

CSE203B Convex Optimization

Lecture 4: Matrix Computations

Ying Yuan

Dept. of Computer Science and Engineering
University of California, San Diego

Outline

- Introduction
- Key Concepts
 - Error and Residual
 - Condition Number
 - Machine Precision
- Iterative Methods
 - Categories
 - Conjugate Gradient
 - Preconditioning
 - GMRES

Introduction: Direct vs. Iterative Methods

- Direct Methods

- Gaussian Elimination
- LU Decomposition

- Iterative Methods

- Stationary Methods
- Krylov Subspace Methods
- Multigrid Methods
- Preconditioned Iterations
- Domain Decomposition Methods

Introduction: Direct vs. Iterative Methods

- Direct Methods

- Gaussian Elimination
- LU Decomposition

General and Robust but can be complicated if $N \geq 1M$

- Iterative Methods

- Stationary Methods
- Krylov Subspace Methods
- Multigrid Methods
- Preconditioned Iterations
- Domain Decomposition Methods

Excellent choice for SPD matrices
Remain an art for arbitrary matrices

Outline

- Introduction
- **Key Concepts**
 - Error and Residual
 - Condition Number
 - Machine Precision
- Iterative Methods
 - Categories
 - Conjugate Gradient
 - Preconditioning
 - GMRES

Key Concepts: Error and Residual

Given $A\mathbf{x} = \mathbf{b}$

- Exact Solution vs. Approximate Solution

- Exact Solution: \mathbf{x}
- Approximation: $\hat{\mathbf{x}} = \mathbf{x} + \Delta\mathbf{x}$

- Error vs. Residual

- Error: Measures how far the approximation is from the true solution.

$$\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x}$$

- Residual: Measures how well the approximation satisfies the original system.

$$\mathbf{r} = \mathbf{b} - A\hat{\mathbf{x}}$$

- Relative Error vs. Relative Residual

- Measures how large that error is compared to the true solution.

- Relative Error: $\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|}{\|\mathbf{x}\|}$

- Relative Residual: $\frac{\|\mathbf{b} - A\hat{\mathbf{x}}\|}{\|\mathbf{b}\|}$

Key Concepts: Error and Residual

Given $A\mathbf{x} = \mathbf{b}$

- Exact Solution vs. Approximate Solution

- Exact Solution: \mathbf{x}
- Approximation: $\hat{\mathbf{x}} = \mathbf{x} + \Delta\mathbf{x}$

Q: How sensitive is \mathbf{x} to small changes in \mathbf{b} ?

- Error vs. Residual

- Error: Measures how far the approximation is from the true solution.

$$\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x}$$

- Residual: Measures how well the approximation satisfies the original system.

$$\mathbf{r} = \mathbf{b} - A\hat{\mathbf{x}}$$

- Relative Error vs. Relative Residual

- Measures how large that error is compared to the true solution.

- Relative Error: $\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|}{\|\mathbf{x}\|}$

- Relative Residual: $\frac{\|\mathbf{b} - A\hat{\mathbf{x}}\|}{\|\mathbf{b}\|}$

Key Concepts: Condition Number

If \mathbf{b} changes by a small amount $\Delta\mathbf{b}$, how much could \mathbf{x} change, i.e., how large is $\Delta\mathbf{x}$?

- Given $\mathbf{A}\mathbf{x} = \mathbf{b}$ as a non-singular system of linear equations, we have

$$\mathbf{A}\hat{\mathbf{x}} = \hat{\mathbf{b}}$$

$$\mathbf{A}(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{b} + \Delta\mathbf{b}$$

$$\mathbf{A}\Delta\mathbf{x} = \Delta\mathbf{b}$$

$$\Delta\mathbf{x} = \mathbf{A}^{-1}\Delta\mathbf{b}$$

Key Concepts: Condition Number

If \mathbf{b} changes by a small amount $\Delta\mathbf{b}$, how much could \mathbf{x} change, i.e., how large is $\Delta\mathbf{x}$?

- To measure the change, we look at *relative* changes

$$\begin{aligned} \frac{\text{Relative Error in } \mathbf{x}}{\text{Relative Error in } \mathbf{b}} &= \frac{\|\hat{\mathbf{x}} - \mathbf{x}\| / \|\mathbf{x}\|}{\|\hat{\mathbf{b}} - \mathbf{b}\| / \|\mathbf{b}\|} \\ &= \frac{\|\Delta\mathbf{x}\| / \|\mathbf{x}\|}{\|\Delta\mathbf{b}\| / \|\mathbf{b}\|} \\ &= \frac{\|\Delta\mathbf{x}\| \|\mathbf{b}\|}{\|\Delta\mathbf{b}\| \|\mathbf{x}\|} \\ &= \frac{\|\mathbf{A}^{-1}\Delta\mathbf{b}\| \|\mathbf{Ax}\|}{\|\Delta\mathbf{b}\| \|\mathbf{x}\|} \leq \frac{\|\mathbf{A}^{-1}\| \|\Delta\mathbf{b}\| \|\mathbf{A}\| \|\mathbf{x}\|}{\|\Delta\mathbf{b}\| \|\mathbf{x}\|} = \|\mathbf{A}\| \|\mathbf{A}^{-1}\| \end{aligned}$$

Key Concepts: Condition Number

If \mathbf{b} changes by a small amount $\Delta\mathbf{b}$, how much could \mathbf{x} change, i.e., how large is $\Delta\mathbf{x}$?

