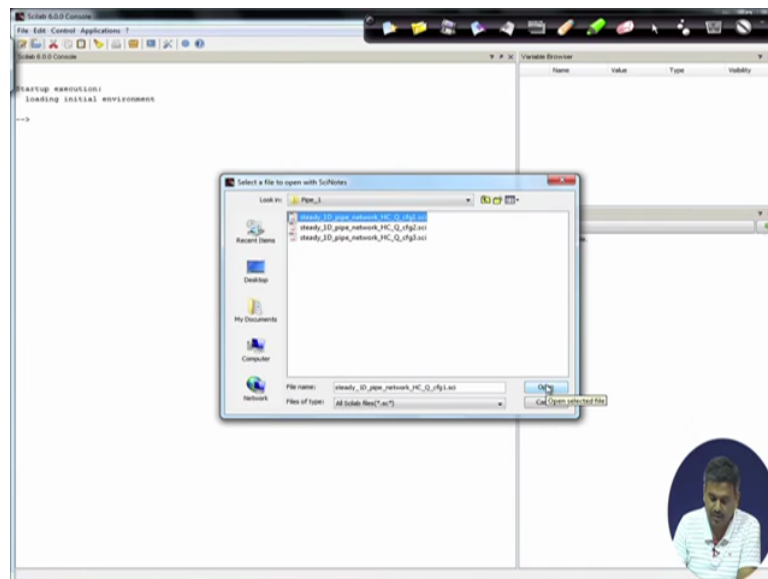


Computational Hydraulics
Professor Anirban Dhar
Department of Civil Engineering
Indian Institute of Technology Kharagpur
Lecture 48
Steady Flow in Pipe Network (Contd.)

So let us start with this steady 1D pipe network HC which is Hardy Cross discharge configuration 1.

(Refer Slide Time: 00:25)



So we will try to utilise the same structure as we have used in case of gradually varied flow. But fundamental difference is that we do not require that much computational effort like our gradually varied flow. Let us start with this clc clear. This was there for gvf but we will not utilise it. This given data g is 9 point 81, global, eps max this is required for convergence, 1 into 10 to the power minus 6.

Beta value, beta it is 2, pipe n is pipe number so we have 8 pipes. Number of maximum, number of pipes connected to a particular loop we have 4. This is required for loop connectivity Matrix that I have discussed already. Now junction number, this junction number is 5. And ploop, ploop is for pseudo loop and iloop is for interior loop. Now loop number which is the total number of loops we have ploop plus iloop. That means number of pseudo loops plus number of interior or internal loops.

(Refer Slide Time: 02:14)

```
1 clear
2 clear
3 -----
4 // Channel Reach: Start + End -
5 // Flow Depth Condition: 1
6 // Flow Rate (Discharge) Condition: 2
7 -----
8 // Given Data
9 g=9.81 //m/s^2
10 global('q')
11 eps_max=1e-6;
12 -----
13 betav=2;
14 pipen=0;
15 nmaxpl=4;
16 junn=5;
17 loop=1; //Interior Loop
18 loop=2; //Interior Loop
19 loopn=loop+loop;
20 -----
21 Qi=[0.319 0.134 0.062 0.019 0.043 0.185 0.035 0.072]; //Initial Discharge
22
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 -1 -2 -3 -4
26           4 2 6 -7 -6
27           3 3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),l)=1;
33     end
34 end
```

Now we have specified the discharge values directly without sign. Please check this thing, I have not used any signs here and we head difference 20 between two this tanks or reservoirs. In this case this is the loop connectivity matrix.

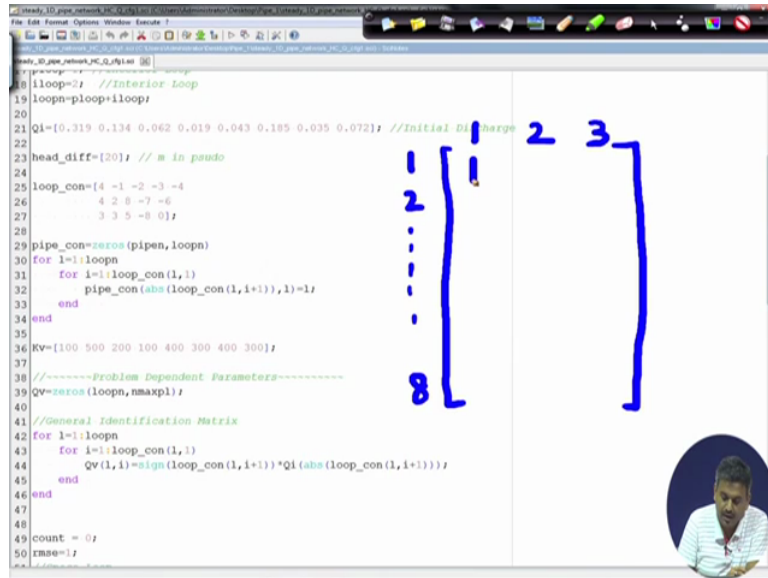
(Refer Slide Time: 02:47)

```
18 loop=2; //Interior Loop
19 loopn=loop+loop;
20 -----
21 Qi=[0.319 0.134 0.062 0.019 0.043 0.185 0.035 0.072]; //Initial Discharge
22
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 -1 -2 -3 -4
26           4 2 6 -7 -6
27           3 3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),l)=1;
33     end
34 end
35
36 Kv=[100 500 200 100 400 300 400 300];
37
38 //-----Problem Dependent Parameters-----
39 Qv=zeros(loopn,nmaxpl);
40
41 //General Identification Matrix
42 for l=1:loopn
43     for i=1:loop_con(l,1)
44         Qv(l,i)=sign(loop_con(l,i+1))*Qi(abs(loop_con(l,i+1)));
45     end
46 end
47
48
49 count = 0;
50 emse=1;
```

Now we need to construct one pipe connectivity matrix. So what is that? That means in starting from pipe 1, 2 up to 8 we need to provide certain information. What is information? Information is that in that matrix we will have first column which is for first loop, second column is second loop, third column is for third loop. Why this is required? This is required

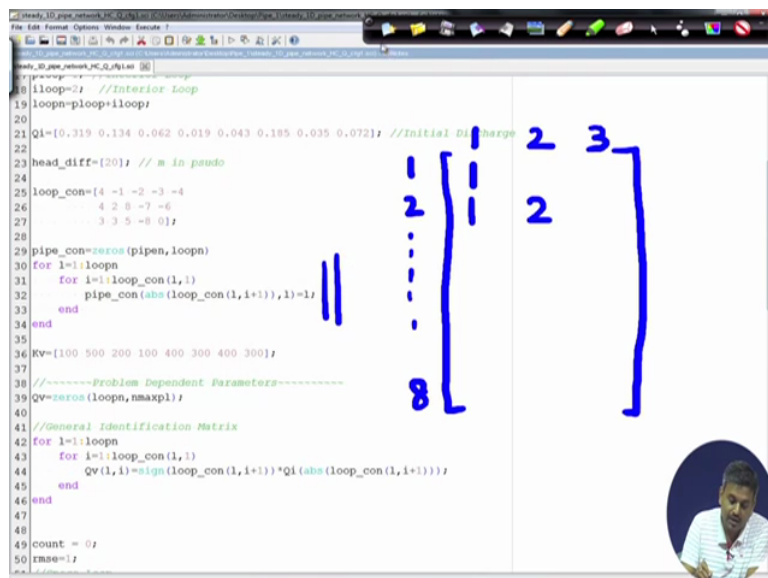
during our delta QL update. So this one it is connected to first loop only. So we will write it as 1.

