Practice Problems Exam

Tiago Salvador (tiago.saldanhasalvador@mail.mcgill.ca)

1. A natural cubic spline for a function f is defined on $[-1, 1]$ by

$$S(x) = \begin{cases} S_0(x) = -4 + x + x^3, & x \in [0, 1), \\ S_1(x) = -2 + 4(x-1) + 3(x-1)^2 - (x-1)^3, & x \in [1, 2]. \end{cases}$$

Compute the Lagrange interpolation polynomial of f on the nodes $x_0 = 0$, $x_1 = 1$, $x_2 = 2$.

Solution: Since S is a cubic spline, $S(x_i) = f(x_i)$. Therefore $f(0) = -4$, $f(1) = -2$, $f(2) = 4$.
We compute the divided differences using the table.

x_i	$f[x_i]$	$f[x_i, x_{i+1}]$	$f[x_i, x_{i+1}, x_{i+2}]$
0	-4		
		2	
1	-2		2
		6	
2	4		

Taking the upper diagonal entries, Newton's divided formula simplifies to

$$L_2(x) = -4 + 2x + 2x(x-1) = -4 + 2x^2.$$

2. (Final Exam Math 317 Fall 2010) Say we have an analytical function $f(x)$ and its numerical approximation $\hat{f}(x)$ such that $\hat{f}(x) = f(x) + e(x)$ where $e(x)$ is then the round-off error and both the function and the error are sufficiently smooth for our purposes. Now we approximate the derivative with a finite difference scheme. Let's use

$$\hat{D}(x) = \frac{\hat{f}(x+h) - \hat{f}(x-h)}{2h}.$$

Show that the final error is the sum of round-off and truncation errors and that one becomes large at small h, while the other becomes large at large h. *Hint: consider the total error $f'(x) - \hat{D}(x)$ and recall that*

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{f^{(3)}(\xi)}{6}h^2.$$

Solution: We have

$$\begin{aligned} f'(x) &= \frac{f(x+h) - f(x-h)}{2h} - \frac{f^{(3)}(\xi)}{6}h^2 \\ &= \frac{\hat{f}(x+h) - e(x+h) - \hat{f}(x-h) + e(x-h)}{2h} - \frac{f^{(3)}(\xi)}{6}h^2 \\ &= \frac{\hat{f}(x+h) - \hat{f}(x-h)}{2h} + \frac{e(x-h) - e(x+h)}{2h} - \frac{f^{(3)}(\xi)}{6}h^2 \end{aligned}$$

Let E be a bound for the round-off error and M a bound for $|f^{(3)}|$. Then

$$\left|f'(x) - \hat{D}(x)\right| \leq \frac{|E|}{h} + \frac{Mh^2}{6}.$$

The first term in the right-end side of the inequality comes from the round-off error and it gets bigger for smaller and smaller values of h. The second term comes from the truncation error in the Taylor series to obtain the approximation and it gets smaller for smaller and smaller values of h.

3. (Final Exam Math 317 Fall 2004)

   (a) Use the Taylor expansion

   $$f(x^*) = f(x) + (x^* - x)f'(x) + \frac{(x^* - x)^2}{2}f''(\xi)$$

   for $f \in C^2[a, b]$ to derive Newton's method for approximating a root x^* of the equation $f(x) = 0$.

   Solution: The idea is that given a guess x_n we want to compute the new guess such that $f(x_{n+1}) \approx 0$. Hence ignoring the error term in the Taylor expansion and taking $x^* = x_{n+1}$ and $x = x_n$ we get

   $$0 = f(x_n) + (x_{n+1} - x_n)f'(x_n)$$

   and so

   $$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

   which is indeed the Newton's method as desired.

   (b) Show that Newton's method can be written as a fixed point iteration $x_{n+1} = g(x_n)$ for a suitable choice of $g(x)$.

   Solution: We just need to take

   $$g(x) = x - \frac{f(x)}{f'(x)}.$$

   (c) Show that $g'(x^*) = 0$ provided that $f'(x^*) \neq 0$. What can we say about the fixed point iteration in such a case?

   Solution: We have that

   $$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{[f'(x)]^2}.$$

   Hence, since $f(x^*) = 0$ and $f'(x^*) \neq 0$, we get $g'(x^*) = 0$. In this case the fixed point iteration will converge at least quadratically to x^* provided x_0 is close enough to x^*, i.e., if $x_0 \in [x^* - \delta, x^* + \delta]$ for some $\delta > 0$.

   (d) Suppose $f'(x^*) = 0$ and $f''(x^*) \neq 0$. Show that we can write $f(x) = (x - x^*)^2 r(x)$ for some function $r(x)$ with $r(x^*) \neq 0$.

   Solution: Taylor expanding f about $x_0 = x^*$ leads to

   $$\begin{aligned} f(x) &= f(x^*) + f'(x^*)(x - x^*) + \frac{f''(\xi(x))}{2!}(x - x^*)^2 \\ &= \frac{f''(\xi(x))}{2!}(x - x^*)^2 \end{aligned}$$

We take then

$$r(x) = \frac{f''(\xi(x))}{2}$$

Note that $r(x^*) \neq 0$ since $\xi(x^*) = x^*$ and $f''(x^*) \neq 0$.

(e) Find $\lim_{x\to x^*} g'(x)$ for Newton's method when $f'(x^*) = 0$ but $f''(x^*) \neq 0$. What can we say about the fixed point iteration in such case?

Solution: Using the result from the previous question, we can write $f(x) = (x - x^*)^2 r(x)$ with $r(x^*) \neq 0$. Then

$$f'(x) = 2(x - x^*)r(x) + (x - x^*)^2 r'(x) = (x - x^*)\left(2r(x) + (x - x^*)r'(x)\right)$$

and

$$f''(x) = 2r(x) + (x - x^*)k(x)$$

for some function k. Then

$$\begin{aligned}
\lim_{x\to x^*} g'(x) &= \lim_{x\to x^*} \frac{f(x)f''(x)}{[f'(x)]^2} \\
&= \lim_{x\to x^*} \frac{(x - x^*)^2 r(x)\left(2r(x) + (x - x^*)k(x)\right)}{(x - x^*)^2\left(2r(x) + (x - x^*)r'(x)\right)^2} \\
&= \frac{1}{2}.
\end{aligned}$$

In this case the fixed point iteration will converge only linear to x^* provided x_0 is close enough to x^*.

