Chapter 5

Simultaneous equations

5.1 overview

5.1.1 motivation

Systems of equations arise from boundary value problems and from problems involving coupled processes. In the former case, we wish to solve a differential equation for the value of a dependent variable at many values of an independent variable. The equations all have exactly the same form but different values for their coefficients. In the latter case, we are interested in solving a series equations that may have different forms but share dependent variables. In order to tackle these problems, we must develop methods for solving systems of "simultaneous" equations.

5.1.2 some definitions

A system of equations is a collection of equations involving the same set of dependent variables. In general, we desire a system with M equations describing M unknowns. If there are more unknowns than equations, there is not enough information to solve the system. Such problems are said to be **underdetermined**. If there are more (linearly independent) equations than unknowns then not all of the equations can be satisfied exactly and the problem is said to be **overdetermined**. It is possible to find an approximate solution to an under- or overdetermined system, in which all the equations are satisfied as well as possible. In the overdetermined case, the method of least squares is used. In the underdetermined case, the goal is to find a minimal norm solution. In this chapter, we will consider systems with the same number of equations and unknowns.

Boundary value problems produce a system of difference equations that approximate the derivatives of the function at nodes within a model domain. Here, we begin with a simpler example:

$$\begin{aligned} x_1 + x_2 &= 2 \\ 3x_1 + x_2 &= 4 \end{aligned} \tag{5.1}$$

for which it is easy to see that the solution is $x_1 = x_2 = 1$. The two lines represented by the equations intersect at a point (1,1). It is not always the case that only one set of values for the dependent variables satisfy the system of equations. Consider

$$\begin{aligned} x_1 + x_2 &= 2 \\ 2x_1 + 2x_2 &= 4 \end{aligned} \tag{5.2}$$

for which an infinite number of solutions can be found, or

$$\begin{aligned} 2x_1 + 2x_2 &= 2 \\ 2x_1 + 2x_2 &= 4 \end{aligned} \tag{5.3}$$

for which no solutions can be found. Systems of equations like (5.2) and (5.3) are called **singular**. A system of equations is **linearly independent** when none of the equations can be formed from algebraic combinations of other equations in the set. In the case of (5.2), no unique solution can be found because the two equations are not linearly independent. The second equation is twice the first and so not provide any new information. The problem is underdetermined: there are M unknowns but only $M - 1$ equations.

Linear systems of equations may be solved directly or by iteration. Iterative solutions to simultaneous equations make successive approximations of the unknowns. Usually, it is more efficient to solve large systems by iteration than to solve them directly. Efficient direct solution methods are possible for special types of systems, such as the tridiagonal systems that represent the two-point boundary value problems. MATLAB has many built-in functions for solving matrix equations.

5.1.3 systems of equations expressed using matrices and vectors

In a general linear system of M independent equations, each of the M equations may contain all of the M unknowns:

$$\begin{aligned} a_{11}y_1 + a_{12}y_2 + a_{13}y_3 &= b_1 \\ a_{21}y_1 + a_{22}y_2 + a_{23}y_3 &= b_2 \\ a_{31}y_1 + a_{32}y_2 + a_{33}y_3 &= b_3 \end{aligned} \tag{5.4}$$

The equations contain 9 coefficients a of the dependent variables y and 3 independent variables b. Together, the coefficients a_{ij} may be represented as a 3 by 3 matrix:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

The variables y_i and b_i may be represented as [3, 1] vectors:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Written as a matrix equation, (5.4) is:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{5.5}$$

or:

$$\boldsymbol{A}\ y\ =\ b \tag{5.6}$$

where $\boldsymbol{A}$ is an $[n\ ,\ m]$ matrix, y is an $[n,\ 1]$ vector, and b is an $[n,\ 1]$ vector. We are concerned now with cases where $n = m$ and in this example, $n = m = 3$.

5.2 linear algebra

Linear algebra concerns the manipulation of vectors and matrices in order to perform a variety of mathematical tasks. Here, we will review how vectors and matrices are used.

