

```

PROGRAM fitr
*****
* Fits a generalized drude form to reflectance data.
* From standard input it reads:
*   inputfilename
*   s/p-polarized light: charactr 's', or 'p'
*   angle of incidence in degree
*   epsilon in direction normal to plane (real number)
*   XMN XMX: frequency interval to be fitted
*   fitflag + plasmafrequency
*   fitflag + drude width
*   fitflag + epsilon_inf
*   fitflag + normal fraction
*   fitflag + wt
*   fitflag + wt*sqrt(S)
*   fitflag + gt
* The fitted curve is flushed to standard output.
*****
parameter(nfys=100000,ma=20,nca=20)
integer nfil,i,lista(ma),mfit,it,flag
real pi,xx(nfys),yy(nfys),work(nfys),tol(nfys)
*   ,a(ma),dyda(ma),covar(nca,nca),alpha(nca,nca),chisq,alamda
*   ,x,y,angle,ec,wp,epsinf,fn,gamma,beta(ma),ochisq,wmn,wmx
*   ,w0,w1,dw,alamol,chilim,wt,wl,gt
character*40 flin
character*1 spol
external reflec
common/sp/spol
common/ang/angle,ec
open(17,file='fit2cmp.log')
c   write(*,*) 'inputfilename ? '
read(*,'(a40)') flin
c   write(*,*) 's/p-polarized ? (1/0) '
read(*,'(a1)') spol
c   write(*,*) 'angle of incidence '
read(*,*) angle
pi=4.*ATAN(1.0)
angle=angle*pi/180
angle=sin(angle)**2
c   write(*,*) 'epsilon in direction normal to plane '
read(*,*) ec
c   write(*,*) 'interval to be fitted: xmn, xmx '
read(*,*) wmn,wmx
c   write(*,*) 'interval to be plotted: xmn, xmx ,dx '
read(*,*) w0, w1, dw
c   initialize fitparameters:
mfit=0
c   write(*,*) 'Should wp be varied ? '
read(*,*) it,wp
if (it.eq.1) then
mfit=1+mfit
lista(mfit)=1
endif
a(1)=wp
c   write(*,*) 'Should g be varied ? '
read(*,*) it,gamma
if (it.eq.1) then
mfit=1+mfit

```

```

        lista(mfit)=2
    endif
    a(2)=sqrt(gamma)
c    write(*,*) 'Should epsinf be varied ? '
    read(*,*) it,epsinf
    if (it.eq.1) then
        mfit=1+mfit
        lista(mfit)=3
    endif
    a(3)=epsinf
c    write(*,*) 'Should fn be varied ? '
    read(*,*) it,fn
    if (it.eq.1) then
        mfit=1+mfit
        lista(mfit)=4
    endif
***** This transformation ensures that  $0 < \text{normal fraction} < 1$  :
    a(4)=sqrt(1/fn-1)
c    write(*,*) 'Should wt be varied ? '
    read(*,*) it,wt
    if (it.eq.1) then
        mfit=1+mfit
        lista(mfit)=5
    endif
    a(5) = wt
c    write(*,*) 'Should S be varied ? '
    read(*,*) it,s
    if (it.eq.1) then
        mfit=1+mfit
        lista(mfit)=6
    endif
    a(6) = wt*sqrt(S)
c    write(*,*) 'Should gt be varied ? '
    read(*,*) it,gt
    if (it.eq.1) then
        mfit=1+mfit
        lista(mfit)=7
    endif
    a(7)=sqrt(gt)

    nfil=0
    OPEN(14,file=flin)
    do 10 i=1,nfys
        read(14,*,END=11) x,y
        if ((x.ge.wmn).and.(x.le.wmx)) then
            nfil=nfil+1
            xx(nfil)=x
            yy(nfil)=y
***** tolerance in fitting procedure:
            tol(nfil)=1.
        endif
10    continue
11    close(14)
    if (nfil.eq.0) then
        write(17,*) 'no datapoints in this data-range. '
        stop
    endif

