# Numerical Programming 2 (CSE) 2015

Worksheet 8

## Exercise 1 (BDF(2) method)

The BDF(s) methods for solving $\dot{y}(t) = f(t, y)$ are given by

$$y_{n+s} = q(t_{n+s}),$$

where $q \in \mathbb{P}_s$ is the polynomial implicitly defined by

$$q(t_{n+k}) = y_{n+k}, \; k = 0, \dots, s-1 \qquad q'(t_{n+s}) = f(t_{n+s}, y_{n+s}).$$

- Find a system of linear equations which can be used to determine the coefficients $a_0^{(n)}, a_1^{(n)}, b_2^{(n)}$ in the representation

$$\sum_{k=0}^{2} a_k^{(n)} y_{n+k} = b_s^{(n)} f(t_{n+2}, y_{n+2})$$

  of the $n$-th step of the BDF(2) method for *arbitrary* grids $t_0, t_1, \dots$.

- Verify that the BDF(2) method on an *uniform* grid is given by

$$y_{n+2} - \frac{4}{3} y_{n+1} + \frac{1}{3} y_n = \frac{2}{3} h f(t_{n+2}, y_{n+2}).$$

- Show that the BDF(2) method on uniform grids is A(0)-stable[1], i.e. show that both roots $w_j(z)$, $j = 1, 2$, of the stability polynomial

$$w \mapsto \sum_{k=0}^{2} (a_k - z b_k) w^k$$

  fulfill $|w_j(z)| \leq 1$ if $z \in \mathbb{R}$, $z \leq 0$.

## Exercise 2 (Implementation with adaptive step size)

Implement the following algorithm for solving autonomous ODEs $\dot{y} = f(y)$, which is similar to the one presented in the lecture, but always uses the BDF(2) method.

We are at time $t_j$ (corresponding to step $n = j - 1$ in Exercise 1) with predicted step size $h_j$ (i.e. $t_{j+1} = t_j + h_j$).

[1] It is even A-stable, but this is more difficult to prove.

1. Compute the predictor polynomial $p \in \mathbb{P}_2$ which fulfills $p(t_k) = y_k$ for $k = j-2, j-1, j$ and evaluate $y^0_{j+1} := p(t_{j+1})$ to obtain the predictor value.

2. Determine the coefficients $a_0^{(j-1)}, a_1^{(j-1)}, b_2^{(j-1)}$ of the BDF(2) method using your results from Exercise 1 ($a_2^{(j-1)} = 1$).

3. Perform simplified Newton iterations started in $y^0_{j+1}$ to obtain $y_{j+1}$ as the solution of

$$F(y_{j+1}) := \sum_{k=0}^{2} a_k^{(j-1)} y_{j-1+k} - b_s^{(j-1)} f(y_{j+1}) \stackrel{!}{=} 0.$$

   As an additional safeguard, check whether the norms of the Newton corrections[2] decrease and restart the step with reduced step size $h_j \leftarrow h_j/10$ if they do not.

4. Compute the local error estimate

$$\epsilon_{j+1} = \frac{1}{t_{j+1} - t_{j-2}} DF(y^0_{j+1})^{-1}(y^0_{j+1} - y_{j+1}).$$

5. Evaluate the predicted step size

$$h_{j+1} = \sqrt[3]{\frac{TOL/10}{\|\epsilon_{j+1}\|}} h_j.$$

6. If $\epsilon_{j+1} < TOL$, continue with the next step. Otherwise restart the current step with $h_j \leftarrow h_{j+1}$.

Use this algorithm to solve the two-dimensional ODE

$$\dot{y} = \begin{pmatrix} -100y_1^3 + y_2 \\ -y_1 \end{pmatrix}, \; y(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \; t \in [0, 20]$$

from Worksheet 7 with $TOL = 10^{-3}$ and initial step size $h = 10^{-2}$. Create a plot showing the step sizes chosen by the algorithm.

As this is a multistep method, you will need initial values also for $y_1, y_2$. You can either obtain these using a single-step method or use the following:

$$t_1 = 0.01, \; y_1 = \begin{pmatrix} 0.009999 \\ 0.999950 \end{pmatrix}, \; t_2 = 0.02, \; y_2 = \begin{pmatrix} 0.019994 \\ 0.999800 \end{pmatrix}$$

[2] In the simplified Newton iteration $x_{k+1} = x_k - DF(x_0)^{-1}F(x_k)$ for solving $F(x) = 0$ the terms $\Delta x_k := -DF(x_0)^{-1}F(x_k)$ are called the *Newton corrections*.