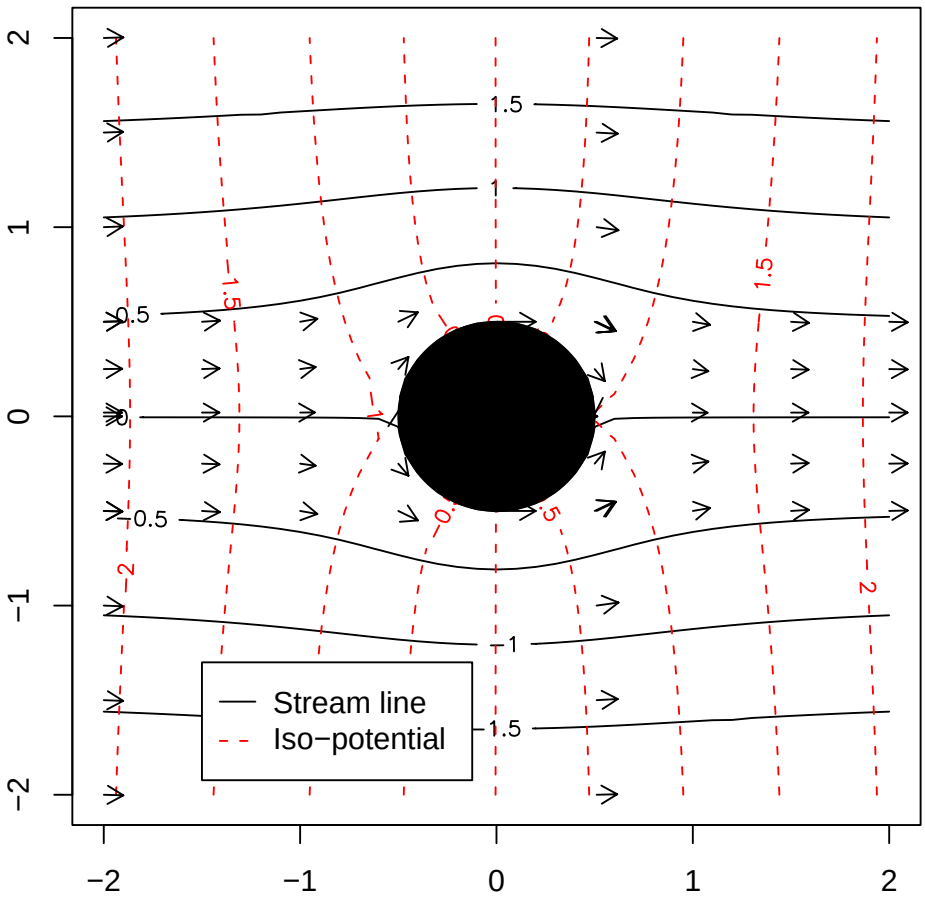
（環境水理学演習／追加ノート： Short album of flows and flowers ）

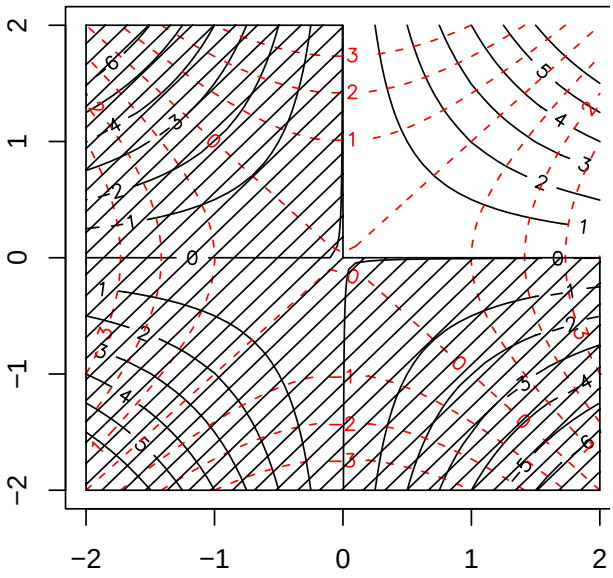
担当：北野 利一（kitano@nitech.ac.jp） ２４号館３１９号室

```
     computing software: R [available at http://www.r-project.org/ ]

     R : Copyright 2003, The R Development Core Team
     Version 1.7.1  (2003-06-16)

     R is free software and comes with ABSOLUTELY NO WARRANTY.
     You are welcome to redistribute it under certain conditions.
     Type `license()' or `licence()' for distribution details.

     R is a collaborative project with many contributors.
     Type `contributors()' for more information.

     Type `demo()' for some demos, `help()' for on-line help, or
     `help.start()' for a HTML browser interface to help.
     Type `q()' to quit R.

1:   > # 1 # flow around cylinder
2:   > # rectangular coordinates on a complex plane
3:   > x <- seq(-2,2,by=0.1)
4:   > y <- seq(-2,2,by=0.1)
5:   > z <- outer(x,1i*y, "+")
6:   > dim(z)
     [1] 41 41

7:   > ## complex velocity potential ##
8:   > W <- z + 0.25/z

9:   > contour(x,y, Im(W))                                  # stream lines
10:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)     # iso-potential

11:  > theta <- 2*pi*(1:100)/100
12:  > polygon(.5*cos(theta), .5*sin(theta), col="black")  # cylinder
```

