Variability in the design storm unit hydrograph model

R. Whitley

Department of Mathematics, University of California, Irvine, CA 92717, USA

T. V. Hromadka II

Williamson and Schmid, 17782 Sky Park Blvd, Irvine, CA 92714, USA

The variability in the maximum discharge is studied for a unit hydrograph model with a fixed design storm by modelling the statistical variation in the unit hydrograph. A program is described which allows the computation of percentiles for the maximum discharge under various probabilistic models describing the random nature of the unit hydrograph.

Key Words: unit hydrograph, uncertainty, statistical variation, probabilistic modelling, maximum discharge.

UNIT HYDROGRAPH MODEL

The calibration of a rainfall-runoff model is uncertain, particularly for a watershed with relatively little rainfall runoff data or undergoing rapid urbanization. Using a chosen design storm, and adjusting this rainfall to obtain the effective rainfall $e(t)$, the unit hydrograph model gives the runoff $Q(t)$, as a function of time t, by the convolution integral

$$Q(t)=\int_0^t e(t-s)u(s)\,ds \tag{1}$$

The major source of variability in this method of predicting the maximal discharge lies in the choice of the unit hydrograph u. There may be several hydrographs for the catchment, associated with various appropriate 'large' storms. Also several hydrographs may be available from other catchments which are similar to the one which is being analysed; these provide useful information for a gauged catchment and the only information for an ungauged catchment.

AN S-GRAPH MODEL

In Hromadka[1] a method was developed to model this uncertainty in the unit hydrograph. For a given mass curve M, which is one of those to be used for predicting the catchment response, the associated unit hydrograph u is defined by

$$M(t)=\int_0^t u(s)\,ds \tag{2}$$

This mass curve comes from a storm with ultimate discharge U and a lag L. (The lag L can be estimated from the time of concentration Tc of the catchment and regression equation $L=0.80Tc$, for which see Hromadka[1] and McCuen *et al.*[2]). When the mass curve is divided by U and scaled by the change of variable $t=\tau L$, the resulting curve $C(\tau)=M(\tau L)/U$, increases from 0 to 1 and has $C(1)=1/2$. When this scaling was studied in Hromadka[1] for a specific flood control project for Orange County, California, the scaled mass curves were found to have similar shapes that could be modelled by

$$C(\tau)=XC_1(\tau)+(1-X)C_2(\tau) \tag{3}$$

where C_1 and C_2 are curves obtained from mass curves for specific catchments, but in general could be enveloping curves for the family of all scaled mass curves, and X is a parameter in $[0,1]$.

The main feature of equation (3) is the simplification of the study of the statistical variation in the family of all curves C to the study of the statistical variation in X. An empirical distribution for X was obtained in Hromadka[1] from a histogram of the X's for the family of curves.

The assumption that any relevant mass curve M can be approximated by

$$M(\tau L)/U=XC_1(\tau)+(1-X)C_2(\tau) \tag{4}$$

is a useful formulation of the random variation in unit hydrographs.

DISCHARGE VARIATION

From equations (1), (2), and (4), the discharge Q can be written as

$$Q(t)/U=\int_0^t e(t-s)\{XC_1'(s/L)+(1-X)C_2'(s/L)\}\,ds/L \tag{5}$$

The criterion variable of interest will be the maximum discharge max$Q$. By means of the approximations encapsulated in (4), the complicated statistical variation in max$Q$, due to the uncertainty in the unit hydrograph u, has been written in terms of the variation in the three real-valued random variables X, L, and U. Now L, U, and X are probably dependent, but the data is insufficient to determine the nature of this dependency. However, the distribution of max$Q$ caused by the variation in each one of the three variables separately can be studied as can the results of joint independent variation.

In this model, the variation in max$Q$ due to the variation in U is merely that due to a multiplicative factor, and so is readily understood. Writing

$$Q(t)/U = X\left\{\int_0^t e(t-s)C_1'(s/L)\,ds/L\right\} + (1-X)\left\{\int_0^t e(t-s)C_2'(s/L)\,ds/L\right\} \tag{6}$$

shows a simple dependence of Q on X. The dependence of max$Q$ on X can be more complicated, but for the specific example analysed below it is not. The dependence on L is the most complicated, and is illustrated below.

