利用全连接网络求解偏微分方程

待解方程

$$\begin{cases} u_t(x,t) = u_{xx}(x,t) - 7 * sin(2\pi x) * u(x,t), \\ u(x,t) = u(x+1,t), \\ u(x,0) = 1, \end{cases} \qquad x,t \in [0,1]. \qquad (1)$$

定义价值函数

定义 $f(x,t)$, $g(x,t)$, $h(x)$分别为:

$$\begin{cases} f := u_t(x,t) - u_{xx}(x,t) + c * sin(2\pi x) * u(x,t), \\ g := u(x,t) - u(x+1,t), \\ h := u(x,0) - 1, \end{cases} \qquad x,t \in [0,1]. \qquad (2)$$

(这里 $c = 7$)

那么:

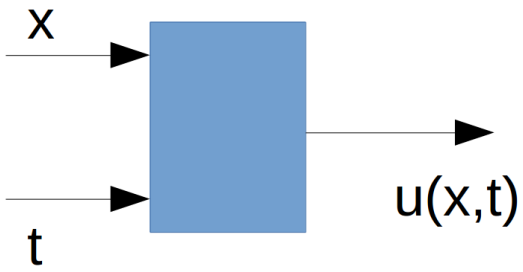
$$\begin{cases} loss_f = \dfrac{1}{N_f} \displaystyle\sum_{i=1}^{N_f} |f(x_f^i, t_f^i)|^2, \\ loss_g = \dfrac{1}{N_g} \displaystyle\sum_{i=1}^{N_g} |g(x_g^i, t_g^i)|^2, \\ loss_h = \dfrac{1}{N_h} \displaystyle\sum_{i=1}^{N_h} |h(x_h^i)|^2. \end{cases} \qquad (3)$$

最后得出最终的价值函数:

$loss_{total} = \alpha * loss_f + \beta_1 * loss_g + \beta_2 * loss_h$

(这里 $\alpha = 1, \beta_1 = \beta_2 = 50$)

网络结构



中间层的层数，每层的网络节点个数可以通过下面提到的参数调整方法得到。

参数调整方法

在实际操作过程中我们发现，求解方程的精确度受到神经网络各种参数的影响。因此，如何科学地确定网络的参数就变得尤其重要。

首先，我们要大致判断出影响网络最重要的几个参数。对于我们研究的问题，以下这几个参数对结果的影响是很大的:

- 网络的层数 α
- 每层网络节点个数 β
- 训练的样本个数 γ_m, γ_{bc} （这里 m, bc 分别代表主方程，边界条件）

其次，执行下面的操作:

1. 取$[\alpha_1, \alpha_2] = [2, 100]$, $[\beta_1, \beta_2] = [2, 100]$, $[\gamma_1, \gamma_2] = [20, 2000]$。
2. 每次**随机取** α, β, γ分别为 $\alpha \in [\alpha_1, \alpha_2]$, $\beta \in [\beta_1, \beta_2]$, $\gamma_m, \gamma_{bc} \in [\gamma_1, \gamma_2]$的任意整数，训练得到一个独立的网络结果，保存网络训练结果。
3. 重复上一个步骤N次，得到N个不同的结果。
4. 对比上一个步骤的结果和参考的解的差异，缩小 α, β, γ 随机取数的范围。
5. 重复2,3,4,直到α, β, γ的取数范围足够小，结束。

最后，我们便找到较优的 $\alpha, \beta, \gamma_m, \gamma_{bc}$ 组合:

$$\begin{cases} \alpha \in [5, 7] \\ \beta \in [20, 30] \\ \gamma_m \in [300, 1000] \\ \gamma_{bc} \in [100, 300] \end{cases} \quad (4)$$

结果对比

计算平台:

```
Distro: Linux Mint 18.3 Sylvia
Kernel: 4.13.0-41-generic x86_64 (64 bit)
CPU: Quad core Intel Core i7-7700HQ (-HT-MCP-)
GPU: GeForce GTX 1050 Ti
Python: 3.6.4 Anaconda, Inc.
Tensorflow: tensorflow-gpu '1.0.1'
```

