自动驾驶汽车的调头轨迹设计

摘要

近年来，随着人工智能这一领域的大热，自动驾驶成了多方关注的前沿技术之一。为了探究自动驾驶在实际生活中的应用，我们将以现实中的无人自动驾驶车调头为模型，研究其如何通过控制车速、方向盘转速来实现自动调头。

针对问题一，需要确定控制点位置，并计算获得无人车能够顺利调头进入对向道路所需的调头轨迹。我们将后轮连线的中点确定为控制点，因为此点的速度方向、轨迹切线方向、车轴方向一致，方便后续的计算。为确定轨迹，我们设定方向盘转速，获得方向盘转过角度与时间的关系；通过传动比，建立方向盘转角与轮胎转角的关系；利用几何关系，确定曲率半径与轮胎转角的关系；以此可以获得无人车在空间转过的角度与时间的关系；最后，利用位移的分解与微积分知识可以得到轨迹上任意点的坐标与时间关系。通过此轨迹群，我们算出无人车外沿四个角的轨迹方程，不会压线且不会超出调车范围的轨迹方程所对应的轨迹群即为最终解集，对于此解集的获得，为达到题目所给定的“快速”这一要求，我们规定无人车控制点在直行 13.5m 后便开始转动方向盘；并且此轨迹由向左匀速转动方向盘、方向盘不动、向右匀速转动方向盘三部分合成得到。

针对问题二，需要我们确定在掉头过程中无需倒车与需要倒车的临界情况，我们的模型进一步给出了需要一次倒车与多次倒车的临界情况。是否需要倒车的临界情况即存在一种轨迹，能够使得第一题中外沿四角的轨迹能够不碰到墙壁，代入第一问方程，我们算得墙壁与车轴的初始位置最小为 12.9454m。一次倒车与多次倒车的临界情况即一次倒车使车身平行于道路时，能够不被阻拦地直行通过道路，再根据曲率不能突变、静止时可以转动方向盘这两个条件，算得墙壁与车轴的初始位置最小为 10.3364m。

针对问题三，需要我们考虑道路上存在除隔离障碍物外的静止障碍物时无人车的避障方案与调头轨迹。为绕过 G 障碍物，我们设定在无人车从初始位置直行 n 米后开始调头。通过计算得，当$14.27205m \leq n \leq 18.34707m$时，存在不会触碰到 G 障碍物与调头边界的轨迹方程。为绕过 F 障碍物，我们延长第一问题模型中的第二部分，即方向盘不动不动部分，使得外沿四角的轨迹能够不触碰到障碍物。

针对问题四，需要我们考虑人行横道的存在对轨迹的影响。由于隔离障碍物与人行横道仅相隔 2m，二车宽为两米，考虑车轮转向会加大车宽，因此，只有在无人车后轮越过人行横道后才能开始转动方向盘，即无人车向前行进 18m 后再开始转动方向盘，得到此时无人车的轨迹群。同时考虑人行横道以及 F、G 障碍物，只需选取问题三与问题四轨迹群的交集即为最终满足要求的轨迹群。

针对问题五，需要我们计算 F、G 障碍物在能够移动的情况下的轨迹。我们将任意方向盘转速以及任意车速情况下的轨迹方程利用运动学与微积分知识算出无人车到达轨迹上任意一点所需的时间，将此时间与障碍物到达此处所需的时间进行比较，得出此条轨迹能否安全通行这一条件，我们选取能够安全通过的道路作为我们通行的轨迹群。

针对问题六，需要我们在保证无人驾驶的安全性的前提下，优化求解成功率和求解耗时。通过观察问题五的轨迹群，我们发现，当 F、G 障碍物向 y 轴正方向运动时，无人车想要比障碍物先到达轨迹线的前提是 G 障碍物运动速度不大于 0.8m/s，F 障碍物运动速度不大于 0.4m/s，此情况不符合马路上现实情况，因此，无人车需要先等待 F、G 障碍物通行后再开始调头。若 F、G 障碍物向 y 轴负方向行进，由于 F、G 障碍物间隔较小，且此间隔在马路中间线上，不符合交通规则，因此我们设定无人车跟在障碍物后前行。

关键词：调头轨迹计算模型　轨迹筛选模型　动态路线规划

一、问题重述

1.1 问题背景

近年来，人工智能已经成为现代最热门的话题，而自动驾驶则是人工智能领域中受到广泛关注的前沿技术之一。为了探究自动驾驶在实际生活中的应用，我们以现实中的无人车调头为例。假设四轮乘用车是后轮驱动，车长、车宽和车的轴距为 5 米、2 米和 2.8 米，方向盘的最大转角为 470 度，转速为 400 度每秒，另外方向盘与前轮转角的传动比为 16：1。

无人车在行进路程中的轨迹是一条包含时间和空间位置坐标信息的曲线，其由一系列轨迹点构成，各个轨迹点中包含了位置坐标、方向角、曲率、速度等信息。而表征无人车轨迹需要一个特定点，这个特定点也就是控制点，通常对于无人车左右对称，在行驶过程中，会与轨迹点重合。另外在无人车行驶过程中，其应满足动力学性质和安全性的要求，曲率变化应该是连续的，其存在一个最大值，方向盘转角等参数不能发生突变，在调头过程中，车辆应该和障碍物之间有一个最小的安全距离。同时无人车在保证动力学的基础上，应该尽可能地具备舒适的驾驶体感、遵守交通法规，以具备高的通行效率。

题目要求我们简化无人车调头场景，利用附件中的数据，建立无人车安全快速调头的数学模型，设计无人车调头轨迹的计算方式。让无人车可以实时根据已有的外部道路和障碍物数据，计算合理的无人车轨迹。

1.2 问题提出

问题一：根据问题一的要求，我们需要通过题目附件所给的无人车初始位置，障碍物及三个对向车道的具体信息，建立无人车调头的数学模型，并阐述控制点，给出合理的算法设计，计算问题一应用场景中的无人车调头轨迹。