COMPUTING DISCUSSION

A Fortran program, maxqt, whose listing is given in section below, simulates the variation of the maximum discharge, due to variation in the variables X and/or L for a fixed given effective rainfall. The effective rainfall used is a simple triangle: increasing linearly from 0 to 1 as t goes from 0 to 1 and decreasing linearly from 1 to 0 as t goes from 1 to 4.

The curves $C_1(\tau)$ and $C_2(\tau)$ were read from Fig. 5 in Hromadka[1] and a piecewise linear approximation, described in the function Q in the program listing, was obtained for each curve. Then each derivative

$$C_i'(t) = \sum_{j=1}^{n} \text{slope}i(j)\chi_{(t_{j-1},t_j)}(t) \tag{7}$$

where slope$i(j)$ is the slope for curve i, $i=1,2$, on the interval (t_{j-1}, t_j) where it is linear, and

$$\chi_{(a,b)}(t) = \begin{cases} 1 & \text{if } x \text{ is in } (a,b) \\ 0 & \text{if } x \text{ is not in } (a,b) \end{cases}$$

The convolution (5) can be written

$$Q(t)/U = \sum_{j=1}^{n} \{X\,\text{slope1}(j) + (1-X)\,\text{slope2}(j)\} \int_{a_j}^{b_j} e(s)\,ds/L \tag{8}$$

with $a_j = t - Lt_j$ and $b_j = t - Lt_{j-1}$. Formula (8) is used in the program function Q to compute $Q(t)$.

In the main body of the program, the user can choose probability distributions to describe the variation in X and L. After these choices are made, the user chooses the number of values of max$Q$ to be simulated. For each simulated value of X and L, the program computes max$Q$ in the subroutine getmax(max$Q$, max$t$), by means of interval halving, and returns the value max$Q$ of the maximum and the point max$t$ at which the maximum occurs. This is the simplest way to allow for variations in both X and L. For variations in X only, equation (6) shows that Q does not need to be repeatedly computed for each value of X, and this fact could be used to speed up simulations of max$Q$ in which only X varied.

The computed value of max$Q$ is then scaled by scale factors which have been computed in the subroutine scale, and the scaled value is used to update a histogram with 400 equally spaced divisions in the subroutine tally. When all the required values of max$Q$ have been simulated, linear interpolation in the histogram, by the subroutine calcpercent, gives the percentile values of max$Q$ which are printed by the subroutine printpercent.

RESULTS

The simulation of X from taken from the empirical histogram and $L=1$ not varying (and taking $U=1$ or, equivalently, thinking of computing max(Q/U)) gives easily interpreted results. The points max$t$ at which the maximum occurs are found to be 2.33 for the percentiles 5%(5%)70% and 2.34 for the percentiles 75%(5%)95%. Write equation (5) in the form

$$Q(t)/U = XQ_1(t) + (1-X)Q_2(t)$$

Computing and graphing $Q_1(t)$ and $Q_2(t)$ shows that both of these functions obtain their maximum approximately at $t=2.33$. *Consequently*,

$$\begin{aligned} \max Q/U &= XQ_1(t_0) + (1-X)Q_2(t_0) \\ &= X(Q_1)(t_0) - Q_2(t_0)) + Q_2(t_0) \end{aligned} \tag{9}$$

where $t_0 = 2.33$. Thus for any X, maxQ/U is approximately just a linear combination $\alpha X + \beta$, and so has a distribution which can be readily calculated.

Table 1.

Percentile	Values of max$Q$ Simulated values	Computed values
10%	71.99	72.01
20%	72.99	73.07
30%	77.13	77.33
40%	77.99	78.04
50%	78.64	78.66
60%	79.04	79.06
70%	79.45	78.46
80%	79.93	79.97
90%	80.69	80.68

Table 2.

Percentile	Value of max$Q$	Point of max$Q$
10%	54.77	1.79
20%	61.94	1.95
30%	65.57	2.10
40%	68.32	2.23
50%	70.53	2.34
60%	72.52	2.48
70%	74.87	2.64
80%	77.73	2.86
90%	80.82	3.33

Table 3.