- To measure the change, we look at *relative* changes

$$\frac{\|\Delta\mathbf{x}\|/\|\mathbf{x}\|}{\|\Delta\mathbf{b}\|/\|\mathbf{b}\|}$$

\leq

$$\|\mathbf{A}\| \|\mathbf{A}^{-1}\|$$

Key Concepts: Condition Number

If \mathbf{b} changes by a small amount $\Delta\mathbf{b}$, how much could \mathbf{x} change, i.e., how large is $\Delta\mathbf{x}$?

- To measure the change, we look at *relative* changes

$$\frac{\|\Delta\mathbf{x}\|/\|\mathbf{x}\|}{\|\Delta\mathbf{b}\|/\|\mathbf{b}\|} \leq \|\mathbf{A}\| \|\mathbf{A}^{-1}\|$$
$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \|\mathbf{A}\| \|\mathbf{A}^{-1}\| \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}$$

- We define the **condition number** of \mathbf{A} as

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^{-1}\|$$

- Thus, we have

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}$$

Key Concepts: Condition Number

In a similar way, we can also know how sensitive the \mathbf{x} is to a perturbation to the matrix \mathbf{A} .

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \frac{\|\Delta \mathbf{A}\|}{\|\mathbf{A}\|}$$

- Then, the error $\|\Delta \mathbf{x}\|/\|\mathbf{x}\|$ of the solution \mathbf{x} is in proportion to the error of both \mathbf{A} and \mathbf{b} by a factor of the condition number.

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \left(\frac{\|\Delta \mathbf{b}\|}{\|\mathbf{b}\|} + \frac{\|\Delta \mathbf{A}\|}{\|\mathbf{A}\|} \right)$$

Key Concepts: Condition Number

- Recall Matrix Norms

- Matrix 1-norm $\|\mathbf{A}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^m |a_{ij}|$ Maximum absolute column sum of the matrix \mathbf{A}

- Matrix ∞ -norm $\|\mathbf{A}\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^n |a_{ij}|$ Maximum absolute row sum of the matrix \mathbf{A}

- Matrix 2-norm $\|\mathbf{A}\|_2 = (\text{largest eigenvalue of } \mathbf{A}^\top \mathbf{A})^{1/2}$
(Spectral Norm)

Key Concepts: Condition Number - Summary

- Definition

- The *condition number* of matrix \mathbf{A} :

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2$$

- Interpretation

- Measures *sensitivity* of the solution \mathbf{x} to small changes in \mathbf{A} and \mathbf{b} .
- A large $\kappa(\mathbf{A}) \gg 1$ means the matrix is *ill-conditioned*, so small perturbations in \mathbf{b} can lead to large changes in \mathbf{x} .

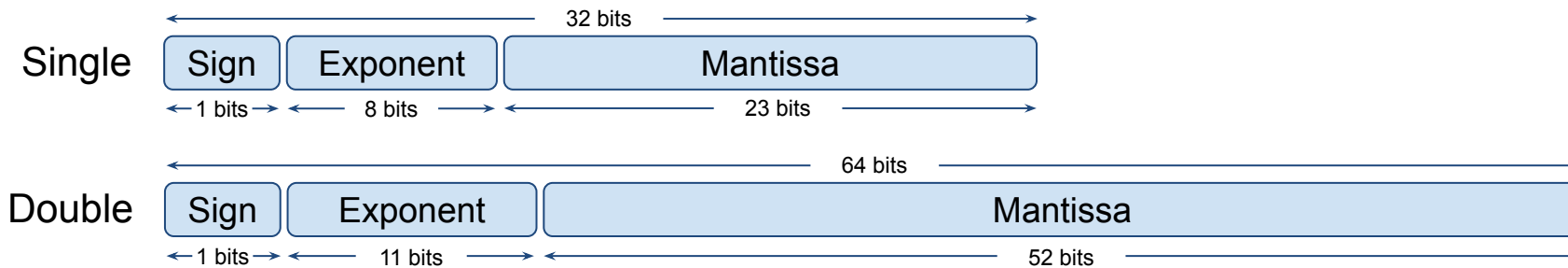
Key Concepts: Condition Number

- Example
 - Use direct method, LU decomposition, to solve $A\mathbf{x} = \mathbf{b}$.
 - A is a Hilbert matrix of size $n \times n$.

