Practice exam 2 problems with solutions

1

Approximate the first two derivatives of $y(x)$ at the given points using second order accurate finite differences (centered if possible).

x	-1	0	1	2
y	2	2	3	4
y'	-	-	-	-
y''	-	-	-	-

Solution: Using the formulas in the lecture notes and homework, the values are

x	-1	0	1	2
y	2	2	3	4
y'	-0.5	0.5	1	1
y''	2	1	0	-1

2

Given the following data, approximate the first four derivatives of $f(x)$ at $x = 1$ using centered finite differences.

x	-1	0	1	2	3
$f(x)$	1	2	4	3	2

Solution: Using the formulas given in the lecture notes (with $\Delta x = 1$):
$f'(1) \approx (f(2) - f(0))/2 = 0.5$
$f''(1) \approx (f(2) - 2 \cdot f(1) + f(0))/1^2 = -3$
$f'''(1) \approx (-f(-1) + 2 \cdot f(0) - 2 \cdot f(2) + f(3))/(2 \cdot 1^3) = -0.5$
$f''''(1) \approx (f(-1) - 4 \cdot f(0) + 6 \cdot f(1) - 4 \cdot f(2) + f(3))/1^4 = 7$

3

(a) Find D_0^2 for Richardson extrapolation to estimate the derivative of $f(x)$ at $x = 1.1$, with initial step size $h_0 = 0.4$. Use the function values given below,

and show at least 7 significant figures.
(b) What is the estimated fractional error in this obtained $f'(x)$?

x	0.5	0.6	0.7	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5
$f(x)$	61255	67322	74199	81907	90488	100000	110513	122106	134870	148902	164311

Solution:

112640	110393.3	**110388.0**
110955	110388.3	
110530		

The fractional error can be estimated as $\frac{|D_1^1 - D_0^2|}{|D_0^2|} = \mathbf{3.0 \times 10^{-6}}$.

4

(a) Find D_0^2 for Richardson extrapolation to estimate the derivative of $f(x)$ at $x = 1$, with initial step size $\Delta_0 = 0.4$. Use the function values given below.
(b) What is the estimated relative error in this estimated $f'(x)$?

x	0.5	0.6	0.7	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5
$f(x)$	70711	73602	77906	83651	90953	100000	111053	124456	140646	160169	183712

Solution: We have
$D_0^0 = (f(1.4) - f(0.6))/0.8 = 108209$
$D_1^0 = (f(1.2) - f(0.8))/0.4 = 102012$
$D_2^0 = (f(1.1) - f(0.9))/0.2 = 100500$
$D_0^1 = (4D_1^0 - D_0^0)/3 = 99947$
$D_1^1 = (4D_2^0 - D_1^0)/3 = 99996$
$D_0^2 = (16D_1^1 - D_0^1)/15 = \mathbf{99999}$
The relative error can be estimated as $|\frac{D_0^2 - D_1^1}{D_0^2}| = \mathbf{3 \times 10^{-5}}$.

5

Consider the composite trapezoid rule approximation T_{78} of $\int_0^1 x^{95} dx$.

(a) Does T_{78} overestimate or underestimate the exact value? Explain how you know.

(b) Find a bound for the absolute error in T_{78} using the result that $\text{Error}(T_N) \leq \frac{M(b-a)^3}{12N^2}$, where M is the least upper bound for all absolute values of the second derivatives over $[a, b]$ of the function being integrated.

Solution: The second derivative is $95 \cdot 94 \cdot x^{93}$, with a maximum absolute value in the interval of $95 \cdot 94$. So the error bound is

$$\frac{95 \cdot 94 \cdot (1-0)^3}{12 \cdot 78^2} = \mathbf{0.12}$$

(the actual error is considerably less, however).

Because the second derivative is always positive, the composite trapezoid rule will in this case **overestimate** the integral.

6

(a) Estimate the integral $\int_0^2 x^{10} dx$ using the simple Simpson's rule.
(b) Find the minimum number N of equal-width subdivisions needed to guarantee that the absolute error in the composite Simpson rule estimate for this integral is under 10^{-7} using the error bound $\frac{K_4(b-a)^5}{2880N^4}$.

Solution: (a) $(2-0)(f(0)+4f(1)+f(2))/6 = \mathbf{342\frac{2}{3}}$.
(b) The 4th derivative for this function is $5040x^6$, which gets as large as 322560 when $x = 2$. The error bound gives for N a minimum of $\left(\frac{322560(2-0)^5}{(2880)(10^{-7})}\right)^{1/4} = 435.10$, so we need N of at least **436**.

