**2.Vehicle Generations in Excel**

2020112921 刘欣豪

* POISSON

Define the mean = 3,

then the probability-distribution (Keep three decimals) is

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 0.050 | 0.199 | 0.423 | 0.647 | 0.815 | 0.916 | 0.966 | 0.988 | 0.996 | 0.999 | 1.000 |

I generate 1000 random numbers between 0 and 1, and choose the number what they belong to. Now results are as follows:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| num | 2 | 3 | 4 | 1 | 5 | 6 | 0 | 7 | 9 | 8 |
| frequency | 0.245 | 0.210 | 0.178 | 0.133 | 0.092 | 0.056 | 0.051 | 0.020 | 0.008 | 0.007 |

High fit

* NED

Define the μ = 0.3

then the probability-distribution (Keep three decimals) is

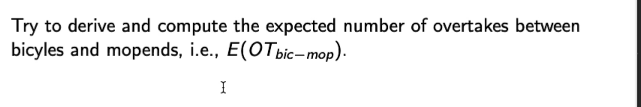
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 0.000 | 0.259 | 0.451 | 0.593 | 0.699 | 0.777 | 0.835 | 0.878 | 0.909 | 0.933 | 0.950 |

I generate 1000 random numbers between 0 and 1, and choose the number what they belong to. Now results are as follows:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| num | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| frequency | 0 | 256 | 204 | 132 | 96 | 79 | 65 | 45 | 23 | 23 |

3.number of overtakes between bicycles and mopends

2020112921 刘欣豪

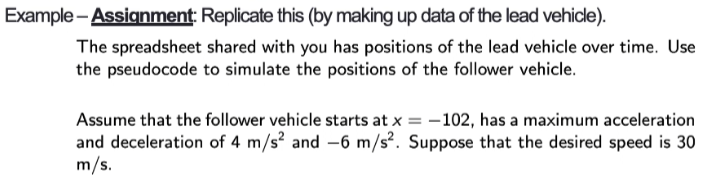


对于bicycle与mopeds，mopeds的速度为vi，bicycle的速度为vj



5.Pips-forbs model simulation

2020112921 刘欣豪



输出output表（部分）：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Time (second)** | **Position (meter) of lead car** | **speed of lead car** | **Position (meter) of follow car** | **speed of follow car** | **acceleration** |
| 0 | 2000 | 0 | -102 | 0 | 4 |
| 1 | 2000 | 0 | -98 | 4 | 4 |
| 2 | 2000 | 0 | -90 | 8 | 4 |
| 3 | 2000 | 0 | -78 | 12 | 4 |
| 4 | 2000 | 0 | -62 | 16 | 4 |
| 5 | 2000 | 0 | -42 | 20 | 4 |
| 6 | 2000 | 0 | -18 | 24 | 4 |
| 7 | 2000 | 0 | 10 | 28 | 4 |
| 8 | 2000 | 0 | 40 | 30 | 4 |
| 9 | 2000 | 0 | 70 | 30 | 4 |
| 10 | 2000 | 0 | 100 | 30 | 4 |

仿真结果：

出现撞车现象且在渐变区域不稳定。

代码：

1. **import** pandas as pd
3. datas = pd.read\_excel('Lead+car+data.xlsx')
5. **for** i **in** range(0,500):
6. s = datas.iloc[i,1]-datas.iloc[i,3]
7. v = datas.iloc[i,4]
8. smin = 6 \* (v/4.47 + 1)
9. #     print(smin)
10. **if** s<smin:
11. datas.iloc[i+1,4] = max(0,v-6)
12. datas.iloc[i,5] = -6
13. **else**:
14. datas.iloc[i+1,4] = min(30,v+4)
15. datas.iloc[i,5] = 4
16. datas.iloc[i+1,3] = datas.iloc[i,3]+datas.iloc[i+1,4]
18. datas.to\_excel('output.xlsx',index  =False)

