Basic Linear Algebra

Definitions

Here I want to introduce some notation that I use (which may be non-standard) and some ideas which are basic and well known, but helpful to keep in mind when reading the ML literature. The vector $\mathbf{u}$ with n elements is either a horizontal or vertical collection of n numbers. In the former $\mathbf{u}$ is *row* vector (and I note its dimension $\mathbf{u} \equiv (1 \times n)$) and the latter it is a *column* vector ($\mathbf{u} \equiv (n \times 1)$). By default all vectors are column vectors. A column vector $\mathbf{u}$ can be written as a row by tranposing (ie $\mathbf{u}^T$). We define the two vector-vector products:

Vector Products

The *dot product* of two vectors $\mathbf{u}$ and $\mathbf{v}$ is the scalar

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^T \mathbf{v} = \sum \mathbf{u}_i \mathbf{v}_i \tag{1}$$

The *vector product* of two vectors $\mathbf{u}$ and $\mathbf{v}$ is the matrix

$$\mathbf{u}\mathbf{v}^T = \begin{pmatrix} u_1 v_1 & u_1 v_2 & \dots & u_1 v_n \\ u_2 v_1 & u_2 v_2 & \dots & u_2 v_n \\ \vdots & \vdots & \ddots & \vdots \\ u_n v_n & u_1 v_n & s & u_n v_n \end{pmatrix} \tag{2}$$

The vectors $\mathbf{u}$ and $\mathbf{v}$ are *orthogonal* if their dot products is zero, ie $\mathbf{u} \cdot \mathbf{v} = 0$.

The $m \times n$ matrix A, where

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{pmatrix} \tag{3}$$

is often decomposed into its m rows

$$A = \begin{pmatrix} \mathtt{r}(A)_1 \\ \mathtt{r}(A)_2 \\ \vdots \\ \mathtt{r}(A)_m \end{pmatrix} \tag{4}$$

or n columns

$$A = \begin{pmatrix} \mathtt{c}(A)_1 & \mathtt{c}(A)_2 & \dots & \mathtt{c}(A)_n \end{pmatrix} \tag{5}$$

so as to better represent matrix operations.

Matrix Products

There are a number of ways to represent matrix-vector and matrix-matrix products, and different representations are usefull in different contexts. Here I will go through some of the main ideas. It is good to go through each a couple of times using a (2×2) or (3×3) to convince yourself of the structure of the following products.

The matrix-vector product $A\mathbf{x}$ is most commonly viewed as the weighted sum of the columns of A ie

$$A\mathbf{x} = \sum x_i \mathtt{c}(A)_i \tag{6}$$

or it can be interpreted as a sequence of dot products using the *rows* of A

$$A\mathbf{x} = \begin{pmatrix} \mathtt{r}(A)_1 \cdot \mathbf{x} \\ \mathtt{r}(A)_2 \cdot \mathbf{x} \\ \vdots \\ \mathtt{r}(A)_N \cdot \mathbf{x} \end{pmatrix} \tag{7}$$

As might be expected, the vector-matrix product $x^T A$ is a sequence of dot products using the *columns* of A

$$x^T A = \left(\mathbf{x}^T \cdot \mathtt{c}(A)_1, \mathbf{x}^T \cdot \mathtt{c}(A)_2, \ldots, \mathbf{x}^T \cdot \mathtt{c}(A)_n \right) \tag{8}$$

Matrix products AB where $A \equiv (m \times n)$ and $B \equiv (n \times m)$ are a matrix of dot products

$$AB = \begin{pmatrix} \mathtt{r}(A)_1 \cdot \mathtt{c}(B)_1 & \mathtt{r}(A)_1 \cdot \mathtt{c}(B)_2 & \cdots & \mathtt{r}(A)_1 \cdot \mathtt{c}(B)_m \\ \mathtt{r}(A)_2 \cdot \mathtt{c}(B)_1 & \mathtt{r}(A)_2 \cdot \mathtt{c}(B)_2 & \cdots & \mathtt{r}(A)_2 \cdot \mathtt{c}(B)_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathtt{r}(A)_m \cdot \mathtt{c}(B)_m & \mathtt{r}(A)_1 \cdot \mathtt{c}(B)_m & \cdots & \mathtt{r}(A)_m \cdot \mathtt{c}(B)_m \end{pmatrix} \tag{9}$$

