# Teaching Statistical Computing Using 3D Graphics in R

Duncan Murdoch

Department of Statistical and Actuarial Sciences
University of Western Ontario

June 11, 2010

# Teaching Using 3D Graphics in R

Duncan Murdoch

Department of Statistical and Actuarial Sciences
University of Western Ontario

June 11, 2010

# Outline

# Why use R?

R is a free software environment for statistical computing and graphics.

- A GNU project distributed under the GPL: students get it for free, and can keep it after the course.
- It runs on a wide variety of platforms. Our university facilities are mostly MS Windows, but the students have a variety of different machines.
- It is highly extensible, with thousands of user-contributed packages available. Our later statistical courses use actuarial packages and others.

# R is an environment for statistical computing and graphics

- Data handling and storage
- Calculations on vectors, matrices and more general arrays and structures
- Tools for data analysis
- Graphical support for interactive display and publication quality printing
- A well-developed programming language
- Software development support, including documentation and testing

# SS 2864: Statistical Programming

- Introductory programming course for 50–80 statistical and actuarial students.
- Starts with programming; uses R.
- Continues with Monte Carlo simulation, computational linear algebra, and numerical optimization.
- Uses both “classic” S graphics and `rgl` for debugging and understanding theory and algorithms.
- Today: singular value decompositions, Nelder-Mead and Newton-Raphson optimization, and *discussion*.

# The Singular Value Decomposition

For a square $n \times n$ matrix $A$, the SVD is

$$A = UDV^T \tag{1}$$

where

- $U$ and $V$ are $n \times n$ orthogonal matrices (i.e. $U^T U = V^T V = I$)
- $D$ is an $n \times n$ diagonal matrix with non-negative entries
- the superscript $T$ indicates matrix transposition.

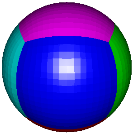
# Displaying a Matrix Graphically

- Matrices are representations of linear operators on vector spaces.
- The matrix $A$ is characterized by the behaviour of $y = Ax$ as we vary $x$.
- Use the `rgl` package to develop a graphical representation of $3 \times 3$ matrices.
- While the action on the basis vectors is mathematically sufficient, it is hard to visualize the overall effect of the transformation.
- We prefer to use coloured spheres.

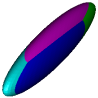
Demo 1

# Displaying the SVD

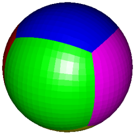
$$A = \begin{pmatrix} 1 & 0.1 & 0.1 \\ 2 & 1 & 0.1 \\ 0.1 & 0.1 & 0.5 \end{pmatrix}$$



Identity

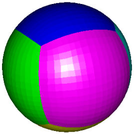


A



U

D



V

# Interpolating the SVD

- Static images of a matrix are harder to interpret than dynamic ones.
- We can make an SVD dynamic by interpolating between the identity and each component.
- When $U$ and $V$ are simply rotations, interpolation is linear interpolation of the rotation angle.
- When the singular values are all positive, linear interpolation on the log scale works well.

# Interpolating the SVD

- Static images of a matrix are harder to interpret than dynamic ones.
- We can make an SVD dynamic by interpolating between the identity and each component.
- When $U$ and $V$ are simply rotations, interpolation is linear interpolation of the rotation angle.
- When the singular values are all positive, linear interpolation on the log scale works well.
- I don't worry about complete generality in the display!

Demo 2

# Does it work?

I put these demos together to:

1. Teach the SVD.
2. Teach programming.
3. Teach visualization techniques.
4. De-mystify computer graphics.

The last two goals place tight constraints on what I can do. Have I achieved the right balance?

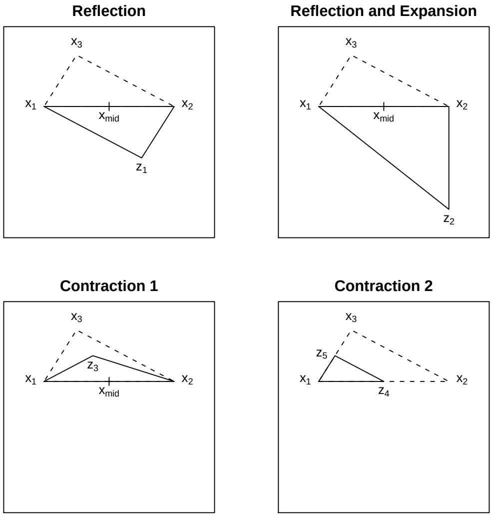
# The Nelder-Mead Simplex Method

- A robust derivative-free multi-dimensional minimizer.
- Easy to describe and to visualize
- Implementations of it are within the reach of our introductory students.
- Not very fast, and the visualizations help to illustrate why.