6.Pips model 均衡分析（基本图关系推导）

2020112921 刘欣豪



已知：

即：

整理得：



7.Pips-forbs model simulation

2020112921 刘欣豪



输出output表（部分）：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Time (second)** | **Position (meter) of lead car** | **speed of lead car** | **Position (meter) of follow car** | **speed of follow car** | **acceleration** |
| 0 | 2000 | 0 | -102 | 0 | 0 |
| 1 | 2000 | 0 | -102 | 0 | 0 |
| 2 | 2000 | 0 | -102 | 0 | 0 |
| 3 | 2000 | 0 | -102 | 0 | 0 |
| 4 | 2000 | 0 | -102 | 0 | 0 |
| 5 | 2000 | 0 | -102 | 0 | 0 |
| 6 | 2000 | 0 | -102 | 0 | 0 |
| 7 | 2000 | 0 | -102 | 0 | 0 |
| 8 | 2000 | 0 | -102 | 0 | 0 |
| 9 | 2000 | 0 | -102 | 0 | 0 |
| 10 | 2000 | 0 | -102 | 0 | 0 |

仿真结果：

轨迹高度重叠，不符合现实

代码：

1. **import** pandas as pd
2. datas = pd.read\_excel('Lead+car+data.xlsx')
4. **for** i **in** range(0,500):
6. datas.iloc[i,5] = datas.iloc[i,2] - datas.iloc[i,4]
7. datas.iloc[i+1,4] = datas.iloc[i,4]+datas.iloc[i,5]
8. datas.iloc[i+1,3] = datas.iloc[i,3]+datas.iloc[i+1,4]
10. datas.to\_excel('output.xlsx',index  =False)

8.GM model simulation

2020112921 刘欣豪

Use the spreadsheet to simulate the GM model

,,,

starting x = 

the speed of follow car is 

输出output表（部分）：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Time (second)** | **Position (meter) of lead car** | **speed of lead car** | **Position (meter) of follow car** | **speed of follow car** | **acceleration** |
| 0 | 2000 | 0 | 467 | 30 | 0 |
| 1 | 2000 | 0 | 496.5303 | 29.53033 | -0.46967 |
| 2 | 2000 | 0 | 525.5967 | 29.06632 | -0.46401 |
| 3 | 2000 | 0 | 554.2046 | 28.60791 | -0.45841 |
| 4 | 2000 | 0 | 582.3596 | 28.15506 | -0.45285 |
| 5 | 2000 | 0 | 610.0673 | 27.70772 | -0.44734 |
| 6 | 2000 | 0 | 637.3332 | 27.26584 | -0.44187 |
| 7 | 2000 | 0 | 664.1626 | 26.82939 | -0.43645 |
| 8 | 2000 | 0 | 690.5609 | 26.39831 | -0.43108 |
| 9 | 2000 | 0 | 716.5334 | 25.97256 | -0.42575 |
| 10 | 2000 | 0 | 742.0855 | 25.55209 | -0.42047 |

仿真结果：

效果较好

代码：

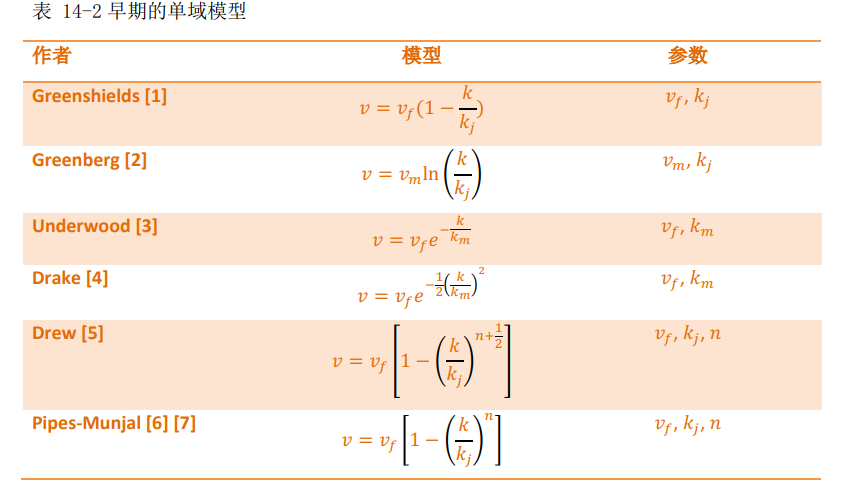
1. **import** pandas as pd
3. datas = pd.read\_excel('Lead+car+data.xlsx')
5. **for** i **in** range(0,500):
7. dv = datas.iloc[i,2]-datas.iloc[i,4]
8. dx = datas.iloc[i,1]-datas.iloc[i,3]
9. datas.iloc[i+1,5] = 0.8\*datas.iloc[i,4]\*dv/dx
10. #     datas.iloc[i+1,5] = 0.8\*dv/dx
12. datas.iloc[i+1,4] = max(0,datas.iloc[i,4]+datas.iloc[i+1,5])
13. datas.iloc[i+1,3] = datas.iloc[i,3]+datas.iloc[i+1,4]
15. datas.to\_excel('output.xlsx',index  =False)

