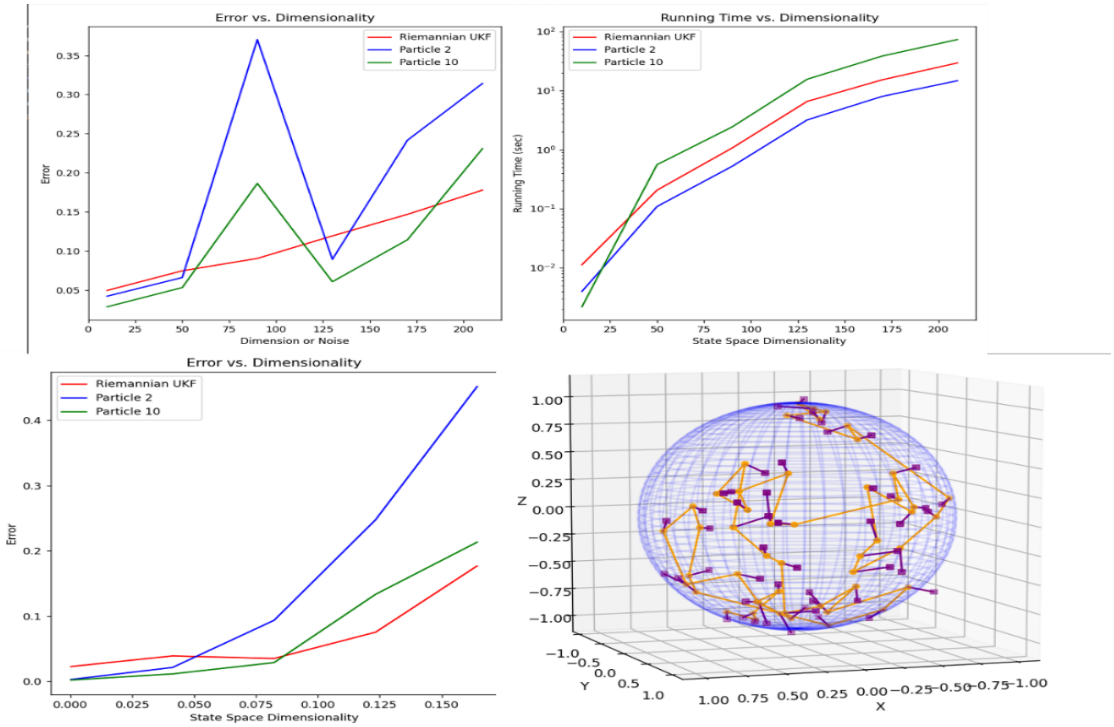
结果展示：



结果分析：上方两图分别是误差和时间随维度的变化，下图左为误差随噪声的变化，右图为仿真数据展示（有一定拉伸）。紫色点是观测，橙色点是真值，上方两图参数分别为 0.1 和 0.5/sqrt(M)（我严重怀疑原文中 0.2 的合理性，观测误差竟小于状态误差，导致高维时误差有些压不住，且此时效果一般比不过直接取 observation），下方两图与论文中一致。我们的代码完成了论文的基本框架，将原方法四步拆成了三个函数按批次实现，具体可见代码。结果与文章中的结果在大体趋势上是一致的，但是细节有区别，可能原因如下：

1. **参数介绍并不全面**。原论文给出的信息实际上不足以完整还原实验，比如初始值和方差的设定，测试时的采样方法，时间 T 等等，都没有给定。同时，本人硬件资源有限，因此为了节省时间，没有像原论文那样进行那么密集的测试，只进行了小成本测试来展示算法有效性。
2. **论文介绍有模糊性。N 维球有着 Log 和 Exp 函数的解析形式**，不需要像附录中说的那样用数值方法计算，而论文实验部分未介绍他使用的 Log 和 Exp。同时，论文也没有详细说明他使用了何种粒子滤波方法，我部署的粒子滤波在 2M+1 情况下耗费时间并没有论文里显示的那么耗时。**。**
3. **编程语言和环境区别**。原论文未给出仿真实验代码，因此不清楚其使用了什么语言。python 的数值计算的精度和速度可能都与极致优化的 matlab 有较大差距。

综上所述，由于种种客观条件限制，我未能完全 1:1 的复刻论文中的结果，但我相信上述结果和分析，以及配套代码已经可以证明我对该论文有了很深刻的理解。

代码说明：代码使用 python 编写，包含绘制上述图的所有函数（main 运行不会一次生成所有图，如画球函数在该版本中仅定义了但没有调用，因为太占时间了后续就删掉了其调用）依赖库参考 requirements.txt，如果运行失败大概率是环境的问题，如有问题欢迎联系作者。注意 python 的代码运行结果受随机性和电脑环境影响，每次运行会略有不同，本人使用了 windows11, python3.9 进行测试。