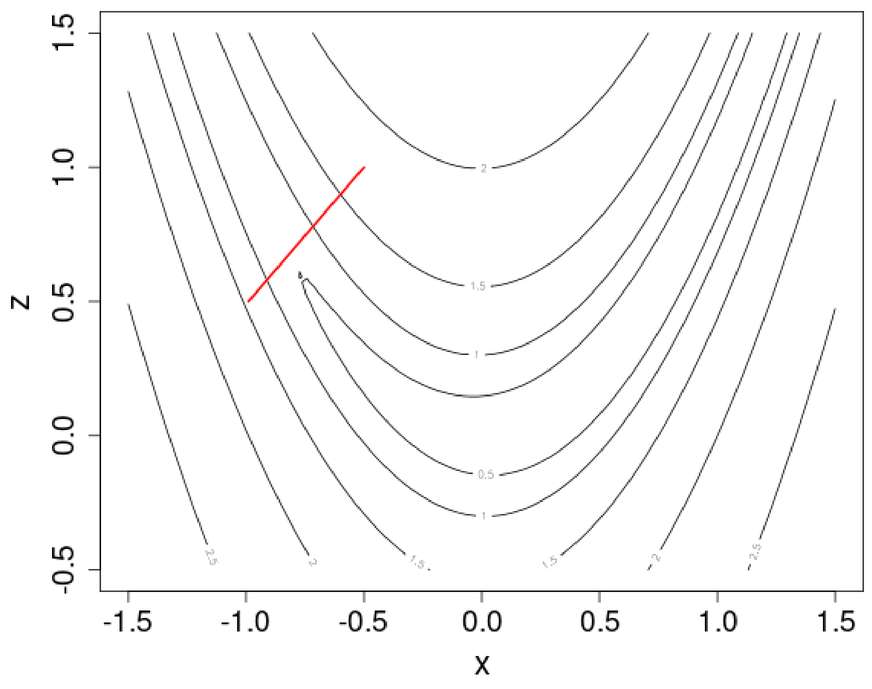
APTS Statistical Computing animations

© Simon N. Wood (2012) s.wood@bath.ac.uk

1 Optimization



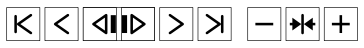


Figure 1: Steepest decent method, with step length selection, applied to Rosenbrock's function (contours). Steps are shown in red. Note that after 40 steps or so the plot is output only every 200 steps. The method takes over 14,000 steps to converge.

