Solving Linear Systems (Direct Methods)

Recommended reading:

- Lectures in [4]: 6,7,8 for QR; 20,21 for LU; 23 for $LL^\top$
- Section I.4 in [3]
- Sections 2.6, 2.7 in [2]

References

[1] R. Rannacher.
Numerik 0 - Einführung in die Numerische Mathematik.
Heidelberg University Publishing, 2017.

[2] G. Strang.
Introduction to Linear Algebra.
Wellesley-Cambridge Press, 2003.

[3] G. Strang.
Linear Algebra and Learning from Data.
Wellesley-Cambridge Press, 2019.

[4] L.N. Trefethen and D. Bau.
Numerical linear algebra.
SIAM, Soc. for Industrial and Applied Math., Philadelphia, 1997.

2 Solving Linear Systems with Direct Methods

Aim:

$$\text{Given } A \in \mathbb{R}^{m\times n} \text{ } (m \neq n \text{ possible}) \text{ and } b \in \mathbb{R}^m, \text{ find } x \in \mathbb{R}^n \text{ such that}$$
$$Ax = b.$$

2.1 The Idea of "Factor and Solve"

A general theme in numerical mathematics is to reduce the general (possibly complicated) problem to one or more simpler problems with the help of transformations for which the property of interest is an invariant.

Simple cases:

- $A = D = \text{diag}(d_1, \ldots, d_n)$ diagonal
- $A = L$ (or $A = U$) lower (or upper) triangular system
  $\rightarrow$ backward/forward substitution (exercise!)
- $A = Q$ orthogonal
  $\rightarrow A^{-1} = Q^\top$
- A has special structure: tridiagonal/banded, Toeplitz, circulant,...

General case:

Assume $A = F_1 \cdots F_k$, where each factor $F_i \in \{D, L, U, Q\}$, then formally

$$x = F_k^{-1} \cdots F_1^{-1} b,$$

where each solving step F_i^{-1} can easily be performed.

(*Remark:* "direct" = finitely many steps to obtain the solution (typically operating on dense arrays))

Separate: Factorization and Solution!

- Since the *factorization* (*elimination/triangularization/orthogonalization*) step is typically much more expensive than the *solution* step, it makes sense to perform them separately, if the same matrix A is used for multiple right-hand sides b.

- The number of factors in a decomposition is the number of systems to be solved in the *solution* step.

  From the factors we may also gain information about the

  - the image and kernel of A
  - invertibility of the matrix A in case $m = n$
  - solvability of the system (i.e., the cardinality $|S|$ of the solution set S)

- We will have a look at the following decompositions
  - $A = QR$ (two systems to be solved)
  - $A = P^T LU$ (three systems to be solved)
  - $A = LL^T$ (two systems to be solved)

Never compute the inverse!

- There are only very rare occasions, where an inverse matrix A^{-1} is needed. In most cases, one needs inverse matrix times a vector, i.e., $A^{-1}b$.
- For computing the inverse of an $(n \times n)$-matrix A we have to solve n linear systems:

$$A \cdot [x_1, \ldots, x_n] = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \tag{3}$$

  where the $x_j \in \mathbb{R}^n$ are the unknown columns of the inverse of A. In practice, this is typically done by computing a factorization of A, which is then used to solve these n linear systems in (3) (see, e.g., the LAPACK routine `getri`).

- Thus, never do

```
x = inv(A) @ b
```

  instead of

```
x = solve(A, b).
```

2.2 The Gram-Schmidt Algorithm and the QR decomposition

2.2.1 Projectors

For a fixed vector $x \in \mathbb{R}^n \setminus \{0\}$ we have derived the orthogonal projection onto its span by

$$\text{proj}_x(y) = \frac{x}{\|x\|_2} \cdot \frac{x^\top y}{\|x\|_2} = \frac{xx^\top}{x^\top x} \cdot y =: P_x y,$$

where $P_x := \frac{xx^\top}{x^\top x} \in \mathbb{R}^{n \times n}$.

We now collect some properties of this matrix:

- $\text{Im}(P_x) = \text{span}(x)$ (note that all columns are multiples of x) and thus $\text{rank}(P_x) = 1$
- P_x orthogonally projects a vector $y \in \mathbb{R}^n$ onto the linear subspace $\text{Im}(P_x) = \text{span}(x)$
- Idempotent:

$$P_x^2 := P_x \cdot P_x = \frac{xx^\top}{x^\top x} \cdot \frac{xx^\top}{x^\top x} = \frac{x(x^\top x)x^\top}{(x^\top x)^2} = \frac{xx^\top}{x^\top x} = P_x.$$

  $\rightarrow$ In words: Projecting multiple times is the same as projecting once.

  We make this a definition:

  Definition 2.1 (*Projector*) *A matrix $P \in \mathbb{R}^{n \times n}$ is called **projector** if $P^2 = P$.*

Example 2.2 (*Projectors*)

- $x = (1,0)^\top, P_x = \ldots$
- $x = (1,1)^\top, P_x = \ldots$
- For $\alpha \in \mathbb{R}$ consider $P_\alpha = \begin{pmatrix} 1 & \alpha \\ 0 & 0 \end{pmatrix}$ (exercise sheet)

For any projector $P \in \mathbb{R}^{n\times n}$ we can show:

Lemma 2.3 (*Projector Properties*) *Let $P \in \mathbb{R}^{n\times n}$ be such $P^2 = P$, then*

i) $\operatorname{Im} P = \ker(I-P)$,
ii) $\ker(P) \cap \ker(I-P) = \{0\}$,
iii) $(I-P)$ *is also a projector,*
iv) $\forall y \in \mathbb{R}^n\ \exists_1 v \in \operatorname{Im}(P), r \in \operatorname{Im}(I-P)\colon\ y = v + r$,
with other words, the mapping $\operatorname{Im}(P) \times \operatorname{Im}(I-P) \to \mathbb{R}^n, (v,r) \mapsto v + r$ is bijective.

