CS 450: Numerical Analysis¹
Linear Systems

University of Illinois at Urbana-Champaign

¹These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book “Scientific Computing: An Introductory Survey” by Michael T. Heath (slides).
Vector Norms

▶ Properties of vector norms

\[ \| x \| \geq 0 \]
\[ \| x \| = 0 \iff x = 0 \]
\[ \| \alpha x \| = |\alpha| \cdot \| x \| \]
\[ \| x + y \| \leq \| x \| + \| y \| \quad \text{(triangle inequality)} \text{ implies continuity} \]

▶ A norm is uniquely defined by its unit sphere: Surface defined by space of vectors \( V \subset \mathbb{R}^n \) such that \( \forall x \in V, \| x \| = 1 \)

▶ \( p \)-norms \( \| x \|_p = \left( \sum_i |x_i|^p \right)^{1/p} \)

▶ \( p = 1 \) gives sum of absolute values of entry (unit sphere is diamond-like)
▶ \( p = \infty \) gives maximum entry in absolute value (unit sphere is box-like)
▶ \( p = 2 \) gives Euclidean distance metric (unit sphere is spherical)
Inner-Product Spaces

▶ Properties of inner-product spaces: Inner products \( \langle x, y \rangle \) must satisfy

\[
\langle x, x \rangle \geq 0
\]
\[
\langle x, x \rangle = 0 \iff x = 0
\]
\[
\langle x, y \rangle = \langle y, x \rangle
\]
\[
\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle
\]
\[
\langle \alpha x, y \rangle = \alpha \langle x, y \rangle
\]

▶ Inner-product-based vector norms

The \( p = 2 \) vector norm is the Euclidean inner-product norm,

\[
||x||_2 = \sqrt{x^T x}
\]
and due to Cauchy-Schwartz inequality \( |\langle x, y \rangle| \leq \sqrt{\langle x, x \rangle \cdot \langle y, y \rangle} \),

\[
|x^T y| \leq ||x||_2 ||y||_2.
\]

Other inner-products can be expressed as \( \langle x, y \rangle = x^T Ay \) where \( A \) is symmetric positive definite, yielding norms \( ||x||_A = \sqrt{x^T Ax} \).
Matrix Norms

- Properties of matrix norms:

\[
\|A\| \geq 0 \\
\|A\| = 0 \iff A = 0 \\
\|\alpha A\| = |\alpha| \cdot \|A\| \\
\|A + B\| \leq \|A\| + \|B\| \quad (\text{triangle inequality})
\]

- Frobenius norm:

\[
\|A\|_F = \left( \sum_{i,j} a_{ij}^2 \right)^{1/2}
\]

- Operator/induced/subordinate matrix norms:

For any vector norm \(\|\cdot\|\), the induced matrix norm is

\[
\|A\| = \max_{x \neq 0} \|Ax\|/\|x\| = \max_{\|x\|=1} |Ax|
\]
Interpreting induced matrix norms: A matrix is uniquely defined with respect to a norm by a unit-ball, which is the space of vectors \( y = Ax \) for all \( x \) on the unit-sphere of the norm.

\[
\|A\|_p = \max_{\|x\|_p=1} \|Ax\|_p
\]

is the maximum possible \( p \)-norm amplification due to application of \( A \)

\[
1/\|A^{-1}\|_p = \min_{\|x\|_p=1} \|Ax\|_p
\]

is the maximum possible \( p \)-norm reduction due to application of \( A \)

General induced matrix norms:

\[
\|A\|_{mp} = \max_{\|x\|_p=1} \|Ax\|_m
\]

typically \( m = p \) so we write \( \|A\|_p \) and almost always we have \( p \in \{1, 2, \infty\} \).

(Computing the matrix norm for certain choices of \( m \neq p \) is NP-complete.)
Matrix Condition Number

- **Definition:** \( \kappa(A) = ||A|| \cdot ||A^{-1}|| \) is the ratio between the shortest/longest distances from the unit-ball center to any point on the surface.