| Dimension n | $\kappa(A)$ | Residual | Relative Residual |
|---------------|-------------|------------|-------------------|
| 5 | 4.77e+05 | 2.7195e-16 | 8.6655e-17 |
| 10 | 1.60e+13 | 1.0533e-15 | 2.2363e-16 |
| 15 | 6.50e+17 | 1.0236e-15 | 1.7350e-16 |
| 20 | 2.54e+18 | 9.9809e-15 | 1.4475e-15 |

Key Concepts: Machine Precision

- Floating-Point Representation (IEEE 754 Standard)



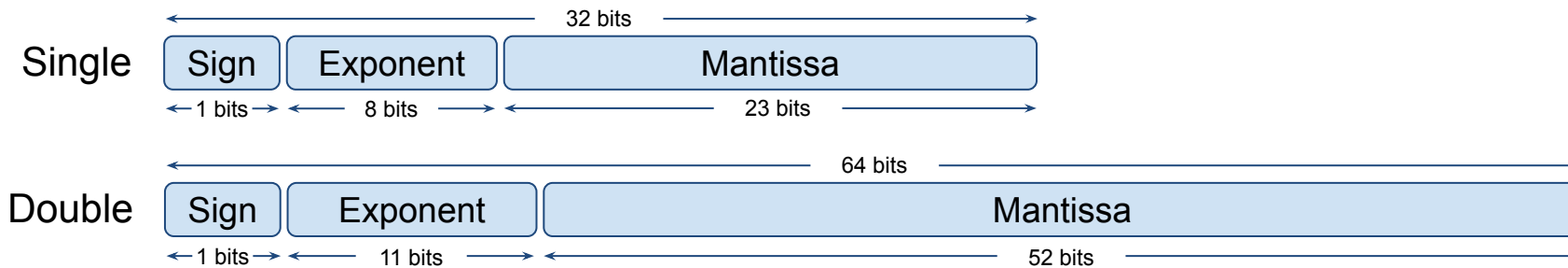
- Machine Precision

- Machine epsilon ϵ is the upper bound on relative error due to rounding in floating-point arithmetic. It is effectively the smallest number such that $1.0 + \epsilon \neq 1.0$ in the floating-point system.

- 32-bit (single) precision $\epsilon_{32\text{-bit}} \approx 10^{-7}$
- 64-bit (double) precision $\epsilon_{64\text{-bit}} \approx 10^{-16}$

Key Concepts: Machine Precision

- Floating-Point Representation (IEEE 754 Standard)



- Machine Precision

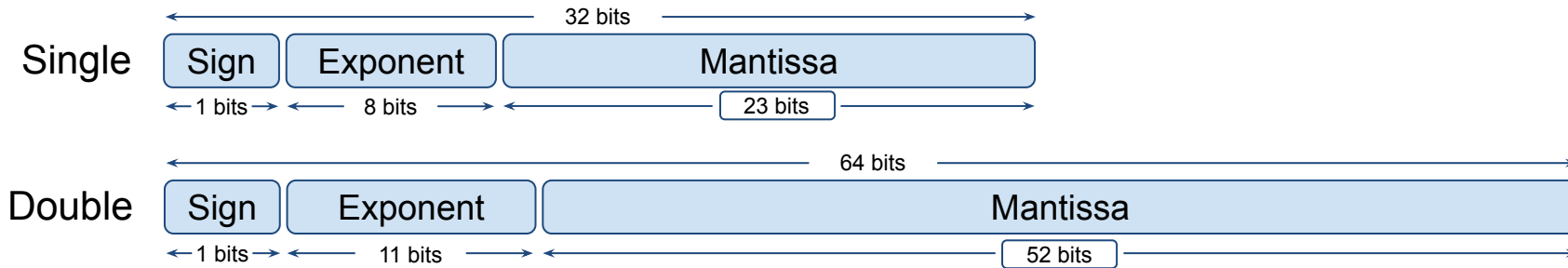
- Machine epsilon ϵ is the upper bound on relative error due to rounding in floating-point arithmetic. It is effectively the smallest number such that $1.0 + \epsilon \neq 1.0$ in the floating-point system.

- 32-bit (single) precision $\epsilon_{32\text{-bit}} \approx 10^{-7}$
- 64-bit (double) precision $\epsilon_{64\text{-bit}} \approx 10^{-16}$

Every arithmetic operation can introduce tiny errors that accumulate.

Key Concepts: Machine Precision

- Floating-Point Representation (IEEE 754 Standard)



- Machine Precision

- Machine epsilon ϵ is the upper bound on relative error due to rounding in floating-point arithmetic. It is effectively the smallest number such that $1.0 + \epsilon \neq 1.0$ in the floating-point system.

Every arithmetic operation can introduce tiny errors that accumulate.

