Some algebra underlying the linear regression model

Loïc Yengo

CNSG, University of Queensland

June 29th, 2016

Contents

Linear Regression

Definition

For an individual i, we model a quantitative trait y_i as

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i \quad (i = 1, \ldots, n)$$

$\beta_0, \ldots, \beta_p$ are the regression coeffients and $x_{i1}, \ldots, x_{ip}$ will be called covariates.

Objective

- Estimate the regression coeffients
- Test assumptions about these coefficients
- Make predictions for new data

Linear Regression

Definition

For an individual i, we model a quantitative trait y_i as

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i \quad (i = 1, \ldots, n)$$

$\beta_0, \ldots, \beta_p$ are the regression coeffients and $x_{i1}, \ldots, x_{ip}$ will be called covariates.

Objective

- Estimate the regression coeffients
- Test assumptions about these coefficients
- Make predictions for new data

(Part I)
Red pill or blue pill?

What is a matrix?

A n-by-p **matrix** $\boldsymbol{X}$ is a two-dimensional table with n rows and p columns that contains (real) numbers. n and p are called matrix dimensions.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & \dots & x_{1p} \\ \vdots & x_{ij} & \vdots \\ x_{n1} & \dots & x_{np} \end{bmatrix}$$

The x_{ij}'s are called matrix entries.

An example

$$\boldsymbol{X} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \text{ is a 2-by-2 matrix.}$$

A **vector** is a special case of a matrix that has only one column.

What is a matrix?

A n-by-p **matrix** $\boldsymbol{X}$ is a two-dimensional table with n rows and p columns that contains (real) numbers. n and p are called matrix dimensions.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & \dots & x_{1p} \\ \vdots & x_{ij} & \vdots \\ x_{n1} & \dots & x_{np} \end{bmatrix}$$

The x_{ij}'s are called matrix entries.

An example

$$\boldsymbol{X} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \text{ is a 2-by-2 matrix.}$$

A **vector** is a special case of a matrix that has only one column.

We assume that $\boldsymbol{X}$ is a n-by-p matrix.

0-Transposition

The transpose of matrix $\boldsymbol{X}$ (denoted ${}^t\boldsymbol{X}$, $\boldsymbol{X}^T$ or $\boldsymbol{X}'$) is a p-by-n matrix defined as

$$\boldsymbol{X}' = \begin{bmatrix} x_{11} & \dots & x_{1n} \\ \vdots & x_{ji} & \vdots \\ x_{p1} & \dots & x_{pn} \end{bmatrix}$$

1-Multiplying by a number a

$a\boldsymbol{X}$ is also n-by-p matrix which entries have all been multiplied by a

$$a\boldsymbol{X} = \begin{bmatrix} a \times x_{11} & \dots & a \times x_{1p} \\ \vdots & a \times x_{ij} & \vdots \\ a \times x_{n1} & \dots & a \times x_{np} \end{bmatrix}$$

2-Matrix addition

We can only add matrices that have the same dimensions. If $\boldsymbol{Z}$ is also a n-by-p matrix then

$$\boldsymbol{X}+\boldsymbol{Z}=\begin{bmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & x_{ij} & \vdots \\ x_{n1} & \cdots & x_{np} \end{bmatrix}+\begin{bmatrix} z_{11} & \cdots & z_{1p} \\ \vdots & z_{ij} & \vdots \\ z_{n1} & \cdots & z_{np} \end{bmatrix}$$

$$=\begin{bmatrix} x_{11}+z_{11} & \cdots & x_{1p}+z_{1p} \\ \vdots & x_{ij}+z_{ij} & \vdots \\ x_{n1}+z_{n1} & \cdots & x_{np}+z_{np} \end{bmatrix}$$

This rule can be generalized to calculate the sum of any number of matrices.

