

Math 6610: Analysis of Numerical Methods, I

Iterative methods for linear systems

Department of Mathematics, University of Utah

Fall 2025

Resources: Trefethen and Bau 1997, Lectures 32, 35, 38
Atkinson 1989, Section 8.6
Salgado and Wise 2022, Sections 6.1-6.3

Direct methods

We have previously discussed *direct* methods, i.e., methods that

- orthogonalize vectors
- solve linear systems
- compute eigenvalues

assuming that operations like $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$ are efficiently computable.

Direct methods

We have previously discussed *direct* methods, i.e., methods that

- orthogonalize vectors
- solve linear systems
- compute eigenvalues

assuming that operations like $\mathbf{x} \mapsto \mathbf{A}\mathbf{x}$ are efficiently computable.

For very large matrices, e.g., $\mathbf{A} \in \mathbb{C}^{10^6 \times 10^6}$, such procedures are not practical.

For such large matrices of general type, there is not much that can be done.

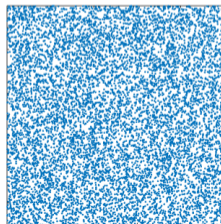
But in practice, matrices are *sparse*, having a reasonably small percentage of nonzero entries.



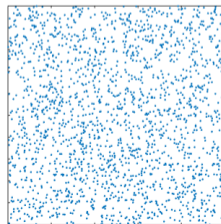
Dense matrix



10% density



1% density



0.2% density

Sparse matrices arise frequently in applications.

This is fortunate since fundamental operations like $x \mapsto Ax$ are very efficient for sparse matrices.

Unfortunately, direct methods operating on sparse matrix frequently result in dense matrices:

- QR factorizations
- LU factorizations
- Eigenvalue algorithms (e.g., for computing eigenvectors)

Even just storing such matrices can be impossible in practice.

In contrast to direct methods, iterative methods produce answers that gradually approach the solution, performing only operations that generally don't require large dense matrices.

There are two major problems that iterative methods often attempt to address:

- Solve linear systems
- Compute eigenvalues/eigenvectors

We'll briefly discuss the general ideas for the first class of problems.

$$\mathbf{Ax} = \mathbf{b}$$

Many iterative methods focus on ensuring the following properties

- A sequence of solution approximations, $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \dots$ is constructed, with a new approximation formed at every iteration.
- The update $\mathbf{x}_k \mapsto \mathbf{x}_{k+1}$ typically involves only efficient matrix-vector multiplications (e.g., exploiting sparsity)
- The sequence $\{\mathbf{x}_k\}$ gradually approaches the solution as $k \uparrow \infty$.
- Termination is frequently judged by inspecting the *residual* $\|\mathbf{Ax}_k - \mathbf{b}\|$.

$$Ax = b$$

Many iterative methods focus on ensuring the following properties

- A sequence of solution approximations, x_0, x_1, x_2, \dots is constructed, with a new approximation formed at every iteration.
- The update $x_k \mapsto x_{k+1}$ typically involves only efficient matrix-vector multiplications (e.g., exploiting sparsity)
- The sequence $\{x_k\}$ gradually approaches the solution as $k \uparrow \infty$.
- Termination is frequently judged by inspecting the *residual* $\|Ax_k - b\|$.

There are two classes of iterative methods we'll discuss:

- Stationary, fixed-point, or relaxation methods
- Krylov subspace methods

$$Ax = b$$

Stationary iterative methods (also "relaxation methods") are among the first types of iterative methods developed.

The basic idea of stationary iterative methods is as follows: consider an additive decomposition of A :

$$A = B + C.$$

$$Ax = b$$

Stationary iterative methods (also “relaxation methods”) are among the first types of iterative methods developed.

The basic idea of stationary iterative methods is as follows: consider an additive decomposition of A :

$$A = B + C.$$

The assumption is that A is difficult to invert (if not, just solve the problem directly).

However, both B and C can be freely chosen. We write,

$$Ax = b \implies Bx = b - Cx.$$

And if we choose B to be “easily” invertible, then

$$x = A^{-1}b \iff x = B^{-1}b - B^{-1}Cx$$

This forms the basis for an iterative scheme.

Since,

$$Bx = b - Cx,$$

we propose the iterative scheme,

$$Bx_{k+1} = b - Cx_k, \quad k \geq 0,$$

where x_0 is freely chosen.

Since,

$$Bx = b - Cx,$$

we propose the iterative scheme,

$$Bx_{k+1} = b - Cx_k, \quad k \geq 0,$$

where x_0 is freely chosen.

This scheme is feasible if we choose B well so that iterating the above several times is computationally tractable.

Of course, one suspects that choosing a “good” x_0 is at least computationally advantageous.

