Julia Tutorial Series - II

Matrix Factorization Applications

Abhijith Chandraprabhu

February 1, 2014

Overview

Introduction

Matrix Applications

What is a matrix?

A matrix is simple a 2-dimensional array of scalar real valued data.

$$A = \begin{matrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{matrix}$$

where $a_{m,n}$, are the elements of the matrix A with m rows and n columns.

Applications of Matrix Methods

- Cryptography
- Mechanical problems
- Physics problems
- Text mining, Recommender systems
- GPS
- Circuit theory
- Imaging science

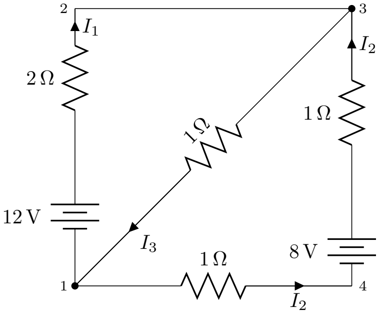
Properties of Matrix

How could a simple 2-D array of numbers solve such diverse problems?

- Matrix provides *vector-space* modeling of the system under study.
- We can perform *linear transformations* through matrices.
- Matrices help in solving *linear system of equations*.
- Infinite matrices can be used to represent derivative operator in *Taylor series*.
- *Determinant* of a matrix gives the area or the volume or parallelogram or the parallelopiped.

Electrical Circuit

How can a matrix solve this?



KCL to nodes 1 and 3:

$$I_1 + I_2 - I_3 = 0 \tag{1}$$

KVL to loops 1-2-3-1 and 1-3-4-1

$$2I_1 + I_3 = 12 \tag{2}$$

$$2I_2 + I_3 = 8 \tag{3}$$

Electrical Circuit contd.

from Eq. 1, 2 and 3

$$\begin{array}{ccccccc} 1I_1 & + & 1I_2 & - & 1I_3 & = & 0 \\ 2I_1 & + & 0I_2 & + & 1I_3 & = & 12 \\ 0I_1 & + & 2I_2 & + & 1I_3 & = & 8 \end{array}$$

Ax=b notation,

$$\begin{bmatrix} 1 & 1 & -1 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 12 \\ 8 \end{bmatrix}$$

We have 3 unknowns and 3 equations : Determined System.

[1] Refer section 1 of JuliaTutorial.jl

Regression Analysis

- Trying to formulate an equation or model which connects *response(dependent)* variable, Y to set of *predictor(independent)* variables, $X_1, X_2 ... X_p$.
- Usually data collected from observations or lab experiments have some error in them.
- Data Fitting : Given m datapoints (t_i, y_i), we try to find a function $f(t, \mathbf{x})$ which best fits the data in a *least squares* sense:

$$\min_{x} \sum_{i=1}^{m} (y_i - f(t_i, \mathbf{x}))^2$$

Polynomial regression

- $f(t, \mathbf{x}) = x_1 + x_2 t + x_3 t^2 + ... + x_n t^n - 1$
- Is a non-linear model in t, but linear in $\mathbf{x}$.
- $f(t, \mathbf{x}) = e_1^x + ln(x_2)t + x_3^2 t^2 + ... + \sqrt{x_n} t^n - 1$, this would be an example for non-linear model in the coefficients $\mathbf{x}$ and non linear in independent variable t.
- Consider a general linear model : $f(t, \mathbf{x}) = x_1 g_1(t) + x_2 g_2(t) + x_3 g_3(t) + ... + x_n g_n(t)$.
- Let a matrix $\mathbf{A}$ represent the $g_j(t_i)$, where i is the observation index and let $\mathbf{b}$ represent the i observation values y_i. If $\mathbf{x}$, is the vector of independent variables, then the above system can be represented as, $\mathbf{A}_{m,n}\mathbf{x}_{n,1} = \mathbf{b}_{m,1}$

Quadratic Regression

Given 20 observations :

t	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0
y	9.8	8.9	8.6	7.5	6.9	5.4	5.1	4.9	3.6	3.0
t	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19.0	20.0
y	2.7	3.2	4.5	5.1	5.9	6.5	6.1	7.4	8.5	9.0

$$\begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ \vdots & \vdots & \vdots \\ 1 & t_{20} & t_{20}^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{20} \end{bmatrix}$$

Text Mining

- Type of *Information retrieval*, where we try to extract relevant information from huge collection of textual data(documents).
- Documents: web pages, biomedical literature, movie reviews, research articles etc.
- Preprocess the documents to for a *corpus*.
- Identify the *lexicon* from the corpus.
- Form the *Term-Document matrix* (Numericized Text).
- Based on *vector-space* model.

