Generating parametrized linear systems for teaching linear algebra

Aaron E. Naiman

INTERNATIONAL JOURNAL OF MATHEMATICAL EDUCATION IN SCIENCE AND TECHNOLOGY
2024, VOL. 55, NO. 1, 164–170
https://doi.org/10.1080/0020739X.2022.2114107

Taylor & Francis
Taylor & Francis Group

CLASSROOM NOTE

Generating parametrized linear systems for teaching linear algebra

Aaron E. Naiman

Department of Applied Mathematics, Jerusalem College of Technology-Machon Lev, Jerusalem, Israel

ABSTRACT
We automate and randomize the building of linear systems with a parameter, appropriate for assigning to students. When the parameter takes on a specific value, the system has no solutions. When the parameter takes on a different value, the system has an infinite number of solutions.

ARTICLE HISTORY
Received 24 September 2021

KEYWORDS
Generating linear systems; input parameter; education; test questions

MSC2010
15A06; 97H60

1. Introduction

Khattak and Jeffrey (2017) introduce the idea of building special, square, integer matrices, for the purpose of subsequent orthogonalization by the Gram–Schmidt process. (In a different paper Camargos Couto & Jeffrey, 2018, the second author presents a similar analysis for the Householder transformation.) Their particular application is a pedagogic one, with the intent of making life easier for students solving such problems, and for teachers grading the same. While it can be argued that perhaps more realistic, not necessarily integer coefficients are a better pedagogic tool, their approach allows the students to concentrate on the problem at hand (e.g. the Gram–Schmidt process), rather than getting bogged down with complicated algebra.

Similarly, in linear algebra courses, a teacher often needs to build a linear system for demonstration in class, or compiling questions for homework or examinations. Many times the system may include input parameters as well. A typical question might be:

> For what values of the input parameter k, does the following linear system, $A\mathbf{x} = \mathbf{b}$, have: no solutions, a unique solution (and what is it), or an infinite number of solutions (and what are they)?

In this context, A is the $m \times n$ coefficient matrix, $\mathbf{x}$ is the n-length solution vector, and $\mathbf{b}$ is the m-length vector of right-hand side values.

In the classroom setting, we furthermore want the system to have relatively small coefficients and right-hand side values. As this question is often posed in the context of a square system, we develop here methods for building such a linear system. However, the final method chosen in the end, generalizes to non-square systems.

CONTACT Aaron E. Naiman naiman@jct.ac.il

Sections 2–5 detail our process of solving this problem and explain challenges discovered along the way. The final section includes an algorithm for constructing a linear system with parameter k, such that one value of this parameter leads to no solutions, and another value leads to an infinite number of solutions. This is all performed while keeping the numbers in the linear system as integers and small.

2. Nonzeros off the diagonal

One way to start this process is to commence with the identity matrix, I_n for the coefficients. In order to generate the cases of no solutions and an infinite number of solutions, we can proceed by replacing two zeros with the parameter k–or better yet, with $k + k_1$ and $k + k_2$, where k_1 and k_2 are any randomized, small (in the absolute value sense) integer values, e.g. $k-3$ and $k+5$.

Ostensibly we would then follow up with choosing random right-hand side values, augmenting these to the right-hand side of the coefficient matrix, and then transforming the augmented matrix with two of the three elementary row operations, i.e. switching rows or adding to a row a multiple of another row. We would refrain from the third elementary row operation of multiplying a row by a nonzero value, in order that the coefficients and right hand side values remain relatively small, as well as integer values.

The problem with this approach is that it will not lead to our desired goal of one parameter value leading to no solutions, and another to an infinite number of solutions. If both placements of the parameter values are above or below the diagonal, then the matrix remains triangular (upper or lower, respectively). Therefore, the matrix's determinant is nonzero and there would be only unique solutions for the system.