问题二：问题二给出了第二种情况的现实场景应用，建立数学模型，计算当调头区域比较狭窄时，无人车需要在什么参数的具体数据下，才能不倒车完成调头；在什么情况下，无人车必须进行至少一次倒车才能实现调头。

问题三：在问题三中，考虑了多种障碍物对于无人车调头轨迹的影响。在足够宽敞的调头区域中，存在了两个障碍物及它们位置的具体数据。需要我们给出合理的模型和算法，计算无人车的实际调头轨迹。

问题四：本问中，增加了人行道对于无人车调头的影响。人行道的具体信息已经给出，要求我们建立模型和算法，先单独计算无人车在不压人行道或后轮开过人行道后开始调头的情况下的调头轨迹。然后结合问题三多障碍物情形，计算在人行道和障碍物同时存在情况下的无人车调头轨迹。

问题五：问题五考虑了实际生活中的应用问题，提出道路障碍物的移动情况。针对两种场景，问题四中的多障碍物和人行道场景与问题三的多障碍物场景，要求我们建立具有普适性的无人车调头轨迹规划模型以及算法策略，使得障碍物以一定速度沿道路平行方向移动时，可以给出无人车调头方式与仿真结果。

问题六：为了提高无人驾驶的安全性能，要求我们改进算法，加快无人车对于路况的反应速度，降低计算的复杂度和求解路径耗时，提高无人车轨迹规划算法的求解成功率。

二、模型假设

1. 假设无人车的轮胎不会打滑、设备较为先进，即能够精准地按照给定的方案行进而不产生误差；
2. 假设无人车有完备的传感器系统，即无人车在发现障碍物时能够立即获取障碍物的位置、大小及运动状态的信息；
3. 假设无人车车身为严格的矩形，即不考虑反光镜等因素的影响。

三、符号说明

符号	符号意义
α	表示方向盘转角
t	表示方向盘转动时间
ω	表示方向盘转速
θ	表示方向盘的转角
L	表示无人车轴距
ρ	表示无人车行驶过程中的曲率半径
v	表示无人车的行进速度
γ	曲率半径与x轴的夹角
(x_1,y_1)	表示曲线轨迹中第一部分结束时的无人车控制点坐标
t_A	表示无人车以行驶速度v行驶到 A 点的时间
$t_A{}'$	表示障碍物G以速度v_G到达 A 的时间
τ	表示方向盘固定时间转过的圆心角角度
y_G	表示表示障碍物到 A 点的距离
v_G	表示障碍物G的移动速度
v_F	表示障碍物F的移动速度
$t_B{}'$	表示障碍物F以速度v_F到达 B 的时间
t_B	表示无人车以行驶速度v行驶到 B 点的时间
k	表示对应点的斜率

注：未列出符号及重复的符号以出现处为准

四、模型的建立与求解

4.1 问题一

4.1.1 问题一的分析

问题一要求我们根据附件中的无人车、障碍物、车道的数据信息，确定控制点位置，建立无人车调头的模型和算法，使无人车能够顺利调头，并绘制调头轨迹。我们首先明确模型中的变量为方向盘转速和无人车行驶速度。运用运动学和微积分的知识，通过计算车轮转角、无人车曲率半径等参数，求解无人车运行轨迹和方向盘转速、无人车行驶速度的函数关系，并求解无人车上所有点会扫过的整个区域。

我们定量化处理题目中所给的舒适度和安全性限制要求，利用算法和模型，计算行驶进入三个车道对应的两个变量的取值范围。运用定量化的限制条件，求解三个车道上对应无人车最快速度行驶入该车道的两个变量的取值，并绘制出每个车道最佳行驶方案的调头轨迹。

4.1.2 问题一的解答

对于问题一的解答，我们首先要清楚方向盘转速、车轮转角、无人车行驶过程中的曲率半径等基本概念，列出公式对这些变量进行分析求解，得到最终的车辆行驶轨迹与这些变量之间的关系，建立模型，结合题目中所给出的已知数据，进行编程，求解并绘制无人车的调头轨迹。我们建立坐标系，以无人车出发行进$13.5m$后的控制点为坐标原点，向左为x轴正方向，向上为y轴正方向。

4.1.2.1 理论准备

下面对建模中所需要用到的专有名词进行解释和分析：

1)方向盘转速

方向盘转速指的是，车辆在行驶过程中，为使车辆向左或向右行进，方向盘转动到相应角度情况下的转动线速度。在保证安全驾驶的前提下，方向盘转速与车辆转弯的角度、车辆行进的速度、车辆的轴距之间存在数学关系。

2)车轮转角

车轮转角表示的是方向盘在规定转速下转动到相应的角度后，车轮所对应转动的角度。根据车辆本身的固有属性，车轮转角与方向盘转角存在相应的传动比。因此，可以根据方向盘转角，求与之相对应车轮转角

3)无人车行驶过程中的曲率半径

曲率半径指的是平面曲线中对于曲线上某个点对应的某段曲线小圆弧的圆弧半径。曲率半径表示这段小圆弧的半径。在无人车行驶过程中，曲率半径是会发生变化的，车轮转角的变化是导致曲率半径变化的原因。因此，我们需要通过车轮转角来表示无人车行驶过程中的曲率半径。

4.1.2.2 模型的建立

我们根据题目要求和分析，需要先计算出车轮的转速和无人车行驶过程中的曲率半径，随后利用运动学中的相关公式计算出无人车运动轨迹曲线和时间的函数关系，并绘制出无人车的调头轨迹。为模型的解答奠定理论基础。

1)控制点的确定与说明

我们将无人车后轮的中心对称点作为该问题的控制点。根据控制点的要求，其应该具备以下作用：行驶过程中，控制点处的速度方向与轨迹点方向角一致；控制点可以简便地表征车上其他点的坐标位置。