5.2.1 vectors

Vectors are are $[n,\ 1]$ or $[1,\ n]$ collections of numbers. They may be arranged in column form, as in (5.5), or in row form:

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

Each x_i in the vector x is called an element of x.

addition and subtraction of vectors are accomplished element by element. The addition of a $[1,\ n]$ vector x and a $[1,\ n]$ vector y produces a $[1,\ n]$ vector z :

$$z_i\ =\ x_i\ +\ y_i \qquad i\ =\ 1,\ \ldots,\ n \tag{5.7}$$

$$\begin{bmatrix} z_1 & z_2 & z_3 & \ldots & z_n \end{bmatrix} = \begin{bmatrix} x_1 + y_1 & x_2 + y_2 & x_3 + y_3 & \ldots & x_n + y_n \end{bmatrix} \tag{5.8}$$

Multiplying a vector by a scalar is performed element by element:

$$z_i \;=\; a\; x_i \tag{5.9}$$

$$\left[\begin{array}{cccc} z_1 & z_2 & \ldots & z_n \end{array}\right] \;=\; \left[\begin{array}{cccc} ax_1 & ax_2 & \ldots & ax_n \end{array}\right] \tag{5.10}$$

Multiplication by a scalar changes the magnitude of each element.

Linear combinations of vectors are made by adding (or subtracting) the products of vectors and scalars. The dimension of the resulting vector is the same as the dimension of the vectors being summed.

Transposing vectors allows us to convert from [1, n] to [n , 1]. These vectors are not equivalent. In math notation, the transpose of a vector (or a matrix) is noted with a superscript capital T or a "prime," x^T or x':

$$\left[\begin{array}{cccc} x_1 & x_2 & \ldots & x_n \end{array}\right]^T \;=\; \left[\begin{array}{c} x_1 \\ x_2 \\ \vdots \\ x_n \end{array}\right] \tag{5.11}$$

In MATLAB, a single quote immediately following the vector (or matrix) name is used to take the transpose.

The inner product or dot product, of two vectors is the sum of the products of the *i'th* elements of the vectors. Thus, it is a scalar quantity. Consider the row vector x and the column vector y:

$$d \;=\; x \;\cdot\; y \tag{5.12}$$

$$d \;=\; \sum_{i=1}^{n} x_i y_i \tag{5.13}$$

As you can guess from the definition, the length of the two vectors x and y must be the same. According to the rules of linear algebra, vector (or matrix) multiplication can only be performed if the number of columns in the first vector (or matrix) is equal to the number of rows in the second vector (or matrix). The the dimension of the resulting quantity is the number of rows in the first vector and the number of columns in the second vector. It is clear from this that the inner product can only be made by multiplying a row vector [1, n] by a column vector [n, 1].

The outer product is the tensor product of two vectors with dimensions equal to the lengths of the vectors. Consider again the row vector x and the column vector y :

$$\boldsymbol{A} = x \otimes y \tag{5.14}$$

$$a_{ij} = y_i x_j \tag{5.15}$$

5.2.2 matrices

Matrices are rectangular arrays of numbers, as in $\boldsymbol{A}$ of equation (5.6). The individual elements of vectors were identified using single subscripts. The elements of matrices are identified with two subscripts. In lab 1, we considered an example of a matrix often used in geology, the stress tensor.

Matrix addition and subtraction is very similar to vector addition and subtraction. Matrices of the same dimensions $[m, n]$ are combined element by element. For addition:

$$\boldsymbol{A} + \boldsymbol{B} = \boldsymbol{C} \tag{5.16}$$

$$a_{ij} + b_{ij} = c_{ij} \qquad i = 1, \ldots, m \; j = 1, \ldots, n \tag{5.17}$$

Multiplication by a scalar is also very similar to the multiplication of a scalar and a vector. The resulting matrix has the same dimensions as the original matrix and its elements are the products of the scalar and the elements of the original matrix.

$$\boldsymbol{B} = \delta \boldsymbol{A} \tag{5.18}$$

$$b_{ij} = \delta \; a_{ij} \tag{5.19}$$

A matrix transpose is similar to a vector transpose. The rows are converted into columns and the columns are converted into rows.

