Chapter 28

Michelle Bodnar, Andrew Lohr

May 5, 2017

Exercise 28.1-1

We get the solution:

$$\begin{pmatrix} 3 \\ 14 - 4 \cdot 3 \\ -7 - 5 \cdot (14 - 4 \cdot 3) + 6 \cdot 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

Exercise 28.1-2

An LU decomposition of the matrix is given by

$$\begin{pmatrix} 4 & -5 & 6 \\ 8 & -6 & 7 \\ 12 & -7 & 12 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix} \begin{pmatrix} 4 & -5 & 6 \\ 0 & 4 & -5 \\ 0 & 0 & 4 \end{pmatrix}.$$

Exercise 28.1-3

First, we find the LUP decomposition of the given matrix

$$\begin{pmatrix} 1 & 5 & 4 \\ 2 & 0 & 3 \\ 5 & 8 & 2 \end{pmatrix}$$

we bring the 5 to the top, and then divide the first column by 5, and use Schur complements to change the rest of the matrix to get

$$\begin{pmatrix} 5 & 8 & 2 \\ .4 & -3.2 & 2.2 \\ .2 & 3.4 & 3.6 \end{pmatrix}$$

Then, we swap the third and second rows, and apply the Schur complement to get

$$\begin{pmatrix} 5 & 8 & 2 \\ .2 & 3.4 & 3.6 \\ .4 & -\frac{3.2}{3.4} & 2.2 + \frac{11.52}{3.4} \end{pmatrix}$$

This gets us the LUP decomposition that

$$L = \begin{pmatrix} 1 & 0 & 0 \\ .2 & 1 & 0 \\ .4 & -\frac{3.2}{3.4} & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} 5 & 8 & 2 \\ 0 & 3.4 & 3.6 \\ 0 & 0 & 2.2 + \frac{11.52}{3.4} \end{pmatrix}$$

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

Using this, we can get that the solution must be

$$\begin{pmatrix} 1 & 0 & 0 \\ .2 & 1 & 0 \\ .4 & -\frac{3.2}{3.4} & 1 \end{pmatrix} \begin{pmatrix} 5 & 8 & 2 \\ 0 & 3.4 & 3.6 \\ 0 & 0 & 2.2 + \frac{11.52}{3.4} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 12 \\ 9 \end{pmatrix}$$

Which, by forward substitution,

$$\begin{pmatrix} 5 & 8 & 2 \\ 0 & 3.4 & 3.6 \\ 0 & 0 & 2.2 + \frac{11.52}{3.4} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 11 \\ 7 + \frac{35.2}{3.4} \end{pmatrix}$$

So, finally by back substitution,

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -\frac{3}{19} \\ -\frac{1}{19} \\ \frac{49}{19} \end{pmatrix}$$

Exercise 28.1-4

The LUP decomposition of a diagonal matrix D is $D = IDI$ where I is the identity matrix.

Exercise 28.1-5

A LU decomposition of a permutation matrix is letting P be the inverse permutation matrix, and let both L and U be the identity matrix. Now, we need show that this representation is unique. We know that the permutation matrix A is non-singular. This means that U has nonzero elements all along its diagonal. Now, suppose that there were some nonzero element off of the diagonal in L, which is to say $L_{i,j} \neq 0$ for $i \neq j$. Then, look at row i in the product LU. This has a nonzero entry both at column j and at column i. Since it has more than one non-zero entry, it cannot be transformed into a permutation matrix by permuting the rows. Similarly, we have that U cannot have any off-diagonal elements. Lastly, since we know that both L and U are diagonal matrices, we

know that L is the identity. Since A only has ones as its nonzero entries, and $LU = U$. U must also only have ones as its nonzero entries. So, we have U is the identity. This means that $PA = I$, which means that $P = A^{-1}$. This completes showing that the given decomposition is unique.

Exercise 28.1-6

The zero matrix always has an LU decomposition by taking L to be any unit lower-triangular matrix and U to be the zero matrix, which is upper triangular.

