Optional Exercise for the Exam

Advance Concept of Scientific Computing - SCCOMP - SoSe 2020)

The discussion of one the following exercises can be used as a starting point for the oral exam (you will have indicatively ∼10 minutes for illustrating results, you may choose some points to focus on and the format you prefer to present the results)

1. **Swift-Hohenberg / Phase-Field Crystal Model**. Let us consider the following 1D energy functional

$$F[u] = \int_\Omega \left(u \left(\left(q_0^2 + (\partial_x)^2 \right)^2 - \epsilon \right) \frac{u}{2} + \frac{u^4}{4} \right) \qquad \text{with} \quad \Omega = \left[0, \frac{2\pi m}{q_0} \right], 1 \ll m \in \mathbb{N} \tag{1}$$

   with $u \equiv u(x)$, $(\partial_x)^n u \equiv \partial^n u / \partial x^n$. F[u] is expected to be minimized by a *periodic function*.

   (a) Derive the L^2 and the H^{-1} gradient flows of $F[u]$ (Hint: remember $\delta F/\delta u = \sum_{i=0}^{n} (-1)^i (\partial_x)^i [\partial f / \partial((\partial_x)^i u]$ with $F[u] = \int f(u)dx$, $(\partial_x)^0 \equiv 1$ and n the highest order of derivatives in $f(u)$.

   (b) Choose a method and devise an integration scheme suitable for the integration of the resulting equations for $u(x,t)$ with *periodic boundary conditions.*

   (c) Implement a computer program to illustrate the dynamics described by the two gradient flows by considering an initial value problem with $u(x,0)$ a periodic function (period $2\pi/q_0$) and/or a random noise around 0.

   (d) Show convergence of the solution for small timesteps, comment the results in terms of expected properties for the gradient flow and show the dependence of the equilibrium state / stationary solution on the parameter ϵ (Hint: suggested range $\epsilon \in [-1.0; 1.0]$)

2. **Generalized logistic model**. The logistic model for population growth is derived by assuming a nonlinear growth rate for $u \equiv u(t)$,

$$u' = a(u)u, \qquad u(0) = u_0 \tag{2}$$

   and the logistic model arises from the simplest possible choice of $a(u) = r(u) = \rho(1 - u/M)$, where M is the maximum value of u that the environment can sustain, and ρ is the growth under unlimited access to resources (i.e. when u is small). An $a(u)$ that generalizes the linear choice is the polynomial form

$$a(u) = \rho(1 - u/M)^p \tag{3}$$

   where $p > 0$ is some real number.

   (a) Formulate a Forward Euler, Backward Euler, and a Crank-Nicolson scheme [Hint. Use a geometric mean approximation in the Crank-Nicolson scheme: $[a(u)u]^{n+1/2} \approx a(u^n)u^{n+1}$]

   (b) Formulate Picard and Newton iteration for the Backward Euler scheme in (a).

   (c) Implement the numerical solution methods from (a) and (b) in a computer program and check the asymptotic limit of the solutions with different p [Hint: You need to experiment to find infinite time as it increases substantially with p]. The case $p = 1$, $\rho = M = 1$, may be compare to Exercise 4 for consistency.

   (d) Perform numerical experiments with Newton and Picard iteration for the model and show qualitatively how they perform with Δt and p

3. **Traveling Salesman Problem:** Let us consider a set of N uniformly distributed points $\mathbf{x}_i \in \Omega \subset \mathbb{R}^2$. We aim at finding the shortest itinerary, which will visit every point exactly once and return to the point of origin. Set up the problem by generating N points uniformly distributed in a square with side 1 (Hint: you may want to generate them with a defined seed, to compare the same initial configuration for a given N). Set an initial path/itinerary through the points, i.e. an initial sequence of points $\mathbf{x}_i$. Through the Simulated Annealing algorithm, find then the sequence of points (indexed by i), that minimizes the following objective function

$$f(\{x_i\}) = ||\mathbf{x}_N - \mathbf{x}_1|| + \sum_{i=1}^{N-1} ||\mathbf{x_i} - \mathbf{x}_{i+1}|| \tag{4}$$

(a) Illustrate the algorithm in terms of: i) choice of the *move* to be evaluated per call to the SA algorithm (e.g. exchange of some indexes); ii) criterion for acceptance/rejection of the *move* [Hint: in the simulate annealing it depends on a parameter T, namely the *temperature*] iii) cooling schedule for decreasing T from an initial value T_{ini} up to 0.

(b) Implement the algorithm in a computer program, and for a fixed N, show the solution of the problem (More simulations may be needed)

(c) Show how your algorithm/code performs by increasing N (so, the number of iterations required, rejected iterations, etc.).

(d) Modify the algorithm and/or reformulate the problem to find now the longest path visiting all the points once and returning to the point of origin, i.e., maximizing (4).