(Refer Slide Time: 03:47)



Second one it is connected to second one and first one. So I will write it as 1 and 2 both. So like that we can enter the pipe connectivity in a particular loop and we can provide this information during calculation.

(Refer Slide Time: 04:19)



Now we have straightway these Kv values. Kv is nothing but Ki cap.

(Refer Slide Time: 04:31)

```
18 loop=2; //Interior Loop
19 loopn=loop+1loop;
20
21 Q1=[0.319 0.134 0.062 0.019 0.043 0.185 0.035 0.072]; //Initial Discharge
22
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 -1 -2 -3 -4
26           4 2 6 -7 -6
27           3 3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),1)=1;
33     end
34 end
35
36 Kv=[100 500 200 100 400 300 400 300];
37
38 -----Problem Dependent Parameters-----
39 Qv=zeros(loopn,nmaxpl);
40
41 //General Identification Matrix
42 for l=1:loopn
43     for i=1:loop_con(l,1)
44         Qv(l,i)=sign(loop_con(l,i+1))*Q1(abs(loop_con(l,i+1)));
45     end
46 end
47
48
49 count = 0;
50 rmse=1;
```

\uparrow
 K_i

Now again we need to specify this Qv. What is Qv? Zeros loop number. So Lth loop and up to maximum number of pipes connected in a particular loop we need to create this Qv matrix because this is a main matrix that is required for our calculation. Because we need to preserve the sign convention in this matrix itself.

(Refer Slide Time: 05:10)

```
18 loop=2; //Interior Loop
19 loopn=loop+1loop;
20
21 Q1=[0.319 0.134 0.062 0.019 0.043 0.185 0.035 0.072]; //Initial Discharge
22
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 -1 -2 -3 -4
26           4 2 6 -7 -6
27           3 3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),1)=1;
33     end
34 end
35
36 Kv=[100 500 200 100 400 300 400 300];
37
38 -----Problem Dependent Parameters-----
39 Qv=zeros(loopn,nmaxpl);
40
41 //General Identification Matrix
42 for l=1:loopn
43     for i=1:loop_con(l,1)
44         Qv(l,i)=sign(loop_con(l,i+1))*Q1(abs(loop_con(l,i+1)));
45     end
46 end
47
48
49 count = 0;
50 rmse=1;
```

Now for general identification we have this Qv. We need to transfer our initial values. If loop connectivity L i plus 1 which is starting from second column of the loop connectivity matrix, if it is positive then Q should be positive for that loop.

(Refer Slide Time: 05:44)

```
44 end
45
46
47
48
49 count = 0;
50 rmse=1;
51 //space Loop
52 while rmse > eps_max
53     rmse=0;
54     delQ=zeros(loopn,1);
55     for l=1:loopn
56         nr=0;
57         dr=0;
58         if (l<=loopn) then
59             nr=nr+head_diff(l);
60         end
61         for i=1:loop_con(l,1)
62             nr=nr+Kv(abs(loop_con(l,i+1))*Qv(l,i)*abs(Qv(l,i))^(betav-1));
63             dr=dr+betav*Kv(abs(loop_con(l,i+1))*abs(Qv(l,i))^(betav-1));
64         end
65         delQ(l)=-(nr/dr);
66     end
67
```

Otherwise what it means? Let us see our loop connectivity matrix. If loop connectivity for first entry this is negative this sign gives us negative for negative value or minus 1 for negative value, plus one for positive value. So sign multiplied by Qi. Qi value I have specified without any sign so I should input these values within this Qv L i with this sign. So sign should be coming from this loop connectivity matrix directly.

(Refer Slide Time: 06:29)

```
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 -1 -2 -3 -4
26           4 2 6 -7 -6
27           3 3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),1)=1;
33     end
34 end
35
36 Kv=[100 500 200 100 400 300 400 300];
37
38 //-----Problem Dependent Parameters-----
39 Qv=zeros(loopn,nmaxpl);
40
41 //General Identification Matrix
42 for l=1:loopn
43     for i=1:loop_con(l,1)
44         Qv(l,i)=sign(loop_con(l,i+1))*Qi(abs(loop_con(l,i+1)));
45     end
46 end
47
48
49 count = 0;
50 rmse=1;
51 //space Loop
52 while rmse > eps_max
53     rmse=0;
54     delQ=zeros(loopn,1);
55     for l=1:loopn
```

Now with this information or initial information I can start this iteration. Obviously in this case we have only space loop or this loop for iteration. Counter equals to zero, rmse equals to

1 so that I can enter into this while loop. This rmse equals to zero, del QL zeros loopn 1. That means for each loop I have specified one del Q.

(Refer Slide Time: 07:12)

```

46 end
47
48
49 count = 0;
50 rmse=1;
51 //space loop
52 while rmse > eps_max
53     rmse=0;
54     delQ=zeros(loopn,1);
55     for l=1:loopn
56         nr=0;
57         dr=0;
58         if(l<=plopn) then
59             nr=nr+head_diff(1);
60         end
61         for i=1:loop_con(l,1)
62             nr=nr+Kv(abs(loop_con(1,i+1))) *qv(1,i) *abs(qv(1,i))^(betav-1);
63             dr=dr+betav*Kv(abs(loop_con(1,i+1))) *abs(qv(1,i))^(betav-1);
64         end
65         delQ(l)=-(nr/dr);
66     end
67
68
69
70     for l=1:loopn
71         for j=1:loop_con(l,1)
72             for k=1:loopn
73                 if(pipe_con(abs(loop_con(1,j+1)),k) <=0) then
74                     if(pipe_con(abs(loop_con(1,j+1)),k)==1) then
75                         Qv(1,j)=Qv(1,j)+delQ(pipe_con(abs(loop_con(1,j+1)),k));
76                     else
77                         Qv(1,j)=Qv(1,j)-delQ(pipe_con(abs(loop_con(1,j+1)),k));
78                     end
79                 end

```

Now this part is important for our calculation. For L equals to 1 to loop numbers, nr dr numerator and denominator, if L less than equals to plopn. That means less than equal to number of ploops then we need to add this nr value equals to nr plus head difference. The head difference we need to add here directly in the numerator because for our problem only this head difference is there.

