Solitony

Symulacje komputerowe

Piotr Szewerniak, Paula Chajduła
Kamil Oratowski, Marcin Caputa

Politechnika Krakowska
Wydział Inżynierii
Materiałowej i Fizyki

Adwekcja, wstrząsy i soliton Russela

Adwekcja – w mechanice płynów i hydrodynamice podziemnej unoszenie substancji przez przepływający płyn w ten sposób, że prędkość unoszonej substancji równa jest prędkości przepływającego płynu. Soliton - samopodtrzymująca się odosobniona fala wywołana przez efekty nieliniowe występujące w materiale, w którym fala ta się rozchodzi. Lub rozwiązanie układu nieliniowych równań różniczkowych, które:

- reprezentuje fale o niezmiennym kształcie;
- jest zlokalizowane tak, że zanika lub osiąga stałą wartość w nieskończoności;
- może oddziaływać silnie z innymi solitonami, ale po kolizji zachowuje niezmienioną formę – występuje tylko przesunięcie fazy.

Adwekcja, wstrząsy i soliton Russela

Russel postanowił wytworzyć takie fale w swoim laboratorium po czym doszedł do równania opisującego prędkość fali:

$$c^2 = g(h+A)$$

gdzie g to przyspieszenie ziemskie, h to głębokość a A to amplituda fali. Z tego wynika, że fale z większą amplitudą poruszają się szybciej niż te z małą.

Ciągłość i równania adwekcji

Ruch płynu opisany jest przy pomocy równania Naviera-Stokesa

$$\frac{\partial \rho(\mathbf{x},t)}{\partial t} + \vec{\nabla}\cdot \mathbf{j} = 0, \quad \mathbf{j} \stackrel{\text{def}}{=} \rho \mathbf{v}(\mathbf{x},t).$$

Ale dla przypadku jednowymiarowego otrzymujemy:

$$\frac{\partial \rho}{\partial t} + c\frac{\partial \rho}{\partial x} = 0.$$

Takie wyrażenie nazywamy równaniem adwekcji - czyli np. przypadku gdy tratwa na wodzie ma taką samą prędkość jak sam nurt.

Opis fal uderzeniowych przy pomocy równania Burgera

Równanie Burgera ma postać:

$$\frac{\partial u}{\partial t} + \epsilon u \frac{\partial u}{\partial x} = 0,$$

$$\frac{\partial u}{\partial t} + \epsilon \frac{\partial (u^2/2)}{\partial x} = 0,$$

Z czego druga postać równania jest "postacią konserwatywną"tj. jest wariacją na temat równania adwekcji Russela, w którym prędkość fali jest proporcjonalna do jej amplitudy.

Opis fal uderzeniowych przy pomocy równania Burgera

Równanie Burgera ma postać:

$$\frac{\partial u}{\partial t} + \epsilon u \frac{\partial u}{\partial x} = 0,$$

$$\frac{\partial u}{\partial t} + \epsilon \frac{\partial (u^2/2)}{\partial x} = 0,$$

Z czego druga postać równania jest "postacią konserwatywną"tj. jest wariacją na temat równania adwekcji Russela, w którym prędkość fali jest proporcjonalna do jej amplitudy.

Jak wziąć pod uwagę dyspersję?

Rozpraszanie się nie powoduje utraty energii fali ale powoduje zanik informacji o niej wraz z upływem czasu.
Załóżmy falę, którą opisuje równanie:

$$u(x,t) = e^{\pm i(kx-\omega t)}$$

Biorąc pod uwagę fakt, że fala rozchodzi się "w prawo" czyli kx-wt nie zmienia się wraz ze zwiększeniem się x w czasie. Po podstawieniu tej funkcji do równania adwekcji otrzymamy:

$$\omega = ck.$$

Jak wziąć pod uwagę dyspersję?

Jest to przykład relacji dyspersji - zależność częstotliwość fali w i wektora falowego k. Prędkość grupowa wyraża się wzorem:

$$v_g = \frac{\partial \omega}{\partial k},$$

relacja dyspersji prowadzi do tego, że dla każdej częstotliwości mamy taką samą prędkość grupową.

