w_4)p_3 + w_4\big]\,dp_3\\ &= 6\,|C|\int_0^1 dp_1\int_0^{1-p_1} dp_2\int_0^{1-p_1-p_2}\exp\big[w_1p_1 + w_2p_2 + w_3p_3 +\\ &\qquad + w_4(1-p_1-p_2-p_3)\big]\,dp_3.\end{aligned}$$

If the new variables t_1, t_2, t_3 are such that $p_1 = 1 - t_1$, $p_2 = t_1 - t_2$, $p_3 = t_3$, then the integral considered takes the following form:

$$6\,|C| \int_0^1 dt_1 \int_0^{t_1} dt_2 \int_0^{t_2} \exp[w_1(1-t_1) + w_2(t_1-t_2) + w_4(t_2-t_3) + w_3 t_3]\,dp_3.$$

As we know (see, for example, Paszkowski [1], p. 53, formula (2.4.4)), this expression is equal to

$$6\,|C| \exp(w_1, w_2, w_3, w_4),$$

which was to be proved.

6. Procedure *integral*. Procedure *integral* evaluates the absorption factor (1).

Parameters of this procedure are as follows.

Data:

qpl, *qe*, *qp* — number of faces, edges and vertices, respectively, of the polyhedron V,

psi[*1* : *qp*, *1* : *3*] — array of coordinates (x, y, z) of vertices of the polyhedron V; *psi*[*i*, *j*] denotes the j-th coordinate of the i-th vertex,

p1e, *p2e* — 1-dimensional arrays; *p1e*[*i*], *p2e*[*i*] denote numbers of vertices lying on the i-th edge ($1 \leqslant i \leqslant qe$),

ppl, *epl* — values of variables *ppl*[*1*], *ppl*[*2*], ... (*epl*[*1*], *epl*[*2*],... respectively) are the numbers of vertices (of edges respectively) on the first face, on the second face, ... on the *qpl*-th face of the polyhedron V,

a1p, *qppl* — 1-dimensional arrays; values *a1p*[*j*] and *qppl*[*j*] for $j = 1, 2, \ldots, qpl$ are such that in array *ppl* (*epl*) numbers of vertices (of edges) lying on the j-th face hold *qppl*[*j*] successive positions beginning at *ppl*[*a1p*[*j*]] (*epl*[*a1p*[*j*]]); in other words, *qppl*[*j*] is the number of vertices (edges) on the j-th face of the polyhedron,

a1pi — number of data recorded in array *ppl* (or *epl*),

ir, *or*[*1* : *3*] — arrays of components of vectors k and k',

max — maximum number of vertices lying on one face of the polyhedron V,

mi — absorption coefficient μ in formula (1).

Results:

int — absorption factor (1).

Remark. Since arrays *p1e*, *p2e*, *ppl*, *epl*, *a1p* and *qppl* are used in the procedure body as working arrays, they should be of the dimensions

p1e, *p2e*[*1* : *a*], *ppl*, *epl*[*1* : *b*], *a1p* and *qppl*[*1* : *c*], respectively, where $a = qe + max$, $b = (qpl + max) \times max$, $c = qpl + max$.

In exceptional cases these dimensions may appear to be too small and it is then to increase the value *max* and again evaluate *a*, *b* and *c*.

Example. We are given a crystal having the form of a prism (Fig. 3) whose faces, edges and vertices are arbitrarily numbered. For this crystal, parameters *qpl*, *qe*, *qp* and *max* have the values 6, 12, 8 and 4, respectively.

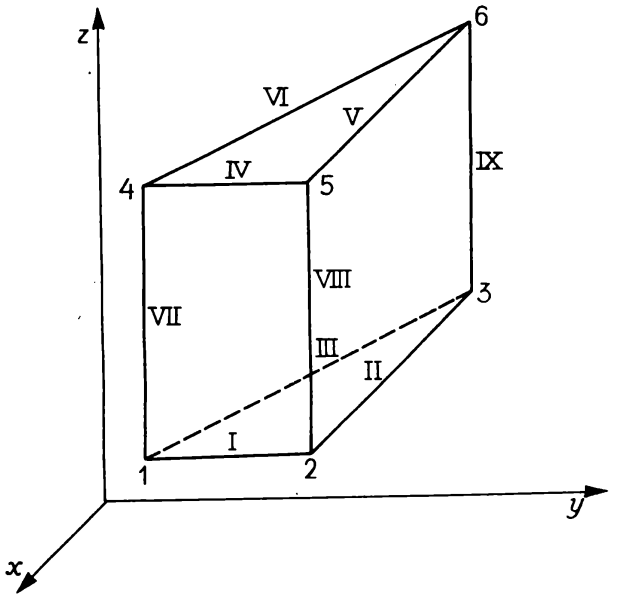

Fig. 3

Values of the variables in arrays *psi*, *p1e*, *p2e*, *a1p*, *qppl*, *ppl* and *epl* are the following:

j	*psi*[*j*, *1*]	*psi*[*j*, *2*]	*psi*[*j*, *3*]
1	−2	0	0
2	−2	4	0
3	−6	6	0
4	−2	0	4
5	−2	4	4
6	−6	6	4

k	*p1e*[*k*]	*p2e*[*k*]
1	1	2
2	2	3
3	3	1
4	4	5
5	5	6
6	6	4
7	4	1
8	5	2
9	6	3

j	*a1p*[*j*]	*qppl*[*j*]	array *ppl*	array *epl*
1	1	3	4 5 6	4 5 6
2	4	4	1 2 5 4	4 7 8 1
3	8	4	5 2 3 6	8 2 9 5
4	12	4	6 4 1 3	6 7 3 9
5	16	3	1 2 3	1 2 3

Remark. Successive rows of "array *ppl*" ("array *epl*") contain numbers of vertices (of edges) lying on successive faces of polyhedron. Every element of array *qppl* is equal to the number of elements in the same row of "array *ppl*" (or of "array *epl*"). The first element of array *a1p* is equal to 1, and every following one is the sum of elements of array *a1p* and *qppl* contained in the preceding row.

In the body of procedure *integral* (see Appendix) the following operations are performed (the corresponding numbers of lines are given in brackets):

1. Forming the matrix *wcp*[*1* : *3*, *1* : *3*] inverse to the matrix of transformation (2) (455-477).

2. The transformation of coordinates of vertices of the polyhedron V in the system $Oxyz$ into the coordinates in the system $O\xi\eta\zeta$, i.e. forming the polyhedron W (478-480).

3. Determining (by the way in Section 3) the inlet faces of polyhedron W, recording their quantity in *qpli* and numbers in *npli*[*1* : *qpli*]; determining the outlet faces, recording their quantity in *qplo* and numbers in *nplo*[*1* : *qplo*] (481-515).

4. Determining and placing in arrays *e1*, *e2*[*1* : *qe*] the coefficients of parametric equations of the edges of polyhedron W (516-521); for the equation of the j-th edge we have

$$x = e1[j, 1]t + e2[j, 1],$$
$$y = e1[j, 2]t + e2[j, 2],$$
$$z = e1[j, 3]t + e2[j, 3].$$

For the j-th inlet face ($j = 1, 2, \ldots, qpli$) procedure performs the following operations 5-9:

5. Excepting from the polyhedron W, using procedure *part* (Section 7), the part W_j corresponding to this face (523-532).

For the k-th outlet face ($k = 1, 2, \ldots, qplo$) procedure performs the following operations 6-9:

6. Evaluating and substituting for the variables *aa1*, *aa2*, *aa3* and *aa4* the coefficients α, β, γ and δ of the triple integral (6) (538-541).

7. Excepting from the polyhedron W_j (again using procedure *part*) the part W_{jk} corresponding to the k-th outlet face (542-543).

8. If the polyhedron W_{jk} is a tetrahedron, evaluating its volume, difference quotient in (7) and integral (7) by means of procedure *sum* (Section 8) (546-557).

9. If the polyhedron W_{jk} is not a tetrahedron, dividing W_{jk} into tetrahedrons W_{jkl} and performing for each of them Operation 8 (558-594).

The common vertex of these tetrahedrons is the centre of gravity of W_{jk}, and the faces of W_{jk} are their bases, the faces being, if necessary, divided into triangles.

Procedure *sum* mentioned in Section 8 sums, moreover, volumes of all tetrahedrons W_{jkl} and integrals (7) of those tetrahedrons. The last operation of procedure *integral* is evaluating the right-hand side of formula (4) (597).