(Refer Slide Time: 08:03)

```

46 end
47
48
49 count = 0;
50 rmse=1;
51 //space loop
52 while rmse > eps_max
53     rmse=0;
54     delQ=zeros(loopn,1);
55     for l=1:loopn
56         nr=0;
57         dr=0;
58         if(l<=plopn) then
59             nr=nr+head_diff(1);
60         end
61         for i=1:loop_con(l,1)
62             nr=nr+Kv(abs(loop_con(1,i+1))) *qv(1,i) *abs(qv(1,i))^(betav-1);
63             dr=dr+betav*Kv(abs(loop_con(1,i+1))) *abs(qv(1,i))^(betav-1);
64         end
65         delQ(l)=-(nr/dr);
66     end
67
68
69
70     for l=1:loopn
71         for j=1:loop_con(l,1)
72             for k=1:loopn
73                 if(pipe_con(abs(loop_con(1,j+1)),k) <=0) then
74                     if(pipe_con(abs(loop_con(1,j+1)),k)==1) then
75                         Qv(1,j)=Qv(1,j)+delQ(pipe_con(abs(loop_con(1,j+1)),k));
76                     else
77                         Qv(1,j)=Qv(1,j)-delQ(pipe_con(abs(loop_con(1,j+1)),k));
78                     end
79                 end

```

So i equals to 1 to loop connectivity L 1. L 1 means it gives the entry in the first column. So for the first loop we have four connected pipes. For four connected pipes I can calculate this nr equals to Kv which is K value into Qv into absolute Qv into the power βv minus 1. Again denominator this is dr equals to dr into βv into Kv abs Qv L i to the power βv minus 1. After taking abs I am taking this power.

(Refer Slide Time: 08:57)

```

66 end
67
68
69
70 for l=1:loopn
71     for j=1:loop_con(l,1)
72         for k=1:loopn
73             if(pipe_con(abs(loop_con(l,j+1)),k) <=0) then
74                 if(pipe_con(abs(loop_con(l,j+1)),k)=1) then
75                     Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
76                 else
77                     Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
78                 end
79             end
80         end
81     end
82 end
83
84 count = 0;
85 rmax=1;
86 //space loop
87 while rmax > eps_max
88     rmax=0;
89     delQ=zeros(loopn,1);
90     for l=1:loopn
91         nr=0;
92         dr=0;
93         if(l<=loop) then
94             nr=nr+head_diff(l);
95         end
96         for i=1:loop_con(l,1)
97             nr=nr+Kv(abs(loop_con(l,i+1)))*Qv(l,i)*abs(Qv(l,i))^(betav-1);
98             dr=dr+betav*Kv(abs(loop_con(l,i+1)))*abs(Qv(l,i))^(betav-1);
99         end
100         delQ(l)=(nr/dr);
101     end
102
103     for l=1:loopn
104         for j=1:loop_con(l,1)
105             for k=1:loopn
106                 if(pipe_con(abs(loop_con(l,j+1)),k) <=0) then
107                     if(pipe_con(abs(loop_con(l,j+1)),k)=1) then
108                         Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
109                     else
110                         Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
111                     end
112                 end
113             end
114         end
115     end
116
117     rmax=max(rmax,abs(max(delQ)));
118     count=count+1;
119 end

```

So if I divide this to numerator by denominator this with a negative sign I will get this QL value or $del QL$ value. Now this $del QL$ value is okay but we need to add this to our different discharge values. So how to add this? So let us start with information with a loop connectivity. So three rows. So L equals to 1 to loop number. So 1, 2, 3.

(Refer Slide Time: 09:49)


```

57  qr=0;
58  if (l<=ploop) then
59    nr=nr+head_diff(l);
60  end
61  for i=1:loop_con(l,1)
62    nr=nr+Kv(abs(loop_con(l,i+1)))*Qv(l,i)*abs(Qv(l,i))^(betav-1);
63    dr=dr+betav*Kv(abs(loop_con(l,i+1)))*abs(Qv(l,i))^(betav-1);
64  end
65  delQ(l)=(nr/dr);
66  end
67
68
69
70  for l=1:loops
71    for j=1:loop_con(l,1)
72      for k=1:loopn
73        if (pipe_con(abs(loop_con(l,j+1)),k) <=0) then
74          if (pipe_con(abs(loop_con(l,j+1)),k)=1) then
75            Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
76          else
77            Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
78          end
79        end
80      end
81    end
82  end
83
84
85  for l=1:loops
86    rmse=rmse+delQ(l)^2;
87  end
88
89  rmse=sqrt(rmse/loops);

```

For each loop you can start and I can go or I can iterate from j equals to 1 to loop connectivity L 1. That means for first loop I should use j equals to 1 to 4, for second loop j equals to 1 to 4, but third loop it is j equals to 1 to 3. So I am checking different entries there. 1234, 1234, 123 and this is zero.

(Refer Slide Time: 10:33)

```

57  qr=0;
58  if (l<=ploop) then
59    nr=nr+head_diff(l);
60  end
61  for i=1:loop_con(l,1)
62    nr=nr+Kv(abs(loop_con(l,i+1)))*Qv(l,i)*abs(Qv(l,i))^(betav-1);
63    dr=dr+betav*Kv(abs(loop_con(l,i+1)))*abs(Qv(l,i))^(betav-1);
64  end
65  delQ(l)=(nr/dr);
66  end
67
68
69
70  for l=1:loops
71    for j=1:loop_con(l,1)
72      for k=1:loopn
73        if (pipe_con(abs(loop_con(l,j+1)),k) <=0) then
74          if (pipe_con(abs(loop_con(l,j+1)),k)=1) then
75            Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
76          else
77            Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
78          end
79        end
80      end
81    end
82  end
83
84
85  for l=1:loops
86    rmse=rmse+delQ(l)^2;
87  end
88
89  rmse=sqrt(rmse/loops);

```

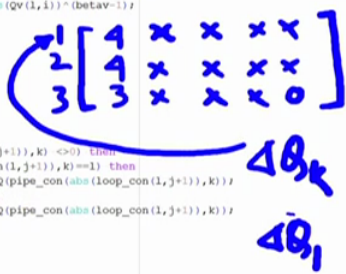
Now in this case next level is K equals to loop number. Now we need to add this del Q value. For del Q I am adding this K index. If K is the index for del Q and it directly matches with 1, let us say we are talking about Q1 and we are in L equals to 1. So I should add that value here.

(Refer Slide Time: 11:13)

```

87  Qr=0;
88  if (l<=ploop) then
89      nr=nr+head_diff(l);
90  end
91  for i=1:loop_con(l,1)
92      nr=nr+Kv(abs(loop_con(l,i+1)))*Qv(l,i)*abs(Qv(l,i))^(betav-1);
93      dr=dr+betav*Kv(abs(loop_con(l,i+1)))*abs(Qv(l,i))^(betav-1);
94  end
95  delQ(l)=(nr/dr);
96  end
97
98  for l=1:loopn
99      for j=1:loop_con(l,1)
100         for k=1:loopn
101             if (pipe_con(abs(loop_con(l,j+1)),k) <=0) then
102                 if (pipe_con(abs(loop_con(l,j+1)),k)=1) then
103                     Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
104                 else
105                     Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
106                 end
107             end
108         end
109     end
110 end
111
112 for l=1:loopn
113     rmse=rmse+delQ(l)^2;
114 end
115
116 rmse=sqrt(rmse/loopn);

```



Otherwise if there is no match between K and L here I should subtract this del Q value because I need to consider or I need to update my discharge in different loops. So after performing this operation I will get the updated Qv values. Obviously in this case we need to calculate the rmse value. What is this rmse? Rmse is equal to already I have used rmse equals to zero. So rmse equals to rmse plus del QL to the power 2.