当以无人车后轮的中心对称点作为控制点时，能够满足以上条件。并且对于任意时刻，该点车轴方向与轨迹速度方向保持一致，在计算无人车轨迹时，可以作为质点简化

模型。另外该控制点速度方向与车轴一致，当确定车轴，轨迹点位置信息时，可以确定出四轮和车的具体位置。

2)模型的分步建立过程

在整个无人车调头的过程中，我们在模型里面将其转化为三个步骤。第一步，方向盘从开始转动至达到最大角度，这一步无人车速率的变化对行车轨迹有影响；第二步，方向盘在最大角度下行驶，此时，无人车速率对行车轨迹无影响，车轮转角不变；第三步，在方向盘转动回到直行角度情况下，无人车进行转弯，当方向盘停止转动，无人车调头过程结束。

无人车运动轨迹求解过程中的步骤如下：

a)车轮转角的计算步骤

在车轮转角的计算步骤中，我们以方向盘转速ω为变量进行求解。

*Step1.*根据物理模型中的运动学公式：

$$\alpha = \omega t \tag{1}$$

其中α表示方向盘转角，方向盘转角我们取它所能达到的极限为它的极值，t表示方向盘转动时间。

我们通过公式求解方向盘的转动角度；

*Step2.*根据题目中的已给条件，即方向盘转角与车轮转角的传动比 16：1 计算车轮的转角，得到：

$$\theta = \frac{\alpha}{16} \tag{2}$$

其中θ表示方向盘的转角。

b)无人车行驶过程中的曲率半径计算步骤

通过物理学中的公式：

$$\frac{L}{\rho} = tan\theta \tag{3}$$

其中L表示无人车轴距，ρ表示无人车行驶过程中的曲率半径。

可以将曲率半径和车轮转角联系起来，最后化简成一个以方向盘转速ω为变量的数学关系式。

c)无人车运动轨迹曲线和时间的函数关系计算步骤

*Step1.*如图所示，我们先求解出任意时刻的点与曲率中心连线构成的曲率半径与x轴的夹角γ；

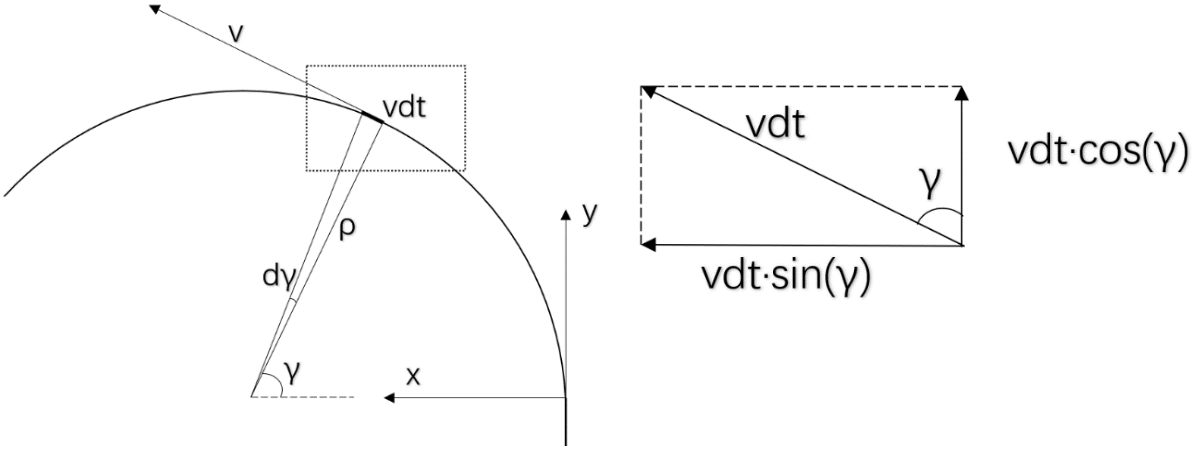


图 1 计算示意图

Step2.利用物理中微分思想，运用公式：

$$d\gamma = \frac{vdt}{\rho} \tag{4}$$

其中v表示无人车在弯道的行进速度，该参数也是一个变量，dt表示行进那一小段距离的时间。

我们可以通过该公式得到曲率半径与x轴的夹角γ和方向盘转速ω、无人车弯道的行进速度v，这两个变量之间的关系；

Step3.运用几何学和数学的知识，我们可以得到两个积分公式：

$$\begin{cases} dx = v \cdot sin\gamma \cdot dt \\ dy = v \cdot cos\gamma \cdot dt \end{cases} \tag{5}$$

可以通过对这两个公式进行积分，得到无人车坐标与时间的关系，进而可以求解出无人车运动轨迹曲线和时间的函数关系。

Step4.基于无人车运动轨迹曲线和时间的函数关系，可以绘制出无人车最终的调头轨迹。

在模型中我们分为三部分求解，但第一步和第三步原理相同。因此只需要讨论第一步和第二步两种情形。这两步的唯一不同之处就体现在方向盘转速ω，第一步和第三步存在该变量，而第二步方向盘角度不变，因此不存在该变量。所以只需要改变其中的一个参数值即可。

在第一步中，$\alpha = \omega t$；而在第二步中$\alpha = C$，其中C为一个常量，由方向盘转动最大角度决定。其余部分三步均相同。

4.1.2.3 模型的求解

根据题目给出的数据和我们建立的数学模型，可以分析得到一个以方向盘转速ω和无人车弯道的行进速度v为变量的无人车运动轨迹曲线。并结合题目的条件，设计一种合理可行的方案。