Załózmy, że fala rozchodzi się z małą dyspersją czyli z częstotliwością, która ma mniejszą niż liniową zależność od wektora falowego k

$$\omega \simeq ck - \beta k^3.$$

Jak wziąć pod uwagę dyspersję?

Z tego wszystkiego dochodzimy do wniosku, że możemy zapisać zależność:

$$v_g = \frac{d\omega}{dk} \simeq c - 3\beta k^2,$$

$$\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} + \beta\frac{\partial^3 u(x,t)}{\partial x^3} = 0.$$

Solitony na mieliźnie i równanie KdeV

Chcemy zrozumieć dziwne zachowanie fal wodnych, które mają miejsce na mieliźnie czy wąskich kanałach. Analityczy opis takich spiętrzeń wody zostało wprowadzone przez Korteweg'a i deVries'a:

$$\frac{\partial u(x,t)}{\partial t} + \varepsilon u(x,t)\frac{\partial u(x,t)}{\partial x} + \mu \frac{\partial^3 u(x,t)}{\partial x^3} = 0.$$

Nasz wyraz środkowy wyraz prowadzi do zaostrzenia się fali a w reazultacie do fali uderzeniowej.

Analityczne rozwiązanie solitonu

Zwykle rozwiązujemy takie równannia poprzez wstawienie odgadniętej formy rozwiązania:

$$u(x,t) = u(\xi = x - ct).$$

Takie wyrażenie oznacza, że jeśli będziemy poruszać się ze stałą prędkością c, zauważymy stałą postać fali. Podstawienie do równania KdeV daje nam rozwiązywalne równanie różniczkowe zwyczajne:

$$-c\frac{\partial u}{\partial \xi} + \epsilon\, u \frac{\partial u}{\partial \xi} + \mu \frac{d^3 u}{d\xi^3} = 0,$$

$$u(x,t) = \frac{-c}{2}\mathrm{sech}^2\left[\frac{1}{2}\sqrt{c}(x - ct - \xi_0)\right],$$

gdzie ξ jest fazą początkową. Zauważamy, że amplituda jest proporcjonalna do prędkośći - jest to typowa forma analityczna solitonu.

Implementacja rozwiązania solitonu KdeV

Zaczynamy od:

$$u(x, t = 0) = \frac{1}{2}\left[1 - \tanh\left(\frac{x - 25}{5}\right)\right],$$

z parametrami: ϵ = 0.2, μ = 0.1, Δx = 0.4, Δt = 0.1

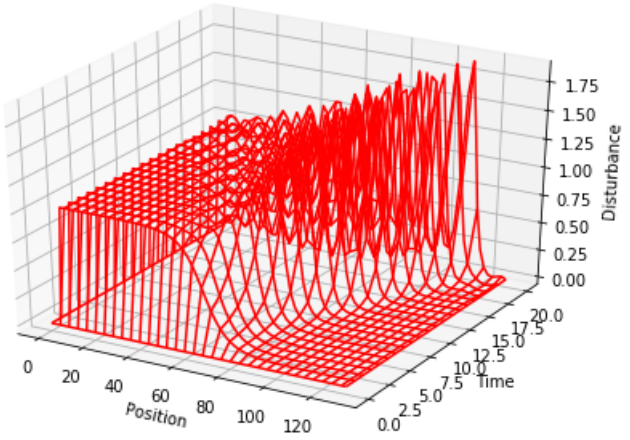
Implementacja rozwiązania solitonu KdeV

```python
import matplotlib.pylab as p
from mpl_toolkits.mplot3d import Axes3D
import numpy as np

ds = 0.4
dt = 0.1
max = 2000
mu = 0.1
eps = 0.2
mx = 131
u = np.zeros((mx,3), float)
spl = np.zeros((max, 21), float)
m = 1

for i in range(0, 131):
    u[i, 0] = 0.5 * ( 1 - ( ( np.exp( 2 * (0.2 * ds * i -5. ) ) -1 ) / (np.exp ( 2 * ( 0.2 * ds * i -5. ) ) +1 ) ) )
u[0, 1] = 1.
u[0, 2] = 1.
u[130, 1] = 0.
u[130, 2] = 0.

for i in range (0, 131, 2):
    spl[i, 0]
fac = mu * dt / (ds ** 3)

for i in range (1, mx - 1):
    a1 = eps * dt * ( u[i + 1, 0] + u[i, 0] + u[i-1, 0] ) / ( ds*6. )
    if i > 1 and i < 129:
        a2 = u[i + 2, 0] + 2. * u[i - 1, 0] - 2. * u[i + 1 , 0] - u[i -2, 0]
    else:
        a2 = u[i - 1, 0] - u[i + 1, 0]
    a3 = u[i + 1, 0] - u[i - 1, 0]
    u[i, 1] = u[i, 0] - a1 * a3 - fac * a2 / 3.
```