- 32-bit (single) precision $\epsilon_{32\text{-bit}} \approx 10^{-7}$
- 64-bit (double) precision $\epsilon_{64\text{-bit}} \approx 10^{-16}$

```
np.info(np.float32).eps
```

Python

$2^{-23} = 1.1920929e-07$

```
np.info(np.float64).eps
```

✓ 0.0s Python

$2^{-52} = 2.220446049250313e-16$

Key Concepts: Machine Precision

- Machine Precision
 - Machine epsilon ϵ is the upper bound on relative error due to rounding in floating-point arithmetic. It is effectively the smallest number such that $1.0 + \epsilon \neq 1.0$ in the floating-point system.
 - 32-bit (single) precision $\epsilon_{32\text{-bit}} \approx 10^{-7}$ • 64-bit (double) precision $\epsilon_{64\text{-bit}} \approx 10^{-16}$
- Why it's important
 - **Rounding Errors:** Every arithmetic operation can introduce tiny errors that accumulate.
 - **Ill-Conditioned Problems:** If $\kappa(\mathbf{A})$ or function is large, even tiny floating-point errors can lead to big inaccuracies in the final result.

Key Concepts: Machine Precision

- Machine Precision
 - Machine epsilon ϵ is the upper bound on relative error due to rounding in floating-point arithmetic. It is effectively the smallest number such that $1.0 + \epsilon \neq 1.0$ in the floating-point system.
 - 32-bit (single) precision $\epsilon_{32\text{-bit}} \approx 10^{-7}$ • 64-bit (double) precision $\epsilon_{64\text{-bit}} \approx 10^{-16}$
- Why it's important
 - **Rounding Errors:** Every arithmetic operation can introduce tiny errors that accumulate. Thus, remember that zero residual is rare.
 - **Ill-Conditioned Problems:** If $\kappa(\mathbf{A})$ or function is large, even tiny floating-point errors can lead to big inaccuracies in the final result.

Key Concepts: Machine Precision

- Example
 - Use direct method, LU decomposition, to solve $A\mathbf{x} = \mathbf{b}$.
 - A is a Hilbert matrix of size $n \times n$, where $n = 12$.

| Precision | Time(s) | Residual Norm | Relative Residual |
|-----------|----------|---------------|-------------------|
| Single | 0.000446 | 1.046057e-06 | 2.005303e-07 |
| Double | 0.000196 | 9.019494e-16 | 1.729047e-16 |

Key Concepts: Example

- Recall the example
 - Use direct method, LU decomposition, to solve $A\mathbf{x} = \mathbf{b}$.
 - A is a Hilbert matrix of size $n \times n$.

| Dimension n | $\kappa(A)$ | Residual | Relative Residual |
|---------------|-------------|------------|-------------------|
| 5 | 4.77e+05 | 2.7195e-16 | 8.6655e-17 |
| 10 | 1.60e+13 | 1.0533e-15 | 2.2363e-16 |
| 15 | 6.50e+17 | 1.0236e-15 | 1.7350e-16 |
| 20 | 2.54e+18 | 9.9809e-15 | 1.4475e-15 |

Key Concepts: Example

- Recall the example
 - Use direct method, LU decomposition, to solve $A\mathbf{x} = \mathbf{b}$.
 - A is a Hilbert matrix of size $n \times n$.

What if we continue increasing dimension?

| Dimension n | $\kappa(A)$ | Residual | Relative Residual |
|---------------|-------------|------------|-------------------|
| 5 | 4.77e+05 | 2.7195e-16 | 8.6655e-17 |
| 10 | 1.60e+13 | 1.0533e-15 | 2.2363e-16 |
| 15 | 6.50e+17 | 1.0236e-15 | 1.7350e-16 |
| 20 | 2.54e+18 | 9.9809e-15 | 1.4475e-15 |

Key Concepts: Example

- Recall the example
 - Use direct method, LU decomposition, to solve $A\mathbf{x} = \mathbf{b}$.
 - A is a Hilbert matrix of size $n \times n$.

What if we continue increasing dimension?

| Dimension n | $\kappa(A)$ | Relative Residual | Peak Mem (MB) | Time (s) |
|---------------|-------------|-------------------|---------------|-----------|
| 1e+01 | 2e+13 | 2.236276e-16 | 0.000000 | 0.000619 |
| 1e+02 | 3e+19 | 3.752015e-14 | 180.179688 | 0.104877 |
| 1e+03 | 2e+21 | 2.475700e-13 | 232.664062 | 0.154595 |
| 1e+04 | 2e+22 | 4.522638e-13 | 1824.644531 | 1.985986 |
| 2e+04 | 2e+23 | 2.793520e-12 | 6403.535156 | 16.786994 |
| 3e+04 | 1e+23 | 3.058805e-12 | 14033.855469 | 55.393063 |

Key Concepts: Example

- Example: Direct vs. Iterative
 - Try using Conjugate Gradient to solve $Ax = b$.