(f) The root $x^* = 5$ of $f(x) = x^3 - 9x^2 + 15x + 25$ is approximated using Newton's method method with $x_0 = 3$. What is the order of convergence?

Solution: We have $f(x) = (x-5)^2(x-1)$. Computing the first two iterations gives $x_1 = \frac{13}{3}$ and $x_2 = \frac{211}{45}$. Hence this fixed point iteration converges indeed to $x^* = 5$ and using (d) we conclude that the order of convergence is 1.

4. Consider the function $f(x) = x^2 - 4x + 3$ and the fixed point iterations $x_{n+1} = g(x_n)$ given by

$$i.\ g(x) = \sqrt{4x - 3}, \qquad ii.\ g(x) = \frac{x^2 + 3}{4}.$$

Analyzing the local convergence of the fixed point iteration, explain which method you would use to approximate each root of f.

Solution: Notice that $f(x) = (x - 1)(x - 3)$ and so the roots are $x^* = 1$ and $x^* = 3$. Recall that a fixed point iteration is local convergent (i.e. it converges if the initial guess is close enough to x^*) if $|g'(x^*)| < 1$. Let $g_1(x) = \sqrt{4x - 3}$ and $g_2(x) = \frac{x^2+3}{4}$.

(a) $x^* = 1$

We have $|g_1'(1)| = 2$ and $|g_2'(1)| = \frac{1}{2}$. Hence to approximate the root $x^* = 1$, we should use g_2 since the fixed point iteration will converge provided we initialize it with a good initial guess.

(b) $x^* = 3$

We have $|g_1'(3)| = \frac{2}{3}$ and $|g_2'(3)| = \frac{3}{2}$. Hence to approximate the root $x^* = 3$, we should use g_1 since the fixed point iteration will converge provided we initialize it with a good initial guess.

5. (Final Exam Math 317 Fall 2010) Suppose we have a computer that can't divide. One way to get around this is to create a routine that finds roots of $f(x) = \frac{1}{x} - a$ where $a \neq 0$. We then input a and get its inverse. (You (human) can use division in formulating the algorithm, but the algorithm itself must not require the computer to divide). Show how you would use Newton's method to do this. Show if/when it converges. What is the order of convergence?.

Solution: We take $f(x) = 1/x - a$. Then $x^* = 1/a$. We know that Newton's method is given by

$$x_{n+1} = x_n - \frac{f^(x_n)}{f'(x_n)}$$

Since $f'(x) = -1/x^2$, we get

$$x_{n+1} = 2x_n - x_n^2 a.$$

Notice that the final formula doesn't use division and so we can use, even though we needed to use division to get it. Our algorithm then consists in the following: given an initial guess x_0, use the fixed point method above to find an approximation for the inverse of a. Multiplying by that approximation will gives an approximation for the division by a. Since $f(x^*) = 0$ and $f'(x^*) = -a^2 \neq 0$, the method will converge quadratically to x^* provide x_0 is close enough to x^*.

6. (Final Exam Math 317 2005)

   (a) What is the key difference between Lagrange and Hermite interpolation? What is the difference between a clamped and a natural cubic spline?

   Solution: The key difference is that in Hermite interpolation we match not only the function value at the nodes (like in the Lagrange interpolation) but also the derivative. In clamped cubic spline we impose the extra conditions $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ (provided f' exists and is given) and in the natural cubic spline the conditions $S''(x_0) = S''(x_n) = 0$.

   (b) A natural cubic spline S on $[0, 2]$ has the formula

$$S(x) = \begin{cases} S_0(x) = 1 + 2x - x^3 & 0 \leq x < 1 \\ S_1(x) = a + b(x-1) + c(x-1)^2 + d(x-1)^3 & 1 \leq x \leq 2. \end{cases}$$

   Find a, b, c and d.

Solution: We have

$$S'(x) = \begin{cases} S_0'(x) = 2 - 3x^2 & 0 \leq x < 1 \\ S_1'(x) = b + 2c(x-1) + 3d(x-1)^2 & 1 \leq x \leq 2. \end{cases}$$

and

$$S''(x) = \begin{cases} S_0''(x) = -6x & 0 \leq x < 1 \\ S_1''(x) = 2c + 6d(x-1) & 1 \leq x \leq 2. \end{cases}$$

We know that $S \in C^2[0, 2]$. Therefore

$$\begin{cases} S_1(1) = S_0(1) \\ S_1'(1) = S_0'(1) \\ S_1''(1) = S_0''(0) \end{cases}$$

which leads to

$$\begin{cases} a = 2 \\ b = -1 \\ c = -3 \end{cases}$$

Since it's a natural cubic spline, $S''(2) = 0$ and so $-6 + 6d = 0 \iff d = 1$.

7. (Final Exam Math 317 Fall 2010) How would you go about calculating quadratic splines)? In other words, let's assume you want to form a piecewise polynomial interpolation using polynomials of degree two. Show how to calculate these for data points x_i and f_i with $i = 0, \ldots, n$.

Solution: We have

$$Q(x) = \begin{cases} Q_0(x) & x \in [x_0, x_1] \\ \vdots & \\ Q_{n-1}(x) & x \in [x_{n-1}, x_n] \end{cases}$$

where the Q_j are quadratic polynomials of the form

$$Q_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2.$$

We have $3n$ unknowns and so we will need to impose $3n$ conditions:

- $Q(x_j) = f(x_j)$ for $j = 0, \ldots, n$ ($(n + 1)$ interpolation conditions).
- $Q_j(x_{j+1}) = Q_{j+1}(x_{j+1})$ for $j = 0, \ldots, n - 2$. ($(n - 1)$ continuity conditions).
- $Q'_j(x_{j+1}) = S'_{j+1}(x_{j+1})$ for $j = 0, \ldots, n - 2$ ($(n - 1)$ smoothness conditions / continuity of the first derivative).

We have a total of $3n - 1$ conditions and so we need to impose an extra boundary condition which can be one of the following:

- $Q'(x_0) = 0$
- $Q''(x_0) = 0$
- $Q'(x_0) = f'(x_0)$

Alternatively, if $n = 2$, we can impose that the spline is $C^2[x_0, x_2]$ instead of just $C^1[x_0, x_2]$ as we did here (see the next exercise).