Exercise 28.1-7

For LU decomposition, it is indeed necessary. If we didn't run the last run of the outermost for loop, u_{nn} would be left its initial value of zero instead of being set equal to a_{nn}. This can clearly produce incorrect results, because the LU decomposition of any non-singular matrix must have both L and U having positive determinant. However, if $u_{nn} = 0$, the determinant of U will be zero by problem D.2-2.

For LUP-decomposition, the iteration of the outermost for loop that occurs with $k = n$ will not change the final answer. Since π would have to be a permutation on a single element, it cannot be non-trivial. and the for loop on line 16 will not run at all.

Exercise 28.2-1

Showing that being able to multiply matrices in time $M(n)$ implies being able to square matrices in time $M(n)$ is trivial because squaring a matrix is just multiplying it by itself.

The more tricky direction is showing that being able to square matrices in time $S(n)$ implies being able to multiply matrices in time $O(S(n))$.

As we do this, we apply the same regularity condition that $S(2n) \in O(S(n))$. Suppose that we are trying to multiply the matrices, A and B, that is, find AB. Then, define the matrix

$$C = \begin{pmatrix} I & A \\ 0 & B \end{pmatrix}$$

Then, we can find C^2 in time $S(2n) \in O(S(n))$. Since

$$C^2 = \begin{pmatrix} I & A + AB \\ 0 & B \end{pmatrix}$$

Then we can just take the upper right quarter of C^2 and subtract A from it to obtain the desired result. Apart from the squaring, we've only done work that is $O(n^2)$. Since $S(n)$ is $\Omega(n^2)$ anyways, we have that the total amount of work we've done is $O(n^2)$.

Exercise 28.2-2

In this problem and the next, we'll follow the approach taken in *Algebraic Complexity Theory* by Burgisser, Claussen, and Shokrollahi. Let A be an $n \times n$ matrix. Without loss of generality we'll assume $n = 2^k$, and impose the regularity condition that $L(n/2) \leq cL(n)$ where $c < 1/2$ and $L(n)$ is the time it takes to find an LUP decomposition of an $n \times n$ matrix. First, decompose A as

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$$

where A_1 is $n/2$ by n. Let $A_1 = L_1 U_1 P_1$ be an LUP decomposition of A_1, where L_1 is $n/2$ by $n/2$, U_1 is $n/2$ by n, and P_1 is n by n. Perform a block decomposition of U_1 and $A_2 P_1^{-1}$ as $U_1 = [\overline{U}_1 | B]$ and $A_2 P_1^{-1} = [C|D]$ where $\overline{U}_1$ and C are $n/2$ by $n/2$ matrices. Since we assume that A is nonsingular, $\overline{U}_1$ must also be nonsingular. Set $F = D - C\overline{U}_1^{-1} B$. Then we have

$$A = \begin{bmatrix} L_1 & 0 \\ C\overline{U}_1^{-1} & I_{n/2} \end{bmatrix} \begin{bmatrix} \overline{U}_1 & B \\ 0 & F \end{bmatrix} P_1.$$

Now let $F = L_2 U_2 P_2$ be an LUP decomposition of F, and let $\overline{P} = \begin{bmatrix} I_{n/2} & 0 \\ 0 & P_2 \end{bmatrix}$. Then we may write

$$A = \begin{bmatrix} L_1 & 0 \\ C\overline{U}_1^{-1} & L_2 \end{bmatrix} \begin{bmatrix} \overline{U}_1 & BP_2^{-1} \\ 0 & U_2 \end{bmatrix} \overline{P} P_1.$$

This is an LUP decomposition of A. To achieve it, we computed two LUP decompositions of half size, a constant number of matrix multiplications, and a constant number of matrix inversions. Since matrix inversion and multiplication are computationally equivalent, we conclude that the runtime is $O(M(n))$.