- **Intuitive derivation:**

  \[
  \kappa(A) = \max_{\text{inputs}} \max_{\text{perturbations in input}} \left| \frac{\text{relative perturbation in output}}{\text{relative perturbation in input}} \right|
  \]

  since a matrix is a linear operator, we can decouple its action on the input \( x \) and the perturbation \( \delta x \) since \( A(x + \delta x) = Ax + A\delta x \), so

\[
\kappa(A) = \frac{||A|| \max_{\text{perturbations in input}} \text{relative perturbation growth}}{1/||A^{-1}|| \max_{\text{inputs}} \text{relative input reduction}}
\]
Review: Matrix Conditioning

- The matrix condition number $\kappa(A)$ is the ratio between the max and min distance from the surface to the center of the unit ball transformed by $\kappa(A)$:
  - The max distance to center is given by the vector maximizing $\max_{||x||=1} ||Ax||_2$.
  - The min distance to center is given by the vector minimizing $\min_{||x||=1} ||Ax||_2 = 1/(\max_{||x||=1} ||A^{-1}x||_2)$.
  - Thus, we have that $\kappa(A) = ||A||_2 ||A^{-1}||_2$

- The matrix condition number bounds the worst-case amplification of error in a matrix-vector product: Consider $y + \delta y = A(x + \delta x)$, assume $||x||_2 = 1$
  - In the worst case, $||y||_2$ is minimized, that is $||y||_2 = 1/||A^{-1}||_2$
  - In the worst case, $||\delta y||_2$ is maximized, that is $||\delta y||_2 = ||A||_2 ||\delta y||_2$
  - So $||\delta y||_2/||y||_2$ is at most $\kappa(A)||\delta x||_2/||x||_2$
Norms and Conditioning of Orthogonal Matrices

- **Orthogonal matrices**: A matrix $Q$ is orthogonal, if its square and its columns are orthonormal, or equivalently $Q^T = Q^{-1}$.

- **Norm and condition number of orthogonal matrices**: For any $\|v\|_2 = 1$,

\[
\|Qv\|_2 = \left( \left( v^T Q^T, Qv \right) \right)^{1/2} = \left( v^T Q^T Qv \right)^{1/2} = \left( v^T v \right)^{1/2} = \|v\|_2
\]

Consequently, $\|Q\|_2 = \|Q^{-1}\|_2 = \kappa(Q) = 1$.

$Qv$ expresses $v$ in a coordinate system whose axes are columns of $Q^T$. 
The singular value decomposition (SVD):

We can express any matrix $A$ as

$$A = U \Sigma V^T$$

where $U$ and $V$ are orthogonal, and $\Sigma$ is square nonnegative and diagonal,

$$\Sigma = \begin{bmatrix} \sigma_{\text{max}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\text{min}} \end{bmatrix}$$

Any matrix is diagonal when expressed as an operator mapping vectors from a coordinate system given by $V$ to a coordinate system given by $U^T$. 
Norms and Conditioning using the SVD

- **Norm and condition number in terms of singular values:**
  When multiplying a vector by matrix $A = U\Sigma V^T$
  - Multiplication by $V^T$ changes coordinate systems, leaving the norm unchanged
  - Multiplication by $U$ changes coordinate systems, leaving the norm unchanged
  so, only multiplication by $\Sigma$ has an effect on the vector norm
  - Note that $||\Sigma||_2 = \sigma_{\text{max}}$, $||\Sigma^{-1}||_2 = 1/\sigma_{\text{min}}$, so

$$\kappa(A) = \kappa(\Sigma) = \frac{\sigma_{\text{max}}}{\sigma_{\text{min}}}$$
Conditioning of Linear Systems

Let's now return to formally deriving the conditioning of solving \( Ax = b \):

Consider a perturbation to the right-hand side (input) \( \hat{b} = b + \delta b \)

\[
A\hat{x} = \hat{b}
\]

\[
A(x + \delta x) = b + \delta b
\]

\[
A\delta x = \delta b
\]

we wish to bound the size of the relative perturbation to the output \( \frac{||\delta x||}{||x||} \)

with respect to the size of the relative perturbation the the input \( \frac{||\delta b||}{||b||} \)

\[
\delta x = A^{-1}\delta b
\]

\[
\frac{||\delta x||}{||x||} = \frac{||A^{-1}\delta b||}{||x||} \leq \frac{||A^{-1}|| \cdot ||\delta b||}{||x||}
\]

we can use that \( ||x|| \geq ||b||/\sigma_{\text{max}} = ||b||/||A|| \) so

\[
\frac{||\delta x||}{||x||} \leq \left( \frac{||A|| \cdot ||A^{-1}||}{\kappa(A)} \right) \cdot \frac{||\delta b||}{||b||} = \frac{\sigma_{\text{max}}||\delta b||}{\sigma_{\text{min}}||b||}
\]
Consider perturbations to the input coefficients $\hat{A} = A + \delta A$:

In this case we solve the perturbed system

$$\hat{A}\hat{x} = b$$
$$Ax + \delta Ax = b - \hat{A}\delta x$$

$$\delta Ax = -\hat{A}\delta x \approx -A\delta x$$

we wish to bound the size of the relative perturbation to the output $||\delta x||/||x||$ with respect to the size of the relative perturbation the the input $||\delta A||/||A||$

$$\delta x = -A^{-1}\delta Ax$$

$$||\delta x|| = ||A^{-1}\delta Ax|| \leq ||A^{-1}|| \cdot ||\delta A|| \cdot ||x||$$

$$\frac{||\delta x||}{||x||} \leq \frac{||A^{-1}|| \cdot ||A|| \cdot ||\delta A||}{\kappa(A) \cdot ||A||}$$
Solving Basic Linear Systems

- Solve $Dx = b$ if $D$ is diagonal
  
  $x_i = b_i/d_{ii}$ with total cost $O(n)$

- Solve $Qx = b$ if $Q$ is orthogonal
  
  $x = Q^Tb$ with total cost $O(n^2)$

- Given SVD $A = U\Sigma V^T$, solve $Ax = b$
  
  - Compute $z = U^Tb$
  - Solve $\Sigma y = z$ (diagonal)
  - Compute $x = V x$
Solving Triangular Systems

- \( Lx = b \) if \( L \) is lower-triangular is solved by forward substitution:

\[
\begin{align*}
l_{11}x_1 &= b_1 & x_1 &= b_1/l_{11} \\
l_{21}x_1 + l_{22}x_2 &= b_2 & \Rightarrow x_2 &= (b_2 - l_{21}x_1)/l_{22} \\
l_{31}x_1 + l_{32}x_2 + l_{33}x_3 &= b_3 & x_3 &= (b_3 - l_{31}x_1 - l_{32}x_2)/l_{33}
\end{align*}
\]

\[\vdots\]

- Algorithm can also be formulated recursively by blocks:

\[
\begin{bmatrix}
l_{11} & & \\
l_{21} & L_{22} & \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} =
\begin{bmatrix}
b_1 \\
b_2
\end{bmatrix}
\]

\( x_1 = b_1/l_{11} \), then solve recursively for \( x_2 \) in \( L_{22}x_2 = b_2 - l_{21}x_1 \).
Solving Triangular Systems

- **Existence of solution to** $Lx = b$:
  
  *If some $l_{ii} = 0$, the solution may not exist, and $L^{-1}$ does not exist.*

- **Uniqueness of solution:** *Even if some $l_{ii} = 0$ and $L^{-1}$ does not exist, the system may have a solution. The solution will not be unique since columns of $L$ are necessarily linearly dependent if a diagonal element is zero. May want to select solution minimizing norm of $x$.*

- **Computational complexity of forward/backward substitution:**
  
  *The recursive algorithm has the cost recurrence,*

  $$T(n) = T(n - 1) + n = \sum_{i=1}^{n} i = n(n + 1)/2.$$  

  *The total cost is $n^2/2$ multiplications and $n^2/2$ additions to leading order.*
Properties of Triangular Matrices

- \( Z = XY \) is lower triangular if \( X \) and \( Y \) are both lower triangular:

\[
\begin{bmatrix}
z_{11} & z_{12} \\
z_{21} & Z_{22}
\end{bmatrix}
= \begin{bmatrix}
x_{11} & \\
x_{21} & X_{22}
\end{bmatrix}
\begin{bmatrix}
y_{11} \\
y_{21}
\end{bmatrix}.
\]

Clearly, \( z_{11} = x_{11}y_{11} \) and \( z_{12} = 0 \), then we proceed by the same argument for the triangular matrix product \( Z_{22} = X_{22}Y_{22} \).

