# x—>U(x) by DeepOnet

利用1D正激波后STS计算数据，训练DeepOnet：x—>U(x)。

工质O2/O，来流条件：V=4440m/s，P=106.658Pa，T=295K。

目的：输入网格坐标，输出U(x)，建立x到U(x)的映射关系。

## 1.生成数据

1. 利用Shock\_1D\_STS.m计算出波后0-5mm的流场参数，dx=5e-3mm，一共1001个点。
2. 确定训练集、验证集和测试集数据比例为7:2:1，因此训练集大小为700，验证集大小为200，测试集大小为101。

Note：没必要把所有数据都用上，证明一部分数据也能学习的很好。

1. 将x、U数据拼在一起，并将[训练的数据进行打乱](https://blog.csdn.net/weixin_44633882/article/details/105307076)，然后依次根据划分的比例生成训练集、验证集和测试集数据。

## 2.处理数据格式

1. 数据要求是二维的（num,1），但是读取出来的数据都是（num,）一维的，因此需要转换。
2. 采用deepxde中数据集TripleCartesianProd，函数要求input中元组第一个元素为（N1,d1），第二个元素为（N2,d2），输出为（N1,N2）。因此需要数据进行改造，N1=N\_train，d1=node，N2=node（相当于传感器的位置），d2=1（传感器是一维分布的），预测的既是在传感器位置上另一个函数的值！

Note:本例中传感器就一个。

1. 将U用波后的值无量纲化训练的loss看起来小一点。

## 3.设置网络

设置DeepONetCartesianProd网络，里面有两个FNN（数字代表宽度）：

第一个网络是branch net：【m=1（只有一个传感器），width=40，w=40】

第二个网络是trunk net：【dimx=1（一维问题），w=40，w=40】

深度可以不同，但最后一层宽度一定要相同，因为要笛卡尔积。

激活函数采用“relu”，初始化“Glorot normal”。

## 4.设置模型

设置优化器“adam”，学习率“0.001”，迭代次数20000次。

## 5.保存数据

三元组没法绘制最好的预测结果，因为x是元组，并不是数组，正常来讲函数与函数之间确实不能画xy图。