Percentile	Value of max$Q$	Point of max$Q$
10%	62.60	1.78
20%	68.57	1.93
30%	72.28	2.08
40%	75.12	2.22
50%	77.62	2.34
60%	79.65	2.47
70%	81.68	2.65
80%	84.11	2.88
90%	87.03	3.40

As an example, for the case where X has the empirical distribution, the simulated percentile values obtained for max$Q$ from a simulation of 5000 values are computed in Table 1 below with the calculated percentile values of $24.07X + 58.67$ obtained by computing $Q_1(2.33) = 82.74$ and $Q_2(2.33) = 58.67$ and using the percentile value from the histogram for X in equation (9).

As an example of finding percentiles for max$Q$ from a simulation where L varies, Table 2 below gives the results for a 1000 point simulation of L where L is chosen to have the empirical histogram distribution and X is chosen to have the fixed value $X = 1/2$ corresponding to the average unit hydrograph.

The final example given will be the case where both X and L vary independently according to their empirical distributions.

Note that as a measure of dispersion, the 80th percentile minus the 20 th percentile is, in Tables 1, 2 and 3: 6.94, 8.56, 15.09. Thus there is somewhat more variation due to L than due to X and, of course, the most variation in max$Q$ is when both L and X vary. By comparing Tables 2 and 3, the variation in L entirely controls the variation in the time when the maximum Q occurs, which is in agreement with the values in Table 1.

CONCLUSIONS

For a unit hydrograph model with a fixed design storm, the statistical distribution of maximum discharge can be modelled as a function of the random variables: ultimate discharge U, lag L, and a random variable X related to the shape of the S-graph. The percentiles for the maximum discharge can be obtained by simulation for specific distributions for U, L, and X.

REFERENCES

1 Hromadka II, T. V. Use of a Hydrologic Model as a Flood Control Policy Statement, *Environmental Software*, June 1987, 2(2)

2 McCuen, R. H., Rawls, W. J. and Wong, S. L. SCS Urban Peak Flow Methods, *ASCE Journal of Hydraulic Engineering*, March 1984, 290–299