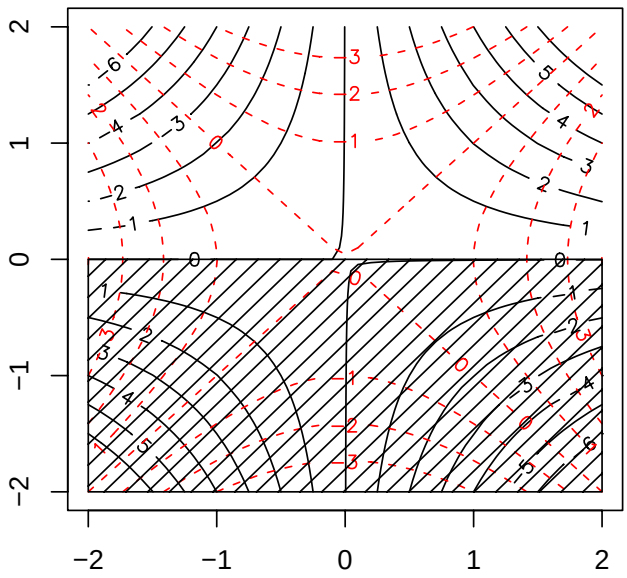


```
13:  > # flow velocity
14:  > Ux <- function(x,y, a=.5) 1 - a^2*(x^2 -y^2)/(x^2 + y^2)^2
15:  > Uy <- function(x,y, a=.5)   - a^2*2*     x*y/(x^2 + y^2)^2

16:  > # rectangular coordinates for velocity vectors
17:  > xx <- seq(-2, 2, by= 0.5)
18:  > yy <- seq(-2, 2, by= 0.5)
19:  > dt <- 0.1

20:  > arrows(xx, .5, xx  + dt*Ux(xx,.5),    .5+ dt*Uy(xx,.5),  length=.1)
21:  > arrows(.51, yy, .51+ dt*Ux(.51,yy),   yy+ dt*Uy(.51,yy), length=.1)

22:  > xx0 <- xx[-5]
23:  > arrows(xx0,  .25, xx0+dt*Ux(xx0,.25),   .25+dt*Uy(xx0, .25), length=.1)
24:  > arrows(xx0, -.25, xx0+dt*Ux(xx0,-.25), -.25+dt*Uy(xx0,-.25), length=.1)
25:  > arrows(xx,  -.50, xx +dt*Ux(xx, -.50), -.50+dt*Uy(xx, -.50), length=.1)
26:  > arrows(xx0,  .02, xx0+dt*Ux(xx0, .02),  .02+dt*Uy(xx0, .02), length=.1)

27:  > arrows(-2, yy, -2+ dt*Ux(-2,yy), yy+ dt*Uy(-2,yy), length=.1)

     > # series of simple flows:
     > # using again the cordinates defined at lines 3-5.
28:  > x <- seq(-2,2,by=0.1)
29:  > y <- seq(-2,2,by=0.1)
30:  > z <- outer(x,1i*y, "+")

31:  > # 2 #
32:  > W <- z
33:  > contour(x,y, Im(W), main="Uniform flow")        # stream lines
34:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE) # iso-potential

35:  > # 3 #
36:  > W <- z^2
37:  > contour(x,y, Im(W), main="1) Flow at corner: pi/2", levels=-6:6)
38:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
39:  > polygon(c(-2,0,0,2, 2,-2),
     +         c( 2,2,0,0,-2,-2), dens=10)

40:  > # same potential for another interpretation
41:  > contour(x,y, Im(W), main="2) Flow at corner: pi", levels=-6:6)
42:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
43:  > polygon(c(-2,2, 2,-2),
     +         c( 0,0,-2,-2), dens=10)
```

1) Flow at corner: pi/2

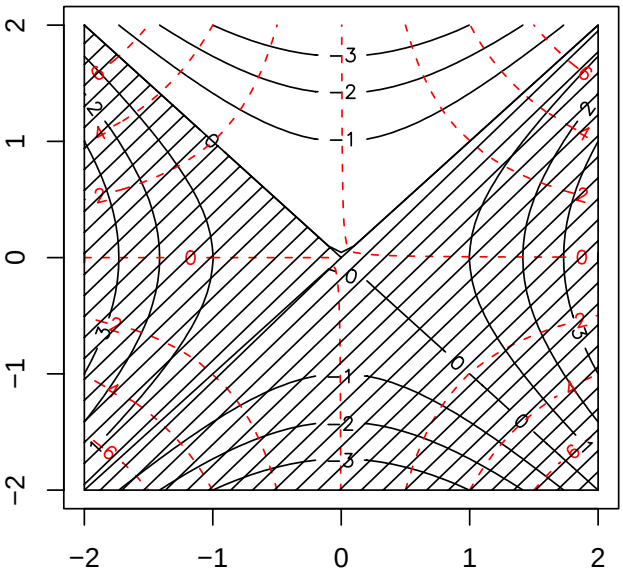


2) Flow at corner: pi

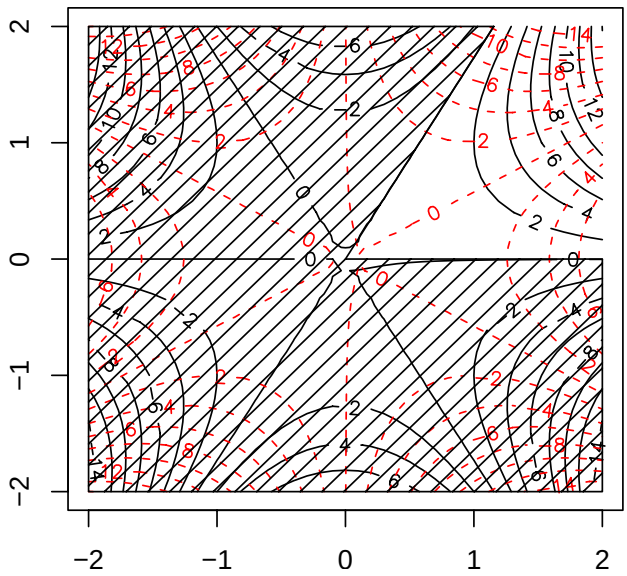


```
44:  > # 4 #
45:  > W <- 1i*z^2
46:  > contour(x,y, Im(W), main="3) Flow at corner: pi/2 (rotate)")
47:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
48:  > polygon(c(-2,0,2, 2,-2), c(2, 0,2,-2,-2), dens=10)
49:  > # 5 #
50:  > W <- z^3
51:  > contour(x,y, Im(W), main="4) Flow at corner: pi/3", nlevels=20)
52:  > contour(x,y, Re(W), nlevels=20, lty=2, col="red", add=TRUE)
53:  > polygon(c(2/tan(pi/3), -2, -2,  2, 2, 0),
     +          c(2,  2, -2, -2, 0, 0), dens=10)
54:  > # 6 #
55:  > W <- z^(3/2)
56:  > contour(x,y, Im(W), main="5) flow at corner: 2*pi/3")
57:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
58:  > polygon(c(-2, -2/sqrt(3), 0, 2,  2, -2),
     +          c( 2,          2, 0, 0, -2, -2), dens=10)
59:  > # 7 #
60:  > contour(x,y, Im(W), main="6) flow at corner: 4*pi/3")
61:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
62:  > polygon(c(-2, -2/sqrt(3), 0, -2/sqrt(3), -2),
     +          c( 2,          2, 0,          -2, -2), dens=10)
```

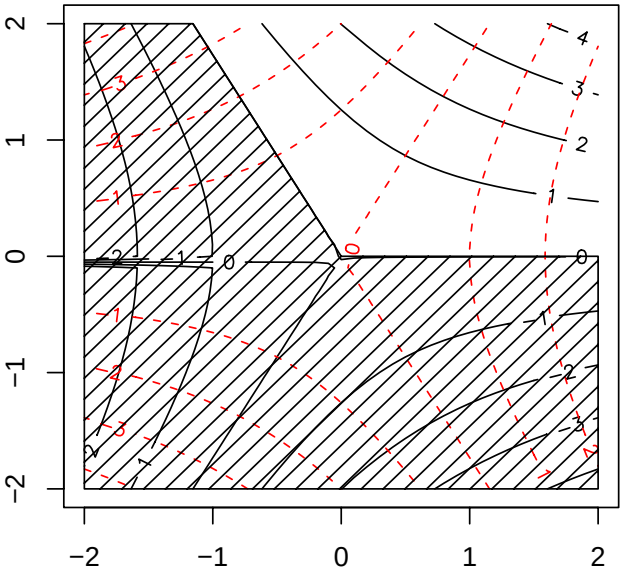
3) Flow at corner: pi/2 (rotate)