- \( L^{-1} \) is lower triangular if it exists:

We give a constructive proof by providing an algorithm for triangular matrix inversion. We need \( Y = X^{-1} \) so

\[
\begin{bmatrix}
Y_{11} & \\
Y_{21} & Y_{22}
\end{bmatrix}
\begin{bmatrix}
X_{11} \\
X_{21} & X_{22}
\end{bmatrix}
= \begin{bmatrix}
I \\
I
\end{bmatrix},
\]

from which we can deduce

\[
Y_{11} = X_{11}^{-1}, \quad Y_{22} = X_{22}^{-1}, \quad Y_{21} = -Y_{22}X_{21}Y_{11}.
\]
An **LU factorization** consists of a unit-diagonal lower-triangular factor $L$ and upper-triangular factor $U$ such that $A = LU$:

- Unit-diagonal implies each $l_{ii} = 1$, leaving $n(n - 1)/2$ unknowns in $L$ and $n(n + 1)/2$ unknowns in $U$, for a total of $n^2$, the same as the size of $A$.

- For rectangular matrices $A \in \mathbb{R}^{m \times n}$, one can consider a full LU factorization, with $L \in \mathbb{R}^{m \times \max(m,n)}$ and $U \in \mathbb{R}^{\max(m,n) \times n}$, but it is fully described by a reduced LU factorization, with lower-trapezoidal $L \in \mathbb{R}^{m \times \min(m,n)}$ and upper-trapezoidal $U \in \mathbb{R}^{\min(m,n) \times n}$.

Given an LU factorization of $A$, we can solve the linear system $Ax = b$:

- using forward substitution $Ly = b$
- using backward substitution to solve $Ux = y$

Backward substitution is the same as forward substitution with a reversal of the ordering of the elements of the vectors and the ordering of the rows/columns of the matrix.
Gaussian Elimination Algorithm

- Algorithm for factorization is derived from equations given by \( A = LU \):

\[
\begin{bmatrix}
    a_{11} & a_{12} \\
    a_{21} & A_{22}
\end{bmatrix}
= \begin{bmatrix}
    1 & 0 \\
    l_{21} & L_{22}
\end{bmatrix}
\begin{bmatrix}
    u_{11} & u_{12} \\
    0 & U_{22}
\end{bmatrix}
= \begin{bmatrix}
    L_{11} & 0 \\
    L_{21} & L_{22}
\end{bmatrix}
\begin{bmatrix}
    U_{11} & U_{12} \\
    0 & U_{22}
\end{bmatrix}
\]

- First, observe \( \begin{bmatrix} u_{11} & u_{12} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \end{bmatrix} \)

- To obtain \( l_{21} \) compute \( l_{21} = a_{21} / u_{11} \)

- Obtain \( L_{22} \) and \( U_{22} \) by recursively computing LU of the Schur complement \( S = A_{22} - l_{21} u_{12} \)

- The computational complexity of LU is \( O(n^3) \):

Computing \( l_{21} = a_{21} / u_{11} \) requires \( O(n) \) operations, finding \( S \) requires \( 2n^2 \), so to leading order the complexity of LU is

\[
T(n) = T(n - 1) + 2n^2 = \sum_{i=1}^{n} 2i^2 \approx 2n^3 / 3
\]
Existence of LU factorization

- **The LU factorization may not exist:** Consider matrix
  \[
  \begin{bmatrix}
  3 & 2 \\
  6 & 4 \\
  0 & 3
  \end{bmatrix}
  \]

  Proceeding with Gaussian elimination we obtain
  \[
  \begin{bmatrix}
  3 & 2 \\
  6 & 4 \\
  0 & 3
  \end{bmatrix} =
  \begin{bmatrix}
  1 & 0 \\
  2 & 1 \\
  0 & l_{32}
  \end{bmatrix}
  \begin{bmatrix}
  3 & 2 \\
  0 & u_{21}
  \end{bmatrix}.
  \]

  Then we need that \(4 = 4 + u_{21}\) so \(u_{21} = 0\), but at the same time \(l_{32}u_{21} = 3\).

