Root Estimation Using the Secant Method

Cameron Bracken
ENGR 325

April 9, 2006

Contents

List of Figures

List of Tables

1 Introduction

Two water storage reservoirs are controlled by an irrigation district. The district wishes to know the pipe diameter necessary to carry water from reservoir 1 at height Z_1 to reservoir 2 at height Z_3 (Figure 1)(Finney 2006).

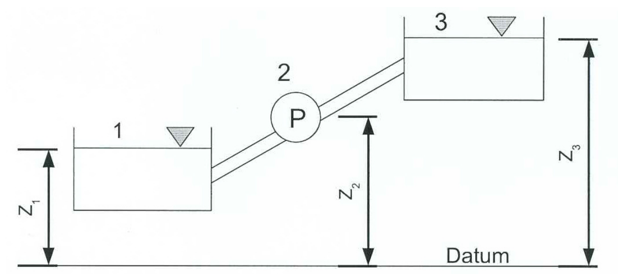

Figure 1: Irrigation water storage system(Finney 2006)

The goal of this exercise is to determine the minimum pipe diameter needed in this situation given pump input power (h_p) and pump efficiency (e).

2 Methodology

The flow of water between the two reservoirs is determined by this general energy equation

$$\frac{p_1}{\gamma} + \frac{v_1^2}{2g} + Z_1 - h_f + E_p = \frac{p_3}{\gamma} + \frac{v_3^2}{2g} + Z_3 \tag{1}$$

where

$$\begin{aligned} p_i &= \text{pressure at point } i \\ \gamma &= \text{specific weight of the fluid} \\ v_i &= \text{fluid velocity at point } i \\ h_f &= \text{friction losses} \\ E_p &= \text{energy head supplied by pump (P)} \end{aligned}$$

By various assumptions, this equation can be shown to simplify to

$$h_f = E_p - h \tag{2}$$

where

$$h = Z_3 - Z_1$$

E_p can be determined from

$$E_p = \frac{76.04e \cdot h_p}{\gamma Q} \tag{3}$$

where

$$\begin{aligned}
Q &= \text{flow rate } (\mathrm{m}^3/\mathrm{s}) \\
h_p &= \text{pump input power (horsepower)} \\
\gamma &= \text{specific weight of water} (\mathrm{kg}/\mathrm{m}^3) \\
e &= \text{pump efficiency} \\
E_p &= \text{pump output head(m)}
\end{aligned}$$

All values in (3) are known or given so E_p can be calculated and then substituting into (2), h_f can be determined.

h_f can be related to other known values including the pipe diameter (d) in the Darcy-Weisbach equation for head loss due to friction by

$$h_f = f\frac{lv^2}{2dg} \tag{4}$$

where

$$\begin{aligned}
h_f &= \text{friction head loss(m)} \\
f &= \text{Darcy-Weisbach friction coefficient } (\mathrm{m}^3/\mathrm{s}) \\
l &= \text{pipe length (m)} \\
v &= \text{fluid velocity (m/s)} \\
d &= \text{pipe diameter (m)} \\
g &= \text{acceleration due to gravity} (9.81\ \mathrm{m/s}^2)\text{(Finney 2006)}
\end{aligned}$$

The Colebrook and White equation further relates the friction coefficient f to pipe diameter d

$$\frac{1}{\sqrt{f}} = -2\log_{10}\left(\frac{k}{3.7d} + \frac{2.51}{Re\sqrt{f}}\right) \tag{5}$$

where

$$\begin{aligned}
k &= \text{absolute roughness of the pipe (m)} \\
Re &= \text{Reynolds number } (vd/\nu) \\
\nu &= \text{fluid viscosity } (\mathrm{m}^2/\mathrm{s})\text{(Finney 2006)}
\end{aligned}$$

The roots of (5) can be determined by rearranging (5) so that it equals 0. Solving (4) for f and substituting into (5) will yield an equation in the unknown, d

$$\frac{Q}{\pi\sqrt{\frac{h_f g d^5}{8l}}} + 2\log_{10}\left(\frac{k}{3.7d} + \frac{.6275\nu}{\sqrt{\frac{2h_f g d^3}{8l}}}\right) = 0 \tag{6}$$

the value of d that satisfies this equation will be the minimum pipe diameter to use in this situation.

