Fluid Flow Stability

Csaba Hős, cshos@hds.bme.hu
Budapest University of Technology and Economics
Dept. of Hydrodynamic Systems

10th October 2017

Contents

1 Numerics **1**
1.1 Introduction . . . 1
1.2 Differentiation matrices . . . 2
1.2.1 First derivatives . . . 2
1.2.2 Higher-order derivatives . . . 6
1.2.3 Boundary conditions . . . 6
1.3 Techniques automatically satisfying BCs . . . 6
1.3.1 Galerkin technique . . . 7
1.3.2 Collocation . . . 7

2 Stability of planar incompressible flows **9**
2.1 Governing equations for Newtonian fluids . . . 9
2.2 Governing equations for general fluids . . . 9
2.3 Homework description . . . 10

1 Numerics

1.1 Introduction

The aim of this document is to provide a brief introduction on the numerical techniques used to turn partial differential equations (PDEs) into (a large system of, possibly nonlinear) ordinary differential equations (ODEs), in the form of $\dot{x} = \mathcal{F}(x,t)$. There are two motivations behind this desire:

- the resulting ODEs can be integrated by means of standard techniques, such as e.g. Runge-Kutta schemes, or,
- we wish to analyse the *stability* of the ODEs and hope to generalize the results to the original PDE.

As an example, let us have a look at the following standard problem:

$$\mu \frac{\partial^2 w}{\partial^2 t} = E\,I\,\frac{\partial^4 w}{\partial^4 x} + q(x,t), \tag{1}$$

which is the problem of the vibrating elastic beam with μ mass per unit length, $w(x,t)$ is the deformed shape, $I = \int\int z^2 dy$ is the moment of inertia, E is the elasticity modulus and $q(x,t)$ is the load, that varies both along the beam (x) and in time (t).

The *boundary conditions*, in the case of fixed ends, take the form of

$$w(x,t)|_{x=0,L} = 0 \quad \text{and} \quad \left.\frac{d}{dx} w(x,t)\right|_{x=0,L} = 0. \tag{2}$$

Let $\hat{w}$ denote the exact solution. Then, the $\mathcal{R}(w)$ *error function* (residual function) is

$$\mathcal{R}(w) := \mu \frac{\partial^2 w}{\partial^2 t} - E\,I\,\frac{\partial^4 w}{\partial^4 x} - q(x,t) \quad \text{és} \quad \mathcal{R}(\hat{w}) = 0. \tag{3}$$

1.2 Differentiation matrices

1.2.1 First derivatives

Let us approximate the first derivative of $y(x)$ by centered difference scheme, that is

$$y'(x)\big|_{x=x_i} \approx \frac{y_{i+1} - y_{i-1}}{2\Delta}, \tag{4}$$

where $y_i := y(x_i)$ and Δ is the grid size. This is equivalent of placing the origin at x_i, fitting a parabola through these points and evaluating the derivative at the origin:

$$\tilde{y}(x) = a + bx + cx^2 \quad \text{where} \quad \tilde{y}(-\Delta) = y_{i-1}, \quad \tilde{y}(0) = y_i \quad \text{and} \quad \tilde{y}(\Delta) = y_{i+1}, \tag{5}$$

whose solution is

$$a = y_i, \quad b = \frac{y_{i+1} - y_{i-1}}{2\Delta} \quad \text{and} \quad c = \frac{y_{i+1} - 2y_i + y_{i-1}}{2\Delta^2}. \tag{6}$$

Clearly, $y'(0) = b$, which is exactly (4).

At the boundaries, we'll face the problem that the nodes x_0 and x_{N+1} are missing. We coudl use *first-order approximation*

$$y'(x_1) \approx \frac{y_2 - y_1}{\Delta} \quad \text{and} \quad y'(x_N) \approx \frac{y_{N-1} - y_N}{\Delta}. \tag{7}$$