9.读书笔记

2020112921 刘欣豪

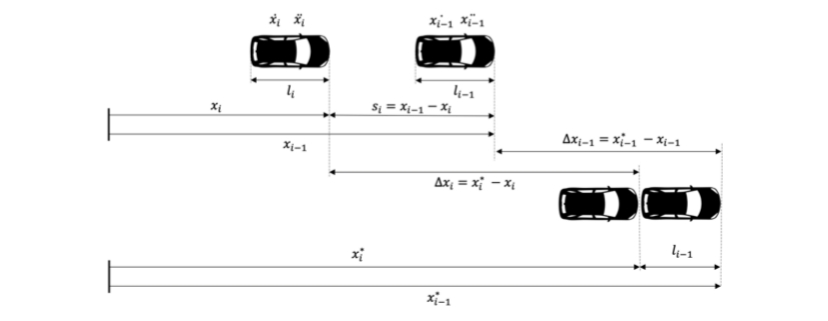
交通流反应刺激模型是一种用于描述人类驾驶员在面对不同交通情形时产生的行为反应的模型，能够为交通规划和设计领域提供了重要参考。该模型将驾驶员对交通刺激的反应划分为三个主要阶段，包括感知、决策和执行。以下是我对该模型的阅读笔记（主要为不同模型的参数标定）。

参数标定是一个重要的步骤，它的目的在于根据实际数据对模型参数进行确定，以使模型能够更好地描述实际场景中的驾驶员行为。不同的参数标定可能会对模型的输出结果产生显著影响。因为调整参数标定可以影响到模型的行为反应特征和预测准确性。一方面，如果模型参数被过度调整，可能会导致模型在描述驾驶员行为时失去真实性和准确性，从而影响到模型的预测结果。另一方面，如果模型参数被过度简化，那么模型也可能会无法准确描述驾驶员的行为。因此需要通过不断地调整和优化参数标定，最终找到最优模型参数以匹配不同的交通情况。