One practical metric we have for stopping is the *residual*, i.e.,

$$r_k = b - Ax_k,$$

Another is the difference in iterates, $\|x_{k+1} - x_k\|_2$.

What can we say about iterative methods governed by *general* affine (linear + shift) mappings?

A general affine mapping between \mathbf{x}_k and \mathbf{x}_{k+1} has the form,

$$\mathbf{B}\mathbf{x}_{k+1} = \mathbf{d} - \mathbf{C}\mathbf{x}_k, \quad \mathbf{B}, \mathbf{C} \in \mathbb{C}^{n \times n}, \quad \mathbf{d} \in \mathbb{C}^n,$$

What can we say about iterative methods governed by *general* affine (linear + shift) mappings?

A general affine mapping between \mathbf{x}_k and \mathbf{x}_{k+1} has the form,

$$\mathbf{B}\mathbf{x}_{k+1} = \mathbf{d} - \mathbf{C}\mathbf{x}_k, \quad \mathbf{B}, \mathbf{C} \in \mathbb{C}^{n \times n}, \quad \mathbf{d} \in \mathbb{C}^n,$$

In order for the sequence \mathbf{x}_k to converge to the vector \mathbf{x} defined by $\mathbf{A}\mathbf{x} = \mathbf{b}$, we at least require consistency, i.e.,

$$\mathbf{C} + \mathbf{B} = \mathbf{A}, \quad \mathbf{d} = \mathbf{b},$$

and hence our previous characterization of stationary iterative methods (requiring $\mathbf{A} = \mathbf{B} + \mathbf{C}$) is comprehensive.

What can we say about iterative methods governed by *general* affine (linear + shift) mappings?

A general affine mapping between \mathbf{x}_k and \mathbf{x}_{k+1} has the form,

$$\mathbf{B}\mathbf{x}_{k+1} = \mathbf{d} - \mathbf{C}\mathbf{x}_k, \quad \mathbf{B}, \mathbf{C} \in \mathbb{C}^{n \times n}, \quad \mathbf{d} \in \mathbb{C}^n,$$

In order for the sequence \mathbf{x}_k to converge to the vector \mathbf{x} defined by $\mathbf{A}\mathbf{x} = \mathbf{b}$, we at least require consistency, i.e.,

$$\mathbf{C} + \mathbf{B} = \mathbf{A}, \quad \mathbf{d} = \mathbf{b},$$

and hence our previous characterization of stationary iterative methods (requiring $\mathbf{A} = \mathbf{B} + \mathbf{C}$) is comprehensive.

One tool to help understand convergence is to scrutinize the *error*:

$$\mathbf{e}_k := \mathbf{x} - \mathbf{x}_k$$

Using the form of our iterative scheme, some direct computations reveal:

$$\mathbf{A}\mathbf{e}_k = \mathbf{r}_k, \quad \mathbf{e}_{k+1} = -\mathbf{B}^{-1}\mathbf{C}\mathbf{e}_k,$$

What can we say about iterative methods governed by *general* affine (linear + shift) mappings?

A general affine mapping between \mathbf{x}_k and \mathbf{x}_{k+1} has the form,

$$\mathbf{B}\mathbf{x}_{k+1} = \mathbf{d} - \mathbf{C}\mathbf{x}_k, \quad \mathbf{B}, \mathbf{C} \in \mathbb{C}^{n \times n}, \quad \mathbf{d} \in \mathbb{C}^n,$$

In order for the sequence \mathbf{x}_k to converge to the vector \mathbf{x} defined by $\mathbf{A}\mathbf{x} = \mathbf{b}$, we at least require consistency, i.e.,

$$\mathbf{C} + \mathbf{B} = \mathbf{A}, \quad \mathbf{d} = \mathbf{b},$$

and hence our previous characterization of stationary iterative methods (requiring $\mathbf{A} = \mathbf{B} + \mathbf{C}$) is comprehensive.

One tool to help understand convergence is to scrutinize the *error*:

$$\mathbf{e}_k := \mathbf{x} - \mathbf{x}_k$$

Using the form of our iterative scheme, some direct computations reveal:

$$\mathbf{A}\mathbf{e}_k = \mathbf{r}_k, \quad \mathbf{e}_{k+1} = -\mathbf{B}^{-1}\mathbf{C}\mathbf{e}_k,$$

This suggests that one needs $(\mathbf{B}^{-1}\mathbf{C})^k \rightarrow \mathbf{0}$.

Using all this information, we can very precisely characterize when stationary iterative methods converge.