Or, ee shall use the previous technique, but now the parabola fitting problem is

$$\tilde{y}(x) = a + bx + cx^2 \quad \text{where} \quad \tilde{y}(0) = y_1, \quad \tilde{y}(\Delta) = y_2 \quad \text{és} \quad \tilde{y}(2\Delta) = y_2, \tag{8}$$

and the coefficients are

$$a = y_1, \quad b = \frac{-3y_1 + 4y_2 - y_3}{2\Delta} \quad \text{és} \quad c = \frac{y_1 - 2y_2 + y_3}{2\Delta^2}, \tag{9}$$

from which

$$y'(0) \approx b = \frac{-3y_1 + 4y_2 - y_3}{2\Delta}. \tag{10}$$

In a similar manner, the derivative at the right end is

$$y'(x_N) \approx b = \frac{3y_N - 4y_{N-1} + y_{N-2}}{2\Delta}. \tag{11}$$

Putting all these together, one might define a differentiation matrix, with which the derivative is approximated as $y' = \mathcal{D}_{CDS} y$ and

$$\mathcal{D}_{CDS} = \frac{1}{2\Delta} \begin{pmatrix} -3 & 4 & -1 & \dots & & & & \\ 0 & -1 & 0 & 1 & \dots & & & \\ \dots & 0 & -1 & 0 & 1 & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & 0 & -1 & 0 & 1 & 0 \\ & & & \dots & 0 & 1 & -4 & 3 \end{pmatrix}. \tag{12}$$

Another popular approach is to use the Chebyshev polynomials for interpolation. These polynomials are the solutions of the Chebyshev equation

$$\left(1 - x^2\right) y'' - xy' + n^2 y = 0 \tag{13}$$

and are given by

$$T_k(x) = \cos\left(k \arccos x\right) \quad -1 \le x \le 1. \tag{14}$$

The first few of them are:

$$T_0(x) = \cos(0) = 1 \tag{15}$$

$$T_1(x) = \cos(\arccos x) = x \tag{16}$$

$$T_2(x) = \cos(2 \arccos x) = 2x^2 - 1 \quad \text{and} \tag{17}$$

$$T_3(x) = \cos(3 \arccos x) = 4x^3 - 3x. \tag{18}$$

(We have exploited that $\cos 2\theta = 2\cos^2\theta - 1$ és $\cos 3\theta = 4\cos^3\theta - 3\cos\theta$.) There is also a recurrence formula $T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$. Hence, we approximate $y(x)$ by

$$y(x) \approx \sum_{i=0}^{N} a_k T_k(x), \quad -1 \le x \le 1. \tag{19}$$

Obviously, the weights a_k must satisfy $y(x_i) = y_i$. The optimal way of using these polynomials is on a special grid, whose nodes are the so-called Lobatto points:

$$x_j = \cos\frac{\pi j}{N}, \quad j = 0, \dots, N, \tag{20}$$

that is, we build an equidistant grid on the *upper arc* of the unit circle and project the points onto the x axis. Notice that this grid is denser around the $x = \pm 1$ end points. Finally, the differentiation matrix is:

$$D_{Cheb,i,j} = \begin{cases} \frac{c_i}{c_j} \frac{(-1)^{i+j}}{x_i - x_j} & i \ne j \\ -\frac{x_i}{2(1-x_i^2)} & 1 \le i = j \le N-1 \\ \frac{2N^2+1}{6} & i = j = 0 \quad \text{és} \\ -\frac{2N^2+1}{6} & i = j = N \end{cases} \quad \text{ahol} \quad c_k = \begin{cases} 2 & k = 0, N \\ 1 & 1 \le k \le N-1. \end{cases} \tag{21}$$

Figure 1 (see the computer code on the next page) demonstrates the use of differentiation matrices. Notice the diabolish accuracy of the Chebyshev differentiation matrix. Notice that

- 10^{-13} is approx. the machine precision, meaning that due to the finite number representation, one cannot gain more accuracy with any other technique and that