(Refer Slide Time: 12:05)

```

87  Qr=0;
88  if (l<=ploop) then
89      nr=nr+head_diff(l);
90  end
91  for i=1:loop_con(l,1)
92      nr=nr+Kv(abs(loop_con(l,i+1)))*Qv(l,i)*abs(Qv(l,i))^(betav-1);
93      dr=dr+betav*Kv(abs(loop_con(l,i+1)))*abs(Qv(l,i))^(betav-1);
94  end
95  delQ(l)=(nr/dr);
96  end
97
98  for l=1:loopn
99      for j=1:loop_con(l,1)
100         for k=1:loopn
101             if (pipe_con(abs(loop_con(l,j+1)),k) <=0) then
102                 if (pipe_con(abs(loop_con(l,j+1)),k)=1) then
103                     Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
104                 else
105                     Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
106                 end
107             end
108         end
109     end
110 end
111
112 for l=1:loopn
113     rmse=rmse+delQ(l)^2;
114 end
115
116 rmse=sqrt(rmse/loopn);

```



So I am taking square of all discharge or increment value there. So with this I can calculate rmse. Rmse equals to square root of rmse divided by loop number because for each loop we will have one del Q. So I can display this count and rmse value.

(Refer Slide Time: 12:36)

```

66     end
67
68
69
70     for l=1:loopn
71         for j=1:loop_con(l,1)
72             for k=1:loopn
73                 if(pipe_con(abs(loop_con(l,j+1)),k) <=0) then
74                     if(pipe_con(abs(loop_con(l,j+1)),k)==1) then
75                         Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
76                     else
77                         Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
78                     end
79                 end
80             end
81         end
82     end
83
84     for l=1:loopn
85         rmse=rmse+delQ(l)^2;
86     end
87
88
89     rmse=sqrt(rmse/loopn);
90     count = count + 1;
91     disp([count rmse]);
92     //disp([count max(delyQ)])
93 end
94
95 //Print output
96 for l=1:loopn
97     disp(['Loop Number:' string(l)])
98     disp('Pipe Number - Discharge(m^3/s)')
99

```

If there is convergence obviously this loop this is for while loop this will terminated and I will get the discharge values for different loops, okay. For loops and these are different connected pipes.

(Refer Slide Time: 13:01)

```

74         if(pipe_con(abs(loop_con(l,j+1)),k)==1) then
75             Qv(l,j)=Qv(l,j)+delQ(pipe_con(abs(loop_con(l,j+1)),k));
76         else
77             Qv(l,j)=Qv(l,j)-delQ(pipe_con(abs(loop_con(l,j+1)),k));
78         end
79     end
80 end
81
82 end
83
84
85 for l=1:loopn
86     rmse=rmse+delQ(l)^2;
87 end
88
89
90 rmse=sqrt(rmse/loopn);
91 count = count + 1;
92 disp([count rmse]);
93 //disp([count max(delyQ)])
94 end
95
96 //Print output
97 for l=1:loopn
98     disp(['Loop Number:' string(l)])
99     disp('Pipe Number - Discharge(m^3/s)')
100     for i=1:loop_con(l,1)
101         disp([abs(loop_con(l,i+1)) Qv(l,i)])
102     end
103 end
104
105
106
107

```

Obviously this abs value means the actual global pipe number and Qv is the discharge value with proper sign for that Lth loop.

(Refer Slide Time: 13:17)

```

74         if(pipe_con(abs(loop_con(1,j+1)),k)==1) then
75             Qv(1,j)=Qv(1,j)+delQ(pipe_con(abs(loop_con(1,j+1)),k));
76         else
77             Qv(1,j)=Qv(1,j)-delQ(pipe_con(abs(loop_con(1,j+1)),k));
78         end
79     end
80 end
81 end
82 end
83
84
85 for l=1:Loopn
86     rmse=rmse+delQ(l)^2;
87 end
88
89
90 rmse=sqrt(rmse/Loopn);
91 count = count + 1;
92 disp([count rmse])
93 //disp([count max(delyQ)])
94 end
95
96 //Print output
97 for l=1:Loopn
98     disp(['Loop Number:' string(l)])
99     disp(['Pipe Number : Discharge(m^3/s)'])
100     for i=1:loop_con(l,1)
101         disp([abs(loop_con(l,i+1)) Qv(l,i)])
102     end
103 end
104
105
106
107

```

Now if I run this I am getting negative discharge value which is consistent with my first loop, loop 1 all negative discharge values. Loop 2 I have second pipe which is having positive, eight pipe which is having positive.

(Refer Slide Time: 13:57)

```

for i=1:loop_con(1,1)
    disp([abs(loop_con(1,i+1)) Qv(1,i)])
end

```

Loop Number: 1

Pipe Number	Discharge (m ³ /s)
1.	-0.3184094
2.	-0.1348150
3.	-0.0424709
4.	-0.0184094

Loop Number: 2

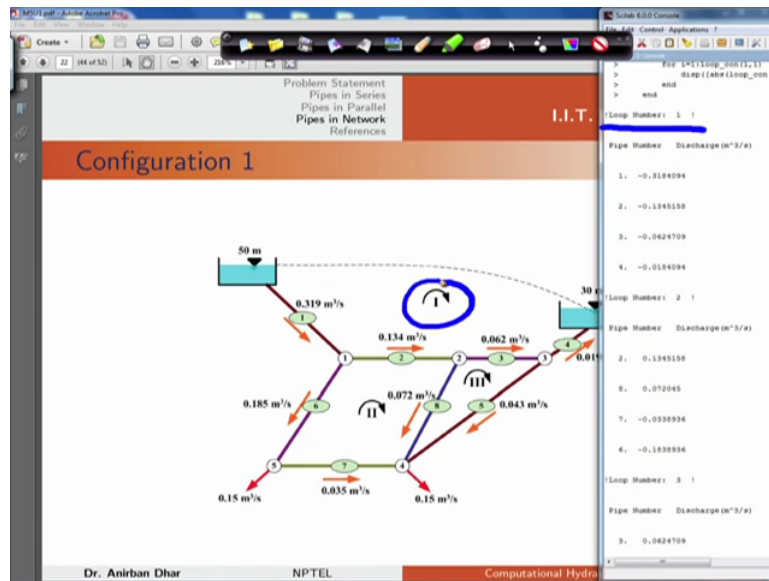
Pipe Number	Discharge (m ³ /s)
2.	0.1348150
8.	0.072045
7.	-0.0389926
6.	-0.1888936

