DIAGRAMMES VOLUME 69+70 , 2013

Construction du complexe de Farey L'algorithme et son code

Saab Abou-Jaoudé
ancien professeur de mathématiques spéciales
docteur d'état en mathématique
abouanpi@gmail.com

A.M.S. Sub. Class : 05 A 04 , 52 A 02 , 52 C 04

Résumé :

L'objectif de ce texte est de construire les composantes connexes du complexe de Farey plan, que nous avons appelé connexes de Farey. Nous avons démontré qu'ils sont des triangles ou des quadrilatères. Nous donnons le code fortran qui permet de produire les connexes de Farey. Le code que nous avons produit est optimal en un sens que nous précisons.

Mots clé :

Complexe de Farey, Connexe de Farey, Polygone convexe, Convexe polygonal direct, Droite orientée, Demi-plan.

I Introduction

Dans le premier article de ce numéro de Diagrammes, nous avons mis en évidence la forme des connexes de Farey. Nous avons, en fin d'article, décrit un algorithme qui permet de construire les connexes de Farey du complexe $CF(m+1,n)$ Ãă partir de ceux du complexe $CF(m,n)$. Nous avons dit et démontré que, au cours des constructions successives, les connexes transitoires sont toujours des triangles ou des quadrilatères. Le programme dont le code est ci-aprés, infirme ce résultat. Après analyse, nous nous sommes apercu que nous avons traité un des cas un peu trop vite et que l'ordre de parcours proposé ne permet pas de contrôler le nombre maximum d'arêtes d'un connexe de farey des étapes transitoires. Nous avons apporté une modification à l'ordre de parcours des droites et nous démontrons qu'avec ce nouveau mode de parcours, le nombre maximum d'arêtes d'une facette est 6. Nous exposons tout cela dans ce qui suit, en rappelant d'abord la méthodologie et la structure des données du programme codé en fortran 90, le langage de prédilection du calcul scientifique.

II De $CF(m,n)$ à $CF(m+1,n)$.

Le but de cette section est de rappeler et de corriger le théorème dont la conséquence est un algorithme efficace pour calculer les composantes

connexes d'un complexe de Farey d'ordre $(m+1, n)$ à partir de celles d'ordre (m, n).

D'abord quelques rappels. On est dans le plan réel affine muni d'un repère (O, i, j). :

- Une droite de Farey d'ordre (m, n) a une équation de la forme : $ux + vy = w$, avec u, v, w entiers, $|u| \leqslant m$, $|v| \leqslant n$.
- On note $D_{m,n}$ l'ensemble des droites de Farey d'ordre (m, n) qui rencontrent le carré unité.
- On note $CF(m, n)$ le complémentaire de $D_{m,n}$ dans le carré unité et on l'appelle *le complexe de Farey* d'ordre (m, n). Ses composantes connexes seront appelées *facettes*. On se conforme ainsi à la nomenclature introduite par la théorie des graphes planaires.
- On appelle *sommet d'ordre* (m, n) *(de Farey)* tout point d'intersection de deux droites de Farey, et situé dans le carré unité.
- On appelle segment de Farey ou *arête* tout segment dont les extrémités sont deux sommets consécutifs d'une droite de Farey
- On qualifiera de *négative* (resp. *positive*) une droite de pente < 0 (resp. > 0). On dira qu'elle est de type N (resp. de type P)
- Le qualificatif *horizontale* (resp. *verticale*) est attribué à une droite dont une équation est de la forme $vy = w$ (resp. $ux = w$). On dira qu'elle est de type H (resp. V).
- On dira qu'une droite est *oblique* si elle est soit positive soit négative.

Enfin, on se permettra de typer une arête par le type de la droite de Farey qui la contient.

Ceci posé, nous avons déjà mis en évidence la forme des composantes connexes de $CF(m, n)$.

Soit K une telle composante connexe, qu'on appellera "facette" pour faire court. Alors la frontière de K, notée $Fr(K)$, est un triangle ou un quadrilatère.

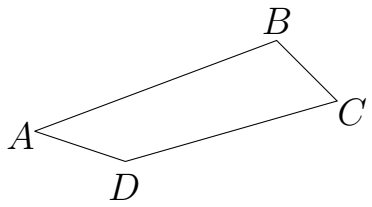
- Si c'est un triangle, ses trois côtés ne peuvent être du même type.
- Si c'est un quadrilatère on peut nommer ses sommets A, B, C, D de sorte que les deux arêtes opposés AB et DC soient de type P

(positives), les deux autres AD et BC étant de type N (négatives). Nous savons de plus que si K a l'un des côtés de sa frontière de type H (horizontal) ou de type V (vertical), alors c'est un triangle et les deux autres côtés sont obliques et de type opposés (N puis P ou P puis N).

Encore quelques notations : l'ensemble $D_{m+1,n} - D_{m,n}$ des droites d'équation $(m+1)x+vy = w$ sera noté DD, le sous-ensemble des droites négatives (resp. positives, resp. verticales) de DD sera noté DN (resp. DP, resp. DV). Notons qu'il n'y a pas dans DD de droites horizontales.