7

Estimate $I = \int_2^4 y(x)dx$ given the values below (a) using the simple Simpson rule,
(b) using the composite Simpson rule with $n = 2$ equal-length intervals.
(c) Based on (a) and (b), estimate what n would be needed for the absolute error in estimating I using the composite Simpson rule to be under 10^{-3}.

x	2	2.5	3	3.5	4
$y(x)$	100	67	50	40	33

Solution: (a) $S_1 = (4-2) \times (y(2)+4y(3)+y(4))/6 = \mathbf{111}$
(b) $S_2 = (4-2) \times (y(2)+4y(2.5)+2y(3)+4y(3.5)+y(4))/12 = \mathbf{110\frac{1}{6}}$
(c) The difference between S_1 and S_2, which we can use as a rough estimate of the error of S_1, is $\frac{5}{6}$, or 833×10^{-3}. To get the absolute error down by a factor of 833, to 10^{-3}, we expect to have to increase n by a factor of $833^{1/4} = 5.4$, since the Simpson rule error is typically proportional to n^{-4}. So $\mathbf{n = 6}$ might be enough.

8

Estimate the integral of $e^{(x^2)}$ between $a = 0$ and $b = 1$ using Simpson's rule with
(a) one subinterval,
(b) two subintervals.
(c) Estimate your fractional error based on the difference between the two results.

Solution: The integrand function values needed are

x	0	0.25	0.5	0.75	1
$f(x)$	1	1.0645	1.2840	1.7551	2.7183

(a) Using the values at 0, 0.5, 1: $S_1 = 1.4757$
(b) Using all the above values: $S_2 = 1.4637$
(c) The fractional difference between S_1 and S_2, $\frac{|S_1 - S_2|}{|S_2|}$ (S_2 being the more accurate estimate), which we can use as an estimate of the error in S_1, is 8.2×10^{-3}. Given that the error in the composite Simpson's rule scales as n^{-4} for smooth functions and large enough n, we could extrapolate to estimate that the fractional error in S_2 might be only 1/16 of that, or 5×10^{-4}, which in this case actually turns out to be only a moderate underestimate (the actual fractional error in S_2 is 7.2×10^{-4}).

9

An irregular lot is 100 ft long, and its width as measured at various points is as follows:

position (ft)	0	25	50	75	100
width (ft)	10	15	20	23	25.

Estimate the lot's area as accurately as possible using the composite Simpson rule.

Solution: With the given data, we can apply Simpson's rule to the two halves of the lot (0 to 50 ft and 50 to 100 ft). The result is $S_2 = \mathbf{1892}$ ft^2.

10

Starting from standstill at $t = 0$, the acceleration of an elevator is measured as

t(s)	0	5	10	15	20	23	30
a(m/s^2)	0	2.2	3.6	5.2	6.0	−4.0	−8.1

(a) Estimate the velocity at $t = 20$ using the composite trapezoid rule.
(b) Write a Python script to estimate the velocity at $t = 20$ by fitting an interpolating polynomial to the data and integrating it.

Solution: (a) The velocity is equal to the integral of the acceleration from $t = 0$ to $t = 20$. Using the composite trapezoid rule, this can be estimated as $(5/2) * (0 + 2 * 2.2 + 2 * 3.6 + 2 * 5.2 + 6.0) = 70$ m/s.
(b)

```
from numpy import concatenate, linspace, array
from numpy import polyfit, polyval, polyint
```

```
t = concatenate((linspace(0, 20, 5), [23, 30]))
a = array([0, 2.2, 3.6, 5.2, 6.0, -4.0, -8.1])
p = polyfit(t, a, len(t)-1)
ip = polyint(p)
v = polyval(ip, 20)
```
(The result is 66.8 m/s.)

11

Write Python code to estimate the probability of falling between $x = 1$ and $x = 2$ for the probability density function $p(x) = e^{-(e^{-x}+x)}$.

Solution: The solution is equal to the definite integral $\int_1^2 p(x)\,dx$, which can be found numerically in Python:

```
from numpy import exp
from scipy.integrate import quad
p = lambda x:  exp(-(exp(-x) + x))
I = quad (p, 1, 2)
I = I[0]  (The result is 0.181)
```

12

Calculate the Romberg integration estimator R_0^2 of

$$\int_0^1 \frac{x^3}{x+1}\,dx.$$

Solution: For $f(x) = \frac{x^3}{x+1}, a = 0, b = 1$, we have
$R_0^0 = (f(0) + f(1))/2 = \frac{1}{4}$
$R_1^0 = (f(0) + 2 * f(0.5) + f(1))/4 = \frac{1}{6}$
$R_2^0 = (f(0) + 2 * f(0.25) + 2 * f(0.5) + 2 * f(0.75) + f(1))/8 = 0.14673$
$R_0^1 = (4R_1^0 - R_0^0)/3 = 901/5400 = 0.13\overline{8}$
$R_1^1 = (4R_2^0 - R_1^0)/3 = 0.14008$
$R_0^2 = (16R_1^1 - R_0^1)/15 = \mathbf{0.14016}$
In this case the true value is $\frac{5}{6} - \log(2) \approx 0.14019$.

