5. Efficient solution of the linear systems arising from finite element discretization

5.1. **Optimization methods.** We have shown that the finite element discretization of Poisson's equation leads to the solution of a linear system $Ax = b$, in which A is a symmetric matrix. It is also easy to check that A is a positive definite matrix, i.e., $x^T Ax > 0$ for $x \neq 0$. For such a problem, the solution of this system is also the minimizer of the functional $\phi(x) = \frac{1}{2}x^T Ax - x^T b$. Note the minimum will occur where $\nabla\phi(x) = 0$. But $\nabla\phi(x) = Ax - b$, so the solution of the minimization problem is the solution of the linear system of equations.

A typical minimization algorithm is to let $\{p^k\}_{k\geq 0}$ be a set of search directions and $\{\alpha_k\}_{k\geq 0}$ a set of scalars and define an iteration

$$x^{k+1} = x^k + \alpha_k p^k.$$

The simplest example is the method of steepest descent, in which we choose

$$p^k = -\nabla\phi(x^k) = b - Ax^k.$$

To determine the best choice of α_k, we then minimize $\phi(x^k + \alpha_k p^k)$ with respect to α_k, considering x^k and p^k now fixed. Since

$$\phi(x^k + \alpha_k p^k) = \frac{1}{2}\left[(x^k)^T Ax^k + 2\alpha_k (p^k)^T Ax^k + \alpha_k^2 (p^k)^T Ap^k\right] - x^T b - \alpha_k p^T b,$$

minimizing with respect to α_k gives:

$$(p^k)^T Ax^k + \alpha_k (p^k)^T Ap^k - (p^k)^T b = 0,$$

i.e.,

$$\alpha_k = \frac{(p^k)^T (b - Ax^k)}{(p^k)^T Ap^k} = \frac{(p^k)^T p^k}{(p^k)^T Ap^k}.$$

Thus, the algorithm looks like:

`choose an initial iterate` x^0
`for` $k = 0, 1, \ldots,$
  `set` $p^k = b - Ax^k$
  `set` $\alpha_k = (p^k)^T p^k/(p^k)^T Ap^k$
  `set` $x^{k+1} = x^k + \alpha_k p^k$
`end`

Writing the iteration in this way, it appears we need two matrix-vector multiplications per iteration, one to compute Ax^k and one to compute Ap^k. We can reduce the work involved by defining $q^k = Ap^k$ and noticing that once we have computed q^k and α_k, we can compute the next residual p^{k+1} without an additional matrix-vector multiplication. Since $x^{k+1} = x^k + \alpha_k p^k$, we have $p^{k+1} = b - Ax^{k+1} = b - Ax^k - \alpha_k Ap^k = p^k - \alpha_k q^k$. Hence, we can write the algorithm as:

`choose an initial iterate` x^0
`Set` $p^0 = b - Ax^0$
`for` $k = 0, 1, \ldots,$
  `set` $q^k = Ap^k$
  `set` $\alpha_k = (p^k)^T p^k/(p^k)^T q^k$

```
   set $x^{k+1} = x^k + \alpha_k p^k$
   set $p^{k+1} = p^k - \alpha_k q^k$
end
```

To understand the convergence of such an algorithm, consider the simpler choice, $\alpha_k = \alpha$ for all k. Then we get the iteration

$$x^{k+1} = x^k - \alpha[Ax^k - b] = [I - \alpha A]x^k + \alpha b.$$

If we let x denote the exact solution of $Ax = b$, then we get the error equation

$$x - x^{k+1} = x - [I - \alpha A]x^k - \alpha b = [I - \alpha A](x - x^k) + \alpha Ax - \alpha b = [I - \alpha A](x - x^k).$$

Iterating this equation, we find that

$$x - x^k = [I - \alpha A]^k (x - x^0).$$

A well known result from linear algebra says that this iteration will converge for all $x^0 \in \mathbb{R}^n$ if and only if $\rho(I - \alpha A) < 1$, where if M is an $n \times n$ matrix with eigenvalues μ_i, then $\rho(M) = \max_i |\mu_i|$. Now if λ is an eigenvalue of A with eigenvector v, then $Av = \lambda v$ and so $(I - \alpha A)v = v - \alpha\lambda v = (1 - \alpha\lambda)v$. Hence $(1 - \alpha\lambda)$ is an eigenvalue of $I - \alpha A$ with eigenvector v. Hence, for convergence, we need $-1 < 1 - \alpha\lambda < 1$ for all eigenvalues λ of the matrix A. Since A is positive definite, all its eigenvalues are positive, so we require $0 < \alpha < 2/\lambda$ for all eigenvalues λ of A, i.e., $0 < \alpha < 2/\rho(A)$.