Proof.

i) $\subseteq$: For $y = Px$ we find $(I-P)y = Px - P^2x = 0$.
$\supseteq$: For $y \in \ker(I-P)$ we have $y = (I-P+P)y = 0 + Py \in \operatorname{Im}(P)$.
ii) Let $Px = 0$ and $(I-P)x = 0$, then $x = (I-P+P)x = 0 + 0 = 0$.
iii) $(I-P)^2 = I - 2P + P^2 = I - 2P + P = I - P$.
iv) Let $y \in \mathbb{R}^n$.
Existence: Choose $v = Py \in \operatorname{Im}(P)$ and $r = (I-P)y \in \operatorname{Im}(I-P)$, then we find

$$v + r = Py + (I-P)y = y.$$

Uniqueness: We assume there is another pair $\tilde{v} \in \operatorname{Im} P = \ker(I-P)$ and $\tilde{r} \in \operatorname{Im}(I-P) = \ker(P)$ with $y = \tilde{v} + \tilde{r}$. Since the kernel of matrix is a linear subspace (therefore closed under linear combinations), also $(v - \tilde{v}) \in \ker(I-P)$ and $(r - \tilde{r}) \in \ker(P)$. Now observe $0 = y - y = v - \tilde{v} + r - \tilde{r}$. Multiplying with P yields

$$0 = P(v-\tilde{v}) + P(r-\tilde{r}) = P(v-\tilde{v}) \quad \Rightarrow \quad (v-\tilde{v}) \in \ker(P) \cap \ker(I-P) = \{0\}.$$

Multiplying with $I-P$ yields

$$0 = (I-P)(v-\tilde{v}) + (I-P)(r-\tilde{r}) = (I-P)(v-\tilde{v}) \quad \Rightarrow \quad (r-\tilde{r}) \in \ker(I-P) \cap \ker(P) = \{0\}.$$

Thus $v = \tilde{v}$ and $r = \tilde{r}$.

Remark:

In view of Lemma 2.3 iv) we say that $\mathbb{R}^n$ is the direct sum of the subspaces $\mathrm{Im}(P)$ and $\mathrm{Im}(I-P)$. Each vector $y \in \mathbb{R}^n$ uniquely splits into the additive components v and r. Due to i) and iii) of this lemma, also the same is true for the subspaces $\mathrm{Im}(P)$ and $\ker(P)$.

We now continue with properties of P_x: Symmetry:

$$P_x^\top = \frac{(xx^\top)^\top}{x^\top x} = \frac{(x^\top)^\top x^\top}{x^\top x} = \frac{xx^\top}{x^\top x} = P_x.$$

In general we define:

Definition 2.4 (*Orthogonal Projector*) *A matrix $P \in \mathbb{R}^{n\times n}$ is called **orthogonal projector** if $P^2 = P$ and $P^\top = P$.*

For any orthogonal projector (such as P_x) one can show:

Lemma 2.5 (*Orthogonal Projector Properties*) *Let $P \in \mathbb{R}^{n\times n}$ be such $P^2 = P$ and $P^\top = P$, then*

$$\mathrm{Im}\, P \perp \mathrm{Im}(I-P).$$

Proof. From Lemma 2.3 and Lemma 1.45 (1.45) as well as the symmetry of P (and thus $I-P$) we find

$$\mathrm{Im}(P) = \ker(I-P) = \ker((I-P)^\top) = \mathrm{Im}(I-P)^\perp.$$

□

In view of Lemma 2.3 iv) we find for an orthogonal projector, that the components $v \in \mathrm{Im}(P)$ and $r \in \ker(P)$ are orthogonal to each other.

Orthogonal Projection with Orthonormal Basis

Let us now consider more than just one vector. In fact, let $q_1, \ldots, q_r \in \mathbb{R}^m$ be orthonormal; thus $r \leq m$. Then let us define the matrices

$$\widehat{Q} := [q_1, \ldots, q_r] \in \mathbb{R}^{m \times r},$$
$$P := P_{q_1,\ldots,q_r} := \widehat{Q}\widehat{Q}^\top \in \mathbb{R}^{m \times m}.$$

Attention: For $r < m$, $\widehat{Q}$ is not an orthogonal matrix and thus $P = \widehat{Q}\widehat{Q}^\top \neq I$ in general. However, for the Gramian matrix which collects all possible pairs of inner products we have $\widehat{Q}^\top \widehat{Q} = I$ ($\widehat{Q}^\top$ is only a left-inverse).

We find the following properties of $P = P_{q_1,\ldots,q_r}$:

- Sum of rank–one (orthogonal) projectors:

$$P = \sum_{j=1}^{r} q_j q_j^\top = \sum_{j=1}^{r} P_{q_j}.$$

- Orthogonal projector:

$$P^2 = \widehat{Q}(\widehat{Q}^\top \widehat{Q})\widehat{Q}^\top = \widehat{Q}\widehat{Q}^\top = P$$
$$P^\top = (\widehat{Q}\widehat{Q}^\top)^\top = (\widehat{Q}^\top)^\top \widehat{Q}^\top = P$$

- P projects on the image of $\widehat{Q}$: In fact, by Lemma 1.47 (1.47) ii) (with $\mathrm{Im}(\widehat{Q}^\top) = \mathbb{R}^r$) we find

$$\mathrm{Im}(P) = \mathrm{Im}(\widehat{Q}\widehat{Q}^\top) = \mathrm{Im}(\widehat{Q}).$$

  In particular: $\mathrm{rank}(P) = r$.

- By Lemma 2.3 i)+iii) and Lemma 1.47(1.47) i) (with $\ker(\widehat{Q}) = \{0\}$) we find

$$\mathrm{Im}(I-P) = \ker(P) = \ker(\widehat{Q}\widehat{Q}^\top) = \ker(\widehat{Q}^\top)$$

- In Lemma 2.5 we recover Lemma 1.45 (1.45):

$$\mathrm{Im}(P) \perp \mathrm{Im}(I-P) \quad \Leftrightarrow \quad \mathrm{Im}(\widehat{Q}) \perp \ker(\widehat{Q}^\top)$$

- By Lemma 2.3 iv) we have

$$\begin{aligned} \forall y \in \mathbb{R}^n \; \exists_1 v \in \mathrm{Im}(P) = \mathrm{Im}(\widehat{Q}), r \in \mathrm{Im}(I-P) = \ker(\widehat{Q}^\top): \\ y &= v + r \\ &= Py + (I-P)y \\ &= \widehat{Q}\widehat{Q}^\top y + (I - \widehat{Q}\widehat{Q}^\top)y \end{aligned}$$

where v is the orthogonal projection of y onto $\mathrm{Im}(\widehat{Q})$ and r the orthogonal residual vector; the components are orthogonal, i.e.,

$$(v, r) = v^\top r = 0.$$

Example 2.6 Let us consider:

$$q_1 = \frac{1}{\sqrt{2}}(1,1,0)^\top, q_2 = \frac{1}{\sqrt{2}}(-1,1,0), y = (0,1,1)^\top \in \mathbb{R}^3$$

Then

$$P = \widehat{Q}\widehat{Q}^\top = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

So that for $y = (y_1, y_2, y_3)^\top \in \mathbb{R}$ we obtain

$$Py = \begin{pmatrix} y_1 \\ y_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},$$

i.e., as expected the orthogonal projection onto the x_1/x_2-plane ($=\mathrm{Im}(\widehat{Q})$).