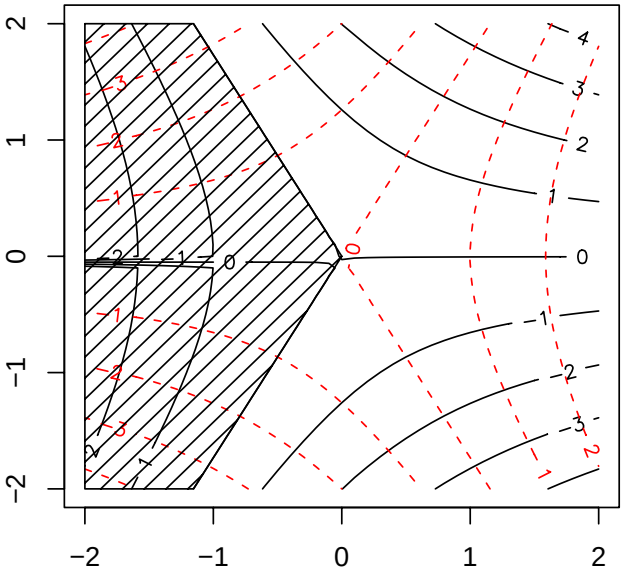
4) Flow at corner: pi/3



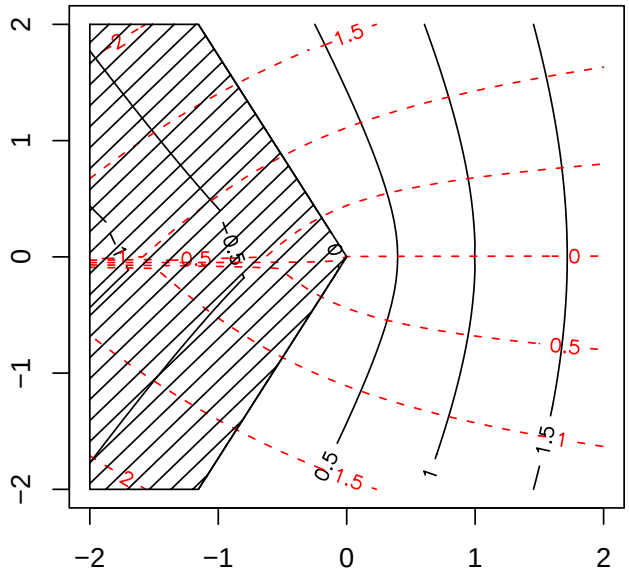
5) flow at corner: 2*pi/3



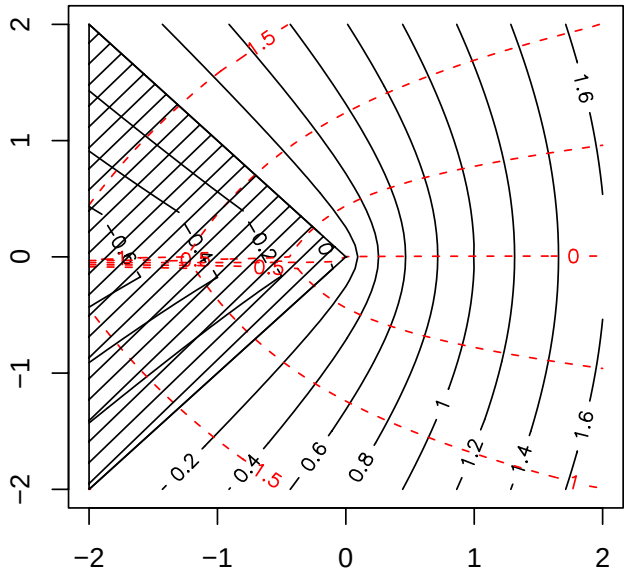
6) flow at corner: 4*pi/3



7) another flow at corner: 4*pi/3

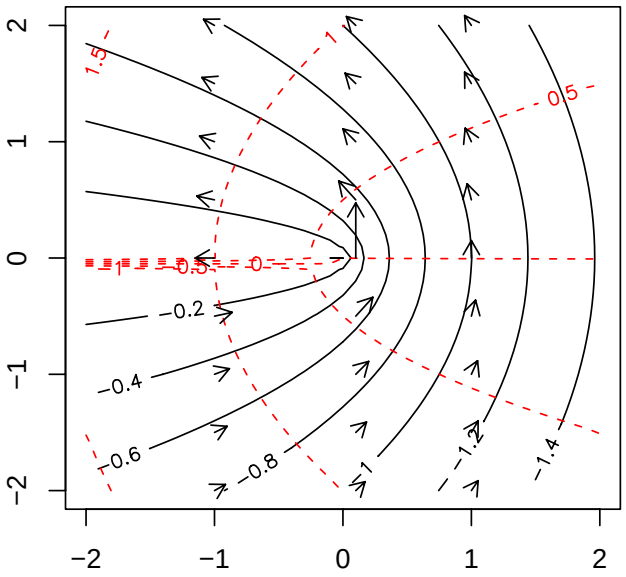


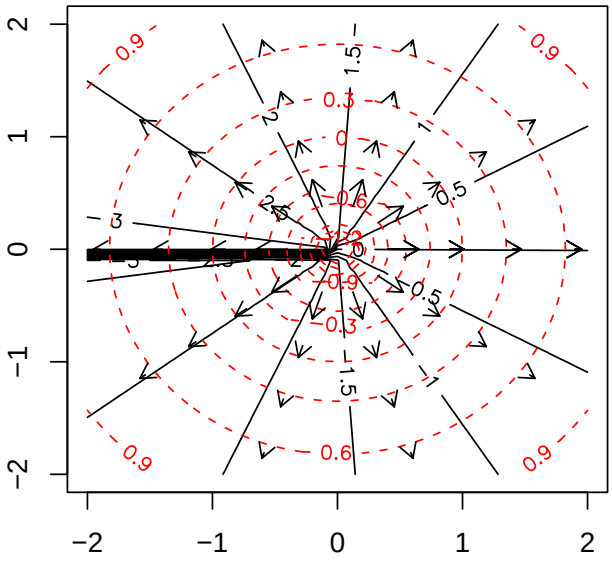
8) flow at corner: 3*pi/2

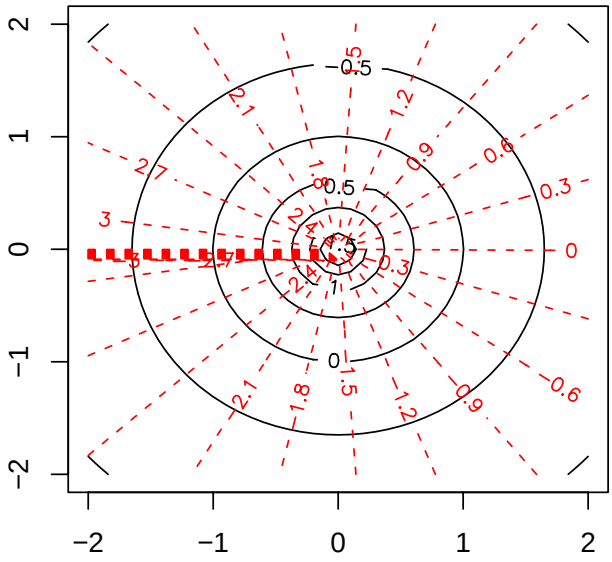



```
63:  > # 8 #
64:  > W <- 1i*z^(3/4)
65:  > contour(x,y, Im(W), main="7) another flow at corner: 4*pi/3")
66:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
67:  > polygon(c(-2, -2/sqrt(3), 0, -2/sqrt(3), -2),
     +         c( 2,          2, 0,         -2, -2), dens=10)

68:  > # 9 #
69:  > W <- 1i*z^(2/3)
70:  > contour(x,y, Im(W), main="8) flow at corner: 3*pi/2")
71:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
72:  > polygon(c(-2, 0, -2),
     +         c( 2, 0, -2), dens=10)

73:  > # 10 #
74:  > W <- -1i*z^(1/2)
75:  > contour(x,y, Im(W), main="9) flow around half plane")
76:  > contour(x,y, Re(W), lty=2, col="red", add=TRUE)

77:  > xc <- c(rep(-1,9), rep(.1,9), rep(1,9))
78:  > yc <- c(yy, yy, yy) # yy defined at line 17
79:  > ux <- numeric(27)
80:  > uy <- numeric(27)
81:  > for (i in 1:27) {
     +     xi <- xc[i]
     +     yi <- yc[i]
     +  angle <- atan2(yi, xi)
     + radius <- sqrt(xi^2+yi^2)
     +  denom <- 2*sqrt(radius)
     +  ur <- sin(angle/2)/denom
     +  ut <- cos(angle/2)/denom
     +  ux[i] <- ur*cos(angle) -
     +            ut*sin(angle)
     +  uy[i] <- ur*sin(angle) +
     +            ut*cos(angle)}
82:  > arrows(xc,
     +         yc,
     +         xc+dt*ux,
     +         yc+dt*uy, length=.1)
83:  > # try to draw the velocity
84:  > # field also in figs. 1)-8).
```