Loop Number: 3

Pipe Number	Discharge (m ³ /s)
3.	0.0424709

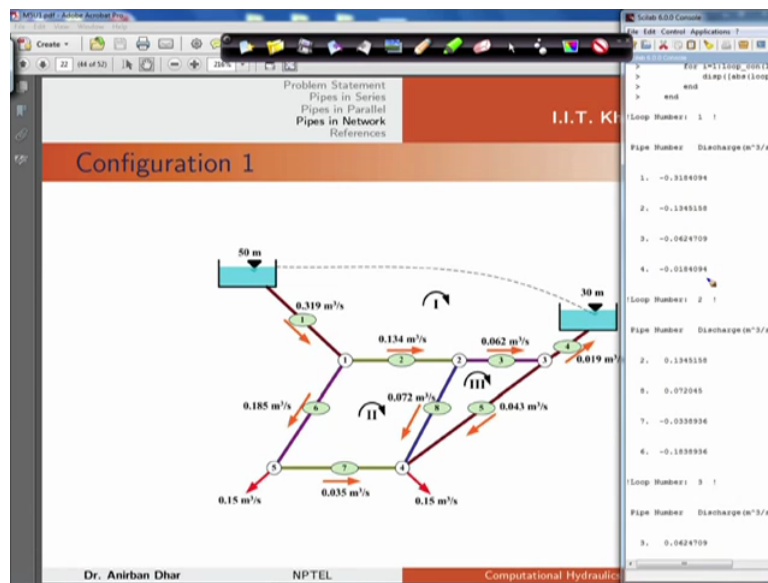
So if I see physically what is the meaning of this one? So here in this case we have loop 1 which is initially loop 1. For loop 1 this is loop 1, for loop 1 we are getting this negative values.

(Refer Slide Time: 14:44)



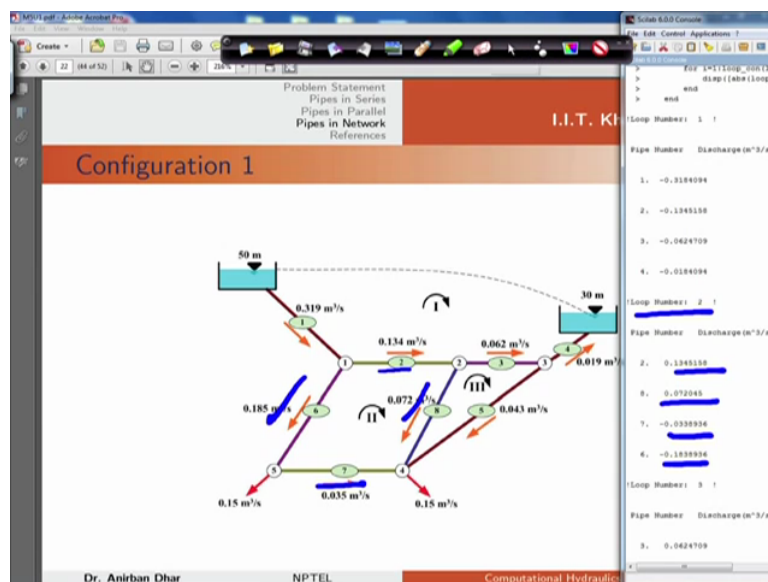
This 3184 we have assumed close values that is why there is no change in values, almost nil. But only change is there for last one. This is point 019, I am getting here as point 0184 which is very close to this one.

(Refer Slide Time: 15:16)



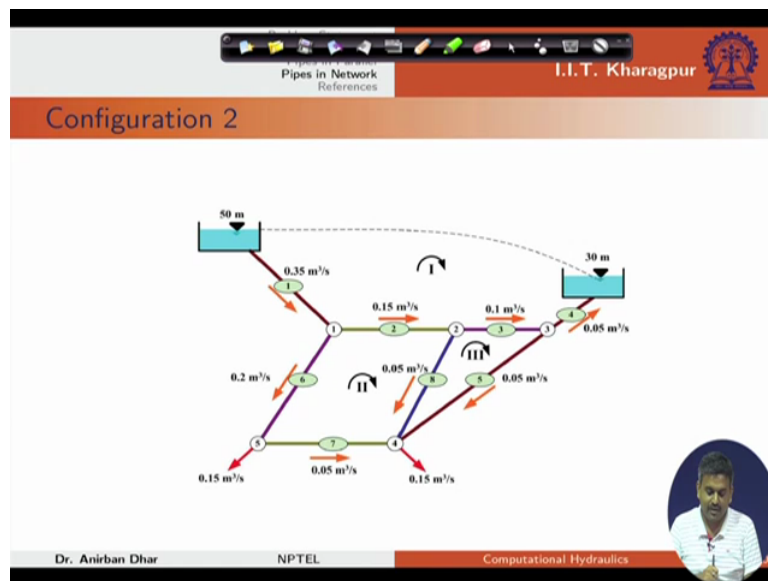
For loop 2 this pipe number 2, pipe number 2 is positive, this is consistent, pipe number 8 again this is positive, this is consistent, pipe number 7 this is negative, pipe number 6 this is negative. Again we are getting the same value here.

(Refer Slide Time: 15:39)



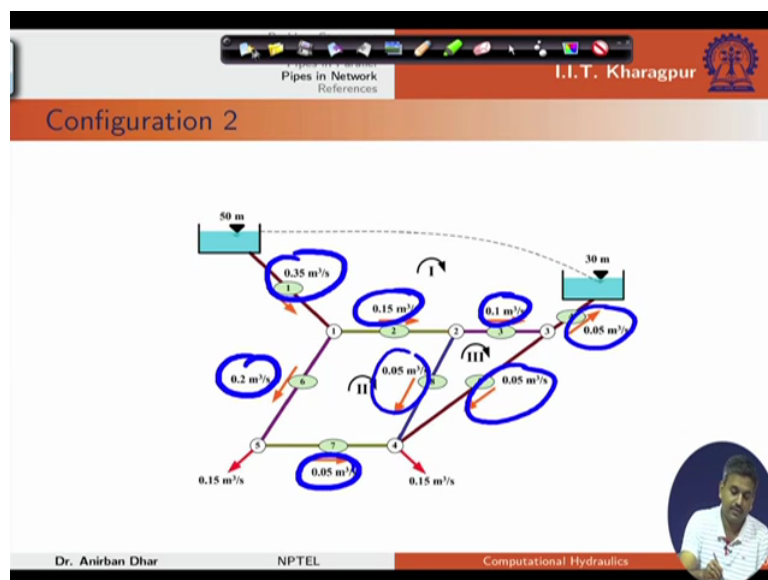
Now we can change the initial distribution of discharge. If I change this initial distribution of discharge then I should get the same value starting from specified initial discharge values. So let us say that this is my configuration.

(Refer Slide Time: 16:11)



So what is the value here? In this case I have changed these values. This is point 35, this is point 2, point 05, point 05, point 05, point 1, point 15, point 05. But one thing is that I have not changed the direction of the flow. Whatever direction I have assumed initially I have calculated my values based on that only.

(Refer Slide Time: 16:50)



Now if I run the same thing with configuration because in this case the configuration part changes. Minus 2, minus 3, minus 4, there is no change. 2, 8 there are no change in that one. Only change is there in terms of magnitude or values of Q. There is no change in this connectivity matrix.

