Linear Algebra and Applications: Numerical Linear Algebra

David S. Watkins
watkins@math.wsu.edu

Department of Mathematics
Washington State University
My Pledge to You
My Pledge to You

I promise not to cover as much material as I previously claimed I would.
Resources
Resources (a biased list)
Resources (a biased list)

Resources (a biased list)


Resources (a biased list)


Common Linear Algebra Computations
Common Linear Algebra Computations

- linear system $Ax = b$
Common Linear Algebra Computations

- linear system $Ax = b$
- overdetermined linear system $Ax = b$
Common Linear Algebra Computations

- linear system $Ax = b$
- overdetermined linear system $Ax = b$
- eigenvalue problem $Av = \lambda v$
Common Linear Algebra Computations

- linear system $Ax = b$
- overdetermined linear system $Ax = b$
- eigenvalue problem $Av = \lambda v$
- various generalized eigenvalue problems, e.g. $Av = \lambda Bv$
Linear Systems
Linear Systems

$Ax = b$, $n \times n$, nonsingular, real or complex
Linear Systems

- $Ax = b$, $n \times n$, nonsingular, real or complex
- Examples: FMC §1.2, 7.1; any linear algebra text
Linear Systems

- $Ax = b$, $n \times n$, nonsingular, real or complex
- Examples: FMC §1.2, 7.1; any linear algebra text
- Major tools:
  - Gaussian elimination (LU Decomp.)
  - various iterative methods
Overdetermined Linear Systems
Overdetermined Linear Systems

\[ Ax = b, \; n \times m, \; n > m \]
Overdetermined Linear Systems

- $Ax = b, n \times m, n > m$
- often $n \gg m$
Overdetermined Linear Systems

- $Ax = b$, $n \times m$, $n > m$
- often $n \gg m$
- Example: fitting data by a straight line
Overdetermined Linear Systems

- $Ax = b, \, n \times m, \, n > m$
- often $n \gg m$
- Example: fitting data by a straight line
- minimize $\| b - Ax \|_2$ (least squares)
Overdetermined Linear Systems

- $Ax = b$, $n \times m$, $n > m$
- often $n \gg m$
- Example: fitting data by a straight line
- minimize $\| b - Ax \|_2$ (least squares)
- Major tools:
  - QR decomposition
  - singular value decomposition
Eigenvalue Problems
Eigenvalue Problems

- standard: $Av = \lambda v$, $n \times n$, real or complex
Eigenvalue Problems

- standard: $Av = \lambda v$, $n \times n$, real or complex
- Examples: FMC § 5.1
Eigenvalue Problems

- standard: \( Av = \lambda v, \ n \times n, \ \text{real or complex} \)
- Examples: FMC § 5.1
- generalized: \( Av = \lambda Bv \)
Eigenvalue Problems

- standard: $Av = \lambda v$, $n \times n$, real or complex
- Examples: FMC § 5.1
- generalized: $Av = \lambda Bv$
- Examples: FMC § 6.7
Eigenvalue Problems

- standard: \( Av = \lambda v \), \( n \times n \), real or complex
  - Examples: FMC § 5.1
- generalized: \( Av = \lambda Bv \)
  - Examples: FMC § 6.7
- product: \( A_1 A_2 \)
Eigenvalue Problems

- standard: $Av = \lambda v$, $n \times n$, real or complex
  - Examples: FMC § 5.1
- generalized: $Av = \lambda Bv$
  - Examples: FMC § 6.7
- product: $A_1 A_2$
  - Examples: generalized $(AB^{-1})$, SVD $(A^* A)$
Eigenvalue Problems

- standard: $Av = \lambda v$, $n \times n$, real or complex
- Examples: FMC § 5.1
- generalized: $Av = \lambda Bv$
- Examples: FMC § 6.7
- product: $A_1A_2$
- Examples: generalized ($AB^{-1}$), SVD ($A^*A$)
- quadratic: $(\lambda^2K + \lambda G + M)v = 0$
Sizes of Linear Algebra Problems
Sizes of Linear Algebra Problems