7. Procedure *part*. Procedure *part* for a given polyhedron U (being the whole polyhedron W or its part W_j) and a natural number k cuts out of U a polyhedron U_j bounded by planes P_l passing through the edges e_l of the k-th face of U and parallel to the axis $O\xi$ or $O\eta$.

Parameters of procedure.

Data:

k — number of the face of U,

i — 2 (respectively 1) for the division of the polyhedron by planes parallel to $O\xi$ (respectively $O\eta$),

wqp, *wqe*, *wqpl*, *wp1e*, *wp2e*, *wa1p*, *wa1pi*, *wqppl*, *wppl*, *wepl*, *wcp* — parameters describing the divided polyhedron corresponding to the parameters *qp*, *qe*, ..., *epl*, *psi* of procedure *integral*,

we1, *we2* — 2-dimensional arrays of the coefficients of equations for edges (given in Section 6, Operation 4).

Results:

rqp, *rqe*, *rqpl*, *rp1e*, *rp2e*, *ra1p*, *rqppl*, *repl*, *rcp* — parameters describing a part of the divided polyhedron in procedure *part*; they correspond to parameters *qp*, *qe*, ..., *epl*, *psi* of procedure *integral*.

In the sequel we present procedure *part* for $i = 2$, i.e. for the case where this procedure is applied to the division of W into parts W_j (Section 2, B). If $i = 1$ (application of division of W_j into parts W_{jk}; see Section 2, C), it suffices in Operation 2 to change $O\xi$ into $O\eta$.

For each edge e_l of the face with number k (denote this face by p) procedure *part* performs Operations 1-5.

1. Determining an arbitrary vertex w of the face p of W, which is not lying on the edge e_l (34-49).

2. Writing the equation of plane P_l passing through the given edge and parallel to the axis $O\xi$ (50-52).

3. Removing vertices of W which are not lying on the same side of P_l as w is. Recording the vertices of W lying on the face S of W_j, S being on plane P_l (53-84).

4. Determining — for every edge with end points on both sides of the plane P_l — the point of intersection of plane P_l and of this edge.

Cutting out of edges which end points are on the opposite side of the plane P_l to the point w (85-122).

5. Excepting in the description of each face of polyhedron W (arrays *wppl*, *wepl*) which is crossed by plane P_l of vertices not belonging to W_j, adding of new vertices which are formed by points of intersection (if they exist; it is possible that P_l passes through the face along a diagonal) and determining the edge connecting those points. Points of intersection and newly formed edge are included into the description of the face S (123-242).

6. Including, into the description of the face, the description of the face S, ordering the arrays with descriptions of vertices, edges and faces belonging to U_j (243-305).

8. Auxiliary procedures. Auxiliary procedure *plane* determines coefficients of equation of the plane passing through given points.

Procedure *sum* (without parameters) evaluates the difference quotient of values of function e^x for arguments $ww[i]$ $(i = 1, 2, 3, 4)$ in such a way that 1° arguments being different at most $5 \cdot 10^{-6}$ are replaced by their arithmetic mean, 2° if needed, formulas for difference quotient with repeated arguments are used.