Implementacja rozwiązania solitonu KdeV

```
34
35 for j in range (1, max + 1):
36     for i in range(1, mx - 2):
37         a1 = eps * dt * ( u[i + 1, 1] + u[i, 1] + u[i - 1, 1] ) / (3. * ds )
38         if i > 1 and i < mx -2:
39             a2 = u[i + 2, 1] + 2. * u[i - 1, 1] - 2. * u[i + 1, 1] - u[i-2, 1]
40         else:
41             a2 = u[i - 1, 1] - u[i + 1, 1]
42         a3 = u[i + 1, 1] - u[i - 1, 1]
43         u[i, 2] = u[i, 0] - a1 * a3 - 2. * fac * a2 / 3.
44     if j%100 == 0:
45         for i in range (1, mx - 2):
46             spl[i, m] = u[i, 2]
47         m = m + 1
48     for k in range(0, mx):
49         u[k, 0] = u[k, 1]
50         u[k, 1] = u[k, 2]
51
52
53 x = list(range(0, mx, 2))
54 y = list(range(0, 21))
55 X, Y = p.meshgrid(x, y)
56 fig = p.figure()
57 ax = Axes3D(fig)
58 ax.plot_wireframe(X, Y, spl[X,Y], color = 'r')
59 ax.set_xlabel('Position')
60 ax.set_ylabel('Time')
61 ax.set_zlabel('Disturbance')
62 p.show()
```

Implementacja rozwiązania solitonu KdeV



Hydrodynamika, równanie Naviera-Stokesa

Podstawowym równaniem hydrodynamiki jest równanie ciągłości:

$$\frac{\partial \rho(\mathbf{x},t)}{\partial t} + \vec{\nabla}\cdot\mathbf{j} = 0, \quad \mathbf{j} \stackrel{\text{def}}{=} \rho\mathbf{v}(\mathbf{x},t).$$

Wprowadźmy napierw specjalną pochodną czasową "hydrodynamiczną pochodną" Dv/Dt, która daje nam obraz jak zmienia się prędkość materiału w elemencie cieczy a także bierze pod uwagę zmiany pochodzące od ruchu cieczy a także wyraźną zależność prędkości od czasu

$$\frac{D\mathbf{v}}{Dt} \stackrel{\text{def}}{=} (\mathbf{v}\cdot\vec{\nabla})\mathbf{v} + \frac{\partial \mathbf{v}}{\partial t}.$$

Hydrodynamika, równannie Naviera-Stokesa

$$\frac{D\mathbf{v}}{Dt} = \nu\nabla^2\mathbf{v} - \frac{1}{\rho}\vec{\nabla}P(\rho,T,x),$$

$$\frac{\partial v_x}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_x}{\partial x_j} = \nu \sum_{j=x}^{z} \frac{\partial^2 v_x}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial x},$$

$$\frac{\partial v_y}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_y}{\partial x_j} = \nu \sum_{j=x}^{z} \frac{\partial^2 v_y}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial y},$$

$$\frac{\partial v_z}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_z}{\partial x_j} = \nu \sum_{j=x}^{z} \frac{\partial^2 v_z}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial z}.$$

gdzie ν jest lepkością kinematyczną, ciśnieniem. Równania te opisują transfer pędu cieczy w niektórych miejscach przestrzeni co jest spowodowane siłami i