- small
Sizes of Linear Algebra Problems

- small
- medium
Sizes of Linear Algebra Problems

- small
- medium
- large
Solving Linear Systems:
Solving Linear Systems: small problems
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
- $A = LU$
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
- $A = LU$
- $PA = LU$ (partial pivoting)
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
- $A = LU$
- $PA = LU$ (partial pivoting)
- forward and back substitution
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
- $A = LU$
- $PA = LU$ (partial pivoting)
- forward and back substitution
- Questions: cost?
Solving Linear Systems: small problems

- $Ax = b$, $n \times n$, $n$ “small”
- store $A$ conventionally
- solve using Gaussian elimination
- $A = LU$
- $PA = LU$ (partial pivoting)
- forward and back substitution
- Questions: cost?, accuracy? (FMC Ch. 2)
Positive Definite Case
Positive Definite Case

- $A = A^*$, \hspace{1cm} x^*Ax > 0 \text{ for all } x \neq 0$
Positive Definite Case

- $A = A^*$, $x^*A x > 0$ for all $x \neq 0$
- $A = R^* R$  Cholesky decomposition
Positive Definite Case

- $A = A^*$, $x^*Ax > 0$ for all $x \neq 0$
- $A = R^*R$  Cholesky decomposition
- symmetric variant of Gaussian elimination
Positive Definite Case

- $A = A^*$, $x^* Ax > 0$ for all $x \neq 0$
- $A = R^* R$  Cholesky decomposition
- symmetric variant of Gaussian elimination
- flop count is halved
Solving Linear Systems:
Solving Linear Systems: medium problems
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
- sparse Gaussian elimination
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
- sparse Gaussian elimination
- $A = LU$
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
- Sparse Gaussian elimination
- \( A = LU \)
- Factors “usually” less sparse than \( A \),
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
- Sparse Gaussian elimination
- $A = LU$
- Factors “usually” less sparse than $A$, but still sparse
Solving Linear Systems: medium problems

- Larger problems are usually sparser.
- Use sparse data structure.
- Sparse Gaussian elimination
- \( A = LU \)
- Factors “usually” less sparse than \( A \), but still sparse
- Crucial question: Can factors fit in main memory?
Solving Linear Systems:
Solving Linear Systems: large problems
Solving Linear Systems: large problems

$L$ and $U$ factors may be too large to store ...
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- direct vs. iterative methods
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- direct vs. iterative methods
- Some buzz words: descent method, conjugate gradients (CG), GMRES, . . .
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- direct vs. iterative methods
- Some buzz words: descent method, conjugate gradients (CG), GMRES, . . .
- preconditioners,
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- Direct vs. iterative methods
- Some buzz words: descent method, conjugate gradients (CG), GMRES, . . .
- Preconditioners, and on and on and on.
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- direct vs. iterative methods
- Some buzz words: descent method, conjugate gradients (CG), GMRES, . . .
- preconditioners, and on and on and on.
- FMC Chapter 7
Solving Linear Systems: large problems

- $L$ and $U$ factors may be too large to store . . .
- Use an iterative method.
- direct vs. iterative methods
- Some buzz words: descent method, conjugate gradients (CG), GMRES, . . .
- preconditioners, and on and on.
- FMC Chapter 7
Moving On
Moving On
Orthogonal Transformations
Moving On Orthogonal Transformations

- generally useful computing tools
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
- standard inner product: \[ \langle x, y \rangle = \sum_{j=1}^{n} x_j y_j \]
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
- standard inner product: \( \langle x, y \rangle = \sum_{j=1}^{n} x_j y_j \)
- Euclidean norm: \( \| x \|_2 = \left( \sum_{j=1}^{n} x_j^2 \right)^{1/2} \)
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
- standard inner product: \[ \langle x, y \rangle = \sum_{j=1}^{n} x_j y_j \]
- Euclidean norm: \[ \| x \|_2 = \left( \sum_{j=1}^{n} x_j^2 \right)^{1/2} \]
- \[ \| x \|_2 = \sqrt{\langle x, x \rangle} \]
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
- standard inner product: \( \langle x, y \rangle = \sum_{j=1}^{n} x_j y_j \)

