Majorizing Cubics on Intervals

Jan de Leeuw

Version 15, December 11, 2016

Abstract

We illustrate uniform quadratic majorization, sharp quadratic majorization, and sublevel quadratic majorization using the example of a univariate cubic.

Contents

Note: This is a working paper which will be expanded/updated frequently. All suggestions for improvement are welcome. The directory deleeuwpdx.net/pubfolders/cubic has a pdf version, the complete Rmd file with all code chunks, the bib file, and the R source code.

1 Introduction

Suppose $\mathcal{I}$ is the closed interval $[L, U]$, and $f : \mathcal{I} \rightarrow \mathbb{R}$. A function $g : \mathcal{I} \otimes \mathcal{I} \rightarrow \mathbb{R}$ is a *majorization scheme* for f on $\mathcal{I}$ if

- $g(x, x) = f(x)$ for all $x \in \mathcal{I}$,
- $g(x, y) \geq f(x)$ for all $x, y \in \mathcal{I}$.

In other words for each y the global minimum of $g(x, y) - f(x)$ over $x \in \mathcal{I}$ is zero, and it is attained at y. If the functions f and g are differentiable and the minimum is attained at an interior point of the interval, we have $\mathcal{D}_1 g(x, y) = \mathcal{D} f(x)$. If the functions are in addition twice differentiable we have $\mathcal{D}_{11} g(x, y) \geq \mathcal{D}^2 f(x)$.

The majorization conditions are not symmetric in x and y, and consequently it sometimes is more clear to write $g_y(x)$ for $g(x, y)$, so that $g_y : \mathcal{I} \rightarrow \mathbb{R}$. We say that g_y majorizes f on $\mathcal{I}$ at y, or with *support point* y.

A *majorization algorithm* is of the form

$$x^{(k+1)} \in \underset{x \in \mathcal{I}}{\mathbf{argmin}}\, g(x, x^{(k)})$$

It then follows that

$$f(x^{(k+1)}) \leq g(x^{(k+1)}, x^{(k)}) \leq g(x^{(k)}, x^{(k)}) = f(x^{(k)}), \tag{1}$$

Thus a majorization step decreases the value of the objective function. The chain (1) is called the *sandwich inequality*. In (1) the inequality $f(x^{(k+1)}) \leq g(x^{(k+1)}, x^{(k)})$ follows from majorization, the inequality $g(x^{(k+1)}, x^{(k)}) \leq g(x^{(k)}, x^{(k)})$ follows from minimization. This explains why majorization algorithms are also called *MM algorithms* (Lange (2016 (in press))). Using the *MM* label has the advantage that it can also be used for the dual family of minorization-maximization algorithms.

In this note we are interested in *quadratic majorization*, i.e. in majorization functions of the form

$$g(x, y) = f(y) + f'(y)(x - y) + \frac{1}{2} K (x - y)^2, \tag{2}$$

and specifically on quadratic majorizers of a cubic on an closed interval of the real line. If $K \leq 0$ in (2) the majorizaton function is concave and attains its minimum at one of endpoints of $\mathcal{I}$. If $K > 0$ then define the *algorithmic map* $\mathcal{A}(x) = x - f'(x)/K$ and

$$x^{(k+1)} = \begin{cases} L & \text{if } \mathcal{A}(x^{(k)}) < L, \\ \mathcal{A}(x^{(k)}) & \text{if } L \leq \mathcal{A}(x^{(k)}) \leq U, \\ U & \text{if } \mathcal{A}(x^{(k)}) > U. \end{cases}$$

Assuming that the sequence $x^{(k)}$ converges to a fixed point x_∞ of $\mathcal{A}$, i.e a point with $\mathcal{D} f(x_\infty) = 0$, the rate of convergence is

$$\rho(x_\infty) = 1 - \frac{\mathcal{D}^2 f(x_\infty)}{K}.$$

2 Majorizing a Cubic

Suppose f is a non-trivial cubic, with non-zero leading coefficient. The function and its derivatives are

$$\begin{aligned} f(x) &= d + cx + bx^2 + ax^3, \\ f'(x) &= c + 2bx + 3ax^2, \\ f''(x) &= 2b + 6ax, \\ f'''(x) &= 6a. \end{aligned}$$