Orthogonal Projection with Arbitrary Basis

Let $a_1, \ldots, a_n$ be linearly independent vectors in $\mathbb{R}^m$ $(m \geq n)$ and let us put the vectors into a matrix $A = [a_1, \ldots, a_n] \in \mathbb{R}^{m \times n}$.
How to define an orthogonal projector on the image of A?

Let us first recall that $\mathrm{Im}(A) \perp \ker(A^\top)$ by Lemma 1.45 (1.45). Thus we are looking for a matrix $P \in \mathbb{R}^{m \times m}$ such that

$$\forall b \in \mathbb{R}^m\colon\ Pb \in \mathrm{Im}(A) \text{ , i.e.,} Pb = Ax \text{ for some } x \text{ to be determined}$$
$$b - Pb \in \mathrm{Im}(A)^\perp = \ker(A^\top)$$

Combining these two gives

$$0 = A^\top(b - Pb) = A^\top(b - Ax) = A^\top b - A^\top A x.$$

Then by Lemma 1.46 (1.46) i) we have $\ker(A^\top A) = \ker(A) = 0$, so that the Gram matrix $A^\top A$ is invertible, which yields

$$x = (A^\top A)^{-1} A^\top b$$

and the orthogonal projection b onto the image of A is given by

$$Ax = A(A^\top A)^{-1} A^\top b =: Pb.$$

Indeed we find, $P^2 = P$ and $P^\top = P$. Also note that for $A = \widehat{Q}$ this collapses to $P = \widehat{Q}\widehat{Q}^\top$. Further note that this coincides with $P_x = x(x^\top x)^{-1} x^\top$.

—

Remark: The final step to bridge between orthogonal projections and least squares, is to show the intuitive result that orthogonal projections are precisely those projections that produce the smallest residual $(I - P)b$. This is shown later in this course.

Example 2.7 Let us consider:

$$a_1 = (1,1,0)^\top, a_2 = (0,1,0), y = (0,1,1)^\top \in \mathbb{R}^3.$$

We observe that $\mathrm{Im}(A) = \mathrm{Im}(\widehat{Q})$, so that we would expect the same projector as in the example above. First we compute

$$A^\top A = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}.$$

From

$$A^\top A x = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \Leftrightarrow x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

and

$$A^\top A x = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \Leftrightarrow x = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

we find

$$(A^\top A)^{-1} = \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}.$$

And thus

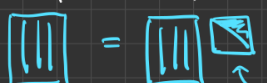
$$P = A(A^\top A)^{-1} A^\top = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

2.2.2 A=QR from the (classical) Gram–Schmidt Algorithm

- **Aim:** ultimately we want a decomposition of A in the following form:

[we "orthogonalize" the columns of A]

$A = QR$



Q: orthogonal (orthonormal) columns

R: upper triangular

$$\Leftrightarrow \quad \text{span}(a_1,\dots,a_n) = \text{span}(q_1,\dots,q_n)$$

($\{q_1,\dots,q_n\}$ orthonormal basis for the column spaces)

$$\Leftrightarrow \quad \begin{aligned} 1&: \; a_1 = r_{11} q_1 \\ 2&: \; a_2 = r_{12} q_1 + r_{22} q_2 \\ &\vdots \\ k&: \; a_k = r_{1k} q_1 + \dots + r_{kk} q_k \end{aligned} \qquad , \; \|q_j\| = 1 \qquad , \; q_j^T q_i = \delta_{ij}$$

$$\Updownarrow$$

$$1: \quad q_1 = \frac{1}{r_{11}} \cdot a_1 \,, \; 1 = \|q_1\| = \frac{1}{|r_{11}|}\|a_1\| \Leftrightarrow r_{11} = \pm \|a_1\|$$

$$2: \quad q_2 = \frac{1}{r_{22}}\left(a_2 - r_{12} q_1\right) = \frac{1}{r_{22}}\left(a_2 - r_{12} q_1\right)$$

$$0 = q_1^T q_2 = \frac{1}{r_{22}}\left(a_2^T q_1 - r_{12}\right) \Leftrightarrow r_{12} = a_2^T q_1$$

$$1 = \|q_2\| = \frac{1}{|r_{22}|} \cdot \|a_2 - r_{12} q_1\|_2 \Leftrightarrow r_{22} = \pm \|a_2 - r_{12} q_1\|_2$$

$$\vdots$$

↑ "GRAM-SCHMIDT ORTHOGONALIZATION ALGORITHM"

Classical Gram–Schmidt Orthogonalization Algorithm

$r_{11} = \|a_1\|$
$q_1 = \frac{a_1}{r_{11}}$

for $k = 2, \ldots, n$ **do**
  for $\ell = 1, \ldots, k-1$ **do**
    $r_{\ell k} = a_k^\top q_\ell$
  end
  $\widetilde{q}_k = a_k - \sum_{\ell=1}^{k-1} r_{\ell k}\, q_\ell \;\; = (I - P_{q_1,\ldots,q_{k-1}})a_k \in \mathrm{Im}(\widehat{Q}_{k-1})^\perp$
  $r_{kk} = \|\widetilde{q}_k\|$
  if $r_{kk} = 0$ **then**
    pick arbitrary $q_k \in \mathrm{Im}(\widehat{Q}_{k-1})^\perp = \ker(\widehat{Q}_{k-1}^\top)$
    // for example by solving $\widehat{Q}_{k-1}^\top q_k = 0$ and normalizing
  end
  else
    $q_k = \frac{\widetilde{q}_k}{r_{kk}}$
  end
end
// For full QR:
Fill up columns of $\widehat{Q} := \widehat{Q}_n$ with $(m-n)$ orthonormal vectors of $\ker(\widehat{Q}_{k-1}^\top)$
Fill up rows of $\widehat{R} := (r_{ij})$ with $(m-n)$ zero rows

Observation: This algorithm successively orthogonalizes the columns of A!