This representation is the most common way for me to work towards proofs of various matrix properties. It is also usefull to note that matrix-matrix products can be seen as a collection of the matrix-vector or vector-matrix products we have just seen. The matrix-matrix product AB is a row of matrix-vector products using the columns of B

$$AB = \left(A\mathtt{c}(B)_1, A\mathtt{c}(B)_2, \ldots, A\mathtt{c}(B)_n \right) \tag{10}$$

or it is a column of vector-matrix products using the rows of B

$$AB = \begin{pmatrix} \mathtt{r}(A)_1 B \\ \mathtt{r}(A)_2 B \\ \vdots \\ \mathtt{r}(A)_n B \end{pmatrix} \tag{11}$$

All of these interpretations will be used in the following notes (I think).

Column Space, Rank and Independance

Return the matrix-vector product $A\mathbf{x}$, which we saw is the weighted sum of the columns of A. The set of all possible combinations of the columns defines the *column space* of A - that is, all possible outputs of the operation $A\mathbf{x}$. This idea is coupled to the more general idea of a *vector space*. More specifically, the *vector space* generated by a set of n vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ is the set of all possible linear combinations of the $\mathbf{u}_i$. If we combine these vectors to form the columns of a matrix A, the we get the linear combination from the above matrix-vector product $A\mathbf{x}$. The column space of a matrix A then is the *vector space* generated by its columns. This *vector space*, which is the *column space* of A is often also called the *range* of A, ie

$$\text{range}(A) = \{\mathbf{u} \in \mathcal{R}^n \text{ where there exists } \mathbf{x} \text{ with } A\mathbf{x} = \mathbf{u}\} \tag{12}$$

since it defines which vectors can be produced by summing the columns in A. The *null space* is a particular set of non-zero weights $\mathbf{x}$ which produce the zero vector, ie

$$\text{null}(A) = \{\text{all } \mathbf{u} \text{ where } A\mathbf{u} = 0\} \tag{13}$$

Not all matrices have a non-empty null space. If such an $\mathbf{u}$ exists, then at least one column can be written as a linear combination of at least one of the others. This has significant implications - it means that one of the rows in A is a redundant piece of information. If no such combination exists - that is, the null space is empty, then the columns are *linearly independant* - no columns can be written as a combination of the others. This concept is again common more generally to sets of vectors - ie a set of vectors is linearly independant if no linear combination of the vectors produces a zero vector. Spaces which are *not* linearly independant include redundant vectors.

The *column rank* of matrix A is the largest number of independant columns in A. If all columns are independant, the matrix has *full rank*. It is denoted colrank (A). The *row rank* is defined similarly, but is used much less frequently - the *rank* of a matrix rank (A) usually refers to the column rank. if $A \equiv (n \times m)$ matrix as m independant vectors then colrank $(A) =$ rank $= m$, which is also the number of columns of A. Such a matrix is called *full rank*. Some examples would be good.

Consider a matrix A and some set of vectors $\begin{pmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_n\end{pmatrix}$. For every possible combination of the columns of A (ie every $\mathbf{v} = A\mathbf{u}$), it may be possible to choose some combination of the vectors $\mathbf{u}_i$ that produces the same $\mathbf{v}$, and vica versa. In this sense, the set of vectors $\begin{pmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_n\end{pmatrix}$ can substitute for the matrix A, since they can both generate the exact same output (ie they have the same range). This might be useful if the vectors $\mathbf{u}_i$ had properties that were more desireable than the columns of A. Indeed, this is done very commonly in numerical methods in linear algebra. If we want to do this substitution however, we shouldnt choose any redundant vectors in the $\mathbf{u}_i$ - we should only use *linearly independant* vectors. Such a set of vectors is said to form a *basis* for A.

Special Matrices

There are a number of matrices with special structure which are found repeatedly in numerical analysis and machine learning. These are diagonal matrices, triangular matrices, symmetric matrices, orthogonal matrices and symmetric positive definite matrices.