2.1 Uniform Quadratic Majorization

Define

$$K_0 = \max_{L \leq x \leq U} f''(x) = \max(f''(A), f''(B)).$$

Thus

$$K_0 = \begin{cases} 6aU + 2b & \text{if } a > 0, \\ 6aL + 2b & \text{if } a < 0. \end{cases}$$

Our majorization function is the quadratic

$$g(x, y) := f(y) + f'(y)(x - y) + \frac{1}{2}K_0(x - y)^2.$$

The corresponding majorization algorithm is

$$x^{(k+1)} = \mathop{\mathbf{argmin}}_{L \leq x \leq U} g(x, x^{(k)})$$

Note the majorizing function can be a concave quadratic, in which case its minimum is always at one of the endpoints of the interval. Assuming we eventualy converge to a value $L < x_\infty < U$ the convergence rate (or asymptotic error constant) is

$$\rho(x_\infty) := 1 - \frac{f''(x_\infty)}{K_0} = \begin{cases} \frac{6a(U - x_\infty)}{6aU + 2b} & \text{if } a > 0, \\ \frac{6a(L - x_\infty)}{6aL + 2b} & \text{if } a < 0. \end{cases}$$

Note that this does not depend on the lower limit L of the interval if $a > 0$. Convergence is faster if the upper limit U happens to be close to x_∞, and in fact it can be close to zero. If $U \rightarrow \infty$ the rate of convergence goes to one.

As a first example, consider $f(x) = \frac{1}{6}(1 - 2x + x^3)$. This cubic has roots at -1.6180339887, 0.6180339887, 1. There is a local maximum at $-\frac{1}{3}\sqrt{6}$ and a local minimum at $\frac{1}{3}\sqrt{6}$. The function (red), its first derivative (blue), and its second derivative (green) are in figure 1.