These ideas lead to

Theorem 2.8 (*QR decomposition*) *Let $A \in \mathbb{R}^{m\times n}$ with $m \geq n$, then there exists a matrix $\widehat{Q} \in \mathbb{R}^{m\times n}$ with orthonormal columns and an upper triangular matrix $\widehat{R} \in \mathbb{R}^{n\times n}$ such that*

$$A = \widehat{Q}\widehat{R}.$$

*We call this the **reduced QR decomposition** of A.*

One can extend the columns of $\widehat{Q}$ with orthonormal columns to obtain an orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ and the rows of $\widehat{R}$ by zero rows to obtain a matrix $R = \begin{pmatrix} \widehat{R} \\ 0 \end{pmatrix} \in \mathbb{R}^{m\times n}$ and thereby obtain

$$A = QR.$$

*We call this the **QR decomposition** of A.*

Proof. Sketch: First note that by construction $\operatorname{span}(a_1,\dots,a_k) = \operatorname{span}(q_1,\dots,q_k)$ for all $1 \leq k \leq n$ and $q_1,\dots,q_k$ orthonormal (exercise: verify this by an induction proof).
If $\ker(A) = \{0\}$ (independent columns), then for all $1 \leq k \leq n$ we have

$$a_k \notin \operatorname{span}(a_1,\dots,a_{k-1}) = \operatorname{span}(q_1,\dots,q_{k-1})$$

and thus for the residual of the projection

$$\widetilde{q}_k = (I - P_{q_1,\dots,q_{k-1}})a_k \neq 0$$

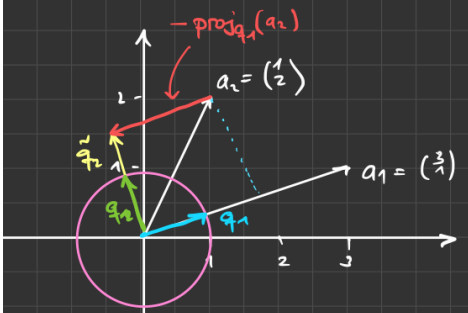
so that $r_{kk} \neq 0$. Thus the classical Gram-Schmidt algorithms produces $\widehat{Q}$ and $\widehat{R}$ so that $A = \widehat{Q}\widehat{R}$.
If columns of A are dependent, then there are k for which

$$a_k \in \operatorname{span}(a_1,\dots,a_{k-1}) = \operatorname{span}(q_1,\dots,q_{k-1})$$

so that the residual is $\widetilde{q}_k = (I - P_{q_1,\dots,q_{k-1}})a_k = 0$, thus we pick an arbitrary vector in $q_k \in \operatorname{Im}(\widehat{Q}_{k-1})^\perp = \ker(\widehat{Q}_{k-1}^\top)$ and continue. □

Example 2.9

Let $A = \begin{pmatrix} 3 & 1 \\ 1 & 2 \end{pmatrix}$ with columns $a_1 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}$ and $a_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$



(a)

$$\tilde{q}_1 := a_1 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \quad q_1 := \frac{\tilde{q}_1}{\|\tilde{q}_1\|} = \frac{1}{\sqrt{10}} \cdot \begin{pmatrix} 3 \\ 1 \end{pmatrix}$$

- Idea: Substract from a_2 the projection of a_2 onto q_1

(b)

$$\tilde{q}_2 := a_2 - \text{proj}_{q_1}(a_2) = a_2 - a_2^T q_1 \cdot q_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix}^T \begin{pmatrix} 3 \\ 1 \end{pmatrix} \cdot \frac{1}{\sqrt{10}} \cdot \begin{pmatrix} 3 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 \\ 2 \end{pmatrix} - \frac{5}{\sqrt{10}} \cdot \frac{1}{\sqrt{10}} \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 1.5 \end{pmatrix}$$

$$q_2 := \frac{\tilde{q}_2}{\|\tilde{q}_2\|} = \underbrace{\frac{1}{\sqrt{2.5}}}_{\frac{\sqrt{2}}{\sqrt{5}} = \frac{2}{\sqrt{10}}} \begin{pmatrix} -0.5 \\ 1.5 \end{pmatrix} = \frac{1}{\sqrt{10}} \cdot \begin{pmatrix} -1 \\ 3 \end{pmatrix}$$

- We first observe:

$$q_1^T q_2 = \frac{1}{\sqrt{10}} \frac{1}{\sqrt{10}} \underbrace{\begin{pmatrix} 3 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ 3 \end{pmatrix}}_{=0} = 0 \quad \checkmark$$

so that $\{q_1, q_2\}$: – orthogonal ✓
– normalized ✓

- Now we write the numbers into matrices:
  - goal: $A = Q \cdot R$, Q orthogonal, R triangular
  - we define: $Q := [q_1, q_2] = \frac{1}{\sqrt{10}} \begin{pmatrix} 3 & -1 \\ 1 & 3 \end{pmatrix}$ ⇝ orthogonal ✓
  - How to define R?

    ↳ from above:

$$r_{11} = \|\tilde{q}_1\| = \sqrt{10}$$

$$r_{12} = q_2^T q_1 = 5/\sqrt{10}$$

$$r_{22} = \|\tilde{q}_2\| = \frac{\sqrt{5}}{\sqrt{2}}$$

$$r_{21} = 0$$

- Let us verify:

$$Q \cdot R = \frac{1}{\sqrt{10}} \begin{pmatrix} 3 & -1 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} \sqrt{10} & 5/\sqrt{10} \\ 0 & \sqrt{5}/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 3 & \frac{1}{2}\cdot 3 - \frac{1}{2} \\ 1 & \frac{1}{2}\cdot 1 + 3\frac{1}{2} \end{pmatrix}$$

$$= \frac{1}{\sqrt{10}} \cdot \begin{pmatrix} 1 & 1/2 \\ 0 & 1/2 \end{pmatrix}$$

$$= \begin{pmatrix} 3 & 1 \\ 1 & 2 \end{pmatrix} = A \checkmark$$

- Extension to higher dimensions and more than 2 vectors:
  - substract the projections onto all previous q_k's

  $\Rightarrow$ thereby we obtain the triangular structure for the coefficients put in the matrix R

Let $m \geq n$.