3-Matrix multiplication

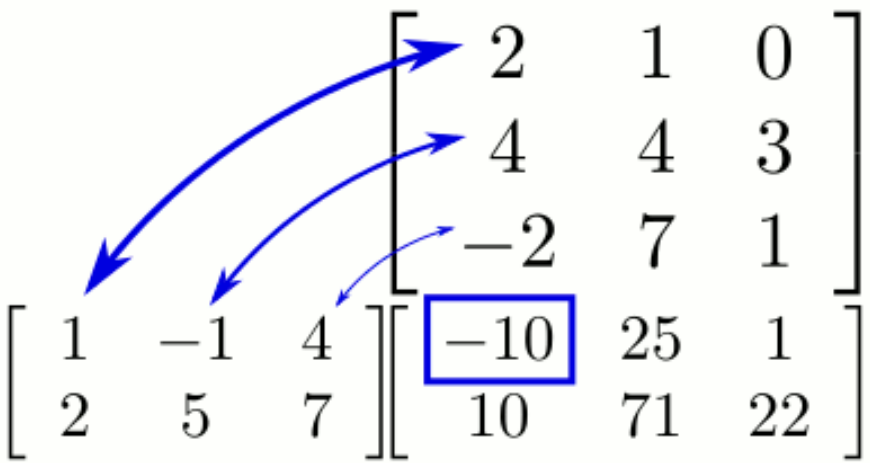
We can only multiply a $n \times p$-matrix with a $p \times m$-matrix. If $\boldsymbol{Z}$ is a p-by-m matrix then $\boldsymbol{XZ}$ is a n-by-m matrix defined as

$$(\boldsymbol{XZ})_{i,k} = x_{i,1} \times z_{1,k} + x_{i,2} \times z_{2,k} + \ldots + x_{i,p} \times z_{p,k}$$

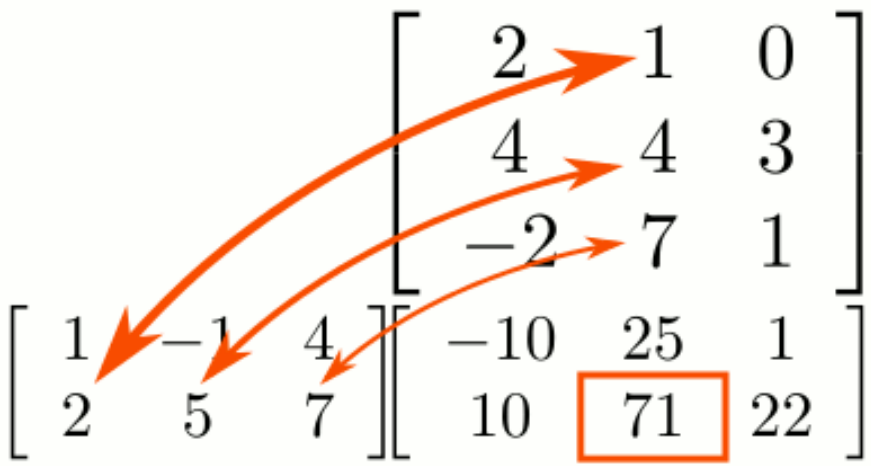
3-Matrix multiplication

We can only multiply a $n \times p$-matrix with a $p \times m$-matrix. If $\boldsymbol{Z}$ is a p-by-m matrix then $\boldsymbol{XZ}$ is a n-by-m matrix defined as

$$(\boldsymbol{XZ})_{i,k} = x_{i,1} \times z_{1,k} + x_{i,2} \times z_{2,k} + \ldots + x_{i,p} \times z_{p,k}$$



$$1 \cdot 2 + (-1) \cdot 4 + 4 \cdot (-2) = -10$$



$$2 \cdot 1 + 5 \cdot 4 + 7 \cdot 7 = 71$$

Identity matrix and inverse

We call a square matrix, a matrix with as many rows as columns. One special square matrix is the **Identity matrix** defined in dimension n as

$$\mathrm{I}_n = \underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & 0 \\ & & \ddots & \\ & 0 & & 1 \\ & & & & 1 \end{bmatrix}}_{n \text{ columns}} \Big\} n \text{ rows}$$

A $n \times n$ matrix $\boldsymbol{A}$ is said invertible, if there exists a unique $n \times n$ matrix $\boldsymbol{B}$ such as $\boldsymbol{AB} = \boldsymbol{BA} = \mathrm{I}_n$. If such a matrix exist then $\boldsymbol{B}$ is called inverse of $\boldsymbol{A}$ and is noted $\boldsymbol{A}^{-1}$.