[1] Refer section 3.1 of JuliaTutorial.jl

Document1

It's also important to point out that even though these arrays are generic, they're not boxed: an Int8 array will take up much less memory than an Int64 array, and both will be laid out as continuous blocks of memory; Julia can deal seamlessly and generically with these different immediate types as well as pointer types like String.

Document2

To celebrate some of the amazing work that's already been done to make Julia usable for day-to-day data analysis, I'd like to give a brief overview of the state of statistical programming in Julia. There are now several packages that, taken as a whole, suggest that Julia may really live up to its potential and become the next generation language for data analysis.

Document3

Only later did I realize what makes Julia different from all the others. Julia breaks down the second wall — the wall between your high-level code and native assembly. Not only can you write code with the performance of C in Julia, you can take a peek behind the curtain of any function into its LLVM Intermediate Representation as well as its generated assembly code — all within the REPL. Check it out.

Document4

Homoiconicity — the code can be operated on by other parts of the code. Again, R kind of has this too! Kind of, because I'm unaware of a good explanation for how to use it productively, and R's syntax and scoping rules make it tricky to pull off. But I'm still excited to see it in Julia, because I've heard good things about macros and I'd like to appreciate them.

Document5

Graphics. One of the big advantages of R over similar languages is the sophisticated graphics available in ggplot2 or lattice. Work is underway on graphics models for Julia but, again, it is early days still.

Term-Document Matrix

	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Query 1	Query 2
Array	1	0	0	0	0	0	0
continous block	1	0	0	0	0	0	0
Julia	1	3	3	1	1	0	0
types	1	0	0	0	0	0	0
string	1	0	0	0	0	0	0
amazing	0	1	0	0	0	0	0
data analysis	0	2	0	0	0	0	0
statistical computing	0	1	0	0	0	0	0
high-level	0	0	1	0	0	0	0
performance	0	0	1	0	0	0	0
LLVM	0	0	1	0	0	0	0
homoiconicity	0	0	0	1	0	0	0
R	0	0	0	2	0	0	0
syntax	0	0	0	4	0	0	0
macros	0	0	0	1	0	0	0
graphics	0	0	0	0	3	0	0
advantages	0	0	0	0	1	0	0

What is Recommender System?

Recommender system is a subclass of Information Filtering, where it predicts how the constituent members of 2 different groups, unfamiliar with each other interact.

$$
\begin{array}{rccc}
 & GroupA & & GroupB \\
NETFLIX: & [Users] & \longrightarrow & [Movies] \\
Linkedin: & [Users] & \longrightarrow & [Connections], [Groups] \\
Amazon: & [Customers] & \longrightarrow & [Products] \\
Newspaper: & [Readers] & \longrightarrow & [Articles] \\
Coursera: & [Students] & \longrightarrow & [Courses]
\end{array}
$$

Types of Recommender Systems

Explicit

NETFLIX

$$\begin{array}{cccc} & M1 & M2 & M3 \\ U1 & 5 & ? & 2 \\ U2 & ? & ? & 4 \\ U3 & 4 & 1 & ? \end{array}$$

$Ratings : 5, 4, 3, 2, 1$

Implicit

Amazon

$$\begin{array}{cccc} & P1 & P2 & P3 \\ U1 & Yes & ? & No \\ U2 & ? & ? & Yes \\ U3 & Yes & No & ? \end{array}$$

$PurchaseHistory : Yes, No$

NETFLIX

- Netflix, Inc is a company based in USA, which provides internet streaming media on-demand.
- 33 million members view over 1 billion hours of TV shows and movies through NETFLIX per month.
- Personalized service through recommendations based on previous ratings.
- In October 2006, NETFLIX announced a prize of 1 Million USD to beat *Cinematch* by 10%
- ACM conference RecSys started from 2007.

Example: Let us consider a simple toy example of 4 users and 4 movies.

	Titanic	Braveheart	The Lion King	Dreamcatcher
John	5	5	2	2
Dave	2	?	3	4
Alice	4	5	?	3
Bob	3	4	2	5

where ? denotes the *user-movie* combinations which has to be predicted.

The End