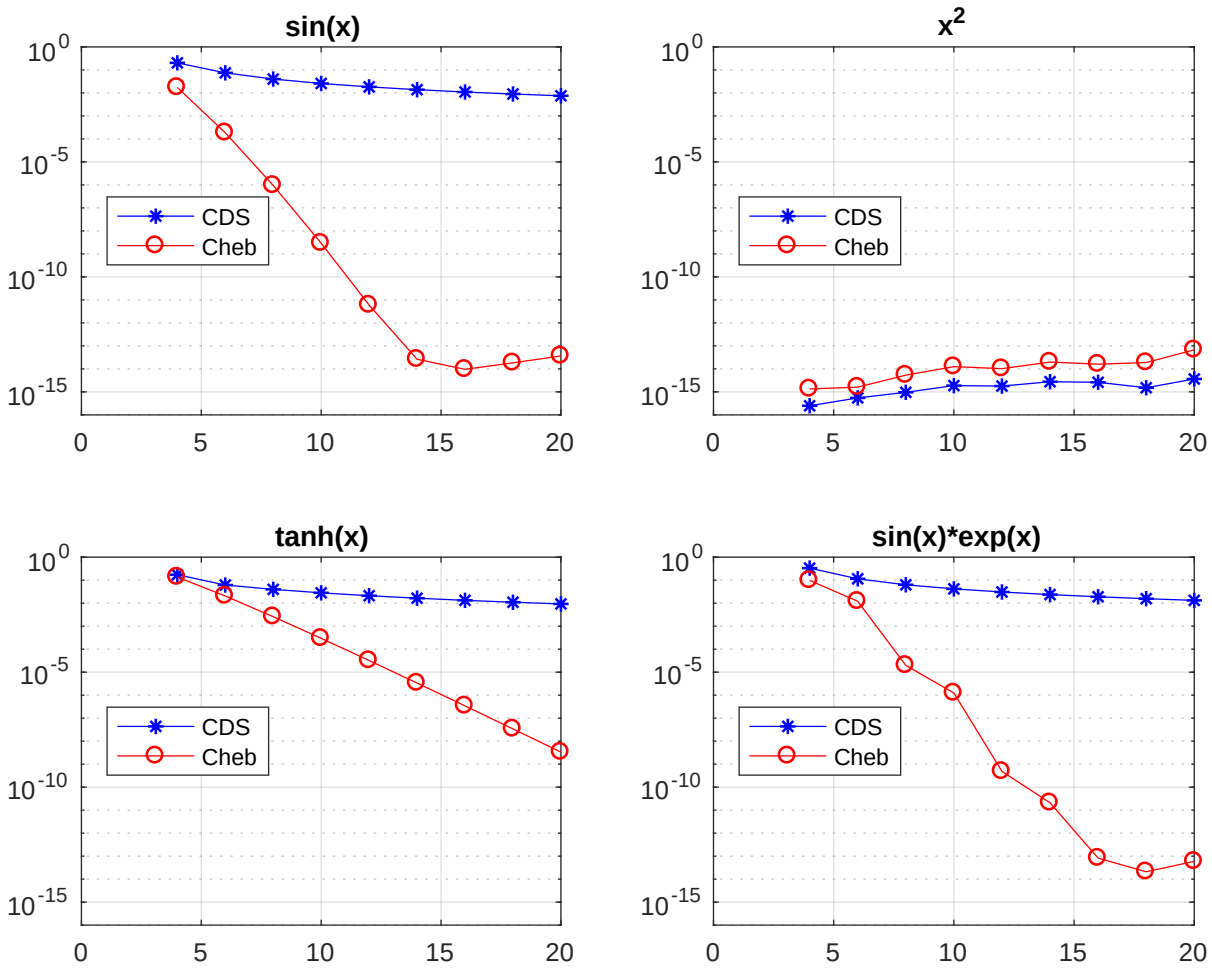

Figure 1: Error of the derivative of several functions by means of centred differentiation (CDS) and Chebyshev polinomials (Cheb) as a function of grid points.

- for the x^2 function, both techniques give the highest possible accuracy.