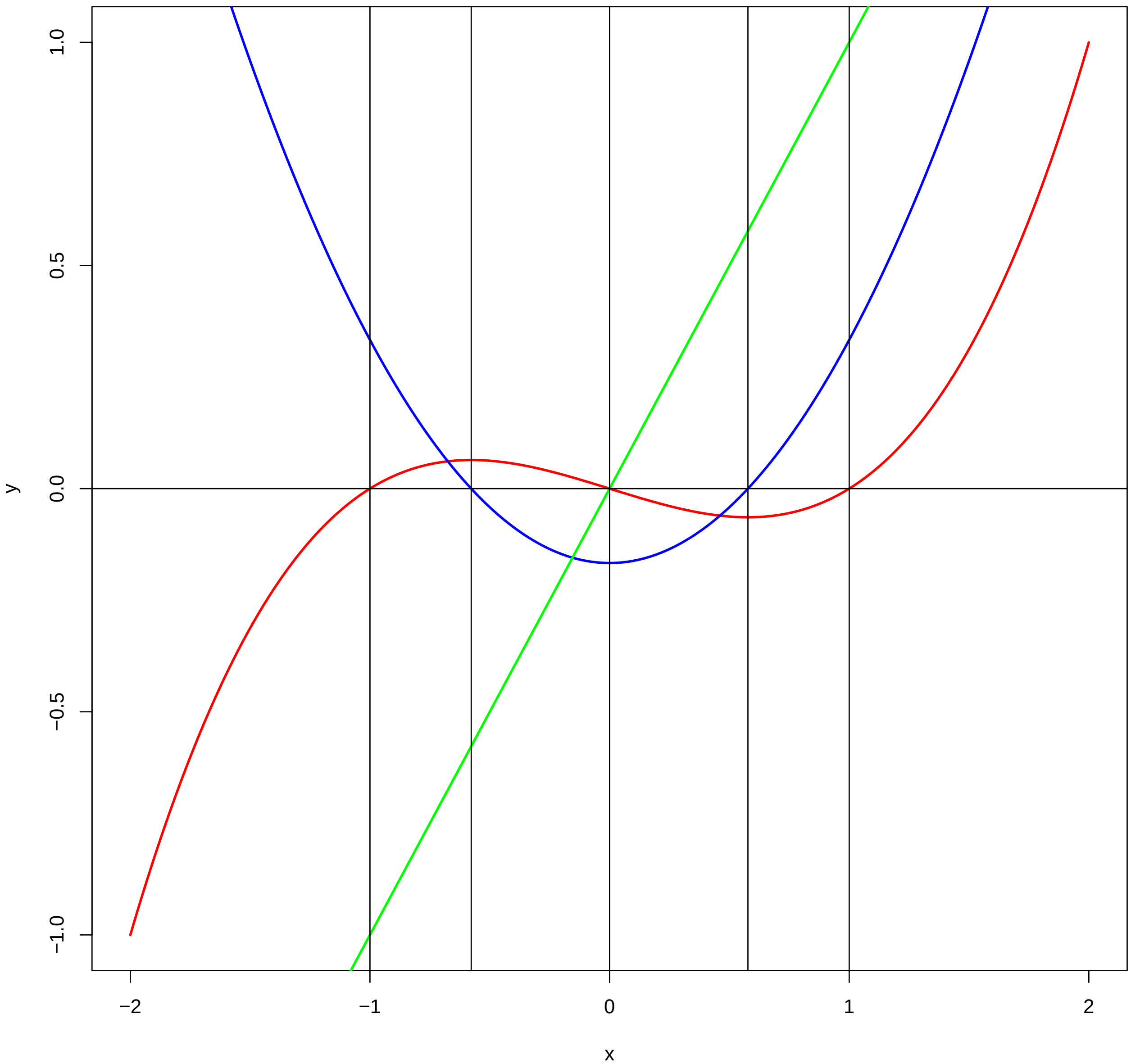

Figure 1: Example Cubic

We will look for a local minimum in the interval $[-2, 2]$, stating with initial value 1. Note that $f''(x) = x$ and thus $K_0 = B = 2$. At $x_\infty = \frac{1}{3}\sqrt{3}$ we have $\rho(x_\infty) = 1 - \frac{1}{6}\sqrt{3}$, i.e. approximately 0.7113248654. We report the results of the final iteration.

```
## Iteration:   35 xinit:  1.00000000 xfinal:  0.57735207 rate:  0.71132334
```

If we look for a local minimum in the interval $[0, 1]$ instead of $[-2, 2]$, we get much faster convergence, because the upper bound is now much closer to the solution.

```
## Iteration:   14 xinit:  0.50000000 xfinal:  0.57734974 rate:  0.42265183
```

If we start at a value to the left of $-\frac{1}{3}\sqrt{3}$ and look for a minimum in $[-2,2]$ then the algorithm converges to the boundary at -2.

```
## Iteration:    3 xinit:  -1.50000000 xfinal:  -2.00000000 rate:  0.00000000
```

Note that alteratively we could have used

$$K_0^+ := \max_{L\leq x\leq U} |f''(x)|$$

in our majorization fuctions. Since f'' is linear we see that $|f''|$ is convex, and thus $K_0^+ = \max(|f''(L)|, |f''(U)|$. Using K_0^+ gives a majorization which is generally less precise, but uses majorization functions that are always convex quadratics. Also note that if there is a strict local minimum in $\mathcal{I}$ then $K_0 > 0$, although we can still have $K_0 < K_0^+$. Think of $[-1, .75]$, for which $K_0 = .75$ and $K_0^+ = 1$.