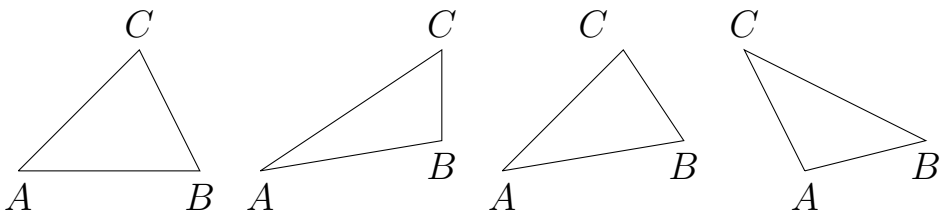
Nous allons typer les facettes par la liste des types de leurs arêtes successives, parcourues dans le sens direct. Modulo le choix de l'arête de départ, les facettes de $CF(m,n)$, ainsi que celles de $CF(m+1,n)$, sont :

• soit des quadrilatères et elles sont du type $PNPN$.



type quadrilatére $PNPN$

• soit des triangles et elles sont de l'un des types : VPN, HNP, NNP, PPN.



type triangle HNP, PVP, PNP, PNN

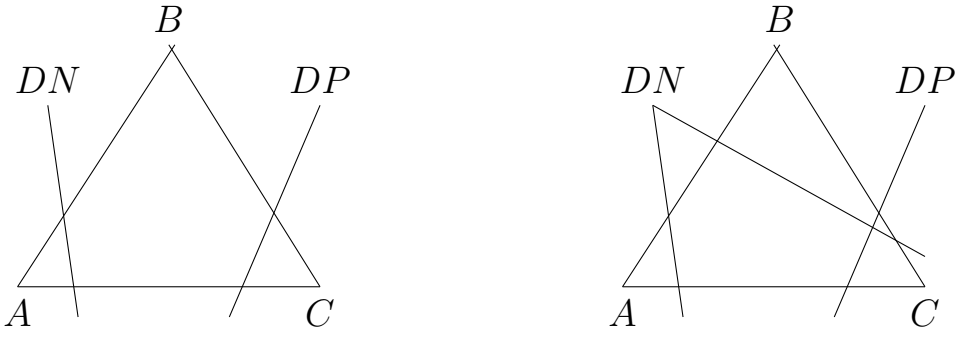
On numérote les droites de DD, de 1 jusqu'à $p = card(DD)$ en numérotant d'abord les droites de DV, puis celles de DN, puis celles de DP [1]. On va construire une suite de complexes $CF_k(m,n)$ avec $CF_0(m,n) = CF(m,n)$. ayant construit $CF_k(m,n)$, pour $k < p$, on construit $CF_{k+1}(m,n)$ en coupant, quand cela est possible, les facettes de $CF_k(m,n)$ par la droite de DD de numéro $k+1$. On obtient, une fois épuisées les droites de DD, $CF_p(m,n) = CF(m+1,n)$.

Nous rappelons le théorème de complexité spatiale suivant que Nous avons énoncé et démontré dans le premier article de ce numÃl'ro :

Théorème 1

Si on construit progressivement $CF(m+1,n)$ à partir de $CF(m,n)$ en coupant par les droites de DV, puis par celles de DN et enfin par celles de DP, la configuration du complexe obtenu à chaque étape n'admet pour composantes connexes que des triangles ou des quadrilatères.

Ce théorème est faux. La figure ci-dessous en témoigne.



1. cet ordre n'est pas le bon. le bon est DN puis DP puis DV

Facette ABC coupé par les droites DN et DP

Cette figure montre un pentagone produit en cours de parcours. ABC est une facette dont le côté AC est horizontal. DN en est une séquente négative et DP en est une sécante positive.

Voici l'énoncé correct, moyennant le bon ordre de numérotation des droites.

Théorème 2

Si on construit progressivement $CF(m+1,n)$ à partir de $CF(m,n)$ en coupant par les droites de DN puis par celles de DP et enfin par celles de DV, la configuration du complexe obtenu à chaque étape admet pour composantes connexes des polygônes de six côtés au plus.

Démonstration : La démonstration est fondée sur la remarque suivante : Si deux droites de DN se coupent Ãă l'intérieur du carré unité, leur point d'intersection est un sommet d'ordre (m,n). Nous laissons au lecteur le soin de terminer la démonstration.

Remarque : en fait, la limite de "six" indiquée dans le théorème ci-dessus peut se réduire à "cinq". Pourquoi ?