1)限制条件定量化

题目中对无人车行驶过程中具有限制性条件，我们对于这些限制条件进行定量化的处理。

安全性：根据交通法规文献资料，我们定义无人车弯道的行驶速度应该满足$v \leq 30km/h$，且无人车在行驶过程中不能压线；

舒适度：查阅文献资料，我们定量化的用向心加速度表示舒适度，其应满足

$$\frac{10m}{s^2} \leq \frac{v^2}{\rho} = a \leq \frac{20m}{s^2} \tag{6}$$

2)求解流程

我们求解出关于两个变量的所有可能路径，并根据定量化的限制性约束条件，对变量和结果进行约束限制，可以得到无人车在满足条件且遵守交规的情况下，行驶进入三个车道各自对应的两个变量的取值范围。

为满足题目所要求的无人车转弯最快速度。我们对行驶到三个车道进行分类讨论，寻找行驶到这三个车道上的无人车的最快速度所对应的两个变量方向盘转速ω和无人车弯道的行进速度v的具体数值。

以此数据为基础，绘制出无人车的调头轨迹。

4.1.2.4 模型的结果分析

我们通过模型和对应的定量化处理后的限制条件，计算得到最终的无人车调头轨迹坐标与方向盘转速ω和无人车弯道的行进速度v两个变量的关系：

$$\begin{cases} dx = v \cdot \sin\,[-\frac{v}{L} \cdot \frac{16}{\omega} \cdot \ln\,(cos\frac{\omega t}{16})]dt \\ dy = v \cdot \cos\,[-\frac{v}{L} \cdot \frac{16}{\omega} \cdot \ln\,(cos\frac{\omega t}{16})]dt \end{cases} \tag{7}$$

我们通过这个函数的积分式，绘制出一个以无人车控制点为原点，左方向为x轴正方向，纵向为y轴正方向的坐标系。描绘了以x轴正方向距离为因变量，v和ω为自变量的三维二元变量图。

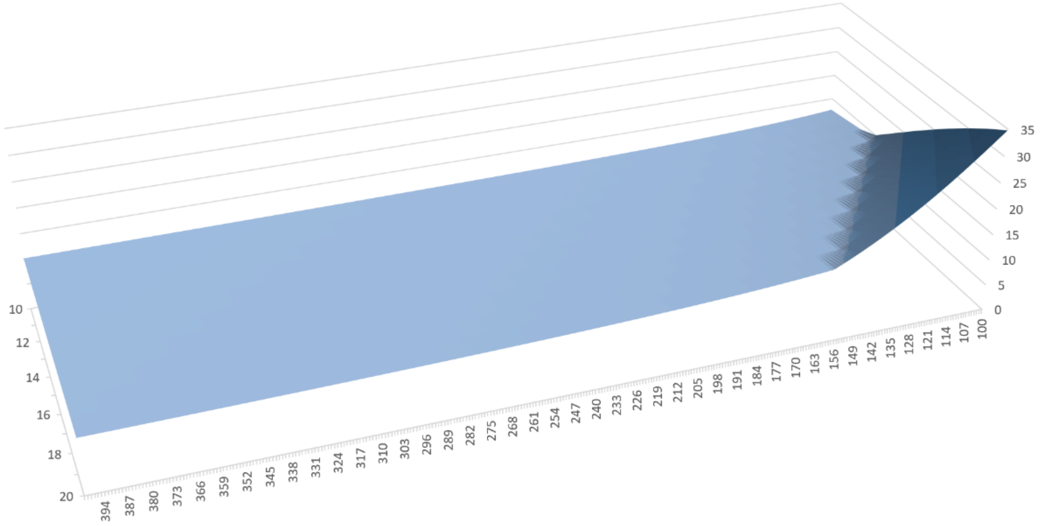


图 2 三维结果示意图

通过轨迹方程我们计算出无人车四个角实际扫过的区域后，结合附件中的信息，可以计算出无人车能够驶入第一车道的x轴取值范围在[4.5,5]，第二车道的x轴取值范围在[7.5,9]，第三车道的x轴取值范围在[11,12.5]，y轴的取值应满足$y \leq 13.1828$。经过求解验证，我们知道第一车道和第二车道在限制因素条件范围内，无人车是无法通过一次调头直接进入的。

