! Program to VORTICITY by Successive relaxation.

! Record of Revisions

! Date Programmer Description of change

! ===== ========== =====================

! 01/04/2021 Nithish Kumar Sriramoju Original Code

Program vorticity

implicit none

! intiating variables

integer :: i,j,l,m,u,v,noi

real :: beta

real :: V\_m,r\_m,b

real, dimension (:),allocatable:: x

real, dimension (:),allocatable:: y

real, dimension (:,:),allocatable :: r

real, dimension (:,:),allocatable :: zeta

real, dimension(:,:),allocatable:: psi

real, dimension(:,:),allocatable::g

real, dimension(:,:),allocatable::k

allocate(x(101))

allocate(y(101))

allocate(r(101,101))

allocate(zeta(101,101))

allocate(psi(101,101))

allocate(g(101,101))

allocate(k(101,101))

V\_m=40

r\_m=100

b = 1

noi=0

! Intial Zeroes in psi function

do i = 0,100

do j = 0,100

psi(i,j) = 0

end do

end do

! Finding Distance from origin

do i = 0,100

do j =0,100

r(i,j) = 20\*((i-50)\*\*2+(j-50)\*\*2)\*\*0.5 ! distance formula assuming origin at 50,50 index

end do

end do

! Deriving vorticity at all points

do i = 0,100

do j = 0,100

zeta(i,j) = 2\*(V\_m/(1000\*r\_m))\*(1-0.5\*((r(i,j)/r\_m)\*\*b))\*exp((1/b)\*(1-(r(i,j)/r\_m)\*\*b))

end do

end do

! Iterating PSI

do

noi=noi+1

do i =1,99

do j =1,100

g(i,j)=psi(i,j)

end do

end do

do i = 1,99

do j = 1,100

psi(i,j) = (g(i+1,j)+psi(i-1,j)+g(i,j+1)+psi(i,j-1) - zeta(i,j)\*((20\*1000)\*\*2))/4

end do

end do

do i =1,99

do j =1,100

k(i,j)=psi(i,j)

end do

end do

if ( k(2,2)-g(2,2) <= 0.000001) exit

end do

! Printing PSI values in a file

open(1,file = 'poisson.txt', status = 'unknown')

do i = 0,100

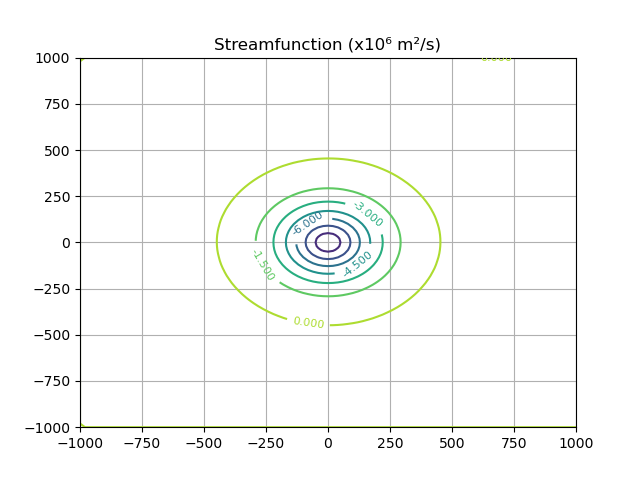
write(1,\*) ( psi(i,j), j=0,100 )

end do

print\*,noi

end program

**RESULT:**



! Program to VORTICITY by SOR method.

! Record of Revisions

! Date Programmer Description of change

! ===== ========== =====================

! 01/04/2021 Nithish Kumar Sriramoju Original Code

Program vorticity

implicit none

! intiating variables

integer :: i,j,l,m,u,v,noi

real :: beta

real :: V\_m,r\_m,b

real, dimension (:),allocatable:: x

real, dimension (:),allocatable:: y

real, dimension (:,:),allocatable :: r

real, dimension (:,:),allocatable :: zeta

real, dimension(:,:),allocatable:: psi

real, dimension(:,:),allocatable::g

real, dimension(:,:),allocatable::k

allocate(x(101))

allocate(y(101))

allocate(r(101,101))

allocate(zeta(101,101))

allocate(psi(101,101))

allocate(g(101,101))

allocate(k(101,101))

print\*,"enter the value of beta"

read(\*,\*)beta

V\_m=40

r\_m=100

b = 1

noi=0

! Intial Zeroes in psi function

do i = 0,100

do j = 0,100

psi(i,j) = 0

end do

end do

! Finding Distance from origin

do i = 0,100

do j =0,100

r(i,j) = 20\*((i-50)\*\*2+(j-50)\*\*2)\*\*0.5 ! distance formula assuming origin at 50,50 index