III Le code

1. Les utilitaires : parcours.f90

module DroiteFarey

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!
!! Une droite coupe les facettes d'un complexe de Farey.
!! Toutes les variables sont entières.
!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
implicit none
integer*4, parameter, private : : nafmax=10
! répertoire de travail et paramètre de taille maxi.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
character (len=len("C :/farey/")), parameter : : dir="C :/farey/"
character (len=len("fr")), parameter : : nom="fr"
integer*4, parameter, public : : mmax=40, nmax=40
integer*4, parameter, private : : nxmax = 2*mmax*nmax*(mmax+nmax)/3
integer*4, parameter, private : : nymax = nxmax
integer*4, parameter, public : : namax = nxmax*5
integer*4, parameter, private : : nsmax = namax/2
integer*4, parameter, private : : nfmax = 2*namax/3
integer*4, parameter, private : : opt = 1
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! le rectangle de travail rect =
!! CoiN Bas Gauche, CoiN Bas Droit,
!! CoiN Haut Gauche, CoiN Haut Droit
!! ici, init à (0,0,2), (1,0,2), (0,1,2), (1,1,2)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, dimension(1 :3), private : : CNBG, CNBD, CNHG, CNHD
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! les bords bas positif BBP, haut négatif BHN
!! gauche positif BGP, gauche négatif BGN
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, parameter, private : : BBP=1, BHN=2, BGP=3, BGN=4
integer*4, dimension(0 :nxmax), private : : BordBP,BordHN
integer*4, dimension(0 :nymax), private : : BordG,BordD
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! ns = nombre de sommets,
!! na = nombre d'arêtes,
!! nf = nombre de facettes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, public : : NS,NA,NF,cnt
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! le "type" arêtes :
!! FG, FD = Facette Gauche, Facette Droite
!! SD, SF = Sommet Début, Sommet Fin
!! AV, AP = arete AVant, arete APres
!! (sur le support orienté)
!! Sare = taille d'une arête.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, parameter, private : : FG=1,FD=2,SD=3,SF=4,AV=5,AP=6
integer*4, parameter, private : : Sare=6
integer*4, dimension(1 :Sare,1 :namax), private : : are
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! le "type sommet" (int,int,int)
!! Ssom = Nombre de coordonnées d'un sommet
!! Ici 3. On calcule en coordonnées homogènes entières
!! avec la 3eme coordonnée strictement positive.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, parameter, private : : Ssom=3
integer*4, dimension(1 :Ssom,1 :nsmax), private : : som
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! type facette. En 0, le nombre d'arêtes
!! suivi de la liste des arêtes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, parameter, private : : Sfac=nafmax, Sfacc=Sfac+2
integer*4, dimension(0 :Sfac,1 :nfmax), private : : fac
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! facettes courantes de travail FCUR et FCUR1
!! nombre d'arête de FCUR dans FCUR(0)
!! idem pour FCUR1
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4, dimension(0 :Sfacc), private : : FCUR,FCUR1
integer*4, private : : u, v, w, nav
```

```
  integer*8, private : : counter=0
  integer*4, parameter, private : : maxch=32
  character(len=maxch), private : : chmax=""
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! liste des fonctions et sous-programme publics
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  public : : Initdata, affiche
  public : : DroiteV, DroiteN,DroiteH,DroiteP
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! liste des fonctions et sous-programme privés.
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  private : : scal, intersec, facettecommune
  private : : RemplaceArete, CouperFacette
  private : : ASF, sgn, RAD, Sortie
  private : : SortieSommet, SortieArete
  private : : VaGauche, VaDroite
  private : : CoupeBord, cvis, gcd
```

contains

integer*4 FUNCTION gcd(x,y)

```
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! renvoie le PGCD des entiers "x" et "y"
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  integer*4, intent(in) : : x,y
  integer*4 : : a,b,c, xx, yy
      do w = 1,v-1 ; call droiteH(0, v, w) ; print*," H ",cnt ; end do
      if (x<0) then ; xx = -x ; else ; xx = x ; end if
      if (y<0) then ; yy = -y ; else ; yy = y ; end if
      if (xx<yy) then ; b=xx ; a=yy ; else ; b=yy ; a=xx ; end if
      do while (b/=0) ; c=mod(a,b) ; a=b ; b=c ; end do
  gcd=a
```

end FUNCTION gcd

```
logical FUNCTION check(u,v,w)
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 !! teste si "(u,v,w)" sont premiers entre eux
 !! renvoie ".FALSE." s'ils le sont
 !! et ".TRUE." sinon (éviter un ".NOT.")
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 integer*4, intent(in) : : u,v,w
 integer*4 : : x
     x = gcd(u,v) ; cnt = 4
     if (x .EQ. 1) then
         check=.FALSE.
     else
         if ( gcd(x,w) .EQ. 1) then
             check=.FALSE.
         else
             check=.TRUE.
         end if
     end if
end FUNCTION check

integer*4 FUNCTION scal(P)
 integer*4, intent(in), dimension(1 :3) : : P
     scal=u*P(1)+v*P(2)-w*P(3)
end FUNCTION scal

SUBROUTINE DroiteV(a,b,c)
 integer*4, intent(in) : : a,b,c
     if (check(a,b,c)) then ; return ; end if
     u=a ; v=b ; w=c
     if (scal(CNBD).LE.0) then ; return ; endif
     if (scal(CNBG).GE.0) then ; return ; endif
     nav=0 ; call CoupeBord(BordBP,BBP)
end SUBROUTINE DroiteV
```

SUBROUTINE DroiteH(a,b,c)