2.2 Sharp Quadratic Majorization

Consider the general representation of a cubic around the point y

$$f(x) = f(y) + f'(y)(x-y) + \frac{1}{2}f''(y)(x-y)^2 + \frac{1}{6}f'''(x-y)^3.$$

For a quadratic function of the form (2) we have

$$g(x,y) - f(x) = \frac{1}{2}(x-y)^2\{K - f''(y) - \frac{1}{3}f'''(x-y)\}. \tag{3}$$

Thus the quadratic function is a majorizer if

$$K \geq \max_{x\in\mathcal{I}} f''(y) + \frac{1}{3}f'''(x-y) \tag{4}$$

which works out to $K \geq f''(y) + \frac{1}{3}\max\left(f'''(U-y), f'''(L-y)\right)$. We get the *sharp quadratic majorization* (De Leeuw and Lange (2009)) by choosing K equal to its lower bound. The corresponding rate in the cubic case, with positive leading coefficient, is

$$\rho(x_\infty) = \frac{\frac{1}{3}f'''(U-x_\infty)}{f''(x_\infty) + \frac{1}{3}f'''(U-x_\infty)}.$$

If we reanalyze our example with the sharp bound we find faster convergence in the first two computing runs.

```
## Iteration:   17 xinit:  1.00000000 xfinal:  0.57735073 rate:  0.45096147
```

```
## Iteration:    8 xinit:  0.50000000 xfinal:  0.57735010 rate:  0.19615217
```

We also find convergence to the local minimum, and not to the nearby boundary, in the third run.

```
## Iteration:   18 xinit:  -1.50000000 xfinal:  0.57735105 rate:  0.45096119
```

2.3 Sublevel Quadratic Majorization

We know that quadratic majorization of a cubic on the whole line is impossible. This is one of the reasons for looking at quadratic majorization on a closed interval, where a continuous second derivative is always bounded. In this section we relax the majorization requirements, using a closed interval that depends on the current solution and becomes smaller if we get closer to a minimum. The reuslting majorization method can be thought of as a safeguarded version of Newton's method.

A function $g : \mathcal{I} \otimes \mathcal{I} \rightarrow \mathbb{R}$ is a *sublevel majorization scheme* for f on $\mathcal{I}$ if

- $g(x, x) = f(x)$ for all $x \in \mathcal{I}$,
- $g(x, y) \geq f(x)$ for all $x, y \in \mathcal{I}$ for which $g(x, y) \leq g(y, y)$.

The second part of the definition says that g majorizes f on the sublevel set $\{x \in \mathcal{I} \mid g(x, y) \leq g(y, y)\}$. If we minimize the sublevel majorization the sandwich inequality (1) is still valid, so we still have monotone convergence of function values.

We quickly specialize this to quadratic majorization functions, suppose $\mathcal{I}$ is the whole real line, and also require $K \geq 0$. If g is given by (2) then the sublevel set is the interval between y and $y - 2f'(y)/K$. Note that either of the two bounds can be the smaller one. Thus we want inequality

$$K \geq f''(y) + \frac{1}{3}f'''(x - y)$$

on the sublevel set, or equivalently at both endpoints. Thus $K \geq \max(f''(y), 0)$ and

$$K \geq f''(y) - \frac{2}{3}\frac{f'''f'(y)}{K}.$$

This means we must have

$$K^2 - Kf''(y) + \frac{2}{3}f'''f'(y) \geq 0. \tag{5}$$

Define $K(y)$ as the smallest $K \geq \max(f''(y), 0)$ satisfying (5), and we have *sharp sublevel quadratic majorization* (De Leeuw (2006)).

If the quadratic equation corresponding to (5) has no real roots or a single real root, then the inequality (5) is satisfied for all K, and thus $K(y) = \max(f''(y), 0)$. If the equation has two real roots, they are written as $p(y) \leq q(y)$. We have $p(y) + q(y) = f''(y)$. Thus if $p(y)$ and $q(y)$ are non-negative, then $0 \leq p(y) \leq q(y) \leq f''(y)$, and consequently $K(y) = f''(y) \geq 0$. If $p(y) \leq 0$ and $q(y) \geq 0$ then $q(y) = f''(y) - p(y) \geq f''(y)$ and thus $K(y) = q(y)$. If both $p(y) \leq 0$ and $q \leq 0$ then $f''(y) \leq p(y) \leq q(y) \leq 0$, and thus $K(y) = 0$ and the sharp sublevel quadratic is linear.

