编译器对堆栈对齐做出错误假设

密集矩阵和数组操作»对齐问题

这似乎是 GCC 4.5 中已修复的 GCC 错误。如果您遇到此问题，请升级到 GCC 4.5 并向我们报告，以便我们更新此页面。

这是一个问题，到目前为止，我们只在 Windows 上遇到过 GCC：例如，MinGW 和 TDM-GCC。

默认情况下，在这样的函数中，

```
void foo()
{
  Eigen::Quaternionf q;
  //...
}
```

GCC 假设堆栈已经是 16 字节对齐的，因此对象 *q* 将在 16 字节对齐的位置创建。因此，正如Eigen要求的那样，明确对齐对象 *q* 不需要任何特别注意。

问题是，在某些特定情况下，这种假设在 Windows 上可能是错误的，因为堆栈只能保证 4 字节对齐。事实上，即使 GCC 负责在主函数中对齐堆栈并尽最大努力保持它对齐，但当从另一个线程或从另一个编译器编译的二进制文件调用函数时，堆栈对齐可能会被破坏。这会导致在未对齐的位置创建对象 “q”，使您的程序因对未对齐数组的断言而崩溃。到目前为止，我们找到了以下三个解决方案。

本地解决方案

一个本地解决方案是用这个属性标记这样的函数：

```
__attribute__((force_align_arg_pointer)) void foo()
{
  Eigen::Quaternionf q;
  //...
}
```

阅读此 GCC 文档以了解其作用。当然，这只能在 Windows 上的 GCC 上完成，因此为了可移植性，您必须将其封装在一个宏中，而在其他平台上则将其留空。此解决方案的优点是您可以精细地选择哪个函数可能具有损坏的堆栈对齐。当然，不利的一面是必须为每个此类功能都这样做，因此您可能更喜欢以下两个全局解决方案之一。

全局解决方案

一个全局的解决方案是编辑您的项目，以便在 Windows 上使用 GCC 进行编译时，将此选项传递给 GCC：

```
-mincoming-stack-boundary=2
```

解释：这告诉 GCC 堆栈只需要对齐到 2^2=4 字节，因此 GCC 现在知道它确实必须在需要时特别注意固定大小的可矢量化特征类型的 16 字节对齐。

另一个全局解决方案是将此选项传递给 gcc：

```
-mstackrealign
```

这与`force_align_arg_pointer`向所有函数添加属性具有相同的效果。

这些全局解决方案易于使用，但请注意，它们可能会减慢您的程序速度，因为它们会为每个函数带来额外的序言/尾声指令。