APPENDIX

```
procedure integral(qpl,qe,qp,psi,p1e,p2e,qppl,a1p,ppl,epl,a1pi,ir,or,max,
 mi,int);
 integer qpl,qe,qp,a1pi,max;
 real mi,int;
 integer array p1e,p2e,qppl,a1p,ppl,epl;
 array ir,or,psi;
 begin
  integer i,qpli,qplo,qpi,w,rqp,wqp,p1,p2,ra1pi,rqpi,wqpi,rqpl,wqpl,rqe,
  wqe,j,j1,j2,i1,p3,k1,k2;
  real a1,a2,a3,a4,aa1,aa2,aa3,aa4,w1,w2,w3,w4,v,v1;
  integer array wp1e,wp2e,rp1e,rp2e,e[1:qe+5],rqppl,ra1p,wqppl,wa1p[1:
  qpl+2],rppl,repl,wepl[1:(qpl+2)×(max+3)],npli,nplo[1:qpl];
  array cp,wcp,rcp[1:qp+5,1:3],re1,re2,e1,e2[1:qe+5,1:3],sc[1:3];
  procedure part(k,i,wqp,wqe,wqpl,wcp,wp1e,wp2e,we1,we2,wa1p,wa1pi,wqppl,
   wppl,wepl,rqp,rqe,rqpl,rcp,rp1e,rp2e,re1,re2,ra1p,ra1pi,rqppl,rppl,
   repl);
   value k,i,wqp,wqe,wqpl,wcp,wp1e,wp2e,we1,we2,wa1p,wa1pi,wqppl,wppl,
   wepl;
   integer k,i,wqp,wqe,wqpl,wa1pi,rqp,rqe,rqpl,ra1pi;
   integer array wp1e,wp2e,wa1p,wqppl,wppl,wepl,rp1e,rp2e,ra1p,rqppl,
   rppl,repl;
   array wcp,we1,we2,rcp,re1,re2;
   begin
    integer j,ji,k1,p,p1,p2,p3,pp2,pp3,pz,rx1,rx2,np,ne,w1,w2,wp,qp1,qp2,
    qp3,sg,zn,zn1,a;
    real b,c,d,t;
    integer array nppl[1:wqp+10],pl[1:wqpl+6],e,npe[1:wqe+10],wa,wa1,pe,
    ppart[1:10],ap[1:2];
    Boolean b1,b2,b3;
    for j:=1 step 1 until wqp do
```

```
      nppl[j]:=1;
     for j:=1 step 1 until wqe do
      e[j]:=1;
     for j:=1 step 1 until wqpl do
      pl[j]:=1;
     a:=a1p[k]-1;
     k1:=qppl[k];
     for j:=1 step 1 until k1 do
      begin
       ne:=epl[a+j];
       w1:=p1e[ne];
       w2:=p2e[ne];
       qp1:=qp2:=qp3:=0;
       for p:=1 step 1 until k1 do
        begin
         np:=ppl[a+p];
         if (w1≠np)∧(w2≠np)
          then go to x1
        end;
x1:    b:=e1[ne,3];
       c:=-e1[ne,1];
       d:=-(cp[w1,3]×c+cp[w1,1]×b);
       sg:=sign(b×cp[np,1]+c×cp[np,3]+d);
       for p:=1 step 1 until wqp do
        begin
         zn:=sign(b×wcp[p,1]+c×wcp[p,3]+d);
         if zn=sg
          then go to x2;
         if zn=0
          then
```

```
            begin
             nppl[p]:=0;
             qp1:=qp1+1;
             ppart[qp1]:=p
            end
            else
            begin
             nppl[p]:=-1;
             qp2:=qp2+1
            end;
x2:     end p;
       if qp2=wqp
        then
        begin
         j1:=-1;
         go to x14
        end;
       if qp2=0
        then
        begin
         for p:=1 step 1 until wqp do
          nppl[p]:=1;
         go to x3
        end;
       for p:=1 step 1 until wqe do
        begin
         p1:=wp1e[p];
         p2:=wp2e[p];