9) flow around half plane

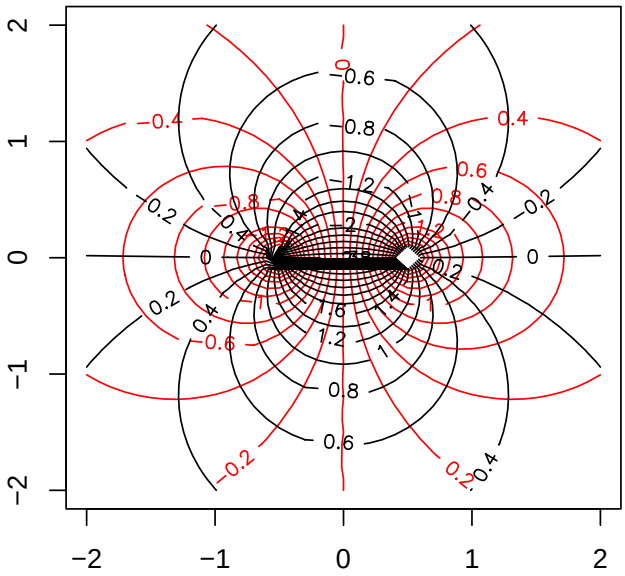


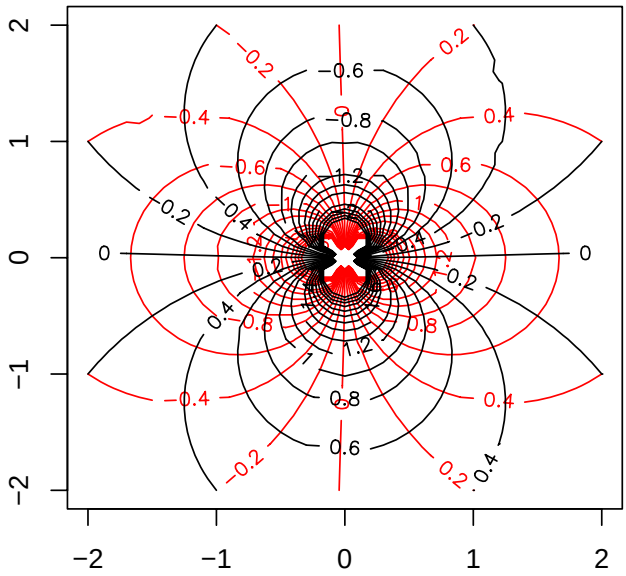


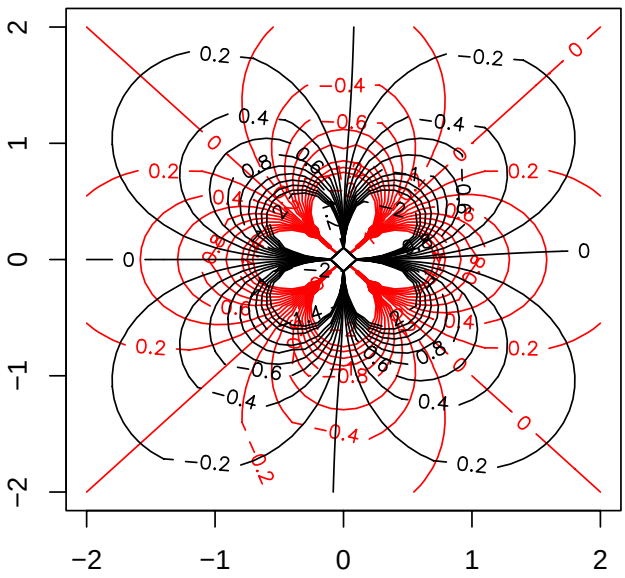


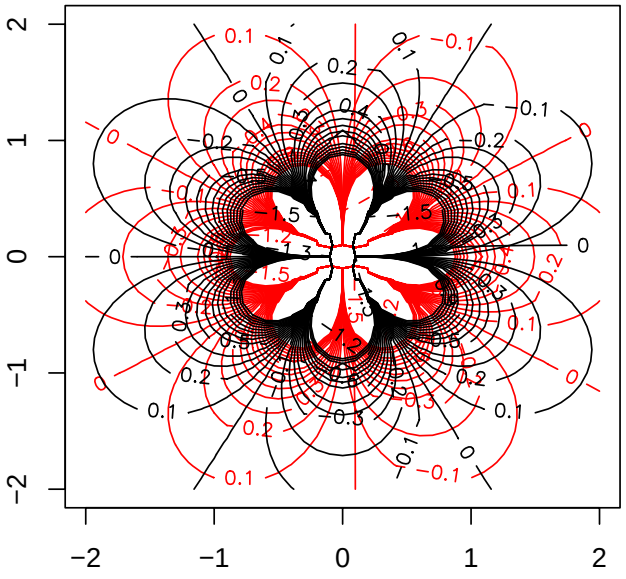
```
> # 11 #
> W <- log(z)
> contour(x,y, Im(W))
> contour(x,y, Re(W), lty=2, col="red", add=TRUE, levels=seq(-3,3,by=.3))
> xf <- cos(2*pi*seq(0,1,by=1/10)); yf <- sin(2*pi*seq(0,1,by=1/10))
> veloc <- function(r) arrows(r*xf, r*yf, (r+dt/r)*xf, (r+dt/r)*yf, len=.1)
> veloc(0.4); veloc(0.9); veloc(1.4); veloc(1.9)

> # 12 #
> W <- - 1i * log(z)
> contour(x,y, Im(W))
> contour(x,y, Re(W), lty=2, col="red", add=TRUE, levels=seq(-3,3,by=.3))

> # 13 #
> plot(c(-7,7),c(-7,7), type="n", main="Rankin Vortex", xlab="X", ylab="Y")
> for (j in c(1:3,5:7))
+ lines(  j*cos(2*pi*(1:101)/100),   j*sin(2*pi*(1:101)/100))
> lines(3.9*cos(2*pi*(1:101)/100), 3.9*sin(2*pi*(1:101)/100))
> lines(4.1*cos(2*pi*(1:101)/100), 4.1*sin(2*pi*(1:101)/100))
> lines(  4*cos(2*pi*(1:101)/100),   4*sin(2*pi*(1:101)/100),
+  col="blue", lty=3)
> abline(h=0, lty=2)

> for (j in 1:4) arrows( j,0, j,     j/3)
> for (j in 4:7) arrows(-j,0,-j,-4/3*4/j)
> lines(seq(0,4, by=.1), seq(0,4, by=.1)/3, lty=3)
> lines(seq(-7,-4, by=.1), 4/3*4/seq(-7,-4, by=.1), lty=3)

> xf <- cos(2*pi*seq(1/8,1,by=1/8)); a <- sqrt(2.5^2 + 0.5^2)
> yf <- sin(2*pi*seq(1/8,1,by=1/8)); b <- atan(-5/25)
> arrows(6.5*xf[-4], 6.5*yf[-4]-.5,
+        6.5*xf[-4], 6.5*yf[-4], angle=150,lwd=2)
> arrows(a*cos(b+2*pi*seq(1/8,7/8,by=1/8)),
+        a*sin(b+2*pi*seq(1/8,7/8,by=1/8)),
+        2.5*xf[-8], 2.5*yf[-8], angle=150,lwd=2)

> x <- seq(-7, 7, length=100); a=4; wg=1/5; h0=1
> wlev <- numeric(100)
> wlev[abs(x) <  a] <- h0 +     wg*         x[abs(x) <  a]^2/2
> wlev[abs(x) >= a] <- h0 + a^2*wg*(1 - a^2/x[abs(x) >= a]^2/2)
> plot(x, wlev, type="l", ylim=c(0,4), ylab="Depth"); abline(h=0)
> abline(v= 4, col="blue", lty=2)
> abline(v=-4, col="blue", lty=2)
```

```
119: > # 12 #
120: > W <- log((z+.5)/(z-.5))

121: > contour(x,y, Re(W), levels=seq(-3,3,by=.2), col="red")
122: > contour(x,y, Im(W), levels=seq(-3,3,by=.2), add=TRUE)

123: > # 13 #
124: > W <- 1/z

125: > contour(x,y, Re(W), levels=seq(-3,3,by=.2), col="red")
126: > contour(x,y, Im(W), levels=seq(-3,3,by=.2), add=TRUE)

127: > # 14 #
128: > W <- 1/z^2

129: > contour(x,y, Re(W), levels=seq(-2,2,by=.2), col="red")
130: > contour(x,y, Im(W), levels=seq(-2,2,by=.2), add=TRUE)

131: > # 15 #
132: > W <- 1/z^3

133: > contour(x,y, Re(W), levels=seq(-1.5,1.5,by=.1), col="red")
134: > contour(x,y, Im(W), levels=seq(-1.5,1.5,by=.1), add=TRUE)
```

