Operations Research Techniques and Algorithms (620-361)

Dr Yao-ban Chan

y.chan@ms.unimelb.edu.au
Telephone: 8344 9073
Office: Room 198, Richard Berry Building

Christina Burt
c.burt@ms.unimelb.edu.au
Telephone: 8344 1797
Office: 139 Barry St

May 7, 2008, 2008

```
function [x ft iterations] = steepestDescent(f, gradf, initial, sigma, mu, tol)
% f : the function we wish to minimise (e.g.f = x(1)^2 + x(2)^2)
% gradf : the gradient function of f (e.g. gradf = [2 * x(1), 2 * x(2)])
% initial: is the starting point (e.g.[10, 10])
% tol : is the epsilon value that we use as stopping criteria (e.g. 0.0000001)
% % We also require two further inputs for the Armijo-Goldstein and Wolff function:
% sigma : an initial sigma value greater than zero but less than one (e.g. 0.25)
% mu : an initial mu value greater than sigma but less than one (e.g. 0.75)

%STEP ONE
%first create a counter for iterations of the algorithm
iterations = 0;
%note that x is an array, and 'initial' are your initial values for x
x = initial;

%STEP TWO
%The descent direction
d = -gradf(x);
```

```
while(norm(d) > tol)

    %STEP THREE
    %The procedure to find the stepsize satisfying AGW conditions
    %Calculate the function and gradient at the start point
    f0 = feval(f, x);
    fd0 = feval(gradf, x) * d';

    %Set initial boundaries on the stepsize
    tlo = 0;
    thi = 10000;
    %Set a stepsize to start from and reset the counter
    t = 1;
    counter = 0;

    %Evaluate what happens with the initial stepsize, t = 1
    ft = feval(f, x + t * d);
    fdt = feval(gradf, x + t * d) * d';

    and so on...
```

```
    %STEP FOUR
    %add another one to your iterations counter
    iterations = iterations + 1;
    %update your position according to the descent direction, d and
    %stepsize t
    x = x + t*d;

    %recalculate d before looping back to the start of the while
    %condition
    d = -gradf(x);

end
```

The steepest descent algorithm

1. Select starting point x^0.
   Set iteration counter $k = 0$.
2. Calculate the descent direction $d^k = -\nabla f(x^k)$.
   If $||d^k|| < \epsilon$ then stop.
3. Select step length t_k by running the procedure to find a stepsize that satisfies both the Armijo-Goldstein and Wolff conditions.
4. Add 1 to the iteration counter: $k = k + 1$.
   Update our position $x^{k+1} = x^k + t_k d^k$.
   Return to step 2.

A procedure for finding a stepsize that satisfies AGW

1.

$$\begin{aligned} t_{lo} &= 0 \\ t_{hi} &= \infty \\ t &= T \end{aligned}$$

2. If $f(t) > f(0) + t\sigma f'(0)$, then

$$\begin{aligned} t_{hi} &= t \\ t &= 1/2(t_{lo} + t) \end{aligned}$$

Else if $f'(t) < \mu f'(0)$, then

$$\begin{aligned} t_{lo} &= t \\ t_{hi} &= \begin{cases} 1/2(t_{lo} + t_{hi}) & \text{if} \quad t_{hi} < \infty \\ 2t & \text{otherwise} \end{cases} \end{aligned}$$

Repeat until $f(t) \leq f(0) + t\sigma f'(0)$ and $f'(t) \geq \mu f'(0)$.

The directional derivative

The directional derivative of f at x in the direction d is defined as

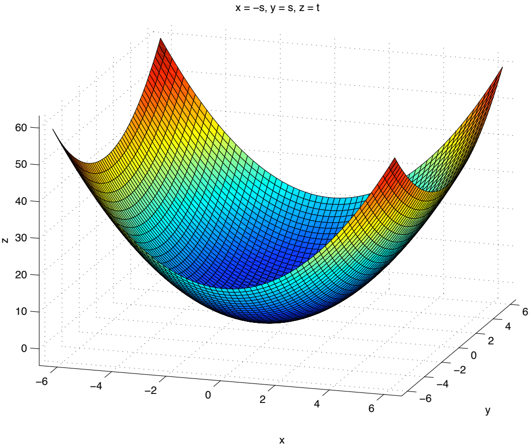
$$f'(x;d) \stackrel{def}{=} \lim_{t \to 0} \frac{f(x+td) - f(x)}{t}$$

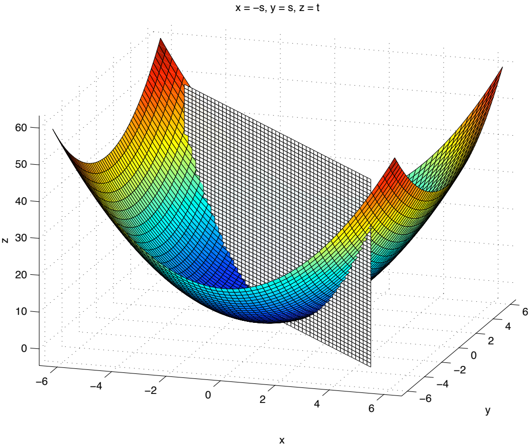
This is essentially the one-dimensional derivative of the slice of f along the (arbitrary) direction d. To say that f is differentiable at x, we require that directional derivatives exist for all directions d.

It is not difficult to show that

$$f'(x;d) = \nabla f(x)^T d.$$

We say f is C^1 or continuously differentiable if it is differentiable and the gradient function ∇f is continuous.


x = −s, y = s, z = t
z
60
50
40
30
20
10
0
−6
−4
−2
0
2
4
6
x
y



x = −s, y = s, z = t
60
50
40
z
30
20
10
0
−6
−4
−2
0
2
4
6
x
6
4
2
0
−2
−4
−6
y