我们以无人车行驶速度$v = 20km/h$为例，绘制出一个x轴和方向盘转速ω的函数关系图：

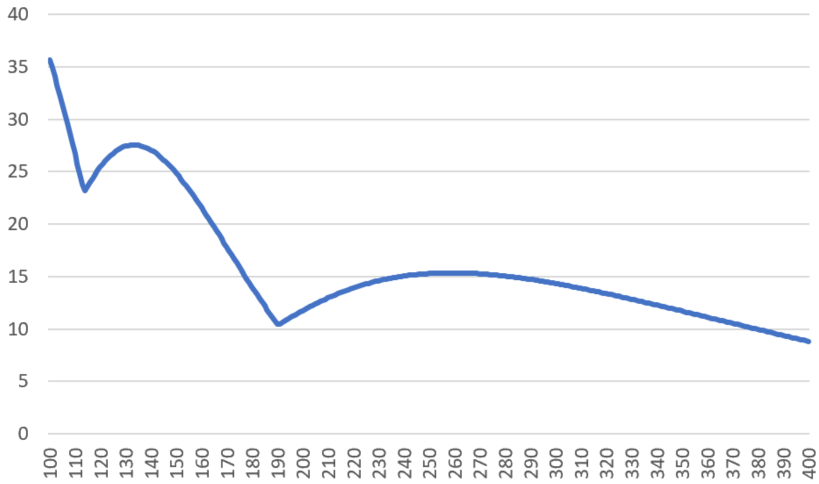


图 3 二维结果示意图

最终我们取方向盘转速$\omega = 400^{\circ}/s$，无人车行驶速度$v = 20km/h$，绘制出一个无人

车调头轨迹图像。

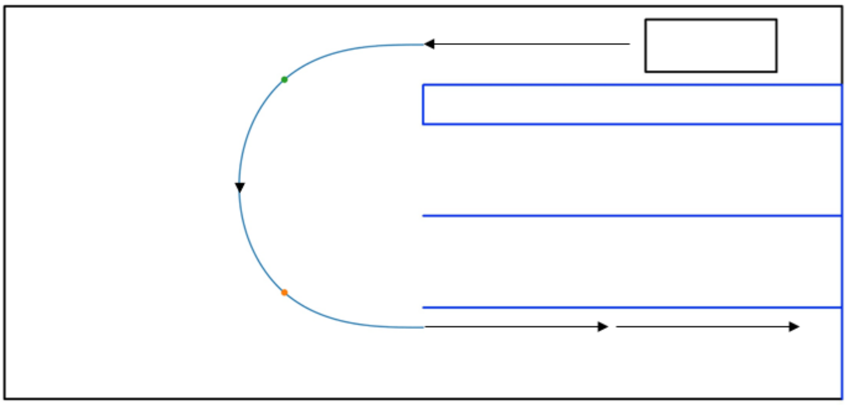


图 4 无人车调头轨迹示意图

图中表示了在该方向盘转速ω和无人车行驶速度v的情况下，无人车控制点行驶进入第三车道的控制点调头轨迹路线。根据验证可知，可以求出在曲线轨迹的第二部分，圆心点位为：

$$\left(x_1+\rho\cdot\sqrt{\frac{1}{k^2+1}},y_1-\rho\cdot\sqrt{\frac{k^2}{k^2+1}}\right) \tag{8}$$

其中$k=\cot\left[-\frac{v}{L}\cdot\frac{16}{\omega}\cdot\ln\left(\cos\left(\frac{\omega t}{16}\right)\right)\right]$，$(x_1,y_1)$为曲线轨迹中第一部分结束时的无人车控制点坐标。

该调头轨迹符合安全性规范，车轮不会压线，能够顺利完成调头。

4.2 问题二

4.2.1 问题二的分析

问题二提出调头区域过窄时的实际应用。需要我们建立模型判断在什么场景下，无人车可以不用倒车实现调头，什么情况下无人车至少需要一次倒车才能实现调头。因此我们基于问题一模型进行改进。根据问题一的结果，得到不倒车情况下的场景临界值数据。

对于在需要倒车的情形下，我们进一步提出一次倒车模型，旨在解决倒车时方向盘的转动角度，使得无人车一次倒车，便能顺利调头。结合实际生活和题目中的极限刹车加速度，我们首先可以确定曲线的行驶轨迹以及停止点的坐标。之后通过无人车能顺利通过车道的坐标取值范围，求解出满足条件的曲率半径范围，并借此曲率半径，利用问题一模型中的车轮转角公式，求解出方向盘的转角，实现一次倒车模型的功能。

4.2.2 问题二的解答

4.2.2.1 模型的补充与改进

我们对于问题一的模型进行临界倒车模型改进和一次倒车模型补充，得到问题二的模型，其余部分不变，关于坐标轴的选取，仍与问题一一样，以无人车出发行进$13.5m$后的控制点为坐标原点，向左为x轴正方向，向上为y轴正方向。

1)临界模型的改进

无人车在行驶过程中存在三种情况，第一种是调头区域足够，直接完成调头；第二种是，区域不够宽敞，需要进行一次倒车才能实现调头的作用；第三种是调头区域过窄，至少需要两次倒车的情况。第一种的临界情况，我们可以通过问题一模型结果中的三维二元变量图像直接观测到临界值。在这里我们统一用坐标进行表示。第二种与第三种情

况的临界值，我们需要通过补充的一次倒车模型进行计算。

2)一次倒车模型的补充

一次倒车模型其目的是计算需要一次倒车实现调头的坐标轴x区域范围。在无人车行驶过程中，为了尽可能地减小倒车次数实现调头，我们应该减慢无人车的行进速度，加快方向盘的转速，但是，题目也给了快速通过路口这一优化方向。因此，我们定义它的方向盘转速取最大值$\omega = 400^{\circ}/s$。无人车行进速度为$v = 10km/h$。基于此模型我们进行一次倒车调头的坐标轴x区域范围，坐标轴x为变量。

4.2.2.2 倒车模型的计算

Step1.我们通过问题一的结果，计算出不倒车实现调头和一次倒车实现调头的临界坐标轴x的取值；

Step2.运用模型中的先决条件，即$\omega = 400^{\circ}/s$和$v = 10km/h$确定该情况下，无人车在调头过程中的曲线轨迹；

Step3.在已知无人车曲线轨迹的情况下，可以求出无人车四个角中距离障碍物最近的一个角，当其距离障碍物仅剩余$1.5m$时，无人车在曲线轨迹上开始以 $2m/s^2$的加速度刹车。它的停止点是能够最大限度实现倒车的临界点，该点的坐标位置可以通过曲线的已知信息和变量x进行表示；

Step4.无人车是可以在原地停止时转动方向盘。考虑实际模型，在一次倒车过程中，当无人车的方向与车道平行时，调头完成。因此我们需要确定调头时的曲率半径，并结合该停止点的坐标，计算出最终完成调头时的无人车横坐标变量x，来判断是否能够正常进入车道；

Step5.根据问题一模型可以知道无人车的曲率半径与方向盘转动角度之间存在关联，因此我们可以反向推导，利用无人车能够实现一次倒车情况下的横坐标x的取值范围，反向求解曲率半径，利用问题一数学公式，推出方向盘转动角度。结合方向盘转动角度α自身的取值范围，求解出一个关于一次倒车实现调头的停止点坐标的取值；

Step6.由于该模型中，变量有且仅有一个x，因此，可以运用停止点的坐标取值范围和实现一次调头的限制条件，求解出一个一次倒车实现调头的取值范围。横坐标小于该取值范围的部分，则至少需要两次倒车才能驶入车道；大于该取值范围的部分，则不用进行倒车就可以直接驶入车道。