To determine the optimal choice of the parameter α, we proceed as follows. We first define the vector norm $\|x\|_2$ and the associated matrix norm $\|A\|_2$ by

$$\|x\|_2 = \Big(\sum_{i=1}^n x_i^2\Big)^{1/2}, \qquad \|A\|_2 = \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}.$$

It follows from the definition that $\|Ax\|_2 \leq \|A\|_2 \|x\|_2$ for all x. It can be shown that $\|A\|_2 = [\rho(A^*A)]^{1/2}$, where for an $n \times n$ matrix B, $\rho(B) = \max_i |\mu_i|$, where $\mu_1, \ldots, \mu_n$ are the eigenvalues of B, and $A^* = (\bar{A})^T$, where $\bar{A}$ is the complex conjugate of A. In particular, if A is real and symmetric, (the case we are considering), then $A^* = A$, so $A^*A = A^2$, and since the eigenvalues of A^2 are the squares of the eigenvalues of A, $\|A\|_2 = \rho(A) = \max_i |\lambda_i|$, where λ_i are the eigenvalues of A.

Since A is assumed real and symmetric, so is $I - \alpha A$. Hence,

$$\|I - \alpha A\|_2 = \rho(I - \alpha A) = \max_i |1 - \alpha\lambda_i|,$$

where λ_i are the eigenvalues of A. Since A is positive definite, we have that $0 < \lambda_1 \leq \ldots \leq \lambda_n$. Using the fact (easy to check) that $(1 - \alpha\lambda)^k$ is an eigenvalue of $(I - \alpha A)^k$ with eigenvector v, we get

$$\|x - x^k\|_2 = \|[I - \alpha A]^k (x - x^0)\|_2 \leq \|[I - \alpha A]^k\|_2 \|x - x^0\|_2 \leq \max_i |1 - \alpha\lambda_i|^k \|x - x^0\|_2.$$

To reduce the error at each iteration as much as possible, we would like to choose α to minimize the expression $\max_i |1 - \alpha\lambda_i|$. Observing that $\max_i |1 - \alpha\lambda_i| = \max\{|1 - \alpha\lambda_1|, |1 -$

$\alpha\lambda_n|\}$, we will minimize the desired expression by choosing α so that the two quantities are equal, i.e., $1-\alpha\lambda_1 = \alpha\lambda_n - 1$. Hence, the optimal value is $\alpha = 2/(\lambda_1+\lambda_n)$. In this case,

$$\rho(I-\alpha A) = 1 - \frac{2\lambda_1}{\lambda_1+\lambda_n} = \frac{\lambda_n-\lambda_1}{\lambda_n+\lambda_1} = \frac{(\lambda_n/\lambda_1)-1}{(\lambda_n/\lambda_1)+1}.$$

Let $\kappa = \|A\|_2\|A^{-1}\|_2$ be the condition number of A measured in the $\|\cdot\|_2$ norm. Since A is symmetric and positive definite, $\|A\|_2 = \rho(A) = \lambda_n$. Since the eigenvalues of A^{-1} are the reciprocals of the eigenvalues of A, $\|A^{-1}\|_2 = \rho(A^{-1}) = 1/\lambda_1$. Hence, $\kappa = \lambda_n/\lambda_1$. Thus, $\rho(I-\alpha A) = (\kappa-1)/(\kappa+1)$, and we have proved the following result.

Theorem 7. *If A is symmetric and positive definite, then the iteration scheme defined by $x^{k+1} = [I-\alpha A]x^k + \alpha b$, with $\alpha = 2/(\lambda_1+\lambda_n)$ satisfies:*

$$\|x-x^k\|_2 \le \left(\frac{\kappa-1}{\kappa+1}\right)^k \|x-x^0\|_2, \qquad \kappa = \lambda_{\max}(A)/\lambda_{\min}(A).$$

For the solution of Poisson's problem by standard finite elements, we can show that there is a constant independent of h such that $\kappa(A) \approx c^2h^{-2}$. Thus, implementing this iteration in its present form leads to a small reduction in error $(1-O(h^2))$ and slow convergence.

