Program Development Document

The procedure outlined in this document is intended to guide the design of a program using a top-to-bottom approach. Initially, one describes the nature of the program at the highest level, specifying only what task the program is to perform and noting the physical assumptions and data required to completely define the problem and the solution. Next, one specifies the design choices used to execute the program's tasks, making note of new data that are introduced. These design choices should be as independent of each other as possible, and any possible interactions should be noted.

Until this point in the design process, most attention has been focused on data management. Next, the program execution is designed starting first with an outline of the main program in program development language, i.e. with plain-English descriptions of each operation instead of language-specific code. Once the operations to be performed by the main program have been described, tasks that are to be performed by procedures (subroutines) are identified. The same process used to write the main program is then repeated for these procedures (perhaps iteratively) until the structure of the entire program is complete. At this point, the program development language provides comments to guide the actual coding of the program in the language of choice. The format of this document is intended for non-object oriented programming, but the use of data structures, modules, and internal procedures is supported
.

Project Title :

first, a longer, more descriptive title

Calculation of velocity profile v_x(y) of a shear-thinning polymer fluid in pressure-driven laminar flow between two infinite, parallel plates separated by a distance B, the top one of which is moving at a specified velocity.

next, a short title used to name the directory containing the project's code

polymer_flow_1D

Author :

Kenneth Beers
MIT ChE
10/23/2001

Program Summary :

In this section, describe at the highest level the purpose of the program and the assumptions used at this level. Avoid any discussion of details that are internal to the program. Merely focus on what goes in and what comes out.

This program calculates the velocity profile of a shear-thinning polymer fluid in pressure-driven flow between two infinite, parallel plates separated by a distance B. The bottom plate is stationary and the top plate is moving at a specified velocity in the x-direction. The pressure gradient in the x-direction is specified and is used to drive the flow. Laminar flow conditions are assumed. The constitutive equation employed is that of Yasuda, Armstrong, and Cohen (Rheological Acta, 20, 163-178, 1981).

Program Input and Output Data Specification :

In this section, list the input and output variables of the program that are independent of the internal details of program execution. Only the variables that are required to uniquely specify the problem or that characterize the output at the level of the Program Summary should be listed. For each variable, list the name to be used in the program, the intent (IN, OUT, or INOUT depending on whether it is a program input, an output, or both), the type (REAL, INT, STRING, or a user-defined type and the dimension), and finally a description of the variable's meaning and use within the program. List first the variables that define the dimensions of the other input and output variables. Also, list the input variables first, the input/output variables next, and finally the output variables. In a separate section, list the output variables whose exact form will not be known until further decisions regarding program implementation are made. Denote with an asterisk a dependence of the dimension of an array on the details of the program implementation. To organize the data in a modular fashion, enclose related variables within a DATAGROUP: group_name, ENDDATAGROUP wrapper. If, during later design stages, additional variables are to be included in these DATAGROUPS, they are added to this list with the intent PROG denoting program-wide scope without being a program input or output variable.

Variable Name
Type Intent
Description

Program Input Variables :

DATAGROUP: System
B
REAL IN
The distance between the two parallel plates.

dp_dx
REAL IN
The pressure gradient in the x direction that drives the flow.

velocity_up
REAL IN
The velocity of the upper plate in the x-direction.
ENDDATAGROUP

DATAGROUP: Fluid
visc_0
REAL IN
The zero shear rate viscosity of the fluid.

visc_inf
REAL IN
The limiting viscosity as the shear rate approaches infinity.

relax_t

REAL IN
The relaxation time of the fluid.

n_pow
REAL IN
The power-law exponent of the fluid.

a
REAL IN
The coefficient controlling the shape of the viscosity curve in the cross-over region.
ENDDATAGROUP

Known output variables :

Tentative output variables :

v_x
REAL (*)
The velocity field at steady state.