  More generally, if and only if for any partitioning
  \[
  \begin{bmatrix}
  A_{11} & A_{12} \\
  A_{21} & A_{22}
  \end{bmatrix}
  \]
  the leading minor is singular (\(\det(A_{11}) = 0\)), \(A\) has no LU factorization.

- **Permutation of rows enables us to transform the matrix so the LU factorization does exist:**

  Gaussian elimination can only fail if dividing by zero. At every recursive step of Gaussian elimination, if the leading entry of the first row is zero, we permute it with a row with an leading nonzero (if \(a_{21} = 0\), we set \(u_{11} = 0\) and \(l_{21} = 0\)).
Gaussian Elimination with Partial Pivoting

- **Partial pivoting** permutes rows to make divisor $u_{ii}$ is maximal at each step:

  Based on our argument above, for any matrix $A$ there exists a permutation matrix $P$ that can permute the rows of $A$ to permit an LU factorization,

  $$PA = LU.$$

  Partial pivoting finds such a permutation matrix $P$ one row at a time. The $i$th row is selected to maximize the magnitude of the leading element (over elements in the first column), which becomes the entry $u_{ii}$. This selection ensures that we are never forced to divide by zero during Gaussian elimination and that the magnitude of any element in $L$ is at most 1.

- A row permutation corresponds to an application of a **row permutation matrix** $P_{jk} = I - (e_j - e_k)(e_j - e_k)^T$:

  If we permute row $i_j$ to be the leading ($i$th) row at the $i$th step, the overall permutation matrix is given by

  $$P^T = \prod_{i=1}^{n-1} P_{ii_j}.$$
Partial Pivoting Example

- Lets consider again the matrix $A = \begin{bmatrix} 3 & 2 \\ 6 & 4 \\ 0 & 3 \end{bmatrix}$.

- The largest magnitude element in the first column is 6, so we select this as our pivot and perform the first step of LU

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 6 & 4 \\ 3 & 2 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1/2 \\ 0 \end{bmatrix} \begin{bmatrix} 6 & 4 \\ 0 & 2 - (1/2) \cdot 4 \\ 0 & 3 - 0 \cdot 4 \end{bmatrix}$$

- The Schur complement is $\begin{bmatrix} 0 & 3 \end{bmatrix}^T$ and we proceed with pivoted LU,

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 3 \end{bmatrix}$$

- The overall LU factorization is then given by $P_1 \begin{bmatrix} 1 & P_2 \end{bmatrix} A = \begin{bmatrix} 1 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} 6 & 4 \\ 0 & 3 \end{bmatrix}$.
Complete Pivoting

- **Complete pivoting** permutes rows and columns to make divisor $u_{ii}$ maximal at each step:
  - Partial pivoting ensures that the magnitude of the multipliers satisfies
    \[ |l_{21}| = |a_{21}|/|u_{11}| \leq 1 \]
  - Complete pivoting also gives $||u_{12}||_{\infty} \leq |u_{11}|$ and consequently
    \[ |l_{21}| \cdot ||u_{12}||_{\infty} = |a_{21}| \cdot ||u_{12}||_{\infty}/|u_{11}| \leq |a_{21}| \]
  - Complete pivoting yields a factorization of the form $LU = PAQ$ where $P$ and $Q$ are permutation matrices

- Complete pivoting is noticeably more expensive than partial pivoting:
  - Partial pivoting requires just $O(n)$ comparison operations and a row permutation
  - Complete pivoting requires $O(n^2)$ comparison operations, which somewhat increases the leading order cost of LU overall
Round-off Error in LU

Let’s consider factorization of \[
\begin{bmatrix}
\epsilon & 1 \\
1 & 1
\end{bmatrix}
\]
where \( \epsilon < \epsilon_{\text{mach}} \):

1. Without pivoting we would compute \( L = \begin{bmatrix} 1 & 0 \\ 1/\epsilon & 1 \end{bmatrix} \), \( U = \begin{bmatrix} \epsilon & 1 \\ 0 & 1 - 1/\epsilon \end{bmatrix} \).

2. Rounding yields \( fl(U) = \begin{bmatrix} \epsilon & 1 \\ 0 & -1/\epsilon \end{bmatrix} \).

This leads to \( L \cdot fl(U) = \begin{bmatrix} \epsilon & 1 \\ 1 & 0 \end{bmatrix} \), a backward error of \( \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \).

