Optimization-Based Modeling and Design of Electronic Circuits
Assignment 3

Prof. J. E. Rayas-Sánchez
March 2019

Develop a Matlab function to implement Broyden's method for solving systems of nonlinear equations.

The Matlab function should be terminated when a solution is found, when the relative change in the optimization variables is small enough, or when a maximum number of iterations is reached:

$$\left\| \boldsymbol{f}(\boldsymbol{x}_{i+1}) \right\|_\infty < \varepsilon_\mathrm{f} \;\vee\; \| \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \|_2 < \varepsilon_\mathrm{x}(\| \boldsymbol{x}_i \|_2 + \varepsilon_\mathrm{x}) \;\vee\; i > i_{\max}$$

where $\boldsymbol{x}_i \in \Re^n$ is the vector containing the n optimization variables at the i-th iteration, and $\boldsymbol{f}: \Re^n \rightarrow \Re^n$ is the system of nonlinear equations. Scalar limits are defined by ε_f, ε_x and $i_{\max}$.

The implemented algorithms should have the following functionality:

```
% Usage: [x,f,i,FunEval,EF] = Broyden(fun,x0,MaxIter,epsf,epsx)
%        fun: name of the multidimensional vector function (string). This
%             function takes a vector argument of length n and returns a
%             row vector of length n.
%        x0: starting point (vector of length n).
%        MaxIter: maximum number of iterations to find a solution.
%        epsf: maximum acceptable error in the root of the function.
%        epsx: minimum relative change in the optimization variables x.
%        x: solution found for the nonlinear system of equations (vector
%        of length n).
%        f: function value at the solution (vector of length n).
%        i: number of iterations needed to find the solution.
%        FunEval: Number of function evaluations needed.
%        EF: exit flag,
%        EF=1: successful optimization (a root was found within epsf).
%        EF=2: algorithm converged (relative change in x is small enough).
%        EF=-1: maximum number of iterations exceeded.
```

Using ε_f = `epsf` $= 10^{-5}$, ε_x = `epsx` $= 10^{-6}$ and $i_{\max}$ = `MaxIter` $= 1000$, test your algorithm with at least the following systems of nonlinear equations and starting points:

Name	$\boldsymbol{f}(\boldsymbol{x}) =$	$\boldsymbol{x}^*$	$\boldsymbol{x}_0^\mathrm{T}$
`SysNL_y2x2_a`	$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_1^2 - 4 \\ x_2^3 - 8 \end{bmatrix}$	$\boldsymbol{x}_a^* = [2 \quad 2]^\mathrm{T}$ $\boldsymbol{x}_b^* = [-2 \quad 2]^\mathrm{T}$	[1 1], [-1 -1], [5 10], [-50 100], [1000 5000]
`SysNL_y3x3_a`	$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_1 \\ x_2^2 + x_2 \\ e^{x_3} - 1 \end{bmatrix}$	$\boldsymbol{x}_a^* = [0 \quad 0 \quad 0]^\mathrm{T}$ $\boldsymbol{x}_b^* = [0 \quad -1 \quad 0]^\mathrm{T}$	[1 1 1], [1 0 -1], [-2 2 -2], [-3 0.1 2], [10 15 20]
`SysNL_y2x2_b`	$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_1^2 + x_2^2 - 2 \\ e^{(x_1 - 1)} + x_2^3 - 2 \end{bmatrix}$	$\boldsymbol{x}_a^* = [1 \quad 1]^\mathrm{T}$ $\boldsymbol{x}_b^* = [-0.7137483 \quad 1.22088731]^\mathrm{T}$	[0 0], [2 0.5], [-0.5 1.5], [1.5 -1.5], [-2 -2]
`SysNL_y1x1_b`	$\boldsymbol{f}(\boldsymbol{x}) = \left[(x_1 - 1)^2 + 2\right]$	$\boldsymbol{x}^* = 1$	0, 1.5, -0.5, 2, 0.9

Periférico Sur 8585
45604 Tlaquepaque, Jal., México
Tel +52 33 3669 3598
www.iteso.mx

SysL_y3x3_a	$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ $\boldsymbol{A} = \begin{bmatrix} 1 & 0.1 & -0.1 \\ -0.1 & 1.1 & 0.1 \\ 0.2 & -0.1 & 0.8 \end{bmatrix}$ $\boldsymbol{b} = \begin{bmatrix} -0.19 \\ 0.133 \\ -0.125 \end{bmatrix}$	$\boldsymbol{x}^* = [0.21 \quad -0.11 \quad 0.09]^{\mathrm{T}}$	[0 0 0], [2 0.5 -1.2], [-1.5 1.5 0.2], [-1 -1 -1], [1 1 1]

For each objective function and starting point $\boldsymbol{x}_0$, you must report the solution found $\boldsymbol{x}$, the number of iterations needed (`Iter`), the number of function evaluations needed (`FunEval`), the exit flag value (`EF`), and the Euclidean norm of the error between the solution found and the corresponding exact solution. Summarize your results in one table per system of equations, with the following structure:

$\boldsymbol{x}_0^{\mathrm{T}}$	$\boldsymbol{x}^{\mathrm{T}}$	`Iter`	`FunEval`	`EF`	$\lVert \boldsymbol{x}^* - \boldsymbol{x} \rVert_2$

Solve the same problems using the Matlab command `fminsearch` on the same test cases (with the same starting points), adjusting the objective function accordingly. Make a comparison with your previous results.

Indicate the reference(s) used to write your algorithm.

Include in your report the conclusions.

Optional: Implement a globally convergent Newton method for solving systems of nonlinear equations, and test your implementation using the same cases described above, making a comparison with your previous results.

Submission deadline: March 20, 2019.

Periférico Sur 8585
45604 Tlaquepaque, Jal., México
Tel +52 33 3669 3598
www.iteso.mx