4.2.3 模型的结果与分析

根据题目要求，通过附件中的给出的障碍物数据信息和模型中假设的数据量及公式，计算出此结果条件下的不需要倒车实现调头，一次倒车实现调头，多次倒车实现调头的x坐标取值范围，以验证模型的正确性。

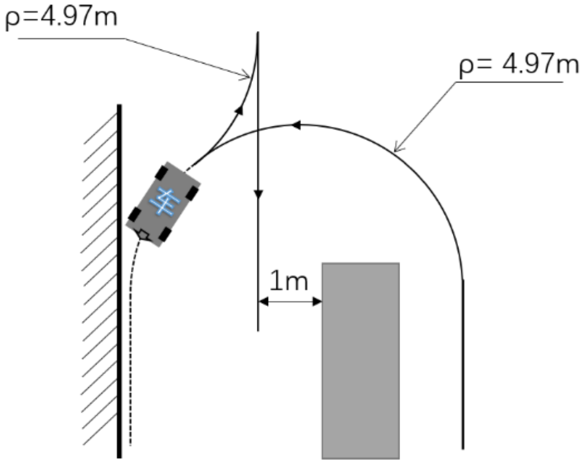


图 5 无人车调头轨迹示意图

经过求解计算可以知道：不需要倒车进行调头的取值范围为$[12.9454,+\infty]$，一次倒车实现调头的取值范围为[10.3364,12.9454]。

我们以一次倒车和多次倒车的临界取值$x = 12.9454\text{m}$ 为例，绘制出的仿真倒车调头图，如上图所示，以验证模型的合理与准确性。

4.3 问题三

4.3.1 问题三的分析

根据问题三要求，我们需要考虑道路上存在其它静止障碍物的情形，题目给出了三种情况下的障碍物，我们要建立合适的无人车调头模型，给出算法设计计算调头轨迹。我们将基于问题一模型进行改进，将模型分为曲线行驶和切线行驶两方面。

针对曲线行驶，我们要考虑影响曲线调头和切线行驶的障碍物位置信息，利用问题一模型求解无人车在曲线行驶过程中所扫过的区域，以此确定无人车曲线行驶的方向盘转速和无人车行驶速度。不同于问题一模型的是，在本题中，无人车曲线行驶的第三步会根据是否存在障碍物而改变，若存在障碍物，则曲线行驶中第三步的时间会根据切线行驶模型来确定，防止出现曲率突变及防止与障碍物相撞。否则将与第一题相同，直至完成调头，进入直道行驶。

针对切线行驶，无人车在启动之前，已经根据障碍物位置信息筛选了合适的行进速度和方向盘转速，因此在曲线行驶的末尾阶段，无人车会实时计算其在曲线上某点沿切线方向开出，无人车是否能够完成调头。当可以实现无人车调头后，无人车会沿着切线离开曲线轨迹，进入直道行驶，并绘制出其调头轨迹。

4.3.2 问题三的解答

4.3.2.1 模型的改进与转化

问题三提出了三种障碍物存在情况，因此我们通过对它给出的这三种情况进行分析。当仅存在G障碍物时，只需要改变曲线行驶的模型；当仅存在F障碍物时，只需要添加切线行驶的模型；当同时存在障碍物G和F时，需要同时运用曲线行驶模型和切线行驶模型。

1)曲线行驶模型

对于曲线行驶模型而言，我们是基于问题一的模型进行改进的。在问题一模型中，无人车是在原点处直接进行调头，而在本题中，需要考虑是否存在影响曲线行驶的障碍物G。若存在障碍物G，则曲线行驶模型需要进行判断，是否需要在原点处先向前行驶部分距离后，再开始调头，并且受到障碍物G的限制，无人车曲线调头过程中存在对方向盘转速和行驶速度的限制，因此在约束条件的限制下，选用合适的速度进行曲线行驶；若障碍物G不存在，则可以根据其它约束条件选用合适的速度调头。另外障碍物F的存在也会影响曲线行驶，因为在曲线行驶过程中，会存在部分速度使得无人车无法直接驶入直道，需要通过倒车来解决问题。因此考虑到无人车调头模型的快速性要求，我们根据障碍物信息，提前对方向盘转速和无人车曲线行驶的车速做好规划，以实现一次性完成调头的工作要求。

2)切线行驶模型

对于切线行驶模型而言，我们是基于障碍物F存在情况下进行补充的。为保证无人车调头的快速性条件，我们提出该模型，即在不停车、不倒车情况下，无人车实现越过障碍物完成一次性调头。切线行驶模型对曲线行驶模型具有一定的要求，当存在障碍物F时，即有切线行驶模型。曲线行驶模型过程中的第三步，会根据切线行驶模型中的切点来确定曲线行驶中第三步的开始时间。开始时间由无人车两个边框与障碍物之间的视线重叠距离确定，当它们之间的距离在满足小于等于无人车行驶速度与方向盘转动回正

的时间的比值后，则曲线行驶中的第三步开始，直至无人车方向回正后，无人车会沿着该点位切线行驶后，无人车整车所扫过的区域不会与障碍物发生碰撞。若不存在障碍物F，则不会切线行驶模型，无人车会按照在问题一改进基础上的曲线行驶模型行驶，最终沿直道完成调头。

无人车在行驶过程中，对于任意存在障碍物的情况存在多种能够实现调头的轨迹路线，模型是根据障碍物信息和当前无人车位置来进行避障规划以实现调头的作用，并绘制出调头轨迹。

4.3.2.2 算法求解过程

1)明确障碍物信息

确定障碍物的分布情况，以判断模型的选用。若仅存在障碍物G，则选用改进的曲线行驶模型即可；若仅存在障碍物F，则需要加入切线行驶模型；若两者同时存在，则应该同时改进两种模型。