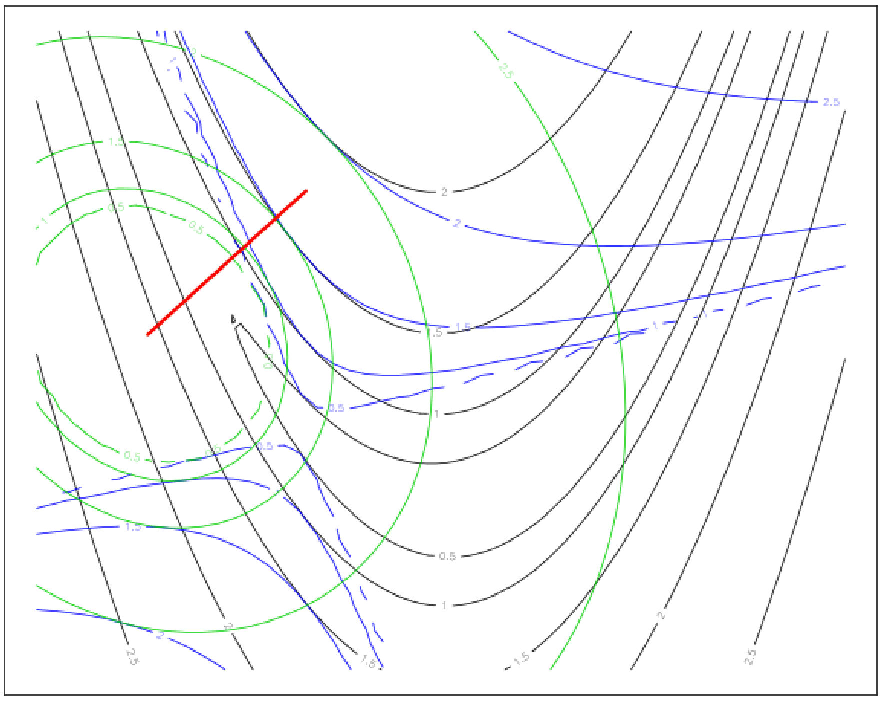


Figure 2: Newton's method with step halving and Hessian forced to be positive definite if needed. The black contours are Rosenbrock's function. Blue contours, if present, show the current quadratic approximation based on the gradient and original Hessian. Green is the quadratic approximation used to find the step, based on the gradient and Hessian, or perturbed Hessian. Red shows the steps taken.

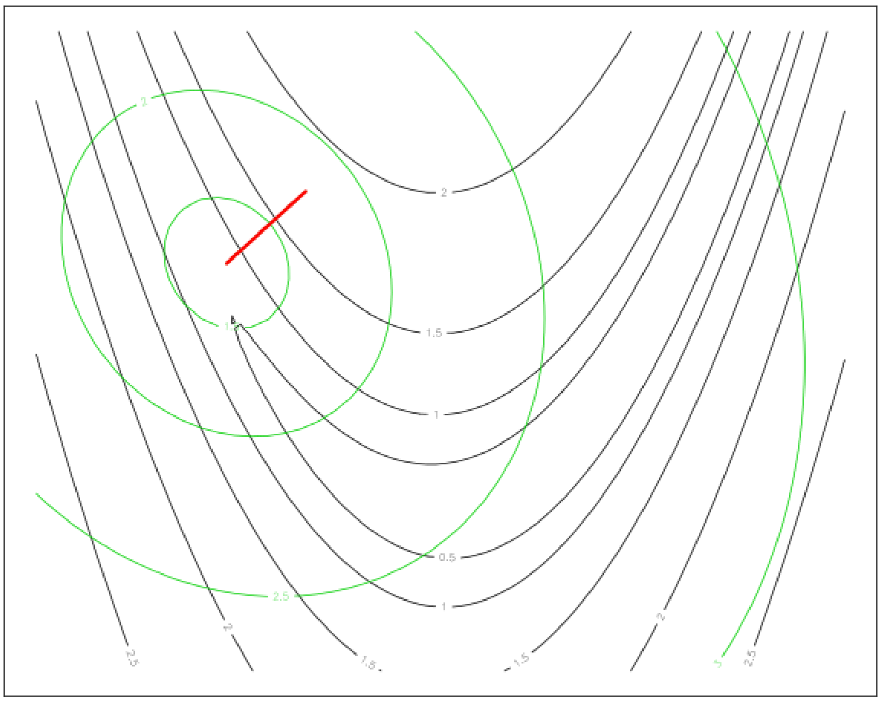


Figure 3: BFGS quasi-Newton method. Black is objective, Green is approximating quadratic model. Red shows steps.