         zn:=sign(b×wcp[p1,i]+c×wcp[p1,3]+d);
         zn1:=sign(b×wcp[p2,i]+c×wcp[p2,3]+d);
```

```
b1:=zn=-sg;
b2:=zn1=-sg;
b3:=zn=sg∧b2;
if zn=0∧zn1=0
 then
 begin
  qp3:=qp3+1;
  pe[qp3]:=p;
  go to x4
 end;
if if b1 then zn1≠sg else b2∧zn=0
 then
 begin
  e[p]:=-2;
  go to x4
 end;
if b3∨b1∧zn1=sg
 then
 begin
  e[p]:=-1;
  t:=-(b×we2[p,1]+c×we2[p,3]+d)/(b×we1[p,1]+c×we1[p,3]);
  wqp:=wqp+1;
  for p1:=1,2,3 do
   wcp[wqp,p1]:=we1[p,p1]×t+we2[p,p1];
  qp1:=qp1+1;
  ppart[qp1]:=npe[p]:=wqp;
  nppl[wqp]:=1;
  if b3
   then wp2e[p]:=wqp
   else wp1e[p]:=wqp
```

```
          end;
x4:     end p;
       for p:=1 step 1 until wqp1 do
        begin
         p2:=wa1p[p]-1;
         p3:=wqpp1[p];
         pp3:=pp2:=pz:=0;
         for p1:=1 step 1 until p3 do
          wa[p1]:=wpp1[p2+p1];
         for p1:=1 step 1 until p3 do
          begin
           wp:=wa[p1];
           if npp1[wp]=-1
            then
            begin
             pp2:=pp2+1;
             wa[p1]:=0
            end
            else
             if npp1[wp]=0
              then
              begin
               pz:=pz+1;
               ap[pz]:=wp
              end
              else pp3:=pp3+1;
x5:       end;
         pp2:=pp2+pz;
         pp3:=pp3+pz;
         if pp2=p3
```

```
                then
                begin
                 pl[p]:=-1;
                 go to x7
                end;
               if pp3=p3
                then go to x7;
               pp2:=pp3:=p3;
               for p1:=1 step 1 until pp3 do
                wa1[p1]:=wepl[p2+p1];
               for p1:=1 step 1 until pp3 do
                begin
                 wp:=wa1[p1];
                 if e[wp]=-1
                  then
                  begin
                   p3:=p3+1;
                   pz:=pz+1;
                   wa[p3]:=ap[pz]:=npe[wp]
                  end;
                 if e[wp]=-2
                  then wa1[p1]: 0;
x6:             end;
               rx1:=ap[1];
               rx2:=ap[2];
               pp3:=pp3+1;
               wqe:=wqe+1;
               e[wqe]:=1;
               qp3:=qp3+1;
               pe[qp3]:=wqe;
```

```
wp1e[wqe]:=rx1;
wp2e[wqe]:=rx2;
for p1:=1,2,3 do
 begin
  we1[wqe,p1]:=wcp[rx1,p1]-wcp[rx2,p1];
  we2[wqe,p1]:=wcp[rx2,p1];
 end;
wa1[pp3]:=wqe;
ji:=0;
for p1:=1 step 1 until p3 do
 if wa[p1]≠0
  then
  begin
   ji:=ji+1;
   wa[ji]:=wa[p1]
  end;
p3:=ji;
ji:=0;
for p1:=1 step 1 until pp3 do
 if wa1[p1]≠0
  then
  begin
   ji:=ji+1;
   wa1[ji]:=wa1[p1]
  end;
ji:=p3-pp2;
if ji=0
 then go to x100
 else
 begin
```