How can I tell if a matrix is invertible?

Tough question in general, but we can admit this somewhat satisfactory answer...$\boldsymbol{A}$ is invertible if and only if its determinant does equal 0. OK then, but what is the determinant?

Example

$$\boldsymbol{X} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \text{ then } |\boldsymbol{X}| = 1 \times 4 - 2 \times 3 = -2.$$

$$\boldsymbol{X} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \text{ then } |\boldsymbol{X}| = 1 \times 4 - 2 \times 2 = 0$$

The determinant measures how the columns or the rows of $\boldsymbol{X}$ are linearly related.

How can I tell if a matrix is invertible?

Tough question in general, but we can admit this somewhat satisfactory answer...$\boldsymbol{A}$ is invertible if and only if its determinant does equal 0. OK then, but what is the determinant?

Example

$$\boldsymbol{X} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \text{ then } |\boldsymbol{X}| = 1 \times 4 - 2 \times 3 = -2.$$

$$\boldsymbol{X} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \text{ then } |\boldsymbol{X}| = 1 \times 4 - 2 \times 2 = 0$$

The determinant measures how the columns or the rows of $\boldsymbol{X}$ are linearly related.

Matrix rank

The **rank** of a $n \times p$ matrix $\boldsymbol{A}$ is the number of linearly independent rows or columns.

$\text{rank}(\boldsymbol{A}) \leq \min(n, p)$.

Matrix rank

The **rank** of a $n \times p$ matrix $\boldsymbol{A}$ is the number of linearly independent rows or columns.
$\text{rank}(\boldsymbol{A}) \leq \min(n, p)$.

When the $n = p$ (square matrices), then $\boldsymbol{A}$ is invertible if and only if $\text{rank}(\boldsymbol{A}) = n$ (full rank matrix).

(Part II)
Linear regression in matrix terms

Reformulation

We first defined the linear regression model using the following equation

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i \quad (i = 1, \ldots, n)$$

this looks pretty much like a matrix product right?

Reformulation

We first defined the linear regression model using the following equation

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i \quad (i = 1, \ldots, n)$$

this looks pretty much like a matrix product right?

$$y_i = \underbrace{\left[1, x_{i1}, x_{i2}, \ldots, x_{ip}\right]}_{1\times(p+1)} \times \underbrace{\begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}}_{(p+1)\times 1} + \varepsilon_i \quad (i = 1, \ldots, n)$$

Therefore, if we note

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \ \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_p \end{bmatrix}, \ \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

and

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & \dots & x_{1p} \\ \vdots & x_{ij} & \vdots \\ x_{n1} & \dots & x_{np} \end{bmatrix}$$

The linear regression model can we reformulated as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

Estimation

I assume that the we know the distribution of the ε_i's: $\varepsilon_i \sim \mathcal{N}\left(0, \sigma^2\right)$. Under this assumption we can estimate $\boldsymbol{\beta}$ using maximum likelihood. The log likelihood is written as

$$\begin{aligned}
\log p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\beta}) &= -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right)'\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right) \\
&= -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(\boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta}\right) \\
&= C - \frac{1}{2\sigma^2}\left[\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y}\right]
\end{aligned}$$

To find the maximum with respect to $\boldsymbol{\beta}$ we have to solve the following equation:

$$\frac{\partial \log p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\frac{1}{2\sigma^2}\left[2\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} - 2\boldsymbol{X}'\boldsymbol{y}\right] = \boldsymbol{0}$$

If the $p \times p$-matrix $\boldsymbol{X}'\boldsymbol{X}$ is invertible, then the maximum likelihood estimate (MLE) of β is given by

$$\widehat{\beta} = \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1} \boldsymbol{X}'\boldsymbol{y}$$

Otherwise the MLE does not exist. Why would this happen?