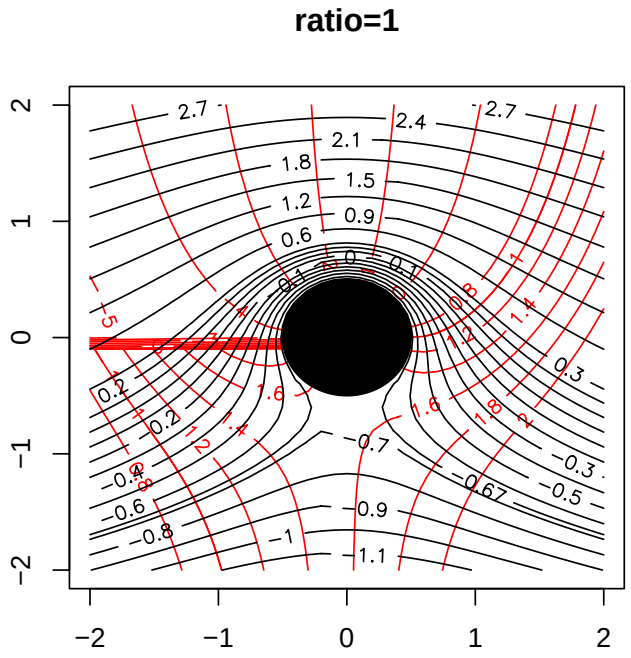


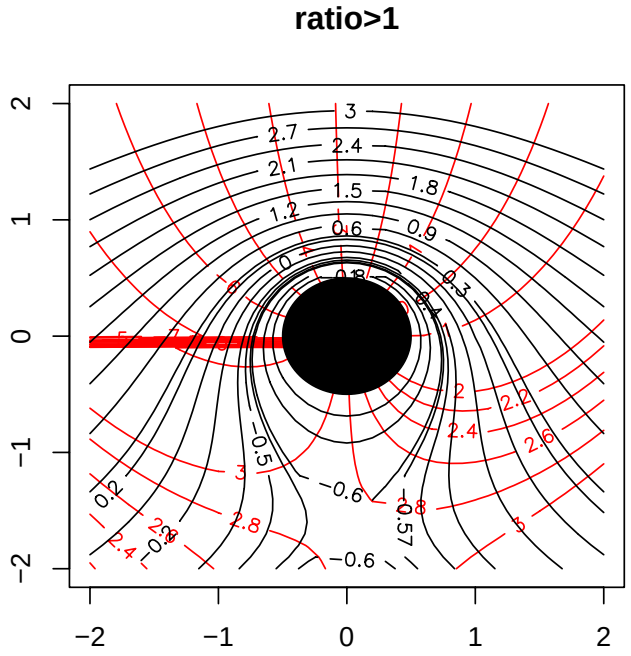


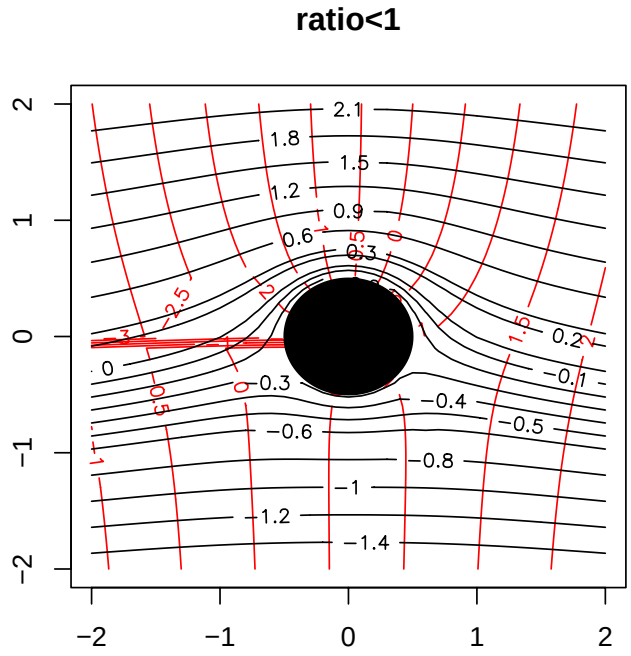




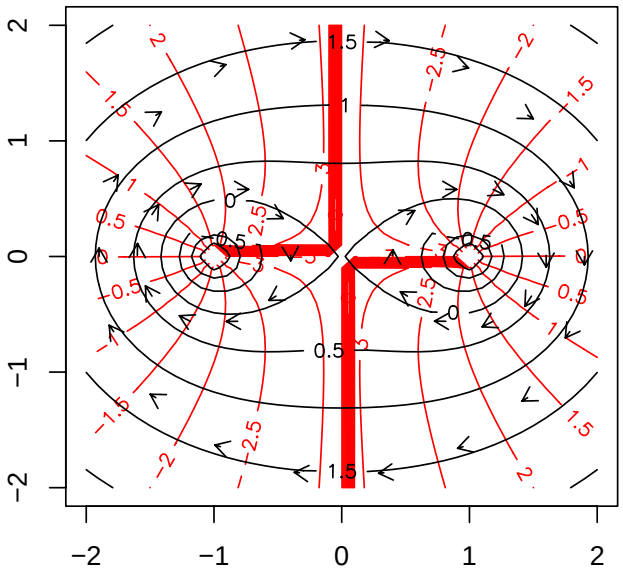
```
135: > # using again the shadow disk
136: > theta <- 2*pi*(1:100)/100

137: > # 16 #
138: > W <- z + .25/z + 1i * log(z)

139: > contour(x,y, Re(W),
     +  levels=c(seq(-5,1),
     +  seq(.8,2,by=.2)),
     +  main="ratio=1", col="red")
140: > contour(x,y, Im(W),
     +  levels=c(seq(-1.1,.2, by=.1),
     +  seq(.3,3,by=.3), -.67),
     +  add=TRUE)

141: > polygon(.5*cos(theta),
     +         .5*sin(theta),
     +  col="black")

142: > # 17 #
143: > W <- z + .25/z + 1.8i * log(z)

144: > contour(x,y, Re(W),
     +  levels=c(seq(-8,4),
     +  seq(2.2, 2.8, by=.2)),
     +  main="ratio>1", col="red")
145: > contour(x,y, Im(W),
     +  levels=c(seq(-1.0,.2, by=.2),
     +  seq(.3,3,by=.3), -.5, -.57),
     +  add=TRUE)

146: > polygon(.5*cos(theta),
     +         .5*sin(theta),
     +  col="black")

147: > # 18 #
148: > W <- z + .25/z + 0.4i * log(z)
149: >
150: > contour(x,y, Re(W),
     +  levels=seq(-3,3, by=.5),
     +  main="ratio<1", col="red")
151: > contour(x,y, Im(W),
     +  levels=c(seq(-2,.2, by=.2),
     +  seq(.3,3,by=.3), -.5, -.3, -.1),
     +  add=TRUE)

152: > polygon(.5*cos(theta),
     +         .5*sin(theta),
     +  col="black")
```