13

(a) Use Romberg integration with $j_{\max} = 2$ to estimate

$$I = \int_0^2 \frac{x}{x+3} dx.$$

(b) Estimate the relative error based on the difference between your two most accurate estimates.

Solution:
0.4 0.46667 **0.46751**
0.45 0.46746
0.46310
Estimated relative error: $\frac{|R_1^1 - R_0^2|}{|R_0^2|} = \frac{|0.46746 - 0.46751|}{|0.46751|} = \mathbf{1.1 \times 10^{-4}}$.

14

(a) Fill in the first 3 levels of Romberg integration to estimate the integral of $\sqrt{x}$ for x between 0 and 3.
(b) Estimate the absolute error based on the difference between the two most accurate estimates above.
(c) Find the actual absolute error of the most accurate estimate above compared to the analytic integral.

Solution: (a) The values are
2.598076 3.315515 **3.417804**
3.136155 3.411411
3.342597
(b) $|3.411411 - 3.417804| = 0.00639$
(c) $|3.417804 - 2\sqrt{3}| = 0.0463$ (in this case the error is underestimated at this stage)

15

(a) Use Romberg integration with $j_{\max} = 3$ to estimate $I = \int_0^1 (e^x - 4x)dx$.
(b) Find the solution analytically, and based on this calculate the fractional error of part a.

Solution: (a) For estimating $I = \int_0^1 (e^x - 4x)dx$, the needed function values for the trapezoid rule estimates up to $j_{\max} = 3$ are those at $2^{j_{\max}} + 1 = 9$ equally spaced points:

x	0	1/8	1/4	3/8	1/2	5/8	3/4	7/8	1
f	1	0.633148	0.284025	−0.045009	−0.351279	−0.631754	−0.883000	−1.101125	−1.281718

Based on these, the trapezoid rule estimates are

N	1	2	4	8
T	−0.14086	−0.24607	−0.27278	−0.27948

Using these for the $j = 0$ column in the table of Romberg integral estimates, we get

i \ j	0	1	2	3
0	−0.14086	−0.28114	−0.28172	−0.28172
1	−0.24607	−0.28168	−0.28172	
2	−0.27278	−0.28172		
3	−0.27948			

The best estimate from this is T_0^3 or $\mathbf{-0.28172}$.

(b) The analytic integral is $e^x - 2x^2]_0^1 = e - 3$, which is only about $\mathbf{3 \times 10^{-10}}$ lower than the Romberg value.

16

Use Romberg integration with $j_{\max} = 2$ to estimate $I = \int_0^2 \frac{x^2}{x^2+3} dx$.

Solution:
0.57143 0.52381 **0.51502**
0.53571 0.51557
0.52060

17

Use the (explicit) Euler method to solve for $y(x = 1)$ if $y(x = 0) = 2$ and $y' = \frac{y}{x^2+2}$.

(a) With step size $h = 1$.

(b) With step size $h = 0.5$.

(c) Explain how you can estimate the fractional error in your solution using the above results.

Solution: (a) $y_1 = 2 + 1 \times \frac{2}{0^2+2} = \mathbf{3}$.

(b) $y_1 = 2 + 0.5 \times \frac{2}{0^2+2} = \frac{5}{2}$.

$y_2 = \frac{5}{2} + 0.5 \times \frac{\frac{5}{2}}{0.5^2+2} = \mathbf{\frac{55}{18}}$

(c) Since the first answer $y_{(a)}$ is less accurate because of its larger step size, its error can be estimated by comparison with the more accurate answer $y_{(b)}$, as $\frac{|y_{(a)} - y_{(b)}|}{|y_{(b)}|}$. This gives an estimated fractional error of 1.8% for $y_{(a)}$, which we can conservatively also assume for the more accurate $y_{(b)}$. (In this particular case, the actual fractional errors are 2.9% for $y_{(a)}$ and 1.1% for $y_{(b)}$.)

18

Estimate $y(5)$ for the initial-value problem

$$\frac{dy}{dt} = \sqrt{y} - t + 1, \;\; y(1) = 3$$

using Euler's method (RK1) with step size $h = 1$.

Solution: Using the iteration $y_{i+1} = y_i + h(\sqrt{y_i} - t_i + 1)$, we get $4.7321, 5.9074, 6.3379, \mathbf{5.8554}$

19

(a) Use Euler's method with step size $h = 1/4$ to estimate $y(1)$ given $\frac{dy}{dt} = -2y, y(0) = 3$.
(b) Find the absolute error of this estimate compared to the analytic solution $y(t) = 3e^{-2t}$.
(c) Write Python code to solve for and display $y(1)$ using `ode45`.