If one placement is above the diagonal, and one below, in a non-symmetric fashion, i.e. $(i_1, j_1) \neq (j_2, i_2)$, then again, two quick elementary row operations return us to the identity matrix, and we are left with only unique solutions to the system. E.g. for a small 4×4 system we might have

$$\begin{pmatrix} 1 & k-3 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ k+5 & 0 & 0 & 1 \end{pmatrix}$$

By subtracting $k-3$ times the second row from the first, we eliminate that term. Subsequently, subtracting $k+5$ times the first row from the fourth, we eliminate the final term, returning to the identity matrix, and therefore only a unique solution exists.

If the placements *are* symmetric, with $(i_1, j_1) = (j_2, i_2)$, then indeed we can obtain a zero along the diagonal but only in one position. E.g. for the following system

$$\begin{pmatrix} 1 & k-3 & 0 & 0 \\ k+5 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

by subtracting $k+5$ times the first row from the second, we arrive at

$$\begin{pmatrix} 1 & k-3 & 0 & 0 \\ 0 & 16-k(k+2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Therefore, depending on the corresponding right-hand side value, *both* parameter values give rise to *either* no solutions, or an infinite number of solutions but not both.

Clearly we need to be able to generate *two* zeros along the diagonal.

3. Zeros along the diagonal

As we saw, it is not enough to have one zero along the diagonal, but we would need two, in order that both parameter values give rise to *either* no solutions, or an infinite number of solutions. We can once again start with our identity matrix, and now substitute two diagonal values with $k+k_1$ and $k+k_2$. As before, we then follow up with randomized right-hand side values, augmenting them to the coeficient matrix, and transforming the augmented matrix with the two elementary row operations.

While this approach leads us to the desired goals, we found that the terms, e.g. $k-3$ and $k+5$ remain in the final system. E.g. starting with a coefficient matrix of

$$\begin{pmatrix} k-3 & 0 & 0 \\ 0 & k+5 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and a right-hand side vector of $(-4\ 1\ -1)^T$, after performing several elementary row operations we end up with

$$\begin{aligned} 3(k-3)x + (k+5)y + 6z &= -1 \\ (3-k)x - (k+5)y + 3z &= 0 \\ (6-2k)x + 3(k+5)y - 6z &= 1 \end{aligned}$$

making the parameter values obvious, with the only question remaining: which leads to no solutions, and which to an infinite number of solutions. This turns out to be trivial. We also saw that for small, 3×3 systems, nearly always at least one of the equations was left with only two variables in it, making the problem unnecessarily easier.

We needed to hide the relevant values of the parameter. This is what we address in the next section.

4. Hiding the parameter values

As before, in order to build the coefficient matrix A, we start with the identity matrix I_n. We want the matrix to remain (upper) triangular. It is irrelevant which rows serve our special cases, but we want to adjust positions above those diagonal elements, so we avoid the first row. For demonstrational purposes only, we choose the last two rows, replacing the final two elements of the diagonal with $k+k_1$ and $k+k_2$, respectively. In order to move the obvious values of the parameter away from the parameter itself (e.g. $k-3$ and $k+5$ from the example above), we perform the following steps:

(1) set to 1 (or any other small, non-zero integer) an element above the second to last diagonal element, say in the first row,
(2) set to 1 (or any other small, non-zero integer) an element above the last diagonal element, in a *different* row from above, say in the second row,

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 1 & 0 & \ddots & \ddots & 1 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & k+k_1 & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & k+k_2 \end{pmatrix} \tag{1}$$

(3) subtract k_1 times the first row, from the second to last row, and
(4) subtract k_2 times the second row, from the last row.