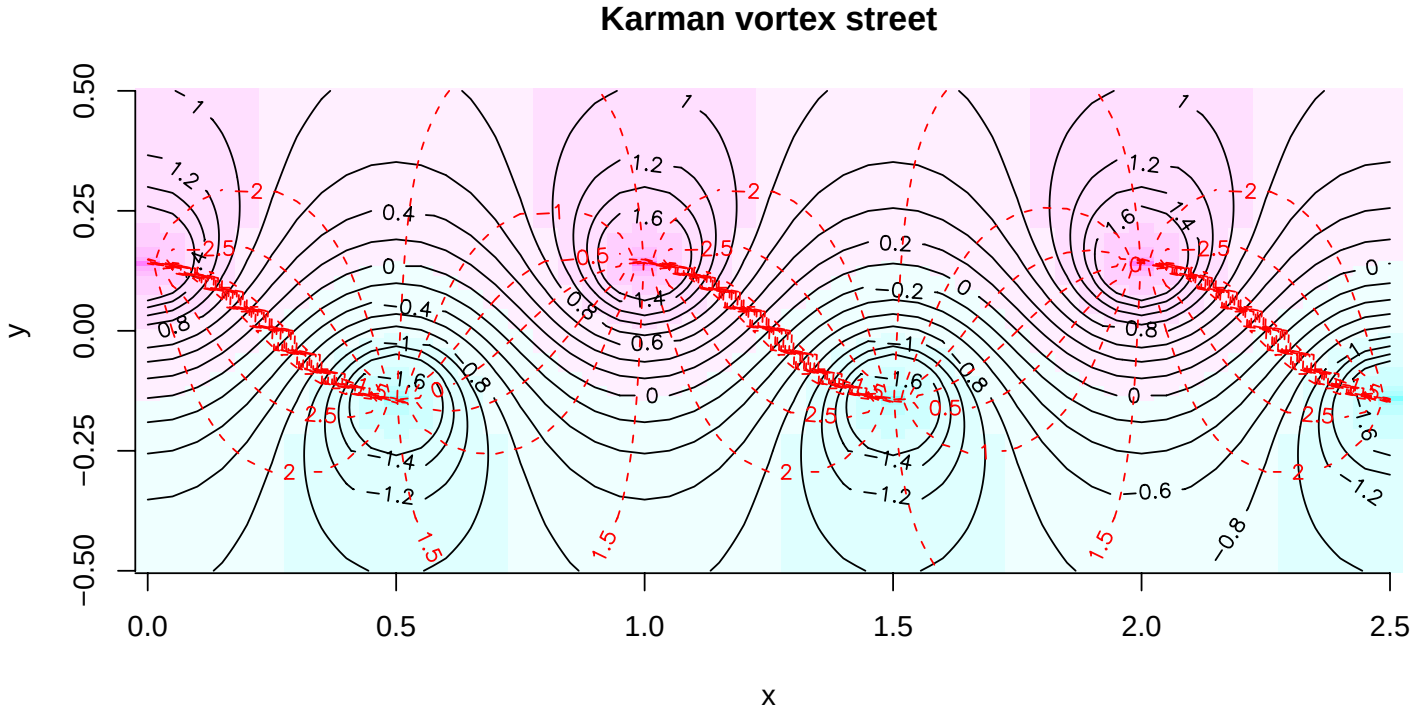


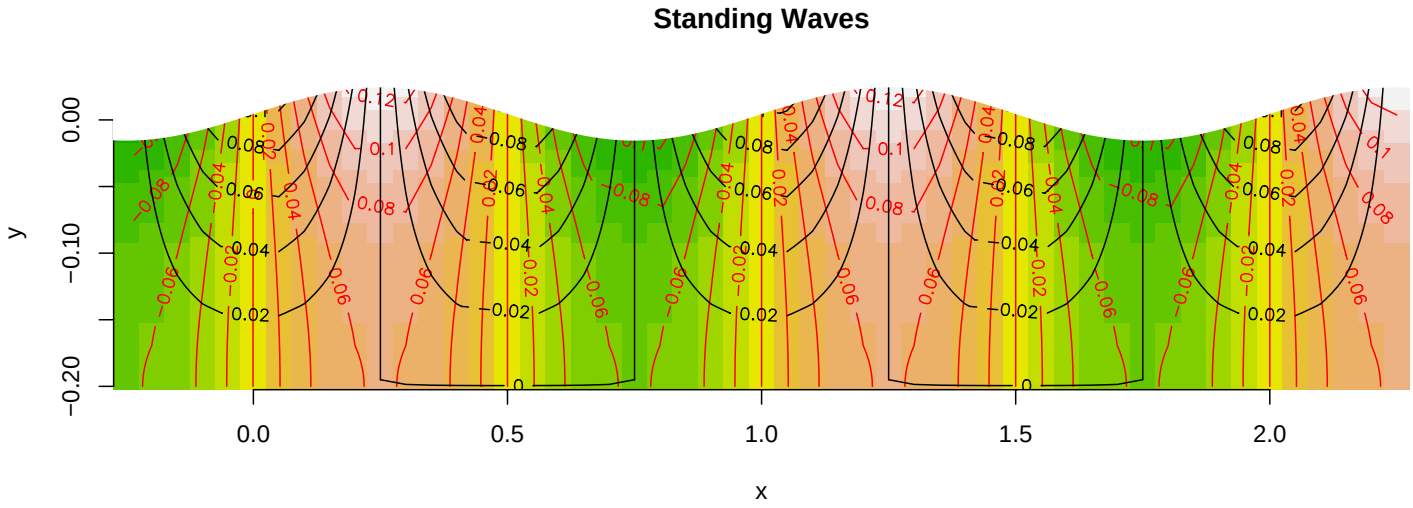


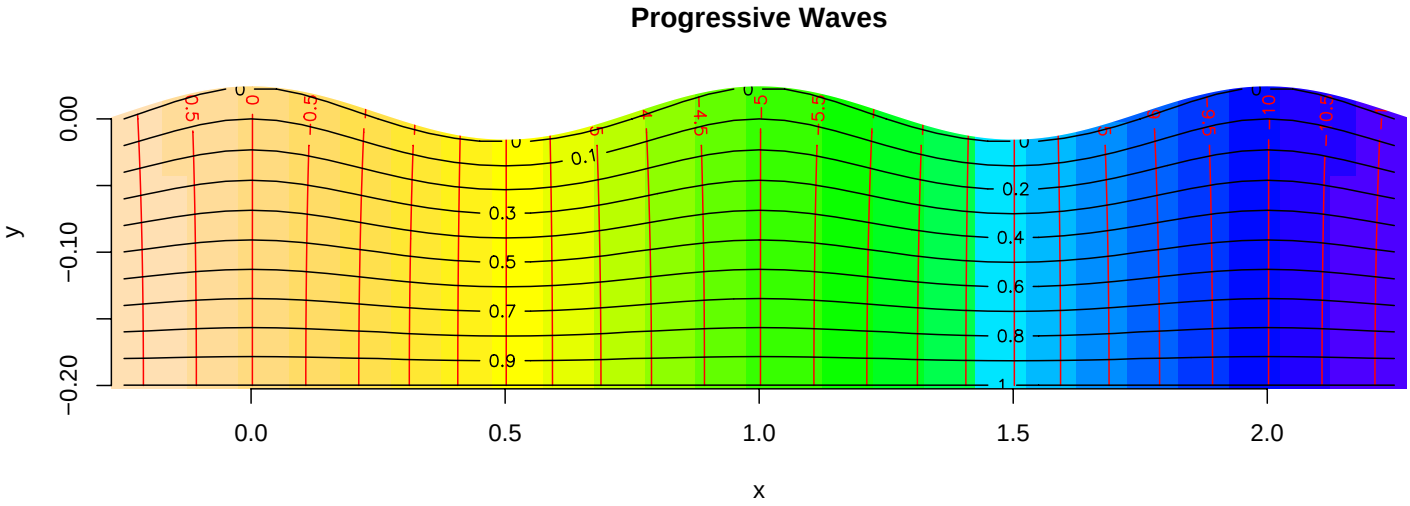


```
153: > # 19 #
154: > W <- 1i*log((z+1)*(z-1))

155: > contour(x,y, Re(W), col="red")
156: > contour(x,y, Im(W), add=TRUE)

157: > dx <- .0001; dy <- .0001
158: > dt <- .05
159: > x1 <- c(
     +  .6*cos(2*pi*1:10/10) + 1,
     +  .6*cos(2*pi*1:10/10) - 1,
     + 1.9*cos(2*pi*1:18/18))
160: > y1 <- c(
     +  .6*sin(2*pi*1:10/10) +.01,
     +  .6*sin(2*pi*1:10/10) +.01,
     + 1.9*sin(2*pi*1:18/18))

161: > (Re(1i*log((x1+dx+1i*y1+1
     +          )*(x1+dx+1i*y1-1))) -
     +  Re(1i*log((x1   +1i*y1+1
     +          )*(x1   +1i*y1-1))))/dx -> ux
162: > (Re(1i*log((x1+1i*(y1+dy)+1)*(x1+1i*(y1+dy)-1))) -
     +  Re(1i*log((x1+1i*y1     +1)*(x1+1i*y1     -1))))/dy -> uy
163: > arrows(x1, y1, x1+ux*dt, y1+uy*dt, length=.1)

164: > # 20 #
165: > # Karman vortex street (stable condition: a/b = 0.2805)

166: > x <- seq(-2,2,by=0.05)
167: > y <- seq(-.5,.5,by=0.01)
168: > z <- outer(x,1i*y, "+")

169: > W <- 1/1i*log(sin(pi*(z    -1i*0.2805/2)
     +             )/sin(pi*(z-1/2+1i*0.2805/2)))

170: > image(x,y, Im(W), col=cm.colors(16), main="Karman vortex street", axes=F)
171: > contour(x,y, Im(W), levels=seq(-1.6,1.6, by=.2), add=T)
172: > contour(x,y, Re(W), lty=2, col="red", add=TRUE)
173: > axis(1, seq(-2,2.5,by=.5))
174: > axis(2, seq(-.5,.5,by=.25))

175: > acosh(sqrt(2))/pi
     [1] 0.2805499
176: > persp(x,y, Im(W), phi=30)
```










```
> # coordinates
> x <- seq(-.25,2.25,by=0.05)
> y <- seq(-.2,.03,by=0.005)
> z <- outer(x,1i*y, "+")

> # 21 # standing waves
> W <- .1i*sinh(2*pi*(.2-1i*z))/sinh(.4*pi)

> image(x,y, Re(W), col=terrain.colors(18),
+  main="Standing Waves", axes=F); axis(1); axis(2)
> contour(x,y, Re(W), col="red", add=T)
> contour(x,y, Im(W), add=T)
> xf <- seq(-.3,     2.3, len=200)
> polygon(c(-.3, xf, 2.3),
+  c(.035, .005+.02*sin(2*pi*xf), .035), col="white", border="white")

> # 22 # progressive waves
> W <- W - 5*z

> image(x,y, Re(W), col=topo.colors(24),
+  main="Progressive Waves", axes=F); axis(1); axis(2)
> contour(x,y, Re(W), levels=seq(-11,2.5,by=.5), col="red", add=T)
> contour(x,y, Im(W), add=T)
> polygon(c(-.3, xf, 2.3),
+  c(.035, .005+.02*cos(2*pi*xf), .035), col="white", border="white")
```

```