```
integer*4, intent(in) : : a,b,c
    if (check(a,b,c)) then; return; end if
    u=a; v=b; w=c
    if (scal(CNHG).LE.0) then; return; endif
    if (scal(CNBG).GE.0) then; return; endif
    nav=0; call CoupeBord(BordG,BGN)
```

end SUBROUTINE DroiteH

SUBROUTINE DroiteN(a,b,c)

```
integer*4, intent(in) : : a,b,c
    if (check(a,b,c)) then; return; end if
    u=a; v=b; w=c
    if(scal(CNHD).LE.0) then; return; endif
    if (scal(CNHG).LE.0) then
        nav=0; call CoupeBord(BordHN,BHN)
        return
    endif
    if (scal(CNBG).GE.0) then; return; endif
    nav=0; call CoupeBord(BordG,BGN)
```

end SUBROUTINE DroiteN

SUBROUTINE DroiteP(a,b,c)

```
integer*4, intent(in) : : a,b,c
    if (check(a,b,c)) then; return; end if
    u=a; v=b; w=c
    if (scal(CNBD).LE.0) then; return; end if
    if (scal(CNBG).LE.0) then
        nav=0; call CoupeBord(BordBP,BBP)
        return
    end if
    if (scal(CNHG).GE.0) then; return; end if
    nav=0; call CoupeBord(BordG,BGP)
```

end SUBROUTINE DroiteP

```
FUNCTION intersec(lc,ld,C,D) result(X)
  integer*4, intent(in) : : lc,ld
  integer*4, intent(in), dimension(1 :Ssom) : : C,D
  integer*4, dimension(1 :Ssom) : : X
  integer*4 : : dd,jc,jd
     if (ld>0) then
        jd=ld ; jc=lc
        else
        jd=-ld ; jc=-lc
     end if
     X=jd*C-jc*D
     if (X(3)/=0) then
        dd=gcd(gcd(X(1),X(2)),X(3))
        if (dd>1) then ; X=X/dd ; end if
     else
  print*,"X=", X,"u,v,w=",u,v,w
  print*,"droites parallèles, jc,jd=",jc,jd,"C,D=",C,D
  X=(/0,0,1/)
  stop
     end if
     if((X(1)<0) .OR. ((X(2)<0))) then
  print*, ("bug dans intersec")
  print*,"X=", X," u = ",trim(cvis(u))," v = ",trim(cvis(v))," w = ",trim(cvis(w))
  print*,"jc,jd=",jc,jd,"C,D=",C,D
  X=(/0,0,1/)
  stop
     end if
end FUNCTION intersec
```

```
FUNCTION facettecommune(x,y) result(k)
  integer*4, intent(in) : : x,y
  integer*4 : : ia, k
  integer*4, dimension(1 :Sare) : : A,B
     if ((x==0).OR.(y==0)) then
        k=0 ;
     else
        A=Are( :,x) ; B=Are( :,y) ; ia=A(FD) ; k=A(FG)
        if ((ia/=0) .and. ((ia==B(FD)) .or. (ia==B(FG)))) then ; k=ia ; end if
     end if
end FUNCTION facettecommune

integer*4 FUNCTION sgn(F)
  integer*4, intent(in), dimension(0 :Sfac) : : F
  integer*4 : : ja,jb,jc,jd
  integer*4, dimension(1 :Sare) : : A,B
  integer*4 : : C,D
     ja=F(1) ; A=Are( :,iabs(ja))
     jb=F(F(0)) ; B=Are( :,iabs(jb))
     if (ja>0) then ; jc=A(SF) ; else ; jc=A(SD) ; end if
     if (jb>0) then ; jd=B(SD) ; else ; jd=B(SF) ; end if
     C=scal(Som( :,jc)) ; D=scal(Som( :,jd))
     if((C==0).OR.(D==0)) then
        sgn=0
     else
        if (C>0) then ; sgn=D ; else ; sgn=-D ; end if
     end if
end FUNCTION sgn
```

```
FUNCTION ASF(s,FF) result(F)
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 !! Accés Sommet Facette. Sommet S, facette F
 !! recherche de la facette F trversée par la droite
 !! passant par le sommet S
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 integer*4, intent(in), dimension(0 :Sfac) : : FF
 integer*4, dimension(0 :Sfac) : : F
 integer*4, intent(in) : : s
 integer*4 : : i,j,kd,kf,L
 integer*4, dimension(1 :Sare) : : A
    L=FF(0)
       do i=1,L ; j=FF(i) ; A=Are( :,iabs(j))
          if (((j<0) .and. (s==A(SF))) .or.((j>0) .and. (s==A(SD)))) then ; exit ; end if
       end do
    if (i .LE. L) then
       if (i>1) then
          F=(/L,(FF(j),j=i,L),(FF(j),j=1,i-1)/)
       else
          F=FF
       end if
    else ! tilt. Gros bug
 print*, "bug dans ASF", s,FF
 do j=1,L
    i=FF(j) ; A=Are( :,iabs(i))
    print*,"i,ns,A", i,s,Som(1 :3,s)
    kd=A(3) ; kf=A(4)
 print*, kd,kf
 print*,"somj",som(1 :3,kd),"somk",som(1 :3,kf),"......."
 end do
 stop
    end if
end FUNCTION ASF
```

