Axisymmetric Stagnation Flow in a Finite Gap

Preprint · December 2021

1 author:

Brian G Higgins
University of California, Davis

BGHiggins/UCDavis/December_2021

Axisymmetric Stagnation Flow in a Finite Gap

Background

In these notes we extend the analysis of axisymmetric stagnation flow originally analyzed by Homann (1936) on the semi-infinite domain to an equivalent flow that is confined between two parallel disks. A full account of the early work on stagnation flow can be found in the monograph on laminar boundary layers edited by Rosenhead (1963), starting with the classical analysis of Heimenz. The top disk, located at z=L, is porous and allows for the uniform injection of a fluid. The governing equations, based on a similarity solution for the velocity field, remain the same but the boundary conditions are different, as discussed below

As before we seek a similarity solution of the form

$$u = U(z), \quad v(z, r) = r f(z) \tag{1}$$

The flow in the gap satisfies

$$\begin{aligned} &\frac{dU}{dz} + 2 f = 0 \\ &U \frac{df}{dz} + f^2 = -\Delta_r + \nu \frac{d^2 f}{dz^2} \end{aligned} \tag{2}$$

where the quantity Δ_r is given by

$$\Delta_r = \frac{1}{\rho r} \frac{\partial p}{\partial r} = \text{constant} \tag{3}$$

The boundary conditions for the flow are now

$$\begin{aligned} &\text{BC1}: \quad U(0) = 0, \quad f(0) = 0 \\ &\text{BC2}: \quad U(L) = -U_{in}, \quad f(L) = 0 \end{aligned} \tag{4}$$

Thus along the top porous disk, we specify an inlet velocity U_{in} as well as the radial velocity $v(r, L) = r f(L)$ along the surface of the disk, which is zero for all r, which implies that f(L) = 0.

For this flow field there is a characteristic length scale L and velocity scale U_{in}. These characteristic scales are used to define the following dimensionless variables:

$$\hat{z} = \frac{z}{L}, \quad \hat{U} = \frac{U}{U_{in}}, \quad \hat{f} = \frac{f\,L}{U_{in}}, \quad \mathbb{R}e = \frac{\rho\, U_{in}\, L}{\mu} \tag{5}$$

The dimensionless form of the equations and boundary conditions becomes

$$\frac{d\hat{U}}{d\hat{z}} + 2\,\hat{f} = 0$$

$$\hat{U}\,\frac{d\hat{f}}{d\hat{z}} + \hat{f}^2 = -\hat{\Delta}_r + \frac{1}{\mathbb{R}e}\,\frac{d^2\hat{f}}{d\hat{z}^2} \tag{6}$$

BC1 : $\hat{U}(0) = 0, \ \hat{f}(0) = 0$

BC2 : $\hat{U}(1) = -1, \ \hat{f}(1) = 0$

where $\hat{\Delta}_r = \Delta_r L^2 / \rho\, U_{in}^2$. Note in this formulation $\hat{\Delta}_r$ is unknown and can be viewed as an eigenvalue (or more precisely a constraint) that must be determined as part of the solution such that the boundary conditions are satisfied. The parameter in the problem is the Reynolds number $\mathbb{R}$e.

Numerical Methods : Shooting Method

We will use two methods to solve for the flow field: A shooting method to solve the nonlinear BVP and a finite difference method.