3. Permuting the rows of \( A \) in partial pivoting gives \( PA = \begin{bmatrix} 1 & 1 \\ \epsilon & 1 \end{bmatrix} \).

We now compute \( L = \begin{bmatrix} 1 & 0 \\ \epsilon & 1 \end{bmatrix} \), \( U = \begin{bmatrix} 1 & 1 \\ 0 & 1 - \epsilon \end{bmatrix} \), so \( fl(U) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \).

This leads to \( L \cdot fl(U) = \begin{bmatrix} 1 & 1 \\ \epsilon & 1 + \epsilon \end{bmatrix} \), a backward error of \( \begin{bmatrix} 0 & 0 \\ 0 & \epsilon \end{bmatrix} \).
Error Analysis of LU

- The main source of round-off error in LU is in the computation of the Schur complement:
  - Recall that division is well-conditioned, while addition can be ill-conditioned
  - After $k$ steps of LU, we are working on Schur complement $A_{22} - L_{21}U_{12}$ where $A_{22}$ is $(n - k) \times (n - k)$, $L_{21}$ and $U_{12}^T$ are $(n - k) \times k$
  - Partial pivoting and complete pivoting improve stability by making sure $L_{21}U_{12}$ is small in norm

- When computed in floating point, absolute backward error $\delta A$ in LU (so $\hat{L}\hat{U} = A + \delta A$) is $|\delta a_{ij}| \leq \epsilon_{\text{mach}}(|\hat{L}| \cdot |\hat{U}|)_{ij}$

For any $a_{ij}$ with $j \geq i$ (lower-triangle is similar), we compute

$$a_{ij} - \sum_{k=1}^{i} \hat{l}_{ik}\hat{u}_{kj} = a_{ij} - \langle \hat{l}_i, \hat{u}_j \rangle,$$

which in floating point incurs round-off error at most $\epsilon_{\text{mach}}|\langle \hat{l}_i, \hat{u}_j \rangle|$. Using this, for complete pivoting, we can show $|\delta a_{ij}| \leq \epsilon_{\text{mach}}n^2||A||_{\infty}$. 

Helpful Matrix Properties

- **Matrix is diagonally dominant**, so $\sum_{i\neq j} |a_{ij}| \leq |a_{ii}|$:
  Pivoting is not required if matrix is strictly diagonally dominant $\sum_{i\neq j} |a_{ij}| < |a_{ii}|$.

- **Matrix is symmetric positive definite (SPD)**, so $\forall x \neq 0, x^T Ax > 0$:
  $L = U$ and pivoting is not required, *Cholesky algorithm* $A = LL^T$ can be used ($L$ in Cholesky is not unit-diagonal).

- **Matrix is symmetric but indefinite**:
  Compute pivoted *LDL factorization* $PAP^T = LDL^T$ (where $L$ is lower-triangular and unit-diagonal, while $D$ is diagonal)

- **Matrix is banded**, $a_{ij} = 0$ if $|i - j| > b$:
  LU without pivoting and Cholesky preserve banded structure and require only $O(nb^2)$ work.
Solving Many Linear Systems

- Suppose we have computed $A = LU$ and want to solve $AX = B$ where $B$ is $n \times k$ with $k < n$:
  
  Cost is $O(n^2k)$ for solving the $k$ independent linear systems

- Suppose we have computed $A = LU$ and now want to solve a perturbed system $(A - uv^T)x = b$:
  
  Can use the Sherman-Morrison-Woodbury formula

  $$(A - uv^T)^{-1} = A^{-1} + \frac{A^{-1}uv^TA^{-1}}{1 - v^TA^{-1}u}$$

  - Consequently we have $Ax = b + \frac{uv^TA^{-1}b}{1 - v^TA^{-1}u} = b + \frac{v^TA^{-1}b}{1 - v^TA^{-1}u}u$
  - Need not form $A^{-1}$ or $L^{-1}$ or $U^{-1}$, suffices to use backward/forward substitution to solve $w^TA = v^T$, i.e. solve $U^TL^T w = v$ and then solve

  $$LUx = b + \left( \frac{w^Tb}{1 - w^Tu} \right) u$$