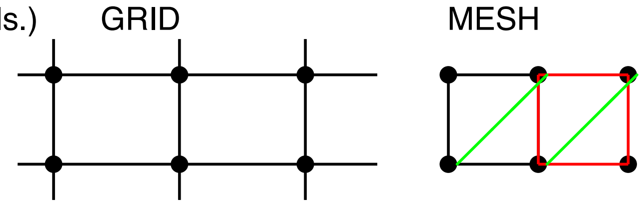
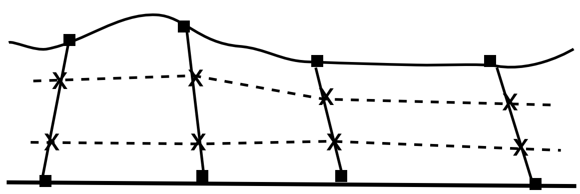
Algebraic grid generation

A. A simple, algebraic grid. Grid is a set of points connected by lines (Mesh is the set of points joined into cells.)



General idea: mesh surface, propagate into interior -- sometimes by solving elliptic boundary value problem for x(i,j).



Choose points on wall, connect them, place tic marks along connecting line = grid; i.e., define surfaces by $(x_{in}(i), y_{in}(i))$

Specify x,y inner and outer then: $x(s) = x_{in} + (x_{out} - x_{in})\, s \quad (0 \leq s \leq 1)$

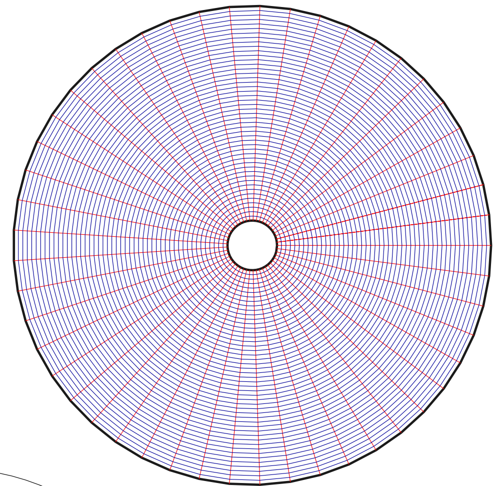
```
DO i=1,I
   DO j=1,J
     x(i,j) = xin(i) + (xout(i)-xin(i))*(j-1)/(J-1)
     y(i,j) = yin(i) + (yout(i)-yin(i))*(j-1)/(J-1)
   ENDDO
ENDDO
```

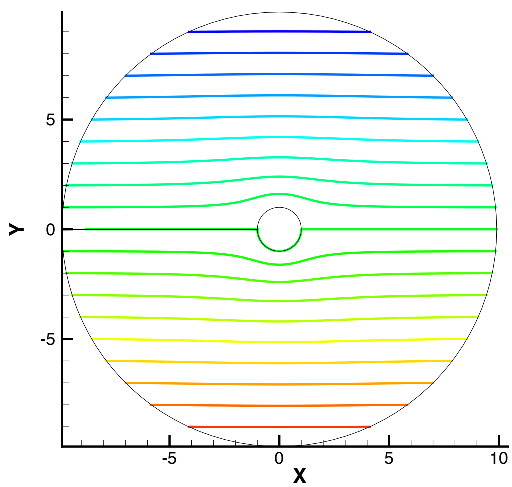
This is the *two-surface* method. Could add a line inside the domain and make it 3 surface.

