TU-Chemnitz, Fakultät für Mathematik
Vorlesung: Prof. Dr. Martin Stoll **Übung: Dr. Roman Unger**
Homepage zur Übung: `https://www.tu-chemnitz.de/mathematik/wire/WS1819/nla.php`

7. Conjugate gradient method

1. Proof that: Let V be a vector space with inner product $(.,.)$, P a subspace of V, and $f \in V$. The solution $\tilde{p}$ of the approximation problem

$$\text{find } \tilde{p} \in P \text{ s.t. } \|f - \tilde{p}\| = \min_{p \in P} \|f - p\|$$

is defined by

$$(f - \tilde{p}, p) = 0 \quad \forall p \in P.$$

[Hint: Assume that $(f - \tilde{p}, p) = a \neq 0$ and define a q that gives a smaller norm, which is a contradiction.]

2. Implement the CG method and test it for the matrix `A=gallery('poisson',n)` for various n and a random right hand side! To illustrate the performance you may plot the norms of the residual $||r_k||_2$ over the iteration number k.

3. For the cg method complete the proof from the lectures and show that
   - $(r_{k+1}, p_j) = 0 \quad \forall j = 1, 2, \ldots, k-1.$
   - $(r_{k+1}, p_j) = (r_{k+1}, r_j) \quad \forall j = 1, 2, \ldots, k-1.$
   - $(Ap_{k+1}, p_j) = (r_{k+1}, r_j) \quad \forall j = 1, 2, \ldots, k-1.$

4. Proof that

$$\beta_k = \frac{-(Ar_{k+1}, p_k)}{(Ap_k, pk)} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}.$$

5. Proof that for the Chebyshev polynomials

$$\tau_0(t) = 1, \qquad \tau_1(t) = t, \qquad \tau_{k+1}(t) = 2t\tau_k(t) - \tau_{k-1}(t)$$

the following idendity holds

$$\tau_k(t) = \frac{1}{2}\left[\left(t + \sqrt{t^2 - 1}\right)^k + \left(t - \sqrt{t^2 - 1}\right)^k\right].$$

6. The following question is from *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain* of J.R. Shewchuk:

Suppose you wish to solve $Ax = b$ for a symmetric, positive-definite $N \times N$ matrix A. Unfortunately, the trauma of your linear algebra course has caused you to repress all memories of the Conjugate Gradient algorithm. Seeing you in distress, the Good Eigenfairy materializes and grants you a list of d distinct eigenvalues (but not the eigenvectors) of A. However, you do **not** know how many times each eigenvalue is repeated.

Clever person that you are, you mumbled the following algorithm in your sleep this morning:

> Choose an arbitrary starting point x_0;
> for $i = 0 : d - 1$
> $r_i = b - A * x_i$;
>   Remove an arbitrary eigenvalue from the list and call it λ_i;
> $x_{i+1} = x_i + \lambda_i^{-1} r_i$;
> end

No eigenvalue is used twice; on termination, the list is empty.

Show that upon termination of this algorithm, x_d is the solution to $Ax = b$.