PROGRAM LISTING

```
      program maxqt
c
c     R.J. Whitley, Math. Dept., University of Calif., Irvine, CA 92717
c     T.V. Hromadka II, Williamson and Schmid, 17782 SkyPark Blvd,
c          Irvine, CA 92714
c
      Real Lsigma,maxt,maxQ,mid(2),percent(2,19),range(2),temp,Xsigma
      Integer answer,Block,count(2,4ØØ),i,j,NoBlocks,NoL,NoX
      parameter (Block=1ØØØ)

      common/histograms/mid,range,count,percent
      common/Xparameters/NoX,Xsigma
      common/Lparameters/NoL,Lsigma


      print *, 'The random S graph is X*C1+(1-X)*C2, C1 and C2 given ',
     &         'S graphs'
      print *,'Choose a distribution for X by number '
      print *,'1. X=Ø.5'
      print *,'2. X normal N(1/2,sigma).'
      print *,'3. X as in 2 but truncated to lie in [Ø,1].'
      print *,'4. X uniform on [Ø,1].'
      print *,'5. X given by an empirical histogram.'
      read *, NoX
      if ((NoX .eq. 2) .or. (NoX .eq. 3)) then
           print *, 'input sigma for X'
           read *,Xsigma
      endif
      print *,'Choose a distribution for the lag L by number '
```

```
      print *,'1. L=1.Ø.'
      print *,'2. L normal N(1,sigma).'
      print *,'3. L as in 2 but truncated to lie in [.5,1.5].'
      print *,'4. L uniform on [.5,2.Ø].'
      print *,'5. L given by an empirical histogram.'
      read *, NoL
      if ((NoL .eq. 2) .or. (NoL .eq. 3)) then
           print *, 'input sigma for L'
           read *,Lsigma
      endif
      temp=rrand()
c     sets seed for random number generator

      print *,'Now Scaling'
      call scale()
c     subroutine Scale computes scale factors used by subroutine tally

      print *,'Input number of blocks in simulation.  '
      read *, NoBlocks
      do 2Ø i=1,2
           do 1Ø j=1,4ØØ
                count(i,j)=Ø
 1Ø        continue
 2Ø   continue
c     count(i,j) is the histogram count; see subroutine tally

      print *,' Completed block numbers:'
           do 4Ø i=1,NoBlocks*Block
                call getmax(maxQ,maxt)
                call tally(1,maxQ)
                call tally(2,maxt)
                if (mod(i,Block) .eq. Ø) then
                     write(*,'("&",I4)') int(i/Block)
                endif
 4Ø        continue

      print *
      print *,'Print out data histograms (1=yes,Ø=no).  '
      print *,'WARNING. The histogram is 4ØØ lines of data.'
      read *, answer
      if (answer.eq.1) call printhistograms(NoBlocks,Block)
      call calcpercent(NoBlocks,Block)
      call printpercent(NoBlocks,Block)

      stop
      end


      subroutine getmax(maxQ,maxt)
c     returns the value maxt, accurate to within
c     (B-A)/(2 to the power 11), at which the (local) maximum value
c     maxQ, which is also returned, is attained. The max is found by
c     interval halving. For the Q's considered in the program, this is a
c     unique max which therefore coincides with the local max maxQ.
c     The function Q is external to this routine.

      real a,b,h,maxQ,maxt,Q,rvL,rvX,X(3),Y(3)
      integer i,j

      call getX(rvX)
      call getL(rvL)
      A=Ø.Ø
      B=5+4/rvL
```

```
c     P is zero outside (Ø,4/rvL) and u is zero outside (Ø,5), so Q,
c     which is the convolution of P and u, is zero outside (Ø,5+4/rvL).

      X(2)=(B+A)/2

      Y(2)=Q(rvX,rvL,X(2))
      h=(B-A)/4

      do 1Ø i=1,1Ø
         X(1)=X(2)-h
         X(3)=X(2)+h
         Y(1)=Q(rvX,rvL,X(1))
         Y(3)=Q(rvX,rvL,X(3))

         j=1
         if ((Y(2) .ge. Y(1)) .and. (Y(2) .ge. Y(3))) j=2
         if ((Y(3) .ge. Y(1)) .and. (Y(3) .ge. Y(2))) j=3

         X(2)=X(j)
         Y(2)=Y(j)
         h=h/2

   1Ø continue
      maxQ=Y(2)
      maxt=X(2)

      end

      function Q(rvX,rvL,t)
c     Q is actually the scaled function Q(.)/U, U ultimate discharge.
c     The main hypothesis is that the mass curve M(.), when scaled by
c     multiplying the argument by the lag L=rvL and dividing by U can be
c     written as
c                  M(t*L)/U=rvX*C1(t)+(1-rvX)*C2(t).                (1)
c     (See Hromadka, T., Use of a Hydrologic Model as a Flood Control
c     Policy Statement, Environmental Software, 2(1987) 6Ø-7Ø.) The