Demo on differentiation matrices

```
function diffmx_example
clear all, close all

f=figure(1)
tit={'sin(x)','x^2','tanh(x)','sin(x)*exp(x)'};
for j=1:4
err1=[]; err2=[]; xx=[];
for Npt=4:2:20
    xx=[xx,Npt];
    [x1,D1]=cdsdif(Npt,1);
    [y1,dy1]=funs(x1,j);
    err1=[err1, norm(dy1-D1*y1)];

    [x2,D2]=chebdif(Npt,1);
    [y2,dy2]=funs(x2,j);
    err2=[err2, norm(dy2-D2*y2)];

    fprintf('\n Npt=%3d, err_CDS=%5.3e, err_Cheb=%5.3e',...
    Npt,err1(end),err2(end));
end

subplot(2,2,j)
semilogy(xx,err1,'b*-',xx,err2,'ro-'), grid on
ylim([1e-16,1]),
title(tit{j}), legend('CDS','Cheb','Location','West')

print(f,'-dpdf','diffmx_example.pdf')
end

function [y,dy]=funs(x,i)
if i==1, y=sin(x); dy=cos(x); end
if i==2, y=x.^2; dy=2*x; end
if i==3, y=tanh(x); dy=1-(tanh(x)).^2; end
if i==4, y=sin(x).*exp(x); dy=cos(x).*exp(x)+sin(x).*exp(x); end
end
```

Centered diff. matrix

```
function [x, DM] = cdsdif(N, M)

x=linspace(-1,1,N)';
dx=x(2)-x(1);
DM=zeros(N,N);
D(1,1)=-3; D(1,2)=4; D(1,3)=-1;
D(N,N)=3;  D(N,N-1)=-4; D(N,N-2)=1;
for i=2:N-1
```

```
    D(i,i-1)=-1;
    D(i,i+1)=1;
end
D=D/2/dx;
DM=D^M;
end
```

1.2.2 Higher-order derivatives

The beauty of this approach is that, upon using differentiation matrices, it is easy to see that e.g.

$$y'' \approx \mathcal{D}^2 y. \tag{22}$$

1.2.3 Boundary conditions

When dealing with PDEs, great care must be devoted to boundary conditions. For example, our motivating example used the boundary conditions

$$w(x,t)|_{x=0,L} = 0 \quad \text{and} \quad \left.\frac{d}{dx} w(x,t)\right|_{x=0,L} = 0. \tag{23}$$

This means that if we use centred difference scheme, we have

- $y(0) = y_1 = 0$ and $y(L) = y_N = 0$ and
- as $y'(0) = 0$, we have – by virtue of (9) – $0 = y'(0) = b = \frac{-3y_1+4y_2-y_3}{2\Delta}$, which gives $4y_2 = y_3$ (as $y_1 = 0$) and
- as $y'(L) = 0$, we have – by virtue of (11) – $0 = y'(L) = b = \frac{3y_N-4y_{N-1}+y_{N-2}}{2\Delta}$, which gives $4y_{N-1} = y_{N-2}$ (as $y_N = 0$).

It is clear if that if one uses e.g. Chebyshev differentiation matrices, the implementation of the boundary conditions is even more complicated.

1.3 Techniques automatically satisfying BCs

We shall approximate the unknown $w(x,t)$ function as

$$w(x,t) \approx \sum_{i=0}^{N} a_i(t) f_i(x), \tag{24}$$

where $a_i(t)$ are unknown time-dependent amplitudes of the shape functions $f_i(x)$. We wish to choose the shape functions in such a way that *they automatically satisfy the boundary conditions*, i.e.

$$w(x,t)|_{x=0,L} = 0 \quad \rightarrow \quad f_i(0) = 0 \quad \text{and} \quad f_i(L) = 0 \quad \text{for all i}, \tag{25}$$

and we also have

$$\left.\frac{d}{dx} w(x,t)\right|_{x=0,L} = 0 \quad \rightarrow \quad f_i'(0) = 0 \quad \text{and} \quad f_i'(L) = 0 \quad \text{for all i}. \tag{26}$$

For example, we might choose $f_i(x) = \cos\left(i\frac{x}{L}2\pi\right) - 1$, which satisfies both of the above conditions.