$$\boldsymbol{B} = \boldsymbol{A}^T \tag{5.20}$$

$$b_{ij} = a_{ji} \qquad i = 1, \ldots, m \; j = 1, \ldots, n \tag{5.21}$$

Matrix-Vector products are found in many numerical methods for solving equations. If we consider just the first row of equation (5.5), we see a row vector a multiplied by a column vector y producing a scalar b_1. This is the inner product of the first row of the matrix $\boldsymbol{A}$ with the column vector y. The operation is exactly that performed in (5.12) and (5.13) except that we are selecting one of the rows of the matrix to be the row vector. For row m :

$$\begin{bmatrix} a_{m1} & a_{m2} & a_{m3} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = b_m \tag{5.22}$$

To complete the matrix-vector multiplication, we perform the same row-vector by column-vector multiplication for each of the rows of $\boldsymbol{A}$:

$$b_i = \sum_{j=1}^{n} a_{ij}\ y_j \tag{5.23}$$

Not surprisingly, the dimension rules for matrix-vector products follow from the rules for vector dot products. The number of columns in the matrix $\boldsymbol{A}$ must be the same as the number of rows in the column vector y. That is, the dimensions must be: $[m,\ n\ ]\ [n,\ 1] = [m,\ 1]$. The inner dimensions of the vector and the matrix must be the same.

The computation represented by equation (5.23) can be written as a **for** loop:

```
b=zeros(M,1);
for i=1:M
        for j=1:N
                b(i) = A(i,j)*y(j) + b(i);
        end
end
```

MATLAB performs this operation automatically when we type:

```
>> b = A * y;
```

as long as the dimensions of the matrix and vector agree.

Matrix products are computed in much the same way, using row and column pairs. Using summation notation, a matrix product is expressed:

$$c_{ij} = \sum_{k=1}^{n} a_{ik}\ d_{kj} \tag{5.24}$$

The dimensions must be: $[m,\ n]\ [n,\ p] = [m,\ p]$. MATLAB performs this operation automatically when we type:

```
>> C = A * D;
```

This is a matrix product, not a collection of element-by-element products.

The inverse $\boldsymbol{A}^{-1}$ of a matrix $\boldsymbol{A}$ is defined:

$$\begin{aligned} \boldsymbol{A}^{-1}\ \boldsymbol{A} &= \boldsymbol{I} \\ \boldsymbol{A}\ \boldsymbol{A}^{-1} &= \boldsymbol{I} \end{aligned} \tag{5.25}$$

where $\boldsymbol{I}$ is a special matrix called the **identity matrix** and the dimension of $\boldsymbol{A}$ is $[m,\ n]$ with $m = n$. The identity matrix is diagonal, with ones on the diagonal. The inverse of $\boldsymbol{A}$ may or may not exist. If $\boldsymbol{A}^{-1}$ exists, then $\boldsymbol{A}$ is classified as invertible and is nonsingular (recall section 5.1.1). Matrix inverses and the identity matrix are useful in solving systems of equations and in evaluating the quality of the calculation being attempted.

The inverse of $\boldsymbol{A}$ may be constructed in one of several ways, for example using Gaussian elimination. An analytic solution, formed using the determinant $|\boldsymbol{A}|$ and the adjugate matrix

$$\boldsymbol{A}^{-1} \ = \ \frac{1}{|\boldsymbol{A}|} \ \text{adj}\,(\boldsymbol{A}) \tag{5.26}$$

is good for demonstration but impractical for large problems. The adjugate matrix is the transpose of the matrix of cofactors C_{ij}

$$C_{ij} \ = \ (-1)^{i+j} M_{ij} \tag{5.27}$$

in which M_{ij} is the determinant of the minor ij.

5.2.3 vector and matrix norms

Norms are measures of the magnitude of vectors and matrices. This is different than the dimension of the vector or matrix. The dimension is simply the number of rows and the number of columns (m and n in the above sections). Norms give us information about the contents of the vector or matrix. There are many types of norms of vectors and matrices, which are used for different purposes. Two uses of norms have some obvious physical meaning to us as geoscientists.