Figure 2 shows for $y = \frac{1}{2}$ and various values of K what sublevel majorization looks like. The function is in red, the quadratic sublevel majorization in blue. Note the different lengths of the sublevel intervals. We see that $K = \frac{1}{2}$ is too small. For $y = \frac{1}{2}$ the quadratic is $K^2 = K - \frac{1}{2}K - \frac{1}{36}$, which has roots -0.0504626063, 0.5504626063 and thus the sharp sublevel quadratic has K equal to 0.5504626063.

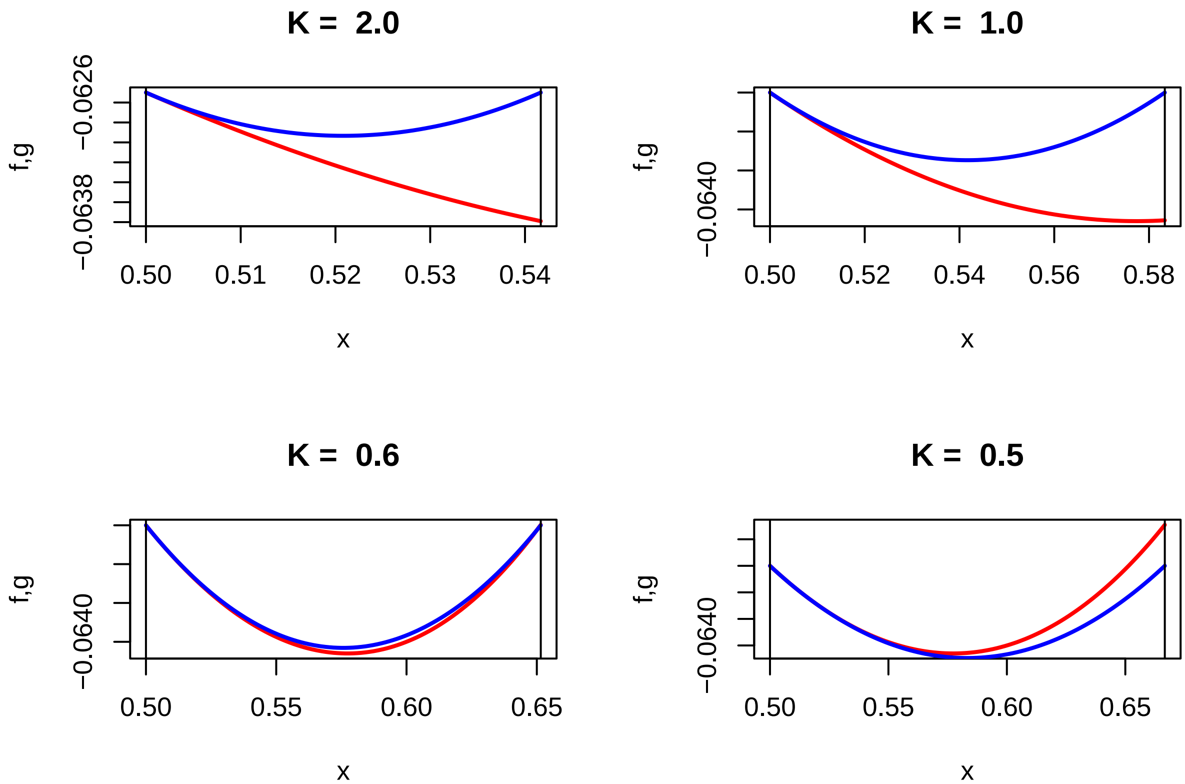

Figure 2: Sublevel Majorization

If we are close to a strict local minimum x we have $f'(x) \approx 0$ and $f''(x) > 0$. Thus the quadratic will have one root approximately zero and one root approximately equal to $f''(x)$, and the iteration is basically a Newton iteration. Thus, at least for cubics, sublevel majorization has quadratic convergence. We illustrate this by analyzing our small example with sublevel majorization, starting from $x = 1$ and $x = \frac{1}{2}$.

```
## Iteration:    1 xold:    1.00000000 xnew:    0.66666667 cnew:    0.33333333 rate:
## Iteration:    2 xold:    0.66666667 xnew:    0.58333333 cnew:    0.08333333 rate:
## Iteration:    3 xold:    0.58333333 xnew:    0.57738095 cnew:    0.00595238 rate:
## Iteration:    4 xold:    0.57738095 xnew:    0.57735027 cnew:    0.00003068 rate:
## Iteration:    5 xold:    0.57735027 xnew:    0.57735027 cnew:    0.00000000 rate:
```

```
## Iteration:    1 xold:    0.50000000 xnew:    0.57569391 cnew:    0.07569391 rate:
## Iteration:    2 xold:    0.57569391 xnew:    0.57734948 cnew:    0.00165557 rate:
## Iteration:    3 xold:    0.57734948 xnew:    0.57735027 cnew:    0.00000079 rate:
```