To implement a shooting method we define the following initial value problem:

$$\frac{d\hat{U}}{d\hat{z}} + 2\,\hat{f} = 0$$

$$\hat{U}\,\frac{d\hat{f}}{d\hat{z}} + \hat{f}^2 = -\hat{\Delta}_r + \frac{1}{\mathbb{R}e}\,\frac{d^2\hat{f}}{d\hat{z}^2} \tag{7}$$

$$\frac{d\hat{\Delta}_r}{dz} = 0$$

ICs : $\hat{U}(0) = 0, \ \hat{f}(0) = 0, \ \frac{d\hat{f}}{dz}(0) = \alpha, \ \hat{\Delta}_r(0) = \beta$

The parameters α and β must be selected such that the conditions at z =1 below are satisfied:

BCs : $\hat{U}(1) = -1, \ \hat{f}(1) = 0.$ (8)

Mathematica Code:

In[1]:=
```
ODE[α_, β_, Re_] := {2 F[z] + U'[z] == 0, F[z]^2 + U[z] × F'[z] == -Δ[z] + 1/Re F''[z],
   Δ'[z] == 0, F[0] == 0, U[0] == 0, F'[0] == α, Δ[0] == β};
sol[α_, β_, Re_] := NDSolve[ODE[α, β, Re], {F, U, Δ}, {z, 0, 1}];
FendBC[α_?NumericQ, β_?NumericQ, Re_?NumericQ] := First[F[1] /. sol[α, β, Re]];
UendBC[α_?NumericQ, β_?NumericQ, Re_?NumericQ] := First[U[1] /. sol[α, β, Re]];
bc =
 FindRoot[{FendBC[α, β, 10] == 0, UendBC[α, β, 10] == -1}, {α, 4.0, 5}, {β, -1.2, -1.7}]
res = sol[α /. bc, β /. bc, 10]
```

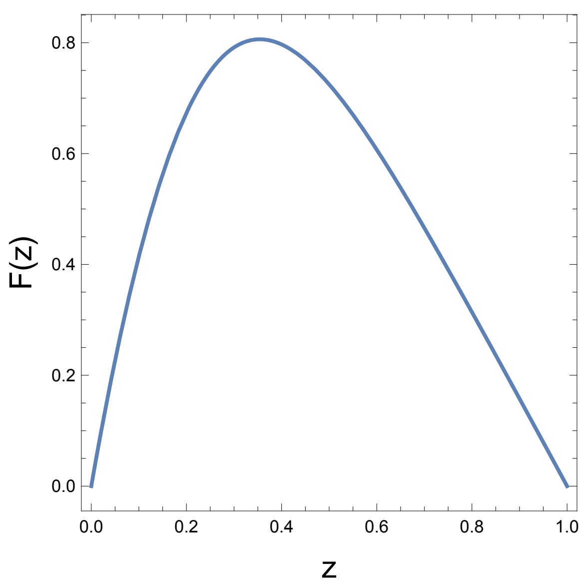
Out[5]=
```
{α → 4.93725, β → -1.58847}
```

Out[6]=
```
{{F → InterpolatingFunction[ Domain: {{0., 1.}} Output: scalar ],
  U → InterpolatingFunction[ Domain: {{0., 1.}} Output: scalar ],
  Δ → InterpolatingFunction[ Domain: {{0., 1.}} Output: scalar ]}}
```

Here are the plots of the solution for Re=10

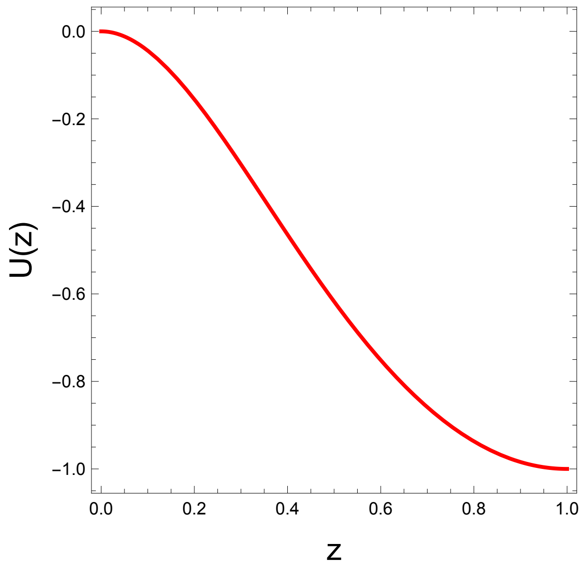
In[7]:=
```
plt1a = Plot[F[z] /. res, {z, 0, 1}, PlotStyle → {Thick}, Frame → True,
  FrameLabel → {Style["z", 16], Style["F(z)", 16]}, AspectRatio → 1, ImageSize → 300]
```