8. Let f be defined on $[a, b]$ and let the nodes $a = x_0 < x_1 < x_2 = b$ be given. A quadratic spline interpolation function S is given by

$$S(x) = \begin{cases} S_0(x) = a_0 + b_0(x - x_0) + c_0(x - x_0)^2 & x \in [x_0, x_1] \\ S_1(x) = a_1 + b_1(x - x_1) + c_1(x - x_1)^2 & x \in [x_1, x_2] \end{cases}$$

such that

i) $S(x_i) = f(x_i)$ for $i = 0, 1, 2$

ii) $S \in C^1[x_0, x_2]$

Show that conditions *i)* and *ii)* lead to five equations in the six unknowns a_0, b_0, c_0, a_1, b_1 and c_1. Determine those equations. The problem is to decide what additional condition to impose to make the solution unique. Does the condition $S \in C^2[x_0, x_2]$ lead to a meaningful solution?

Solution: Let's first compute S' and S'':

$$S'(x) = \begin{cases} b_0 + 2c_0(x - x_0) & x \in [x_0, x_1] \\ b_1 + 2c_1(x - x_1) & x \in [x_1, x_2] \end{cases}$$

and

$$S''(x) = \begin{cases} 2c_0 & x \in [x_0, x_1] \\ 2c_1 & x \in [x_1, x_2] \end{cases}$$

Conditions $i)$ and $ii)$ lead to the equations

$$\begin{cases} S(x_0) = f(x_0) \\ S(x_1) = f(x_1) \\ S(x_2) = f(x_2) \\ S_0(x_1) = S_1(x_1) \\ S_0'(x_1) = S_1'(x_1) \end{cases} \iff \begin{cases} a_0 = f(x_0) \\ a_1 = f(x_1) \\ a_1 + b_1 + (x_2 - x_1) + c_1(x_2 - x_1)^2 = f(x_2) \\ a_0 + b_0(x_1 - x_0) + c_0(x_1 - x_0)^2 = a_1 \\ b_0 + 2c_0(x_1 - x_0) = b_1 \end{cases}$$

The condition $S \in C^2[x_0, x_2]$ imposes the additional condition $S_0''(x_1) = S_1''(x_1) \Leftrightarrow c_0 = c_1$. The unknowns a_0 and a_1 are already determined. The 4 remaining equations lead to a linear system with matrix

$$\begin{bmatrix} 0 & 0 & x_2 - x_1 & (x_2 - x_1)^2 \\ x_1 - x_0 & (x_1 - x_0)^2 & 0 & 0 \\ 1 & 2(x_1 - x_0) & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

This matrix is non-singular since the determinant is nonzero

$$\begin{aligned} &1 \begin{vmatrix} 0 & x_2 - x_1 & (x_2 - x_1)^2 \\ x_1 - x_0 & 0 & 0 \\ 1 & -1 & 0 \end{vmatrix} + (-1) \begin{vmatrix} 0 & 0 & x_2 - x_1 \\ x_1 - x_0 & (x_1 - x_0)^2 & 0 \\ 1 & 2(x_1 - x_0) & -1 \end{vmatrix} \\ &= -(x_1 - x_0) \begin{vmatrix} x_2 - x_1 & (x_2 - x_1)^2 \\ -1 & 0 \end{vmatrix} - (x_2 - x_1) \begin{vmatrix} x_1 - x_0 & (x_1 - x_0)^2 \\ 1 & 2(x_1 - x_0) \end{vmatrix} \\ &= -(x_1 - x_0)(x_2 - x_1)^2 - (x_2 - x_1)(x_1 - x_0)^2 \\ &= (x_1 - x_0)(x_2 - x_1)(-x_2 + x_1 - x_1 + x_0) \\ &= (x_1 - x_0)(x_2 - x_1)(x_0 - x_2) \\ &\neq 0 \end{aligned}$$

The condition is then meaningful since such a spline always exist.

9. Determine a quadratic spline Q that interpolates the data $f(0) = 0$, $f(1) = 1$, $f(2) = 2$ and satisfies $Q'(0) = 2$.

Solution: We are given three points $x_0 = 0$, $x_1 = 1$ and $x_2 = 2$ and so $n = 2$. Therefore there will be two pieces Q_0 and Q_1 to the spline. The goal is then to determine a_0, b_0, c_0, a_1, b_1 and c_1 where

$$Q_0(x) = a_0 + b_0(x - x_0) + c_0(x - x_0)^2$$

and

$$Q_1(x) = a_1 + b_1(x - x_1) + c_1(x - x_1)^2.$$

This is done by imposing the eight conditions from the definition of free cubic spline.

- Three interpolation conditions: $Q_0(x_0) = f(x_0)$, $Q_1(x_1) = f(x_1)$ and $Q_1(x_2) = f(x_2)$.

This leads to

$$\begin{cases} a_0 = f(x_0) = 0 \\ a_1 = f(x_1) = 1 \\ 2 = Q_1(x_2) = a_1 + b_1(x_2 - x_1) + c_1(x_2 - x_1)^2 = 1 + b_1 + c_1 \end{cases}$$

- One continuity condition: $Q_0(x_1) = Q_1(x_1)$.

Thus

$$1 = Q_1(x_1) = Q_0(x_1) = a_0 + b_0(x_1 - x_0) + c_1(x_1 - x_0)^2 = b_0 + c_0.$$

- One smoothness condition: $Q_0'(x_1) = Q_1'(x_1)$

We then get $b_0 + 2c_0 = b_1$

- Extra boundary conditions: $Q'(0) = 2$.

We then have $Q'(0) = b_0 = 2$.
Let us summarize all the equations.

$$\begin{cases} a_0 = 0 \\ a_1 = 1 \\ 1 + b_1 + c_1 = 2 \\ b_0 + c_0 = 1 \\ b_0 + 2c_0 = b_1 \\ b_0 = 2 \end{cases} \iff \begin{cases} a_0 = 0 \\ a_1 = 1 \\ b_0 = 2 \\ b_1 = 1 \\ c_0 = -1 \\ c_1 = 0 \end{cases}$$

The cubic spline is then given by

$$Q(x) = \begin{cases} Q_0(x) = 2x - x^2 & \text{if } x \in [0, 1) \\ Q_1(x) = 1 + (x - 1)^2 & \text{if } x \in [1, 2] \end{cases}$$

10. (a) Define the degree of accuracy (also known as the degree of precision) of a quadrature formula $I_h(f)$ for approximating the integral

$$I(f) = \int_a^b f(x)\, dx.$$

Solution: The degree of accuracy of a quadrature formula $I_h(f)$ for approximating the integral $I(f)$ is the largest positive integer p such that $I(x^i) = I_h(x^i)$ for each $i = 0, \ldots, p$.