Exercise 28.2-3

From problem 28.2-2, we can find a LU-decomposition algorithm that only takes time $O(M(n))$. So, we run that algorithm and multiply together all of the entries along the diagonal of U, this will be the determinant of the original matrix.

We have no short answer for the second direction of this problem. Typically we strive to complete all the exercises independently as a way of gaining a greater understanding ourselves. For the second half of this problem, this was unfortunately not something that we achieved. After being stuck for several weeks, I asked a number of other graduate students, and even two professors, none were able to help with how the proof went. One did however refer me to the book Algebraic Complexity Theory by Burgisser, Claussen, and Shokrollahi. They have the proof in their section 16.4. This however is not at all self contained and so would be very long to try and include here as a solution. This was the

hardest exercise in the book. If you know a good proof of this fact, I would love to hear it, please let me know by emailing ajl213 "at" math.rutger.edu.

Exercise 28.2-4

Suppose we can multiply boolean matrices in $M(n)$ time, where we assume this means that if we're multiplying boolean matrices A and B, then $(AB)_{ij} = (a_{i1} \wedge b_{1j}) \vee \ldots \vee (a_{in} \wedge b_{nj})$. To find the transitive closure of a boolean matrix A we just need to find the n^{th} power of A. We can do this by computing A^2, then $(A^2)^2$, then $((A^2)^2)^2$), and so on. This requires only $\lg n$ multiplications, so the transitive closure can be computed in $O(M(n) \lg n)$.

For the other direction, first view A and B as adjacency matrices, and impose the regularity condition $T(3n) = O(T(n))$, where $T(n)$ is the time to compute the transitive closure of a graph on n vertices. We will define a new graph whose transitive closure matrix contains the boolean product of A and B. Start by placing $3n$ vertices down, labeling them $1, 2, \ldots, n, 1', 2', \ldots, n', 1'', 2'', \ldots, n''$. Connect vertex i to vertex j' if and only if $A_{ij} = 1$. Connect vertex j' to vertex k'' if and only if $B_{jk} = 1$. In the resulting graph, the only way to get from the first set of n vertices to the third set is to first take an edge which "looks like" an edge in A, then take an edge which "looks like " an edge in B. In particular, the transitive closure of this graph is:

$$\begin{bmatrix} I & A & AB \\ 0 & I & B \\ 0 & 0 & I \end{bmatrix}.$$

Since the graph is only of size $3n$, computing its transitive closure can be done in $O(T(3n)) = O(T(n))$ by the regularity condition. Therefore multiplying matrices and finding transitive closure are are equally hard.

Exercise 28.2-5

It does not work necessarily over the field of two elements. The problem comes in in applying theorem D.6 to conclude that $A^T A$ is positive definite. In the proof of that theorem they obtain that $||Ax||^2 \geq 0$ and only zero if every entry of Ax is zero. This second part is not true over the field with two elements, all that would be required is that there is an even number of ones in Ax. This means that we can only say that $A^T A$ is positive semi-definite instead of the positive definiteness that the algorithm requires.

Exercise 28.2-6

We may again assume that our matrix is a power of 2, this time with complex entries. For the moment we assume our matrix A is Hermitian and positive-definite. The proof goes through exactly as before, with matrix transposes replaced by conjugate transposes, and using the fact that Hermitian positive-definite matrices are invertible. Finally, we need to justify that we can obtain the

same asymptotic running time for matrix multiplication as for matrix inversion when A is invertible, but not Hermitian positive-definite. For any nonsingular matrix A, the matrix A^*A is Hermitian and positive definite, since for any x we have $x^*A^*Ax = \langle Ax, Ax\rangle > 0$ by the definition of inner product. To invert A, we first compute $(A^*A)^{-1} = A^{-1}(A^*)^{-1}$. Then we need only multiply this result on the right by A^*. Each of these steps takes $O(M(n))$ time, so we can invert any nonsingluar matrix with complex entries in $O(M(n))$ time.