c     discharge Q is given by the convolution of the effective rainfall
c     P with the derivative M' of M. We take the curves C1 and C2 to be
c     piecewise linear on the intervals (tc(i-1),tc(i)),for i=1,..., NC;
c     as such they are piecewise linear approximations to the curves
c     given in the reference above. (For these variables, see the
c     description of the curves C1 and C2 given in this subroutine.)
c     From (1), M'(t) is a step function and the convolution giving Q is
c     a sum of the terms
c             (rvX*slope1(i)+(1-rvX)slope2(i)) times Z
c     where Z is the integral from t-L*tc(i) to t-L*tc(i-1) of P(s)ds,
c     after a change of variable.

      integer Nc
      parameter(Nc=22)
      real integralP,Q,rvL,rvX,slope1(Nc),slope2(Nc),sum,t,tc(Ø:Nc),temp
      integer i

c     set parameters for curves C1 and C1
      data tc/Ø.Ø,Ø.222222,Ø.444444,Ø.666666,Ø.888888,1.111111,1.333333,
     &        1.555555,1.777777,2.ØØØØØØ,2.222222,2.444444,2.666666,
     &        2.888888,3.111111,3.333333,3.555555,3.777777,4.ØØØØØØ,
     &        4.222222,4.444444,4.666666,4.888888/
      data slope1/9,27,63,72,9Ø,76.5,81,22.5,4.5,4.5,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
     &           Ø,Ø,Ø,Ø/
      data slope2/4Ø.5,72,58.5,4Ø.5,36,31.5,27,22.5,22.5,18,13.5,13.5,
     &           9,9,4.5,4.5,4.5,4.5,4.5,4.5,4.5,4.5/
c     end of curve parameters
```

```
      sum=Ø
      do 1Ø i=1,Nc
           temp=rvX*slope1(i)+(1-rvX)*slope2(i)
           temp=temp*integralP(t-(tc(i)*rvL),t-(tc(i-1)*rvL))
           sum=sum+temp
   1Ø continue
      Q=sum/rvL
      end

      function integralP(a,b)
c     The curve P increases linearly from Ø at pØ to a value of maxP at
c     p1, and decreasing linearly to Ø at the point p2. This subroutine
c     computes the integral from Ø to b and subtracts the integral
c     from Ø to a.

      real a,b,c(2),integral(2),integralP,maxP,m1,m2,pØ,p1,p2
      integer i
      parameter (pØ=Ø.Ø,p1=1.Ø,p2=4.Ø,maxp=1.Ø )

      m1=(maxp-Ø)/(p1-pØ)
      m2=(Ø-maxp)/(p2-p1)

      integral(1)=Ø.Ø
      c(1)=a
      integral(2)=Ø.Ø
      c(2)=b
      do 1Ø i=1,2
           if (c(i) .le. pØ) integral(i)=Ø.Ø
           if ((c(i) .gt. pØ) .and. (c(i) .le. p1)) then
            integral(i)=(Ø.5)*(c(i)-pØ)*(m1*(c(i)-pØ))
           endif
           if ((c(i) .gt. p1) .and. (c(i) .le. p2)) then
            integral(i)=(Ø.5)*(p1-pØ)*maxp
            integral(i)=integral(i)+(Ø.5)*(c(i)-p1)*(maxp+m2*(c(i)-p2))
           endif
           if (c(i) .gt. p2) then
            integral(i)=(Ø.5)*(p1-pØ)*maxp+(Ø.5)*(p2-p1)*maxp
           endif
   1Ø continue
      integralP=Integral(2)-integral(1)
      end

      subroutine getX(rvX)
c     Generates a random variable value for rvX depending on the
c     distribution choosen by the input number noX in the main program.

      real r1,r2,rvX,temp1,temp2,Xsigma
      integer noX
      common/Xparameters/noX,Xsigma

      if (noX .eq. 1) rvX=Ø.5
      if (noX .eq. 2) then
           call randomnormal(temp1,temp2)
           rvX=Xsigma*temp1+Ø.5
      endif
      if (noX .eq. 3) then
   1Ø      call randomnormal(temp1,temp2)
           if (abs(Xsigma*temp1) .le. Ø.5 ) then
                rvX=Xsigma*temp1+Ø.5
                goto 2Ø
           endif
```

```
      real h,L,mid(2),range(2),stat,temp
      integer count(2,400),itemp,j

      common/histograms/mid,range,count

      data h/.01/,L/2.0/

      temp=(stat-mid(j))/(range(j)/4)
      temp=(temp+L)/h

      if((temp.le.399.0) .and. (temp.ge.0.0)) then
           itemp=aint(temp)
           count(j,itemp+1)=count(j,itemp+1)+1
      else
           if (temp.gt.399.0) then
               count(j,400)=count(j,400)+1
           endif
           if (temp.lt.0.0) then

               count(j,1)=count(j,1)+1
           endif
      endif
      return
      end

      subroutine printhistograms(NoBlocks,Block)
c     prints the histogram values of count(j,k) together with the
c     scale factors necessary to convert them to percentiles

      real mid(2),range(2)
      integer Block,count(2,400),i,j,NoBlocks,sum(2,400)