- Euclidean norm: \( \| x \|_2 = \left( \sum_{j=1}^{n} x_j^2 \right)^{1/2} \)

- \( \| x \|_2 = \sqrt{\langle x, x \rangle} \)

- definition of orthogonal: \( Q^T = Q^{-1} \)
Moving On
Orthogonal Transformations

- generally useful computing tools
- sticking to real case for simplicity
- standard inner product: \[ \langle x, y \rangle = \sum_{j=1}^{n} x_j y_j \]
- Euclidean norm: \[ \| x \|_2 = \left( \sum_{j=1}^{n} x_j^2 \right)^{1/2} \]
- \[ \| x \|_2 = \sqrt{\langle x, x \rangle} \]
- definition of orthogonal: \[ Q^T = Q^{-1} \]
- properties of orthogonal matrices
Elementary Reflectors
Elementary Reflectors

= Householder transformations
Elementary Reflectors

- Householder transformations
- one of two major classes of computationally useful orthogonal transformations
Elementary Reflectors

- Householder transformations
- one of two major classes of computationally useful orthogonal transformations
- \( Q = I - 2uu^T, \quad \|u\|_2 = 1 \)
Elementary Reflectors

- = Householder transformations
- one of two major classes of computationally useful orthogonal transformations
- \[ Q = I - 2uu^T, \quad \| u \|_2 = 1 \]
- geometric action
Elementary Reflectors

- Householder transformations
- one of two major classes of computationally useful orthogonal transformations
- \( Q = I - 2uu^T, \quad \| u \|_2 = 1 \)
- geometric action
- \( Qx = y \)
Elementary Reflectors

- Householder transformations
- One of two major classes of computationally useful orthogonal transformations
- \( Q = I - 2uu^T, \quad \| u \|_2 = 1 \)
- Geometric action
- \( Qx = y \)
- Creating zeros
Elementary Reflectors

- Householder transformations
- one of two major classes of computationally useful orthogonal transformations
- \[ Q = I - 2uu^T, \quad \| u \|_2 = 1 \]
- geometric action
- \[ Qx = y \]
- creating zeros
- details: FMC Chapter 3
Elementary Reflectors

- Householder transformations
- One of two major classes of computationally useful orthogonal transformations

\[ Q = I - 2uu^T, \quad \| u \|_2 = 1 \]
- Geometric action
- \( Qx = y \)
- Creating zeros
- Details: FMC Chapter 3
- \( QR \) decomposition
Uses of the $QR$ Decomposition
Uses of the $QR$ Decomposition

- $Ax = b, \quad n \times n$
Uses of the $QR$ Decomposition

- $Ax = b$, $n \times n$
- overdetermined system
Uses of the $QR$ Decomposition

- $Ax = b$, $n \times n$
- overdetermined system
- orthonormalizing vectors
The Gram-Schmidt Process
The Gram-Schmidt Process

- orthonormalization of vectors
The Gram-Schmidt Process

- orthonormalization of vectors
- relationship to $QR$ decomposition
The Gram-Schmidt Process

- orthonormalization of vectors
- relationship to $QR$ decomposition
- reorthogonalization
The SVD
The SVD

- singular value decomposition
The SVD

- singular value decomposition
- \( A = U \Sigma V^T \)
The SVD

- singular value decomposition
- \( A = U\Sigma V^T \)
- product eigenvalue problem
The SVD

- singular value decomposition
- $A = U\Sigma V^T$
- product eigenvalue problem
- FMC Chapter 4
The SVD

- singular value decomposition
- $A = U\Sigma V^T$
- product eigenvalue problem
- FMC Chapter 4
- numerical rank determination
The SVD

- singular value decomposition
- \( A = U \Sigma V^T \)
- product eigenvalue problem
- FMC Chapter 4
- numerical rank determination
- solution of least-squares problem
End of Part I