Exercise 28.3-1

To see this, let e_i be the vector that is zeroes except for a one in the ith position. Then, we consider the quantity $e_i^T A e_i$ for every i. Ae_i takes each row of A and pulls out the ith column of it, and puts those values into a column vector. Then, multiplying that on the left by e_i^T, pulls out the ith row of this quantity, which means that the quantity $e_i^T A e_i$ is exactly the value of $A_{i,i}$. So, we have that by positive definiteness, since e_i is nonzero, that quantity must be positive. Since we do this for every i, we have that every entry along the diagonal must be positive.

Exercise 28.3-2

Let $x = -by/a$. Since A is positive-definite we must have

$$\begin{aligned} 0 &< [xy]^T A \begin{bmatrix} x \\ y \end{bmatrix} \\ &= [xy]^T \begin{bmatrix} ax + by \\ bx + cy \end{bmatrix} \\ &= ax^2 + 2bxy + cy^2 \\ &= cy^2 - \frac{b^2y^2}{a} \\ &= (c - b^2/a)y^2. \end{aligned}$$

Thus, $c - b^2/a > 0$ which implies $ca - b^2 > 0$, since $a > 0$.

Exercise 28.3-3

Suppose to a contradiction that there were some element a_{ij} with $i \neq j$ so that a_{ij} were a largest element. We will use e_i to denote the vector that is all zeroes except for having a 1 at position i. Then, we consider the value $(e_i - e_j)^T A(e_i - e_j)$. When we compute $A(e_i - e_j)$ this will return a vector which is column i minus column j. Then, when we do the last multiplication, we will get the quantity which is the ith row minus the jth row. So,

$$\begin{aligned} (e_i - e_j)^T A(e_i - e_j) &= a_{ii} - a_{ij} - a_{ji} + a_{jj} \\ &= a_{ii} + a_{jj} - 2a_{ij} \leq 0 \end{aligned}$$

Where we used symmetry to get that $a_{ij} = a_{ji}$. This result contradicts the fact that A was positive definite. So, our assumption that there was a element tied for largest off the diagonal must of been false.

Exercise 28.3-4

The claim clearly holds for matrices of size 1 because the single entry in the matrix is positive the only leading submatrix is the matrix itself. Now suppose the claim holds for matrices of size n, and let A be an $(n+1)\times(n+1)$ symmetric positive-definite matrix. We can write A as

$$A = \left[\begin{array}{c|c} A' & w \\ \hline v & c \end{array}\right].$$

Then A' is clearly symmetric, and for any x we have $x^T A' x = [x0]A \begin{bmatrix} x \\ 0 \end{bmatrix} > 0$, so A' is positive-definite. By our induction hypothesis, every leading submatrix of A' has positive determinant, so we are left only to show that A has positive determinant. By Theorem D.4, the determinant of A is equal to the determinant of the matrix

$$B = \left[\begin{array}{c|c} c & v \\ \hline w & A' \end{array}\right].$$

Theorem D.4 also tells us that the determinant is unchanged if we add a multiple of one column of a matrix to another. Since $0 < e_{n+1}^T A e_{n+1} = c$, we can use multiples of the first column to zero out every entry in the first row other than c. Specifically, the determinant of B is the same as the determinant of the matrix obtained in this way, which looks like

$$C = \left[\begin{array}{c|c} c & 0 \\ \hline w & A'' \end{array}\right].$$

By definition, $\det(A) = c\det(A'')$. By our induction hypothesis, $\det(A'') > 0$. Since $c > 0$ as well, we conclude that $\det(A) > 0$, which completes the proof.

Exercise 28.3-5

When we do an LU decomposition of a positive definite symmetric matrix, we never need to permute the rows. This means that the pivot value being used from the first operation is the entry in the upper left corner. This gets us that for the case $k = 1$, it holds because we were told to define $det(A_0) = 1$, getting us, $a_{11} = det(A_1)/det(A_0)$. When Diagonalizing a matrix, the product of the

pivot values used gives the determinant of the matrix. So, we have that the determinant of A_k is a product of the kth pivot value with all the previous values. By induction, the product of all the previous values is $det(A_{k-1})$. So, we have that if x is the kth pivot value, $det(A_k) = xdet(A_{k-1})$, giving us the desired result that the k th pivot value is $det(A_k)/det(A_{k-1})$.