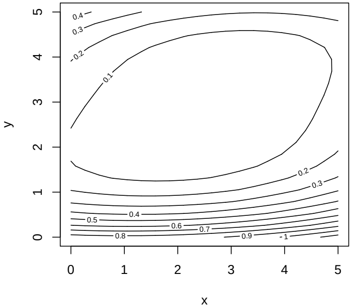
# How Nelder-Mead Works

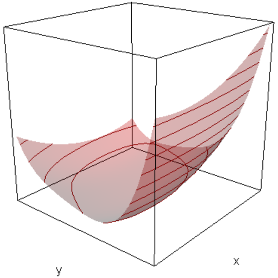
- Start with a non-degenerate simplex in the space of the arguments to the target function.
- Iterate through updates of the simplex until the simplex is determined to be close enough to a local minimum.
- Updates replace the vertex with the highest function value with a new one, either by shrinking, expanding, or reflecting the simplex through the centroid of the other vertices, or shrinking the entire simplex.

# Nelder-Mead Proposals



# Two Dimensional Example

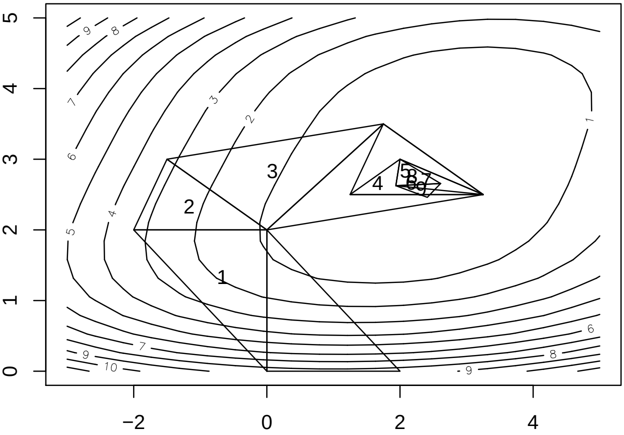



$$f(x,y) = [(x-y)^2 + (x-2)^2 + (y-3)^4]/100$$

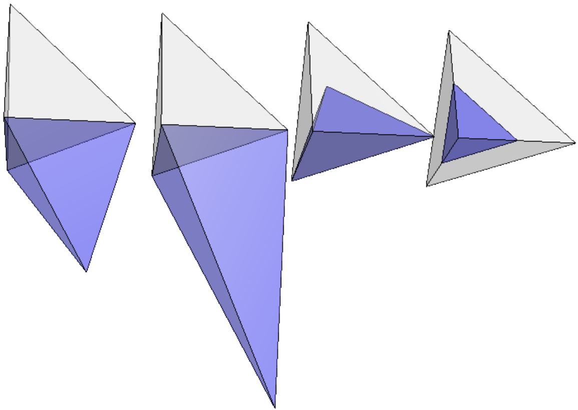
Demo 3

# Nelder-Mead path

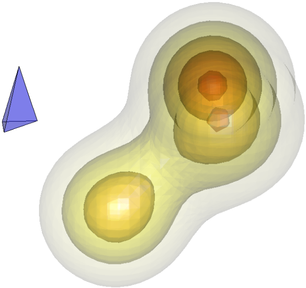


# Nelder-Mead in higher dimensions

One of the nice features of the Nelder-Mead description is that it is dimension-independent. The four moves in 3-D:



# Three Dimensional Example



Density of mixture of three normals (from `misc3d` package), together with initial simplex.

Demo 4

# Newton-Raphson

Newton-Raphson minimizes a function by a sequence of quadratic approximations. We can display these for functions of two variables.

Demo 5

# Does it work?

I put these demos together to:

1. Teach optimization.
2. Teach programming.
3. Teach visualization techniques.
4. De-mystify computer graphics.

The last two goals place tight constraints on what I can do. Have I achieved the right balance?

# Conclusions

- We show students that it is possible to generate relatively sophisticated graphics in a fairly easy way.
- Students are already computer users (as game players, etc.); in our class they learn how to be in control.
- They also learn something about linear algebra, optimization, Monte Carlo methods.

What other demos would work?