To estimate Pi by using a circle, it’s circumscribed square and (lots of) random points within said square. Booth used Stata to estimate Pi, but here’s some R code to do the same thing…

x <- 0.5 # center x

y <- 0.5 # center y

n <- 1000 # nr of pts

r <- 0.5 # radius

pts <- seq(0, 2 \* pi, length.out = n)

plot(sin(pts), cos(pts), type = 'l', asp = 1) # test

require(sp)

xy <- cbind(x + r \* sin(pts), y + r \* cos(pts))

sl <- SpatialPolygons(list(Polygons(list(Polygon(xy)), "polygon")))

plot(sl, add=FALSE, col = 'red', axes=T )

# the square

xy <- cbind(c(0, 1, 1, 0), c(0, 0, 1, 1))

sq <- SpatialPolygons(list(Polygons(list(Polygon(xy)), "polygon")))

plot(sq, add = TRUE)

N <- 1e6

x <- runif(N, 0, 1)

y <- runif(N, 0, 1)

sp <- SpatialPoints(cbind(x, y))

plot(sp, add = TRUE, col = "green")

require(rgeos)

(sim\_pi <- (sum(gIntersects(sp, sl, byid = TRUE))/N) \*4)

sim\_pi - pi

Note the use of sp and rgeos packages to calculate the intersections.