```

```

write(17,*) 'wp,gamma,epsinf,fn,wl,wt,gt before fitting : '
write(17,*) wp,gamma,epsinf,fn,wl,wt,gt
*****Fit the two-fluid-model to the data
* stop if the curve fits better than 0.00001 (absolute) per point:
chilim=0.00001
chilim=nfil*(chilim**2)
alamda=-1
flag=0
alamol=alamda
write(17,'(9(a10,1h ))') 'alamda','chisq','wp'
*      , 'gamma','epsinf','fn','wt','S','gt'
do 45 i=1,100
  call mrqmin(xx,yy,tol,nfil,a,dyda,ma,lista,mfit,
*      covar,alpha,beta,nca,chisq,ochisq,alamda,reflec)
  write(17,'(9(e10.4,1h ))') alamda,chisq,a(1),a(2)**2
*      ,a(3),1/(a(4)**2+1),a(5),(a(6)/a(5))**2,a(7)**2
  if (alamda.gt.alamol) then
    flag=flag+1
  else
    flag=0
  endif
  alamol=alamda
  if (chisq.lt.chilim) then
    write(17,*) 'Iteration interrupted. Reason: '
    write(17,*) chisq,' = chisq dropped below limit = ',chilim
    goto 46
  endif
  if (flag.gt.6) then
    write(17,*) 'Iteration interrupted. Reason: '
    write(17,*) alamda,' = alamda increased six times '
    goto 46
  endif
45 continue
write(17,*) 'Iteration interrupted. Reason: '
write(17,*) 'number of iterations equals ', i
46 alamda=0
call mrqmin(xx,yy,tol,nfil,a,dyda,ma,lista,mfit,
*      covar,alpha,beta,nca,chisq,ochisq,alamda,reflec)
wp=abs(a(1))
gamma=a(2)**2
epsinf=a(3)
fn=1/(a(4)**2+1)
wt=abs(a(5))
wl=abs(a(6))
gt=a(7)**2
write(17,*) 'wp,gamma,epsinf,fn,wt,S,gt after fitting : '
write(17,*) wp,gamma,epsinf,fn,wt,(wl/wt)**2,gt
*****fit is finished

***** Plot result between w0 and w1
do 250 x=w0,w1,dw
  call reflec(x,a,y,dyda,ma)
  write(*,*) x,y
250 continue
END
*****
*****

```

```

c      PROGRAM test
c      parameter(nfys=10000,ma=4)
c      integer nfil,ndat,ndrd,i,spol
c      real e1,e2,e3,e4,e5,pi,xx(nfys),yy(nfys)
c      * ,wdrd(nfys),rdrd(nfys)
c      * ,dl,a(4),rr0,rr1,dyda(4),sn2,ec
c      character*40 flin,flout
c      common/sp/spol
c      common/ang/sn2,ec
c      ec=4.
c      sn2=0.5
c      write(*,*) 's or p (1/0) ? '
c      read(*,*) spol
c      a(1)=800
c      a(2)=100
c      a(2)=sqrt(a(2))
c      a(3)=4
c      a(4)=0.5
c      call reflec(200.,a,rr0,dyda,ma)
c      dl=0.001
c      do 777 i=1,ma
c          a(i)=a(i)+dl
c          call reflec(200.,a,rr1,dyda,ma)
c          a(i)=a(i)-dl
c          write(*,*) i,rr0,rr1,(rr1-rr0)/dl,dyda(i)
c777  continue
c      END
*****

*****
      subroutine reflec(x,a,y,dyda,na)
      parameter(nfit=7)
      real den
      dimension a(na),dyda(na)
      integer i
      complex eps,est,dyde,de(nfit),n
      call epseud(x,a,na,eps,de,nfit)
      n=csqrt(eps)
      den=(cabs(1+n))**4
      est=real(eps)-(0.,1.)*aimag(eps)
      dyde=(est-1)/(n*den)
      y=(cabs((1-n)/(1+n)))**2
      do 10 i=1,nfit
          dyda(i)=real(2*dyde*de(i))
10      continue
      return
      end
*****

*****
      subroutine epseud(x,a,na,eps,de,nfit)
      real x,sn2,cs2,ec
      dimension a(na)
      integer i
      character*1 spol
      complex e,eps,de(nfit)
      common/sp/spol
      common/ang/sn2,ec