(Refer Slide Time: 17:28)

Pipes in Network
References
I.I.T. Kharagpur

Configuration 2 Loop Connectivity

$$loop_{con} = \begin{bmatrix} 4 & -1 & -2 & -3 & -4 \\ 4 & 2 & 8 & -7 & -6 \\ 3 & 3 & 5 & -8 & 0 \end{bmatrix}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

So for this connectivity matrix or loop connectivity if I open this configuration 2 which is this one, these are the specified discharge values. This point 35, point 15, point 1, point 05 these are initial discharge values and there is no change in the program otherwise.

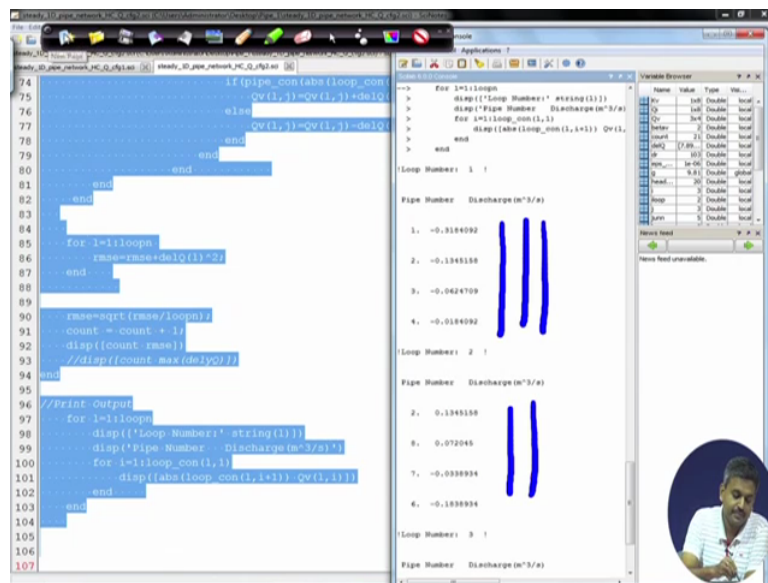
(Refer Slide Time: 18:02)

```
11 eps_max=10^-6;  
12 -----  
13 betav=2;  
14 pipen=0;  
15 nmaxpl=0;  
16 junn=5;  
17 ploop=1 //Interior Loop  
18 lloop=2 //Interior Loop  
19 loopn=ploop+lloop;  
20  
21 Ql=[0.35 0.15 0.1 0.05 0.05 0.2 0.05 0.05]; //Initial Discharge  
22  
23 head_diff=[20]; // m in pseudo  
24  
25 loop_con=[4 -1 -2 -3 -4  
26           4 2 8 -7 -6  
27           3 3 5 -8 0];  
28  
29 pipe_con=zeros(pipen,loopn)  
30 for l=1:loopn  
31     for i=1:loop_con(l,1)  
32         pipe_con(abs(loop_con(l,i+1)),l)=1;  
33     end  
34 end  
35  
36 Kv=[100 500 200 100 400 300 400 300];  
37  
38 -----Problem Dependent Parameters-----  
39 qv=zeros(loopn,nmaxpl);  
40  
41 //General Identification Matrix  
42 for l=1:loopn  
43     for i=1:loop_con(l,1)  
44         Qv(l,i)=sign(loop_con(l,i+1))*Ql(abs(loop_con(l,i+1)));
```

Line 21, Column 43.

So I can again iterate using this method and interestingly I am getting convergence again and discharge values these are again same values are coming. Minus point 318, point 134, point 062. These values are same as we have got from our configuration 1.

(Refer Slide Time: 18:45)



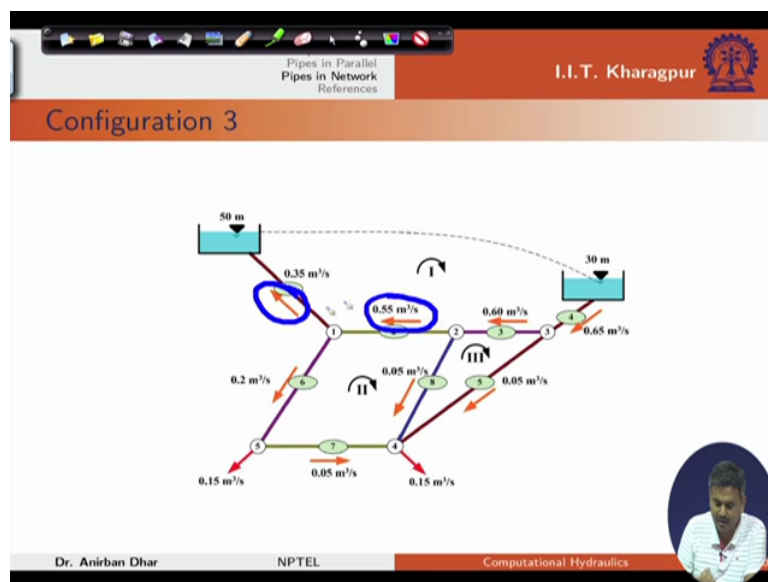
```
74 if(pipe_con(abs(loop_con
75     Qv(i,j)=Qv(i,j)+delQ
76 else
77     Qv(i,j)=Qv(i,j)-delQ
78 end
79 end
80 end
81 end
82 end
83
84 for l=1:loopa
85     rms=rms+delQ(i)^2
86 end
87
88
89 rms=sqrt(rms/loopa)
90 count = count + 1;
91 disp([count rms])
92 //disp([count max(delyQ)])
93 end
94
95
96 //Print output
97 for l=1:loopa
98     disp(['Loop Number:' string(l)])
99     disp(['Pipe Number Discharge(m^3/s)'])
100     for i=1:loop_con(l,1)
101         disp([abs(loop_con(l,i+1)) Qv(i,i)])
102     end
103 end
104
105
106
107
```

Console Output:

```
For l=1:loopa
> disp(['Loop Number:' string(l)])
> disp(['Pipe Number Discharge(m^3/s)'])
> For i=1:loop_con(l,1)
> disp([abs(loop_con(l,i+1)) Qv(i,i)])
> end
Loop Number: 1
Pipe Number Discharge (m^3/s)
1. -0.3184092
2. -0.1345150
3. -0.0424709
4. -0.0184092
Loop Number: 2
Pipe Number Discharge (m^3/s)
2. 0.1345150
3. 0.072048
7. -0.0388934
6. -0.1888934
Loop Number: 3
Pipe Number Discharge (m^3/s)
```