(181-210)

```
                if p=wqp1
                 then go to x101;
                rx1:=p+1;
                if ji<0
                 then
                 begin
                  pp2:=wa1p[rx1];
                  pp3:=wa1pi;
                  w1:=1
                 end
                 else
                 begin
                  pp2:=wa1pi;
                  pp3:=wa1p[rx1];
                  w1:=-1
                 end;
                for p1:=pp2 step w1 until pp3 do
                 begin
                  wpp1[p1+ji]:=wpp1[p1];
                  wep1[p1+ji]:=wep1[p1]
                 end;
                for p1:=rx1 step 1 until wqp1 do
                 wa1p[p1]:=wa1p[p1]+ji
               end;
x101:     wa1pi:=wa1pi+ji;
          wqpp1[p]:=p3;
x100:     for p1:=1 step 1 until p3 do
           begin
            wpp1[p2+p1]:=wa[p1];
            wep1[p2+p1]:=wa1[p1]
```

```
        end;
x7:    end p;
      wqp1:=wqp1+1;
      p1[wqp1]:=1;
      p2:=wa1p[wqp1]:=wa1p1+1;
      wqpp1[wqp1]:=qp1;
      for p1:=1 step 1 until qp1 do
       begin
        pp2:=p2+p1-1;
        wpp1[pp2]:=ppart[p1];
        wep1[pp2]:=pe[p1]
       end;
      ji:=0;
      for p1:=1 step 1 until wqp do
       if nppl[p1]≠-1
        then
        begin
         ji:=ji+1;
         nppl[p1]:=ji;
         for p:=1,2,3 do
          wcp[ji,p]:=wcp[p1,p]
        end;
      wqp:=ji;
      ji:=0;
      for p1:=1 step 1 until wqe do
       if e[p1]≠-2
        then
        begin
         ji:=ji+1;
         e[p1]:=ji;
```

```
    wp1e[ji]:=nppl[wp1e[p1]];
    wp2e[ji]:=nppl[wp2e[p1]];
    for p:=1,2,3 do
     begin
      we1[ji,p]:=we1[p1,p];
      we2[ji,p]:=we2[p1,p]
     end
   end;
wqe:=ji;
ji:=pp2:=0;
for p1:=1 step 1 until wqpl do
 if pl[p1]=1
  then
  begin
   ji:=ji+1;
   pl[p1]:=ji;
   p2:=wa1p[p1]-1;
   p3:=wqppl[ji]:=wqppl[p1];
   wa1p[ji]:=pp2+1;
   for p:=1 step 1 until p3 do
    begin
     wppl[pp2+p]:=nppl[wppl[p2+p]];
     wepl[pl2+p]:=e[wepl[p2+p]]
    end;
   pp2:=pp2+p3
  end;
wqpl:=ji;
wa1pi:=pp2;
for p:=1 step 1 until wqp do
 nppl[p]:=1;
```

```
      for p:=1 step 1 until wqe do
       e[p]:=1;
      for p:=1 step 1 until wqpl do
       pl[p]:=1;
x3:  end j;
    for j:=1 step 1 until wqp do
     for p:=1,2,3 do
      rcp[j,p]:=wcp[j,p];
    rqp:=wqp;
    for j:=1 step 1 until wqe do
     begin
      rp1e[j]:=wp1e[j];
      rp2e[j]:=wp2e[j]
     end;
    rqe:=wqe;
    for j:=1 step 1 until wqpl do
     begin
      rqppl[j]:=wqppl[j];
      ra1p[j]:=wa1p[j]
     end;
    rqpl:=wqpl;
    for j=1 step 1 until wa1pi do
     repl[j]:=wepl[j];
    if i=1
     then go to x14;
    for j:=1 step 1 until wqe do
     for p:=1,2,3 do
      begin
       re1[j,p]:=we1[j,p];
       re2[j,p]:=we2[j,p]
```

```
      end;
     for j:=1 step 1 until wa1pi do
      rpp1[j]:=wpp1[j];
     ra1pi:=wa1pi;
x14:
    end part;
   procedure plane(i,a1,a2,a3,a,b,c,d);
    value a1,a2,a3;
    integer a1,a2,a3,i;
    real a,b,c,d;
    begin
     real x1,x2,x3,y1,y2,y3,z1,z2,z3;
     if i=1
      then
      begin
       x1:=cp[a1,1];
       x2:=cp[a2,1];
       x3:=cp[a3,1];
       y1:=cp[a1,2];
       y2:=cp[a2,2];