```
FUNCTION CouperFacette() result(T)
 integer*4, dimension(1 :4) : : T
 integer*4 : : ja,jb,jc,jd=0,i,j,L,ar=0,k1,n1,n2,s3,s4=0
 integer*4, dimension (1 :Ssom) : : C,D
 integer*4, dimension(1 :Sare) : : B,A1,A2
 integer*4, dimension(0 :Sfacc) : : F
    L=FCUR(0) ; ar=FCUR(1)
    if (ar>0) then ; B=Are( :, ar) ; jb=SF ; k1=B(FD)
    else ; B=Are( :,-ar) ; jb=SD ; k1=B(FG)
    end if
    s4=B(jb) ; D=Som( :,s4) ; jd=u*D(1)+v*D(2)-w*D(3) ;
    do i=2,L ; ar=FCUR(i) ; s3=s4 ; C=D ; jc=jd
       if (ar>0) then ; B=Are( :, ar) ; ja=SD ; jb=SF ; k1=B(FD)
       else ; B=Are( :,-ar) ; ja=SF ; jb=SD ; k1=B(FG)
       end if
       s4=B(jb) ; D=Som( :,s4) ; jd=u*D(1)+v*D(2)-w*D(3) ;
       if (jd==0) then ; T=(/0,i,s4,0/) ; return ; end if
       if (jc<0) then if(jd>0) then ; exit ; end if
       else if(jd<0) then ; exit ; end if
       end if
    end do
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 !! sortie arête.
 !!
 !! créer un nouveau sommet et deux nouvelles arêtes et les insérer dans A
 !! mettre à jour le nouveau sommet. mettre à jour le chainage arête.
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    NS=NS+1 ; Som( :,NS)=Intersec(jc,jd,C,D) ! créer un nouveu sommet.
    A1=B ; A2=B ! créer deux nouvelles arêtes
    n1=iabs(ar) ; ja=A1(AV) ; if (ja/=0) then ; Are(AP,ja)=n1 ; end if
    n2=NA+2 ; ja=A2(AP) ; if (ja/=0) then ; Are(AV,ja)=n2 ; end if
    A1(AP)=n2 ; A2(AV)=n1 ; A1(SF)=NS ; A2(SD)=NS ;
    Are( :,n1)=A1 ; Are( :,n2)=A2 ; F=FCUR
    if (ar>0) then
       FCUR=(/L+1,(F(j),j=1,i-1),n1,n2,(F(j),j=i+1,L),0/)
       if (k1/=0) then ; call RAD(-ar,-n2,-n1,k1) ; end if
    else
       FCUR=(/L+1,(F(j),j=1,i-1),-n2,-n1,(F(j),j=i+1,L),0/)
       if (k1/=0) then ; call RAD(-ar,n1,n2,k1) ; end if
    end if ;
    T=(/1,i,NS,k1/)
end FUNCTION CouperFacette
```

```
integer*4 FUNCTION Sortie(OldFac)
  integer*4, intent(in) : : OldFac
  integer*4, dimension(1 :4) : : T
  integer*4, dimension(1 :Sare) : : B, C
  integer*4 : : i, j, k, L, sk, NewAre, NewFac
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! T=(ar,i,ns,k1). sortie entre arete A(i) et A(i+1) au sommet ns
  !! Si ar=0 alors sortie sommet ns avec k1=0
  !! Si ar=1 sortie arête déjà coupé (i) au sommet ns vers la facette k1
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     T=CouperFacette()
     NA=NA+1 ; NewAre=NA ; NA=NA+T(1)
     NF=NF+1 ; NewFac=NF
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! créer la nouvelle arête en récupérant sommet debut de A(1). Le sommet fin est ns
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     j=FCUR(1) ; i=T(2)
     if (j>0) then ; B=Are( :,j) ; sk=B(SD) ;
     else ; B=Are( :,-j) ; sk=B(SF)
     end if
     C=(/OldFac,NewFac,sk,T(3),NAV,0/) ; ! très moche mais efficace.
     Are( :,NewAre)=C
     if (NAV/=0) then ; Are(AP,NAV)=NewAre ; end if
     NAV=NewAre
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! remplir les contenus de Fac(NewFac) et Fac(OldFac)
  !! Ne pas oublier les mises à jour des Are(i) à partir de A
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     L=FCUR(0)
     do k=1,i
        j=FCUR(k)
        if (j>0) then ; Are(FG,j)=NewFac ; else ; Are(FD,-j)=NewFac ; end if
     end do
     Fac( :,NewFac)=(/i+1 ,(FCUR(k), k=1,i), -NewAre,0,0/)
     Fac( :,OldFac)=(/L-i+1, NewAre,(FCUR(k),k=i+1,L),0,0/)
     cnt = max(cnt,i+1,l-i+1)
     if (T(1)==0) then
        Sortie=SortieSommet(T(3),FCUR(i),FCUR(i+1))
     else
        Sortie=SortieArete(T(4))
     end if
end FUNCTION Sortie
```