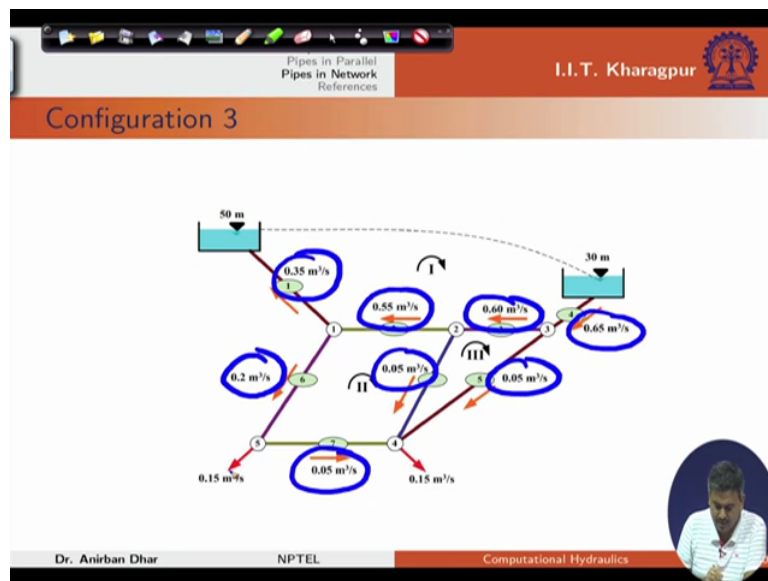
Now let us change this configuration or direction of flow. So let us change this direction. Now in this case let us consider that flow is from right to left. So flow is towards this tank which is having higher elevation.

(Refer Slide Time: 19:13)



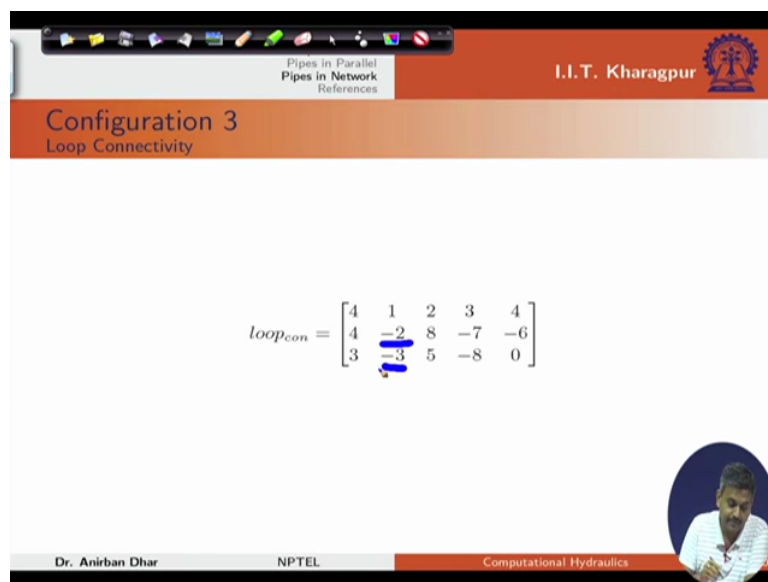
These are the values which I have got by specifying the discharge by maintaining discharge continuity at the junctions. So these are assumed values to maintain the discharge continuity at the junction itself.

(Refer Slide Time: 19:35)



So in this case if I see the connectivity matrix, connectivity matrix I can see that 1234. So these four entries are actually positive entries. That means the flow is clockwise and loop connectivity if I see this is minus 2 and minus 3. These are only two changes here.

(Refer Slide Time: 20:13)



So with this configuration if I try to run my problem using Hardy Cross method I can see in this case we have got convergence with 21 iterations. The convergence parameter was 1 into 10 to the power minus 6 up to that level we have set the accuracy.

(Refer Slide Time: 20:50)

Configuration 3
Loop Connectivity

$$loop_{con} = \begin{bmatrix} 4 \\ 4 \\ 3 \end{bmatrix}$$

Pipe Number	Discharge (m ³ /s)
13.	0.0000296
14.	0.0000184
15.	0.0000118
16.	0.0000074
17.	0.0000047
18.	0.0000029
19.	0.0000019
20.	0.0000012
21.	0.0000007

Handwritten: 1×10^{-6}

Circled value: 0.0000007

So if I open this third configuration or cfg 3, now only change is there for this one, connectivity matrix, this is 1234. These values are there.

(Refer Slide Time: 21:17)

```

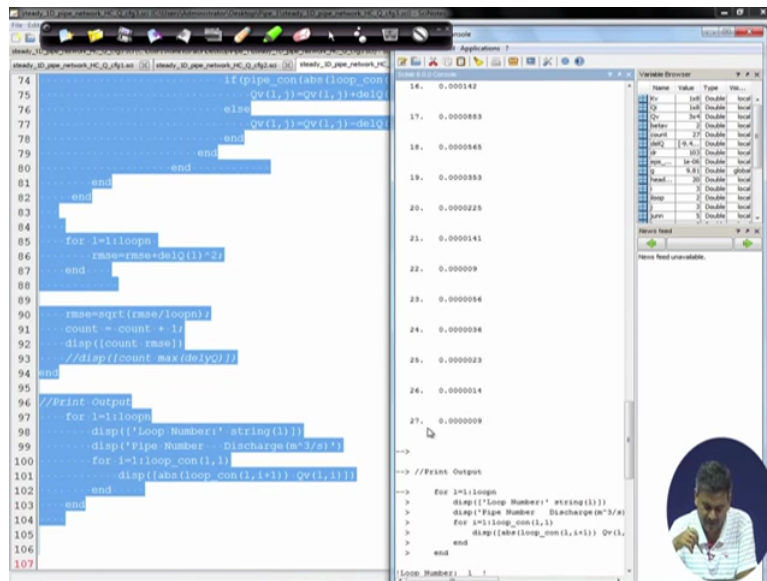
21 Q1=[0.35 0.55 0.6 0.65 0.05 0.2 0.05 0.05]; //Initial Discharge
22
23 head_diff=[20]; // m in pseudo
24
25 loop_con=[4 1 2 3 4
26           4 -2 8 -7 -6
27           3 -3 5 -8 0];
28
29 pipe_con=zeros(pipen,loopn)
30 for l=1:loopn
31     for i=1:loop_con(l,1)
32         pipe_con(abs(loop_con(l,i+1)),1)=1;
33     end
34 end
35
36 Kw=[100 500 200 100 400 300 400 300];
37
38 //-----Problem Dependent Parameters-----
39 Qv=zeros(loopn, nmaxpl);
40
41 //General Identification Matrix
42 for l=1:loopn
43     for i=1:loop_con(l,1)
44         Qv(l,i)=sign(loop_con(l,i+1))*Q1(abs(loop_con(l,i+1)));
45     end
46 end
47
48
49 count = 0;
50 rmse=1;
51 //Space Loop
52 while rmse > eps_max
53     rmse=0;
54     delQ=zeros(loopn,1);

```

Slide 14, Column 8.

So now if I run it I am getting negative value in that here. Again this is 27, 27 is the number of iterations that is required for this one.