       y3:=cp[a3,2];
       z1:=cp[a1,3];
       z2:=cp[a2,3];
       z3:=cp[a3,3]
      end
      else
      begin
       x1:=wcp[a1,1];
       x2:=wcp[a2,1];
       x3:=wcp[a3,1];
```

```
    y1:=wcp[a1,2];
    y2:=wcp[a2,2];
    y3:=wcp[a3,2];
    z1:=wcp[a1,3];
    z2:=wcp[a2,3];
    z3:=wcp[a3,3];
   end;
  a:=y1×(z2-z3)+y2×(z3-z1)+y3×(z1-z2);
  b:=-(x1×(z2-z3)+x2×(z3-z1)+x3×(z1-z2));
  c:=x1×(y2-y3)+x2×(y3-y1)+x3×(y1-y2);
  d:=-(x1×(y2×z3-y3×z2)+x2×(y3×z1-y1×z3)+x3×(y1×z2-y2×z1))
 end plane;
procedure sum;
 begin
  integer i,j;
  array ww[1:4];
  ww[1]:=(aa1×wcp[p1,1]+aa2×wcp[p1,2]+aa3×wcp[p1,3]+aa4)×mi;
  ww[2]:=(aa1×wcp[p2,1]+aa2×wcp[p2,2]+aa3×wcp[p2,3]+aa4)×mi;
  ww[3]:=(aa1×wcp[p3,1]+aa2×wcp[p3,2]+aa3×wcp[p3,3]+aa4)×mi;
  ww[4]:=(aa1×sc[1]+aa2×sc[2]+aa3×sc[3]+aa4)×mi;
  for j:=1,2,3 do
   for i:=j+1 step 1 until 4 do
    if ww[j]>ww[i]
     then
     begin
      w1:=ww[j];
      ww[j]:=ww[i];
      ww[i]:=w1
     end;
  w1:=ww[1];
```

```
w2:=ww[2];
w3:=ww[3];
w4:=ww[4];
if abs(w1-w4)<.000005
 then
 begin
  a1:=exp(.25×(w1+w2+w3+w4))/6;
  go to ps1
 end;
if abs(w3-w1)<.000005
 then go to ps2;
if abs(w2-w4)<.000005
 then
 begin
  a1:=w1;
  w1:=w4;
  w4:=a1;
  go to ps2
 end;
if abs(w1-w2)<.000005∧abs(w3-w4)<.000005
 then
 begin
  a1:=(exp(w2)-exp(w3))/(w2-w3);
  a1:=((exp(.5×(w1+w2))-a1)/(w1-w3)-(a1-exp(.5×(w3+w4)))/(w2-w4))/(
      w1-w4);
  go to ps1
 end;
if abs(w1-w2)<.000005
 then go to ps3;
if abs(w2-w3)<.000005
```

```
      then
      begin
       a1:=w3;
       w3:=w1;
       w1:=a1;
       go to ps3
      end;
     if abs(w3-w4)<.000005
      then
      begin
       a1:=w3;
       w3:=w1;
       w1:=a1;
       a1:=w4;
       w4:=w2;
       w2:=a1;
       go to ps3
      end;
     a1:=(exp(w2)-exp(w3))/(w2-w3);
     a1:=(((exp(w1)-exp(w2))/(w1-w2)-a1)/(w1-w3)-(a1-(exp(w3)-exp(w4))/(
         w3-w4))/(w2-w4))/(w1-w4);
     go to ps1;
ps2:a1:=(exp((w1+w2+w3)/3)/2-(exp(.5×(w2+w3))-(exp(w3)-exp(w4))/(w3-w4))/
         (w2-w4))/(w1-w4);
     go to ps1;
ps3:a1:=(exp(w2)-exp(w3))/(w2-w3);
     a1:=((exp(.5×(w1+w2))-a1)/(w1-w3)-(a1-(exp(w3)-exp(w4))/(w3-w4))/(w2-
         w4))/(w1-w4);
ps1:plane(2,p1,p2,p3,w1,w2,w3,w4);
     v1:=abs(w1×sc[1]+w2×sc[2]+w3×sc[3]+w4);
```

```
    v:=v+v1;
    int:=int+v1×a1
   end sum;
  mi:=-mi;
  a1:=ir[1];
  a2:=ir[2];
  a3:=ir[3];
  aa1:=or[1];
  aa2:=or[2];
  aa3:=or[3];
  w3:=a1×aa2-aa1×a2;
  w1:=(aa3×a2-a3×aa2)/w3;
  w2:=(aa1×a3-a1×aa3)/w3;
  w3:=sqrt(w1×w1+w2×w2+1);
  w1:=w1/w3;
  w2:=w2/w3;
  w3:=1.0/w3;
  w4:=a1×(aa2×w3-w2×aa3)-a2×(aa1×w3-w1×aa3)+a3×(aa1×w2-w1×aa2);
  wcp[1,1]:=(aa2×w3-w2×aa3)/w4;
  wcp[1,2]:=(w1×aa3-aa1×w3)/w4;
  wcp[1,3]:=(w2×aa1-aa2×w1)/w4;
  wcp[2,1]:=(w2×a3-a2×w3)/w4;
  wcp[2,2]:=(w3×a1-a3×w1)/w4;
  wcp[2,3]:=(w1×a2-a1×w2)/w4;
  wcp[3,1]:=(aa3×a2-a3×aa2)/w4;
  wcp[3,2]:=(aa1×a3-a1×aa3)/w4;
  wcp[3,3]:=(aa2×a1-a2×aa1)/w4;
  for i:=1 step 1 until qp do
   for j:=1,2,3 do
    cp[i,j]:=wcp[j,1]×psi[i,1]+wcp[j,2]×psi[i,2]+wcp[j,3]×psi[i,3];
```

```
for j:=1,2,3 do
 begin
  w1:=0;
  for i:=1 step 1 until qp do
   w1:=w1+cp[i,j];
  sc[j]:=w1/qp
 end j;
qpli:=qplo:=0;
for i:=1 step 1 until qpl do
 begin