```

\*\*\*\*\*calculation of the pseudo dielectric function and derivatives, for  
 \*\*\*\*\*p- or s- polarized light. The dielectric constant perpendicular  
 \*\*\*\*\*to the surface is assumed to be fixed at a real value, wich is  
 \*\*\*\*\*introduced through the commonblock as 'ec'.

```

    call etfl(x,a,na,e,de,nfit)
    cs2=1.-sn2
    if (spol.eq.'s') then
      eps = (e-sn2)/cs2
      do 10 i=1,nfit
        de(i)=de(i)/cs2
10    continue
    else
      eps = (e*ec*cs2)/(ec-sn2)
      do 20 i=1,nfit
        de(i)=de(i)*eps/e
20    continue
    endif
    return
  end

```

\*\*\*\*\*

\*\*\*\*\*

```

  subroutine etfl(x,a,na,eps,de,nfit)
  real x,x2,wp,wp2,g,g2,epsinf,fn,pn,wt,wt2,wl,wl2
  dimension a(na)
  complex eps,denom,denom2,de(nfit),ci,fac,fac2
  ci=cplx(0.,1.)
  wp=a(1)
  g=a(2)
  epsinf=a(3)
  pn=a(4)
  fn=1/(pn**2+1)
  wp2=wp**2
  g2=g**2
  denom=(x+ci*g2)
  denom2=denom**2
  x2=x**2
  eps=epsinf-wp2*(fn/(x*denom)+(1-fn)/x2)
  de(1)=-2*wp*(fn/(x*denom)+(1-fn)/(x2))
  de(2)=2*ci*g*wp2*fn/(x*denom2)
  de(3)=1.
  de(4)=wp2*(1/(x**2)-1/(x*denom))
  de(4)=de(4)*(-2*a(4))*(fn**2)

  wt=a(5)
  wt2=wt**2
  wl=a(6)
  wl2=w1**2
  g=a(7)
  g2=g**2
  fac=(1.,0.)/(wt2-x2-ci*g2*x)
  fac2=fac**2
  eps=eps+wl2*fac
  de(5)=-2*wt*wl2*fac2
  de(6)= 2*w1*fac
  de(7)= 2*ci*g*x*wl2*fac2
  return

