MathLight

A lightweight matrix manipulation language

Boya Song (bs3065) Chunli Fu(cf2710)
Mingye Chen (mc4414) Yuli Han(yh2986)
Motivation

- Increasing and common usage of matrices.
- Matlab: expensive, not lightweight enough
- make it as an easy, fast and flexible language and the basic syntax is similar to C.
• Design an imperative language with matrix manipulations.

• Matrix data type with convenient matrix operations.

• Rich matrix related built-in functions.
Overview

- C-like syntax
- New data type: matrix with powerful matrix-related operations and built-in functions
- Imperative
- Static scope
- Statically-typed
Data Types

<table>
<thead>
<tr>
<th>Type names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>32-bit signed integer</td>
</tr>
<tr>
<td>double</td>
<td>64-bit double precision float-point number</td>
</tr>
<tr>
<td>boolean</td>
<td>1-bit logical value</td>
</tr>
<tr>
<td>string</td>
<td>string data</td>
</tr>
<tr>
<td>matrix</td>
<td>one or two dimensional matrix data with double type</td>
</tr>
<tr>
<td>void</td>
<td>no type</td>
</tr>
</tbody>
</table>

Matrix Literal

matrix m<2,2> = [1,2;3,4];  \[1.0 \quad 2.0 \\
3.0 \quad 4.0\]

Declare

int a;
matrix b<2,3>;
matrix b<5>;

Declare & Assign

int a = 0;
matrix b<2,3> = [1,2,3;3,4,4];
matrix c<5> = [1,2,3,4,5];
matrix d<2,2> = fill(2,2,3.0);
# Arithmetic Operators & Built-in Functions

<table>
<thead>
<tr>
<th>Operators</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition(int, double, matrix)</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction(int, double, matrix)</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication(int, double, matrix)</td>
</tr>
<tr>
<td>/</td>
<td>Division(int, double)</td>
</tr>
<tr>
<td>^</td>
<td>Power(int, double)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>.*</td>
<td>Element-wise multiplication for matrix</td>
</tr>
<tr>
<td>./</td>
<td>Element-wise division for matrix</td>
</tr>
<tr>
<td>'</td>
<td>Transpose for matrix</td>
</tr>
</tbody>
</table>
func int main() {
    int a = 1;
    double b = 2.5;
    matrix c<2,3> = [2.3,4.2,3.3;-7.6,-3.4,4.5];
    matrix d<3,2> = [2,4;3,4;5,6];
    print(a + d);
    print(" ");
    print(c * d);
    return 0;
}

dyn-209-2-227-157:PLT-MathLight yulihan$ lli test.lli
3 5
4 5
6 7

33.7 45.8
-2.9 -17
### Arithmetic Operators & Built-in Functions

<table>
<thead>
<tr>
<th>Operators</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition(int, double, matrix)</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction(int, double, matrix)</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication(int, double, matrix)</td>
</tr>
<tr>
<td>/</td>
<td>Division(int, double)</td>
</tr>
<tr>
<td>^</td>
<td>Power(int, double)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>.*</td>
<td>Element-wise multiplication for matrix</td>
</tr>
<tr>
<td>./</td>
<td>Element-wise division for matrix</td>
</tr>
<tr>
<td>.'</td>
<td>Transpose for matrix</td>
</tr>
</tbody>
</table>

Operators: int, double, matrix; Functions: +, -, ×
General built-in functions:

print: support printing for int, double, string and matrix

sqrt(int a)/sqrt(double a)      log(int a)/log(double a)

```c
func int main() {
    int a=1;
    double b = 2;
    matrix c<2,3> = [2.3, 4.2, 3.3; -7.6, -3.4, 4.5];
    string s = "hello world";
    print(a);
    print(b);
    print(c);
    print(s);
    return 0;
}
```

```
dyn-209-2-226-156:PLT-MathLight yulihan$ .test-pre.exe
1
2.5
2.3 4.2 3.3
-7.6 -3.4 4.5
hello world
```
Arithmetic Operators & Built-in Functions

matrix-related built-in functions:

inv(matrix m): inverse matrix

det(matrix m): determinant

fill(int r, int c, double value): initialize matrix with given size and given default value

