Assignment 3

This assignment consists of three exercises, covering Chapters 2, 3 and 4 of the book, respectively.

1 Exercise 1: 2-point BVPs

In this exercise, we want to solve the nonlinear BVP (equation (2.91) in the book) with Dirichlet boundary conditions

$$\epsilon u'' + u(u' - 1) = 0, \quad 0 \leq t \leq 1, \tag{1a}$$

$$u(0) = \alpha, \ u(1) = \beta \tag{1b}$$

using a certain number of methods. For the single shooting method, it means that we will transform the BVP (1) into the following initial value problem (IVP)

$$\epsilon u'' + u(u' - 1) = 0, \quad 0 \leq t \leq 1, \tag{2a}$$

$$u(0) = \alpha, \ u'(0) = \sigma, \tag{2b}$$

where σ is unknown and must be chosen to satisfy the boundary condition $u_\sigma(1) = \beta$. You may use `ode45` or any other Runge-Kutta solver (including the solvers you developed for Assignment 1) to numerically solve (2).

For this exercise, we will take $\alpha = -1$, $\beta = 1.5$, and you may start considering $\epsilon = 0.1$.

1.1 Newton's method for solving nonlinear BVPs

1. Derive a second-order accurate scheme on a uniform grid to numerically solve (1). Compute the local truncation τ_j at any grid point x_j and show that it is actually a $O(h^2)$. What is the dominant term in τ_j?

2. Solve the BVP (1) using the Newton's method on a uniform grid. Hint: Remember that the Newton's method requires a suitable initial guess. You may use the initial guess $\bar{u}(x)$ provided in (2.105) in the book. Equation (2.106) will help you to compute the inverse of the Jacobian involved in Newton iterations.

3. How would you improve the scheme based on Newton's method to achieve a better accuracy while using the same number of grid points?

1.2 Shooting methods

1. Rewrite (2) as an IVP in standard form, ie.

$$x'(t) = f(x(t)), \tag{3a}$$
$$x(0) = x_0. \tag{3b}$$

2. Solve (1) using a single shooting method. **Hint:** For this question, you will have to find $u'(0)$ such that $u_\sigma(1) - \beta = 0$. Since it is a scalar function, you can either use the Newton's method, the secant or the bisection method to solve the nonlinear equation $u_\sigma(1) - \beta = 0$(briefly argue your choice).

3. Sensitivity analysis: Compute numerically the sensitivity of your solution at the final time $t = 1$ with respect to σ, ie. compute

$$S_\sigma(t = 1) = \frac{\partial u_\sigma}{\partial \sigma}(t = 1). \tag{4}$$

   Comment on the result.

4. Try the single shooting with different values of ϵ. Which issue occurs when ϵ becomes too small? How would you explain this and solve this issue (you are not required to implement the idea you come up with, but you can try if you have time)?

2 Exercise 2: 9-point Laplacian

In this exercise, we will assume that our spatial domain is a unit square $[0, 1] \times [0, 1]$ discretized with a regular grid with $m \times m$ interior points. Nodal points therefore have coordinates (ih, jh), $0 \leq i, j \leq m+1$ (both interior and boundary nodes) or $1 \leq i, j \leq m$ (interior nodes only); $h = 1/(m + 1)$ as usual. We will also assume that the Dirichlet boundary conditions are prescribed everywhere on the boundary.

Write a code implementing the nine-point Laplacian scheme with a corrected right-hand side (see Section 3.5; use the equation following (3.19) in the book.). Debug your code eg. on a problem with

$$u_{0,exact}(x, y) = \sin(4\pi(x + y)) + \cos(4\pi xy)$$

and make sure your code achieves convergence of $O(h^4)$ in terms of the global error.

Now consider the following two cases:

$$u_{1,exact}(x, y) = x^2 + y^2$$

and

$$u_{2,exact}(x, y) = \sin(2\pi|x - y|^{2.5}).$$

Estimate the rates of convergence of the global error for both of these cases. Explain why these rates differ from $O(h^4)$.

