Out: March 2, 2020
Due: March 15, 2020

NTNU – Trondheim
Norwegian University of
Science and Technology

TTK4130 Modeling and Simulation
Assignment 8

Introduction

Newton's method is one of the most important and powerful root-finding methods. Under the correct circumstances (see "Fourier conditions"), the convergence of this method is quadratic.

In this course, we will use Newton's method to solve the implicit equations that arise from the implementation of implicit Runge-Kutta schemes (IRK), and from the simulation of implicit ODEs and DAEs.

Before we dwell into IRK schemes and implicit differential equations, we will implement and test our own Newton's method. Under the testing, we will be specially interested in the situations where Newton's method fails. Finally, we will use the developed Newton's method to simulate models using the implicit Euler scheme.

Newton's method and their most relevant aspects for this course are explained in the additional lecture notes, which can be found on Blackboard.

Problem 1 (Newton's method)

(a) Implement Newton's method as a Matlab function. This function should have as arguments the objective function $\boldsymbol{f}$, whose root we want to find, its Jacobian $\boldsymbol{J}$ and an initial estimate $\boldsymbol{x}_0$. You are free to add arguments for the tolerance and the maximum number of iterations, or to hardcode some values for these parameters. The Matlab function should return at least the final root estimate.

Hint: A template for such a Matlab function can be found on Blackboard.

(b) Use your implemented Newton's method to solve the equation

$$xy = 2 \tag{1a}$$

$$\frac{x^4}{4} + \frac{y^3}{3} = 1. \tag{1b}$$

Use the initial estimate $x_0 = y_0 = -1$.

Add a semilogarithmic plot of the infinity norm of the residuals ($||\boldsymbol{f}(\boldsymbol{x}_k)||_\infty$) to your answer.

Display the iteration values, and comment on the results.

(c) Test your implemented Newton's method on the function

$$f(x) = (x-1)(x-2)(x-3) + 1 = x^3 - 6x^2 + 11x - 5. \tag{2}$$

Use the initial estimate $x_0 = 3$.

Add a plot of the obtained results to your answer.

Explain the reason behind these results.

How you would modify Newton's method to improve the root estimates?

Hint: What happens if one iteration is near a singular point?

(d) Test your implemented Newton's method on the function

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_1 - 1 + (\cos(x_2)x_1 + 1)\cos(x_2) \\ -x_1 \sin(x_2)(\cos(x_2)x_1 + 1) \end{bmatrix}. \tag{3}$$

Use the initial estimate $\boldsymbol{x}_0 = [1,\, 3]^\top$.

Add a semilogarithmic plot of the infinity norm of the residuals ($||\boldsymbol{f}(\boldsymbol{x}_k)||_\infty$) to your answer.

Display the iteration values, and explain the reason behind the obtained results.

How you would modify Newton's method to improve the root estimates?

Hint: The closest root to $\boldsymbol{x}_0$ is $[1,\ \pi]^\top$. What is the Jacobian at this root?

Since a necessary condition for an extremum of a scalar function is that the gradient is zero at that point, Newton's method can also be used to find maxima and minima of a scalar function by finding the roots of the gradient.

(e) Use your implemented Newton's method to find the minimum of the function

$$f(\boldsymbol{x}) = 100(x_2 - x_1)^2 + (x_1 - 1)^4. \tag{4}$$

Use the initial estimate $\boldsymbol{x}_0 = [10,\ 10]^\top$.

What is the converge order of the iterations?

Explain the reason behind the obtained results.

Problem 2 (Implicit Euler method)

(a) Based on the root-finding method developed in the previous problem, implement the implicit Euler method for a generic vector field $\boldsymbol{f}(t, \boldsymbol{x})$ (i.e. $\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x})$) as a Matlab function.

Explain what implicit equation has to be solved at each iteration of this IRK.

Add the implemented code to your answer.

Hint: A template for such a Matlab function can be found on Blackboard.

(b) Test the implemented integrator on the classic test system:

$$\dot{x} = \lambda x\,, \tag{5}$$

where $\lambda = -2$ and $x(0) = 1$.

Simulate until a final time $t_f = 2$ with a time-step $\Delta t = 0.2$.

Compare the results to the actual solution, and add a plot to your answer.