Solution: (a) With $\frac{dy}{dt} = -2y$, the Euler method iteration is
$y_{i+1} = y_i + hy'(t_i, y_i) = (1 - 2h)y_i$.
With $h = 1/4$, this gives the estimated values

t	y
0	3
1/4	1.5
1/2	0.75
3/4	0.375
1	**0.1875**.

(b) $E = |3e^{-2} - 0.1875| = \mathbf{0.21851}$.
(c) `f = @(t, y) -2*y;Ω [ts, ys] = ode45(f, [0 1], 3);Ω ys(end)`

20

(a) Use Euler's method with a step size h of $1/3$ to estimate $y(t = 1)$ if $y(t = 0) = 0, dy/dt = y + 2\sqrt{t}$.
(b) Estimate $y(t = 1)$ for the same problem with the RK4 method and $h = 1$.
(c) Write Python code that uses `scipy.integrate.solve_ivp` to estimate $y(t = 1)$ and displays the estimated value.

Solution: (a) The estimates of y after each step are 0, 0.38490, **1.05753**.
(b) $K_1 = 0, K_2 = 1.4142, K_3 = 2.1213, K_4 = 4.1213, y_1 = \mathbf{1.86540}$
(c)

```
from numpy import sqrt
from scipy.integrate import solve_ivp
f = lambda t, y:  y + 2 * sqrt(t)
ts = [0, 1]
y0 = [0]
sol = solve_ivp(f, ts, y0)
sol.y[0,-1]
```

(The result is 2.060)

21

Let $y(t)$ be the solution to $y' = 4te^{-y}$ satisfying $y(0) = 3$. Use Euler's Method with time step $h = 0.1$ to approximate $y(0.1), y(0.2), \ldots, y(0.5)$.

Solution: The estimated values of y at the first 5 timesteps are $3, 3.002, 3.006, 3.012, 3.020$

22

(a) Use Euler's method with a step size h of $1/3$ to estimate $y(t = 1)$ if $y(t = 0) = 0, dy/dt = y + 4t^2$.
(b) Estimate $y(t = 1)$ for the same problem with the RK4 method and $h = 1$.

Solution: (a) The estimates of y after each step are 0, $4/27$, **64/81**.
(b) $K_1 = 0, K_2 = 1, K_3 = 1.5, K_4 = 5.5, y_1 = \mathbf{1.75}$

23

Estimate $y(3)$ for the initial-value problem
$y' = \frac{\sqrt{y}}{2}, y(1) = 1$
using Heun's method (RK2) with step size $h = 1$.

Solution: The result is 1.5562 after 1 step ($K_1 = 0.5, K_2 = 0.61237$) and **2.2372** after 2 steps ($K_1 = 0.62374, K_2 = 0.73823$).

24

Estimate $y(3)$ for the initial-value problem $y'' = \frac{\sqrt{y}}{2}, y(1) = 1, y'(1) = 0$ using Heun's method (RK2) with step size $h = 1$.

Solution: The result for $[y\, y']$ is [1.25 0.5] after 1 step ($K_1 = [0\, 0.5], K_2 = [0.5\, 0.5]$) and [**2.02951** 1.11023] after 2 steps ($K_1 = [0.5\, 0.55902], K_2 = [1.05902\, 0.66144]$).

25

(a) Use 1 step of the RK4 method to solve for $y(x = 6)$ if $y(x = 0) = 2$ and $y' = x\sqrt{y}$.
(b) Write Python code for solving this problem and displaying the result.

Solution: (a) The stages are $K_1 = 0, K_2 = 25.456, K_3 = 69.079, K_4 = 303.51$, giving $y_1 = \mathbf{84.10}$

(b)
```
from numpy import sqrt
from scipy.integrate import solve_ivp
f = lambda x, y:  x * sqrt(y)
xs = [0, 6]
y0 = [2]
sol = solve_ivp(f, xs, y0)
sol.y[0,-1]
```
(The result is 108.49)

26

Estimate $y(t = 2)$ if $y(t = 0) = 0, dy/dt = -2y + 3\sqrt{t}$
with the RK4 method and $h = 1$.

Solution: First step $K_1, K_2, K_3, K_4, y_{i+1}$: 0, 2.1213, 0, 3, 1.2071.
Second step: 0.58579, 0.67423, 0.58579, 0.65685, **1.8342**.

27

(a) Use the implicit Euler method with step size $h = 1/6$ to estimate $y(1)$ given $\frac{dy}{dt} = -2y, y(0) = 3$.
(b) Find the absolute error of the estimated $y(1)$ compared to the analytic solution $y(t) = 3e^{-2t}$.
(c) Write Python code to solve for and display $y(1)$.