2)计算方向盘转速ω和无人车曲线行驶速度v

Step1.根据障碍物的信息和相应的数据，制定约束条件。若存在障碍物G，则应当先判断无人车应该先直行路程n米后进行调头，对所有情况进行讨论；

Step2.之后再判断所有直行距离n情况下的调头过程中的两个速度变量，可以通过障碍物的位置信息来确定。问题一模型中的曲线轨迹是控制点的曲线轨迹，但本题考虑的是整个无人车的通过情况，因此需要根据控制点求出无人车两个边框所经过的轨迹，通过这两个无人车扫过区域的临界范围和障碍物的信息，确定两个速度变量的取值范围；

Step3.根据障碍物F的信息，判断是否需要进一步约束方向盘转速ω和无人车曲线行驶速度v的取值范围。寻找每一种直行路线情况下的每一种速度可能，若以此速度，不能够切线行驶入直道，则排除该答案可能性，最终求解出所有可能解。

3)实际行驶过程中的判断

Step1.无人车按照可能解中的方向盘转速ω和无人车曲线行驶速度v以及直行距离n，通过道路上是否存在静止障碍物F来判断是否需要加入切线行驶模型；

Step2.若不存在障碍物F，则无人车按照 *Step1* 中确定的三个可能解的变量，以问题一改进的曲线行驶模型，遵循模型中的三个步骤，最终以直行的方式进入直道，完成调头；

Step3.若存在障碍物F，则曲线行驶模型的第二步，方向盘转动角度固定的行驶时间应该适当增加，按照无人车与障碍物视线之间的重叠距离来判断第三步的开始时间；当它们之间的重叠距离满足小于等于无人车曲线行驶速度和方向盘转动回原角度的时间的比值，则无人车开始曲线行驶模型中的第三步，方向盘开始匀速转动，当方向盘转动回正后，无人车沿切线行驶，驶入直道，实现调头。

4)回正方向并绘制调头轨迹图

在无人车进入切线行驶入直道后，其需要根据自身所处位置进行判断。当无人车车身距离障碍物F的最近距离大于2米后，无人车可以根据车道的信息自动回正方向，从斜线转入直线，在车道内行驶。

我们可以根据实际的无人车曲线行驶与切线行驶或直线行驶的轨迹，绘制出无人车调头的实际轨迹图。

4.3.2.3 模型的结果与分析

根据题目要求，我们通过算法求解出了所有可能的方向盘转速ω和无人车曲线行驶速度v，在附件中呈现所有可能的数据。

但在结果中，无人车需要从原点出发，相对于坐标原点直行$[0.77205m, 4.84707m]$的区间范围后，再取用问题一的模型进行计算。我们一种特例解，$\omega = 400^{\circ}/s$和$v =$

$20km/h$为例，呈现出我们绘制的无人车调头轨迹，如下所示。

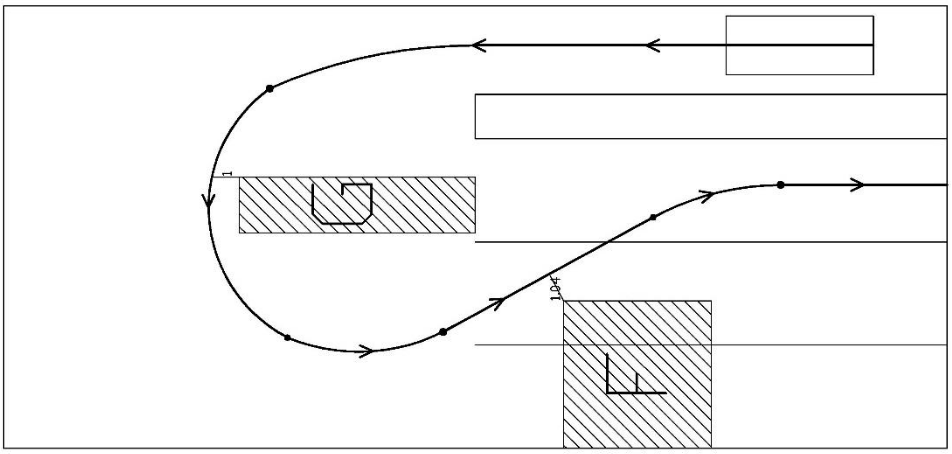


图 6 无人车调头轨迹示意图

4.4 问题四

4.4.1 问题四的分析

问题四考虑了现实场景中的应用，引入了道路人行横道模型，并在附件中给出了人行横道的数据信息。题目要求我们建立相关的无人车调头模型，且无人车在调头过程中需要尽可能不压人行横道，若无法做到，则要求无人车直行，在后轮越过人行横道后，再开始调头。问题最后还提出了如何通过算法和模型进一步处理当斑马线和问题三中障碍物同时存在的情况，如何做到既满足交规，又能避障通行，并绘制出无人车的调头轨迹。

我们对于仅存在人行横道的情况，基于问题一的模型进行分析。首先判断其能否不越过斑马线进行调头，如若不能则在问题一中的模型基础上，加入直行模型，即无人车向前行驶相应的距离，直到其后轮越过人行横道以后，再进行调头。其余部分的求解与问题一一致。

当问题中同时引入人行横道和障碍物后，我们可以在仅存人行横道的模型基础上进行求解，通过求解出人行横道模型的所有可能解，与问题三的多种障碍物的可能解，求交集。所有的交集即为同时满足交规并进行避障绕行的两个变量，方向盘转速和无人车曲线行驶速度。其内在的模型和算法与问题三一致。最终可以通过具体的解绘制出无人车调头轨迹。

4.4.2 问题四的解答

4.4.2.1 模型的改进与转化

在问题四中存在两种情况，因此我们需要引入两个改进模型，一个是人行横道模型，另一个是人行横道和障碍物模型，对问题进行分析和求解计算，绘制出无人车最终的调头轨迹。

1)人行横道模型

人行横道模型的作用是求解仅存在人行横道情况下，无人车实现调头的方向盘转速和曲线行驶速度两个变量。此模型是基于问题一模型进行改进的，它在问题一模型的基础上加入了直线行驶模型部分。问题一模型中，无人车是从起步开始，直接进入曲线行驶部分。因此我们需要通过人行横道的已知数据，结合问题一模型中的结果进行分析，判断无人车是否可以在满足不压人行横道的情况下，直接实现调头。如果无法实现，则需要通过人行横道的数据引入无人车直线行驶的距离。在这之后，依旧利用问题一模型

