**Algorithm K,Z**

Algorithm Z: a-0, b=d=-1

Algorithm K: a=b=n/2, d=0

(G.1) [Algorithm X]

(G.1.1 initialize the reservoir X1…Xn to be the leading n records  
t 🡨 n

(G.1.2) u 🡨 random()

(G.1.3) if (t ≥ A) goto (G.2)

(G.1.4) search for next record Y. if file exhausted before record found, return X1…Xn

(G.1.5) t 🡨 t + 1  
u 🡨 u \* t / (t – n)  
if (1 > u) goto (G.1.3)

(G.1.6) X[1 + floor(n \* random())] 🡨 Y

goto (G.1.2)

(G.2) *[initialize the acceptance-rejection method]*

m 🡨 t - n + 1  
W 🡨 exp(–log(random()) / n)

(G.3) *[generate* s *and test for acceptance using squeeze function]*

(G.3.1) u 🡨 random()  
x 🡨 (t – a) \* (W – 1)  
s 🡨 floor(x)

(G.3.2) rhs 🡨 (t + x – a) / (m + s) \* m / (t – a)

lhs 🡨 exp(log( u \* (t + 1) / rhs / (t – a) \* (t – b) / (m – d – 1) ) / n)

if (lhs > rhs) goto (G.4)

(G.3.3) W 🡨 rhs/lhs

goto (G.5)

(G.4) *[test for acceptance using f]*

(G.4.1) y 🡨 u \* (t – b) / (m – d – l) \* (t + s + 1) / (t – a + x)

if (s == 0) goto (G.4.6)

(G.4.2) denom 🡨 m

if (n < s) numer 🡨 m + s else numer 🡨 t + 1

(G.4.3) i 🡨 numer

(G.4.4) y 🡨 y \* i / denom  
denom 🡨 denom + 1

(G.4.5) i 🡨 i + 1

if (i ≤ t + s) goto (G.4.4)

(G.4.6) W 🡨 exp(–log(random()) / n)

(G.4.7 if ( exp( log(y)/ n ) > (t - a + x) / (t – a) ) goto (G.3)

(G.5) [Search for the next potential record]

Search for the next (s+1)th record, say Y.  
If the file is exhausted before the record is found, deliver X1…Xn and terminate

(G.6) [Update t and the reservoir, and repeat]

t 🡨 t + s + 1  
m 🡨 m + s + 1  
X[1 + floor(n \* random())] 🡨 Y  
goto (G.3)

**Algorithm Z**

(G.1) [Algorithm X]

(G.1.1 initialize the reservoir X1…Xn to be the leading n records

(G.1.2) u 🡨 random()

(G.1.3) if (t ≥ A) goto (G.2)

(G.1.4) search for next record Y. if file exhausted before record found, return X1…Xn

(G.1.5) t 🡨 t + 1  
u 🡨 u \* t / (t – n)  
if (1 > u) goto (G.1.3)

(G.1.6) X[1 + floor(n \* random())] 🡨 Y

goto (G.1.2)

(G.2) *[initialize the acceptance-rejection method]*

m 🡨 t - n + 1  
W 🡨 exp(–log(random()) / n)

(G.3) *[generate* s *and test for acceptance using squeeze function]*

(G.3.1) u 🡨 random()  
x 🡨 t \* (W – 1)  
s 🡨 floor(x)

(G.3.2) rhs 🡨 (t + x) / (m + s) \* m / t

lhs 🡨 exp(log( u \* (t + 1) / rhs / t \* (t + 1) / m ) / n)

if (lhs > rhs) goto (G.4)

(G.3.3) W 🡨 rhs/lhs

goto (G.5)

(G.4) *[test for acceptance using f]*

(G.4.1) y 🡨 u \* (t + 1) / m \* (t + s + 1) / (t + x)

if (s == 0) goto (G.4.6)

(G.4.2) denom 🡨 m

if (n < s) numer 🡨 m + s else numer 🡨 t + 1

(G.4.3) i 🡨 numer

(G.4.4) y 🡨 y \* i / denom  
denom 🡨 denom + 1

(G.4.5) i 🡨 i + 1

if (i ≤ t + s) goto (G.4.4)

(G.4.6) W 🡨 exp(–log(random()) / n)

(G.4.7 if ( exp( log(y)/ n ) > (t + x) / t ) goto (G.3)

(G.5) [Search for the next potential record]

Search for the next (s+1)th record, say Y.  
If the file is exhausted before the record is found, deliver X1…Xn and terminate

(G.6) [Update t and the reservoir, and repeat]

t 🡨 t + s + 1  
m 🡨 m + s + 1  
X[1 + floor(n \* random())] 🡨 Y  
goto (G.3)

**Algorithm M**

Integer : n, S, c, r

Real : W, q, t, u, z

(M.1) [Initialize]

Initialize the reservoir X1,…,Xn to be the first n records

Set t 🡨 0 and u 🡨 random()

(M.2) [Algorithm X]

Repeat the following operations c times:

search for next record, say Y

if file exhausted before record is found, deliver X1,…,Xn and terminate

t 🡨 t+1

u 🡨 u\*(1 + n/t)

if (1 ≤ u) X[floor(n\*random())] 🡨 Y, u 🡨 random()

(M.3) [Generate W]

Generate W from the Beta(n, c+1) distribution

(M.4) [Compute q and generate S]

q 🡨 log(l – W)

S 🡨 floor(log(u)/q)

(M.5) [Search for next potential record]

search for next (S+l)th record, say Y

if file exhausted before record is found, deliver Xl,…,Xn and terminate

(M.6) [Update X]

X[l+floor(n\*random())] 🡨 Y

t 🡨 0

u 🡨 random()

(M.7) [Algorithm X]

Repeat the following operations r times:

s 🡨 floor(log(random())/q)

search for next (s+l)th record, say Y

if file exhausted before record is found, deliver Xl,…,Xn and terminate

t 🡨 t+1

u 🡨 u\*(1 + n/t)

if (1<u) X[1+floor(n\*random()] 🡨 Y, u 🡨 random()

(M.8) [Update W]

generate Z from the Beta(n, r+1) distribution

W 🡨 W\*Z

goto (M.4)

H[0] 🡨 0

H[u>0] 🡨 sum(i🡨1…u, 1/i)

r 🡨 floor(2.07\*sqrt(n))

c 🡨 floor(10.5\*(3.14245+r)/(H[n+r]-H[n-1])-n)

large n:

c 🡨 floor(10.5\*(3.14245+r)/log(n+r)/(n-1))-n)