Theorem

Suppose $A = B + C \in \mathbb{C}^{n \times n}$ with A and B invertible. Let $\mathbf{b} \in \mathbb{C}^n$ be given, and let $\mathbf{x} := A^{-1}\mathbf{b}$. Then: the iterative scheme with an arbitrary starting value $\mathbf{x}_0 \in \mathbb{C}^n$ defined by,

$$B\mathbf{x}_{k+1} = \mathbf{b} - C\mathbf{x}_k, \quad k \geq 0,$$

satisfies $\lim_{k \rightarrow \infty} \mathbf{x}_k = \mathbf{x}$ iff $\rho(B^{-1}C) < 1$.

This completely characterizes the asymptotic performance of stationary iterative methods.

$$Bx_{k+1} = b - Cx_k,$$

$$A = B + C.$$

Suppose that $A = L + D + U$, where

- L and U are the (strictly) lower- and upper-triangular portions of A , respectively,
- D is the diagonal portion of A .

$$Bx_{k+1} = b - Cx_k,$$

$$A = B + C.$$

Suppose that $A = L + D + U$, where

- L and U are the (strictly) lower- and upper-triangular portions of A , respectively,
- D is the diagonal portion of A .

Many methods boil down to how B and C are chosen:

- Jacobi method: $B = D$, $C = L + U$.
- Gauss-Seidel method: $B = D + L$, $C = U$.
- Successive over-relaxation method: $B = \frac{1}{\alpha}D + L$, $C = \frac{(\alpha-1)}{\alpha}D + U$.

These methods have strong theory when applied to discretizations of Laplace's equation.

The second large class of methods are *Krylov subspace methods*.

First a definition: Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ and $\mathbf{b} \in \mathbb{C}^n$ be given. The **Krylov subspace** of order s , $K_s(\mathbf{A}, \mathbf{b})$ is defined as,

$$K_s(\mathbf{A}, \mathbf{b}) := \text{span}\{\mathbf{b}, \mathbf{A}\mathbf{b}, \mathbf{A}^2\mathbf{b}, \dots, \mathbf{A}^{s-1}\mathbf{b}\}.$$

The dimension of this subspace can be as large as s , but can be smaller.

The dimension of this subspace can be as large as n , but no larger.

The second large class of methods are *Krylov subspace methods*.

First a definition: Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ and $\mathbf{b} \in \mathbb{C}^n$ be given. The **Krylov subspace** of order s , $K_s(\mathbf{A}, \mathbf{b})$ is defined as,

$$K_s(\mathbf{A}, \mathbf{b}) := \text{span}\{\mathbf{b}, \mathbf{A}\mathbf{b}, \mathbf{A}^2\mathbf{b}, \dots, \mathbf{A}^{s-1}\mathbf{b}\}.$$

The dimension of this subspace can be as large as s , but can be smaller.

The dimension of this subspace can be as large as n , but no larger.

The class of Krylov subspace methods are iterative, and for a given order k and initial guess \mathbf{x}_0 , implicitly solve optimization problems of the form,

$$\min_{\mathbf{y} \in \mathbf{x}_0 + K_s(\mathbf{A}, \mathbf{b})} f(\mathbf{y}, \mathbf{A}, \mathbf{b}).$$

While they solve these types of problems, the goal is typically to write algorithms that don't explicitly appeal to this optimization problem directly.

Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ be Hermitian and positive semi-definite, and let $\mathbf{b}, \mathbf{x}_0 \in \mathbb{C}^n$. The Conjugate Gradient (CG) method forms a sequence of iterates given by,

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} (\mathbf{y} - \mathbf{x})^* \mathbf{A}(\mathbf{y} - \mathbf{x}),$$

where \mathbf{x} is the exact solution satisfying $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ be Hermitian and positive semi-definite, and let $\mathbf{b}, \mathbf{x}_0 \in \mathbb{C}^n$. The Conjugate Gradient (CG) method forms a sequence of iterates given by,

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} (\mathbf{y} - \mathbf{x})^* \mathbf{A}(\mathbf{y} - \mathbf{x}),$$

where \mathbf{x} is the exact solution satisfying $\mathbf{A}\mathbf{x} = \mathbf{b}$.

As usual, the optimization problem isn't solved directly. Instead, one iteratively computes,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \alpha_k \in \mathbb{R}, \quad \mathbf{p}_k \in \mathbb{C}^n,$$

where α_k and \mathbf{p}_k must be chosen.

Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ be Hermitian and positive semi-definite, and let $\mathbf{b}, \mathbf{x}_0 \in \mathbb{C}^n$. The Conjugate Gradient (CG) method forms a sequence of iterates given by,

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} (\mathbf{y} - \mathbf{x})^* \mathbf{A}(\mathbf{y} - \mathbf{x}),$$

where \mathbf{x} is the exact solution satisfying $\mathbf{A}\mathbf{x} = \mathbf{b}$.

