Problem set 8

[Revised Feb 04, 9:31 PM]

Nonlinear problems

1. You should complete the code in `bisection.py` to solve the following nonlinear problem

   $$3x^3 = -2x^2 + 5x + 20$$

   for x using the bisection method.

   `bisection.py` contains two functions `residual(x)` and `solve_bisection(x_min, x_max, func)`.

   - Write down (on paper) the nonlinear residual $F(x)$ for this problem
   - Modify `bisection.py` and insert the definition of the nonlinear residual within the function `residual(x)`

   The following parts require you to edit `solve_bisection()`.

   - Add the bisection criteria to shrink the search interval depending on whether $F(x_{\text{left}})F(x^*) > 0$ or $F(x_{\text{left}})F(x^*) < 0$.
   - Modify `solve_bisection()` to include the following stopping conditions at iteration k

     - Stop if the absolute residual

       $$\mid F(x_k) \mid$$

       is $< 10^{-12}$
     - Stop if the solution x is not improving between iterations. We do this by checking the relative error

       $$\frac{|x_k - x_{k-1}|}{|x_k|}.$$

Change `solve_bisection()` to stop iterating if the condition

$$| x_k - x_{k-1} | < 10^{-12} | x_k |$$

is met.

- For all stopping conditions, use `break` to exit the bisection iteration loop, and report (using `print()`) which stopping condition was satisified and how many iterations were performed.

Finally, run `bisection.py` using the starting interval $[1.0, 2.0]$ and report: (i) how many iterations were required to obtain a solution to $F(x) = 0$; (ii) the final solution x obtained.

2. We will solve the solve problem as in question 1 using Newton's method. You will edit functions within `newton.py` for this question.

   - The residual $F(x)$ is the same as in Q1. Add the residial definition within `residual(x)`.
   - Write down (on paper) the Jacobian of $F(x)$.
   - Insert the definition of the Jacobian into the function `jacobian(x)`.
   - Add the same stopping conditions as for Q1 into `solve_newton()`.

   Finally, run `newton.py` using an initial guess `x0 = 8.0` and report: (i) how many iterations were required by Newton's method to obtain a solution to $F(x) = 0$; (ii) the final solution x obtained. Compare your answer with Q1. Discuss differences in the iterations required to converge, reason the iterations were terminated.

3. Write down (on paper) the Jacobaim for the following nonlinear problems

   - $$\begin{aligned} F_1(x_1, x_2) &= x_1 + \exp(x_2) - 2 \\ F_2(x_1, x_2) &= x_2 + x_1^2 \end{aligned}$$
   - $$\begin{aligned} F_1(x_1, x_2) &= x_1 - \tfrac{1}{2} x_1 x_2 - 1 \\ F_2(x_1, x_2) &= x_2 + \tfrac{1}{2} x_1 x_2 - 1 \end{aligned}$$
   -

$$F_1(x_1, x_2) = x_1 - 2\exp(-x_1 x_2)$$
$$F_2(x_1, x_2) = x_2 + \exp(-x_1 x_2) - 1$$

Hint: Use the approach described in class. That is first compute the directional directive of $\mathbf{F}(\mathbf{x}^*)$ in the direction $\delta\mathbf{x}$, then isolate the $\delta\mathbf{x}$ terms so that you have $\mathbf{J}(\mathbf{x}^*)\delta\mathbf{x}$. Lastly you can replace $\mathbf{x}^*$ with $\mathbf{x}$ to yield $\mathbf{J}(\mathbf{x})$. Also recall

$$\mathbf{J}(\mathbf{x}^*)\delta\mathbf{x} = \nabla\mathbf{F}_x(\mathbf{x}^*)\delta\mathbf{x}$$

can be computed with the following threes steps
- Insert $\mathbf{x}^* + \epsilon\delta\mathbf{x}$ into the nonlinear residual, i.e. evaluate

$$\mathbf{F}(\mathbf{x}^* + \epsilon\delta\mathbf{x}).$$

- Then evaluate

$$\frac{d}{d\epsilon}\left[\mathbf{F}(\mathbf{x}^* + \epsilon\delta\mathbf{x})\right].$$

- Lastly evaluate

$$\frac{d}{d\epsilon}\left[\mathbf{F}(\mathbf{x}^* + \epsilon\delta\mathbf{x})\right]\Big|_{\epsilon=0}.$$

4. We will use Newton's method to solve the following system of nonlinear equations

$$\exp(x_2 - x_1) = 2.0$$
$$x_1 x_2 = -x_3$$
$$x_2 x_3 = -x_1^2 + x_2$$

for $\mathbf{x} = (x_1, x_2, x_3)$.

A skeleton code is provided in `newton_system.py`.

- Write down (on paper) the nonlinear residual $\mathbf{F}(\mathbf{x})$
- Insert the definition of the residual in `residual_sys(x)`
- Write down (on paper) the Jacobian of $\mathbf{F}(\mathbf{x})$.
- Insert the definition of the Jacobian in `jacobian_sys(x)`
- We will stop the Newton iterations when the 2-norm of the nonlinear residual F_n

$$F_n = \| \mathbf{F} \|_2$$

is below some tolerance. Modify `solve_newton_sys()` to stop the iterations when

$$F_n < 10^{-12}$$

Finally, run `newton_system.py` with the initial guess $\mathbf{x}_0 = (0, 0, 0)$ and report: (i) how many iterations were required by Newton's method to obtain a solution to $\mathbf{F}(\mathbf{x}) = \mathbf{0}$; (ii) the solution $\mathbf{x}$ you obtained.