Out[7]=



In[8]:=
```
plt1b = Plot[U[z] /. res, {z, 0, 1}, PlotStyle → {Thick, Red}, Frame → True,
  FrameLabel → {Style["z", 16], Style["U(z)", 16]}, AspectRatio → 1, ImageSize → 300]
```

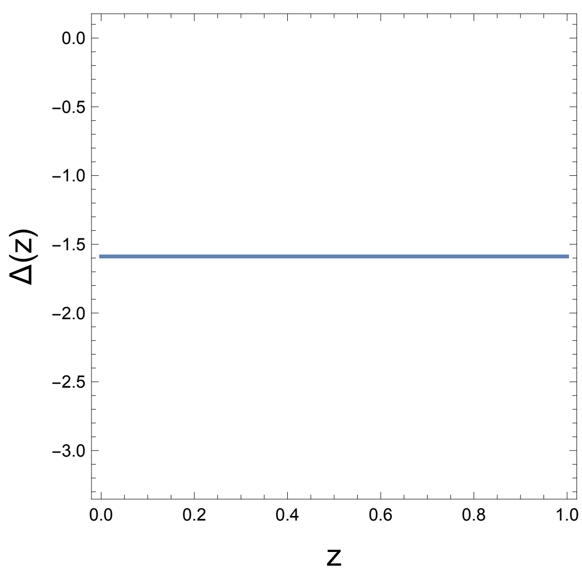
Out[8]=



As one would expect the function Δ(z) should be a constant:

In[9]:=
```
plt1c = Plot[Δ[z] /. res, {z, 0, 1}, PlotStyle → {Thick}, Frame → True,
  FrameLabel → {Style["z", 16], Style["Δ(z)", 16]}, AspectRatio → 1, ImageSize → 300]
```

Out[9]=



Finite Difference Formulation

We use a finite difference scheme with a strategy for determining the constraint parameter $\hat{\Delta}_r$.

First we discretize the finite domain by selecting the nodes of the grid to be spaced uniformly such that

$$\hat{z}_i = (i - 1)\,\Delta\hat{z}, \quad i = 1, 2, \ldots, NP, \qquad \Delta\hat{z} = \frac{\hat{z}_{end}}{NP - 1} \tag{9}$$

where NP is the number of nodes in our finite difference grid. Note that

$$\hat{z}_1 = 0, \quad \hat{z}_{NP} = \hat{z}_{end} = 1 \tag{10}$$

To determine the constraint we introduce the trivial equation

$$\frac{d\,\hat{\Delta}_r}{d\hat{z}} = 0 \tag{11}$$

and use a forward difference scheme to discretize the ODE over the domain

Our discretized form for the governing equations becomes

$$\frac{\hat{U}_j - \hat{U}_{j-1}}{\Delta \hat{z}} + 2\left(\frac{\hat{f}_{j-1} + \hat{f}_j}{2}\right) = 0$$

$$\hat{U}_j \left(\frac{\hat{f}_{j+1} - \hat{f}_j}{\Delta \hat{z}}\right) + \hat{f}_j^2 = -\hat{\Delta}_j + \frac{1}{\mathbb{R}e}\left(\frac{\hat{f}_{j+1} - 2\hat{f}_j + \hat{f}_{j-1}}{\Delta \hat{z}^2}\right) \tag{12}$$

$$\frac{\hat{\Delta}_{j+1} - \hat{\Delta}_j}{\Delta \hat{z}} = 0$$

with the boundary conditions

$$\hat{f}_1 = 0, \quad \hat{U}_1 = 0, \quad \hat{f}_{NP} = 0, \quad \hat{U}_{NP} = -1 \tag{13}$$

The strategy then is to determine the value $\hat{\Delta}_j$ over the grid such that the boundary conditions are satisfied. The above equations are evaluated on the nodes $j = 2, 3, \ldots,$ NP – 1. In addition the continuity equation is evaluated at j = NP, and the constraint equation is evaluated at j = 1. This generates 3NP-4 equations for the 3NP variables. The four additional equations are the boundary conditions given by Eq. (13).