(6) has no analytical solution so in order to determine the roots of (6), numerical methods must be employed. Finding the derivative of (6) is not a simple calculation, so Newton's method is an undesirable root finding method. The bisection method is too slow and it may be difficult to bracket the root. The secant method is the best method for numerical root finding in this situation. This method requires only one functional evaluation per iteration, requires no derivative to be taken, and requires the least computational effort of the three methods. The secant method algorithm

takes the form

$$x_{k+1} = x_k - f(x_k)\left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}\right] \quad (7)$$

where

$$\begin{aligned} x_{k+1} &= \text{updated root estimate} \\ x_k &= \text{first root estimate} \\ x_{x-1} &= \text{second root estimate} \end{aligned}$$

In program pipe dim (appendix A) the secant method is utilized to find the root of (6).

Figure 2: sample secant method algorithm taken from program pipe dim

```
do
    xnew=xold-(fxold*((xold-xolder)/(fxold-fxolder)))
              !root update
    fxnew=f(xnew)
    if(stopping criteria met)
      exit
    end if
    fxolder=fxold    !swap values to reduce number
    fxold=fxnew      !of functional evaluations
    xolder=xold
    xold=xnew
end do
```

The values of the variables are swapped after every iteration to reduce the number of functional evaluations. Stopping criteria includes a max iteration test, slow progress test, divergence test, and root found test.

3 Application

The pipe diameter (d) is dependent on the value of various parameters (Table 1) which are know in this problem.

Table 1: Parameters associated with determining pipe diameter

Parameter	**Variable**	**Value**
Change in height	h	10m
Energy head supplied by pump	E_p	15.24m
Flow rate	Q	.3m^3/s
Specific weight of water	γ	998.2kg/m^3
Friction head loss	h_f	5.24m
Pipe length	l	95m
Absolute roughens	k	2.591x10-4m
Fluid viscosity	ν	1.007x10$^-$6m^2/s
Acceleration due to gravity	g	9.81m/s^2)
Epsilon 1	ε_1	10$^-$6
Epsilon 2	ε_2	10$^-$9

To test the sensitivity of the model, different parameters can be varied (Table 2).

Table 2: Variation of Parameters

Run #	**Variable**	**Initial value**	**New value**	**Percent varied**
1	e	60%	66%	10%
2			54%	-10%
3	h_p	100 hp	110	10%
4			90	-10%
5	Q	.3 m^3	0.33	10%
6			0.27	-10%
7	h_f	5.24 m	5.764	10%
8			4.716	-10%
9	l	95 m	104.5	10%
10			85.5	-10%
11	k	0.0002591	0.00028501	10%
12			0.00023319	-10%
13	ε_1	10^{-6}	11^{-6}	10%
14			9^{-7}	-10%
15	ε_2	10^{-8}	11^{-8}	10%
16			9^{-9}	-10%

4 Results

Plugging the values form table 2 into (6) and graphing y as a function of pipe diameter (d) yields a graph with one apparent root (Figure 3). Utilizing the secant method within program pipe dim reveals the root is approximately .3038 m. This value is the minimum pipe diameter that should be used between the two reservoirs.

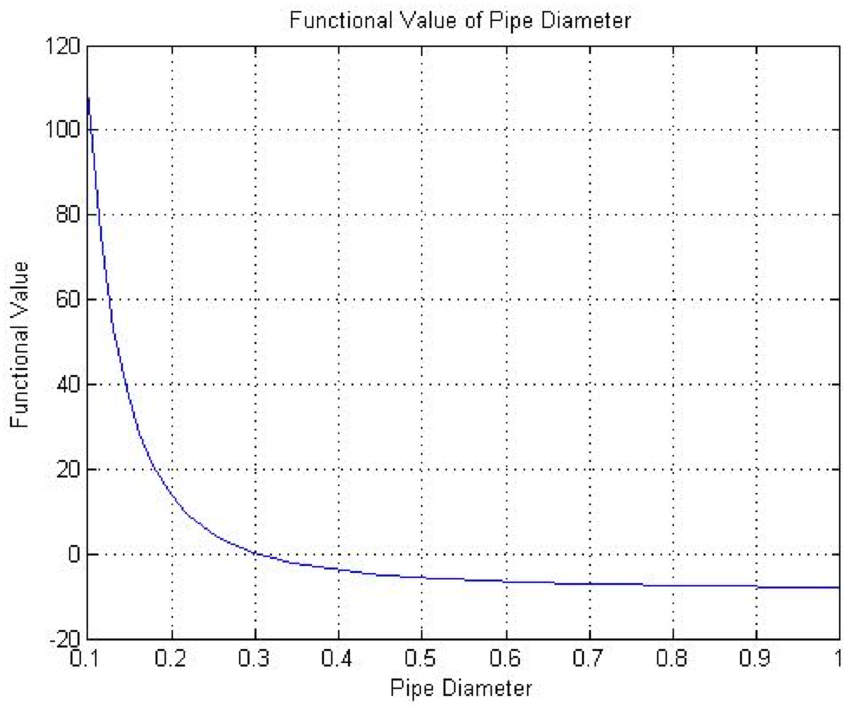

Figure 3: Plot of pipe diameter function

The sensitivity analysis reveals how d changes with the variation of parameters (Table 3).