Diagonal Matrices

Diagonal Matrices are denoted D and are all zero except the main diagonal, ie $D_{ij} = 0$ if $i \neq j$. Diagonal matrices are one of two types of *elementary matrices*

- Diagonal or Scaling matrices
- Permutation matrices

though we will delay discussion of permulation matrices until a little later. It is good to note that as elementary matrices, both diagonal and permulation matrices should be seen as matrices which perform basic transformations of other matrices, such as scaling columns/rows or by switching rows/columns around. Diagonal matrices do that former - ie *Diagonal matrices are a scaling operation*. Diagonal matrices are denoted D, and permutation matrices P. A general elementary matrix is denoted E. Multiplication by an elementary matrix E transforms either rows or columns of its operand depending on whether we pre- or post- multiply (ie EA or AE).

Operation	Effect
pre-multiply (ie EA)	row transformation
post-multiply (ie AE)	column transformation

Here, put in some examples of elementary matrices.

As with other *elementary matrices*, pre-multiplying is a row transformation, post-multiplying is a column transformation. To see this, recognise the matrix product as a matrix of dot products. In

the case of pre-multiplication, where D is diagonal with $D_{ii} = di$

$$\begin{aligned}(DA)_{ij} &= \mathtt{r}(D)_i \cdot \mathtt{c}(A)_j \\ &= D_{ii}A_{ij} \\ &= d_i A_{ij}\end{aligned}$$

where the second line occurs because all elements in $\mathtt{r}(D)_i = 0$ except the i^{th} element, so only the i^{th} element in $\mathtt{c}(A)_i$ (ie A_{ij}), survives the dot product. The result is the whole of the i^{th} row of A is multiplied through by d_i. The analgous result happens for post-multiplication. If A is *both* pre and post multiplied we will end up with all pairwise combinations of the diagonals D_{ii}, ie

$$(DAD)_{ij} = d_i d_j A_{ij} \tag{14}$$

This is usefull in particular with correlation and covariance matrices.

A particular case of the diagonal matrix is the identity matrix I, where $D_{ii} = 1$ for all i. This scaling matrix has the property of leaving the matrix A unchanged after either pre or post multiplication.

Triangular Matrix

Triangular matrices are ubiquitious in computation linear algebra, because they can be solved with little effort. Matrices can be upper or lower triangular:

- *Upper Triangular:* U is upper triangular if all $U_{ij} = 0$ when $i > j$
- *Lower Triangular:* L is lower triangular if all $L_{ij} = 0$ when $i < j$

I often refer to lower and upper as being types of triangular matrices. Assume everywhere that T is a triangular matrix of some type, and U/L are upper/lower triangular matrices unless otherwise specified. Triangular matrices (aside from diagonal matrices, which are of course both upper and lower triangular) are very common because the systems of equations they represent are very easy to solve. Note that

- If T_1, T_2 have the same type, then the sum $T = T_1 + T_2$ also has the same type.
- If T_1, T_2 have the same type, then the product $T = T_1 T_2$ also has the same type.

Symmetric Matrix

Symmetric Matrices have corresponding elements across the main diagonal equal, ie $S_{ij} = S_{ji}$. This simple structure results in many useful properties, and many important matrices, such as correlation matrices are symmetric.

Orthogonal Matrix

Orthogonal Matrices are matrices made up of orthogonal columns. Remember that two vectors $\mathbf{u}$ and $\mathbf{v}$ are orthogonal if $\mathbf{u} \cdot \mathbf{v} = 0$. If a matrix Q is orthogonal (often Q is used to represent orthogonal matrices) $\mathtt{c}(Q)_i \cdot \mathtt{c}(Q)_j = 0$ for any columns i, j. The special case Q where $\mathtt{c}(Q)_i \cdot \mathtt{c}(Q)_i = 1$ is referred to as *orthonormal*. Orthogonal matrices have excellent numerical properties, so many common numerical methods rely on finding orthogonal matrices. The critical feature of orthogonal matrices is the product $Q^T Q = I$, since the ij^{th} element in the product

$$\begin{aligned}(Q^T Q)_{ij} &= \mathtt{c}(Q)_i^T \cdot \mathtt{c}(Q)_j \\ &= 0 \text{ if } i \neq j\end{aligned}$$

The critical point is that its transpose (which is easy to find) is its inverse. We will discuss inverses shortly.