```
integer*4 FUNCTION SortieSommet(s,ad,ag)
  integer*4, intent(in) : : s, ad, ag
  integer*4 : : i, j, kn
  integer*4, dimension(1 :Sare) : : A
     if (ad<0) then; kn=AV; else; kn=AP; end if
     A=Are( :,iabs(ad)); j=A(kn)
     if (ag>0) then; kn=AV; else; kn=AP; end if
     A=Are( :,iabs(ag)); i=A(kn)
     kn=FacetteCommune(i,j)
     if (kn>0) then; FCUR(0 :sfac)=ASF(s,Fac( :,kn)); end if
     SortieSommet=kn
end FUNCTION SortieSommet

integer*4 FUNCTION SortieArete(k)
  integer*4, intent(in) : : k
  integer*4 : : nb,y,z
     if (k/=0) then
        FCUR=FCUR1
     else
        y=Som(2,NS); z=Som(3,NS)
        if (y*CNBG(3)-z*CNBG(2)==0) then
           nb=BordBP(0); nb=nb+1
           BordBP(nb)=NA; BordBP(0)=nb
        else if (y*CNHG(3)-z*CNHG(2)==0) then
           nb=BordHN(0); nb=nb+1
           BordHN(nb)=NA; BordHN(0)=nb
        else
           nb=BordD(0); nb=nb+1
           BordD(nb)=NA; BordD(0)=nb
        end if
     end if
     SortieArete=k
end FUNCTION SortieArete
```

```
integer*4 FUNCTION VaGauche(ia,ka)
  integer*4, intent(in) : : ia,ka
  integer*4 : : i,ib,k
  integer*4, dimension(1 :Sare) : : A
  integer*4, dimension(0 :Sfac) : : F
     k=ka; F=Fac( :,k) ; ib=F(0)
     do
        if (ia==0) then
  print*, "... va gauche",u,v,w,ka
  stop
        end if
        F=ASF(ia,F)
        if (sgn(F)<0) then ; FCUR(0 :sfac)=F ; exit ; end if
        i=F(ib) ; A=Are( :,iabs(i)) ;
        if (i>0) then ; k=A(FD) ; else ; k=A(FG) ; end if
        if (k>0) then
           F=Fac( :,k) ; ib=F(0)
        else
  print*, "bug dans VaGauche : ia = ",trim(cvis(ia))," ka = ",trim(cvis(ka))
  print*, "Droite",u,v,w
  print*,"facette ",F
  stop
        end if
     end do
     VaGauche=k
end FUNCTION VaGauche

SUBROUTINE RemplaceArete(ar,n1,n2,k)
  integer*4, intent(in) : : ar, n1, n2, k
  integer*4 : : i, j, L
  integer*4, dimension(0 :Sfac) : : F
     F=Fac( :,k) ; L=F(0)
     do i=1,L ; if (ar==F(i)) then ; exit ; end if ; end do
     FCUR=(/L+1,n2, (F(j),j=i+1,L), (F(j),j=1,i-1),n1,0/)
end SUBROUTINE RemplaceArete
```

```
integer*4 FUNCTION VaDroite(ia,ka)
  integer*4, intent(in) : : ia,ka
  integer*4 : : i,ib,k
  integer*4, dimension(1 :Sare) : : A
  integer*4, dimension(0 :Sfac) : : F
     k=ka; F=Fac( :,k) ; ib=F(0)
     do
        if (ia==0) then
  print*, "... va droite",u,v,w,ka
  stop
        end if
        F=ASF(ia,F)
        if (sgn(F)<0) then ; FCUR(0 :sfac)=F ; exit ; end if
        i=F(1) ; A=Are( :,iabs(i))
        if (i>0) then ; k=A(FD) ; else ; k=A(FG) ; end if
        if (k>0) then
           F=Fac( :,k) ; ib=F(0)
        else
  print*, "bug dans VaDroite : ia = ",trim(cvis(ia))," ka = ",trim(cvis(ka))
  print*, "Droite",u,v,w
  print*,"facette ",F
  stop
        end if
     end do
     VaDroite=k
end FUNCTION Vadroite

SUBROUTINE RAD(ar,n1,n2,k)
  integer*4, intent(in) : : ar, n1, n2, k
  integer*4 : : i, j, L
  integer*4, dimension(0 :Sfac) : : F
     F=Fac( :,k) ; L=F(0) ;
     do i=1,L ; if (ar==F(i)) then ; exit ; end if ; end do
     FCUR1=(/L+1,n2,(F(j),j=i+1,L),(F(j),j=1,i-1),n1,0/)
end SUBROUTINE RAD
```

SUBROUTINE CoupeBord(Bord,brd)