Conjugate Gradient
uses less memory
and CPU time.

| Dimension n | kappa(A) | Relative Residual | Peak Mem (MB) | Time (s) |
|-------------|----------|-------------------|---------------|----------|
| 1e+01 | 2e+13 | 6.877954e-09 | 0.000000 | 0.000494 |
| 1e+02 | 3e+19 | 5.845504e-10 | 178.015625 | 0.010352 |
| 1e+03 | 2e+21 | 9.132607e-09 | 215.421875 | 0.010427 |
| 1e+04 | 2e+22 | 1.804021e-09 | 975.652344 | 0.493960 |
| 2e+04 | 2e+23 | 6.620469e-09 | 3264.742188 | 1.475800 |
| 3e+04 | 1e+23 | 4.944259e-09 | 7079.707031 | 3.785903 |

Outline

- Introduction
- Key Concepts
 - Error and Residual
 - Condition Number
 - Machine Precision
- **Iterative Methods**
 - Categories
 - Conjugate Gradient
 - Preconditioning
 - GMRES

Iterative Methods: Categories

- Stationary Methods
 - Jacobi, Gauss-Seidel, SOR, etc.
- Krylov Subspace Methods
 - Conjugate Gradient (CG), Generalized Minimal Residual (GMRES), BiCGSTAB, etc.
- Multigrid Methods
 - Hierarchical approach using coarser/finer grids.
- Preconditioned Iterations
- Domain Decomposition Methods

Iterative Methods: Categories

- Stationary Methods

- $x^{(k+1)} = Gx^{(k)} + c$, where G and c do not depend on iteration count (k).

- Non Stationary Methods

- $x^{(k+1)} = x^{(k)} + a_k p^{(k)}$, where computation involves information that change at each iteration.

Iterative Methods: Categories

- Stationary Methods
 - Jacobi, Gauss-Seidel, SOR, etc.
- Krylov Subspace Methods
 - **Conjugate Gradient (CG), Generalized Minimal Residual (GMRES), BiCGSTAB, etc.**
- Multigrid Methods
 - Hierarchical approach using coarser/finer grids.
- **Preconditioned Iterations**
- Domain Decomposition Methods

Iterative Methods: Conjugate Gradient

- Originally developed for solving the quadratic problem of vector \mathbf{x} :

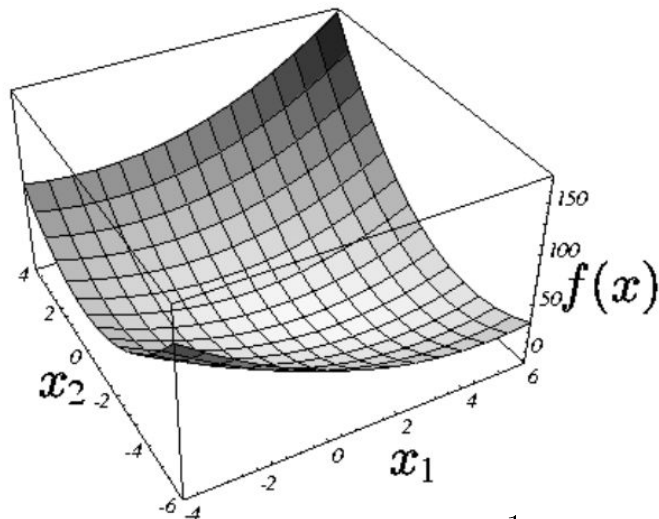
$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}$$

- Let $f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}$, then the derivative is $f'(\mathbf{x}) = \frac{1}{2} \mathbf{A}^T \mathbf{x} + \frac{1}{2} \mathbf{A} \mathbf{x} - \mathbf{b}$.
- If \mathbf{A} is symmetric, $f'(\mathbf{x}) = \mathbf{A} \mathbf{x} - \mathbf{b}$.
- If \mathbf{A} is symmetric, positive-definite, $f(\mathbf{x})$ is minimized by setting $f'(\mathbf{x}) = 0$, which is the solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$.