(b) Find the degree of accuracy p of the quadrature formula

$$I_h(f) = \frac{3}{2}h\left[f(x_1) + f(x_2)\right]$$

where $a = x_0$, $b = x_3$ and $h = x_{i+1} - x_i$.

Solution: We start by observing that $h = \frac{b-a}{3}$ and $x_i = a + ih = b - (3 - ih)$. Taking $f(x) = 1$, we get

$$I(f) = b - a$$

and

$$I_h(f) = \frac{3}{2}h(1+1) = 3h = b - a.$$

Taking now $f(x) = x$ leads to

$$I(f) = \frac{b^2 - a^2}{2}$$

and

$$I_h(f) = \frac{3}{2}h(a + h + b - h) = \frac{b-a}{2}(b+a) = \frac{b^2 - a^2}{2}.$$

Taking now $f(x) = x^2$ we get

$$I(f) = \frac{b^3 - a^3}{3}$$

and

$$\begin{aligned} I_h(f) &= \frac{3h}{2}\left((a+h)^2 + (b-h)^2\right) \\ &= \frac{b-a}{2}(a + 2ha + h^2 + b^2 - 2hb + h^2) \\ &= \frac{b-a}{18}(5a^2 + 8ab + 5b^2) \\ &\neq \frac{b^3 - a^3}{3}. \end{aligned}$$

Therefore the degree of accuracy is 1.

(c) (Final Exam Math 317 Fall 2005) Find constants α, β and γ such that the degree of accuracy of the quadrature formula

$$I_h(f) = h\left[\alpha f(a) + \beta f(a + \gamma h)\right]$$

is as large as possible, where $h = b - a$.

Solution: Taking $f(x) = 1$ leads to

$$I(f) = b - a$$

and

$$I_h(f) = h(\alpha + \beta) = (b-a)(\alpha + \beta).$$

We then get the equation $\alpha + \beta = 1$. Taking now $f(x) = x$ leads to

$$I(f) = \frac{b^2 - a^2}{2} = \frac{h}{2}(b + a)$$

and

$$I_h(f) = h\left[\alpha a + \beta(a + \gamma h)\right].$$

We now get the equation $\alpha a + \beta(a + \gamma h) = \frac{b+a}{2}$. Finally taking $f(x) = x^2$ we get

$$I(f) = \frac{b^3 - a^3}{3} = \frac{a^2 + ab + b^2}{3}h$$

and

$$I_h(f) = h\left[\alpha a^2 + \beta(a + \gamma h)^2\right].$$

This leads to the equation

$$\alpha a^2 + \beta(a + \gamma h)^2 = \frac{a^2 + ab + b^2}{3}.$$

We then get the nonlinear system

$$\begin{cases} \alpha + \beta = 1 \\ \alpha a + \beta a + \beta\gamma h = \frac{b+a}{2} \\ \alpha a^2 + \beta(a+\gamma h)^2 = \frac{a^2+ab+b^2}{3} \end{cases}$$

which has the solution

$$\begin{cases} \alpha = \frac{1}{4} \\ \beta = \frac{3}{4} \\ \gamma = \frac{2}{3} \end{cases}$$

(d) Determine constants a, b, c, d and e that will produce a quadrature formula

$$\int_{-1}^{1} f(x)\,dx \approx af(-1) + bf(0) + cf(1) + df'(-1) + ef'(1)$$

that has degree of precision at least 4.

Solution: The quadrature needs to be exact for $f(x) = 1, x, x^2, x^3, x^4$. We then obtain the following linear system for a, b, c, d and e:

$$\begin{cases} 2 = a + b + c \\ 0 = -a + c + d + e \\ \frac{2}{3} = a + c - 2d + 2e \\ 0 = -a + c + 3d + 3e \\ \frac{2}{5} = a + c - 4d + 4e \end{cases}$$

The weights a, b, c, d and e are then given by

$$\begin{cases} a = \frac{7}{15} \\ b = \frac{16}{15} \\ c = \frac{7}{15} \\ d = -\frac{1}{15} \\ e = -\frac{1}{15} \end{cases}$$

11. (Final Exam Math 317 Fall 2004) Consider the initial value problem

$$\begin{cases} y'(t) = f(t, y(t)) = \lambda y(t), \quad 0 \le t \le T, \\ y(0) = y_0 > 0, \end{cases}$$

where $\lambda < 0$. Suppose you approximate the solution $y(\cdot)$ using the Runge-Kutta method

$$y_{n+1} = y_n + \frac{1}{4}hf(t_n, y_n) + \frac{3}{4}hf\left(t_n + \frac{2}{3}h, y_n + \frac{2}{3}hf(t_n, y_n)\right).$$

(a) Show that $y(t_{n+1}) = e^{h\lambda}y(t_n)$.

Solution: The solution is given by $y(t) = y_0e^{\lambda t}$. Since $t_n = nh$, we have

$$\frac{y(t_{n+1})}{y(t_n)} = \frac{y_0e^{\lambda t_{n+1}}}{y_0e^{\lambda t_n}} = e^{\lambda(t_{n+1}-t_n)} = e^{\lambda h}$$

and so we are done.

(b) Show that $y_{n+1} = \left(1 + h\lambda + \frac{(h\lambda)^2}{2}\right) y_n$.

Solution: We have

$$\begin{aligned} y_{n+1} &= y_n + \frac{h}{4}\lambda y_n + \frac{3}{4}h\lambda\left(y_n + \frac{2}{3}h\lambda y_n\right) \\ &= y_n + h\lambda y_n + \frac{(h\lambda)^2}{2} y_n \end{aligned}$$

as desired.

(c) Under what conditions on h does $\lim_{n\to\infty} y_n = 0$?