训练步数: 60000

虚线: 参考值（由`Crank Nicolson`解法得出）

实线: 网络输出值

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [5, 21, 222, 50], Time: 5m36s

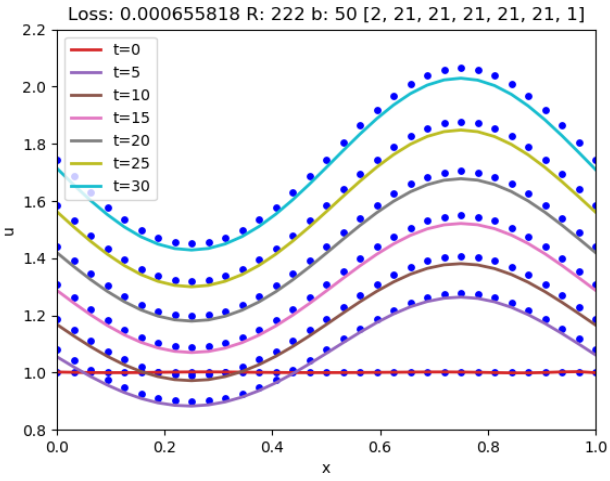


图 1

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [5, 21, 1424, 233], Time: 7m8s

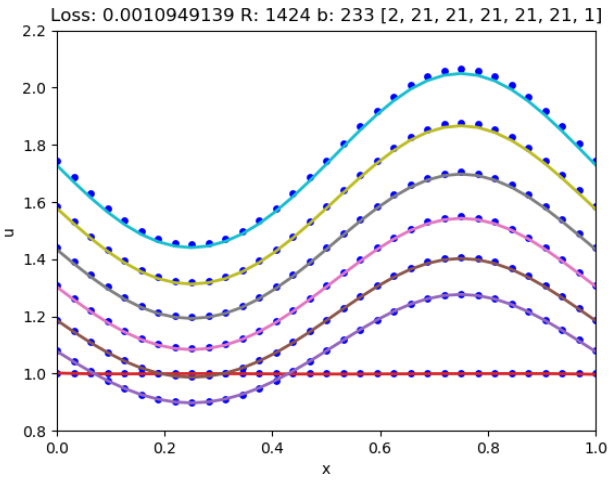


图 2

对比以上两图，在网络结构一致的情况下:

- 训练点的个数对结果影响较大。图1, 训练点数少，loss降到 10^{-4} 时,其结果仍差于图二具有较高loss(10^{-3})的训练结果。
- 训练点个数越多，训练同样步数消耗的时间更多。

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [6, 24, 134, 47], Time: 5m55s

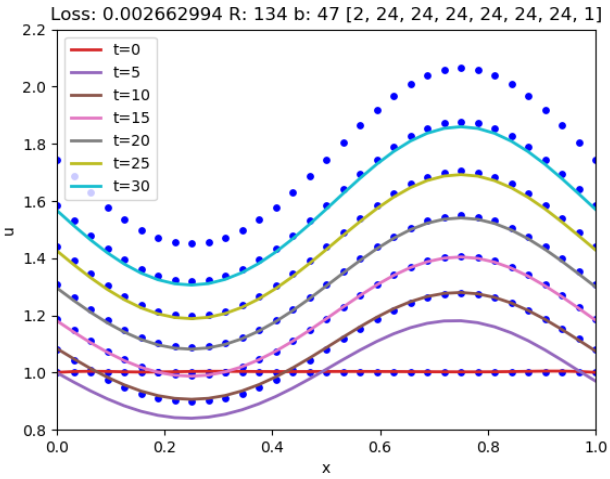


图 3

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [6, 24, 666, 87], Time: 6m25s