Other built-in functions: size, Euclidean norm, absolute norm, sum, mean, trace, max eigenvalue...
```go
func int main()
    matrix a<3,3> = [1,2,3;4,5,6;7,8,9];
    print("row number is:");
    print(sizeof_row(a));
    print("column number is:");
    print(sizeof_col(a));
    print("inverse matrix:");
    print(inv(a));
    print("transpose matrix:");
    print(a');
    print("determinant is:");
    print(det(a));
    print("trace is:");
    print(tr(a));
    print("the maximal eigenvalue is:");
    print(max_eigvalue(a));
    print("the absolute norm is:");
    print(norm1(a));
    print("the Euclidean norm is:");
    print(norm2(a));
    return 0;
}
Arithmetic Operators & Built-in Functions

matrix a<3,3>

1 2 3
4 5 6
7 8 9

row number is:
3
column number is:
3
inverse matrix:
0.588235 -0.235294 0.147059
-0.411765 0.102941 -0.0588235
0.411765 -0.102941 0.0588235
transpose matrix:
1 4 7
2 5 8
3 6 9
determinant is:
68
trace is:
15
the maximal eigenvalue is:
16.1168
the absolute norm is:
18
the Euclidean norm is:
16.1168
func matrix mat_add(matrix a, matrix b) {
    return a+b;
}

func int main() {
    matrix a<2,2> = [1,3;5,2];
    matrix b<2,2> = fill(2,2,3.0);
    print(mat_add(a, b));
    return 0;
}
Other features

- Support both vectors and matrices.
  
  \[
  \text{matrix } a<3> = [1,2,3];
  \]

- Matrix concatenation

  \[
  \text{matrix } a <2, 3> = [b ; c];
  \]

- Int to double casting.

  \[
  \text{int } a = 1;
  \text{double } b = 2.0;
  \text{double } res = a + b;
  \]

```c
func int main(){
    matrix col1 <2, 1> = [3.0; 1.0];
    matrix col2 <2, 1> = [2.0; 4.0];
    matrix arr <2,2> = [col1, col2];
    int a = 1;
    double b = -arr[0,0] - arr[1,1];
    double c = arr[0,0] * arr[1,1] - arr[0,1] * arr[1,0];
    double eigv1 = ( -b + sqrt(b*b - 4 * a * c)) / (2 * a);
    double eigv2 = ( -b - sqrt(b*b - 4 * a * c)) / (2 * a);
    print("Calculate eigenvalue:");
    print(eigv1);
    print(eigv2);
    return 0;
}
```
func int main() {
    matrix a <3, 3>;
    a = [1.1, 2.1, 3.1; 1.0, 2.0, 3.0; 4.1, 4.2, 4.3];
    print(a[3,3]);
    return 0;
}

~/Desktop/Courses/plt/Project/mathlight(master x) ./mathlight.native tests/failed-matrixaccess.txt > test.ll
Fatal error: exception Failure("expression SMatrix2DElement a[3, 3] out of boundary, matrix size: (3, 3")
func int main () {
    matrix i <3, 3>
    matrix j <2, 2>
    i = [1.0,2.0,3.0;1.0,2.0,3.0];
    j = [4.0,5.0;4.0,5.0];
    print(i);
    print([i; j]);
    return 0;
}

Fatal error: exception Failure("illegal Matrix Concat operator: matrix of size (3,3) : matrix of size (2,2) in MatrixOp i:j")
Work Division

- **Boya Song**: Manager / Tester
  - Integration of the whole project.
  - Implemented the basic structure of codegen.
  - Implementation of matrix inner structure, function, and some built-in functions.
  - Testing

- **Chunli Fu**: System Architect / Tester
  - Semantic checking for expressions and statements.
  - Testing for semantic checking.

- **Mingye Chen**: Language Guru / Tester
  - Syntax designing for the language.
  - Scanning and parsing for the program.
  - Testing.

- **Yuli Han**: System Architect / Tester
  - Implementation of arithmetic expressions and built-in functions.
  - Integration testing.
Thanks!

MathLight

A lightweight matrix manipulation language

Boya Song (bs3065) Chunli Fu(cf2710)
Mingye Chen (mc4414) Yuli Han(yh2986)