(1) Factorization
Gram-Schmidt orthogonalization algorithm, Householder reflections or Givens Rotations can be used.

In Python

$$Q, R = \texttt{scipy.linalg.qr(A)}$$

or for the reduced version run

$$\widehat{Q}, \widehat{R} = \texttt{scipy.linalg.qr(A, mode="economic")}$$

(2) Solving
Let us consider the reduced QR decomposition $\widehat{Q}\widehat{R} = A$. Then project b on the image of A by $\widehat{Q}\widehat{Q}^\top b$. Then we obtain the solvable auxiliary problem

$$Ax = \widehat{Q}\widehat{Q}^\top b \quad \Leftrightarrow \quad \widehat{Q}\widehat{R}x = \widehat{Q}\widehat{Q}^\top b \quad \overset{\ker \widehat{Q} = \{0\}}{\Leftrightarrow} \quad \widehat{R}x = \widehat{Q}^\top b.$$

Note that if $b \in \mathrm{Im}(A)$ then $b = \widehat{Q}\widehat{Q}^\top b$, so that in this case we solve the original problem

$$Ax = \widehat{Q}\widehat{Q}^\top b = b.$$

In Python

```
x = scipy.linalg.solve_triangular(R, Q.T @ b)
```

2.3 Gaussian Elimination and the LU Decomposition

(1) Factorization: Row elementary operations

Apply transformations to A (and b), which do not affect the solution x (more precisely the solution set S will not change), to bring A into a simple form and collect all transformations on the fly.

- the factorization **process** is known as *Gaussian Elimination*
- the **result** will be an invertible lower triangular matrix L, some sort of upper triangular matrix U and an orthogonal (more precisely a permutation) matrix P, such that

$$A = P^T LU.$$

(2) Solve: Forward/Backward substitution

$$Ax = b \Leftrightarrow Ux = L^{-1}Pb$$

Example 1: Frobenius Matrices

$$A = \begin{pmatrix} 1 & 3 & 5 \\ 0 & 2 & 3 \\ 2 & 4 & 6 \end{pmatrix}, \; b = \begin{pmatrix} 4 \\ 2 \\ 6 \end{pmatrix} \qquad Ax = b \;\Leftrightarrow\; \begin{array}{rrrl} x_1 & +3x_2 & +5x_3 & = 4 \\ & 2x_2 & +3x_3 & = 2 \\ 2x_1 & +4x_2 & +6x_3 & = 6 \end{array}$$

We apply the transformation: *1) scaling one row and adding it to another row*

$$\left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 2 & 4 & 6 & 6 \end{array}\right) \overset{\text{III'=III-2I}}{\Leftrightarrow} \left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 0 & -2 & -4 & -2 \end{array}\right) \overset{\text{III'=III+II}}{\Leftrightarrow} \left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 0 & 0 & -1 & 0 \end{array}\right)$$

$$A|b \xrightarrow[L_1\cdot|]{} L_1A \mid L_1b \xrightarrow[L_2\cdot|]{} L_2L_1A \mid L_2L_1b$$

$$\begin{array}{rcl} \text{III)} & \Rightarrow & x_3 = 0 \\ \text{II)} & \Rightarrow & x_2 = 1 \\ \text{I)} & \Rightarrow & x_1 = 1 \end{array} \quad \Rightarrow \quad x = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

Observation:
The transformations L_i can be written as:

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}, \; L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Let us verify this for the first step

$$L_1A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 3 & 5 \\ 0 & 2 & 3 \\ 2 & 4 & 6 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 5 \\ 0 & 2 & 3 \\ 0 & -2 & -4 \end{pmatrix}, \quad L_1b = \begin{pmatrix} 4 \\ 2 \\ -2 \end{pmatrix}$$

And since the transformations L_i are injective (independent columns), after the first step, we obtain an equivalent linear system:

$$Ax = b \;\Leftrightarrow\; L_1Ax = L_1b$$

All in all:

By defining the upper triangular matrix $U := L_2 L_1 A$ and the lower triangular matrix $L := L_1^{-1} L_2^{-1}$ we find

$$A = LU.$$

In general:

If "A permits", we obtain:

$$\underbrace{(L_{n-1} \cdots L_2 L_1)}_{\text{(lower triangular + invertible)}} \cdot A = U \quad \Leftrightarrow \quad A = LU, \quad L := (L_{n-1} \cdots L_2 L_1)^{-1}.$$

Note:

- The lower triangular structure in the matrices L_j is obtained by following a "top to bottom" approach.
- Thereby we also make sure that we ultimately obtain an "upper triangular" system.

Frobenius Matrices

$$\underbrace{\begin{pmatrix} 1 & 0 & & & & \cdots & 0 \\ 0 & 1 & 0 & & & & \vdots \\ & 0 & \ddots & 0 & & & \\ & & 0 & 1 & 0 & & \\ & & 0 & \ell_{j+1,j} & \ddots & 0 & \\ \vdots & & \vdots & \vdots & 0 & 1 & 0 \\ 0 & \cdots & 0 & \ell_{m,j} & 0 & 0 & 1 \end{pmatrix}}_{=:L_j} \begin{pmatrix} -- a_1 -- \\ -- a_2 -- \\ \vdots \\ -- a_j -- \\ \vdots \\ -- a_m -- \end{pmatrix} = \begin{pmatrix} -- a_1 -- \\ \vdots \\ -- a_j -- \\ \ell_{j+1,j} a_j + a_{j+1} \\ \vdots \\ \ell_{m,j} a_j + a_m \end{pmatrix}$$

- If L_j is defined in this way, then $\ell_{i,j}$ is the multiple for the j-th row which is then added to the i-th row.
- Since we want to produce zeroes underneath the diagonal element in our system, we choose

$$\ell_{i,j} = -\frac{a_{ij}}{a_{jj}} \ (a_{jj} \neq 0).$$

$\rightarrow$ We will now illuminate properties of such Frobenius matrices which come in handy when analyzing the Gaussian elimination procedure!