Solution: (a) To use the implicit Euler method, we need to solve (explicitly) for y_{i+1} given y_i. This is straightforward in this case:

$$y_{i+1} = y_i + h \cdot f(t_{i+1}, y_{i+1}) = y_i - 2hy_{i+1} \rightarrow y_{i+1} = \frac{y_i}{1+2h}.$$

So at each step the estimated y is multiplied by a constant factor of $\frac{1}{1+2h} = \frac{3}{4}$, giving after 6 steps $\mathbf{\frac{2187}{4096}}$ or 0.53394
(b) Since the true value is $3e^{-2} = 0.40601$, $E_a = |0.53394 - 0.40601| = \mathbf{0.128}$
(c)
```
from numpy import sqrt
from scipy.integrate import solve_ivp
f = lambda t, y:  -2*y
ts = [0, 1]; y0 = [3]
sol = solve_ivp(f, ts, y0)
sol.y[0,-1]
```
(The result is 0.406)

28

Estimate $y(3)$ for the initial-value problem
$y'' = \frac{\sqrt{y}}{2}, y(1) = 5, y'(1) = 0$
using Heun's method (RK2) with step size $h = 1$.

Solution: Writing as a first-order system $\mathbf{z} = \begin{pmatrix} y \\ y' \end{pmatrix}, \mathbf{z}' = \begin{pmatrix} z_2 \\ \frac{\sqrt{z_1}}{2} \end{pmatrix}, \mathbf{z}(1) = \begin{pmatrix} 5 \\ 0 \end{pmatrix}$,

the result is $\begin{pmatrix} 5.5590 \\ 1.1180 \end{pmatrix}$ after 1 step ($K_1 = \begin{pmatrix} 0 \\ 1.11803 \end{pmatrix}, K_2 = \begin{pmatrix} 1.11803 \\ 1.11803 \end{pmatrix}$) and $\begin{pmatrix} 7.2665 \\ 2.3535 \end{pmatrix}$ after 2 steps ($K_1 = \begin{pmatrix} 1.11803 \\ 1.1789 \end{pmatrix}, K_2 = \begin{pmatrix} 2.2969 \\ 1.2920 \end{pmatrix}$), so $y(3) \approx \mathbf{7.2665}$.

29

For the mass-spring system with driving force as given below, use 2 equal-length steps of Euler's method to estimate $x(t)$ at $t = 1$. The initial conditions at $t = 0$ are $x = 1, x' = 0$.

$$x'' + x = 3\cos(t)$$

Solution: After writing the differential equation in standard form ($x'' = 3\cos(t) - x$) and converting to a first-order system ($y_1 \equiv x, y_2 \equiv x'$) $y_1' = y_2, y_2' = 3\cos(t) - y_1$ with $y_1(0) = 1, y_2(0) = 0$, the Euler method gives $y_1 = 1, y_2 = 1$ after one step of length $h = 0.5$, and $y_1 = \mathbf{1.5}, y_2 = 1.8164$ after two steps.

30

(a) Write a Python function, similar to the RK2 one we did in lab, to implement Euler's method for solving a system of first-order differential equations with given initial values. The first line should be
`def rk1(f,tspan,y0,n):`
(b) Write a Python script that uses this `rk1` and a step size h =0.01 to display $y(2)$ if $y'' = \sin(1/y)$ and $y(1) = 10, y'(1) = 3$.

Solution: (a)

```
def rk1(f,tspan,y0,n):
            from numpy import linspace, empty
            a = tspan[0]
            b = tspan[1]
            h = (b-a)/n
            T = linspace(a, b, n+1)
```

```
    Y = empty([n+1, len(y0)])
    Y[0, :]  = y0
    for i in range(n):
                K1 = h * f(T[i], Y[i, :])
                Y[i+1, :]  = Y[i, :]  + K1
    return T, Y
```

(b)

```
from numpy import sin, array
dzdt = lambda t, z:  array([z[1], sin(1 / z[0])])
tspan = [1, 2]
y0 = [10, 3]
h = 0.01
n = round((tspan[1] - tspan[0]) / h)
T, Z = rk1(dzdt, tspan, y0, n)
Z[-1, 0]
```

(The result is 13.045)

31

Write a Python script that solves numerically the initial-value problem
$y'' + y' + y^2 = 1, y(0) = 2, y'(0) = -2$
and plot $y(t)$ for $0 \leq t \leq 10$.

Solution: Need to convert to a first-order ODE system ($z_1 = y, z_2 = y'$), after which an IVP solver like `solve_ivp` can be called:

```
from scipy.integrate import solve_ivp
from matplotlib.pyplot import plot
f = lambda t, z:  [z[1], -z[1] - z[0]**2 + 1]
zi = [2, -2]
ts = [0, 10]
sol = solve_ivp (f, ts, zi)
plot (sol.t, sol.y[0, :])
```

($y(x = 10) \approx 1.008$)

32

Write a Python script that (a) numerically estimates $\Theta(2)$ if $\Theta'' = -9\Theta$ and $\Theta(0) = 1, \Theta'(0) = 0$ and (b) finds the absolute error of this estimate relative to the analytic solution $\Theta(t) = \cos(3t)$.