1.3.1 Galerkin technique

We now apply the Galerkin technique, which requires that the residual vector $\mathcal{R}(w)$ is perpendicular to all of the basis vectors, i.e.

$$0 = \langle \mathcal{R}(w), f_j(x) \rangle \quad \text{for all } j = 1 \dots N. \tag{27}$$

This means that

$$\begin{aligned}
0 = \langle \mathcal{R}(w), f_j(x) \rangle = & \tag{28}\\
= \left\langle \mu \frac{\partial^2 w}{\partial^2 t} - E\,I\,\frac{\partial^4 w}{\partial^4 x} - q(x), f_j(x) \right\rangle = & \tag{29}\\
= \left\langle \mu \frac{\partial^2 \sum_{i=0}^{N} a_i(t) f_i(x)}{\partial^2 t} - E\,I\,\frac{\partial^4 \sum_{i=0}^{N} a_i(t) f_i(x)}{\partial^4 x} - q(x), f_j(x) \right\rangle = & \tag{30}\\
= \left\langle \mu \sum_{i=0}^{N} \ddot{a}_i(t) f_i(x) - E\,I \sum_{i=0}^{N} a_i(t) f_i^{(iv)}(x) - q(x), f_j(x) \right\rangle = & \tag{31}\\
= \mu \sum_{i=0}^{N} \ddot{a}_i(t) \langle f_i(x), f_j(x) \rangle - E\,I \sum_{i=0}^{N} a_i(t) \left\langle f_i^{(iv)}(x), f_j(x) \right\rangle - \langle q(x), f_j(x) \rangle . & \tag{32}
\end{aligned}$$

Note that we have $j = 1 \dots N$ of the above equations, which can be conveniently written as

$$\mu\, \underline{\underline{\mathcal{A}}}^G\, \underline{\ddot{a}} = EI\, \underline{\underline{\mathcal{B}}}^G\, \underline{a} + \underline{\mathcal{C}}^G. \tag{33}$$

With our previous choice $f_i(x) = \cos\left(i \frac{x}{L} 2\pi\right) - 1$ the coefficients are

$$\mathcal{A}_{ji}^G = \int_0^L f_i(x) f_j(x) dx = \begin{cases} L & \text{if} \quad i \neq j \\ 3L/2 & \text{if} \quad i = j \end{cases} \tag{34}$$

$$\mathcal{B}_{ji}^G = \int_0^L f_i^{(iv)}(x) f_j(x) dx = \begin{cases} 0 & \text{if} \quad i \neq j \\ 8i^4\pi^4/L^3 & \text{if} \quad i = j \end{cases} \quad \text{and} \tag{35}$$

$$\mathcal{C}_j^G = \int_0^L q(x) f_j(x) dx. \tag{36}$$

(The superscript 'G' stands for Galerkin.) The above ordinary differential equation can be easily solved for $a_i(t)$ and the actual shape of the beam can then be reconstructed.

1.3.2 Collocation

When applying the collocation technique, instead of forcing the approximation to be optimal over an interval (in our case, [0,L]), we force it to be accurate at pre-prescribed *collocation points* x_j, giving

$$0 = \mathcal{R}(w(t, x_j)) \quad \text{for all } j = 1 \dots N. \tag{37}$$

This means that we have

$$0 = \mu \sum_{i=0}^{N} \ddot{a}_i(t) f_i(x_j) - E\,I \sum_{i=0}^{N} a_i(t) f_i^{(iv)}(x_j) - q(x_j). \tag{38}$$

This can be formally written the same way as (33) but now the coefficient matrices are different:

$$\mathcal{A}^C_{ji} = f_i(x_j), \quad \mathcal{B}^C_{ji} = f_i^{(iv)}(x_j) \quad \text{and} \quad \mathcal{C}^C_j = q(x_j). \tag{39}$$

There are many ways of choosing the collocation points, the simplest way is to have them uniformly spaced. Once the mesh is set up and the above matrices are computed, one can solve (33) and reconstruct the surface.

One solves the resulting system of ODEs by means of numerical integration, which requires rewriting

$$\mu\mathcal{A}\ddot{a} = EI\mathcal{B}a + \mathcal{C}$$

as a first-order system of ODEs. Let us introduce $y_1 = a$ and $y_2 = \dot{a}$ (note that both y_1 and y_2 are $N \times 1$ matrices). Then, we have

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} 0 & \mathcal{I}_N \\ \frac{EI}{\mu}\mathcal{A}^{-1}\mathcal{B} & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{\mu}\mathcal{A}^{-1}\mathcal{C} \end{pmatrix}, \tag{40}$$

which can be integrated by standard ODE solvers.