A norm we use often in geographic problems is called the "L_2" norm:

$$\|x\|_2 \ = \ (x_1^2 \ + \ x_2^2 + \ \ldots \ + \ x_n^2)^{1/2} \tag{5.28}$$

$$\|x\|_2 \ = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}$$

which we recognize as a calculation of the length of the vector x. The more general expression for a group of norms called the "p-norms" is:

$$\|x\|_p \ = \left(\sum_{i=1}^{n} x_i^p\right)^{1/p} \tag{5.29}$$

The double vertical bars are the "norm operator" (this is the same type of language we used when considering finite differences, the operator represents a specific manipulation of the terms it contains). For the L_2 norm, $p = 2$.

Another useful norm is calculates the angle θ between two vectors x and y:

$$\cos\ \theta\ =\ \frac{x^T y}{\|x\|_2\ \|y\|_2} \tag{5.30}$$

A special set of vectors that are very useful in numerical models are **orthogonal**, or perpendicular, vectors. According to (5.30), the angle between two vectors is $\pi/2$ when $x^T y = 0$. That is, vectors are orthogonal when their inner product (5.12 and 5.13) is zero.

Norms can be used to determine how similar two vectors are. It was easy to see that equations (5.2) were singular because they are a small system. That determination would be more difficult if we were dealing with, say, 100 equations and 100 unknowns. Norms provide us with an efficient mathematical means by which to evaluate singularity or near-singularity. This is very important when we build numerical models because if a problem is ill-posed, it will produce poor results (if any at all).

Matrix norms provide a scalar measure of the magnitude of a matrix, just as vector norms do for vectors. The formulas for matrix norms, and their solution, are a bit more complicated than those for vectors. One simple norm can be calculated by repeating the operation in (5.29) twice, with one summation over the columns and a second summation over the rows.

5.3 solving systems of equations

5.3.1 the basics

A linear system of equations may be represented

$$\boldsymbol{A}\ y\ =\ b$$

in which $\boldsymbol{A}$ is an $n \times n$ matrix and y and b are $n \times 1$ column vectors. We know the coefficients in $\boldsymbol{A}$ and the independent variables in b and wish to solve for y. Recalling the definition of the inverse $\boldsymbol{A}^{-1}$, we recognize that if $\boldsymbol{A}$ on the left-hand side of the equation is multiplied by its inverse, we will be left with the identity matrix multiplied by y. That is, we will be left with y alone on the left-hand side of the equation. Start by multiplying both sides of the equation by $\boldsymbol{A}^{-1}$.

$$\boldsymbol{A}^{-1}\ \boldsymbol{A}\ y\ =\ \boldsymbol{A}^{-1}\ b \tag{5.31}$$

$$\boldsymbol{I}\ y\ =\ \boldsymbol{A}^{-1}\ b \tag{5.32}$$

$$y\ =\ \boldsymbol{A}^{-1}\ b \tag{5.33}$$

Finding $\boldsymbol{A}^{-1}$ becomes very time consuming as the dimensions of the matrices and vectors increase. Fortunately, a variety of clever solution methods exist. In this class, all we need to know is that such alternatives exist and that MATLAB has functions to carry them out for us. MATLAB's default solver is the backslash operator. A summary of how the backslash operator behaves is given in section 8.4.3 of your text.

5.3.2 an example: coupled linear ODEs

Back in chapter 3, we solved a set of coupled ordinary differential equations for a radioactive isotope decay series by summing terms in a nested **for** loop. We can accomplish the same task using linear algebra. The decay equation for a single isotope is:

$$\frac{dc}{dt} = -\lambda\, c \tag{5.34}$$

where c represents the concentration of the isotope at any time t and λ is an empirically-derived decay constant for that isotope. For a decay chain with one radioactive daughter and one stable daughter, we would write:

$$\frac{dc_1}{dt} = -\lambda_1\, c_1 \tag{5.35}$$

$$\frac{dc_2}{dt} = \lambda_1\, c_1 - \lambda_2\, c_2 \tag{5.36}$$

$$\frac{dc_3}{dt} = \lambda_2\, c_2 - \lambda_3\, c_3 \tag{5.37}$$

where the subscripted numeral indicates a particular isotope. The system of equations may be written in matrix form:

$$\begin{bmatrix} c_1^{(1)} \\ c_2^{(1)} \\ c_3^{(1)} \end{bmatrix} = \begin{bmatrix} -\lambda_1 & 0 & 0 \\ \lambda_1 & -\lambda_2 & 0 \\ 0 & \lambda_2 & -\lambda_3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} \tag{5.38}$$

where the parenthetical superscript indicates the order of differentiation with respect to t. More compactly, we may write:

$$c^{(1)} = \boldsymbol{A}c \tag{5.39}$$

in which the matrix $\boldsymbol{A}$ contains the coefficients in the system of equations.