  for j:=1,2,3 do
   e[j]:=ppl[a1p[i]+j-1];
  plane(1,e[1],e[2],e[3],aa1,aa2,aa3,aa4);
  w4:=aa1×sc[1]+aa2×sc[2]+aa3×sc[3]+aa4;
  if w4>0
   then
   begin
    aa1:=-aa1;
    aa2:=-aa2;
    aa3:=-aa3;
    aa4:=-aa4
   end;
  if aa1<0
   then
   begin
    qpli:=qpli+1;
    npli[qpli]:=i
   end;
  if aa2>0
   then
```

```
    begin
     qplo:=qplo+1;
     nplo[qplo]:=i
    end
  end i;
 for i:=1 step 1 until qe do
  for j:=1,2,3 do
   begin
    e2[i,j]:=w1:=cp[p2e[i],j];
    e1[i,j]:=cp[p1e[i],j]-w1
   end;
 v:=int:=0;
 for i:=1 step 1 until qpli do
  begin
   j1:=npli[i];
   j2:=a1p[j1];
   plane(1,ppl[j2],ppl[j2+1],ppl[j2+2],a1,a2,a3,a4);
   a2:=a2/a1;
   a3:=a3/a1;
   a4:=a4/a1;
   part(j1,2,qp,qe,qpl,cp,p1e,p2e,e1,e2,a1p,a1pi,qppl,ppl,epl,rqp,rqe,
   rqpl,rcp,rp1e,rp2e,re1,re2,ra1p,ra1pi,rqppl,rppl,repl);
   for i1:=1 step 1 until qplo do
    begin
     j1:=nplo[i1];
     j2:=a1p[j1];
     plane(1,ppl[j2],ppl[j2+1],ppl[j2+2],aa1,aa2,aa3,aa4);
     aa1:=1-aa1/aa2;
     aa3:=a3-aa3/aa2;
     aa4:=a4-aa4/aa2;
```

```
aa2:=a2-1;
part(j1,1,rqp,rqe,rqpl,rcp,rp1e,rp2e,re1,re2,ra1p,ra1pi,rqppl,rppl,
repl,wqp,wqe,wqpl,wcp,wp1e,wp2e,re1,re2,wa1p,ra1pi,wqppl,rppl,wepl);
if j1<0
 then go to xx3;
if wqp=4
 then
 begin
  p1:=1;
  p2:=2;
  p3:=3;
  sc[1]:=wcp[4,1];
  sc[2]:=wcp[4,2];
  sc[3]:=wcp[4,3];
  sum;
  go to xx3
 end;
for j:=1,2,3 do
 begin
  w1:=0;
  for j1:=1 step 1 until wqp do
   w1:=w1+wcp[j1,j];
  sc[j]:=w1/wqp
 end j;
for j:=1 step 1 until wqpl do
 begin
  qpi:=wqppl[j];
  w:=wa1p[j]-1;
  p2:=wepl[w+1];
```

```
            p1:=wp1e[p2];
            p2:=wp2e[p2];
            for j1:=2 step 1 until qpi do
             e[j1]:=1;
            for j1:=2 step 1 until qpi-1 do
             begin
              for j2:=2 step 1 until qpi do
               if e[j2]=1
                then
                begin
                 k2:=wep1[w+j2];
                 k1:=wp1e[k2];
                 k2:=wp2e[k2];
                 if k1=p2∨k2=p2
                  then
                  begin
                   e[j2]:=0;
                   p3:=if k1=p2 then k2 else k1;
                   go to xx2
                  end
                end j2;
xx2:          sum;
              p2:=p3
             end j1
           end j;
xx3: end i1
   end i;
  int:=6×int/v
 end integral
```

Reference

[1] S. Paszkowski, *Zbiór zadań z teorii metod numerycznych*, Part I, Łódź 1969.

MATHEMATICAL INSTITUTE
UNIVERSITY OF WROCŁAW
50-384 WROCŁAW

Received on 10. 4. 1973

• PASZKOWSKA (Wrocław)

OBLICZANIE CZYNNIKA ABSORPCJI

STRESZCZENIE

W pracy rozważa się metodę obliczania czynnika absorpcji ν określonego wzorem (1), gdzie

1° V jest kryształem, a $|V|$ — jego objętością;

2° $l(x, y, z)$ jest długością drogi promienia w krysztale, załamującego się w punkcie (x, y, z) i równoległego przed załamaniem do danego wektora k, a po załamaniu — do danego wektora k';

3° μ jest współczynnikiem absorpcji.

Szkic metody jest podany w paragrafie 2, w paragrafach 3-5 opisano ją bardziej szczegółowo, a paragrafy 6-9 zawierają opisy procedur w języku ALGOL 60, realizację metody oraz wyjaśnienia niezbędne do ich zastosowania.