Solution: We need $\left|1 + h\lambda + \frac{(h\lambda)^2}{2}\right| < 1$. First notice that

$$1 + h\lambda + \frac{(h\lambda)^2}{2} = \frac{1}{2}(h\lambda + 1)^2 + \frac{1}{2} \geq \frac{1}{2} \geq 0$$

Hence

$$\begin{aligned} \left|1 + h\lambda + \frac{(h\lambda)^2}{2}\right| < 1 &\iff 1 + h\lambda + \frac{(h\lambda)^2}{2} < 1 \\ &\iff h\lambda\left(1 + \frac{h\lambda}{2}\right) < 0 \\ &\iff 1 + \frac{h\lambda}{2} < 0 \quad \text{since } h\lambda < 0 \\ &\iff h < -\frac{2}{\lambda}. \end{aligned}$$

We need $h < -\frac{2}{\lambda}$.

(d) Define local truncation error $\tau_h(t_n)$ and show that for this problem

$$\tau_h(t_n) = \frac{h^3\lambda^3}{6} y(\xi),$$

where $\xi \in (t_n, t_{n+1})$.

Solution: The local truncation error is given by

$$\tau_h(t_n) = y(t_{n+1}) - \left(1 + h\lambda + \frac{(h\lambda)^2}{2}\right) y(t_n).$$

Using the Taylor expansion we have

$$\begin{aligned} y(t_{n+1}) &= y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \frac{h^3}{6}y'''(\xi) \\ &= y(t_n) + h\lambda y(t_n) + \frac{(h\lambda)^2}{2}y(t_n) + \frac{(h\lambda)^3}{3}y(\xi) \end{aligned}$$

where $\xi \in (t_n, t_{n+1})$. Hence

$$\tau_{n+1}(h) = \frac{h^3\lambda^3}{6} y(\xi).$$

12. Show that the Newton-Cotes quadrature with $n + 1$ points has degree of accuracy at least n.

Solution: The Newton-Cotes quadrature with $n+1$ points is obtained by approximating the integral by the integral of the Lagrange interpolation polynomial over $n+1$ equally spaced. We then have

$$I(f) = \int_a^b f(x)\,dx \approx I_h(f) = \int_a^b L_n(x).$$

The error of the Lagrange interpolation is given by

$$f(x) = L_n(x) + \frac{f^(n+1)(\xi(x))}{(n+1)!}(x-x_0)\dots(x-x_n).$$

Hence we will have $I(f) = I_h(f)$ when $E_n(f) = 0$ where

$$E_n(f) = \int_a^b \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)\dots(x-x_n)\,dx$$

For $i = 0, \dots, n$, $E_n(x^i) = 0$ since the $(n+1)$-th derivative of x^i is identically 0.

13. The k-step Adam-Bashforth method is given by

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} p(t)\,dt$$

where $p(t)$ is the polynomial of degree $k-1$ such that $p(t_{n-i}) = f_{n-i}$, for $i = 0, \dots, k-1$.

(a) Derive the 2-step Adam-Bashforth method.

Solution: We have $k = 2$ and so

$$p(t) = f_n + \frac{f_n - f_{n-1}}{h}(t - t_n)$$

Hence

$$\int_{t_n}^{t_{n+1}} p(t)\,dt = f_n h + \frac{f_n - f_{n-1}}{h}\frac{h^2}{2} = \frac{h}{2}\left(3f_n - f_{n-1}\right)$$

and so the 2-step Adam-Bashforth method which is given by

$$y_{n+1} = y_n + \frac{h}{2}\left(3f_n - f_{n-1}\right).$$

(b) Using the Taylor expansion, show that the local truncation error of the 2-step Adam-Bashforth method is $\mathcal{O}(h^3)$

Solution: The local truncation error is given by

$$\tau_h(t_n) = y(t_{n+1}) - y(t_n) - \frac{3}{2}hf(t_n, y(t_n)) + \frac{1}{2}hf(t_{n-1}, y(t_{n-1}))$$

for $n = 1, 2 \dots, N-1$. The idea is to Taylor expand every term in the above expression which is not evaluated at $t = t_n$ and use the fact that $y(t)$ satisfies $y'(t) = f(t, y(t))$ (since it is the solution of the I.V.P.). We have

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \mathcal{O}(h^3), \quad f(t_n, y(t_n)) = y'(t_n)$$

and

$$f(t_{n-1}, y(t_{n-1})) = y'(t_{n-1}) = y'(t_n) - hy''(t_n) + \mathcal{O}(h^2)$$

Hence,

$$\begin{aligned}\tau_h(t_n) &= y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \mathcal{O}(h^2) - y(t_n) - \frac{3}{2}hy'(t_n) + \frac{1}{2}h\left(y'(t_n) - hy''(t_n) + \mathcal{O}(h^2)\right) \\ &= y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \mathcal{O}(h^2) - y(t_n) - \frac{3}{2}hy'(t_n) + \frac{1}{2}hy'(t_n) - h^2y''(t_n) + \mathcal{O}(h^3) \\ &= \mathcal{O}(h^3)\end{aligned}$$

14. The k-step Adam-Moulton method is given by

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} p(t)\, dt$$

where $p(t)$ is the polynomial of degree k such that $p(t_{n-i}) = f_{n-i}$, for $i = -1, \ldots, k-1$.

(a) Derive the 2-step Adam-Moulton method.

Solution: We have $k = 2$ and so

$$p(t) = f_n + \frac{f_{n+1} - f_n}{h}(t - t_n) + \frac{f_{n+1} - 2f_n + f_{n-1}}{2h^2}(t - t_n)(t - t_{n+1})$$

Hence

$$\int_{t_n}^{t_{n+1}} p(t)dt = f_n h + \frac{f_{n+1} - f_n}{h}\frac{h^2}{2} + \frac{f_{n+1} - 2f_n + f_{n-1}}{2h^2}\left(-\frac{h^3}{6}\right) = h\left(\frac{5}{12}f_{n+1} + \frac{2}{3}f_n - \frac{1}{12}f_{n-1}\right)$$

and so the 2-step Adam-Moulton method which is given by

$$y_{n+1} = y_n + h\left(\frac{5}{12}f_{n+1} + \frac{2}{3}f_n - \frac{1}{12}f_{n-1}\right).$$

(b) Using the Taylor expansion, show that the local truncation error of the 2-step Adam-Moulton method is $\mathcal{O}(h^4)$.