An Euler single-step solution to the system of equations is:

$$\begin{bmatrix} c_1(j+1) \\ c_2(j+1) \\ c_3(j+1) \end{bmatrix} = \begin{bmatrix} c_1(j) \\ c_2(j) \\ c_3(j) \end{bmatrix} + \triangle t \begin{bmatrix} -\lambda_1 & 0 & 0 \\ \lambda_1 & -\lambda_2 & 0 \\ 0 & \lambda_2 & -\lambda_3 \end{bmatrix} \begin{bmatrix} c_1(j) \\ c_2(j) \\ c_3(j) \end{bmatrix} \tag{5.40}$$

or

$$c(j+1,:) = c(j,:) + \triangle t\ \boldsymbol{A}c(j,:) \tag{5.41}$$

The following lines of MATLAB code set up the parts of equation (5.41) for a parent, one radioactive daughter and one stable daughter isotope. The only tricky part is to keep the signs straight in the coefficient matrix, A. Each row of A represents an isotope in the decay series. The coefficients along the diagonal of A represent the radioactive decay of that isotope while the (lower) off-diagonal coefficients represent the accumulation of that isotope due to decay of its parent. In the script below, the coefficients are inserted into A in one **for** loop and the signs, positive for accumulation and negative for loss, are adjusted in a second **for** loop.

```
%* Euler single-step solution for decay series system of ODEs
clear

lambda=[1 1 0];          % decay constants, incl. stable daughter
c0=[1 0 0];              % initial conditions on c
t0=0;                    % initial time
tf=10;                   % ending time for calculation
h=0.05;                  % step size

t=[t0:h:tf]';            % independent variable
M=length(lambda);        % number of isotopes
N=length(t)-1;           % number of steps, N+1 = number of nodes

C=zeros(N+1, M);         % dependent variable
C(1,:)=c0;               % initial conditions

A=zeros(M,M);            % coefficients, -ve loss, +ve accumulation
A(1,1)=-lambda(1);       % primary parent, decay
for m=2:M                % daughters
        A(m,m-1:m)=lambda(m-1:m);
        A(m,m)=-1*A(m,m);        % decay is negative
end
```

5.4 exercises

In exercises 1 through 5, you are asked to carry out some matrix calculations using MATLAB, all of which can be performed in the command window. If a question has multiple parts, you need only turn in written answers for the parts of the question identified by letters (a, b, etc.).

1. The expression

$$\gamma\, x \;+\; \delta\, y$$

   is a linear combination of vectors γx and δy where γ and δ represent scalars and x and y represent vectors. Suppose that x and y are [4, 1] vectors. Write out on paper a full vector equation (show all the terms, as in equations (5.8) and (5.10)) that represents the original equation and the result of its operations, using subscript notation.

   In MATLAB, the multiplications and addition would be accomplished:

```
>> gamma*x + del*y
```

2. Suppose you have two [1, 3] vectors: $x = [1\ 2\ 3]$, $y = [1\ 1\ 4]$

   (a) Write out on paper the equation you would use to compute the inner product $x \cdot y$.

   (b) In MATLAB, the inner product $x \cdot y$ would be calculated:

   ```
   >> x * y'
   ```

   Why is the transpose necessary?

   (c) What is the value of the inner product of the two vectors given in this question?

   (d) What is the outer product $x \otimes y$?

3. Using MATLAB, construct two [3, 3] arrays called `A` and `B`. You may place any numeric values you choose in the 9 elements of the arrays. In MATLAB, matrix addition and subtraction is performed simply by writing the relevant algebraic expression:

   ```
   >> C = A + B
   ```

   Try that out.