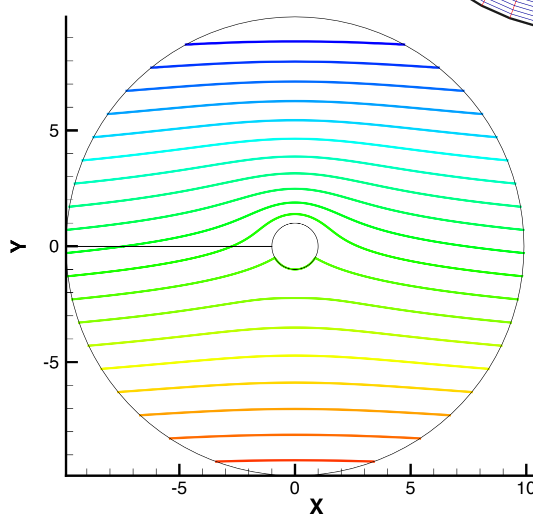
O-grid for cylinder by same method:

```
 DO it = 1,NTH ! grid walls
   th = 2π (it-1)/float(NTH-1)  !1=NTH
   xo = 10 cos(th)
   yo = 10 sin(th)
   xi = cos(th)
   yi = sin(th)

  DO ir=1,NR     ! grid interior
    xx(it,ir)=xi+(xo-xi)*(ir-1)/(NR-1)
    yy(it,ir)=yi+(yo-yi)*(ir-1)(NR-1)
  ENDDO
ENDDO
```







B. Grid stretching: change to (0 < S < 1): either S(j) is non-linear or ΔS(j) is specified.

```
DO i=1,I
   DO j=1,J
     x(i,j) = xin(i) + (xout(i)-xin(i))*S(j)
     y(i,j) = yin(i) + (yout(i)-yin(i))*S(j)
   ENDDO
ENDDO
```

where $0 \le S(j) \le 1$ is non-uniformly spaced.

1. ΔS method: Compound interest grid:

   $S(j+1) = S(j) + r^j \Delta S_1$, $j > 1$ with $S(1)=0$

   r is expansion ratio of grid spacing.

   $r = 1.05$, $J = 100$ gives $\Delta S_1 = 4\ 10^{-4}$

   $r = .95$, $J = 100$ gives $\Delta S_1 = 0.05$

$$\sum_1^{J-1} S(j+1) = \sum_1^{J-1} S(j) + \sum_1^{J-1} r^j \Delta S_1$$

$$S(J) = S(1) + \Delta S_1 \sum_1^{J-1} r^j = \Delta S_1 \frac{r^J - 1}{r - 1} = 1$$