```

end

\*\*\*\*\*

\*\*\*\*\*

```

SUBROUTINE MRQMIN(X,Y,SIG,NDATA,A,DYDA,MA,LISTA,MFIT,
*   COVAR,ALPHA,beta,NCA,CHISQ,ochisq,ALAMDA,myfunc)
PARAMETER (MMAX=20)
REAL X(NDATA),Y(NDATA),SIG(NDATA),A(MA),DYDA(MA),
*   COVAR(NCA,NCA),ALPHA(NCA,NCA),ATRY(MMAX),BETA(MA),DA(MMAX)
INTEGER LISTA(MA)
integer kk,j,ihit,k
external myfunc
IF(ALAMDA.LT.0.)THEN
  KK=MFIT+1
  DO 12 J=1,MA
    IHIT=0
    DO 11 K=1,MFIT
      IF(LISTA(K).EQ.J)IHIT=IHIT+1
11    CONTINUE
      IF (IHIT.EQ.0) THEN
        LISTA(KK)=J
        KK=KK+1
      ELSE IF (IHIT.GT.1) THEN
        PAUSE 'Improper permutation in LISTA'
      ENDIF
12    CONTINUE
      IF (KK.NE.(MA+1)) PAUSE 'Improper permutation in LISTA'
      ALAMDA=0.001
      CALL MRQCOF(X,Y,SIG,NDATA,A,MA,LISTA,MFIT,ALPHA,BETA,NCA,
*        CHISQ,dyda,myfunc)
      OCHISQ=CHISQ
      DO 13 J=1,MA
        ATRY(J)=A(J)
13    CONTINUE
      ENDIF
      DO 15 J=1,MFIT
        DO 14 K=1,MFIT
          COVAR(J,K)=ALPHA(J,K)
14    CONTINUE
          COVAR(J,J)=ALPHA(J,J)*(1.+ALAMDA)
          DA(J)=BETA(J)
15    CONTINUE
          CALL GAUSSJ(COVAR,MFIT,NCA,DA,1,1)
          IF(ALAMDA.EQ.0.)THEN
            CALL COVSRT(COVAR,NCA,MA,LISTA,MFIT)
            RETURN
          ENDIF
          do 36 j=1,ma
            atry(j)=a(j)
36    continue
          DO 16 J=1,MFIT
```

```

    ATRY(LISTA(J))=A(LISTA(J))+DA(J)
16  CONTINUE
    CALL MRQCOF(X,Y,SIG,NDATA,ATRY,MA,LISTA,MFIT,COVAR,DA,NCA,
*   CHISQ,dyda,myfunc)
    IF(CHISQ.LT.OCHISQ)THEN
        ALAMDA=0.1*ALAMDA
        OCHISQ=CHISQ
        DO 18 J=1,MFIT
            DO 17 K=1,MFIT
                ALPHA(J,K)=COVAR(J,K)
17      CONTINUE
            BETA(J)=DA(J)
            A(LISTA(J))=ATRY(LISTA(J))
18      CONTINUE
    ELSE
        ALAMDA=10.*ALAMDA
        CHISQ=OCHISQ
    ENDIF
    RETURN
    END
*****

```

```

*****
SUBROUTINE MRQCOF(X,Y,SIG,NDATA,A,MA,LISTA,MFIT
* ,ALPHA,BETA,nalp,CHISQ,dyda,myfunc)
REAL X(NDATA),Y(NDATA),SIG(NDATA),ALPHA(NALP,NALP),BETA(MA),
*   A(MA),DYDA(MA)
INTEGER LISTA(MFIT)
integer j,k,i
real ymod,sig2i,dy,wt
external myfunc
DO 12 J=1,MFIT
    DO 11 K=1,J
        ALPHA(J,K)=0.
11      CONTINUE
        BETA(J)=0.
12      CONTINUE
    CHISQ=0.
    DO 15 I=1,NDATA
        CALL myfunc(X(I),A,YMOD,DYDA,MA)
        SIG2I=1./(SIG(I)*SIG(I))
        DY=Y(I)-YMOD
        DO 14 J=1,MFIT
            WT=DYDA(LISTA(J))*SIG2I
            DO 13 K=1,J
                ALPHA(J,K)=ALPHA(J,K)+WT*DYDA(LISTA(K))
13          CONTINUE
            BETA(J)=BETA(J)+DY*WT
14          CONTINUE
        CHISQ=CHISQ+DY*DY*SIG2I
15      CONTINUE
    DO 17 J=2,MFIT
        DO 16 K=1,J-1
            ALPHA(K,J)=ALPHA(J,K)
16          CONTINUE
17      CONTINUE
    RETURN