      common/histograms/mid,range,count

      do 5 j=1,2
           sum(j,1)=count(j,1)
 5    continue
      do 20 j=1,2
           do 10 i=2,400
               sum(j,i)=sum(j,i-1)+count(j,i)
10         continue
20    continue
      open(unit=2,file='prn')
      write(2,30) NoBlocks,Block
30    format(1x,'Simulation histograms from ',I2,' blocks of ',
     &          'size ',I4)
      write(2,40)
40    format()
      write(2,'(1x,"Two statistics are tabulated below.")')
      write(2,'(1x,"The first is maximal discharge" )')
      write(2,'(1x,"The second is time to maximum discharge")')
      write(2,50)
50    format()
      write(2,55)
55    format(1x,'For the j-th statistic s(j) j=1 and 2 '/
     &  1x 'the scale factors listed below are used to produce'/
     &  1x,'the scaled statistic ss(j)=(s(j)-mid(j))/(range(j)',
     &  '/4). The histogram given'/1x,'below lists ',
     &  'in the k-th row k=1,...,400, and j-th column the number ',
     &  'of times'/1x,'ss(j) < -2+k*.01. Linear interpolation and ',
     &  'rescaling of these counts gives'/1x,'approximate ',
     &  'percentiles for the distributions of these statistics.')
```

```
      write(2,60)
      format()
      do 90 j=1,2
           write(2,80) j,mid(j),range(j)
           format(1x,'j= ',I1,'    mid= ',F8.4,'    range= ',F8.4)
      continue
      write(2,100)
      format()
      write(2,110)
      format(1x,'       k     j=1     j=2')
      do 130 j=1,400
           write(2,120) j,sum(1,j),sum(2,j)
           format(1x,3I7)
      continue
      write(2,140)
      format('1')

      form feed
      close(unit=2,file='prn')
      return
      end

      subroutine calcpercent(NoBlocks,Block)
c     does linear interpolation in the count histogram to find
c     percentiles for 5%(5%)95% and unscales to get percentiles
c     for the statistics of interest

      real h,interp,L,mid(2),percent(2,19),range(2),x1,x2,y,y1,y2
      integer Block,count(2,400),i,j,k,NoBlocks,Pm,s

      common/histograms/mid,range,count,percent

      L=2.0
      h=.01
      do 40 j=1,2
           do 30 i=1,19
                pM=i*5*NoBlocks*10
c                     1000/100=10; Integer Block =1000
                k=1
                s=count(j,1)
10              if (s.lt.pM) then
                     k=k+1
                     s=s+count(j,k)
                     goto 10
                endif
c     k is the first index with s>pM, s=the sum of count(j,m),m=1,...,k
c     (therefore count(j,k)>0)
                y=real(pM)
                y2=real(s)
                y1=y2-real(count(j,k))
                x2=k*h-L-h/2
                x1=x2-h
                interp=((y-y1)/(y2-y1))*(x2-x1)+x1
                percent(j,i)=interp*(range(j)/4)+mid(j)
30           continue
40      continue
      return
      end

      subroutine printpercent(NoBlocks,Block)
c     prints the percentiles percent(j,k) from "calcpercent"
```

```
      real Lsigma,mid(2),p(2),percent(2,19),range(2),Xsigma
      integer Block,count(2,400),i,j,NoBlocks,noL,noX

      common/histograms/mid,range,count,percent
      common/Xparameters/noX,Xsigma
      common/Lparameters/noL,Lsigma
      open(unit=2,file='prn')
      write(2,'(1x,"Percentiles.")')
      write(2,'(1x,"The percentiles in column one are for ",
     &          "maximal discharge")')
      write(2,'(1x,"The percentiles in column two are for ",
     &          "time to maximal discharge")')
      write(2,'(1x,"The distribution number for X is ",I3)') noX
      write(2,'(1x,"The distribution number for L is ",I3)') noL
      write(2,40) NoBlocks,Block
40    format(5x,'The simulation was ',I2,' blocks of size ',I5)
      write(2,50)
50    format()
      write(2,60)
60    format(1x,'percentile    column1    column2')
      write(2,70)
70    format(1x,'-----------------------------------')
      do 100 j=1,19
          do 80 i=1,2
              p(i)=percent(i,j)
80        continue
          write(2,90) 5*j,p(1),p(2)
90        format(1xI6,4x,2(3x,F7.2))
100   continue
      write(2,110)
110   format('1')
      close(unit=2,file='prn')
      end
```