$$\Delta S_1 = \frac{r-1}{r^J - 1}$$

2. Stretching formula example (Fletcher's book).

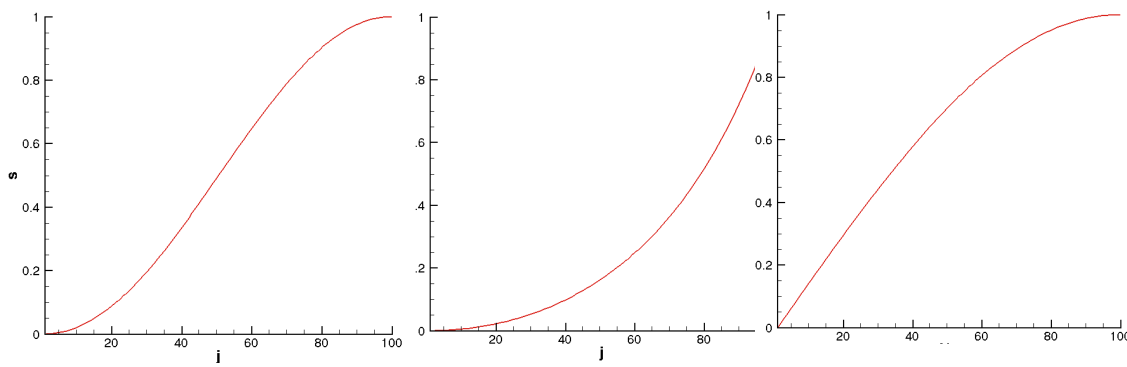
   Let $r_j = j\text{-}1/J\text{-}1$

   $$S(j) = P\, r_j + (1-P)\left\{ 1 - \tanh[Q(1-r_j)] / \tanh Q \right\}$$

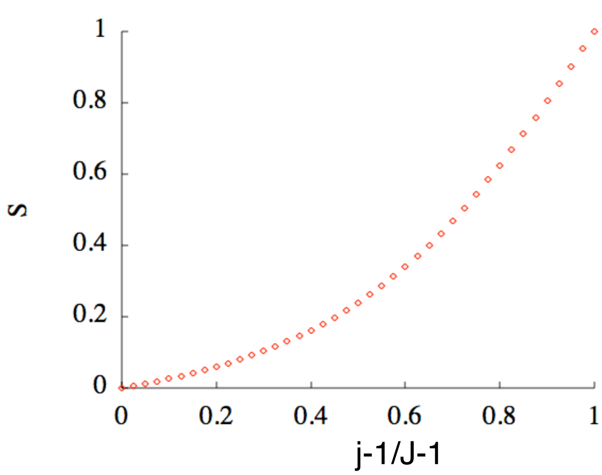
   E.g.: `P=0.1; Q=2.0`

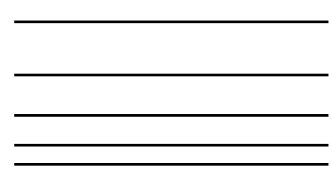
```
  DO i=1,I
   DO j=1,J
     r = float(j-1)/float(J-1)
     S(j) = r*P+(1-P)*(1.-tanh(Q*(1-r))/tanh(Q))
     x(i,j) = xin(i) + (xout(i)-xin(i))*S(j)
     y(i,j) = yin(i) + (yout(i)-yin(i))*S(j)
   ENDDO
 ENDDO
```

General idea:



Stretch at both ends   Stretch at lower wall   or upper wall

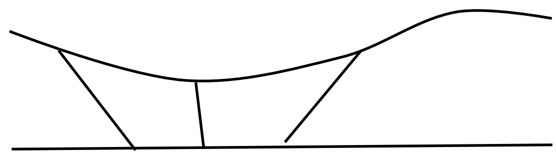




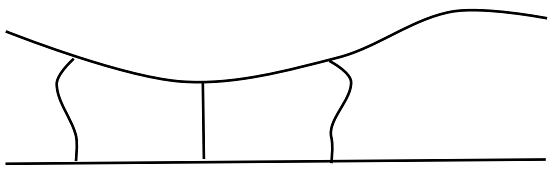
grid lines

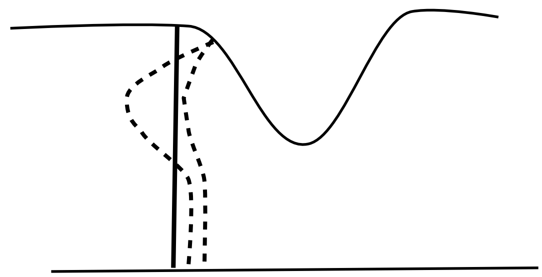
C. Orthogonality at walls

1. Simply connecting lines between wall points may produce a very skewed grid



Make normal to walls





Why orthogonal? Heat flux, tri-diagonal implicit matrix

2. Interpolate grid line based on value and derivative at ends, i.e., X(1), dX(1)/dn

   Value only: $(x,y) = \mathbf{X} = \mathbf{a} + \mathbf{b}\, s$, $0< s <1$ is straight line between walls: $a = X_{in}$, $b = X_{out}-X_{in}$

   If slope is specified at both walls, need two more parameters

   $\mathbf{X} = \mathbf{a} + \mathbf{b}\, s + \mathbf{c}\, s^2 + \mathbf{d}\, s^3$

3. Hermite interpolation

   Value and derivative

   Given wall shape, can find tangent, then normal
   $\mathbf{t}_{in} = (dx_{in}, dy_{in})/ dl_{in}$ $dl^2= dx^2 + dy^2$ unit tangent.

   $(\mathbf{n}_x, \mathbf{n}_y)_{in} = \pm (-dy_{in}, dx_{in})/ dl_{in}$ unit normal ($\mathbf{t}_1 \times \mathbf{t}_2$; in 2-D $\mathbf{t}_2 = \hat{\mathbf{z}}$)

   Numerically $dx_{surface}(i) = (x_{surface}(i+1) - x_{surface}(i-1))/2 = \delta x/\delta\xi$

   Grid line ∥ to normal: $(dx/ds,dy/ds)_{in} = P\, \mathbf{n}_{in}$ where P is a constant
   Similarly $(dx,dy)_{out} = Q\, \mathbf{n}_{out}$ where Q is a constant. These plus specified endpoints $(x,y)_{in}$, $(x,y)_{out}$ provide 4 conditions in x and y.
   Choose P,Q small enough to avoid crossing, but ~1.

