Technische Universität München
Institut für Informatik
Dr. Miriam Mehl

ST 2008

Scientific Computing II

Final Exam, July 11, 2008

Iterative Solvers (31 pts + 4 extra pts, 60 min)

a) Steepest Descent and Conjugate Gradients (13 pts)

(i) The steepest descent method is an iterative method to detect the minimum of a given function. However, we can use it to solve systems of linear equations as well. Shortly describe and justify the step from a system of linear equations to a problem to which we can apply the steepest descent method (no formulas required, approx. two sentences, 2 pts).

(ii) What happens in one iteration of the steepest descent method? (one sentence, no details!, 1 pt)

(iii) Name the three steps of one iteration of the steepest descent method in their correct algorithmic order (3 pts).

(iv) Name the main difference between the steepest descent and the conjugate graduent method (one sentence, no formulas, 1 pt).

(v) Assume you have to solve the system

$$Au = f$$

of linear equations. Write a pseudo code for one iteration of the conjugate gradient method (including declaration of the repective function with all output and input parameters, 6 pts). You may use the operations

$v_1^T v_2$	for the scalar product of two vectors v_1 and v_2,
Bv	for the multiplication of a matrix B with a vector v.

b) Relaxation Methods and Multigrid (12 pts + 4 extra pts)

We have to solve the two-dimensional Poisson equation

$$u_{xx}(x,y) + u_{yy}(x,y) = f(x,y) \text{ in }]0;1[^2$$

with boundary conditions

$$u(x,y) = 1 \text{ on } \partial]0;1[^2.$$

We discretise the equation on a regular cartesian grid with N unknowns per coordinate direction and use the well-known five-point stencil

$$\frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

to discretise the Laplace operator.

(i) Write down a pseudo-code for a function performing one Jacobi iteration for the resulting system of linear equations (including declaration of the respective function with all output and input parameters, 4 pts).

(ii) We know from the lecture the eigenvectors

$$q^{m,n} = (\sin(\pi mih)\sin(\pi njh))_{i,j=1,\dots,N}$$

and the respective eigenvalues

$$\lambda^{m,n} = \frac{1}{2}\left(\cos(\pi mh) + \cos(\pi nh)\right)$$

of the iteration matrix of the Jacobi method, where $N = \frac{1}{h} - 1$ is the number of unknowns per coordinate direction.

Is the Jacobi method a good smoother? Shortly justify your answer (one or two sentences, 2 pts).

(iii) Name the seven main steps of a multigrid v-cycle in their correct algorithmic order (5 pts).

(iv) Why do we need good smothers for a good multigrid method (give one reason, 1 pt)?

(v) [1] We introduce the damped Jacobi method. That is, we multiply the correction term by which we change the local values of the unknown variable in each iteration with a damping factor λ. In this case, we still have the same eigenvectors of the iteration matrix as for the original Jacobi method but the modified eigenvalues

$$\lambda^{m,n} = 1 - \omega + \frac{\omega}{2}\left(\cos(m\pi h) + \cos(n\pi h)\right).$$

Determine

$$\min_{m,n} \lambda^{m,n}$$

and a damping factor for which the damped Jacobi method would be a good smoother (4 pts).

[1] With this question, you can earn extra points (over 100%), it is not neccessary to solve to get the full score in the exam.

c) Convergence of Iterative Solvers (6 pts)

In the following table, you see iteration numbers for different iterative solvers for the two-dimensional Poisson equation described in **b)**. Complete the tabular with iteration numbers you would expect in the empty fields (6 pts).

h	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$	$\frac{1}{256}$
Jacobi	$2,500$			
Gauss-Seidel	$1,250$			
SOR	90			
Multigrid	5			
steepest descent	$2,700$			
conjugate gradients	80			

Molecular Dynamics (19 pts, 30 min)

a) General Overview (14 pts)

A general task in scientific computing is to simulate a physical problem. There are usually different steps to be done to get from a physical scenario to the simulation. You'll have to describe the different steps involved in the case of a molecular dynamics simulation with n molecules.

(i) Which kind of problems are tackled by molecular dynamics? Give two examples where a molecular dynamics simulation could be useful. (2 pts)

(ii) The first step is to build a physical model of the real world problem. One of these models is the Lennard-Jones 12-6 potential

$$U_{LJ}(r_{ij}) = 4\epsilon \left(\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right),$$

which is composed of two parts, one of them is responsible for attraction and the other one for repulsion. Shortly describe and justify which part has which effect on the two involved molecules. Give two examples of forces on molecules which can't be modeled using the Lennary-Jones 12-6 potential. (3 pts)

(iii) In molecular dynamics simulations, the positions of all molecules are calculated for successive time steps. Shortly explain (no details, no mathematical derivations) how to get from the Lennard-Jones potential to a formula which allows the calculation of the molecules positions at a new time step. (3 pts)

(iv) During the numerical calculation of the new positions, errors occur (discretisation errors, roundoff errors). Now assume you are performing a simulation with 100000 time steps. How strong will the calculated position of the particles differ from the "real" positions? Is this error critical for the usability of the simulation? (2 pts)

(v) In (iii), you have described the necessary steps to get a formula for the new positions of all n molecules (interacting vai LJ potentials). Now pick one of those molecules at an arbitrary time step. What are the costs (consider all necessary operations and use the $O()$-notation) for calculating the new position of this single molecule? Rely on the most efficient algorithm you know and justify your answer! (3 pts)

b) Discretisation (5 pts)

Assume that in a molecular dynamics program, the following discretisation scheme is used to calculate new positions for the molecules:

$$\vec{x}(t + \Delta t) = \vec{x}(t) + \Delta t \cdot \vec{v}(t) + \Delta t^2 \cdot \vec{a}(t) \tag{1}$$

(i) The scheme is missing a method for the calculation of the velocity. Construct a formula for $\vec{v}(t + \Delta t)$ in such a way that the discretisation scheme is time reversible. You have to prove the time reversibility for the position equation, but not for the velocity equation.

(ii) Is the discretisation scheme (1) a good discretisation scheme? Shortly Justify your answer!