```
integer*4, intent(in) : : brd
integer*4, intent(inout), dimension(0 :) : : Bord
integer*4 : : nb,i,k,ia,ib,ja,jb,jd,jf,flg
integer*4, dimension(1 :Sare) : : A,AA
integer*4, dimension(1 :Ssom) : : B,C
    NAV=0 ; nb=Bord(0) ; flg=1
    do i=1,nb
        ia=Bord(i) ; A=Are( :,ia)
        ja=A(SD) ; B=Som( :,ja) ; jd=u*B(1)+v*B(2)-w*B(3)
        jb=A(SF) ; C=Som( :,jb) ; jf=u*C(1)+v*C(2)-w*C(3)
        if ((jd==0).OR.(jf==0)) then ; flg=0 ; exit ; end if
        if((jd>0.AND.jf<0).OR.(jd<0.AND.jf>0)) then ; flg=-1 ; exit ; end if
    end do ;
    if (flg>0) then
print*, "bug dans coupebord"
stop
    end if
    if (flg<0) then ! On traverse l'arête 'ia'.
        nb=nb+1 ; Bord(0)=nb
        NA=NA+1 ; Bord(nb)=NA
        NS=NS+1 ; Som( :,NS)=Intersec(jd,jf,B,C)
        AA=A ; A(SF)=NS ; AA(SD)= NS
        A(AP)=NA ; AA(AV)=ia ;
        ib=AA(AP) ; if (ib>0) then ; Are(AV,ib)=NA ; end if
        Are( :,NA)=AA ; Are( :,ia)=A ;
        if (brd==BBP) then ; k=A(FG) ; call RemplaceArete(ia,ia,NA,k)
        else ; k=A(FD) ; call RemplaceArete(-ia,-NA,-ia,k)
        end if
    else ! on a coupé le bord en un sommet
        if (jf==0) then ; ia=A(AP) ; A=Are( :,ia) ; ja=A(SD) ; end if
        if (brd==BBP) then ; k=A(FG) ; k=VaGauche(ja,k)
        else if (brd==BHN) then ; k=A(FD) ; k=VaDroite(ja,k)
        else if (brd==BGP) then ; k=A(FD) ; k=VaDroite(ja,k)
        else if (brd==BGN) then ; ia=A(AV) ; A=Are( :,ia) ; k=A(FD) ; k=VaGauche(ja,k)
        end if
    end if
    do while (k/=0)
```

```
        if (NA.GE.namax-2) then
  print*, "Débordement NA", NA
  print*," NS,NF", NS, NF
  print*, "droite ",u,v,w
  stop
        else; counter=counter+1 ; k=Sortie(k)
        end if
     end do
end SUBROUTINE CoupeBord

FUNCTION cvis(n) result(ch)
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !! renvoie une chaine de caractère
  !! "ch" représentant l'entier "n"
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  integer*4, intent(in) : : n
  character(len=maxch) : : ch
  character(len=10) : : digit="0123456789"
  integer*4 : : i,j,k,m
  ch=chmax ; k=abs(n)
  if (k==0) then ; m=2 ; else ; m=floor(log10(real(k)))+2 ; end if
  do i=m,2,-1 ; j=mod(k,10) ; k=k/10 ; ch(i :i)=digit(j+1 :j+1) ; end do
  if (n<0) then ; ch(1 :1)="-" ; else ; do i=1,m ; ch(i :i)=ch(i+1 :i+1) ; end do ; end if
end FUNCTION cvis
```

SUBROUTINE initdata

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! initialisation des données au carré
!! (0,0),(1,0),(0,1),(1,1). Le cas "m=1,n=1"
!! est complété en début du programme
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
ns=4 ; na=4 ; nf=1
Som=0 ; are=0 ; fac=0
CNBG=(/0,0,1/) ; CNHG=(/0,1,opt/)
CNBD=(/1,0,opt/) ; CNHD=(/1,1,opt/)
Som( :,1)=CNBG ; Som( :,2)=CNBD
Som( :,3)=CNHG ; Som( :,4)=CNHD
are( :,1)=(/1,0,1,2,0,0/) ;are( :,2)=(/1,0,2,4,0,0/)
are( :,3)=(/0,1,1,3,0,0/) ;are( :,4)=(/0,1,3,4,0,0/)
fac(0 :4,1)=(/4,1,2,-4,-3/)
BordBP(0)=1 ; BordHN(0)=1 ; BordG(0)=1 ; BordD(0)=1
BordBP(1)=1 ; BordHN(1)=4 ; BordG(1)=3 ; BordD(1)=2
```