Properties

Lemma 2.10 *Let $\ell_j := (0,\dots,0,\ell_{j+1,j},\dots,\ell_{m,j})^\top \in \mathbb{R}^m$, $e_j \in \mathbb{R}^m$ be the j-th unit vector and $I \in \mathbb{R}^{m\times m}$ be the identity matrix. Then the matrix*

$$L_j := I + \ell_j e_j^\top \in \mathbb{R}^{m\times m}$$

satisfies:

i) *The matrix L_j is an invertible lower triangular matrix.*

ii) *The inverse of L_j is given by $L_j^{-1} = I - \ell_j e_j^\top \in \mathbb{R}^{m\times m}$.*

iii) *For $i \le j$ it holds that $L_i L_j = I + \ell_j e_j^\top + \ell_i e_i^\top$ and $L_i^{-1} L_j^{-1} = I - \ell_j e_j^\top - \ell_i e_i^\top$.*

Proof. i) First note that $\ell_j e_j^\top$ is a lower triangular matrix with zeroes on its diagonal because $\ell_{i,j} = 0$ for $i \le j$. Therefore L_j is a lower triangular matrix with ones on its diagonal and thus invertible (see, e.g., $\det(L_j) = 1 \neq 0$).

ii) Since the inverse matrix is unique it is sufficient to show that $L_j(I - \ell_j e_j^\top) = I$. By inserting the definition we find that

$$\begin{aligned} L_j(I - \ell_j e_j^\top) &= (I + \ell_j e_j^\top)(I - \ell_j e_j^\top) = I + \ell_j e_j^\top - \ell_j e_j^\top - \ell_j e_j^\top \ell_j e_j^\top \\ &= I - \ell_j (e_j^\top \ell_j) e_j^\top \\ &= I, \end{aligned}$$

where we have exploited $e_j^\top \ell_j = 0$ which follows from $\ell_{j,j} = 0$.

iii) We insert definitions and compute the products. For the first product we get

$$\begin{aligned} L_i L_j &= (I + \ell_i e_i^\top)(I + \ell_j e_j^\top) = I + \ell_i e_i^\top + \ell_j e_j^\top + \ell_i e_i^\top \ell_j e_j^\top \\ &= I + \ell_i e_i^\top + \ell_j e_j^\top + \ell_i (e_i^\top \ell_j) e_j^\top \\ &= I + \ell_i e_i^\top + \ell_j e_j^\top, \end{aligned}$$

where we have exploited $e_i^\top \ell_j = 0$, which follows from $\ell_{i,j} = 0$ for all $i \le j$. The second statement follows along the same lines.

□

Example 2: Permutation Matrices

We now additionally allow for a second type of transformation: *2) row swap (= partial pivoting)*

Let us now consider an example where the "top-to-bottom" approach would fail:

$$\left(\begin{array}{ccc|c} 0 & 2 & 3 & 2 \\ 1 & 3 & 5 & 4 \\ 2 & 4 & 6 & 6 \\ 1 & 5 & 8 & 6 \end{array}\right) \begin{array}{c} \text{I} \leftrightarrow \text{II} \\ \Leftrightarrow \end{array} \left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 2 & 4 & 6 & 6 \\ 1 & 5 & 8 & 6 \end{array}\right) \begin{array}{c} \text{III'=III-2I} \\ \Leftrightarrow \\ \text{IV'=IV-I} \end{array} \left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 0 & -2 & -4 & -2 \\ 0 & 2 & 3 & 2 \end{array}\right)$$

$$P_{12} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad L_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -2 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{array}{c} \text{III'=III+II} \\ \Leftrightarrow \\ \text{IV'=IV-II} \end{array} \left(\begin{array}{ccc|c} 1 & 3 & 5 & 4 \\ 0 & 2 & 3 & 2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \begin{array}{l} \Rightarrow x_1 = 1 \\ \Rightarrow x_2 = 1 \\ \Rightarrow x_3 = 0 \end{array}$$

$$L_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{pmatrix}$$

Remark: The multiples from the Frobenius matrices can be stored in–place. We even store them with reverse sign, because we are interested in their inverses.

All in all:

$$L_2 L_1 P_{12} A = U \quad \text{is not upper triangular}$$

- We observe that U is not a "perfect" upper triangular matrix.
  – The structure of U is called Row Echolon Form (REF).
  – Any matrix $A \in \mathbb{R}^{m \times n}$ can be transformed into a matrix $U \in \mathbb{R}^{m \times n}$ of REF with the help of matrices P_{jk_j}, L_j from above.

- In order to reduce rounding errors one chooses the element in the current column which has largest absolute value as the pivot element by permuting rows first.
  $\rightarrow$ We do not further study stability issues in this course.

Permutation Matrices

Definition 2.11 (*Permutation Matrix*) *A matrix $P \in \mathbb{R}^{m \times m}$ is called **permutation matrix**, if it has exactly one entry "1" in each row and column and zero otherwise.*

In each step of the Gaussian elimination (with pivoting) we only swap two rows at a time: The matrix

$$P_{ik} = \begin{pmatrix} 1 & 0 & & & & \cdots & & & & 0 \\ 0 & \ddots & & & & & & & & \\ & & 1 & & & & & & & \\ & & & 0 & & \cdots & 0 & 1 & & \\ & & & & 1 & & & 0 & & \vdots \\ \vdots & & & \vdots & & \ddots & & \vdots & & \\ & & & 0 & & & 1 & & & \\ & & & 1 & 0 & \cdots & & 0 & & \\ & & & & & & & & \ddots & 0 \\ 0 & & & & \cdots & & & & 0 & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \leftarrow\ k-th \\ \\ \\ \end{matrix}$$

swaps rows k and i, when multiplied from the left to a matrix.