Mathematica Solution

In our *Mathematica* solution we let

$$U[j] \equiv \hat{U}_j, \quad f[j] = \hat{f}_j, \quad \Delta[j] = \hat{\Delta}_j$$

We will switch off the spelling error message

```
Off[General::"spell1"]
```

The finite difference equations become

```
In[10]:= Remove[U, F];
```

```
In[11]:= FDEqns = {U[j] == -(f[j - 1] + f[j]) Δz + U[j - 1], f[j] (2/(ℝe Δz^2) - U[j]/Δz + f[j]) ==
           (-Δ[j] + (f[j + 1] + f[j - 1])/(ℝe Δz^2)) - (U[j] × f[j + 1])/Δz, (Δ[j + 1] - Δ[j])/Δz^2 == 0};
```

Next we define the parameters for the grid and define a value for the Reynolds number ℝe (Note: we use ℝe not Re as the latter is a defined function in *Mathematica.*)

In[12]:=
```
zmin = 0; zend = 1.; NP = 60; ℝe = 10;
Δz = (zend - zmin)/(NP - 1)
```

Out[13]=
```
0.0169492
```

The values of z on the grid are giving by

In[14]:=
```
mygrid = Table[z[i] → (i - 1) Δz, {i, 1, NP}];
```

We need to define the boundary conditions and we also specify the continuity equation at j = NP and the constraint equation for Δ at j = 1

In[15]:=
```
BCs = {U[1] == 0, f[1] == 0, f[NP] == 0, U[NP] == -1,
    U[NP] == -(f[NP - 1] + f[NP]) Δz + U[NP - 1], (Δ[2] - Δ[1])/Δz^2 == 0};
```

Next we generate the equations on the internal nodes of the grid j = 2 through j = NP – 1 using *Mathematica's* **Table** function:

In[16]:=
```
GridEqns = Table[FDEqns, {j, 2, NP - 1}];
```

Next we combine the two sets of equations using the function **Join**:

In[17]:=
```
FDEqns = Join[GridEqns, BCs] // Flatten;
```

We have 3NP-6=174 equations generated on the grid plus 6 boundary conditions for a total of 180 equations for the 180 variables

In[18]:=
```
Length[FDEqns]
```

Out[18]=
```
180
```

The system of equations is nonlinear; thus we use a Newton's method to solve the set. Hence we will need initial guesses for the variables that appear in our nonlinear set of equations. We generate the initial guesses as follows

In[19]:=
```
Uvar = Table[{U[i], 0.5}, {i, 1, NP}];
fvar = Table[{f[i], 0.5}, {i, 1, NP}];
Δvar = Table[{Δ[i], 0.5}, {i, 1, NP}];
```

We solve the system of nonlinear equations using FindRoot, with the above set of initial guesses.