10.Gipps推导

2020112921 刘欣豪



有如下关系：



整理后可得



即：





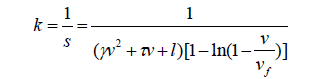
解得



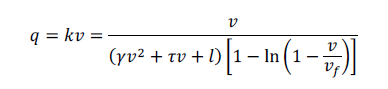
11.Find the capacity of LCM

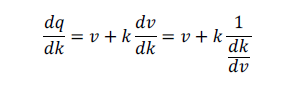
2020112921 刘欣豪

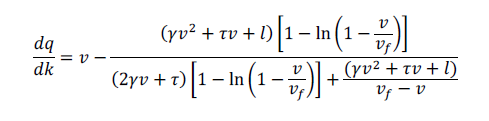
已知：



则：

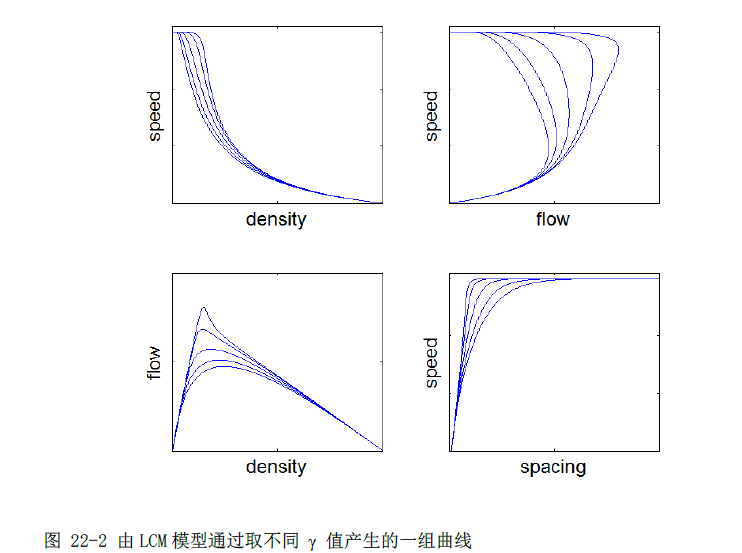






确定capacity时，可先令流量对密度的一阶导为0，求得最佳速度或最佳密度，最后回代球的流量的最大值。

根据课件内容：



12.CF=KW

2020112921 刘欣豪

CF：

当时：

已知



则；即：

KW：同理如上

对于另半边：

CF：





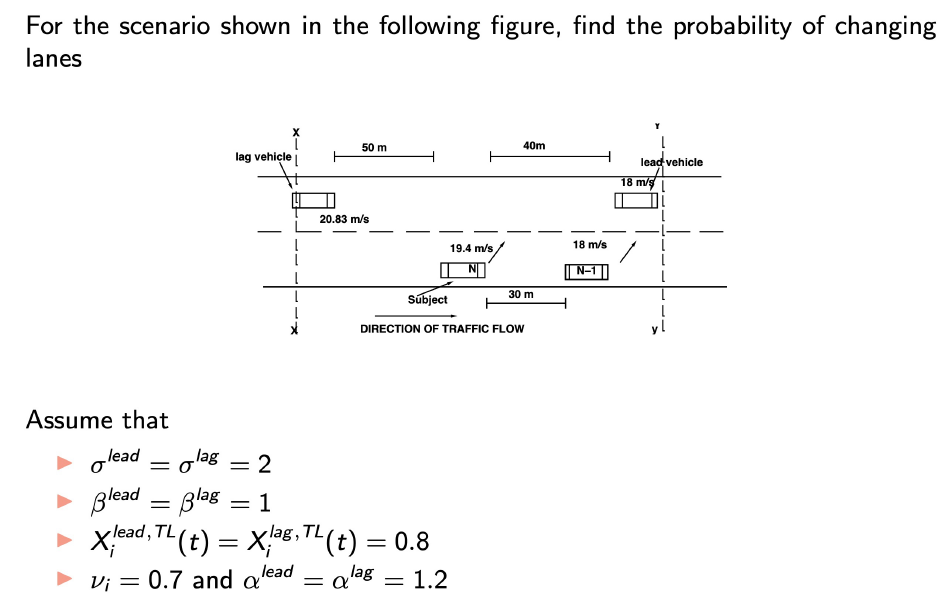
KW：



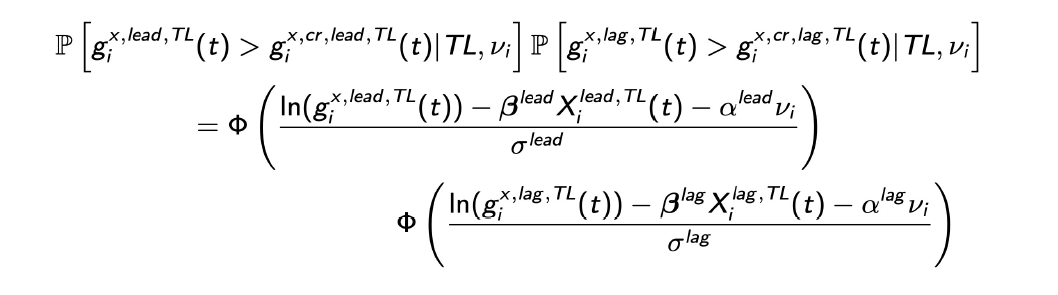


13.Changing lanes

2020112921 刘欣豪



代入：



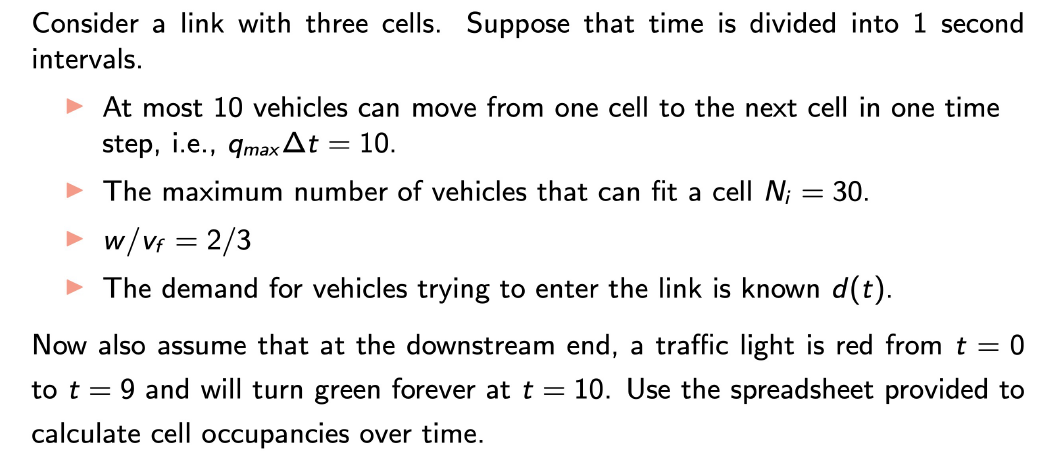




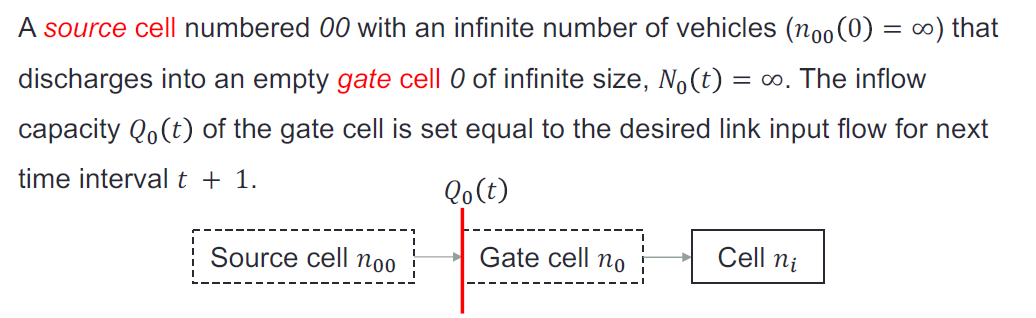
经查表0.8729\*0.8461=0.7386=73.86%

15.Cell Transmission Model

2020112921 刘欣豪



设置demand by a cell pair，



则如图1：