1. Colinearity: some variables are too correlated with one another.
2. High dimensionality: the number of covariates (p) exceeds the sample size (n).

(Part III)
Matrix decomposition and inverse

The case of square matrices

Once upon a time, a famous mathematician named Karl Weierstrass proposed the **Spectral Decomposition Theorem** which says in substance that a symmetric matrix $\boldsymbol{X}$ filled with real numbers can be discomposed as $\boldsymbol{A} = \boldsymbol{PDP}^{-1}$.
where $\boldsymbol{P}$ verifies that $\boldsymbol{P}'\boldsymbol{P} = \mathrm{I}$ (i.e. $\boldsymbol{P}^{-1} = \boldsymbol{P}'$) and $\boldsymbol{D}$ is a diagonal matrix.

The case of square matrices

Once upon a time, a famous mathematician named Karl Weierstrass proposed the **Spectral Decomposition Theorem** which says in substance that a symmetric matrix $\boldsymbol{X}$ filled with real numbers can be discomposed as $\boldsymbol{A} = \boldsymbol{PDP}^{-1}$.
where $\boldsymbol{P}$ verifies that $\boldsymbol{P}'\boldsymbol{P} = \mathrm{I}$ (i.e. $\boldsymbol{P}^{-1} = \boldsymbol{P}'$) and $\boldsymbol{D}$ is a diagonal matrix.

Few more definition/operations

A symmetric matrix $\boldsymbol{A}$ is a square matrix such as $\boldsymbol{A}' = \boldsymbol{A}$.
E.g:

$$\boldsymbol{X} = \begin{bmatrix} 1 & -3 \\ -3 & 4 \end{bmatrix} \text{ then } \boldsymbol{X}' = \boldsymbol{X}$$

One trick we did not mention earlier is that $(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$ and another a bit similar is $(\boldsymbol{AB})' = \boldsymbol{B}'\boldsymbol{A}'$.

The case of square matrices

Once upon a time, a famous mathematician named Karl Weierstrass proposed the **Spectral Decomposition Theorem** which says in substance that a symmetric matrix $\boldsymbol{X}$ filled with real numbers can be discomposed as $\boldsymbol{A} = \boldsymbol{PDP}^{-1}$.
where $\boldsymbol{P}$ verifies that $\boldsymbol{P}'\boldsymbol{P} = \mathrm{I}$ (i.e. $\boldsymbol{P}^{-1} = \boldsymbol{P}'$) and $\boldsymbol{D}$ is a diagonal matrix.

Few more definition/operations

A symmetric matrix $\boldsymbol{A}$ is a square matrix such as $\boldsymbol{A}' = \boldsymbol{A}$.
E.g:

$$\boldsymbol{X} = \begin{bmatrix} 1 & -3 \\ -3 & 4 \end{bmatrix} \text{ then } \boldsymbol{X}' = \boldsymbol{X}$$

One trick we did not mention earlier is that $(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$ and another a bit similar is $(\boldsymbol{AB})' = \boldsymbol{B}'\boldsymbol{A}'$.

The case of square matrices

Once upon a time, a famous mathematician named Karl Weierstrass proposed the **Spectral Decomposition Theorem** which says in substance that a symmetric matrix $\boldsymbol{X}$ filled with real numbers can be factorized as $\boldsymbol{A} = \boldsymbol{PDP}^{-1}$.
where $\boldsymbol{P}$ verifies that $\boldsymbol{P}'\boldsymbol{P} = \mathrm{I}$ (i.e. $\boldsymbol{P}^{-1} = \boldsymbol{P}'$) and $\boldsymbol{D}$ is a diagonal matrix.

We can remark that $\boldsymbol{X}'\boldsymbol{X}$ is symmetric because $\left(\boldsymbol{X}'\boldsymbol{X}\right)' = \boldsymbol{X}'\left(\boldsymbol{X}'\right)' = \boldsymbol{X}'\boldsymbol{X}$ and therefore can decomposed as $\boldsymbol{X}'\boldsymbol{X} = \boldsymbol{PDP}^{-1}$.