As usual, the optimization problem isn't solved directly. Instead, one iteratively computes,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \alpha_k \in \mathbb{R}, \quad \mathbf{p}_k \in \mathbb{C}^n,$$

where α_k and \mathbf{p}_k must be chosen.

Note that $\mathbf{x}_k \in \mathbf{x}_0 + K_{k+1}(\mathbf{A}, \mathbf{b})$, and we require $\mathbf{x}_{k+1} \in \mathbf{x}_0 + K_{k+1}(\mathbf{A}, \mathbf{b})$.

Therefore, the new direction \mathbf{p}_k should be chosen as something like $\mathbf{A}^k \mathbf{b}$, and for stability should be orthogonal to \mathbf{p}_j , $j < k$.

However, CG computes \mathbf{p}_k using two clever realizations.

- Our minimization problem is a quadratic form involving the Hermitian semi-definite matrix \mathbf{A} . Therefore, “orthogonal” should be with respect to the quadratic form defined by \mathbf{A} .
- Since $\mathbf{x}_k \in K_k(\mathbf{A}, \mathbf{b})$, then $\mathbf{Ax}_k \in K_{k+1}(\mathbf{A}, \mathbf{b})$, and better yet the residual $\mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k \in K_{k+1}(\mathbf{A}, \mathbf{b})$.

However, CG computes \mathbf{p}_k using two clever realizations.

- Our minimization problem is a quadratic form involving the Hermitian semi-definite matrix \mathbf{A} . Therefore, “orthogonal” should be with respect to the quadratic form defined by \mathbf{A} .
- Since $\mathbf{x}_k \in K_k(\mathbf{A}, \mathbf{b})$, then $\mathbf{A}\mathbf{x}_k \in K_{k+1}(\mathbf{A}, \mathbf{b})$, and better yet the residual $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k \in K_{k+1}(\mathbf{A}, \mathbf{b})$.

Using these properties, at each iteration, CG:

- Given $\mathbf{x}_k, \mathbf{p}_k$, computes α_k by minimizing the quadratic objective on the line $\mathbf{x}_k + \alpha_k \mathbf{p}_k$ (this can be done explicitly and easily)
- Computes $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.
- Computes $\mathbf{r}_{k+1} = \mathbf{b} - \mathbf{A}(\mathbf{x}_k + \alpha_k \mathbf{p}_k) = \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k$.
- Computes $\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$. (This implicitly enforces \mathbf{A} -orthogonality.)

There are numerous other Krylov subspace algorithms. Two other popular ones are:

- The Minimum Residual (MINRES) method. If \mathbf{A} is Hermitian (not necessarily definite), we solve the problem,

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$$

Like CG, steps are explicit and involve updates of the form $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.

There are numerous other Krylov subspace algorithms. Two other popular ones are:

- The Minimum Residual (MINRES) method. If \mathbf{A} is Hermitian (not necessarily definite), we solve the problem,

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$$

Like CG, steps are explicit and involve updates of the form $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.

- The Generalized Minimum Residual (GMRES) method can be used for arbitrary invertible matrices $\mathbf{A} \in \mathbb{C}^{n \times n}$. It again solves

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$$

There are numerous other Krylov subspace algorithms. Two other popular ones are:

- The Minimum Residual (MINRES) method. If \mathbf{A} is Hermitian (not necessarily definite), we solve the problem,




$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$$

Like CG, steps are explicit and involve updates of the form $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.

- The Generalized Minimum Residual (GMRES) method can be used for arbitrary invertible matrices $\mathbf{A} \in \mathbb{C}^{n \times n}$. It again solves

$$\mathbf{x}_k = \min_{\mathbf{y} \in \mathbf{x}_0 + K_k(\mathbf{A}, \mathbf{b})} \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$$

- For both algorithms (and most Krylov subspace methods) a common convergence criterion is the size of the residual (which is automatically computed during iteration).
- All these algorithms can suffer numerical difficulties if \mathbf{A} is ill-conditioned. In such cases a *preconditioner* \mathbf{M} is devised as an easily invertible matrix such that, e.g., $\mathbf{M}^{-1}\mathbf{A}$ has better conditioning. Then one solves, e.g., $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$.

-  Atkinson, Kendall (1989). *An Introduction to Numerical Analysis*. New York: Wiley. ISBN: 978-0-471-62489-9.
-  Salgado, Abner J. and Steven M. Wise (2022). *Classical Numerical Analysis: A Comprehensive Course*. Cambridge: Cambridge University Press. ISBN: 978-1-108-83770-5. DOI: 10.1017/9781108942607.
-  Trefethen, Lloyd N. and David Bau (1997). *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics. ISBN: 0-89871-361-7.