If $\mathcal{I} = [L, U]$ then our analysis must be modified slightly. We want inequality (4) on the intersection of the sublevel interval and $[L, U]$. If the sublevel interval is in $[L, U]$ our previous analysis applies. Because we always have $y \in [L, U]$ the other possible intervals are $[y, U]$ if $y \leq U \leq y - 2f'(y)/K$ and $[L, y]$ if $y - 2f'(y)/K \leq L \leq y$.

3 Appendix: Code

3.1 auxilary.R

```
mprint <- function (x, d = 2, w = 5) {
  print (noquote (formatC (
    x, di = d, wi = w, fo = "f"
  )))
}

cobwebPlotter <-
  function (xold,
            func,
            lowx = 0,
            hghx = 1,
            lowy = lowx,
            hghy = hghx,
            eps = 1e-10,
            itmax = 25,
            ...) {
    x <- seq (lowx, hghx, length = 100)
    y <- sapply (x, function (x)
      func (x, ...))
    plot (
      x,
      y,
      xlim = c(lowx , hghx),
      ylim = c(lowy, hghy),
      type = "l",
      col = "RED",
      lwd = 2
    )
    abline (0, 1, col = "BLUE")
    base <- 0
    itel <- 1
    repeat {
      xnew <- func (xold, ...)
      if (itel > 1) {
        lines (matrix(c(xold, xold, base, xnew), 2, 2))
      }
      lines (matrix(c(xold, xnew, xnew, xnew), 2, 2))
      if ((abs (xnew - xold) < eps) || (itel == itmax)) {
        break ()
```

```
      }
      base <- xnew
      xold <- xnew
      itel <- itel + 1
    }
  }

minQuadratic <- function (a, lw, up) {
  f <- polynomial (a)
  fup <- predict (f, up)
  flw <- predict (f, lw)
  if (a[3] <= 0) {
    if (fup <= flw) return (list (x = up, f = fup))
    if (fup >= flw) return (list (x = lw, f = flw))
  }
  xmn <- - a[2] / (2 * a[3])
  fmn <- predict (f, xmn)
  if (xmn >= up) return (list (x = up, f = fup))
  if (xmn <= lw) return (list (x = lw, f = flw))
  return (list (x =  xmn, f = fmn))
}
```

