测试不同规模下枚举法和利用yalmip软件包计算的结果，其中枚举法采用的是公式



Yalmip软件包使用的是clpex求解器，使用的是公式：



数据结果如下：

首先是4x32规模的情况，测试10次，每次迭代100下，得到数据如下，单位为秒：：

|  |  |  |
| --- | --- | --- |
| 次数 | 枚举 | yalmip |
| 1 | 0.371 | 33 |
| 2 | 0.36 | 33 |
| 3 | 0.371 | 33 |
| 4 | 0.393 | 33 |
| 5 | 0.369 | 33 |
| 6 | 0.4 | 33 |
| 7 | 0.389 | 33 |
| 8 | 0.402 | 33 |
| 9 | 0.391 | 33 |
| 10 | 0.375 | 33 |
| 平均值 | 0.3821 | 33 |

可以发现在，4x32规模下，枚举法的速度明显优于yalmip。

接下来是8x32规模的MIMO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 次数 | yalmip(不插电) | yalmip(插电) | 枚举(插电) | 枚举(不插电) |
| 1 | 61 | 36 | 50 | 54 |
| 2 | 58 | 36 | 48 | 53 |
| 3 | 55 | 36 | 48 | 55 |
| 4 | 82 | 35 | 48 | 54 |
| 5 | 77 | 36 | 51 | 54 |
| 6 | 65 | 36 | 49 | 53 |
| 7 | 79 | 36 | 50 | 52 |
| 8 | 93 | 37 | 50 | 53 |
| 9 | 61 | 36 | 49 | 56 |
| 10 | 53 | 36 | 54 | 52 |
| 平均值 | 68.4 | 36 | 49.7 | 53.6 |

可以发现电脑的性能会影响求解速度，而且在8x32规模情况下，yalmip算法的求解速度快于枚举法的速度。

接下来是8x128规模：

|  |  |  |
| --- | --- | --- |
| 次数 | yalmip | 枚举 |
| 1 | 49 | 95 |
| 2 | 52 | 87 |
| 3 | 50 | 85 |
| 4 | 48 | 93 |
| 5 | 48 | 87 |
| 6 | 49 | 86 |
| 7 | 50 | 94 |
| 8 | 48 | 91 |
| 9 | 50 | 88 |
| 10 | 78 | 89 |
| 平均值 | 52.2 | 89.5 |

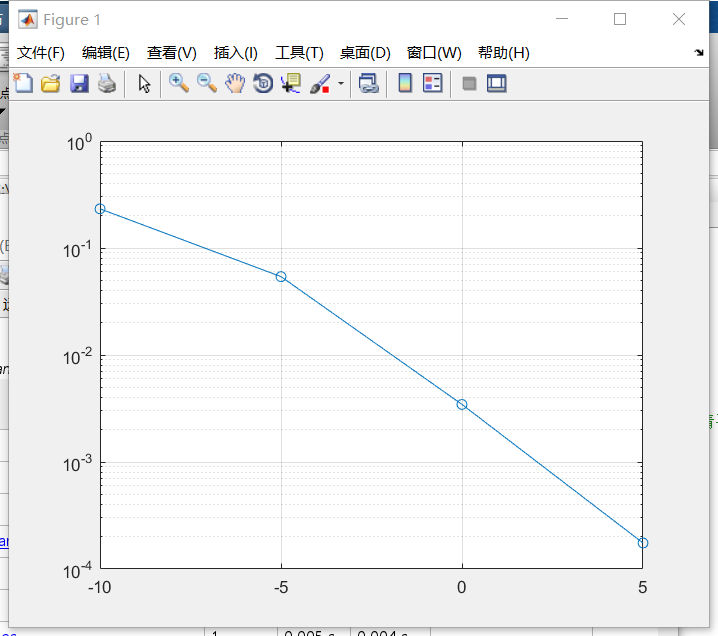
可以发现在这种规模下，yalmip明显优于枚举法。

接下来是16x128规模，此时matlab无法创建2^32大小的矩阵，已经无法使用枚举法，所以只有yalmip软件包计算的数据。

|  |  |
| --- | --- |
| 次数 | yalmip |
| 1 | 97 |
| 2 | 98 |
| 3 | 106 |
| 4 | 105 |
| 5 | 100 |
| 6 | 96 |
| 7 | 100 |
| 8 | 96 |
| 9 | 93 |
| 10 | 94 |
| 平均值 | 98.5 |

最后是4x32规模迭代10000次之后的SER曲线图：

首先是枚举法：



下面是Yalmip软件包求解结果(耗时46分钟)：