3 Exercise 3: Iterative solvers in 2D

In this exercise, we will ultimately implement a multigrid-based solver for the Poisson equation in 2D. For simplicity, we will use the same assumptions as in Exercise 2, ie. that our spatial domain is a unit square $[0,1] \times [0,1]$ discretized with a regular grid with $m \times m$ interior points. Nodal points therefore have coordinates (ih, jh), $0 \leq i, j \leq m+1$ (both interior and boundary nodes) or $1 \leq i, j \leq m$ (interior nodes only); $h = 1/(m+1)$ as usual. We will also assume that the Dirichlet boundary conditions are prescribed everywhere on the boundary. Thus we end up with m^2 linear equations and m^2 unknowns

$$AU = F$$

where the contributions from the boundary conditions are moved to the right hand side F.

3.1 Matrix-free 5-point Laplacian

Implement a function taking an m^2-vector U as an argument and producing a product $-A^h U$ as an output, where A^h corresponds to the 5-point Laplacian discretization in 2D. The code must not store the matrix A^h (matrix-free method):

```
function AU=Amult(U,m)
```

Use Matlabs implementation of the Conjugate Gradient algorithm (`help pcg`) to solve the discretized Poisson equation up to the to the tolerance $O(h^2)$.

Note that we need to use the equations $-A^h U = -F$ as the PCG requires a positive definite system of equations, whereas A^h is negative definite - hence the minus sign.

3.2 Under/over-relaxed Jacobi smoothing

Go through the analysis of the Jacobi/underrelaxed Jacobi iteration (Section 4.6.1) for the 5-point Laplacian discretization in 2D and compute a closed form expression for the eigenvalues $\gamma_{p,q}, 1 \leq p, q \leq m$ as a function of h, p, q, and ω (you may want to take a look at Section 3.4 here). Write a Matlab program that plots $\max_{m/2 \leq p,q \leq m} |\gamma_{p,q}|$ for $0 \leq \omega \leq 2$ for a given m. Experiment with several values of m and find a good value of ω which "visually minimizes" $\max_{m/2 \leq p,q \leq m} |\gamma_{p,q}|$.

Note that for smoothing to work properly, the value $\max_{m/2 \leq p,q \leq m} |\gamma_{p,q}|$ should be as small as possible and certainly smaller than one, as this value determines how the high frequency components in the error are damped after one under/over-relaxed Jacobi iteration.

Implement a matrix-free relaxed Jacobi iteration for the 5-point Laplacian discretization in 2D and the optimal value of ω that you have found as a Matlab

function, say

```
function Unew=smooth(U,omega,m,F)
```

where `U` is the current iterate, `Unew` is the new iterate, `omega` is the relaxation parameter, `m` is the size of the grid, and `F` is the right-hand size vector of length m^2 (incorporating the boundary conditions!).

3.3 Multigrid solver

Now you have two ingredients out of four for the multigrid solver: the computation of the residual `Amult` (since `Amult` computes $-A^h U$, the residual is `F + Amult(U, m))` and a pre/post smoother `smooth`. There are two missing pieces: coarsening of the residual, and interpolation of the error (see Section 4.6.2).

Assuming that $m = 2^k - 1$ for some $k > 1$, the coarsened grid will have $m_c \times m_c$ internal nodes where $m_c = (m-1)/2$. Generalize the 1-dimensional implementation of coarsening/interpolation from the Matlab script `mgrid2level` to the present 2D case, and implement them as two separate functions:

```
function Rc=coarsen(R,m)
```

and

```
function R=interpolate(Rc,m).
```

Finally, combine everything into a V-cycle of the multigrid, ie. update the provided script `VCycle.m`.

Keeping the tolerance constant, study the dependence of the number of the "outer" multigrid iterations needed to achieve this tolerance on the number of grid points $m = 2^k - 1$.

Optional: You may change your code to implement the W-cycle of the multigrid (it only requires a minor alteration of the code) and compare it with the V-cycle.

Hand in a report describing your findings and the code to Anne-Mette Larsen, Building 303B, Office 105. We must receive the report no later than: April 23, 2018, 17:00. You must also upload an electronic version to CampusNet (a pdf file, and a zip file with all Matlab code, LaTeX code etc used to answer the questions).

On the report you should clearly write your name, study number and group number.

REMEMBER: Your report should include an answer of the questions and the code documenting how you found the solution. In CampusNet you should upload the pdf as well as zip file with your code.