Symmetric Positive Definite Matrix

Symmetric Positive Definite Matrices or *SPD* matrices are common and incredibly useful in both numerical and analytic work. An SPD matrix is a special case of a symmetric matrix (ie if S is SPD it is symmetric, though the converse is not necessarily true), where the *quadratic form* $x^T S x > 0$ is positive for any x. The *quadratic form* is a quantity that reappears a number of times in different contexts, so we should look at it briefly. The quadratic form is a function of x, and is essentially a weighted sum of squares of x where the elements in S make up the weights. For example the 2×2 case

$$\begin{aligned} x^T A x &= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= x_1^2 A_{11} + x_1 x_2 \left(A_{12} + A_{21} \right) + x_2^2 \end{aligned}$$

More generally, the vector-matrix-vector product $\mathbf{y}^T A x$, which is a more general expression than the quadratic form, can be written as a sum of all elements in A in the form

$$\mathbf{y}^T A x = \sum_i \sum_j A_{ij} y_i x_j \tag{15}$$

which is shown below as it is a quite usefull algebra exercise

$$\begin{aligned} \mathbf{y}^T A x &= \left(\mathbf{y}^T \mathsf{c}\left(A\right)_1, \ \ldots, \ \mathbf{y}^T \mathsf{c}\left(A\right)_n \right) \mathbf{x} \\ &= x_1 \left(y_1 A_{11} + y_2 A_{21} + \ldots + y_n A_{n1} \right) + \ldots + x_n \left(y_n A_{1n} + y_2 A_{2n} + \ldots + y_n A_{nn} \right) \end{aligned}$$

where each element A_{ij} appears once in the sum in an expression $A_{ij} y_i x_j$, which gives the above identity. The special case here, where $\mathbf{y} = \mathbf{x}$ gives

$$\mathbf{x}^T A x = \sum_i \sum_j A_{ij} x_i x_j \tag{16}$$

Last, a common matrix expression in statistics/ML is the matrix $S = A^T A$, which is often called the Grammian. It is a particularly useful symmetric positive definite matrix. It is symmetric, since

$$\begin{aligned} S^T &= \left(A^T A \right)^T \\ &= A^T \left(A^T \right)^T \\ &= A^T A \\ &= S \end{aligned}$$

and positive definite, since

$$\begin{aligned} \mathbf{x}^T S \mathbf{x} &= \mathbf{x}^T A^T A \mathbf{x} \\ &= A \mathbf{x}^T A \mathbf{x} \\ &= \mathbf{u}^T \mathbf{u} \\ &= \sum u_i^2 \geq 0 \end{aligned}$$

where $\mathbf{u} = A\mathbf{x}$. This actually guarantees *semi-positive definiteness* only, but if the columns of A are independant, then $S = A^T A$ is positive definite. We will look at independance in the next section,

but note that being *positive definite* requires the following is non-zero (it cant be negative)

$$\begin{aligned}\mathbf{x}^T A^T A\mathbf{x} &= \mathbf{u}^T\mathbf{u} \\ &= |\mathbf{u}| \\ &= |A\mathbf{x}| > 0\end{aligned}$$

which is only zero when $A\mathbf{x} = 0$. This has implications for the *dependance* of the columns of A - this is discussed in detail below. If we have independance in A, we are guaranteed $S = A^T A$ is symmetric positive definite.

Matrix Inverse and Non-Singular Matrices

Recall the matrix-vector product $A\mathbf{x} = \mathbf{y}$ where A is square and size n. Is it possible to work out the particular $\mathbf{x}$ which results in a particular $\mathbf{y}$? Clearly $\mathbf{y}$ should be in column space of A, ie $\mathbf{y} \in \mathsf{c}(A)$. Assume there exists B where $BA = I$, so that

$$\begin{aligned}A\mathbf{x} &= \mathbf{y} \\ BA\mathbf{x} &= B\mathbf{y} \\ \mathbf{x} &= B\mathbf{y}.\end{aligned}$$

This matrix B is called the *inverse* matrix of A and is denoted A^{-1}. This matrix has many interesting properties which will be discussed below.

Definitions

An $(n \times n)$ matrix A is *non-singular* if there exists another matrix B where $AB = BA = I$.
If such a matrix exists is it unique, and we denote it $B = A^{-1}$. Note that the symmetry is assumed as part of the definition.