Varying the efficiency (e) and the horsepower (h_p) affects the the pipe diameter (d) in the same way because they are of equal proportion in (3). 10.08%, The largest variation of the result was seen when varying the flow rate Q by positive 10%. A 10% increase in the flow rate necessitates a 3 cm increase in the pipe diameter. A 10% variation if the absolute roughness (k) has the least effect on the outcome. A variation of both epsilon values (ε_1,ε_2) by $\pm$10% had no effect on the resultant pipe diameter.

Table 3: Results of variation of parameters (all values tested with initial root estimates of .25,.26)

Run #	Variable	New value	Percent varied	Pipe diameter	Variation
1	e	66%	10%	0.2893	4.71%
2		54%	-10%	0.3245	6.88%
3	h_p	110	10%	0.2893	4.71%
4		90	-10%	0.3245	6.88%
5	Q	0.33	10%	0.3342	10.08%
6		0.27	-10%	0.2766	8.89%
7	h_f	5.764	10%	0.3093	1.89%
8		4.716	-10%	0.3214	5.86%
9	l	104.5	10%	0.3091	1.81%
10		85.5	-10%	0.2974	2.04%
11	k	0.00028501	10%	0.3048	0.40%
12		0.00023319	-10%	0.3021	0.49%
13	ε_1	11^{-6}	10%	0.3038	0%
14		9^{-7}	-10%	0.3038	0%
15	ε_2	11^{-8}	10%	0.3038	0%
16		9^{-9}	-10%	0.3038	0%

5 Conclusions

- The minimum pipe diameter that can be used with a 60% efficient, 100hp pump is .3038 m
- The absolute roughness (k) is the least influential parameter
- The flow rate (Q) is the most influential parameter
- A ±10% variation in the stopping criteria, ε_1,ε_2, has no effect on the solution

6 References

Finney,Brad. Lab 7 handout, Humboldt State University, Spring 2006.