Solution: With $z_1 = \Theta, z_2 = \Theta'$,

```
from scipy.integrate import solve_ivp
from math import cos
f = lambda t, z:  [z[1], -9*z[0]]
```

```
zi = [1, 0]
ts = [0, 2]
sol = solve_ivp (f, ts, zi)
err = abs (sol.y[0, -1] - cos(3*ts[-1]))
```

33

(a) Set up a system of linear equations (in matrix form) using finite difference with $n = 4$ to solve the equation $y'' + y' = x + 2$ subject to the boundary conditions $y(x = 0) = 0$ and $y(x = 3) = 1$.
(b) Solve this system using Gaussian elimination or LU decomposition, and sketch $y(x)$.

Solution: A finite-difference approximation to the ODE would be

$$(h^{-2} - (2h)^{-1})y_{i-1} - 2h^{-2}y_i + (h^{-2} + (2h)^{-1})y_{i+1} = x_i + 2.$$

Applying this here with $h = \frac{3}{4}$ for $i = 1, 2, 3$, $x_i = 0.75, 1.5, 2.25$, we get

$$\left(\begin{array}{ccc|c} -32/9 & 22/9 & 0 & 2.75 \\ 10/9 & -32/9 & 22/9 & 3.5 \\ 0 & 10/9 & -32/9 & 4.25 - (22/9)(1), \end{array}\right.$$

which can be solved to get the estimates $y(0.75) = -2.67, y(1.5) = -2.76, y(2.25) = -1.37$. Draw a smooth curve connecting these with the boundary conditions to sketch $y(x)$.

34

Set up a linear system in matrix form to solve the boundary-value problem $y'' + 5y' = 10$, $y(0) = 1$, $y(4) = 3$ for values of the function y(x), using second-order-accurate centered finite-difference approximations for the derivatives, with n=5. You do not need to solve the linear system.

Solution: At each interior grid point, the differential equation can be approximated with the finite differences as $\frac{y_{i-1}-2y_i+y_{i+1}}{h^2} + 5\frac{-y_{i-1}+y_{i+1}}{2h} = 10$, with $h = 0.8$ here. Including also the boundary conditions for y_0 and y_5 as the first and last equations, the linear system is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{25}{16} & -\frac{25}{8} & \frac{75}{16} & 0 & 0 & 0 \\ 0 & -\frac{25}{16} & -\frac{25}{8} & \frac{75}{16} & 0 & 0 \\ 0 & 0 & -\frac{25}{16} & -\frac{25}{8} & \frac{75}{16} & 0 \\ 0 & 0 & 0 & -\frac{25}{16} & -\frac{25}{8} & \frac{75}{16} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 1 \\ 10 \\ 10 \\ 10 \\ 10 \\ 3 \end{pmatrix}$$

35

Set up a system of algebraic equations (in matrix form) using finite difference with $n = 5$ for $y'' - y' + y = 2$ subject to the boundary conditions $y(0) = 0$ and $y'(3) = 2$. Use second-order-accurate (centered where possible) finite difference approximations. You only need to write down the system, not to solve it.

Solution: Replacing the derivatives with second-order-accurate centered finite difference approximations, we have for the interior grid points

$$\left(\frac{1}{h^2} + \frac{1}{2h}\right) y_{i-1} + \left(1 - \frac{2}{h^2}\right) y_i + \left(\frac{1}{h^2} - \frac{1}{2h}\right) y_{i+1} = 2$$

and for the first-derivative boundary condition

$$\frac{1}{2h} y_{n-2} - \frac{2}{h} y_{n-1} + \frac{3}{2h} y_n = 2.$$

Here $h = 3/5$, so the linear system is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 65/18 & -41/9 & 35/18 & 0 & 0 & 0 \\ 0 & 65/18 & -41/9 & 35/18 & 0 & 0 \\ 0 & 0 & 65/18 & -41/9 & 35/18 & 0 \\ 0 & 0 & 0 & 65/18 & -41/9 & 35/18 \\ 0 & 0 & 0 & 5/6 & -10/3 & 5/2 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

The unknowns y_i are the values of $y(x)$ at the grid points, which are equally spaced x_i from $x = 0$ to $x = 3$.

36

Set up a system of algebraic equations (in matrix form) using finite difference with $n = 5$ to solve $y'' + y' - y = 4$ subject to the boundary conditions $y(x = 0) = 0$ and $y'(x = 3) = 2$. Use second-order-accurate (centered where possible) finite difference approximations. For full credit, solve this system and sketch $y(x)$.