Exercise 28.3-6

First we form the A matrix

$$A = \begin{bmatrix} 1 & 0 & e \\ 1 & 2 & e^2 \\ 1 & 3\lg 3 & e^3 \\ 1 & 8 & e^4 \end{bmatrix}.$$

We'll compute the pseudoinverse using Wolfram Alpha, then multiply it by y, to obtain the coefficient vector

$$c = \begin{bmatrix} .411741 \\ -.20487 \\ .16546 \end{bmatrix}.$$

Exercise 28.3-7

$$\begin{aligned} AA^+A &= A((A^TA)^{-1}A^T)A \\ &= A(A^TA)^{-1}(A^TA) \\ &= A \end{aligned}$$

$$\begin{aligned} A^+AA^+ &= ((A^TA)^{-1}A^T)A((A^TA)^{-1}A^T) \\ &= (A^TA)^{-1}(A^TA)(A^TA)^{-1}A^T \\ &= (A^TA)^{-1}A^T \\ &= A^+ \end{aligned}$$

$$\begin{aligned} (AA^+)^T &= (A(A^TA)^{-1}A^T)^T \\ &= A((A^TA)^{-1})^TA^T \\ &= A((A^TA)^T)^{-1}A^T \\ &= A(A^TA)^{-1}A^T \\ &= AA^+ \end{aligned}$$

$$
\begin{aligned}
(AA^+)^T &= ((A^TA)^{-1}A^TA)^T \\
&= ((A^TA)^{-1}(A^TA))^T \\
&= I^T \\
&= I \\
&= (A^TA)^{-1}(A^TA) \\
&= A^+A
\end{aligned}
$$

Problem 28-1

a. By applying the procedure of the chapter, we obtain that

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

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

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

b. We first do back substitution to obtain that

$$
Ux = \begin{pmatrix}
5 \\ 4 \\ 3 \\ 2 \\ 1
\end{pmatrix}
$$

So, by forward substitution, we have that

$$
x = \begin{pmatrix}
5 \\ 9 \\ 12 \\ 14 \\ 15
\end{pmatrix}
$$

c. We will set $Ax = e_i$ for each i, where e_i is the vector that is all zeroes except for a one in the ith position. Then, we will just concatenate all of these solutions together to get the desired inverse.

equation	solution
$Ax_1 = e_1$	$x_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$
$Ax_2 = e_2$	$x_2 = \begin{pmatrix} 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$
$Ax_3 = e_3$	$x_3 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 3 \\ 3 \end{pmatrix}$
$Ax_4 = e_4$	$x_4 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 4 \end{pmatrix}$
$Ax_5 = e_5$	$x_5 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$

This gets us the solution that

$$A^{-1} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 3 & 3 & 3 \\ 1 & 2 & 3 & 4 & 4 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

d. When performing the LU decomposition, we only need to take the max over at most two different rows, so the loop on line 7 of LUP-DECOMPOSITION drops to $O(1)$. There are only some constant number of nonzero entries in each row, so the loop on line 14 can also be reduced to being $O(1)$. Lastly, there are only some constant number of nonzero entries of the form a_{ik} and a_{kj}. since the square of a constant is also a constant, this means that the nested for loops on lines 16-19 also only take time O(1) to run. Since the

for loops on lines 3 and 5 both run $O(n)$ times and take $O(1)$ time each to run(provided we are smart to not consider a bunch of zero entries in the matrix), the total runtime can be brought down to $O(n)$.

Since for a Tridiagonal matrix, it will only ever have finitely many nonzero entries in any row, we can do both the forward and back substitution each in time only $O(n)$.

