全局数组类型定义

密集矩阵和数组操作»参考»核心模块

详细说明

Eigen 为最常见的一维和二维数组类型定义了几个 typedef 快捷方式。

一般模式如下：

`ArrayRowsColsType`： `Rows` 和 `Cols` 可以是 `2`， `3`， `4` 对于固定大小的方阵或 `X` 动态大小， `Type` 可以是 `i` 整数，`f` 浮点数，`d` 双精度数，`cf` 复数浮点数，`cd` 复数双精度数。

例如，`Array33d` 是固定大小的 3x3 数组类型的双精度数，`ArrayXXf` 是动态大小的浮点数矩阵。

还有一些 `ArraySizeType` 是不言自明的。例如，`Array4cf` 是一个由 4 个复杂浮点数组成的固定大小的一维数组。

使用**[c++11]**，还为常见大小定义了模板别名。它们遵循与上面相同的模式，除了标量类型后缀被模板参数替换，即：

- `ArrayRowsCols<Type>` 其中 `Rows` 和 `Cols` 可以是 `2`、`3`、`4` 或 `X` 用于固定或动态大小。
- `ArraySize<Type>` 其中 `Size` 可以是 `2`、`3`、`4` 或 `X` 用于固定或动态大小的一维数组。

也可以看看

类数组

类型定义

```
template<typename Type >
using   Eigen::Array2 = Array< Type, 2, 1 >

template<typename Type >
using   Eigen::Array22 = Array< Type, 2, 2 >

template<typename Type >
using   Eigen::Array2X = Array< Type, 2, Dynamic >

template<typename Type >
using   Eigen::Array3 = Array< Type, 3, 1 >

template<typename Type >
using   Eigen::Array33 = Array< Type, 3, 3 >

template<typename Type >
using   Eigen::Array3X = Array< Type, 3, Dynamic >

template<typename Type >
using   Eigen::Array4 = Array< Type, 4, 1 >

template<typename Type >
using   Eigen::Array44 = Array< Type, 4, 4 >

template<typename Type >
```

```
using   Eigen::Array4X = Array< Type, 4, Dynamic >

template<typename Type >
using   Eigen::ArrayX = Array< Type, Dynamic, 1 >

template<typename Type >
using   Eigen::ArrayX2 = Array< Type, Dynamic, 2 >

template<typename Type >
using   Eigen::ArrayX3 = Array< Type, Dynamic, 3 >

template<typename Type >
using   Eigen::ArrayX4 = Array< Type, Dynamic, 4 >

template<typename Type >
using   Eigen::ArrayXX = Array< Type, Dynamic, Dynamic >
```