2 Stability of planar incompressible flows

2.1 Governing equations for Newtonian fluids

In the case 2D planar flow of a Newtonian, incompressible liquid, the continuity and momentum equation reads

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \tag{41}$$

$$\frac{\partial v_x}{\partial t} + v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu \left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \right) \tag{42}$$

$$\frac{\partial v_y}{\partial t} + v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu \left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2} \right) \tag{43}$$

After introducing the stream function Ψ, for which $\Psi_x = -v_y$ and $\Psi_y = v_x$, the continuity equation is automatically satisfied. Then we compute $\partial_y \mathrm{Mom}_x - \partial_x \mathrm{Mom}_y$, which gives

$$\frac{\partial \phi}{\partial t} + \frac{\partial \Psi}{\partial y}\frac{\partial \phi}{\partial x} - \frac{\partial \Psi}{\partial x}\frac{\partial \phi}{\partial y} = \nu \Delta \phi, \tag{44}$$

where $\phi = \Delta \Psi = \Psi_{xx} + \Psi_{yy}$.

2.2 Governing equations for general fluids

The general form of the equation of motion is

$$\rho \frac{Dv_i}{Dt} = \frac{\partial \tau_{ij}}{\partial x_j} + \rho G_i, \tag{45}$$

where we used Einstein's notation, i.e. $\frac{\partial z_{ij}}{\partial x_j} = \sum_j \frac{\partial z_{ij}}{\partial x_j}$ (reoccuring indices mean summitation). In the case 2D planar flow of a general, incompressible liquid, the continuity and momentum equation reads

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \tag{46}$$

$$\frac{\partial v_x}{\partial t} + v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} = \frac{1}{\rho} \left(-\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} \right) \tag{47}$$

$$\frac{\partial v_y}{\partial t} + v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} = \frac{1}{\rho} \left(-\frac{\partial p}{\partial y} + \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} \right), \tag{48}$$

where the stress tensor is $((i,j) = (x,y))$

$$\tau_{ij} = 2\mu(\dot{\gamma})\, D_{ij}, \quad D_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{49}$$

and the shear rate is

$$\dot{\gamma} = \sqrt{2\, D : D} = \sqrt{2\left(\frac{\partial v_x}{\partial x}\right)^2 + 2\left(\frac{\partial v_y}{\partial y}\right)^2 + \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)^2} \tag{50}$$

As an example, let us recompute the Newtonian case:

$$\begin{aligned}
\frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} &= \frac{\partial}{\partial x}\left(2\mu D_{xx}\right) + \frac{\partial}{\partial y}\left(2\mu D_{xy}\right) \\
&= \frac{\partial}{\partial x}\left(2\mu \frac{1}{2}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_x}{\partial x}\right)\right) + \frac{\partial}{\partial y}\left(2\mu \frac{1}{2}\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)\right) \\
&= \mu\left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} + \frac{\partial}{\partial x}\left(\underbrace{\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}}_{=0,\text{continuity eq.}}\right)\right) \\
&= \mu\left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2}\right)
\end{aligned}$$

For a power-law fluid, we have $\mu = K\dot{\gamma}^{n-1}$.

2.3 Homework description

For the homework problems ("Linear stability of the Couette/Poiseuille flow"), please follow the next steps:

- Rewrite equation (44) assuming power-law fluid.
- Find the steady-state solution. Remember that in this case, one assumes $v_x = V_x(y)$ (V_x does not vary along x) and $v_y(x, y) = 0$.
- Decompose the flow field into $v_x(x, y) = V_x(y) + \tilde{u}(x, y)$ and $v_y(x, y) = 0 + v(x, y)$ and rewrite (44).
- Boundary conditions for Couette flow: $v_x(x, -h) = 0$, $v_x(x, +h) = v_{wall}$ and $v_y(x, \pm h) = 0$.
- Boundary conditions for Poiseuille flow: $v_x(x, \pm h) = 0$ and $v_y(x, \pm h) = 0$ but $p(x, y) = \frac{\Delta p}{L} := \Pi = \text{const}$.
- For both cases, find a suitable function series in the y direction that satisfies the boundary conditions:
$$\Psi(x, y) = e^{i\omega x} \sum_i a_i(t) f_i(y).$$
- Apply the prescribed numerical technique to obtain stability.