Solution: The local truncation error is given by

$$\tau_h(t_n) = y(t_{n+1}) - y(t_n) - \frac{5}{12}hf(t_{n+1}, y(t_{n+1})) - \frac{2}{3}hf(t_n, y(t_n)) + \frac{1}{12}hf(t_{n-1}, y(t_{n-1}))$$

for $n = 1, 2 \ldots, N-1$. The idea is to Taylor expand every term in the above expression which is not evaluated at $t = t_n$. We

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \frac{h^3}{6}y^{(3)}(t_n) + \mathcal{O}(h^4), \quad f(t_n, y(t_n)) = y'(t_n),$$

and

$$f(t_{n\pm1}, y(t_{n\pm1})) = y'(t_{n\pm1}) = y'(t_n) \pm hy''(t_n) + \frac{h^2}{2}y^{(3)}(t_n) + \mathcal{O}(h^3).$$

Plugging in the above formulas into the expression of $\tau_{n+1}(h)$ and simplifying shows that $\tau_h(t_n) = \mathcal{O}(h^4)$ as desired.

15. The improved Euler's method is given by

$$\tilde{y}_{n+1} = y_n + hf(t_n, y_n)$$
$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, \tilde{y}_{n+1})\right).$$

Show that the method has order 2. *Hint: Show first that*

$$f(t+h, y+ch) = f(t,y) + hf_t(t,y) + chf_y(t,y) + \mathcal{O}(h^2).$$

by Taylor expanding $g(x) = f(t+x, y+cx)$ *at* $x = h$ *about* $x_0 = 0$.

Solution: Following the hint leads to

$$g(h) = g(0) + g'(0)h + \mathcal{O}(h^2).$$

We have $g(0) = f(t,y)$ and, using the chain rule

$$g'(x) = f_t(t+x, y+cx)\frac{\partial}{\partial x}(t+x) + f_y(t+x, y+cx)\frac{\partial}{\partial x}(y+cx) = f_t(t+x, y+cx) + f_y(t+x, y+cx)c.$$

Then

$$g'(0) = f_t(t,y) + cf_y(t,y)$$

and so

$$f(t+h, y+ch) = f(t,y) + hf_t(t,y) + chf_y(t,y) + \mathcal{O}(h^2)$$

as desired.

The improved Euler's method can be written as

$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))\right).$$

The local truncation error is then given by

$$\tau_h(t_n) = y(t_{n+1}) - y(t_n) - \frac{h}{2}\left(f(t_n, y(t_n)) + f(t_{n+1}, y(t_n) + hf(t_n, y(t_n)))\right) \tag{1}$$

Using the result proven above, we can write

$$f(t_{n+1}, y(t_n) + hf(t_n, y(t_n))) = f(t_n, y(t_n)) + hf_t(t_n, y(t_n)) + hf(t_n, y(t_n))f_y(t_n, y(t_n)) + \mathcal{O}(h^2)$$

Differentiating $y'(t) = f(t, y(t))$ with respect to t implies

$$y''(t) = f_t(t, y(t)) + f_y(t, y(t))f(t, y(t)).$$

Plugging in $t = t_n$, leads to

$$y''(t_n) = f_t(t_n, y(t_n)) + f_y(t_n, y(t_n))f(t_n, y(t_n)).$$

and so

$$f(t_{n+1}, y(t_n) + hf(t_n, y(t_n))) = f(t_n, y(t_n)) + hy''(t_n) + \mathcal{O}(h^2) \tag{2}$$

On the other hand,

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \mathcal{O}(h^3)$$

Since $y'(t) = f(t, y(t))$ and we obtain $y'(t_n) = f(t_n, y(t_n))$ by plugging in $t = t_n$. Therefore

$$y(t_{n+1}) = y(t_n) + hf(t_n, y(t_n)) + \frac{h^2}{2}y''(t_n) + \mathcal{O}(h^3) \tag{3}$$

Simplifying (1) using (2) and (3) leads to

$$\tau_h(t_n) = \mathcal{O}(h^3).$$

Thus the method has order $\mathcal{O}(h^2)$.

16. Consider the boundary value problem on $[0,1]$,

$$\begin{cases} -u''(x) = f(x), \\ u(0) = 0 \\ u(1) = 0 \end{cases}$$

(a) Using the approximation $u''(x_k) = \frac{u_{k+1}-2u_k+u_{k-1}}{h^2} + \mathcal{O}(h^2)$ using finite differences, write down the linear system of the form $A_h\vec{u_h} = \vec{f_h}$ using equally-spaced points $x_k = kh$ for $k = 0, \ldots, N$ with $h = \frac{1}{N}$.

Solution: We have for $k = 1, \ldots, N-1$

$$\frac{-u_{k+1} + 2u_k - u_{k-1}}{h^2} = f(x_k)$$

In particular, for $k = 1$ we have

$$\frac{-u_2 + 2u_1}{h^2} = f(x_1)$$

since $u_0 = u(0) = 0$ and for $k = N-1$

$$\frac{+2u_{N-1} - u_{N-2}}{h^2} = f(x_{N-1})$$

since $u_N = u(1) = 0$. Hence

$$A_h = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & & & \\ -1 & & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}, \quad \vec{f_h} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_{N-1}) \end{pmatrix}$$

(b) For the discretization of part (a), show that the consistency error measured in the l_2 norm is $\mathcal{O}(h^p)$ for some $p > 0$. Find the exponent p in this case.

Solution:

$$\begin{aligned} (A_h\vec{u} - \vec{f_h})_k &= \frac{-u_{k+1} + 2u_k - u_{k-1}}{h^2} - f(x_k) \\ &= -u''(x_k) - f(x_k) + \mathcal{O}(h^2) \\ &= \mathcal{O}(h^2). \end{aligned}$$

Then

$$\left\|A_h\vec{u} - \vec{f_h}\right\|_2^2 = \sum_{k=1}^{N-1} \mathcal{O}(h^4) = (N-1)\mathcal{O}(h^4) = \mathcal{O}(h^3),$$

since $h = \frac{1}{N}$. Hence

$$\left\|A_h\vec{u} - \vec{f_h}\right\|_2 = \mathcal{O}\left(h^{\frac{3}{2}}\right)$$

and so $p = \frac{3}{2}$.

(c) Show that the discretization of part (a) is stable. *Hint: Recall that for constants $a \in \mathbb{R}$, $b < 0$, an*

$n \times n$ matrix of the form

$$A = \begin{pmatrix} a & b & & & \\ b & a & b & & \\ & \ddots & \ddots & \ddots & \\ & & b & a & b \\ & & & b & a \end{pmatrix}$$

has eigenvalues $\lambda_k = a + 2b\cos(\frac{k\pi}{n+1})$ for $k = 1, \ldots, n$. Also note that for $\frac{\sin(x)}{x} \geq \frac{2}{\pi}$ for $0 < x < \frac{\pi}{2}$.