Since the asymptotics of performing the LU decomposition on a positive definite tridiagonal matrix is $O(n)$, and this decomposition can be used to solve the equation in time $O(n)$, the total time for solving it with this method is $O(n)$. However, to simply record the inverse of the tridiagonal matrix would take time $O(n^2)$ since there are that many entries, so, any method based on computing the inverse of the matrix would take time $\Omega(n^2)$ which is clearly slower than the previous method.

e. The runtime of our LUP decomposition algorithm drops to being $O(n)$ because we know there are only ever a constant number of nonzero entries in each row and column, as before. Once we have an LUP decomposition, we also know that that decomposition have both L and U having only a constant number of non-zero entries in each row and column. This means that when we perform the forward and backward substitution, we only spend a constant amount of time per entry in x, and so, only takes $O(n)$ time.

Problem 28-2

a. We have $a_i = f_i(0) = y_i$ and $b_i = f_i'(0) = f'(x_i) = D_i$. Since $f_i(1) = a_i + b_i + c_i + d_i$ and $f_i'(1) = b_i + 2c_i + 3d_i$ we have $d_i = D_{i+1} - 2y_{i+1} + 2y_i + D_i$ which implies $c_i = 3y_{i+1} - 3y_i - D_{i+1} - 2D_i$. Since each coefficient can be computed in constant time from the known values, we can compute the $4n$ coefficients in linear time.

b. By the continuity constraints we have $f_i''(1) = f_{i+1}''(0)$ which implies that $2c_i + 6d_i = 2c_{i+1}$, or $c_i + 3d_i = c_{i+1}$. Using our equations from above, this is equivalent to

$$D_i + 2D_{i+1} + 3y_i - 3y_{i+1} = 3y_{i+2} - 3y_{i+1} - D_{i+2} - 2D_{i+1}.$$

Rearranging gives the desired equation

$$D_i + 4D_{i+1} + D_{i+2} = 3(y_{i+2} - y_i).$$

c. The condition on the left endpoint tells us that $f_0''(0) = 0$, which implies $2c_0 = 0$. By part a, this means $3(y_1 - y_0) = 2D_0 + D_1$. The condition on the right endpoint tells us that $f_{n-1}''(1) = 0$, which implies $c_{n-1} + 3d_{n-1} = 0$. By part a, this means $3(y_n - y_{n-1}) = D_{n-1} + 2D_n$.

d. The matrix equation has the form $AD = Y$, where A is symmetric and tridiagonal. It looks like this:

$$\begin{bmatrix} 2 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \cdots & \vdots \\ \vdots & \cdots & 1 & 4 & 1 & 0 \\ 0 & \cdots & 0 & 1 & 4 & 1 \\ 0 & \cdots & 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ \vdots \\ D_{n-1} \\ D_n \end{bmatrix} = \begin{bmatrix} 3(y_1 - y_0) \\ 3(y_2 - y_0) \\ 3(y_3 - y_1) \\ \vdots \\ 3(y_n - y_{n-2}) \\ 3(y_n - y_{n-1}) \end{bmatrix}.$$

e. Since the matrix is symmetric and tridiagonal, Problem 28-1 e tells us that we can solve the equation in $O(n)$ time by performing an LUP decomposition. By part a of this problem, once we know each D_i we can compute each f_i in $O(n)$ time.

f. For the general case of solving the nonuniform natural cubic spline problem, we require that $f(x_{i+1}) = f_i(x_{i+1} - x_i) = y_{i+1}$, $f'(x_{i+1}) = f'_i(x_{i+1} - x_i) = f'_{i+1}(0)$ and $f''(x_{i+1}) = f''_i(x_{i+1} - x_i) = f''_{i+1}(0)$. We can still solve for each of a_i, b_i, c_i, and d_i in terms of y_i, y_{i+1}, D_i, and D_{i+1}, so we still get a tridiagonal matrix equation. The solution will be slightly messier, but ultimately it is solved just like the simpler case, in $O(n)$ time.