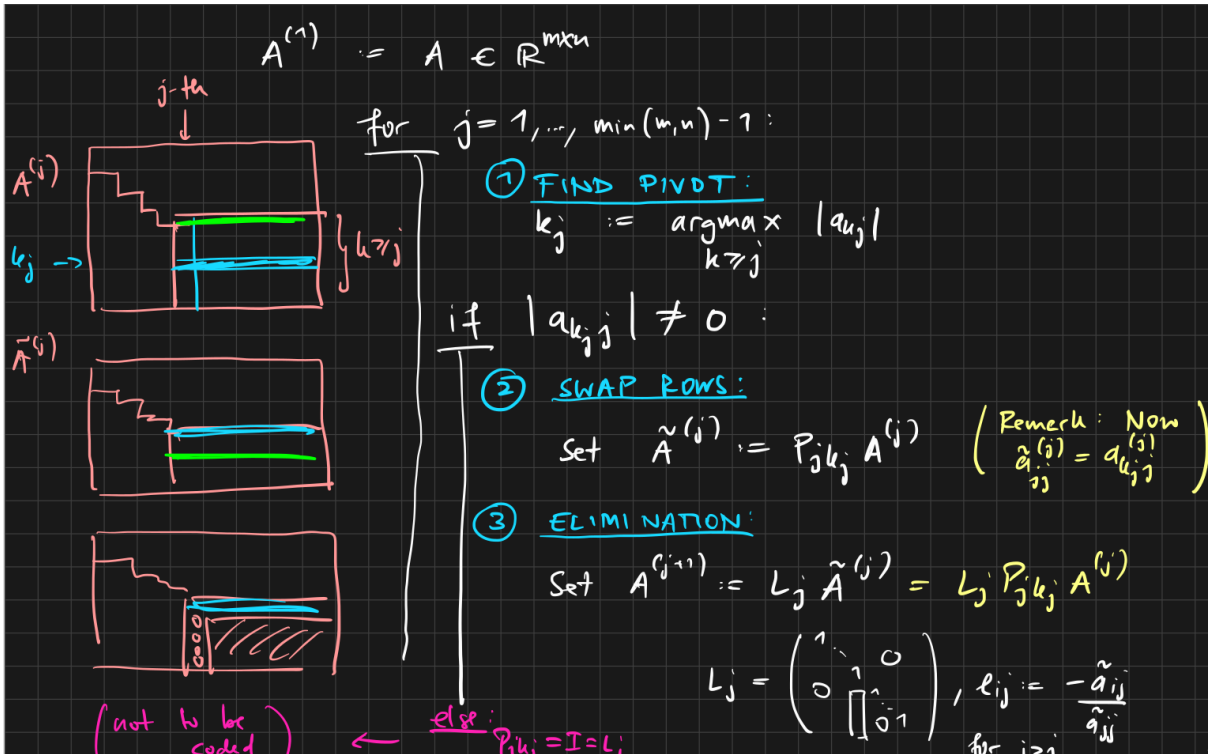
$\rightarrow$ *We illuminate further properties in the exercises.*

Example 2.12 (*Permutation Matrix)* Consider $P := P_{23} \in \mathbb{R}^{3\times 3}$, i.e.,

$$P = P_{23} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

- Multiply from the left to a matrix $A \in \mathbb{R}^{3\times 3}$ and observe how it acts on the rows.
- Observe that $P^\top P = I$, so that $P^\top = P^{-1}$. Even $P^\top = P$, i.e., P is self-inverse.
- Derive its CSR format
  - `data = [1,1,1]`
  - `indices = [0,2,1]` ← *the only relevant information*
  - `indptr = [0,1,2,3]`

Algorithm: Elimination with row pivoting



Remark: We can work *in-place* and store numbers for L and U in the the same array.

In-place Gaussian Elimination with Row Pivoting (for $m = n$)

INPUT: $A \in \mathbb{R}^{n \times n}$ (and $b \in \mathbb{R}^n$)
OUTPUT: LU decomposition $PA = LU$ (and if $Ax = b$ is uniquely solvable the solution $x \in \mathbb{R}^n$)

```
# FACTORIZATION
initialize piv = [1,2,...,n]
for j = 1,...,n − 1 do
    # Find the j-th pivot
    k_j := arg max_{k≥j} |a_kj|
    if a_{k_j j} ≠ 0 then
        # Swap rows
        A[k_j,:] ↔ A[j,:]
        # by hand we also transform b on the fly
        b[k_j] ↔ b[j]
        piv[k_j] ↔ piv[j]
        # Elimination
        for k = j + 1,...,n do
            ℓ_kj := a_kj / a_jj
            a_kj = ℓ_kj
            for i = j + 1,...,n do
                a_ki = a_ki − ℓ_kj a_ji
            end
            # by hand we also transform b on the fly
            (b_k = b_k − ℓ_kj b_j)
        end
    end
end
# SOLVE
...
```

As in the algorithm consider $A \in \mathbb{R}^{n\times n}$. Then the algorithm eventually leads to

$$U := A^{(n)} = L_{(n-1)}P_{(n-1)k_{(n-1)}} \dots L_3P_{3k_3}L_2P_{2k_2}L_1P_{1k_1}A.$$

By construction of the algorithm, $A^{(n)} =: U$ has row echelon form (no rigorous proof provided in this course).

Question: Can we group all L_i and P_{ik_i} in such a way that $PA = LU$?

Lemma 2.13 *Let $m \in \mathbb{N}$. Let $P_{ik_i} \in \mathbb{R}^{m\times m}$ be the permutation matrix which results from interchanging the i-th and k_i-th column of the identity matrix in $\mathbb{R}^{m\times m}$, where $k_i \geq i$. Further for $\ell_j := (0,\dots,0,\ell_{j+1,j},\dots,\ell_{m,j})^\top \in \mathbb{R}^m$ and the j-th unit vector $e_j \in \mathbb{R}^m$, let $L_j := I + \ell_j e_j^\top \in \mathbb{R}^{m\times m}$. Then show that for all $1 \leq j < i \leq k_i \leq m$ we have*

$$P_{ik_i}L_j = \widehat{L}_j P_{ik_i}$$

where $\widehat{L}_j := I + (P_{ik_i}\ell_j)e_j^\top$.