Solution: We need to show that $\left\|A_h^{-1}\right\| \leq C$ for sufficiently small h. We have

$$\left\|A_h^{-1}\right\| = \frac{1}{\min_{k=1,\ldots,N-1} |\lambda_k(A_h)|}$$

Using the hint, we have

$$\lambda_k(A_h) = \frac{2}{h^2}\left(1 - \cos\left(\frac{k\pi}{N}\right)\right) = \frac{4}{h^2}\sin^2\left(\frac{k\pi}{2N}\right) \geq \frac{4}{h^2}\sin^2\left(\frac{\pi}{2N}\right) = \frac{4}{h^2}\sin^2\left(\frac{\pi h}{2}\right).$$

Now, since $\frac{\sin(x)}{x} \geq \frac{2}{\pi}$ for $0 < x < \frac{\pi}{2}$, we have for all $h \leq 1$

$$\lambda_k(A_h) \geq \frac{4}{h^2}\left(\frac{2}{\pi}\frac{\pi h}{2}\right)^2 = 4.$$

Hence $\left\|A_h^{-1}\right\| \leq \frac{1}{4}$.

(d) Conclude that $\vec{u_h}$ from part (a) converges to the exact solution $\vec{u}$ as $h \to 0$.

Solution: In part (c), we showed that $\lambda_k(A) \geq 4$ and so A_h is an invertible matrix. Then, for $h \leq \frac{1}{4}$, we have

$$\begin{aligned} \|\vec{u} - \vec{u_h}\|_2 &= \left\|(A_h^{-1}A_h(\vec{u} - \vec{u_h})\right\|_2 \\ &\leq \left\|A_h^{-1}\left(A_h\vec{u} - \vec{f_h}\right)\right\|_2 \\ &\leq \left\|A_h^{-1}\right\|_2 \left\|A_h\vec{u} - \vec{f_h}\right\|_2 \\ &\leq \frac{1}{4}\left\|A_h\vec{u} - \vec{f_h}\right\|_2. \end{aligned}$$

Since $NormA_h\vec{u} - \vec{f}_{h2} \to 0$ as $h \to 0$ by (b), we conclude that $\vec{u_h}$ converges to $\vec{u}$ as $h \to 0$.

17. Consider the B.V.P on $[a, b]$ given by

$$\begin{cases} u''(t) = f(x, u, u') \\ u(a) = \alpha \\ u(b) = \beta \end{cases} \tag{4}$$

and the I.V.P.

$$\begin{cases} y''(t) = f(x, y, y') \\ y(a) = \alpha \\ y'(a) = s \end{cases} \tag{5}$$

where s is an (unknown) parameter. For a fixed $s \in \mathbb{R}$, denote by $y(x; s)$ the exact solution of (5). Denote by $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$ the numerical solution of (5) with $s = s_k$.

Algorithm Shooting method with secant method

1: Pick two guesses s_0, s_1 and N, *tol*.
2: Compute $\left\{y_0^{(0)}, \ldots, y_N^{(0)}\right\}$ and $\left\{y_0^{(1)}, \ldots, y_N^{(1)}\right\}$.
3: Set $k = 1$.
4: **while** $\left|y_N^{(k)} - \beta\right| < tol$ **do**
5: Set $s_{k+1} = s_k - \frac{y_N^{(k)} - \beta}{y_N^{(k)} - y_N^{(k-1)}}(s_k - s_{k-1})$.
6: Compute $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$.
7: Set $k = k + 1$.
8: **end while**
9: Output $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$.

(a) Consider the function $z(x; s) := \frac{\partial y(x;s)}{\partial s}$. Determine the augmented second order I.V.P for y and z. *Hint: What ODE does z'' satisfy? Do not forget to include the initial conditions.*

Solution: We have

$$\begin{aligned}
z'' &= \frac{\partial^2}{\partial x^2}\left(\frac{\partial y(x;s)}{\partial s}\right) \\
&= \frac{\partial}{\partial s}\left(\frac{\partial^2 y(x;s)}{\partial x^2}\right) \\
&= \frac{\partial}{\partial s}\left(y''(x;s)\right) \\
&= \frac{\partial}{\partial s}\left(f\left(x, y(x;s), y'(x;s)\right)\right) \\
&= \frac{\partial}{\partial s}\left(f\left(x, y(x;s), \frac{\partial y(x;s)}{\partial x}\right)\right) \\
&= f_y\left(x, y(x;s), y'(x;s)\right)\frac{\partial y(x;s)}{\partial s} + f_{y'}\left(x, y(x;s), y'(x;s)\right)\frac{\partial}{\partial x}\left(\frac{\partial y(x;s)}{\partial s}\right) \\
&= f_y(x, y, y')z + f_{y'}(x, y, y')z'.
\end{aligned}$$

Here f_y and $f_{y'}$ denote the partial derivatives of f with respect to its second and third argument, respectively. In addition,

$$z(a;s) = \frac{\partial y(a;s)}{\partial s} = \frac{\partial}{\partial s}(\alpha) = 0, \quad z'(a;s) = \frac{\partial y'(a;s)}{\partial s} = \frac{\partial}{\partial s}(s) = 1.$$

Thus for each s we are interested in augmented I.V.P.

$$\begin{cases}
y'' = f(x, y, y') \\
z'' = f_y(x, y, y')z + f_{y'}(x, y, y')z' \\
y(a) = \alpha \\
y'(a) = s \\
z(a) = 0 \\
z'(a) = 1
\end{cases} \tag{6}$$

(b) Write the algorithm for the shooting method with Newton's method. *Modify the above algorithm as needed.*

Solution: For an initial guess of slope s_0, Newton's method gives the sequence

$$s_{k+1} = s_k - \frac{\phi(s_k)}{\phi'(s_k)}$$

Given the augmented I.V.P. (6) we will have

$$s_{k+1} = s_k - \frac{y_N^{(k)} - \beta}{z_N^{(k)}}$$

Denote by $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$, $\left\{z_0^{(k)}, \ldots, z_N^{(k)}\right\}$ the numerical solution of (6) with $s = s_k$. The algorithm for the shooting method with Newton's method is given by

---

Algorithm Shooting method with Newton's method

1: Pick a guess s_0 and N, *tol*.
2: Compute $\left\{y_0^{(0)}, \ldots, y_N^{(0)}\right\}$ and $\left\{z_0^{(0)}, \ldots, z_N^{(0)}\right\}$.
3: Set $k = 1$.
4: **while** $\left|y_N^{(k)} - \beta\right| < tol$ **do**
5: Set $s_{k+1} = s_k - \frac{y_N^{(k)} - \beta}{z_N^{(k)}}$.
6: Compute $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$, $\left\{z_0^{(k)}, \ldots, z_N^{(k)}\right\}$.
7: Set $k = k + 1$.
8: **end while**
9: Output $\left\{y_0^{(k)}, \ldots, y_N^{(k)}\right\}$.