Cubic polynomial for x and y coordinates

$x = a + b\,s + c\,s^2 + d\,s^3 \quad 0< s <1$
$y = e + f\,s + g\,s^2 + h\,s^3$

4. Hermite interpretation (recall Lagrange)

$H(s) = a + b\,s + c\,s^2 + d\,s^3$

	H1	**H2**	**H3**	**H4**
H(0)	1	0	0	0
H'(0)	0	0	1	0
H(1)	0	1	0	0
H'(1)	0	0	0	1

$H1 = a + b\,s + c\,s^2 + d\,s^3$

At s=0 a=1, b=0 obvious. At s=1 1+c+d=0, 2c+3d=0; c=-3d/2, d=2

H2: a=b=0, c+d=1, 2c+3d=0; c=-3d/2, d=-2

etc.

$H1(s) = 1 - 3s^2 + 2\,s^3$
$H2(s) = 3s^2 - 2\,s^3$
$H3(s) = s - 2s^2 + s^3$
$H4(s) = - s^2 + s^3$

Used as

$x(s) = x_{in}\,H1 + x_{out}\,H2 + Pn_{x\,in}\,H3 + Q\,n_{x\,out}\,H4$
$y(s) = y_{in}\,H1 + y_{out}\,H2 + Pn_{y\,in}\,H3 + Q\,n_{y\,out}\,H4$

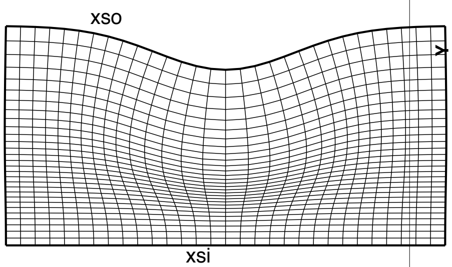
Grid

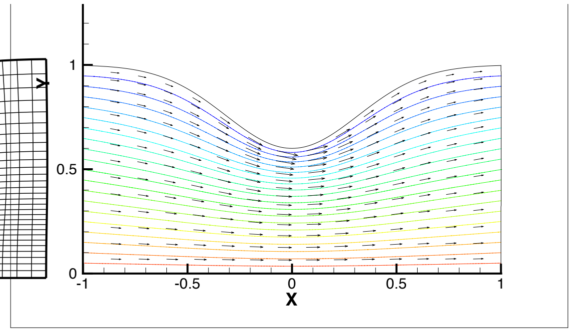
DO i ; Do j $s_j = f(j)$, e.g. (j-1)/J-1)

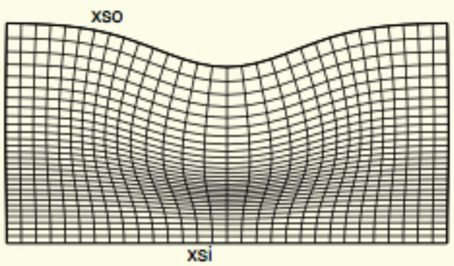
$x(i,j) = x_{in}(i)\,H1(s_j) + x_{out}(i)\,H2(s_j) + Pn_{x\,in}(i)\,H3(s_j) + Q\,n_{x\,out}(s)\,H4(s_j)$
$y(i,j) = y_{in}(i)\,H1(s_j) + y_{out}(i)\,H2(s_j) + Pn_{y\,in}(i)\,H3(s_j) + Q\,n_{y\,out}(s)\,H4(s_j)$

ENDDO

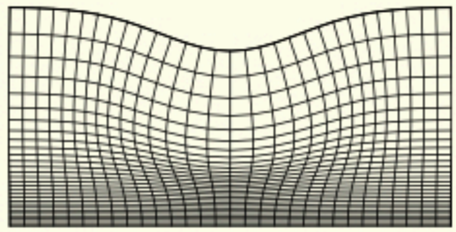
Two-surface method → *Multi-surface method*