3.2 iterate.R

```
myIterator <-
  function (xinit,
            f,
            eps = 1e-6,
            itmax = 100,
            verbose = FALSE,
            final = TRUE,
            ...) {
    xold <- xinit
    cold <- Inf
    itel <- 1
    repeat {
      xnew <- f (xold, ...)
      cnew <- abs (xnew - xold)
      rate <- cnew / cold
      if (verbose)
        cat(
          "Iteration: ",
```

```
      formatC (itel, width = 3, format = "d"),
      "xold: ",
      formatC (
        xold,
        digits = 8,
        width = 12,
        format = "f"
      ),
      "xnew: ",
      formatC (
        xnew,
        digits = 8,
        width = 12,
        format = "f"
      ),
      "cnew: ",
      formatC (
        cnew,
        digits = 8,
        width = 12,
        format = "f"
      ),
      "rate: ",
      formatC (
        rate,
        digits = 8,
        width = 12,
        format = "f"
      ),
      "\n"
    )
  if ((cnew < eps) || (itel == itmax))
    break
  xold <- xnew
  cold <- cnew
  itel <- itel + 1
}
if (final)
  cat(
    "Iteration: ",
    formatC (itel, width = 3, format = "d"),
    "xinit: ",
    formatC (
      xinit,
```

```
          digits = 8,
          width =  6,
          format = "f"
        ),
        "xfinal: ",
        formatC (
          xnew,
          digits = 8,
          width =  6,
          format = "f"
        ),
        "rate: ",
        formatC (
          rate,
          digits = 8,
          width =  6,
          format = "f"
        ),
        "\n"
      )
    return (list (
      itel = itel,
      xinit = xinit,
      xfinal = xnew,
      change = cnew,
      rate = rate
    ))
  }

cubicUQ <- function (x, a, up, lw, sharp = FALSE) {
  f <- polynomial (a)
  g <- deriv (f)
  h <- deriv (g)
  i <- deriv (h)
  if (!sharp) {
  if (a[4] > 0)
    k <- predict (h, up)
  if (a[4] < 0)
    k <- predict (h, lw)
  }
  if (sharp) {
    if (a[4] > 0)
      k <- predict (h, x) + predict (i, x) * (up - x) / 3
    if (a[4] < 0)
```

```
    k <- predict (h, x) + predict (i, x) * (lw - x) / 3
  }
  xmin <- x - predict (g, x) / k
  if ((xmin <= up) && (xmin >= lw))
    return (xmin)
  fup <- predict (f, up)
  flw <- predict (f, lw)
  return (ifelse (fup < flw, up, lw))
}
```

3.3 sublevel.R

```
tester <- function (y, k, func, grad) {
  qmaj <- function (x) func (y) + grad (y) * (x - y) + .5 * k * (x - y) ^ 2
    ybnd <- y - 2 * grad (y) / k
    up <- max (y, ybnd)
    lw <- min (y, ybnd)
    x <- seq (lw, up, length = 100)
    s <- paste ("K = ", formatC(k, digits = 1, format= "f"))
    plot (x, func (x), col = "RED", lwd = 2, type = "l", ylab = "f,g", main = s)
    lines (x, qmaj (x), col = "BLUE", lwd = 2)
    abline (v = up)
    abline (v = lw)
}

f <- function (x) log (1 + exp (x))
g <- function (x) exp (x) / (1 + exp (x))

a <- function (x) (x ^ 3 - x) / 6
b <- function (x) (3 * x ^ 2 - 1) / 6

cubicSublevel <- function (y, a, sharp = FALSE) {
  f <- polynomial (a)
  g <- deriv (f)
  h <- deriv (g)
  i <- deriv (h)
  dfy <- predict (g, y)
  dgy <- predict (h, y)
  dhy <- predict (i, y)
  disk <- dgy ^ 2 - 4 * (2 / 3) * dhy * dfy
  if (disk <= 0) k <- max (0, dgy)
  else {
```

```
    r <- sort (solve (polynomial (c((2 / 3) * dhy * dfy, -dgy, 1))))
    if ((r[1] >= 0) && (r[2] >= 0)) k <- dgy
    if ((r[1] <= 0) && (r[2] >= 0)) k <- r[2]
    if ((r[1] <= 0) && (r[2] <= 0)) stop ("unbounded")
  }
  return (y - dfy / k)
}
```

References

De Leeuw, J. 2006. "Sharp Local Quadratic Majorization."

De Leeuw, J., and K. Lange. 2009. "Sharp Quadratic Majorization in One Dimension." *Computational Statistics and Data Analysis* 53: 2471–84.

Lange, K. 2016 (in press). *MM Optimization Algorithms*.