Appendix A

Source Code and Program Output

```
module constants
  double precision,parameter::pi=3.1415926,Q=.3,L=95,h=10,specw=998.2,&
                              visc=.000001007,g=9.81,k=.0002591
  double precision::hf
end module constants

Program pipe_dim
```

```
use constants
implicit none
double precision::di,x1,x2,hp,e
integer::tries
logical::success
interface
  subroutine secant(xold,xolder,maxit,epsi1,epsi2,root,numit,exitflag,f)
    double precision,intent(inout)::xold,xolder
    double precision,intent(in)::epsi1,epsi2
    double precision,intent(out)::root
    double precision::xnew,fxnew,fxold,fxolder
    integer,intent(in)::maxit
    integer,intent(out)::numit
    logical,intent(out)::exitflag
    interface
      function f(x)
        double precision::x
        double precision::f
      end function f
    end interface
  end subroutine secant
end interface
interface
  function f(x)
    double precision::x
    double precision::f
  end function f
end interface

!This program will find the optimal pipe diameter between two water
!storage tanks at different heights with a pump in between as described
!in engr 325 lab assignment 7.  To find the pipe diameter this program will
!call a root finding subroutine which will call a function subprogram to
!evaluate the given pipe diameter function.
!
!variable list:
! local variables:
!hf       =friction loss(pump energy-change of potential) (m)
!specw    =specific weight of water (kg/m^3)
!Q        =flow rate(m^3/s)
! inputs:
!hp       =horsepower of pump (horsepower)
!e        =pump efficiency
!x1,x2    =initial pipe diameter guesses (m)
! outputs:
!di       =pipe diameter (m)
!tries    =number of iterations the rootfinding subroutine went through to
!          determine the root
```

```
  !success =value is .true. if root successfully found,else .false.

  write(*,*)"This Program will calculate..."
  write(*,*)"Enter the horsepower of the pump"
  read(*,*)hp
  write(*,*)"Enter the efficiency of the pump"
  read(*,*)e
  hf=(76.04d0*e*hp)/(specw*Q)-h
  !write(*,*)"hf=",hf
  write(*,*)"Enter the first root estimate"
  read(*,*)x1
  write(*,*)"Enter the second root estimate"
  read(*,*)x2
  call secant(x1,x2,20,.00001d0,.00000001d0,di,tries,success,f)
  if(success)then
    write(*,"(a18,x,f10.5)")"The pipe diameter=",di
    write(*,"(a7,x,i2,x,a27)")"It took",tries,"iterations to find the root"
  end if
  stop
end program pipe_dim


subroutine
secant(xold,xolder,maxit,epsi1,epsi2,root,numit,exitflag,f)
  implicit none
  double precision,intent(inout)::xold,xolder
  double precision,intent(in)::epsi1,epsi2
  double precision,intent(out)::root
  double precision::xnew,fxnew,fxold,fxolder     !local variables
  integer,intent(in)::maxit
  integer,intent(out)::numit
  logical,intent(out)::exitflag
  interface
    function f(x)
      double precision::x
      double precision::f
    end function f
  end interface

  !This is a general rootfinding subroutine that employs the secant method
  !it must be used in conjunction with an external function subprogram that
  !will evaluate the function in question
  !
  !variable list:
  ! local variables:
  !xnew      =updated root estimate
  !fxnew     =function evaluated at updated root estimate
  !fxold     =function evaluated at old root estimate
```

```
!fxolder   =function evaluated at older root estimate
! inputs:
!xold      =old root estimate
!xolder    =older root estimate
!epsi1     =stopping criteria for root found, if f(xnew)<epsi1 then
!           root found
!epsi2     =stopping criteria for slow progress, if abs(xold-xolder)<epsi2)
!           then too slow
!maxit     =maxumum allowable iterations subroutine will perform
!           until it exits
! outputs:
!root      =value of particular found root
!numit     =records number of iterations done by subroutine
!exitflag  =value is .true. if root successfully found,else .false.

numit=0
fxold=f(xold)
fxolder=f(xolder)
do
  xnew=xold-(fxold*((xold-xolder)/(fxold-fxolder)))
            !fxold*((xold-xolder)/(fxold-fxolder)) is root update
  fxnew=f(xnew)
  !write(*,*)"fxnew=",fxnew
  numit=numit+1
  if(abs(fxnew)<epsi1)then
    write(*,*)"root found"
    root=xnew
    exitflag=.true.
    return
    exit
  else if(numit>maxit)then
    write(*,*)"No root found (max iterations exceeded), try a better guess."
    exitflag=.false.
    return
    exit
  else if(abs(xold-xolder)<epsi2)then
    write(*,*)"No root found (progress too slow),try better guess."
    exitflag=.false.
    return
  else if ((abs(xnew-xold))>=(abs(xold-xolder)) .and. numit/=1)then
    write(*,"(a38,x,i2,x,a28)")"No root found (solution diverged after",numit,&
                               "iterations),try better guess."
    exitflag=.false.
    return
  end if
  fxolder=fxold    !swap values to reduce number of functional evaluations
  fxold=fxnew
  xolder=xold
```

```
    xold=xnew
    !write(*,*)"fxolder=",fxolder
    !write(*,*)"fxold=",fxold
    !write(*,*)"xolder=",xolder
    !write(*,*)"xold=",xold
  end do
end subroutine secant


function f(d)
  use constants
  double precision,intent(in)::d
  double precision::A,v,fdw,Re,f    !local variables except f

  !This function subprogram evaluates the diameter function below
  !variable list:
  ! local variables
  !A     =cross sectional area of the pipe (m^2)
  !v     =fluid velocity (m/s)
  !fdw   =Darcy-Weisbach friction coefficient (m^2/s)
  !Re    =Reynolds number (dimensionless)
  !pi    =pi (3.1415926)
  !Q     =flow rate(m^3/s)
  !hf    =friction loss(pump energy-change of potential) (m)
  !g     =gravitational constant (9.81 m/s^2)
  !L     =pipe length (m)
  !visc  =fluid viscocity (m^2/s)
  ! inputs:
  !d     =pipe diameter (m)
  ! outputs:
  !f     =function value at given pipe diameter

  A=(pi*d**2d0)/4d0
  v=(4d0*Q)/(pi*d**2d0)
  fdw=(2d0*hf*g*d)/(L*v**2d0)
  Re=(v*d)/(visc)
   !write(*,*)"A=",A
   !write(*,*)"v=",v
   !write(*,*)"fdw=",fdw
   !write(*,*)"re=",Re
  f=-1d0/sqrt(fdw)-2d0*log10(k/(3.7d0*d)+2.51d0/(Re*sqrt(fdw)))
   !f=-d**(2d0)+1  !check function root=1
end function f
```

This was typeset with $\LaTeX$