**计算结果对比**

**一、计算效率对比**

部件数：2

单元数：2+6

时间步长：0.001s

时间推进步数：1000步

|  |  |  |  |
| --- | --- | --- | --- |
| 维数/方法 | Matlab的fsolve函数 | Matlab牛顿迭代法 | Fortran牛顿迭代法 |
| 二维模型 | 1609s | 189s | 139s或9s |
| 三维模型 | 太长，难计 | 1857s | 1524s或60s |

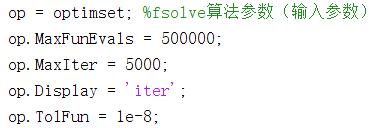
从理论上讲适当调整newmark方法的参数beta和gamma也可以提高计算效率，但如果参数选取不当可能会导致计算发散。

注意使用fortran时release模式会比debug模式快很多，以上Fortran牛顿迭代法左边的运行时间是debug模式，右边是release模式。

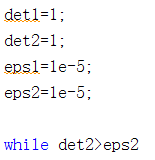
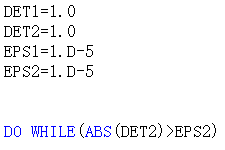
**二、计算精度对比**

三种方法得到的结果都是正确的，计算结果与真解之间的精度差只在于求解参数的选取。

在使用fsolve函数求解时求解精度取决于 op.TolFun 的设定，如图：



在matlab和Fortran中使用牛顿迭代法求解时求解精度取决于牛顿迭代法的收敛判据，如图：

下面将计算结果进行对比：

用三种方法在相同的条件下求解二维问题，得到第1000个时间步的 q\_sys 如下：

|  |  |  |  |
| --- | --- | --- | --- |
| q\_sys坐标 | fsolve | matlab牛顿 | fortran牛顿 |
| 1 | 9.08E-14 | 7.09E-29 | -1.00E-29 |
| 2 | -3.23E-13 | 2.49E-27 | 6.38E-30 |
| 3 | -0.988426 | -0.988426 | -0.988426 |
| 4 | -0.151738 | -0.151738 | -0.151738 |
| 5 | 0.1517349 | 0.1517349 | 0.1517348 |
| 6 | -0.98842 | -0.98842 | -0.98842 |
| 7 | -0.247108 | -0.247108 | -0.247108 |
| 8 | -0.037924 | -0.037924 | -0.037924 |
| 9 | -0.988439 | -0.988439 | -0.988439 |
| 10 | -0.151656 | -0.151656 | -0.151656 |
| 11 | 0.1516562 | 0.1516562 | 0.1516562 |
| 12 | -0.988432 | -0.988432 | -0.988432 |
| 13 | -0.494216 | -0.494216 | -0.494216 |
| 14 | -0.075849 | -0.075849 | -0.075849 |
| 15 | -0.988425 | -0.988425 | -0.988425 |
| 16 | -0.151744 | -0.151744 | -0.151744 |
| 17 | 0.1517406 | 0.1517406 | 0.1517405 |
| 18 | -0.988419 | -0.988419 | -0.988419 |
| 19 | -0.494216 | -0.494216 | -0.494216 |
| 20 | -0.075849 | -0.075849 | -0.075849 |
| 21 | -0.773083 | -0.773083 | -0.773083 |
| 22 | -0.634312 | -0.634312 | -0.634312 |
| 23 | 0.6343115 | 0.6343115 | 0.6343114 |
| 24 | -0.773076 | -0.773076 | -0.773076 |
| 25 | -0.687455 | -0.687455 | -0.687455 |
| 26 | -0.234466 | -0.234466 | -0.234466 |
| 27 | -0.772827 | -0.772827 | -0.772827 |
| 28 | -0.634626 | -0.634626 | -0.634625 |
| 29 | 0.634621 | 0.634621 | 0.6346209 |
| 30 | -0.772822 | -0.772822 | -0.772822 |
| 31 | -0.880628 | -0.880628 | -0.880628 |
| 32 | -0.393163 | -0.393163 | -0.393163 |
| 33 | -0.772561 | -0.772561 | -0.772561 |
| 34 | -0.634949 | -0.634949 | -0.634949 |
| 35 | 0.634944 | 0.634944 | 0.6349439 |
| 36 | -0.772556 | -0.772556 | -0.772556 |
| 37 | -1.073768 | -1.073768 | -1.073768 |
| 38 | -0.5519 | -0.5519 | -0.5519 |
| 39 | -0.77256 | -0.77256 | -0.772561 |
| 40 | -0.634949 | -0.634949 | -0.634949 |
| 41 | 0.6349437 | 0.6349437 | 0.6349435 |
| 42 | -0.772557 | -0.772557 | -0.772557 |
| 43 | -1.266942 | -1.266942 | -1.266942 |
| 44 | -0.710596 | -0.710596 | -0.710596 |
| 45 | -0.772832 | -0.772832 | -0.772832 |
| 46 | -0.634617 | -0.634617 | -0.634617 |
| 47 | 0.634614 | 0.634614 | 0.6346139 |
| 48 | -0.772828 | -0.772828 | -0.772828 |
| 49 | -1.460184 | -1.460184 | -1.460184 |
| 50 | -0.869209 | -0.869209 | -0.869208 |
| 51 | -0.773104 | -0.773104 | -0.773104 |
| 52 | -0.634282 | -0.634282 | -0.634282 |
| 53 | 0.6342819 | 0.6342819 | 0.6342818 |
| 54 | -0.773101 | -0.773101 | -0.773101 |
| 55 | -1.653467 | -1.653467 | -1.653467 |
| 56 | -1.027771 | -1.027771 | -1.02777 |
| 57 | -0.773157 | -0.773157 | -0.773157 |
| 58 | -0.634214 | -0.634214 | -0.634214 |
| 59 | 0.6342143 | 0.6342143 | 0.6342142 |
| 60 | -0.773157 | -0.773157 | -0.773157 |