end do

end do

! Deriving vorticity at all points

do i = 0,100

do j = 0,100

zeta(i,j) = 2\*(V\_m/(1000\*r\_m))\*(1-0.5\*((r(i,j)/r\_m)\*\*b))\*exp((1/b)\*(1-(r(i,j)/r\_m)\*\*b))

end do

end do

! Iterating PSI

do

noi=noi+1

do i =1,99

do j =1,100

g(i,j)=psi(i,j)

end do

end do

do i = 1,99

do j = 1,100

psi(i,j) = beta\*(psi(i+1,j)+psi(i-1,j)+psi(i,j+1)+psi(i,j-1) - zeta(i,j)\*((20\*1000)\*\*2))/4 + (1-beta)\*g(i,j)

end do

end do

do i =1,99

do j =1,100

k(i,j)=psi(i,j)

end do

end do

if ( k(2,2)-g(2,2) <= 0.000001) exit

end do

! Printing PSI values in a file

open(1,file = 'poisson.txt', status = 'unknown')

do i = 0,100

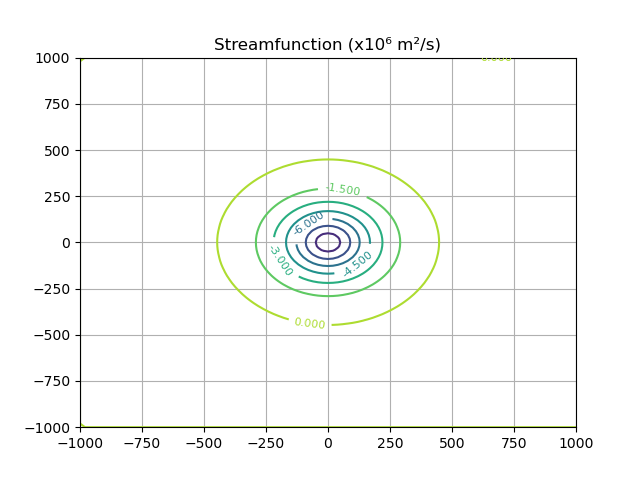
write(1,\*) ( psi(i,j), j=0,100 )

end do

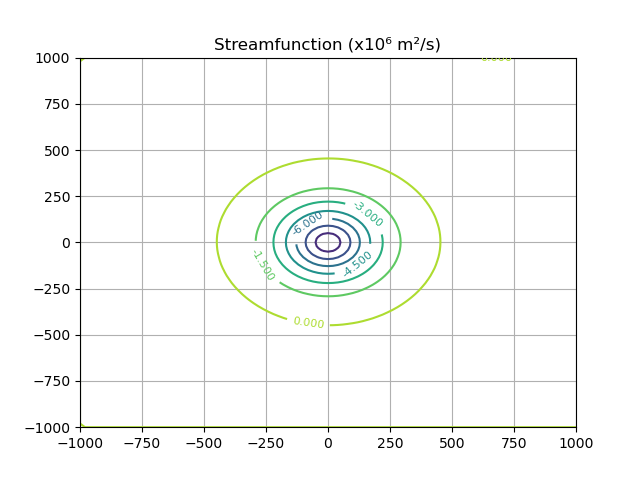
print\*,noi

end program

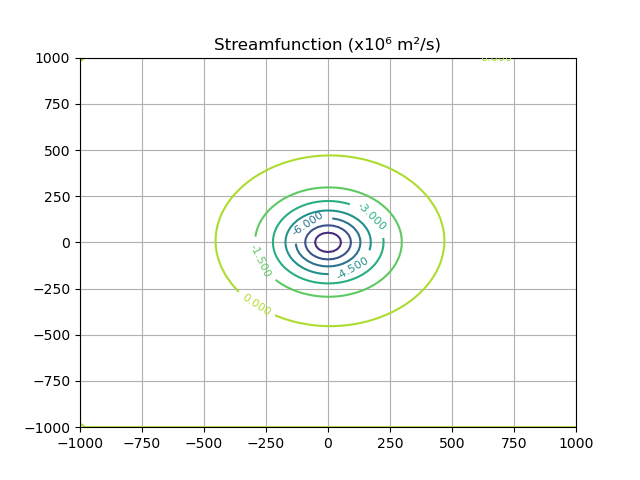
**RESULT: for beta = 0.5 steps taken 2195**



**RESULT: for beta = 1 steps taken 754**



**RESULT: for beta = 1.5 (optimum value) steps taken 274**



**RESULT: for beta = 2 steps taken 79**