end SUBROUTINE initdata

SUBROUTINE affiche

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!! affiche les données sur l'écran
!! et les écrit dans le fichier
!! "G :/SBN/bin/farey.txt"
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
integer*4 : : i,j
print*, ns,",",na,",",nf
do i=1,ns ; print*,i, (trim(cvis(som(j,i)))//" ",j=1,Ssom) ; end do
do i=1,na ; print*,i, (trim(cvis(are(j,i)))//" ",j=1,Sare) ; end do
do i=1,nf ; print*,i, (trim(cvis(fac(j,i)))//" ",j=0,fac(0,i)) ; end do
print*, "BordG", (trim(cvis(BordG(j)))//" ",j=0,BordG(0))
print*, "BordD", (trim(cvis(BordD(j)))//" ",j=0,BordD(0))
print*, "BordBP", (trim(cvis(BordBP(j)))//" ",j=0,BordBP(0))
print*, "BordHN", (trim(cvis(BordHN(j)))//" ",j=0,BordHN(0))
OPEN (UNIT=1,FILE='G :/SBN/bin/farey.txt',STATUS='REPLACE')
write(1,*) ns,",",na,",",nf
do i=1,ns ; write(1,*)i,",", ((trim(cvis(som(j,i)))//",",j=1,Ssom) ; end do
do i=1,na ; write(1,*)i,",", (trim(cvis(are(j,i)))//",",j=1,Sare) ; end do
do i=1,nf ; write(1,*)i, ",",(trim(cvis(fac(j,i)))//",",j=0,fac(0,i)) ; end do
write(1,*) "BordG ",",", (trim(cvis(BordG(j)))//",",j=0,BordG(0))
write(1,*) "BordD ",",", (trim(cvis(BordD(j)))//",",j=0,BordD(0))
write(1,*) "BordBP ",",", (trim(cvis(BordBP(j)))//",",j=0,BordBP(0))
write(1,*) "BordHN ",",", (trim(cvis(BordHN(j)))//",",j=0,BordHN(0))
close(1)
```

end SUBROUTINE affiche

end module droitefarey

2. Le programme : farey.f90

program fareymn

```
use DroiteFarey
integer*4 : : m,n,u,v,w
m = 3 ; n = 2
print*, "entrée programme"
    call initdata
    call droiteN(1,1,1)
    call droiteP(1,-1,0)
call affiche
print*," cas u <= 1"
    do v = 2,n
        do w = 1,v-1 ; call droiteH(0, v, w) ; print*," H ",cnt ; end do
        do w = 1,v ; call droiteN(1, v, w) ; print*," N ",cnt ; end do
        do w = -v+1,0 ; call droiteP(1,-v, w) ; print*," P ",cnt ; end do
    end do
print*," cas u>1 "
    do u = 2,m
        do w = 1,u-1 ; call droiteV(u, 0, w) ; print*," V ",cnt ; end do
        do v=1,n
            do w = 1,u+v-1 ; call droiteN(u, v, w) ; print*," N ",cnt ; end do
            do w = -v+1,u-1 ; call droiteP(u,-v, w) ; print*," P ",cnt ; end do
        end do     end do
! call affiche
print*, "m = ",m,"n = ",n
print*, "ns = ",ns, "na = ",na,"nf = ",nf
```

end program fareymn

∗ [2]

2. *La mauvaise version*

program fareymn

```
use DroiteFarey
integer*4 : : m,n,u,v,w
m = 3 ; n = 2
print*, "entrée programme"
    call initdata
    call droiteN(1,1,1)
    call droiteP(1,-1,0)
    call affiche
print*," cas u <= 1"
    do v = 2,n
        do w = 1,v ; call droiteN(1, v, w) ; print*," N ",cnt ; end do
        do w = -v+1,0 ; call droiteP(1,-v, w) ; print*," P ",cnt ; end do
        do w = 1,v-1 ; call droiteH(0, v, w) ; print*," H ",cnt ; end do
    end do
print*," cas u>1 "
    do u = 2,m
        do v = 1,n
            do w = 1,u+v-1 ; call droiteN(u, v, w) ; print*," N ",cnt ; end do
            do w = -v+1,u-1 ; call droiteP(u,-v, w) ; print*," P ",cnt ; end do
            do w = 1,u-1 ; call droiteV(u, 0, w) ; print*," V ",cnt ; end do
        end do
    end do
! call affiche
print*, "m = ",m,"n = ",n
print*, "ns = ",ns, "na = ",na,"nf = ",nf
```

end program fareymn

**[3]

3. *La bonne version*

Références :

[1] Saab Abou-Jaoudé, Forme des connexes de Farey. /arxiv.org/pdf/1312.4306.pdf (2013)

[2] Saab Abou-Jaoudé, Dénombrement de triplets d'entiers NDN.PDF (2014) diffusé sur Whaller.com et à l'université de Strasbourg.

[3] Malcolm Douglas McIlroy. A note on discrete representation of lines. ATT Technical Journal, 64(2) :481-490, 2984.

[4] Alain Daurat, Mohamed Tajine, Mahdi Zouaoui. About the frequencies of some patterns in digital planes. application to area estimators. Computer Graphics, 2008.