The columns of $\boldsymbol{P}$ are called eigenvectors of $\boldsymbol{X}$ and the values of the diagonal of $\boldsymbol{D}$ are called eigenvalues of $\boldsymbol{X}$.

Cholesky decomposition

We now assume that matrix $\boldsymbol{A}$ is symmetric and has a determinant strictly positive. Therefore, we can find a triangular matrix $\boldsymbol{L}$ such as $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}'$. This is also known as the LU decomposition for Lower-Upper.

$$\boldsymbol{L} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix}$$

Cholesky decomposition

We now assume that matrix $\boldsymbol{A}$ is symmetric and has a determinant strictly positive. Therefore, we can find a triangular matrix $\boldsymbol{L}$ such as $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}'$. This is also known as the LU decomposition for Lower-Upper.

$$\boldsymbol{L} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix}$$

This decomposition is also interesting because calculating the inverse of a relatively easy.

Singular Vector Decomposition

More generally, any $n \times p$-matrix $\boldsymbol{X}$ can be decomposed as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}'$ where $\boldsymbol{S}$ is $n \times p$ diagonal matrix and $\boldsymbol{U}$ and $\boldsymbol{V}$ are respectively $n \times n$ and $p \times p$ such as $\boldsymbol{U}'\boldsymbol{U} = \mathrm{I}_n$ and $\boldsymbol{V}'\boldsymbol{V} = \mathrm{I}_n$

$$\boldsymbol{S} = \underbrace{\begin{bmatrix} s_{11} & & & \\ 0 & s_{22} & & \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & s_{np} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{\text{a rectangular diagonal matrix } (n<p)}$$

QR decomposition

Another usefull decomposition of a $n \times p$ matrix is $\boldsymbol{A} = \boldsymbol{QR}$ where $\boldsymbol{Q}$ is an orthonormal matrix, i.e. $\boldsymbol{Q}'\boldsymbol{Q} = \mathrm{I}_n$ and $\boldsymbol{R}$ is an upper triangular matrix.

QR decomposition

Another usefull decomposition of a $n \times p$ matrix is $\boldsymbol{A} = \boldsymbol{QR}$ where $\boldsymbol{Q}$ is an orthonormal matrix, i.e. $\boldsymbol{Q}'\boldsymbol{Q} = \mathrm{I}_n$ and $\boldsymbol{R}$ is an upper triangular matrix.

This is decomposition has a lower complexity than the SVD or the Cholesky decomposition.

How can these matrices decompositions be obtained?

There are known and efficient algorithms implemented in most statistical suites.

1. in R: **svd()**, **chol()** or **solve()**
2. in C++ (Eigen library): **SelfAdjointEigenSolver()**, **llt()** or **inverse()**

How can these matrices decompositions be obtained?

There are known and efficient algorithms implemented in most statistical suites.

1. in R: **svd()**, **chol()** or **solve()**
2. in C++ (Eigen library): **SelfAdjointEigenSolver()**, **llt()** or **inverse()**

...but you should know these involve **intensive** calculations. For a $n \times p$

1. SVD complexity : $\mathcal{O}(np^2 - p^3/3)$ operations
2. Cholesky Decomposition complexity: $\mathcal{O}(np^2 + p^3)$ operations.

Why do we care if this is already implemented somewhere?

Well...

1. Existing packages can be bugged, so if you know how it works you can fix it!

Why do we care if this is already implemented somewhere?

Well...

1. Existing packages can be bugged, so if you know how it works you can fix it!
2. Improve existing packages or implement faster algorithm for your specific problem

Why do we care if this is already implemented somewhere?

Well...

1. Existing packages can be bugged, so if you know how it works you can fix it!
2. Improve existing packages or implement faster algorithm for your specific problem
3. Understand when an approximation is needed

Some ongoing research in this area

The main driving force in this area is the improvement of computational algorithms to meet the challenges of using large dataset. There are researches on

1. Improving performances for sparse matrices (filled with a lot of zeros) decomposition.
2. Low rank approximations for SVD