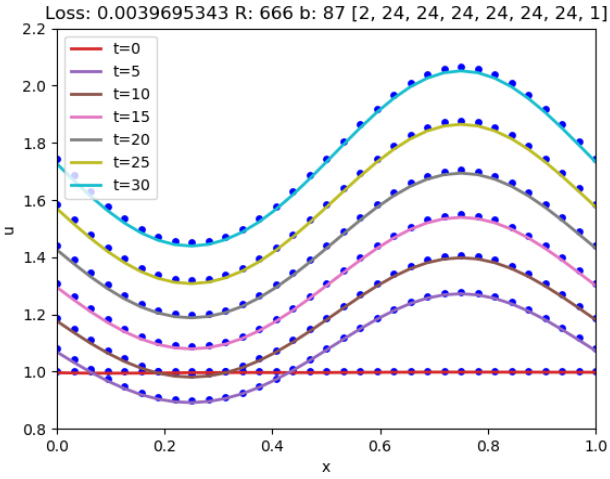


图 4

对比图3，图4， 网络结构一致的情况下，训练点数为$\gamma_m = 134, \gamma_{bc} = 47$ 时，已不能得出正确结果。因此：

- 训练点数不能太少

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [7, 27, 540, 243], Time: 6m51s

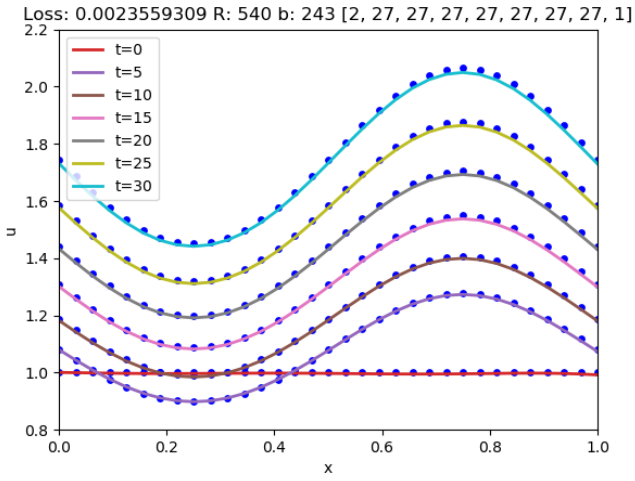


图 5

$[\alpha, \beta, \gamma_m, \gamma_{bc}]$ = [7, 27, 1154, 190], Time: 9m6s

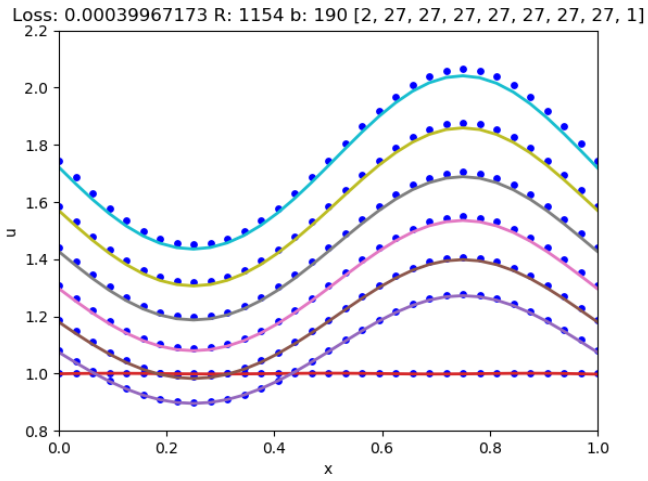


图 6

从图 5, 图 6中可以看出：

- 参数选择合适的情况下都可以取得较好的结果。

结论

1. 利用全连接网络能够很好地求解我们的偏微分方程，求解误差在可以接受的范围内。
2. 文中提到的参数调整方法对于确定神经网络的参数是很重要的。
3. 有限个(较少)的随机坐标训练点也可以较好地求解我们的方程。
4. 训练样本数越多，结果越精确，耗时越长。