Program Implementation Design Notes :

In this section, describe the major design choices for performing the task outlined in the Program Summary. Try to make and express each design choice independently. After each new design choice, examine the previous sections for possible interactions and interferences, and note any that occur. In each design section, include a table of the new program variables introduced. If the value of a variable must be input, specify the intent as PIN, the P prefix denoting that the format of this variable is dependent on the details of the program implementation. If a new variable is neither input or output but has scope throughout the program, specify the intent as PROG. If a variable is to be output but in a format that depends on the program implementation, the intent is POUT. All "tentative output variables" from the previous section should now be declared in their form dictated by the program design. Include only those variables absolutely needed to define the state of the system within the main program. At this stage in the design, the organization of the program data is the primary concern.

Section 1. Discretization of PDE:

Central finite differences will be used to convert the partial differential equation for the steady state velocity profile into a set of nonlinear algebraic equations. These are written to account for the spatially-varying viscosity due to the shear-thinning behavior of the fluid.

Variable Name	Type	Intent	Description
DATAGROUP: Grid			
num_pts	INT	PIN	The number of grid points in the y-direction.
y	REAL(num_pts)	PIN	The values of the y coordinate at each grid point. For simplicity, we will use a uniform grid.
ENDDATAGROUP			
v_x	REAL(num_pts)	POUT	The x-direction velocity profile for the shear thinning fluid, calculated numerically.
v_x_Newton	REAL(num_pts)	POUT	The analytical velocity profile obtained for a Newtonian fluid.
viscosity	REAL(num_pts)	POUT	A plot of the viscosity as a function of position for the steady-state velocity profile.

Section 2. Numerical solution of NAE system :

The set of nonlinear algebraic equations will be solved using the MATLAB routine fsolve that is part of the optimization toolkit. As an initial guess of the velocity profile, we will use the analytical solution for a Newtonian fluid under the same circumstances. Since the problem is of large dimension, we will use the large-scale algorithm of fsolve, and will provide a matrix that specifies the sparsity pattern of the Jacobian.

Variable Name
Type Intent
Description

S_Jac
SPARSE (num_pts,num_pts), # non-zero = 3*num_pts PROG
A sparse matrix that contains a non-zero component (e.g. 1) only at those elements at which the Jacobian is thought to be non-zero.

Program Outline in Program Development Language :

In this section, outline the operation of the program in program development language (PDL), i.e. in plain English that is organized similarly to code in a computer program. Avoid any language-specific syntax. One starts with the main program, listing all tasks to be performed and enclosing sets of operations to be performed by a procedure within a wrapper. The input/output interfaces and purposes of each procedure are then written. Then, the operations of the procedures are outlined in program development language, identifying further procedures that must be added. This process is repeated, perhaps iteratively, until the structure of the entire program is complete. The PDL outline developed in this section should satisfactorily comment the final code.

main program PDL :

First, outline the operation of the main program, without attempting to specify the nature of the procedures required to perform the tasks. Then, identify the set of related tasks that are to be performed within a procedure rather than within the main program. Enclose these tasks together within a PROCEDURE : proc_name, ENDPROCEDURE wrapper.

PDL> Prompt the user for the name of the input m-file that sets the values of the simulation parameters. Then, use feval to run this function indirectly. See the PDL code for the function polymer_flow_1D below for the form that this input file should take.

PDL> For these parameters, calculate the analytical velocity profile for a Newtonian fluid and use this profile as an initial guess for the velocity profile with the shear thinning fluid.

PDL> Initialize the options for the numerical solver routine fsolve and set the matrix that specifies the sparsity pattern of the Jacobian. Use the default large scale algorithm and change as needed the level of information that the solver routine displays to the screen.

PDL> Call the MATLAB built-in routine fsolve to numerically solve the set of nonlinear algebraic equations for the velocity profile of the shear-thinning fluid, where the function that calculates the function vector is polymer_flow_1D_calc_f.

