Unconstrained Multivariable Optimization

This program (`demomopt`) demonstrates how to interact with a ***Numerit*** function named `mdopt` that solves the *unconstrained, multivariable optimization problem* defined by

$$\min f(x_1, x_2, \ldots, x_n) \qquad \text{where } x_1, x_2, \ldots, x_n \in \Re^n \tag{1}$$

The function derivatives required by this algorithm are estimated numerically using either Ridder's method of polynomial extrapolation or a central divided difference method. This ***Numerit*** function attempts to solve this problem using either a *conjugate gradient* or *quasi-Newton* method. The user can choose the solution method and the type of derivative calculations to use. This numerical method is based on ACM Algorithm #500.

This demonstration program solves the *Generalized Rosenbrock Function* which is given by the following expression

$$f(x_1, \ldots, x_n) = 1 + \sum_{i=2}^{n} \left[100\left(x_i - x_{i-1}^2\right)^2 + \left(1 - x_{i-1}\right)^2 \right] \tag{2}$$

With $n = 10$ and an initial guess for this example given by

$$\left(\frac{1}{11}, \frac{2}{11}, \frac{3}{11}, \ldots, \frac{10}{11}\right)^T$$

the optimal solution of this problem is

$$x^* = (1, 1, 1, \ldots, 1)^T$$

and the optimal function value $f(x^*) = 1$.

The following is the syntax of this ***Numerit*** function:

```
function mdopt(method, gtype, n, eps, maxiter, iflag, niter, f, x)

` multivariable minimization

` numerical gradient method

` input

`  method  = method of solution
`        1 = conjugate gradient
`        2 = quasi-newton
`  gtype   = type of gradient calculation
`        1 = Ridder's method
`        2 = central differences
`  n       = number of variables
```

```
`  eps     = convergence criterion
`  maxiter = maximum number of iterations
`  x       = initial guess for solution vector

` output

`  niter = number of algorithm iterations
`  f     = final objective function value
`  x     = final solution vector
`  iflag = diagnostic flag
`      0 = converged
`      1 = maximum number of function evaluations
`      2 = linear search failure
`      3 = search vector failure
```

This function requires a user-coded function which defines the objective function. The format of this function is

```
function mdofunc (n, x, f)

` user-defined function

` input

`  n = number of variables
`  x = function argument vector

` output

`  f = scalar value of objective function at x
```

where `mdofunc` is the actual name of the user's function and the argument list is as defined here.

The main program must "reroute" the name of the objective function prior to calling `mdopt` function. The ***Numerit*** statement which reroutes the name of the objective function for this example is

```
mdofunc -> mdofunc1
```

The syntax and source code for the user-defined objective function for this example are as follows:

```
function mdofunc1 (n, x, f)

` user-defined function

` input

`  n = number of variables
`  x = function argument vector

` output
```

```
`   f = scalar value of objective function at x

` Numerical Analysis with Numerit

``````````````````````````````````

t1 = 0
t2 = 0

` compute function value

for i = 2 to n
    xi = x[i]
    xim1 = x[i-1]

    t1 = t1 + (xi - xim1^2)^2

    t2 = t2 + (1 - xim1)^2

f = 1 + 100 * t1 + t2
```

The following is a typical draft output created by this software.

```
demomopt - multivariable optimization

solution vector

x[ 1 ] =  0.9999999999993558
x[ 2 ] =  1.000000000002518
x[ 3 ] =  1.000000000007952
x[ 4 ] =  1.000000000006929
x[ 5 ] =  0.9999999999992524
x[ 6 ] =  0.9999999999984744
x[ 7 ] =  1.000000000002153
x[ 8 ] =  1.000000000004311
x[ 9 ] =  0.9999999999922578
x[ 10 ] =  0.9999999999965242

function value =  1
iflag          =  0
eps            =  1e-08
niter          =  75
```

For this example the quasi-Newton technique and Ridder's method for derivative calculations were used. The convergence criterion was 1.0e-8 and the maximum number of iterations was set to 500.
```