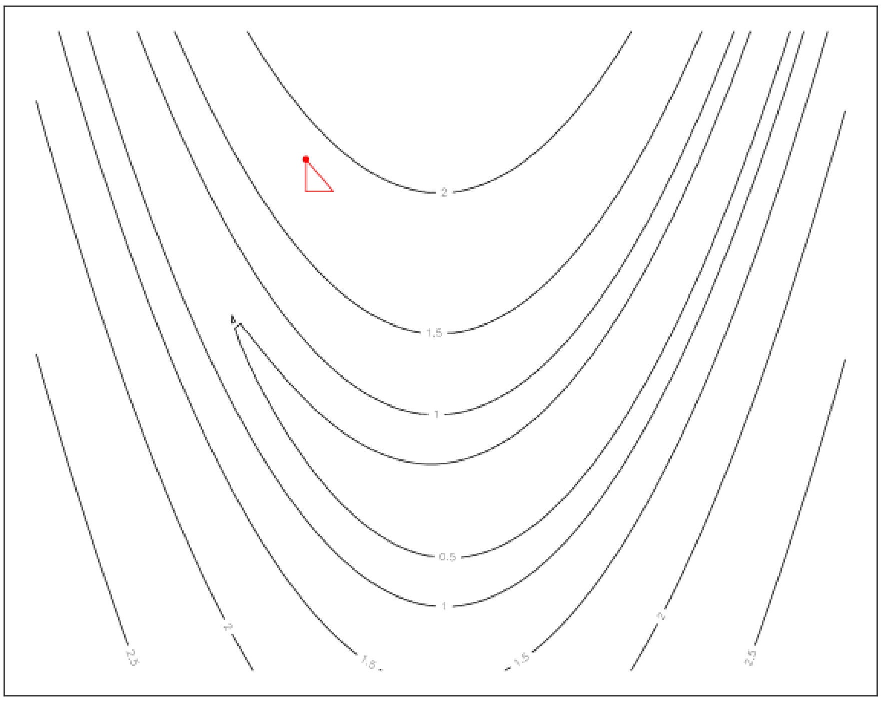


Figure 4: Nelder-Meade polytope method. Objective function is contoured in black. Successive polytopes are shown in different colours. The worst point (highest objective) in each polytope is marked with a filled circle (blob).

2 Auto-differentiation

These animations are based on a figure in Chapter 8 of Nocedal, J. & S.J. Wright (2006) *Numerical Optimization* (2nd ed.), Springer.

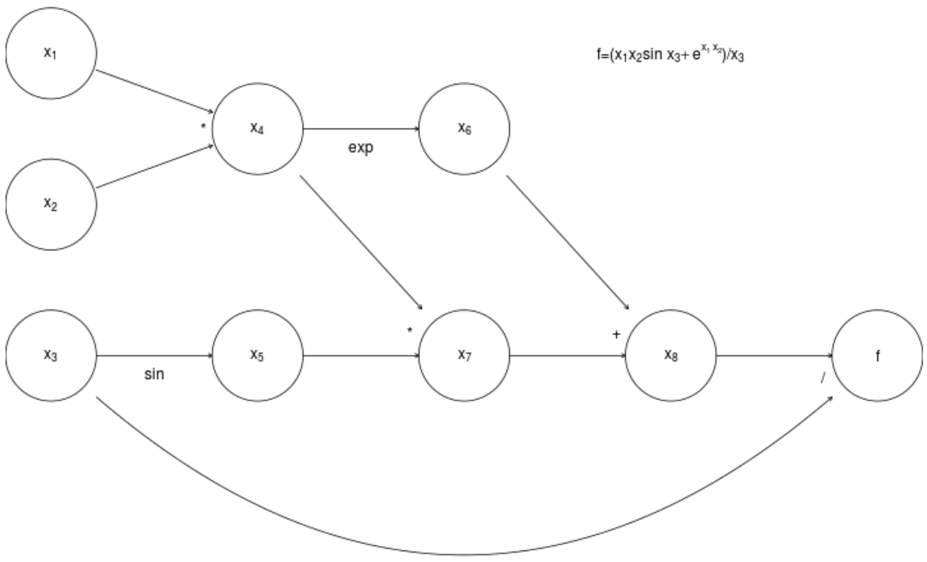


Figure 5: Example computational graph, showing how computation is broken down into elementary operations and functions.