PROCEDURE: calc_viscosity_profile
PDL> For the steady-state velocity profile, calculate the viscosity and velocity gradient at each grid point.
ENDPROCEDURE

PDL> Plot the Newtonian and shear thinning velocity profiles, the shear gradient, and the viscosity as a function of y on a master plot

list of procedures :

This section first defines the interfaces to the routines used in the main program. The input, input/output, and output variables to the procedure are listed. To simplify the interface, one can use DATAGROUP: group_name, ALL to signify that all variables within the named DATAGROUP are used in the procedure interface. If only certain variables in a DATAGROUP are to be used, change ALL to ONLY: and list the individual variables. If all variables in a data group are to be used except for only a few, use ALL EXCEPT:. In the PURPOSE: section, describe in plain English what the procedure does – this section will serve as the procedure header in the final code. In the EXISTS: section, if the code for the procedure already exists from a previous project, make a note of where to find it. In the CALLED BY: section, make a list of every procedure in the project that calls this procedure; initially, the main program will be the only listing. In CALLS:, list the other procedures that the current procedure calls; this will generally not be known until the PDL: section, containing the outline of the procedure operation in project development language, has been written. If the procedure is to be grouped with others in a module, enter the name in the MODULE: section. Any internal routines to the procedure are to be added in the CONTAINS: section. The format of each procedure's outline is the following :

PROCEDURE:

INPUT :

INPUT/OUTPUT :

OUTPUT :

PURPOSE :

EXISTS :

CALLED BY :

CALLS :

MODULE :

PDL :

CONTAINS :

PROCEDURE: polymer_flow_1D_input

INPUT :

INPUT/OUTPUT :

OUTPUT :

System : data structure containing system parameters
.B
REAL
The distance between the two parallel plates.
.dp_dx
REAL
The pressure gradient in the x direction that drives the flow.
.velocity_up
REAL
The velocity of the upper plate in the x-direction.

Fluid : data structure containing fluid parameters
.visc_0
REAL
The zero shear rate viscosity of the fluid.
.visc_inf
REAL
The limiting viscosity as the shear rate approaches infinity.
.relax_t
REAL
The relaxation time of the fluid.
.n_pow
REAL
The power-law exponent of the fluid.
.a
REAL
The coefficient controlling the shape of the viscosity curve in the cross-over region.

Grid : data structure containing the computational grid data
.num_pts
INT
The number of grid points in the y-direction.
.y
REAL(num_pts)
The values of the y coordinate at each grid point. For simplicity, we will use a uniform grid.

PURPOSE :

This function sets the simulation parameters for the 1D polymer flow numerical calculation, and is used as the primary means to input data into the simulation.

EXISTS :

CALLED BY :
main

CALLS :

MODULE :

PDL :

PDL> Set the value of the distance between the plates, the pressure gradient, and the velocity of the upper plate.

PDL> Set the parameters that define the constitutive equation relating shear rate to viscosity for the polymer fluid.

PDL> Specify the number of points in the computational grid, and set their locations based on a uniform grid spacing.

CONTAINS :

PROCEDURE: calc_viscosity_profile

INPUT :
v_x
REAL(Grid.num_pts)
A vector of the x-direction velocities at each grid point

Grid : data structure containing the computational grid data
.num_pts
INT
The number of grid points in the y-direction.
.y
REAL(num_pts)
The values of the y coordinate at each grid point. For simplicity, we will use a uniform grid.

Fluid : data structure containing fluid parameters
.visc_0
REAL
The zero shear rate viscosity of the fluid.
.visc_inf
REAL
The limiting viscosity as the shear rate approaches infinity.
.relax_t
REAL
The relaxation time of the fluid.
.n_pow
REAL
The power-law exponent of the fluid.
.a
REAL
The coefficient controlling the shape of the viscosity curve in the cross-over region.

INPUT/OUTPUT :

OUTPUT :
viscosity
REAL(Grid.num_pts)
The value of the fluid viscosity at each grid point

dvx_dy
REAL(Grid.num_pts)
The velocity gradient value at each grid point.