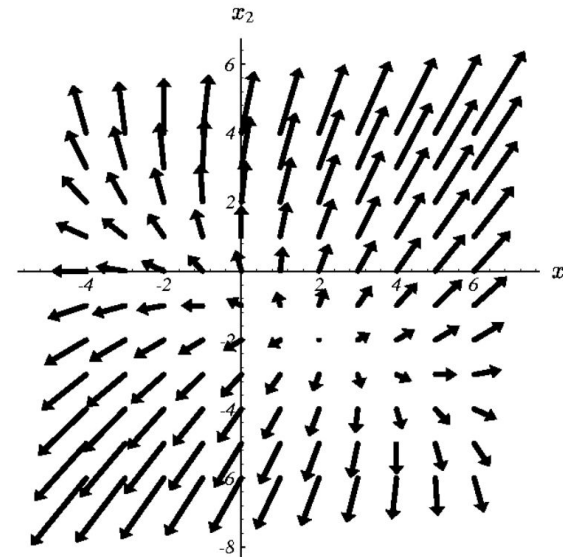
Iterative Methods: Conjugate Gradient

- Originally developed for solving the quadratic problem of vector \mathbf{x} :

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}$$



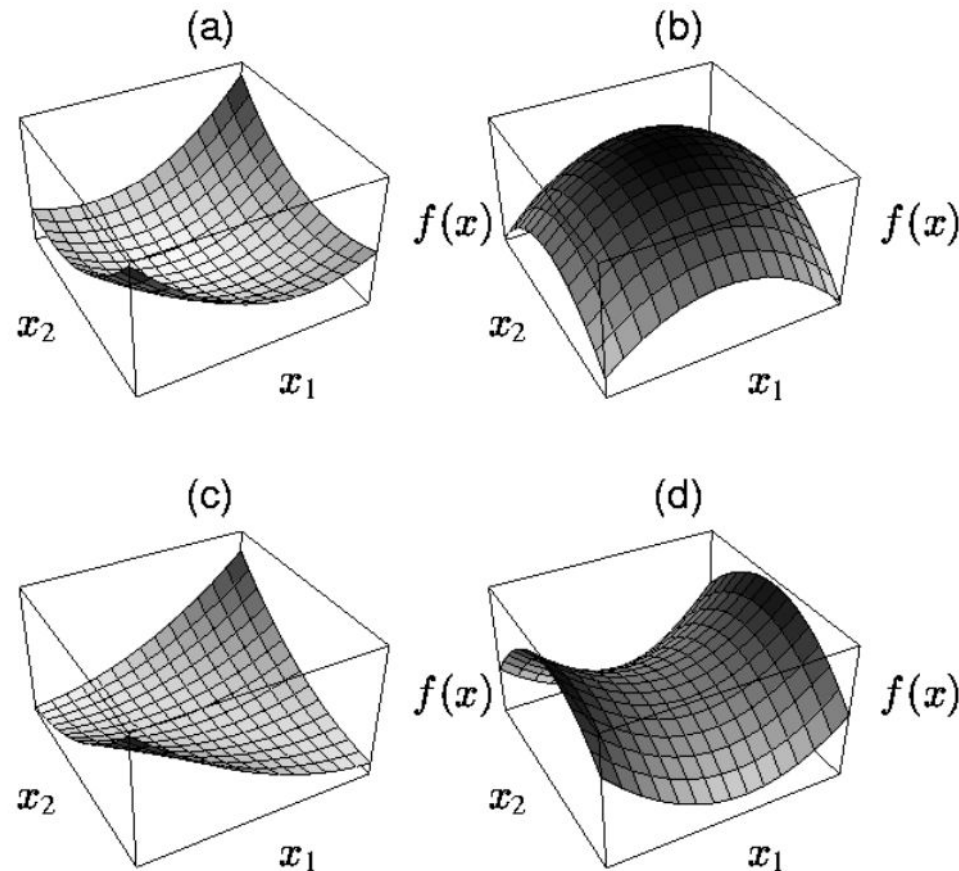
Graph of quadratic form $f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}$.
The minimum point of this surface is the
solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$.



The points in the direction of
steepest increase of $f(\mathbf{x})$.

Iterative Methods: Conjugate Gradient

- Originally developed for solving the quadratic problem of vector \mathbf{x} :



- a) Positive definite matrix
- b) Negative-definite matrix
- c) Singular matrix
- d) Positive indefinite matrix

Iterative Methods: Conjugate Gradient

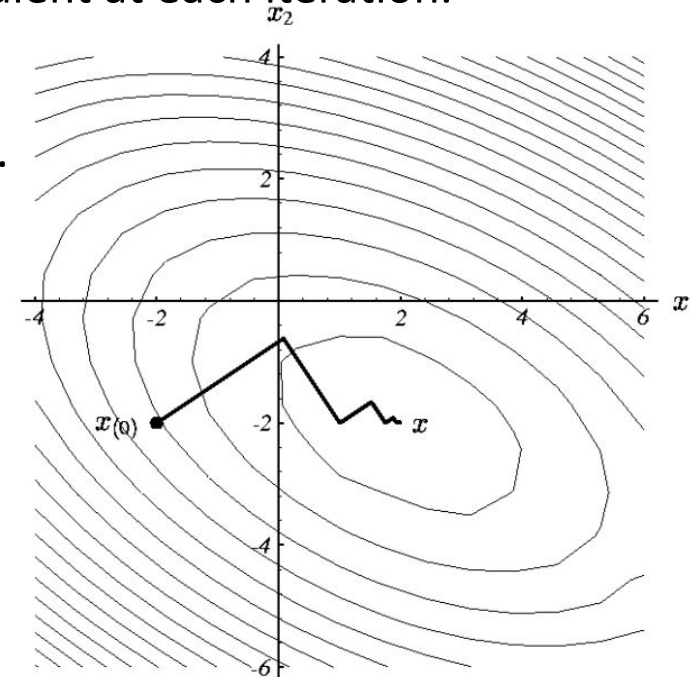
- Originally developed for solving the quadratic problem of vector \mathbf{x} :