```

END

\*\*\*\*\*

\*\*\*\*\*

```
      SUBROUTINE COVSRT(COVAR,NCVM,MA,LISTA,MFIT)
      real COVAR(NCVM,NCVM)
      integer LISTA(MFIT)
      integer j,i
      real swap
      DO 12 J=1,MA-1
        DO 11 I=J+1,MA
          COVAR(I,J)=0.
11      CONTINUE
12     CONTINUE
      DO 14 I=1,MFIT-1
        DO 13 J=I+1,MFIT
          IF(LISTA(J).GT.LISTA(I)) THEN
            COVAR(LISTA(J),LISTA(I))=COVAR(I,J)
          ELSE
            COVAR(LISTA(I),LISTA(J))=COVAR(I,J)
          ENDIF
13      CONTINUE
14     CONTINUE
      SWAP=COVAR(1,1)
      DO 15 J=1,MA
        COVAR(1,J)=COVAR(J,J)
        COVAR(J,J)=0.
15     CONTINUE
      COVAR(LISTA(1),LISTA(1))=SWAP
      DO 16 J=2,MFIT
        COVAR(LISTA(J),LISTA(J))=COVAR(1,J)
16     CONTINUE
      DO 18 J=2,MA
        DO 17 I=1,J-1
          COVAR(I,J)=COVAR(J,I)
17     CONTINUE
18     CONTINUE
      RETURN
      END
```

\*\*\*\*\*

\*\*\*\*\*

```
      SUBROUTINE GAUSSJ(A,N,NP,B,M,MP)
      PARAMETER (NMAX=20)
      real A(NP,NP),B(NP,MP)
      integer IPIV(NMAX),INDXR(NMAX),INDXC(NMAX)
      integer j,i,k,irow,icol,l,ll
      real big,dum,pivinv
      DO 11 J=1,N
        IPIV(J)=0
11     CONTINUE
      DO 22 I=1,N
        BIG=0.
        DO 13 J=1,N
          IF(IPIV(J).NE.1)THEN
            DO 12 K=1,N
```



```

        IF (IPIV(K).EQ.0) THEN
            IF (ABS(A(J,K)).GE.BIG)THEN
                BIG=ABS(A(J,K))
                IROW=J
                ICOL=K
            ENDIF
        ELSE IF (IPIV(K).GT.1) THEN
            PAUSE 'Singular matrix'
        ENDIF
12     CONTINUE
    ENDIF
13     CONTINUE
    IPIV(ICOL)=IPIV(ICOL)+1
    IF (IROW.NE.ICOL) THEN
        DO 14 L=1,N
            DUM=A(IROW,L)
            A(IROW,L)=A(ICOL,L)
            A(ICOL,L)=DUM
14     CONTINUE
        DO 15 L=1,M
            DUM=B(IROW,L)
            B(IROW,L)=B(ICOL,L)
            B(ICOL,L)=DUM
15     CONTINUE
    ENDIF
    INDXR(I)=IROW
    INDXC(I)=ICOL
    IF (A(ICOL,ICOL).EQ.0.) PAUSE 'Singular matrix.'
    PIVINV=1./A(ICOL,ICOL)
    A(ICOL,ICOL)=1.
    DO 16 L=1,N
        A(ICOL,L)=A(ICOL,L)*PIVINV
16     CONTINUE
    DO 17 L=1,M
        B(ICOL,L)=B(ICOL,L)*PIVINV
17     CONTINUE
    DO 21 LL=1,N
        IF(LL.NE.ICOL)THEN
            DUM=A(LL,ICOL)
            A(LL,ICOL)=0.
            DO 18 L=1,N
                A(LL,L)=A(LL,L)-A(ICOL,L)*DUM
18     CONTINUE
            DO 19 L=1,M
                B(LL,L)=B(LL,L)-B(ICOL,L)*DUM
19     CONTINUE
        ENDIF
    CONTINUE
21     CONTINUE
22     CONTINUE
    DO 24 L=N,1,-1
        IF (INDXR(L).NE.INDXC(L)) THEN
            DO 23 K=1,N
                DUM=A(K,INDXR(L))
                A(K,INDXR(L))=A(K,INDXC(L))
                A(K,INDXC(L))=DUM
23     CONTINUE
            ENDIF
24     CONTINUE

```

RETURN  
END

\*\*\*\*\*