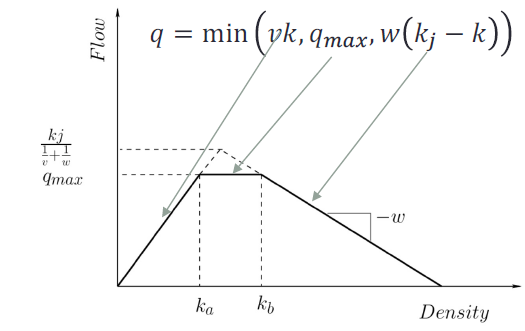


图1：0-22s仿真结果

附件1：

40s仿真完整表格



附件2：

代码

1. **import** pandas as pd
3. table = pd.read\_excel('demand.xlsx')
4. table.head()
6. Q = 10
7. N = 30
8. table['n00'] = 9999
9. table.iloc[0,2:7]=0
10. table.head()
12. **for** i **in** range(len(table)):
13. # 当0s时已初始化完毕
14. **if** i==0:
15. **continue**
17. **for** j **in** range(2,6,1):
18. q1 = min(10,table.iloc[i-1,j-1],2/3\*(N-table.iloc[i-1,j])) #q1为流入，q2为流出
19. **if** i<=9 **and** j==5: #此为红灯状态
20. q2 = 0
21. **elif** j==5: #红灯结束但为最后一个cell
22. q2 = min(10,table.iloc[i-1,j])
23. **else**: #红灯结束其它cell
24. q2 = min(10,table.iloc[i-1,j],2/3\*(N-table.iloc[i-1,j+1]) )
25. table.iloc[i,j] = table.iloc[i-1,j]+q1-q2
26. # 统计流出量
27. table.iloc[i,6] = table.iloc[i-1,6]+q2
28. table.to\_excel('output.xlsx',index=False)