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}$$

- **Assumption:** \mathbf{A} is a symmetric positive-definite (SPD) matrix.
- Equivalent to solving $\mathbf{A} \mathbf{x} = \mathbf{b}$.

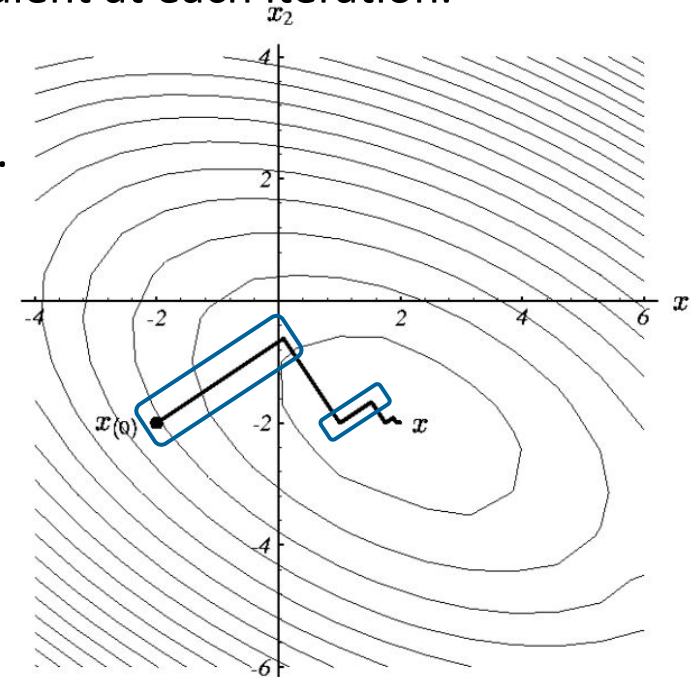
Iterative Methods: Conjugate Gradient

- One goal of conjugate gradient is to accelerate the steepest descent.
- Steepest descent
 - **Main Idea:**
 - Move in the direction of the steepest negative gradient at each iteration.
 - **Iteration:**
 - Compute negative gradient as search direction $d^{(k)}$.
 - Compute optimal step size α_k .
 - Update solution $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k d^{(k)}$.
 - **Drawback:**
 - Previously corrected directions can repeat.
 - “Zigzag” behavior can occur, leading to slow convergence.



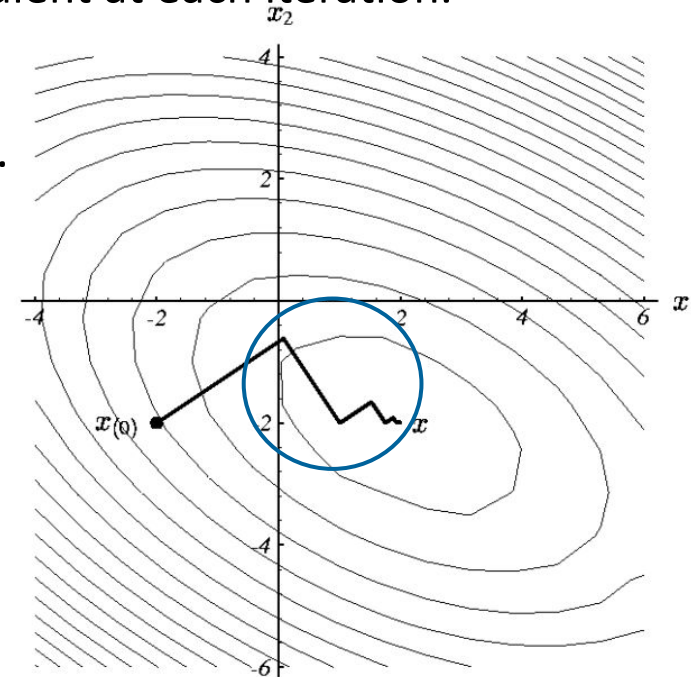
Iterative Methods: Conjugate Gradient

- One goal of conjugate gradient is to accelerate the steepest descent.
- Steepest descent
 - **Main Idea:**
 - Move in the direction of the steepest negative gradient at each iteration.
 - **Iteration:**
 - Compute negative gradient as search direction $d^{(k)}$.
 - Compute optimal step size α_k .
 - Update solution $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k d^{(k)}$.
 - **Drawback:**
 - **Previously corrected directions** can repeat.
 - “Zigzag” behavior can occur, leading to slow convergence.