The result is

$$A(k) = \begin{pmatrix} 1 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 1 & 0 & \ddots & \ddots & 1 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & 1 & \ddots & \vdots \\ -k_1 & 0 & \cdots & 0 & k & 0 \\ 0 & -k_2 & \cdots & 0 & 0 & k \end{pmatrix} \tag{2}$$

Since the coefficient matrix (2) is no longer diagonal, nor even triangular, we cannot simply assign right-hand side values to obtain the linear system with the desired characteristics. Instead, we choose values of the solution vector **x**, and then multiply the coefficient matrix (2) by the solution vector, generating the right-hand side values **b**. We demonstrate with the following example, using values $k_1 = -3$ and $k_2 = 5$, as we did in the previous sections

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 3 & 0 & k & 0 \\ 0 & -5 & 0 & k \end{pmatrix} \tag{3}$$

For the first $n-2$ values of the solution vector, we let x_i be any random, small integer values. The last value, x_n is set to zero, allowing for the infinite number of solutions. The second to last value, x_{n-1} is set to a *nonzero* random value, so that the subsequent matrix-vector multiplication, $A\mathbf{x}$, leaves k in the second to last value of the right-hand side vector **b**. For our small example, the solution vector might be: $(7\ -6\ 4 0)^T$. The matrix-vector multiplication is now performed, generating a right-hand side vector of: $(11\ -6\ (4k+21) 30)^T$.

At this point, we need to generate a contradition for the parameter value of $-k_1$. Without any change, each of the two parameter values of $-k_1$ and $-k_2$ would lead to an infinite number of solutions. We therefore set the second to last value on the right-hand side to a random, small integer, different from the value obtained from the matrix-vector multiplication ($A\mathbf{x}$) already performed. The value of $k = -k_1 = 3$ produces a value of 33, so any other number would suffice. As with the methods in the previous sections, we now perform the two elementary row operations mentioned there, transforming the augmented matrix.

While this method indeed leads to a system with one parameter value, $-k_1$, leading to no solutions, and $-k_2$ leading to an infinite number of solutions, the matrix-vector multiplication of $A\mathbf{x}$ can easily lead to quite (relatively) large right-hand side values in $\mathbf{b}$, as seen in the small example above. This is less than desirable, and can be prevented with the method in the following section.

5. Direct selection of right-hand sides

In order to avoid the matrix-vector multiplication ($A\mathbf{x}$) from the last section, we skip over choosing values of the solution vector, and directly address the right side values. However, we now take advantage of a little linear algebra knowledge, in order to make a smart selection.

We denote the coefficient matrix (2) for given values of mb$k$: $A_1 = A(k_1)$ and $A_2 = A(k_2)$. Since they are square matrices, leading to either no solutions or an infinite number of solutions, we therefore know that their columns do not span $\mathbb{R}^n$. So we want a vector which is in the column space of one–but not the other.

We can easily generate a basis for the column space of, say, A_2, by row reducing A_2^T and then transposing back. Then, we proceed from the rightmost vector of the basis (as it contains the most zeros) to the first, stopping when we have a vector which is not in the column space of A_1, and this we choose for the right-hand side.

While this *always* seems to work, who is to say that perhaps some coefficient matrices will lead to cases with *all* of the vectors in the basis of the column space of A_2, in fact being in the column space of A_1? This indeed cannot happen, and demonstrating it leads to an even simpler approach.

We recall that the coefficient matrices A_1 and A_2 are simply the matrix (2) with k set to the value of $-k_1$ and $-k_2$, respectively. The coefficient matrix (2), in turn, is row equivalent to (1), which is upper triangular, and with a zero in one of the last two diagonal positions.

Therefore, if the right-hand side is chosen to be the vector $(0\ 0\ \cdots\ 0\ 1\ 0)^T$ then it is in the column space of A_2. We then have k set to $-k_1$ leading to no solutions, and k set to $-k_2$ leads to an infinite number of solutions. If, on the other hand, the right-hand side is chosen to be the vector $(0\ 0\ \cdots\ 0\ 1)^T$ then we have $-k_2$ leading to no solutions, and $-k_2$ to an infinite number of solutions. Of course, one does not need a 1 in that position, as any non-zero is fine.