(a) Compute the first iteration of the Gauss-Seidel method for the linear system

$$\begin{aligned} 4x_1 - 2x_2 &= 2 \\ -2x_1 + 4x_2 - 2x_3 &= 2 \\ -2x_2 + 4x_3 &= -1 \end{aligned} \qquad \text{with } \boldsymbol{x}_0 = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

Solution: The first iteration of the Gauss-Seidel method is given by

$$x_{GS}^{(1)} = \begin{bmatrix} 1 \\ \frac{3}{2} \\ \frac{1}{2} \end{bmatrix}$$

(b) Does the Gauss-Seidel method converge in this case? Justify your answer brifly.

Solution: Yes, the Gauss-Seidel method converges since the matrix A is diagonally dominant.

18. Recall that the S.O.R. method can be written as

$$(D + \theta L)x^{(k+1)} = -\left(\theta U + (\theta - 1)D\right)x^{(k)} + \theta b.$$

Show that $0 < \theta < 2$ is a necessary condition for convergence. *Hint: show first that for any matrix B, if $|\det(B)| \geq 1$ then $\rho(B) \geq 1$.*

Solution: The fact that $|\det(B)| \geq 1$ implies that $\rho(B) \geq 1$ is a simple consequence of the fact that the determinant of a matrix is always the product of its eigenvalues. We can rewrite the S.O.R. method in the form

$$x^{(k+1)} = B(\theta)x^{(k)} + c(\theta),$$

where $B(\theta) = (D + \theta L)^{-1}((1-\theta)D - \theta U)$. For convergence we must have $\rho(B(\theta) < 1$ and so, using the hint, for convergence it is necessary that $|\det(B(\theta))| < 1$.

We then proceed by computing $\det(B(\theta))$. Recalling that the determinant of a triangular matrix (upper or lower) is equal to the product of the diagonal entries, we have

$$\begin{aligned}
\det(B(\theta)) &= \frac{1}{\det(D+\theta L)} \det\left((1-\theta)D - \omega U\right) \\
&= \frac{1}{\det(D)} \det\left((1-\theta)D\right) \\
&= \frac{(1-\theta)^n \det(D)}{\det(D)} \\
&= (1-\theta)^n
\end{aligned}$$

Thus $|\det(B(\theta))| = |1-\theta|^n$. Since $|1-\theta| < 1$ is equivalent to $0 < \theta < 2$, we are done.

19. Let $A \in \mathbb{R}^{n\times n}$ be such that $A = (1+\omega)M - (N + \omega M)$, with $M^{-1}N$ nonsingular and with real eigenvalues $1 > \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and $\omega \neq -1$. Find the values of $\omega \in \mathbb{R}$ for which the following iterative method

$$(1+\omega)Mx^{(k+1)} = (N + \omega M)x^{(k)} + b, \quad k \geq 0,$$

converges for any $x^{(0)}$ to the solution of the linear system $Ax = b$. Determine also the value of ω for which the convergence rate is optimal.

Solution: We can write the iterative method as

$$x^{(k+1)} = B(\omega)x^{(k)} + c(\omega),$$

where

$$B(\omega) = \frac{1}{1+\omega}M^{-1}(N + \omega M) = \frac{1}{1+\omega}(M^{-1}N + \omega I).$$

$B(\omega)$ has eigenvalues

$$\frac{\lambda_1 + \omega}{1+\omega}, \frac{\lambda_2 + \omega}{1+\omega}, \ldots, \frac{\lambda_n + \omega}{1+\omega}.$$

Thus, the iterative method is convergent if and only if $\rho(B(\omega)) < 1$, i.e., for $i = 1, \ldots, n$

$$-1 < \frac{\lambda_i + \omega}{1+\omega} < 1$$

Suppose $1 + \omega < 0$. Then

$$\frac{\lambda_i + \omega}{1+\omega} < 1 \iff \lambda_i + \omega > 1 + \omega \iff \lambda_i > 1$$

which is a contradiction with the assumption on λ_i. Hence $1 + \omega > 0$.

Consequently,

$$\frac{\lambda_i + \omega}{1+\omega} < 1 \iff \lambda_i < 1,$$

which is true by assumption and

$$-1 < \frac{\lambda_i + \omega}{1+\omega} \iff \omega > -\frac{1+\lambda_i}{2}.$$

The above condition must be true for $i = 1, \ldots, n$ and so we can conclude the method is convergent if and only if

$$\omega > -\frac{1+\lambda_n}{2}.$$

The convergence rate is given by $\rho(B(\omega))$ and so the optimal ω is such that $\rho(B(\omega))$ is minimum. We have

$$\rho(B(\omega)) = \max\left\{\frac{|\lambda_1+\omega|}{1+\omega}, \ldots, \frac{|\lambda_n+\omega|}{1+\omega}\right\}$$

We have

$$\frac{|\lambda_i+\omega|}{1+\omega} \leq \max\left\{\frac{|\lambda_1+\omega|}{1+\omega}, \frac{|\lambda_n+\omega|}{1+\omega}\right\}$$

for all $i = 1, \ldots, n$. This can be shown by considering the cases $\lambda_i + \omega > 0$ and $\lambda_i + \omega \leq 0$. Hence

$$\rho(B(\omega)) = \max\left\{\frac{|\lambda_1+\omega|}{1+\omega}, \frac{|\lambda_n+\omega|}{1+\omega}\right\}$$

A graphical argument shows then that the minimum of $\rho(B(\omega))$ is attained at ω such that

$$-\frac{\lambda_1+\omega}{1+\omega} = \frac{\lambda_n+\omega}{1+\omega}$$

Hence

$$\omega_{opt} = -\frac{\lambda_1+\lambda_n}{2}.$$