   (a) Next, make a third array, `x` that is a [3, 1] vector. Try adding `A` and `x`. We know by the rules of matrix algebra that this won't work. What is the error message that MATLAB produces?

   (b) Suppose you wanted to compute a the product of the matrix `A` and the vector `x`. One way is to emulate how we would perform the calculation manually, using a **for** loop structure like the one in section 5.2.2. How many **for** loops would be needed to do this?

   Happily, MATLAB relieves us from these very cumbersome matrix-vector multiplications. Try performing the matrix and vector multiplication using MATLAB:

   ```
   >> C = A * x
   ```

   (c) What is the dimension of `C`?

   (d) Try multiplying `A` and the transpose `x'` (a row vector instead of a column vector). What is the error message that MATLAB produces?

4. Create a set of arrays to represent the following system of equations:

$$\begin{aligned} 2y_1 + y_2 - 2y_3 &= 9 \\ 5y_1 + y_2 + y_3 &= 15 \\ 2y_1 - 3y_2 + 3y_3 &= 5 \end{aligned} \tag{5.42}$$

   Store the coefficients in an array called `A`, the y's in an array called `y`, and the constants in an array called `b`.

(a) What are the dimensions of A, y, and b?

(b) What is the inverse of the matrix A ? You may wish to use MATLAB's **inv** function.

(c) Solve the system of equations 5.42 using the method described in section 5.3.1. What are the resulting values of y?

5. A variety of matrix simplification and solution methods are available, such as Gaussian elimination, depending on the characteristics of the matrix involved. We will use MATLAB's backslash operator \ (or mldivide, "matrix left divide"). The backslash operator can handle exactly-, over-, and underdetermined systems of equations.

```
>> y = A\b;
```

(a) Use the backslash operator to solve the system of equations (5.42). What are the resulting values of y?

(b) Is this result exactly the same as the result when you followed equation (5.33)?

6. Solve the linear system of equations:

$$\begin{aligned} 3x_1 + 4x_2 - x_3 &= 20 \\ 2x_1 - 3x_2 &= 0 \\ 5x_2 - 3x_3 &= 20 \end{aligned} \tag{5.43}$$

for the set of x_i. There is more than one way to do this so be sure to show your work as well as the result.

7. Solve the linear system of equations:

$$\begin{aligned} x_1 + 3x_2 &= -2 \\ 3x_1 + 4x_2 + x_3 &= 0 \\ x_2 + 8x_3 + 2x_4 &= 0 \\ 5x_3 - 3x_4 &= -2 \end{aligned} \tag{5.44}$$

for the set of x_i. There is more than one way to do this so be sure to show your work as well as the result.

8. Solve the linear system of equations:

$$\begin{aligned} x_1 &= -\gamma \\ -\frac{\alpha}{2\beta}x_1 + x_2 + \frac{\alpha}{2\beta}x_3 &= 0 \\ -\frac{\alpha}{2\beta}x_2 + x_3 + \frac{\alpha}{2\beta}x_4 &= 0 \\ -\frac{\alpha}{2\beta}x_3 + x_4 + \frac{\alpha}{2\beta}x_5 &= 0 \\ -\frac{\alpha}{2\beta}x_4 + x_5 + \frac{\alpha}{2\beta}x_6 &= 0 \\ x_6 &= -\gamma \end{aligned} \tag{5.45}$$

in which $\alpha = 2$, $\beta = 0.2$, and $\gamma = 5$. Write a for loop to construct a matrix A to store the coefficients in the equations. Be sure to show your work as well as the result.

9. Following the procedure in section 5.3.2, write an Euler single step numerical integration scheme for the equation of a series RLC circuit as a system of simultaneous equations. The equation of the circuit is

$$L\frac{\mathrm{d}^2 i}{\mathrm{d}t^2} + R\frac{\mathrm{d}i}{\mathrm{d}t} + \frac{1}{C}i \;=\; 0 \tag{5.46}$$

in which i represents the current and t represents time. R, L, and C represent resistance of the resistor, inductance of the inductor, and capacitance of the capacitor in the circuit, respectively. The first derivative of the current is the product of the voltage v in the circuit and the inverse of the inductance L.