In[22]:=
```
sol2 = FindRoot[FDEqns, Join[Uvar, fvar, Δvar]];
```

The solution is in a the form of a set of replacement rules for the nodal values. Here is an example of the solution and nodes 10 and 11 for the function $\hat{U}_{10}$, $\hat{U}_{11}$:

In[23]:=
```
sol2[[{10, 11}]]
```

Out[23]=
```
{U[10] → -0.0959745, U[11] → -0.11594}
```

To plot the nodal values $\hat{U}_j$ as a function of $\hat{z}_i$we need to get the solution in an appropriate form so that we can use ListPlot. The form of the solution should be

$$\{\{z_1, \hat{U}_1\}, \{z_2, \hat{U}_2\}, \ldots, \{z_{59}, \hat{U}_{59}\}\}$$

We can achieve this by taking the first NP nodal values out of the solution sol and then combining those values with the location values of the nodes ($\hat{z}_i$) using Transpose to get a list of ordered pairs of rules that have the structure:

```
{{z[1] → zval1, U[1] → Uval1},
 {z[2] → zval2, U[2] → Uval2}, …, {z[60] → zval60, U[60] → Uval60}}
```

We then use the following replacement rule

```
{_→U_,_→z_}→{z,U}
```

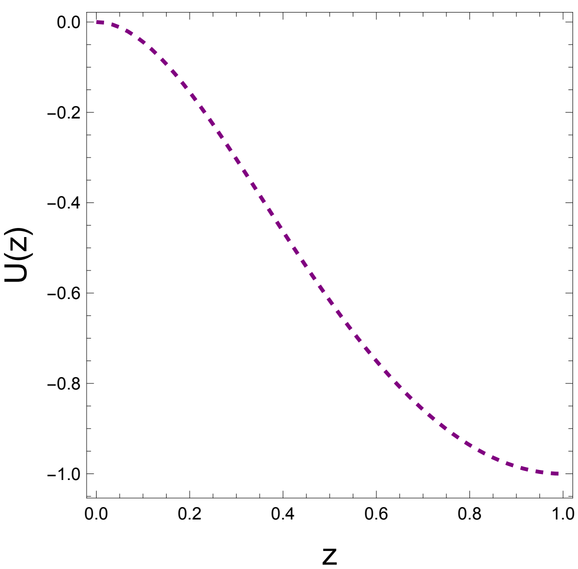
to get a list of ordered pairs of values

```
{{zval1, Uval1,}, { zval2, Uval2}, …, { zval60, Uval60}}
```

Note that the ordering of each ordered pair is flipped for plotting purposes. Here is the result

In[29]:=
```
plt2b = ListPlot[Transpose[{Take[sol2, NP], mygrid}] /. {_ → U_, _ → z_} → {z, U},
  Joined → True, AspectRatio → 1, Frame → True, PlotStyle → {Thick, Purple, Dashed},
  FrameLabel → {Style["z", 16], Style["U(z)", 16]}, ImageSize → 300]
```

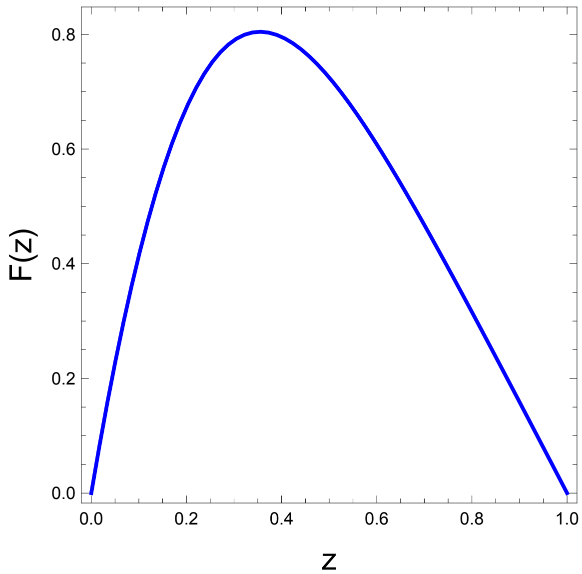
Out[29]=



Note that $\hat{U}(z) = -1$ at $\hat{z}$=1. Next we plot the function $\hat{f}(\hat{z})$