(Refer Slide Time: 21:52)



The screenshot shows a MATLAB script on the left and its execution output on the right. The script includes conditional logic for flow direction and iterative calculations for discharge. The console output shows a list of values for 'Pipe Number Discharge (m^3/s)' for pipes 1 through 7.

```
74 if(pipe_con(abs(loop_con
75 Qv(i,j)=Qv(i,j)+delQ
76 else
77 Qv(i,j)=Qv(i,j)-delQ
78 end
79 end
80 end
81 end
82 end
83 end
84 for i=1:loops
85 rmse=rmse+delQ(i)^2
86 end
87 end
88 end
89 rmse=sqrt(rmse/loops)
90 count = count + 1;
91 disp(count rmse)
92 //disp(count max(delQ))
93 end
94 end
95 //Print output
96 for i=1:loops
97 disp(['Loop Number:' string(i)])
98 disp(['Pipe Number Discharge(m^3/s)'])
99 for i=1:loop_con(i,1)
100 disp([abs(loop_con(i,i+1)) Qv(i,1)])
101 end
102 end
103 end
104 end
105 end
106 end
107
```

Console Output:

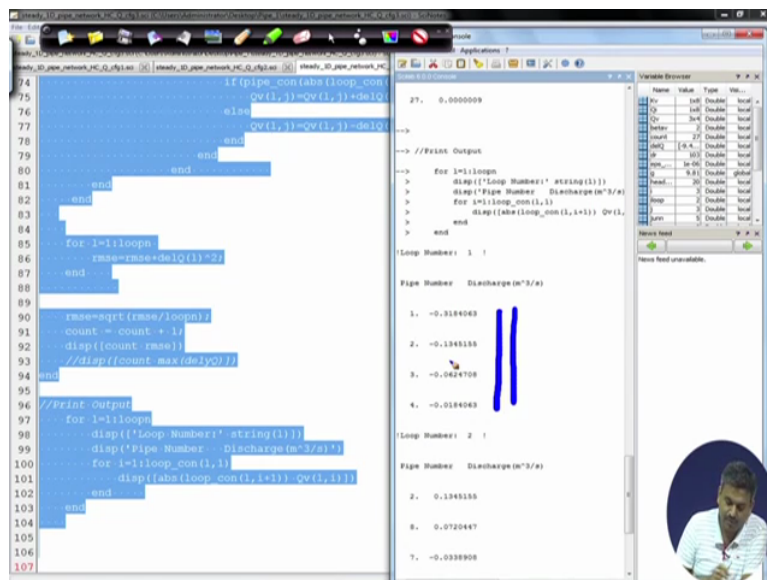
```
14. 0.000142
17. 0.0000883
18. 0.0000845
19. 0.0000383
20. 0.0000225
21. 0.0000141
22. 0.000009
23. 0.0000054
24. 0.0000034
25. 0.0000023
26. 0.0000014
27. 0.0000009
```

Final Output:

```
Loop Number: 1
Pipe Number Discharge (m^3/s)
1. -0.3184063
2. -0.1345155
3. -0.0424708
4. -0.0184063
```

So what it means this negative? Negative means this flow is counter clockwise. We have considered the flow as in clockwise direction but it is showing that whatever value you assume by maintaining the junction continuity you will get the exact direction of the flow.

(Refer Slide Time: 22:17)



This screenshot shows the same MATLAB script as the previous one, but the console output indicates convergence. The final output for 'Pipe Number Discharge (m^3/s)' shows both positive and negative values, with blue vertical lines highlighting the first four pipes (1-4) which have positive discharge values.

```
27. 0.0000009
```

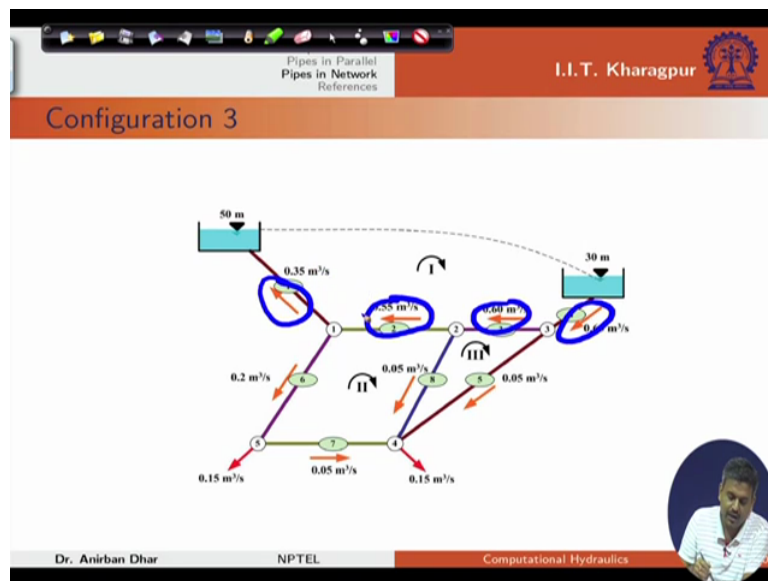
Final Output:

```
Loop Number: 1
Pipe Number Discharge (m^3/s)
1. -0.3184063
2. -0.1345155
3. -0.0424708
4. -0.0184063

Loop Number: 2
Pipe Number Discharge (m^3/s)
5. 0.1345155
6. 0.0720447
7. -0.0389005
```

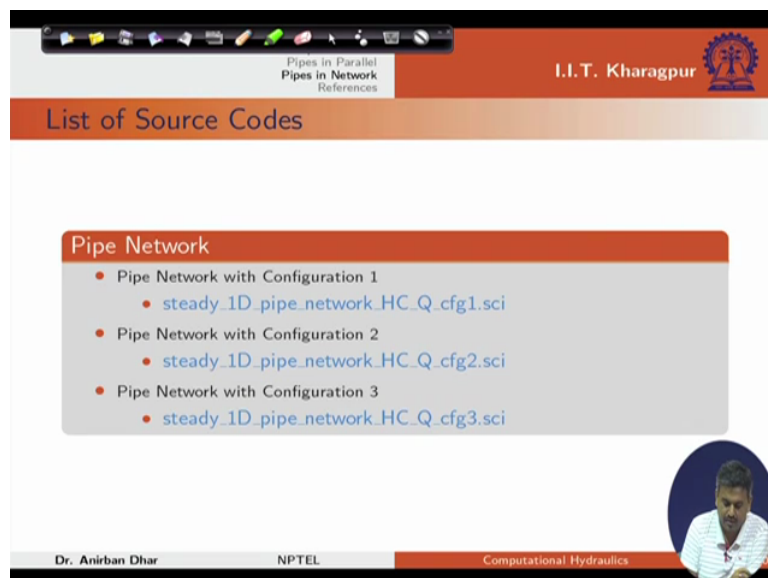
So obviously with the configuration 3 we are getting convergence here and at the same time we can visualize the actual direction of flow from the final results. So this was having a positive value because positive Q for these four.

(Refer Slide Time: 22:44)



But finally when I have got the result, results are showing negative values. That means the flow is just opposite to the assumed direction. So with these three you can change different values and get the desired value for a specific network. You can also change the network configuration for maybe you can change K values or difference in elevation Δh or you can add pump to this problem to see what are the changes occurring during the solution process.

(Refer Slide Time: 23:40)



Obviously I have not included the pump system within this source code. You can suitably modify these codes to add the pump consideration in pseudo loop calculation. Thank you.