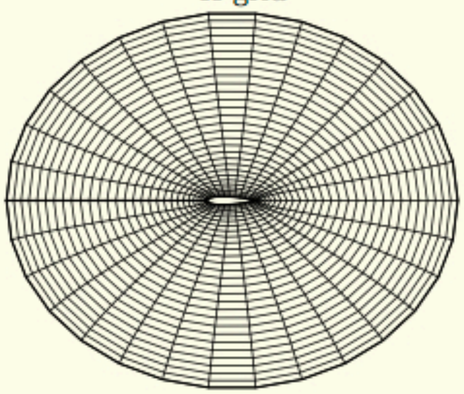




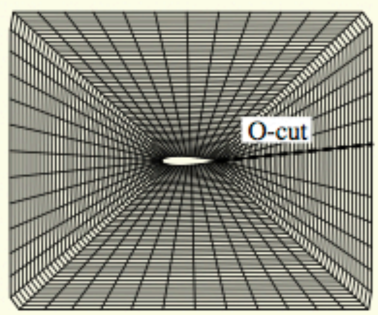
H-grid



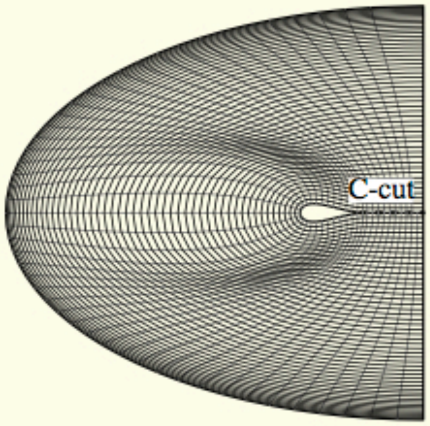
H-grid, with strecthing



O-grid (periodic)



O grid in rectangular domain



C-grid

Pseudo Code

! Prescribe or read in the shape of the inner and outer surfaces for i=1,NX

```
 geometry:  SELECT CASE(geom)
   case(1)              !  O-grid
     kperiodic = .true.
     xi,yi   = ellipse for example
     xo,yo = circle  for example
   case(2)              ! H-grid
     -1< xi< 1,   yi = 0         is lower wall
     -1< xo< 1,  yo= H(xo)   is duct shape
   case(3)              !  C-grid
     xso , yo is half ellipse
     xsi , yi is airfoil plus C-cut
  END SELECT geometry

!----     Two Surface Method  Pseudo code    ----
!  Arrays (xo,yo) and (xi,yi) define outer and inner walls
!  NB: Assumes that normal direction is (-dy,dx):
!        if grid goes into wall, reverse sign on r.h.s.
!----
  xloop: DO i = 1,NX
! ----
! Evaluate unit normal at inner and outer walls
! ----
    (xni,yni)  =  (-dyi,dxi)/sqrt(dxi^2+dyi^2)         = normal to inner wall
    (xno,yno)  =  (-dyo,dxo)/sqrt(dxo^2+dyo^2)  = normal to outer wall
```

E.G. : *dxi(i) = [xi(i+1)-xi(i-1)]/2 etc for yi and xo, xo ; IF(i=1,NX) use periodicity (O-grid) or 1-sided formula*

```
!----
! Hermite interpolation: define cubic polynomials H_i(s) such that
!----
     Pp = 1.0   ! adjustable parameter for inner wall
     Qq = 2.0   ! parameter for outer wall

   yloop: DO j=1,NY
        s  =  float(j-1)/float(NY-1)     ! recall stretching s = r*P+(1.-P)*(1.-tanh(Q*(1.-r))/tanh(Q))
        H1 =  1-3s^2+2s^3
        H2 =  3s^2-2s^3
        H3 = (s^3-2s^2+s)
        H4 = (s^3-s^2)
        xx(i,j) = xi*H1+xo*H2+Pp*xni*H3+Qq*xno*H4   ! x-coordinates of grid
        yy(i,j) = yi*H1+yo*H2+Pp*yni*H3+Qq*yno*H4   ! y-coorginates of grid
   ENDDO yloop
 ENDDO  xloop
```