PURPOSE :

This function uses second order finite difference approximations to obtain the velocity gradient at each interior point and at the walls (where 1-sided equations based on Lagrange interpolation must be used). This is written to be compatible with a grid that has non-uniform spacing.

EXISTS :

CALLED BY :
main, polymer_flow_1D_calc_f

CALLS :

MODULE :

PDL :

```
PDL>  For every interior point
FOR igrid = 2 : (Grid.num_pts-1)

***PDL> Use central finite differences to calculate the velocity gradient and store the value

PRODCEDURE: calc_viscosity_polymer
***PDL> Pass the absolute value of this velocity gradient to a function that returns the value of the viscosity at this point based on the constitutive relation and store the value in the appropriate location.
ENDPROCEDURE

PDL> ENDFOR

PDL> Repeat this calculation for each wall grid point using one-sided Lagrange interpolation to estimate the velocity gradient at the wall.  Use the pre-existing Lagrange interpolation routine from the program TR_1D_model1_SS.
```

CONTAINS :

PROCEDURE: calc_viscosity_polymer

INPUT :

gamma_dot
REAL
The local rate of shear.

Fluid : data structure containing fluid parameters
.visc_0
REAL
The zero shear rate viscosity of the fluid.
.visc_inf
REAL
The limiting viscosity as the shear rate approaches infinity.
.relax_t
REAL
The relaxation time of the fluid.
.n_pow
REAL
The power-law exponent of the fluid.
.a
REAL
The coefficient controlling the shape of the viscosity curve in the cross-over region.

INPUT/OUTPUT :

OUTPUT :

viscosity
REAL
The value of the shear rate dependent viscosity.

PURPOSE :

This function calculates the viscosity of a shear-thinning polymer fluid using the constitutive model of Yasuda, Armstrong, and Cohen, Rheol. Acta, v20, 163-178, 1981.

EXISTS :

CALLED BY :
calc_viscosity_profile

CALLS :

MODULE :

PDL :

PDL> Check to make sure that the input shear rate is non-negative.

PDL> Calculate the viscosity for this shear rate according to the constitutive law.

CONTAINS :

PROCEDURE: polymer_flow_1D_calc_f

INPUT :
v_x
REAL(Grid.num_pts)
A vector of the x-direction velocities at each grid point

Grid : data structure containing the computational grid data
.num_pts
INT
The number of grid points in the y-direction.
.y
REAL(num_pts)
The values of the y coordinate at each grid point. For simplicity, we will use a uniform grid.

System : data structure containing system parameters
.B
REAL
The distance between the two parallel plates.
.dp_dx
REAL
The pressure gradient in the x direction that drives the flow.
.velocity_up
REAL
The velocity of the upper plate in the x-direction.

Fluid : data structure containing fluid parameters
.visc_0
REAL
The zero shear rate viscosity of the fluid.
.visc_inf
REAL
The limiting viscosity as the shear rate approaches infinity.
.relax_t
REAL
The relaxation time of the fluid.
.n_pow
REAL
The power-law exponent of the fluid.
.a
REAL
The coefficient controlling the shape of the viscosity curve in the cross-over region.

INPUT/OUTPUT :

OUTPUT :

fval
REAL(Grid.num_pts)
The vector of the function values of each nonlinear equation. A solution makes this vector have all zeros.

PURPOSE :

This function takes as input an estimate of the velocity profile and calculates the value of each algebraic equation.

EXISTS :

CALLED BY :
indirectly by main through fsolve

CALLS :
calc_viscosity_profile

MODULE :

PDL :

```
PROCEDURE: calc_viscosity_profile
PDL> First, use finite differences to calculate the viscosity at each grid point.
ENDPROCEDURE

PDL> For each interior grid point
FOR igrid = 2: (Grid.num_pts-1)

***PDL> Calculate the value of the function for this grid point.

PDL>  ENDFOR

PDL> For each wall grid point, calculate the values of the function enforcing the boundary condition.
```

CONTAINS :