Proof. We find

$$\begin{aligned}
P_{ik_i}L_j &= P_{ik_i}\left(I + \ell_j e_j^\top\right)\\
&= P_{ik_i} + P_{ik_i}\ell_j e_j^\top\\
&= P_{ik_i} + P_{ik_i}\ell_j e_j^\top P_{ik_i}^\top P_{ik_i}\\
&= (I + P_{ik_i}\ell_j e_j^\top P_{ik_i}^\top)P_{ik_i}\\
&= (I + P_{ik_i}\ell_j (P_{ik_i}e_j)^\top)P_{ik_i}.\\
&= (I + P_{ik_i}\ell_j e_j^\top)P_{ik_i}.
\end{aligned}$$

Since $j < i \leq k_i$ we find that $P_{ik_i}e_j = e_j$, since only zeroes are swapped. □

By applying this result multiple times, we obtain

$$\underbrace{(\hat{L}_{n-1}\cdots\hat{L}_2\hat{L}_1)}_{=:\widehat{L}}\underbrace{(P_{(n-1)k_{(n-1)}}\cdots P_{2k_2}P_{1k_1})}_{=:P}\cdot A = U$$

$$\overset{L:=\widehat{L}^{-1}}{\Leftrightarrow}$$

$$PA = LU.$$

Summary: **Row echelon form and LU decomposition**

Definition 2.14 (*REF*)

a) ***Row elementary operations*** *are*
   1) *add a nonzero multiple of one row to another*
   2) *swap two rows*

b) *A matrix is in* ***row echelon form (REF)*** *when it satisfies the following conditions:*
   1) *The first non-zero element in each row (called the leading entry) is in a column to the right of the leading entry in the previous row.*
   2) *Rows with all zero elements, if any, are below rows having a non-zero element.*

By applying these types of operations we find:

Theorem 2.15 (*LU Decomposition*) *Every matrix $A \in \mathbb{F}^{m\times n}$ can be transformed to REF by row elementary operations. Thus there is a matrix $U \in \mathbb{F}^{m\times n}$ in REF, a lower triangular matrix $L \in GL(m,\mathbb{F})$ with $\ell_{ij} = 0, \forall j > i$, and $\ell_{ii} = 1, \forall i$, and a permutation matrix $P \in GL(m,\mathbb{F})$ with exactly one entry "1" in each row and column and zero otherwise, such that*

$$P \cdot A = L \cdot U.$$

Since triangular matrices are invertible if and only if the diagonal elements are nonzero, we find for the square case $m = n$, that:

Corollary 2.16 (*Invertibility of A*) *A matrix $A \in \mathbb{F}^{n\times n}$ is invertible, if and only if its REF $U \in \mathbb{F}^{n\times n}$ has a nonzero diagonal, i.e, $u_{ii} \neq 0, \forall i = 1, \ldots, n$.*

(1) Factorization: `Lu, Piv = scipy.linalg.lu_factor(A)`

- determine factors L, U and permutation matrix P such that $LU = PA$
- the factors L, U are compactly stored in one matrix `Lu` $\in \mathbb{R}^{n \times n}$ of the same size as A and the permutation matrix P in CSR format as integer vector `Piv` $\in \mathbb{N}^n$.

(2) Solution: `x = scipy.linalg.lu_solve((Lu, Piv), b)`

(2.1) permute right-hand side $\bar{b} = Pb$

(2.2) solve lower triangular system $Lz = \bar{b}$ for z (forward substitution)

(2.3) solve REF system $Ux = z$ for x (backward substitution)

Both, (1) and (2), are combined in the routine

`x` $=$ `scipy.linalg.solve(A,b)`.

2.3.1 Identify the number of solutions from the REF

First, by inserting the LU decomposition we obtain

$$S := \{x \in \mathbb{R}^n : \ Ax = b\} \overset{A=P^TLU}{=} \{x \in \mathbb{R}^n : \ P^TLUx = b\} = \{x \in \mathbb{R}^n : \ Ux = \underbrace{L^{-1}Pb}_{=:z}\}.$$

Then we find for the three possible cases of the solution set (exercise):

1) U does not have a zero row (i.e., $m \leq n$)

   1.1) $m = n$: Then U is invertible and $|S| = 1$ with $x = U^{-1}L^{-1}Pb$

   1.2) $m < n$: Then has dependent columns but $\mathrm{Im}(U) = \mathbb{R}^m$, so that $|S| = \infty$
   Note: The m rows of U are independent. Thus $m = \dim \mathrm{Im}\, U^\top$. Also from dimension formula $\dim \ker U^\top = m - \dim \mathrm{Im}\, U^\top = 0$, so that $\ker U^\top = \{0\}$ and thus $\mathrm{Im}(U) = (\ker U^\top)^\perp = \{0\}^\perp = \mathbb{R}^m$. ,

2) U has at least one zero row

   2.1) For all zero rows in U, the transformed right-hand side z is also zero there:
   $\Rightarrow$ $Ux = z$ has infinitely many solutions, i.e., $|S| = \infty$
   (Note: $0x_1 + \cdots + 0x_n = z_i = 0$ is true for any x)

   2.2) else: $|S| = 0$
   (Note: $0x_1 + \cdots + 0x_n = z_i \neq 0$ false for any x)

2.4 The Cholesky Decomposition

For symmetric and positive definite matrices $A \in \mathbb{R}^{n\times n}_{\text{spd}} \subset \mathrm{GL}_n(\mathbb{R})$ we obtain the following improvement:

Theorem 2.17 *We have the equivalence*

$$A \in \mathbb{R}^{n\times n}_{spd} \quad \Leftrightarrow \quad \exists_1\, L \in \mathbb{R}^{n\times n} \text{lower triangular, } \ell_{ii} > 0\colon \quad A = LL^\top.$$

- The Decomposition $A = LL^\top$ is called the **Cholesky decomposition of** A.
- Named after the french Mathematician André-Louis Cholesky (1875-1918) who developed this decomposition during his surveying work to solve the normal equation $A^\top Ax = A^\top b$.
- We can derive an algorithm to compute the factor L.

- **Solving using** $A = LL^\top$

$$Ax = b \quad \Leftrightarrow \quad LL^\top x = b \quad \Leftrightarrow \quad Lz = b, \quad L^\top x = z \quad \text{(forward/backward Subst.)}$$

In Python:

(1) Factorization: `L, lower = scipy.linalg.cho_factor(A)`

(2) Solution: `x = scipy.linalg.cho_solve((L, lower), b)`

Numerical Comparison: LU vs. Choleskly

- **Computational Costs:**
  The Cholesky decomposition is roughly twice as fast as Gaussian elimination (in terms of number of floating point operations). Clearly, we only need to compute one instead of two factors.

- **Stability** (="robustness against rounding errors") :
  - Gaussian Elimination: Is only stable if a pivoting strategy is applied.
  - Cholesky: Is stable even without pivoting.
    (To avoid square roots, one can compute the LDL decomposition)

All in all:

For symmetric and positive definite matrices (of moderate size),
the Cholesky factorization is the preferred algorithm!