Hydrodynamika, równannie Naviera-Stokesa

Zajmujemy się przepływem ustalonym wokół obiektu, więc wszystkie pochodne po czasie prędkości znikają. Ponadto przyjmujemy, że ciecz jest nieściśliwa więc pochodna z gęstości również zanika. Tak więc:

$$\vec{\nabla} \cdot \mathbf{v} \equiv \sum_i \frac{\partial v_i}{\partial x_i} = 0,$$

$$(\mathbf{v} \cdot \vec{\nabla})\mathbf{v} = \nu \nabla^2 \mathbf{v} - \frac{1}{\rho} \vec{\nabla} P.$$

Hydrodynamika, równannie Naviera-Stokesa

Pierwsze równanie opisuje równość ilości cieczy wpływającej i wypływającej. Ignorujemy zależność prędkości od składowej Z. Nasze równania mają postać:

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0,$$

$$\nu \left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \right) = v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x},$$

$$\nu \left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2} \right) = v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial y}.$$

Hydrodynamika, równannie Naviera-Stokesa

Pierwsze równanie opisuje równość ilości cieczy wpływającej i wypływającej. Ignorujemy zależność prędkości od składowej Z. Nasze równania mają postać:

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0,$$

$$\nu \left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \right) = v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x},$$

$$\nu \left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2} \right) = v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial y}.$$

Wirowość równania Naveira-Stokesa

Wprowadzamy funkcję strumienia gdzie prędkośc jest zdefiniowana przez rotację

$$\mathbf{v} \stackrel{\text{def}}{=} \vec{\nabla} \times \mathbf{u}(\mathbf{x}) = \hat{\epsilon}_x \left(\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z} \right) + \hat{\epsilon}_y \left(\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x} \right)$$

$$u_z \equiv u \quad \Rightarrow \quad v_x = \frac{\partial u}{\partial y}, \quad v_y = -\frac{\partial u}{\partial x}.$$

Oprócz tego potrzebne nam będzie pole wirowe

$$\mathbf{w} \stackrel{\text{def}}{=} \vec{\nabla} \times \mathbf{v}(\mathbf{x}).$$

Wirowość równania Naveira-Stokesa

Jako, że prędkość w kierunku Z się nie zmienia

$$w_z = \left(\frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \right)$$

Jest to miara tego jak bardzo prędkość cieczy rotuje(obraca się).
Linie pola są ciągłe i poruszają się tak jakby były przyczepione do cząstek cieczy.

$$\mathbf{w} = \vec{\nabla} \times \mathbf{v} = \vec{\nabla} \times (\vec{\nabla} \times \mathbf{u}) = \vec{\nabla}(\vec{\nabla} \cdot \mathbf{u}) - \nabla^2 \mathbf{u}$$

Wirowość równania Naveira-Stokesa

Finalnie dostajemy podstawową relację pomiędzy strumieniem u a wirowością w:

$$\vec{\nabla}^2 \mathbf{u} = -\mathbf{w}.$$

Równanie to jest analogią do równania Poisson'a w elektrostatyce.
Po przekształceniach uzyskujemy

$$\nu \nabla^2 \mathbf{w} = [(\vec{\nabla} \times \mathbf{u}) \cdot \vec{\nabla}]\mathbf{w}.$$

Wirowość równania Naveira-Stokesa

Pozostaje nam rozwiązać układ równań złożony z 2 ostanich tj.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -w,$$

$$\nu \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} \right) = \frac{\partial u}{\partial y} \frac{\partial w}{\partial x} - \frac{\partial u}{\partial x} \frac{\partial w}{\partial y}.$$

Różniczki skończone i algorytm SOR

SOR(successive over-relaxation)

Aby rozwiązać taki układ na siatce N_x x N_y jednolitej przestrzeni h z

$$x = i\Delta x = ih, \quad i = 0, \ldots, N_x, \quad y = j\Delta y = jh, \quad j = 0, \ldots, N_y.$$

Strumień jest symetryczny potrzebujemy rozwiązania tylko w górnej połowie płaszczyzny

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \simeq \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2}.$$

$$\frac{\partial u}{\partial y}\frac{\partial w}{\partial x} \simeq \frac{u_{i,j+1} - u_{i,j-1}}{2h}\frac{w_{i+1,j} - w_{i-1,j}}{2h}.$$