To get a more precise understanding of what the method is doing, we consider an eigenfunction expansion of the error, i.e., we suppose that $A\phi_i = \lambda_i\phi_i$, where $\{\phi_i\}_{i=1}^n$ are a set of orthonormal eigenvectors of A. We then set $e^k = x - x^k$ and write

$$e^0 = \sum_{i=1}^n [(e^0)^T\phi_i]\phi_i.$$

Suppose we choose $\alpha = 1/\lambda_n$, the largest eigenvalue of A. Then

$$e^k = [I-\alpha A]^k e^0 = \sum_{i=1}^n [(e^0)^T\phi_i](1-\lambda_i/\lambda_n)^k\phi_i.$$

Now for large eigenvalues $1-\lambda_i/\lambda_n$ is small, so the high frequency components of the error are damped out quickly, while for small eigenvalues $1-\lambda_i/\lambda_n \approx 1$, and there is not much decay in the error and so the low frequency components are not changed much. Thus, a few iterations of this method has the effect of "smoothing" the error. We shall come back to this idea in a later lecture.

In fact, the method of steepest descent has the same convergence rate as this simplified method, so we look for alternatives.

5.2. **Conjugate-Gradient method (CG).** A better choice of search directions $\{p^k\}$ is to choose them to be A-orthogonal, i.e, to satisfy $(p^j)^TAp^i = 0$ for $i \neq j$. In this case, the best choice of the α_k are given by

$$\alpha_k = \frac{(p^k)^T[b-Ax^k]}{(p^k)^TAp^k}.$$

The CG method generates the directions p^k recursively using the Gram-Schmidt orthogonalization process, but can be written in a simplified way (not obvious).

choose an initial iterate x^0
Set $p^0 = r^0 = b - Ax^0$
for $k = 0, 1, \ldots,$
  set $\alpha_k = (r^k)^T r^k / [(p^k)^T A p^k]$
  set $x^{k+1} = x^k + \alpha_k p^k$
  set $r^{k+1} = r^k - \alpha_k A p^k$
  set $p^{k+1} = r^{k+1} + \frac{r^{k+1})^T r^{k+1}}{(r^k)^T r^k} p^k$
end

If A is an $n \times n$ matrix, the CG method gives the exact solution in n iterations. However, it is most commonly used as an iterative method. If we stop after k iterations, we get the following error estimate:

$$\|x - x^k\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|x - x^0\|_A,$$

where $\|x\|_A^2 = x^T A x$. Since now $\sqrt{\kappa}$ enters, the reduction is like $1 - O(h)$, better than before, but still slow.

In practice, one uses the idea of preconditioning. Instead of solving the system $Ax = b$, we solve the system $B^{-1}Ax = B^{-1}b$, where B^{-1} is an approximation to A^{-1}, for which the linear system $Bz = c$ is easy to solve. Then the rate of convergence depends on the condition number of $B^{-1}A$ instead of A. If B^{-1} is a good approximation to A^{-1}, then $B^{-1}A \approx I$, and so $\kappa(B^{-1}A)$ will be close to 1, and we will get a substantial error reduction at each iteration.

One can show that the CG iteration for the linear system $B^{-1}Ax = B^{-1}b$ can be written in the following form.

choose an initial iterate x^0
Set $r^0 = b - Ax^0$, $p^0 = B^{-1} r^0$
for $k = 0, 1, \ldots,$
  set $\alpha_k = (r^k)^T B^{-1} r^k / [(p^k)^T A p^k]$
  set $x^{k+1} = x^k + \alpha_k p^k$
  set $r^{k+1} = r^k - \alpha_k A p^k$
  set $p^{k+1} = B^{-1} r^{k+1} + \frac{(r^{k+1})^T B^{-1} r^{k+1}}{(r^k)^T B^{-1} r^k} p^k$
end

Hence, we need to compute $z^k \equiv B^{-1} r^k$ at each iteration (which we do by solving the system $Bz^k = r^k$). If this can be done quickly, the work involved will be essentially the same as for the CG method applied to the system $Ax = b$.

Some common choices for the matrix B are the diagonal of A, a banded piece of A, an incomplete factorization of A, domain decomposition methods, and multigrid methods. Multigrid is one of the most effective and we shall treat this next.