To see that the inverse is unique, assume instead that $A^{-1} = B_1$ and $A^{-1} = B_2$ where $B_1 \neq B_2$ - that is, there are two different matrices which are inverses of A. By definition

$$\begin{aligned}AB_1 &= I = B_1 A \\ AB_2 &= I = B_2 A\end{aligned}$$

and by multiplying the second equation by B_1 we see

$$\begin{aligned}(B_1 A)\, B_2 &= B_1 I \\ B_2 &= B_1\end{aligned}$$

where the bracketed term is I and we have $B_2 = B_1$. We have a contradiction so have proved that inverses are unique.

There are many properties that guarantee the existence or non-existance of the inverse. The most important three, which involve the previous definitions of *rank*, *range* and *null space* are given here.

When determing if A^{-1} exists, remember

$$\begin{aligned} A^{-1}\,\text{exists} &\iff \text{null}\,(A) = \{\varnothing\} \\ A^{-1}\,\text{exists} &\iff \text{range}\,(A) = \mathcal{R}^n \\ A^{-1}\,\text{exists} &\iff \text{rank}\,(A) = n \\ A^{-1}\,\text{exists} &\iff \texttt{det}\,(A) \neq 0 \end{aligned}$$

When finding if A is singular analytically, I ususally check if

$$\text{null}\,(A) = \{\varnothing\} \tag{17}$$

ie can I find $\mathbf{x}$ where $A\mathbf{x} = \mathbf{0}$. Some important identities that are used regularly are

$$\begin{aligned} (AB)^{-1} &= B^{-1}A^{-1} \\ \left(A^T\right)^{-1} &= \left(A^{-1}\right)^T \end{aligned}$$

To check the first, let $M = AB$, and note that

$$B^{-1}A^{-1}M = B^{-1}\left(A^{-1}A\right)B = I$$

so that $M^{-1} = B^{-1}A^{-1} = (AB)^{-1}$. For the second, consider $\left(A^{-1}\right)^T A^T = \left(AA^{-1}\right)^T = I$

I haven't spoken about determinants in great length because I find they are not particularly useful analytically or numerically except in special cases. But in these special cases, they are enormously useful. If A is *diagonal* or *triangular*, its determinant is particular simple

$$\texttt{det}\,(A) = \prod A_{ii} \tag{18}$$

A non-zero determinant guarantees a non-singular matrix, and vica versa. This above formula shows that diagonal matrices are non-singular if all diagonals are non-zero, and triangular matrices are never singular (since triangular matrices have no zeros on their diagonals). Also, if A^{-1} exists, then

$$\begin{aligned} A\text{ is diagonal} &\iff A^{-1}\text{ is diagonal} \\ A\text{ is upper triangular} &\iff A^{-1}\text{ is upper triangular} \\ A\text{ is lower triangular} &\iff A^{-1}\text{ is lower triangular} \end{aligned}$$

The inverse of a diagonal matrix D has

$$\left(D^{-1}\right)_{ii} = 1/D_{ii} \tag{19}$$

which is what we might expect when viewing D as a scaling operation (ie to make the inverse, just use the opposite scaling factor!).

Analytic Matrix Inverse

Here, add a discussion of elementary matrix operations, and finding inverses.

Numeric Matrix Inverse

Include a section on why we dont actually computin matrix inverses, conditioning etc.

Most numeric methods for solving systems of equations

$$A\mathbf{x} = \mathbf{y} \tag{20}$$

involve decomposing A into a product of two matrices (ie perform a *matrix decomposition*), one of which is triangular (or has some other simple form which makes solving fast and easy). The solution then usually proceeds in two steps. The most basic is the LU decomposition, where we find lower and upper triangular matrices L, U where $A = LU$. The above equation then reduces to

$$\begin{aligned} A\mathbf{x} &= \mathbf{y} \\ L\left(U\mathbf{x}\right) &= \mathbf{y} \\ L\mathbf{v} &= \mathbf{y} \end{aligned}$$

where $U\mathbf{x} = \mathbf{v}$. Here we solve for $\mathbf{v}$, which is very fast since L is triangular (as we will see below). Once we have $\mathbf{v}$, we use it to solve for $\mathbf{x}$ in $U\mathbf{x} = \mathbf{v}$. Again, since U is triangular this is very fast.