Różniczki skończone i algorytm SOR

Forma różniczkowa równania wirowości Naviera-Stokesa przybiera postać:

$$u_{i,j} = \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} + h^2 w_{i,j}\right),$$

$$w_{i,j} = \frac{1}{4}(w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1}) - \frac{R}{16}\left\{[u_{i,j+1} - u_{i,j-1}]\right.$$

$$\left. \times [w_{i+1,j} - w_{i-1,j}] - [u_{i+1,j} - u_{i-1,j}]\,[w_{i,j+1} - w_{i,j-1}]\right\},$$

$$R = \frac{1}{\nu} = \frac{V_0 h}{\nu} \quad \text{(in normal units)}.$$

Parametr R jest powiązany z liczbą Reynoldsa i jest miarą siły związania nieliniowych wyrazów w równaniu.

Warunki brzegowe dla strumienia

- Przepływ swobodny

Gdy nie ma w cieczy "belki" mamy przepływ sobodny, cała ciecz posiada prędkość wlotową.

$$v_x \equiv V_0, \quad v_y = 0, \quad \Rightarrow \quad u = V_0 y, \quad w = 0.$$

Jeśli linia centralna dzieli układ wzdłuż płaszczyzny symetrii z takimi samymi przepływami i pod i nad nią dostajemy:

$$v_y = 0, \quad \Rightarrow \quad \frac{\partial u}{\partial x} = 0 \quad \text{(centerline AE)}.$$

Warunki brzegowe dla strumienia

- Linia centralna
  To linia strumienia z u=constans, ponieważ nie ma składowej prędkości, która byłaby do niej prostopadła.

- Wlot

$$v_y = -\frac{\partial u}{\partial x} = 0, \quad w = 0 \quad \text{(inlet F)}, \quad v_x = \frac{\partial u}{\partial y} = V_0$$

Warunki brzegowe dla strumienia

- Powierzchnia
  "Belka" jest wystarczająco zanurzona na tyle, żeby nie zakłucać przepływu na powierzchni

$$v_x = \frac{\partial u}{\partial y} = V_0, \quad w = 0 \quad \text{(surface G)}.$$

- Ujście
  Warunki na dnie mają bardzo mały wpływ na to co się dzieje wyżej

$$\frac{\partial u}{\partial x} = \frac{\partial w}{\partial x} = 0 \quad \text{(outlet H)}.$$

Warunki brzegowe dla strumienia

- Wokół "belki"

  Ciecz "klei się" do "belki"

$$u(x, y+h) = u(x, y) + \frac{\partial u}{\partial y}(x, y)h + \frac{\partial^2 u}{\partial y^2}(x, y)\frac{h^2}{2} + \cdots .$$

$$w \equiv w_z = \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y}.$$

$$v_x = \frac{\partial u}{\partial y} = 0 \quad \text{(beam top)}.$$

Warunki brzegowe dla strumienia

$$\frac{\partial v_y}{\partial x} = 0 \;\Rightarrow\; w = -\frac{\partial v_x}{\partial y} = -\frac{\partial^2 u}{\partial y^2}.$$

$$w \simeq -2\frac{u(x, y+h) - u(x, y)}{h^2} \;\Rightarrow\; w_{i,j} = -2\frac{u_{i,j+1} - u_{i,j}}{h^2} \quad \text{(top)}.$$

$u = 0;$	$w = 0$	Centerline EA
$u = 0,$	$w_{i,j} = -2(u_{i+1,j} - u_{i,j})/h^2$	Beam back B
$u = 0,$	$w_{i,j} = -2(u_{i,j+1} - u_{i,j})/h^2$	Beam top C
$u = 0,$	$w_{i,j} = -2(u_{i-1,j} - u_{i,j})/h^2$	Beam front D
$\partial u/\partial x = 0,$	$w = 0$	Inlet F
$\partial u/\partial y = V_0,$	$w = 0$	Surface G
$\partial u/\partial x = 0,$	$\partial w/\partial x = 0$	Outlet H

