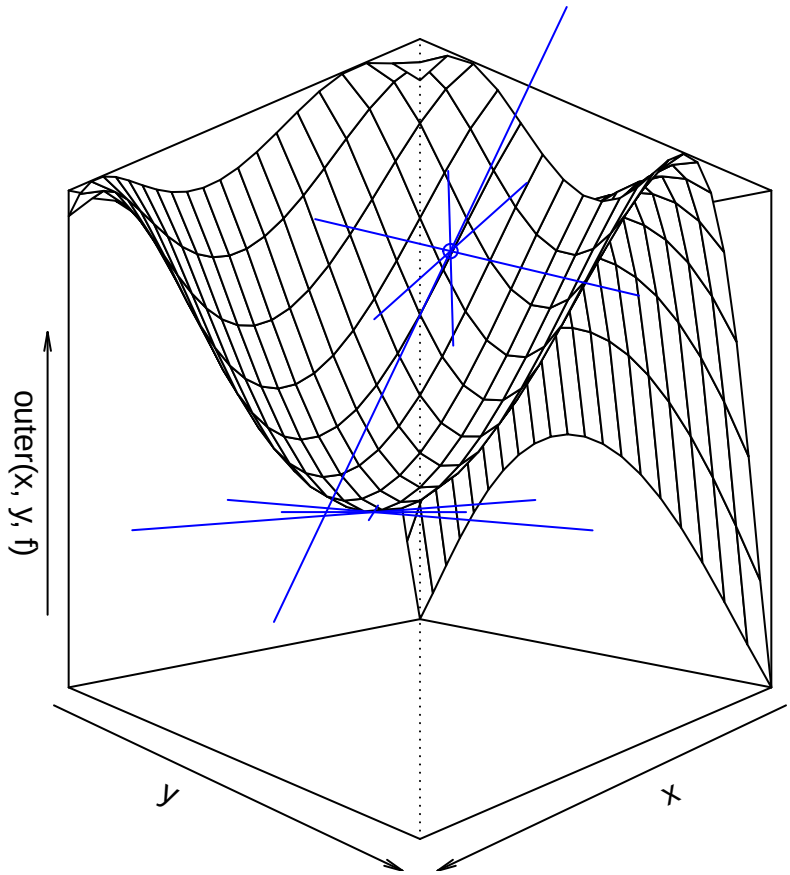
Symbolic-autodiff-in-R

```r
# Base R casually allows for AD black magic!

# Create function and gradient function
f <- deriv(expression(sin(2 * x^2 + y^2 - x)), c("x", "y"), func = T)

# Set values of independent variable
x <- y <- -10:10 / 10
l <- length(x)

# Function to plot tangent lines with gradients in direction of a vector
plot_tangent <- function(px, py, g, v){
  vxy <- rbind(x, y) * v
  lines(trans3d(vxy[1, ] + px, vxy[2, ] + py, f(px, py) + t(g %*% vxy),
                pmat = res), col = 'blue')
}

# Randomly choose points to evaluate gradient
n_pts <- 1
px <- x[sample(length(x), n_pts)]
py <- y[sample(length(y), n_pts)]

# Get gradient
g <- attr(f(px, py), "gradient")

# Plot function values
par(mar = rep(0, 4))
res <- persp(x, y, outer(x, y, f), phi = 0, theta = 135)

# Plot tangent points
points(trans3d(px, py, f(px, py), pmat = res), col = 'blue')

# Set direction vectors for gradients
sr2i <- 2^-1/2
v_mat <- matrix(c(1, 0, 0, 1, sr2i, sr2i, sr2i, -sr2i), nrow = 2)

# Plot tangent lines
for (i in 1:n_pts) {
  apply(v_mat, 2, function(v) plot_tangent(px[i], py[i], g[i, ], v))
}

# Find minimum using analytic gradient!
o <- optim(par = c(0.5, 0.5),
           fn = function(x) f(x[1], x[2]),
           gr = function(x) attr(f(x[1], x[2]), "gradient"),
           method = "BFGS")
```

```r
# Plot tangents at minimum
apply(v_mat, 2, function(v)
  plot_tangent(o$par[1], o$par[2], attr(f(o$par[1], o$par[2]), "gradient"), v))
```




```
## NULL
```