Iterative Methods: Conjugate Gradient

- One goal of conjugate gradient is to accelerate the steepest descent.
- Steepest descent
 - **Main Idea:**
 - Move in the direction of the steepest negative gradient at each iteration.
 - **Iteration:**
 - Compute negative gradient as search direction $\mathbf{d}^{(k)}$.
 - Compute optimal step size α_k .
 - Update solution $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$.
 - **Drawback:**
 - **Previously corrected directions** can repeat.
 - “Zigzag” behavior can occur, leading to **slow convergence**.

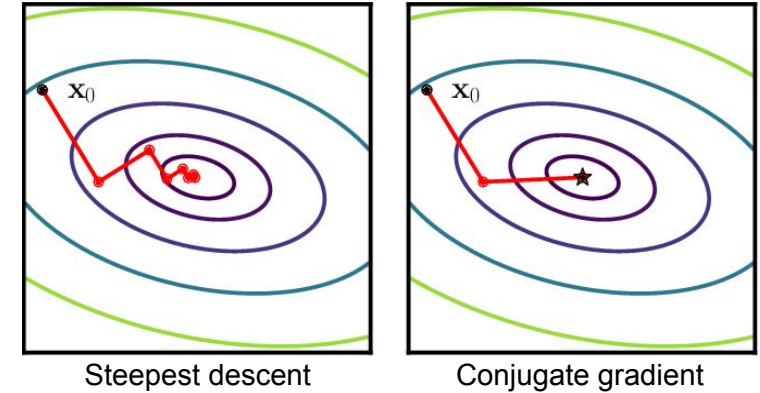


Iterative Methods: Conjugate Gradient

- Conjugate Gradient

- Main Idea:

- Make search direction **A-orthogonal** or **A-conjugate**: $\mathbf{d}^{(i)\top} \mathbf{A} \mathbf{d}^{(j)}$.
 - Each new direction is chosen to be orthogonal to all previous directions. This ensures each direction is “new” and not re-doing work from the previous iterations.



- Iteration:

- Start with $\mathbf{x}^{(0)}$, residual $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$, and direction $\mathbf{d}^{(0)} = \mathbf{r}^{(0)}$.
 - At each step

$$\begin{array}{lll} 1. \alpha_k = \frac{\mathbf{r}^{(k)\top} \mathbf{r}^{(k)}}{\mathbf{d}^{(k)\top} \mathbf{A} \mathbf{d}^{(k)}} & 3. \mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha_k \mathbf{A} \mathbf{d}^{(k)} & 5. \mathbf{d}^{(k+1)} = \mathbf{r}^{(k+1)} + \beta_{k+1} \mathbf{d}^{(k)} \\ 2. \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)} & 4. \beta_{k+1} = \frac{\mathbf{r}^{(k+1)\top} \mathbf{r}^{(k+1)}}{\mathbf{r}^{(k)\top} \mathbf{r}^{(k)}} & \end{array}$$

- Drawback:

- Only for SPD.
 - May still need advanced preconditioners for ill-conditioned problems.

Iterative Methods: Preconditioners

- Recall Condition Number

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^{-1}\|$$

- High $\kappa(\mathbf{A}) \Rightarrow$ slow (or unstable) convergence for iterative methods.

- Preconditioning

- **Main Idea:**

- Solve $\mathbf{B}^{-1}\mathbf{A}\mathbf{x} = \mathbf{B}^{-1}\mathbf{b}$ instead of $\mathbf{A}\mathbf{x} = \mathbf{b}$, where \mathbf{B} is a matrix that approximates \mathbf{A} but is easier to invert.

- **Goal**

- Reduce $\kappa(\mathbf{B}^{-1}\mathbf{A})$.
- Speed up convergence.

Iterative Methods: GMRES

- GMRES

- **Main Idea:**

- Works for **general** matrices (not necessarily SPD).
 - Builds an orthonormal basis of the Krylov subspace $K_i(\mathbf{A}, \mathbf{b})$ via Arnoldi iteration.
 - Minimizes the residual in each subspace.
 - Equivalently, we view GMRES as solving a least-squares objective (minimizing $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2$), the Hessian ends up being $\mathbf{A}^\top \mathbf{A}$, whose **condition number** is $\kappa(\mathbf{A}^\top \mathbf{A})$.

- **Iteration:**

- For each iteration, find $\mathbf{x}_i \in K_i(\mathbf{A}, \mathbf{b})$ such that $(\mathbf{A}\mathbf{x}_i - \mathbf{b}) \perp K_i(\mathbf{A}, \mathbf{b})$.

- **Drawback:**

- Need to store multiple previous directions. Memory usage grows with the iteration count.
 - Usually “restarted” every k iterations to use less space.