In terms of the small example (3) in Section 4, by the same logic it is row equivalent to

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & k-3 & 0 \\ 0 & 0 & 0 & k+5 \end{pmatrix}$$

Algorithm 1 Generate linear system with parameter

```
procedure GLSP(m, n, k_min, k_max, c_min, c_max, b_min, b_max)
    // Build an m × n linear system with parameter k which takes on
    // one value (in [k_min, k_max]) rendering the system with no solution,
    // and a different value (same range) producing an infinite number
    // of solutions. The initial right-hand side non-zero values are in
    // the range of [b_min, b_max]. Initial bounds for coefficients and solution
    // vector elements: [c_min, c_max].

    min_mn ← min (m, n)
    k_1, k_2 ← twoDistinctRandomIntegers ([k_min, k_max])
    coeff ← identity (m, n)
    d_1, d_2 ← twoDistinctRandomIntegersSorted ([2, min_mn])

    coeff [d_1, d_1] ← k + k_1
    coeff [d_2, d_2] ← k + k_2
    r_1 ← randomInteger ([1, d_1 − 1])
    coeff [r_1, d_1] ← randomNonZeroInteger([c_min, c_max])
    r_2 ← randomIntegerWithSkip ([1, d_2 − 1] , d_1)
    coeff [r_2, d_2] ← randomNonZeroInteger([c_min, c_max])
    coeff [d_1, :] ← coeff [d_1, :] − k_1 ∗ coeff [r_1, :]
    coeff [d_2, :] ← coeff [d_2, :] − k_2 ∗ coeff [r_2, :]

    b ← zeros (m)
    b [randomInteger ([0, 1]) ? d_1 : d_2] ←
  randomNonZeroInteger ([b_min, b_max])

    if m < n then
        coeff [:, m + 1 : n] ← randomInteger (m, n − m, [c_min, c_max])
        [b ← b + randomInteger () ∗ coeff [:, j] for j ← m + 1, n]
    end if

    augment(coeff, b)
    loop
        // swap rows or add (integer) multiple of one row to another
        //         and/or reorder the columns of coefficients
    end loop

    return coeff
end procedure
```

Therefore, if the right-hand side is chosen to be the vector $(0\ 0\ 1\ 0)^T$ then it is in the column space of our matrix with $k = -5$, as a linear combination of the first and third columns. We then have $k = 3$ leading to no solutions, and $k = -5$ leads to an infinite number of solutions. If, on the other hand, the right-hand side is chosen to be the vector $(0\ 0\ 0\ 1)^T$ then it is in the column space of our matrix with $k = 3$, as a linear combination of the second and fourth columns. We then have $k = -5$ leading to no solutions, and $k = 3$ leads to an infinite number of solutions.

We summarize this last approach, in Algorithm 1. It is written in a vector-based paradigm, similar to *Mathematica*, Matlab, Python's NumPy, etc. This was purposely provided in pseudocode and not, e.g. *Mathematica*, in order to make it easier for anyone to apply it, in their language of choice.

We note that this algorithm is easily generalized to non-square matrices, with the new elements neither adding new conflicts, nor possibilities for infinite solutions. For the underdetermined system (i.e. $m < n$), adding $n-m$ columns, random values are generated for them, as well as for the $n-m$ additional elements of the solution vector. Each of these columns is multiplied by the respective solution vector element, and added to the right hand side. For the overdetermined system (i.e. $m > n$), $m-n$ rows of zeros are added, and the right-hand side is extended by $m-n$ zeros as well.

With the right-hand side chosen, the transformation of the augmented matrix, with the two elementary row operations as before finishes the process, hiding rows of zeros as well, for overdetermined systems. We can also randomly reorder the columns of the coefficient matrix.

Disclosure statement

No potential conflict of interest was reported by the author.

ORCID

Aaron E. Naiman http://orcid.org/0000-0003-3594-3116

References

Camargos Couto, A. C., & Jeffrey, D. (2018, September 20–23). *Rational householder transformations* [Paper presentation]. 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Romania. (pp. 61–64).

Khattak, N., & Jeffrey, D. J. (2017, September 21–24). Rational orthonormal matrices. [Paper presentation]. 19th International Symposiumon Symbolic andNumeric Algorithms for Scientific Computing (SYNASC), Romania. (pp. 71–74).