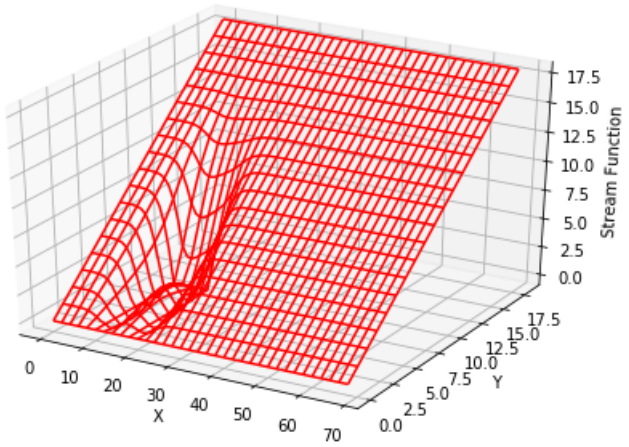
Implementacja SOR na siatkę

```python
import matplotlib.pylab as p
from mpl_toolkits.mplot3d import Axes3D
import numpy as np

Nxmax=70
Nymax=20
IL=10
H=8
T=8
h=1.
u=np.zeros((Nxmax+1,Nymax+1),float)
w=np.zeros((Nxmax+1,Nymax+1),float)
V0=1.0
omega=0.1
nu=1.
iter=0
R=V0*h/nu

def borders():
    for i in range(0,Nxmax+1):
        for j in range(0,Nymax+1):
            w[i,j]=0.
            u[i,j]=j*V0
    for i in range(0,Nxmax+1):
        u[i,Nymax]=u[i,Nymax-1]+V0*h
        w[i,Nymax-1]=0.
    for j in range(0,Nymax+1):
        u[1,j]=u[0,j]
        w[0,j]=0.
    for i in range(0,Nymax+1):
        if i<=IL and i>=IL+T:
            u[i,0]=0.
            w[i,0]=0.
    for j in range(1,Nymax):
        w[Nxmax,j]=w[Nxmax-1,j]
        u[Nxmax,j]=u[Nxmax-1,j]

def beam():
    for j in range(0,H+1):
        w[IL,j]=-2*u[IL-1,j]/(h*h)
```

Implementacja SOR na siatkę

```
    for i in range(IL,IL+T+1):
        w[i,H-1]=-2*u[i,H]/(h*h)
    for i in range(IL,IL+T+1):
        for j in range(0,H+1):
            u[IL,j]=0.
            u[IL+T,j]=0.
            u[i,H]=0

def relax():
    beam()
    for i in range(1,Nxmax):
        for j in range(1,Nymax):
            r1=omega*((u[i+1,j]+u[i-1,j]+u[i,j+1]+u[i,j-1]+h*h*w[i,j])*0.25-u[i,j])
            u[i,j]+=r1
    for i in range(1,Nxmax):
        for j in range(1,Nymax):
            a1=w[i+1,j]+w[i-1,j]+w[i,j+1]+w[i,j-1]
            a2=(u[i,j+1]-u[i,j-1])*(w[i+1,j]-w[i-1,j])
            a3=(u[i+1,j]-u[i-1,j])*(w[i,j+1]-w[i,j-1])
            r2=omega*((a1-(R/4.)*(a2-a3))/4.0-w[i,j])
            w[i,j]+=r2
borders()
while(iter<=100):
    iter+=1
    if iter%10==0:
        print(iter)
    relax()
for i in range(0,Nxmax+1):
    for j in range(0,Nymax+1):
        u[i,j]=u[i,j]/(V0*h)
x=range(0,Nxmax-1)
y=range(0,Nymax-1)
X,Y=p.meshgrid(x,y)

def functz(u):
    z=u[X,Y]
    return z

Z=functz(u)
fig=p.figure()
ax=Axes3D(fig)
ax.plot_wireframe(X,Y,Z,color='r')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Stream Function')
```

Implementacja SOR na siatkę



Bibliografia

- Materiały dydaktyczne udostępnione przez prowadzącego: 3Solitony.pdf
- https://en.wikipedia.org/wiki/Soliton
- https://www.physics.utoronto.ca/ phy326/sol/sol.pdf