Solution: Replacing the derivatives with second-order-accurate finite difference approximations, we have for the interior grid points

$$\left(\frac{1}{h^2} - \frac{1}{2h}\right) y_{i-1} + \left(-1 - \frac{2}{h^2}\right) y_i + \left(\frac{1}{h^2} + \frac{1}{2h}\right) y_{i+1} = 4$$

and for the first-derivative boundary condition

$$\frac{1}{2h} y_{n-2} - \frac{2}{h} y_{n-1} + \frac{3}{2h} y_n = 2.$$

Here $h = 3/5$, so the linear system is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 35/18 & -59/9 & 65/18 & 0 & 0 & 0 \\ 0 & 35/18 & -59/9 & 65/18 & 0 & 0 \\ 0 & 0 & 35/18 & -59/9 & 65/18 & 0 \\ 0 & 0 & 0 & 35/18 & -59/9 & 65/18 \\ 0 & 0 & 0 & 5/6 & -10/3 & 5/2 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \\ 4 \\ 4 \\ 4 \\ 2 \end{pmatrix}$$

The unknowns y_i are the values of $y(x)$ at the grid points, which are equally spaced x_i from $x = 0$ to $x = 3$. The solution to the linear system is

$$\mathbf{y} = \begin{pmatrix} 0 \\ -1.921 \\ -2.380 \\ -2.178 \\ -1.565 \\ -0.561 \end{pmatrix},$$

which points can be connected to sketch $y(x)$.

37

Set up a system of algebraic equations (in matrix form) using finite difference with $n = 5$ to solve $y'' + y' + y = 2$ subject to the boundary conditions $y(x = 0) = 0$ and $y'(x = 1) = 2$. For full credit, solve this system and sketch $y(x)$.

Solution: Replacing the derivatives with second-order-accurate finite difference approximations, we have for the interior grid points

$$\left(\frac{1}{h^2} - \frac{1}{2h}\right) y_{i-1} + \left(1 - \frac{2}{h^2}\right) y_i + \left(\frac{1}{h^2} + \frac{1}{2h}\right) y_{i+1} = 2$$

Here $h = 1/5$, so the linear system is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 22.5 & -49 & 27.5 & 0 & 0 & 0 \\ 0 & 22.5 & -49 & 27.5 & 0 & 0 \\ 0 & 0 & 22.5 & -49 & 27.5 & 0 \\ 0 & 0 & 0 & 22.5 & -49 & 27.5 \\ 0 & 0 & 0 & 2.5 & -10 & 7.5 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

The unknowns y_i are the values of $y(x)$ at the grid points, which are equally spaced x_i from $x = 0$ to $x = 1$. The solution to the linear system is

$$\mathbf{y} = \begin{pmatrix} 0 \\ 1.433 \\ 2.627 \\ 3.580 \\ 4.303 \\ 4.810 \end{pmatrix},$$

which points can be connected to sketch $y(x)$.

38

(a) Set up a system of linear algebraic equations (in matrix form) using finite difference with $n = 5$ to solve $y'' + y = x + 4$ subject to the boundary conditions $y(x = 0) = 0$ and $y(x = 5) = 2$. Explain what the unknowns in this system represent.
(b) Write Python code to solve the same problem numerically and to plot $y(x)$.

Solution: (a) Replacing the second derivative with a second-order-accurate finite difference approximation, we have for the interior grid points

$$\frac{1}{h^2} y_{i-1} + \left(1 - \frac{2}{h^2}\right) y_i + \frac{1}{h^2} y_{i+1} = x_i + 4$$

Here $h = 1$, so the linear system is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 5 \\ 6 \\ 7 \\ 8 \\ 2 \end{pmatrix}$$

The unknowns y_i are the values of $y(x)$ at the grid points x_i.

(b)

```
from scipy.integrate import solve_bvp
from matplotlib.pyplot import plot
dzdx = lambda x, z:  [z[1], x + 4 - z[0]]
bc = lambda za, zb:  [za[0], zb[0] - 2]
xi = [0, 5]
zi = [[1, 0.4], [1, 0.4]]
sol = solve_bvp(dzdx, bc, xi, zi)
plot (sol.x, sol.y[0, :])
```

(For example, $y(x = 2.5) \approx 13.366$)

39

(a) Set up a system of linear equations (in matrix form) using finite difference with $n = 4$ to solve the equation $y'' - y = x + 3$ subject to the boundary conditions $y(x = 0) = 0$ and $y(x = 2) = 0$.
(b) Solve this system (for example, using Gaussian elimination) and sketch $y(x)$.
(c) Write Python code to solve the boundary-value problem numerically and plot $y(x)$.