In[25]:=
```
plt2a =
 ListPlot[Transpose[{Take[sol2, {NP + 1, 2 NP}], mygrid}] /. {_ → f_, _ → z_} → {z, f},
  Joined → True, AspectRatio → 1, Frame → True, PlotStyle → {Thick, Blue},
  FrameLabel → { Style["z", 16], Style["F(z)", 16]}, ImageSize → 300]
```

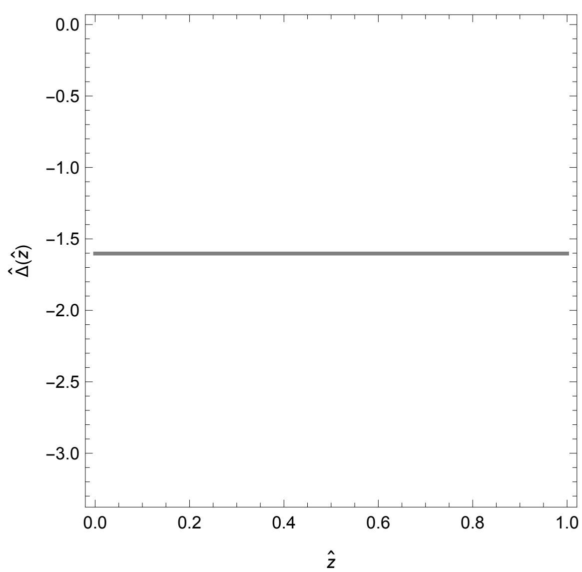
Out[25]=



Note that the boundary condition at $\hat{z} = 1$ is $\hat{f}(1) = 0$. Finally we can plot the value of our eigenvalue, which should have a constant value

In[26]:=
```
plt2c =
 ListPlot[Transpose[{Take[sol2, {2 NP + 1, 3 NP}], mygrid}] /. {_ → Δ_, _ → z_} → {z, Δ},
  Joined → True, AspectRatio → 1, Frame → True,
  PlotStyle → {Thick, Gray}, FrameLabel → {"ẑ", "Δ̂(ẑ)"}, ImageSize → 300]
```

Out[26]=



The value of $\hat{\Delta}_r$ is

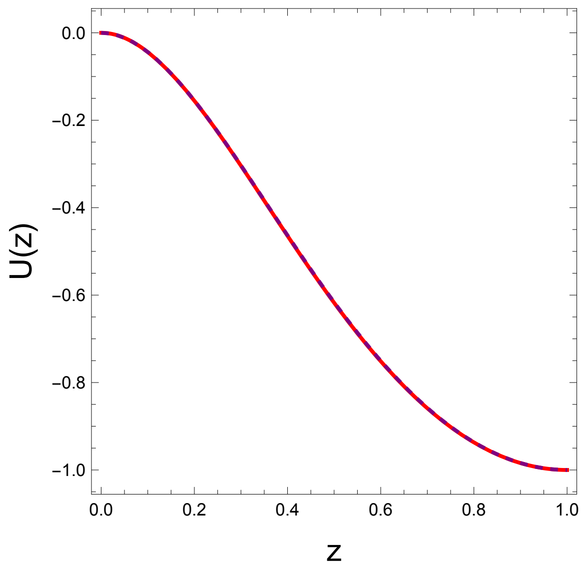
$$\hat{\Delta}_r = -1.60245 \tag{14}$$

which slightly larger than the value determined by the Shooting Method.

Here is a comparison of the U(z) function determined by the two methods. Visually the plots are the same.

In[30]:=
```
Show[plt1b, plt2b]
```

Out[30]=



References

The finite difference algorithm used in these notes was adapted from the textbook of Kee et al.

1. F. Homann, *The effect of high viscosity on the flow around a cylinder and around as sphere*, Z. angew. Math. Mech, 16, 153-64, 1936
2. R. J. Kee, M.E. Coltrin & P. Garborg, *Chemically Reacting Flow. Theory and Practice*, Wiley-Interscience, 2003
3. L. Rosenhead, *Laminar Boundary Layers*, Dover Publications, 1988