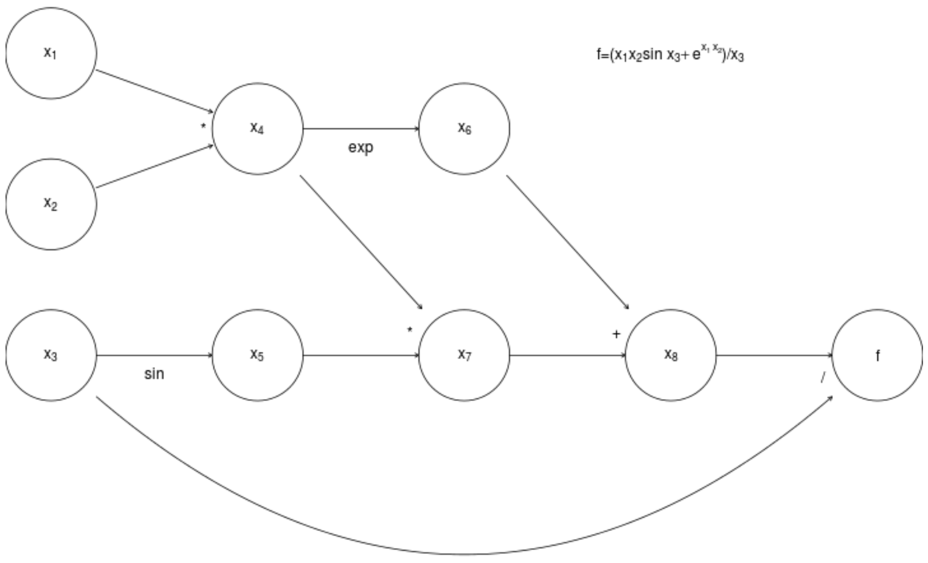


Figure 6: Forward mode autodifferentiation. The chain rule is used to combine derivatives of each node w.r.t. its parents to obtain derivative of node with respect to input nodes (leftmost). Eventually we obtain the derivative of the function (rightmost node) w.r.t. the input nodes. To avoid clutter only the propagation of the derivative w.r.t. x_1 is shown, but in reality derivatives w.r.t. all input nodes are propagated.

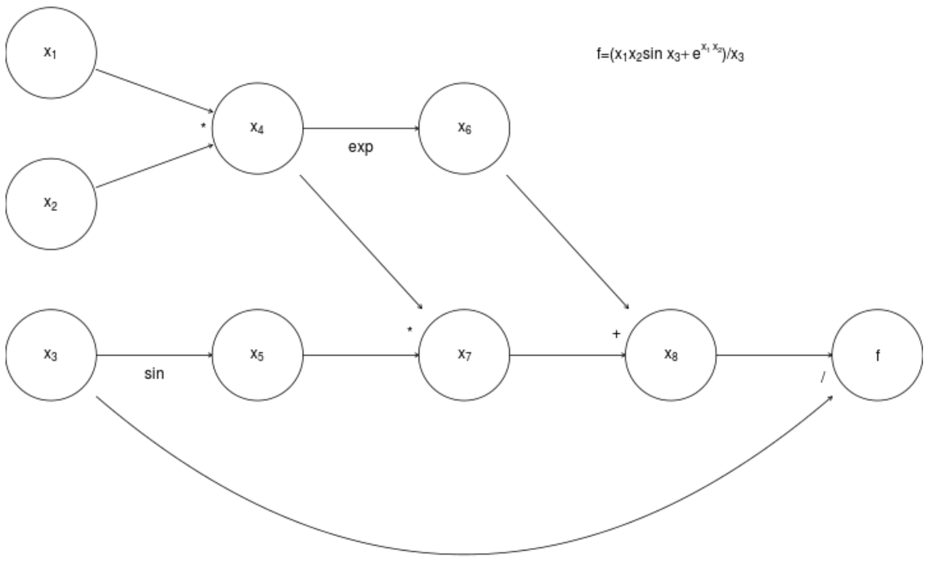


Figure 7: Reverse mode auto-differentiation. The computational graph is first traversed in the forward direction. Nodes are evaluated and stored along with derivatives of nodes w.r.t. parents. Then the graph is traversed in the reverse direction: the chain rule is used to find the derivative of the function w.r.t. the current node based on the already computed parent child derivatives. At the end we have derivatives of the function w.r.t. all input nodes, at much less cost in than forward mode, since we needed to compute only one derivative per node. But this gain in flop count is paid for by the memory cost of having to store all the intermediate node and derivative values.