Solution: A finite-difference approximation to the ODE would be

$$h^{-2}y_{i-1} - (1 + 2h^{-2})y_i + h^{-2}y_{i+1} = x_i + 3.$$

Applying this here with $h = \frac{1}{2}$ for $i = 1, 2, 3$, $x_i = 0.5, 1, 1.5$, we get

$$\begin{array}{ccc|c} -9 & 4 & 0 & 3.5 \\ 4 & -9 & 4 & 4 \\ 0 & 4 & -9 & 4.5, \end{array}$$

which can be solved to get the estimates $y(0.5) = -1.01, y(1) = -1.39, y(1.5) = -1.12$. Draw a smooth curve connecting these with the boundary conditions to sketch $y(x)$.
(c) Converting to a first-order system with $z_1 \equiv y, z_2 \equiv y', z_1' = z_2, z_2' = z_1 + x + 3$ and boundary conditions $z_1(0) = z_1(2) = 0$,

```
from scipy.integrate import solve_bvp
from matplotlib.pyplot import plot
from numpy import zeros
dzdx = lambda x, z:  [z[1], x + 3 + z[0]]
bc = lambda za, zb:  [za[0], zb[0]]
xi = [0, 2]
zi = zeros((2, 2))
sol = solve_bvp(dzdx, bc, xi, zi)
plot (sol.x, sol.y[0, :])
```

(For example, $y(x = 1) \approx -1.408$)

40

(a) Set up a system of linear equations (in matrix form) using finite difference with $n = 6$ to solve the equation $y'' + 3y' = y + 4$ subject to the boundary conditions $y(x = 0) = 0$ and $y(x = 1) = 0$.
(b) Solve this system (for example, using Gaussian elimination) and sketch the solution $y(x)$.

Solution: (a) A discretized approximation for $y'' + 3y' = y + 4$, assuming a grid with equal spacings h, is

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + 3\frac{y_{i+1} - y_{i-1}}{2h} = y_i + 4,$$

or

$$(h^{-2} - 1.5h^{-1})y_{i-1} + (-2h^{-2} - 1)y_i + (h^{-2} + 1.5h^{-1})y_{i+1} = 4.$$

With $h = 1/6$, the coefficients are $27, -73, 45$ respectively. Also putting in the boundary conditions, the linear system in augmented matrix form becomes

$$\left[\begin{array}{ccccc|c} -73 & 45 & 0 & 0 & 0 & 4 \\ 27 & -73 & 45 & 0 & 0 & 4 \\ 0 & 27 & -73 & 45 & 0 & 4 \\ 0 & 0 & 27 & -73 & 45 & 4 \\ 0 & 0 & 0 & 27 & -73 & 4, \end{array}\right]$$

where the unknowns $y_1, y_2, \ldots y_5$ are the approximate function values at $1/6, 1/3, \ldots, 5/6$.

(b) Since this is a tridiagonal system, it can be solved relatively quickly by Gaussian elimination (without pivoting). The steps are

$$\left[\begin{array}{ccccc|c} -73 & 45 & 0 & 0 & 0 & 4 \\ 27 & -73 & 45 & 0 & 0 & 4 \\ 0 & 27 & -73 & 45 & 0 & 4 \\ 0 & 0 & 27 & -73 & 45 & 4 \\ 0 & 0 & 0 & 27 & -73 & 4, \end{array}\right] \rightarrow \left[\begin{array}{ccccc|c} -73 & 45 & 0 & 0 & 0 & 4 \\ 0 & -56.356 & 45 & 0 & 0 & 5.479 \\ 0 & 27 & -73 & 45 & 0 & 4 \\ 0 & 0 & 27 & -73 & 45 & 4 \\ 0 & 0 & 0 & 27 & -73 & 4, \end{array}\right] \rightarrow$$

$$\left[\begin{array}{ccccc|c} -73 & 45 & 0 & 0 & 0 & 4 \\ 0 & -56.356 & 45 & 0 & 0 & 5.479 \\ 0 & 0 & -51.441 & 45 & 0 & 6.625 \\ 0 & 0 & 27 & -73 & 45 & 4 \\ 0 & 0 & 0 & 27 & -73 & 4, \end{array}\right] \rightarrow \left[\begin{array}{ccccc|c} -73 & 45 & 0 & 0 & 0 & 4 \\ 0 & -56.356 & 45 & 0 & 0 & 5.479 \\ 0 & 0 & -51.441 & 45 & 0 & 6.625 \\ 0 & 0 & 0 & -49.381 & 45 & 7.477 \\ 0 & 0 & 0 & 0 & -48.395 & 8.088, \end{array}\right]$$

After which back substitution gives $y_5 = -0.1671, y_4 = -0.3037, y_3 = -0.3945, y_2 = -0.4122, y_1 = -0.3089$. Together with the boundary conditions $y_0 = 0, y_6 = 0$, these values can be used to sketch $y(x)$ as a downward-pointing parabola-like (but asymmetric) curve with a minimum (y less than -0.41) near $x = 0.4$.