Solving Triangular Matrices

Triangular matrices can be solved very quickly using backward/forward substitution. Consider solving $U\mathbf{x} = \mathbf{y}$. The bottom row only has one one non-zero element $U_{nn}x_n = y_n$, so

$$x_n = y_n / U_{nn} \tag{21}$$

. The next row has two:

$$\begin{aligned} U_{n-1,n-1}x_{n-1} + U_{n,n}x_n &= y_{n-1} \\ x_{n-1} &= \left(y_{n-1} - U_{n,n}x_n\right) / U_{n-1,n-1} \end{aligned}$$

and we know x_n from the previous step. We continue on recursively. An upper triangular matrix is solved the identical way except starting at the top row instead of the bottom. Clearly the process breaks down if T has a zero element on the diagonal, but we expected that since this would cause $\texttt{det}\left(T\right) = 0$. A triangular matrix of size N will take $\mathcal{O}\left(N^2\right)$ operations to solve in this way. Consider a particular row in the algorithm described above. The operations are

$$\begin{aligned} y_i &= \sum_{j=1}^{N} U_{ij}x_j \\ &= U_{ii}x_i + \sum_{j=i+1}^{N} U_{ij}x_j \\ x_i &= \left(y_i - \sum_{j=i+1}^{N} U_{ij}x_j\right) / U_{ii} \end{aligned}$$

where the second line follows since the first elements U_{i1} to $U_{i,i-1}$ are all zero, and in last last recognise that we have already found the next elements we are solving for, so that for x_i we already have found $x_{i+1}, x_{i+2}, \ldots, x_N$. Computation of x_i involves the summation of $U_{ij}x_j$ for $j = i+1, \ldots, N$, which is $\mathcal{O}\left(N\right)$. This calculation is performed N times, once for each of the N rows, so in total we require $N\mathcal{O}\left(N\right) = \mathcal{O}\left(N^2\right)$ operations.

Creating Triangular Matrices with LU decompositions

The LU decomposition of a matrix A are the upper and lower triangular matrices U and L where $A = LU$. Conventionally, the main diagonal of L is $L_{ii} = 1$. This ensures uniqueness, since A has

n^2 elements, while L has

$$\begin{aligned}\text{Elements} &= n + (n-1) + \ldots + 2 + 1 \\ &= \sum_{i=1}^{n} i \\ &= n(n+1)/2\end{aligned}$$

non-zero elements, as does U. So we have $n(n+1)$ non-zero variables in total to set using the n^2 elements of A. To make this match set n of them, the diagonal of U, to be all one. When computing L and U analytically, we use sequences of transformations via elementary matrices. In practice LU are computed numerically in $\mathcal{O}\left(()\, N^3\right)$ operations. Here, I should add both numeric and analytic examples of LU decompositions. Note that many authors refer to decompositions where rows have been swapped (ie *pivoting*) as a *LU decomposition with pivoting*. If a LU decomposition is attempted directly without any pivoting the algorithm will fail anywhere with a zero element on the main diagonal. For example the algorithm will fail to decompose

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{22}$$

where it is obvious that exchanging the rows leads to an easy solution. If the pivot operations on A are collected in the matrix P, the the decomposition is $PA = LU$. Here P is *not*-unique, but once P is specified LU is.

The benefit of the LU decomposition $PA = LU$ is that we can use forward and backward substitution to very quickly solve linear systems. Consider the system $A\mathbf{x} = \mathbf{y}$. To solve, find P which decomposes $PA = LU$, and solve $PA\mathbf{x} = LU\mathbf{x} = P\mathbf{y}$ in two steps. The trick is to let $U\mathbf{x} = \mathbf{v}$, and solve in two steps as follows

$$\begin{aligned} A\mathbf{x} &= \mathbf{y} \\ PA\mathbf{x} &= P\mathbf{y} \\ L\left(U\mathbf{x}\right) &= P\mathbf{y} \\ L\mathbf{v} &= P\mathbf{y} \end{aligned}$$

for $\mathbf{v}$, and then solve

$$U\mathbf{x} = \mathbf{v} \tag{23}$$

for $\mathbf{x}$, as required.