> sol <- a <- (2*pi/10)^2*20/9.8; x <- seq(0.6, 1.4, by=.01)
> for (j in 1:6) sol <- c(a/tanh(sol[1]), sol)
> plot(x, a/tanh(x), type="l", ylim=c(a,1.3))
> points(rev(sol[-1]), rev(sol)[-1]); abline(0,1); abline(h=a, lty=3)
> rep(sol, rep(2, length(sol)))[-1] -> p
> lines(rev(p[-1]), rev(p)[-1], col="red")

> plot(x, a/tanh(x), type="l", xlim=c(.55,1.45), ylim=c(.5,1.5))
> x0 <- seq(0.6, 1.4, by=.05); points(x0, rep(.5, 17))
> for (j in x0) lines(c(j,j,.55), c(.5, a/tanh(j), a/tanh(j)), lty=2)
> points(rep(.55, 17), a/tanh(x0), pch=18);abline(h=a, lty=3)

> # shoaling coefficient
> par(fig=c(.3,1,0,1))
> kh <- 10^seq(-.5,.7,length=100)
> Ks <- 1/sqrt(tanh(kh) + kh * (1 - tanh(kh)^2))
> plot(kh,Ks, type="l", log="x"); axis(1,c(.3,.5)); abline(h=1, lty=3)
> par(new=T, fig=c(0,.32,0,1))
> kh <- 10^seq(-2,-.3,length=100)
> Ks <- 1/sqrt(tanh(kh) + kh * (1 - tanh(kh)^2))
> plot(kh,Ks, type="l", log="x", axes=F)
> axis(1, c(.01,.05,.5)); axis(2); box()

> # advanced (tricky !)
> k0h <- 10^seq(-1,.7,length=100); kh <- numeric(100); tol <- .01
> f <- function(x, a) a - x*tanh(x)
> for (j in 1:100)
+  kh[j] <- uniroot(f, c(0-tol,log(2)*2/5+tol) + k0h[j], a=k0h[j])$root
> Ks <- 1/sqrt(tanh(kh) + kh * (1 - tanh(kh)^2))
> plot(k0h,Ks, type="l",log="x",xlab="x"); axis(1,.3); abline(h=1,lty=3)
> lines(kh, Ks, col="blue")
> legend(2,1.25, c("x = k0h", "x = kh"), lty=1, col=c("black", "blue"))
> # cf. fixed-point method described in NOTE
> kh.bis <- numeric(100); for (j in 1:100) {z<- z0<- k0h[j]; zz<- z0/tanh(z)
+           while (abs(1 - z/zz) > 10^-6) {z<- zz;          zz<- z0/tanh(z)}
+ kh.bis[j] <- z}; hist(kh.bis - kh)

> # Hargen-Poiseuille flow and the related ones
> a <- 1; x <- seq(-1, 1, by=.01); u <- a^2 - x^2
> uu <- function(r, b) (a^2*log(r/b) + b^2*log(a/r))/log(a/b) - r^2
> plot(u, x, type="l", xlab="U/Umax", ylab="x")
> x1<- seq(.01,1, by=.01); lines(uu(x1,.01), x1); lines(uu(x1,.01), -x1)
> x2<- seq(.1, 1, by=.01); lines(uu(x2, .1), x2); lines(uu(x2, .1), -x2)
> x3<- seq(.3, 1, by=.01); lines(uu(x3, .3), x3); lines(uu(x3, .3), -x3)
> x4<- seq(.5, 1, by=.01); lines(uu(x4, .5), x4); lines(uu(x4, .5), -x4)
> x5<- seq(.7, 1, by=.01); lines(uu(x5, .7), x5); lines(uu(x5, .7), -x5)
> x6<- seq(.8, 1, by=.01); lines(uu(x6, .8), x6); lines(uu(x6, .8), -x6)
> # END: continue to draw others by yourself
```