

# I Boundary value problems

A second-order **boundary value problem** has the form

$$\partial^2 y(t) + \partial y(t) = f(t, y), \quad t \in [a, b] \subset \mathbb{R}, \quad y(a) = \alpha, y(b) = \beta. \text{(BVP)} \quad (\text{I.1})$$

**A.I.1– Example:** Deflecting beam.

- 

$$\partial^2 y + y = 0, \quad y(0) = y(\pi) = 0$$

unique solution  $y = \sin$ .

- 

$$\partial^2 y + ay = 0, \quad y(0) = y(\pi) = 0$$

no solution for  $a < 0$ .

- 

$$\partial^2 y + y = 0, \quad y(0) = y(2\pi), \partial y(0) = \partial y(2\pi)$$

has two solutions  $y_1 = \sin, y_2 = \cos$ .

## a Numerical Solution

**A.I.2– Example:** (also see de Boor’s Spline Toolbox)

We want to approximately solve the following *nonlinear, stiff boundary value problem* in one variable:

$$\begin{aligned} \epsilon \partial^2 g + g^2 &= 1 \quad \text{on } [0, 1] \\ \partial g(0) &= g(1) = 0. \end{aligned}$$

Picture the solution for smaller  $\epsilon$ !

To represent the solution we choose degree 3  $C^1$  splines on  $n = 4$  equal-length pieces. We enforce *collocation* at two Gauss points per piece, and use Newton iteration with starting guess  $y^{[0]}(t) = t^2 - 1$ .

The *input* consists of

(1) the differential equation:  $\epsilon \partial^2 g + g^2 = 1$

(2) the domain:  $[0, 1]$

(3) the boundary conditions:  $\partial g(0) = g(1) = 0$

The *algorithm* specifies

(4) the space of approximating functions:  $C^1$  splines of degree 3 on  $n = 4$  equal-length pieces

(5) the approximation criterion: collocation at two Gauss points per piece

(6) the solution technique: Newton iteration with starting guess  $y^{[0]}$

We need to work out the details of the algorithm by determining the knot vector  $x$ , the vector of collocation points  $t$  and the initial choice of spline coefficients  $\mathbf{a}^{[0]}$ . (A degree 3 cubic spline has double knots and we need a total of 10 B-splines, hence 14 knots. Draw it!) In Matlab notation

```

h = 1/n
breakpts = [0 : h : 1]
x = sort([0 0 breakpts breakpts 1 1])
mid = breakpts(1 : n) + h/2
gamma = 1/sqrt(2)
t = (t_j)_{j=1..(2n)} = sort(mid - gamma/(2n), mid + gamma/(2n))
a^{[0]} = splcoeff(y^{[0]})

```

Then we derive a *discretization* of the problem:

$$\begin{aligned}
 \partial y(0) &= 0 \\
 y^2(t_j) + \epsilon \partial^2 y(t_j) &= 1 \quad j = 1..(2n) \\
 y(1) &= 0
 \end{aligned}$$

The *linearization* of the problem for Newton's method is for  $j = 1..(2n)$

$$\begin{aligned} \partial y^{[i+1]}(0) &= 0 \\ 2y^{[i]}(t_j)y^{[i+1]}(t_j) + \epsilon \partial^2 y^{[i+1]}(t_j) &= (y^{[i]}(t_j))^2 + 1 \\ y^{[i+1]}(1) &= 0 \end{aligned}$$

This is a 10 by 10 system of equations. We bring it into matrix form. At iteration  $i$

$$b^{[i]} := \begin{bmatrix} 0 \\ (y^{[i]}(t_1))^2 + 1 \\ \vdots \\ (y^{[i]}(t_{2n}))^2 + 1 \\ 0 \end{bmatrix}, \quad W^{[i]} := \begin{bmatrix} 0 & 1 & 0 \\ 2y^{[i]}(t_1) & 0 & \epsilon \\ \vdots & \vdots & \vdots \\ 2y^{[i]}(t_{2n}) & 0 & \epsilon \\ 1 & 0 & 0 \end{bmatrix}$$

Now we still need  $2n+2$  matrices  $M^{[i]}(x, t_j)$  of size 3 by  $(2n+2)$  that determine  $y^{[i]}(t_j)$ ,  $Dy^{[i]}(t_j)$ ,  $D^2y^{[i]}(t_j)$  from the coefficients  $a_k^{[i]}$ ,  $k = 1..10$ . We write the application of each  $M^{[i]}(x, t_j)$  to row  $j$  of  $W^{[i]}$  as  $W^{[i]}M^{[i]}$ , so  $W^{[i]}M^{[i]}$  is a  $(2n+2)$  by  $(2n+2)$  matrix. Then each Newton step solves

$$(W^{[i]}M^{[i]})\mathbf{a}^{[i+1]} = b^{[i]}$$

and the iteration stops when  $\|\mathbf{a}^{[i+1]} - \mathbf{a}^{[i]}\|_\infty$  is sufficiently small.

Starting with  $\epsilon = 0.1$  a decreased  $\epsilon$  requires additional control in the region close to 1.