的算法，实现曲线轨迹的计算，求出所有能够实现无人车调头情况下的方向盘转速ω和无人车曲线行驶速度v两个变量解值。

2)人行横道和多障碍物模型

人行横道和多障碍物模型的作用是求解出在同时存在两种限制因素条件下，无人车实现调头的方向盘转速和曲线行驶速度。通过对题目给出障碍物位置和人行横道数据进行分析，我们知道这两种限制因素在本题中是直接进行物理叠加的。即在问题四中人行横道的基础上，直接添加多种障碍物，且这些障碍物的位置、尺寸数据信息与问题三中的信息是一致的。因此，我们考虑通过寻找问题三的结果数据和问题四中人行横道模型求解出的结果数据的交集，来得到该情况下，实现无人车调头的所有方向盘转速ω和无人车曲线行驶速度v解值。

无人车在行驶过程中，对于障碍物分布的情况存在多种能够实现调头的具体轨迹路线以及路线所对应的无人车方向盘转速和曲线行驶速度。在实际过程中，无人车会根据其现处于的阶段进行实时判断，来决定是否进入任意模型的任意阶段以实现成功调头的目标。最终根据无人车所走的路线绘制出无人车的调头轨迹。

4.4.2.2 算法求解过程

1)人行横道模型的计算

Step1.判断无人车从起步开始，能否直接进入曲线行驶。选用合适的无人车方向盘转速和曲线行驶速度，计算出控制点的最小圆弧曲线轨迹。通过车身计算出无人车在该轨迹下所扫过的区域面积；

Step2.利用题目附件给出的人行横道已知数据，和无人车扫过的区域面积进行对比。如果存在交集，则无人车无法在不压人行横道的情况下实现调头，需要让无人车在后轮越过人行横道的后开始调头；

Step3.运用人行横道的已知数据，计算无人车在后轮越过人行横道的情形下，需要先直行多少距离；

Step4.求解出直行距离后，利用问题一的模型直接求解人行横道模型中的曲线轨迹，结合题目中的人行横道和车道信息的约束范围，得到所有可能轨迹下对应的无人车方向盘转速和曲线行驶速度。无人车会在再次压到人行横道之前进入直线行驶，随后越过人行横道，进入车道直行。最终可以绘制出无人车在人行横道模型中的调头轨迹。

2)人行横道和多障碍物模型的计算

Step1.分别求解出人行横道模型和问题三多障碍物模型情况下的两种变量无人车方向盘转速和曲线行驶速度的取值范围；

Step2.该题中单一的多障碍物模型的原理、计算过程与分析与问题三中的模型一模一样，因此在这里不做过多表述；

Step3.计算两个模型单一情况下的解的交集，所有交集情况下的解，即为满足该模型无人车实现调头功能的两个变量，无人车方向盘转速和曲线行驶速度的解。

4.4.2.3 模型的结果与分析

根据题目要求，我们通过算法求解出了所有可能的方向盘转速ω和无人车曲线行驶速度v，在附件中呈现所有可能的数据。

但在结果中，无人车需要从原点出发，相对于坐标原点直行$[4.5m, 4.84707m]$的区间范围后，再结合问题一的模型计算。我们以一种特例解，$\omega = 400^{\circ}/s$和$v = 20km/h$为例，呈现我们绘制出无人车的调头轨迹，如下所示。

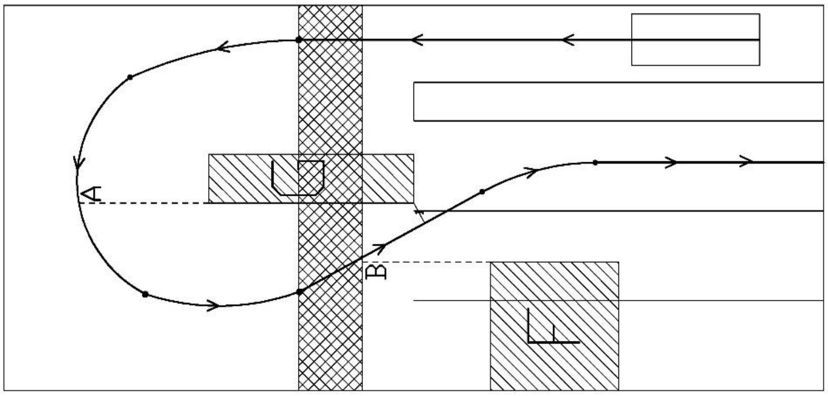


图 7 无人车调头轨迹示意图

4.5 问题五

4.5.1 问题五的分析

问题五是在问题四的基础上提出了更高的现实应用要求，原本问题四中的障碍物是静止的，但问题五中的障碍物是移动的，且它们以一定的速度沿道路平行方向移动。题目要求我们建立一个具有普适性、一般化的无人车调头轨迹规划模型，给出相应的算法设计和策略，并给出部分仿真结果验证结果的正确性。

我们考虑通过在问题四模型的基础上引入时间模型，对无人车的轨迹规划进行计算，如图 所示，无人车在应该在障碍物G之前抵达 A 点，应该在障碍物F之前抵达 B 点。因此我们可以通过这两个约束条件，量化地表示障碍物移动速度与无人车方向盘转速和行驶速度这两个变量之间的关系。通过这两个约束条件，结合问题四中的人行横道和多障碍物模型，我们可以在无人车调头有解的情况下，求解出任意障碍物移动速度对应能够实现无人车调头功能的无人车方向盘转速和行驶速度，并绘制出调头轨迹。

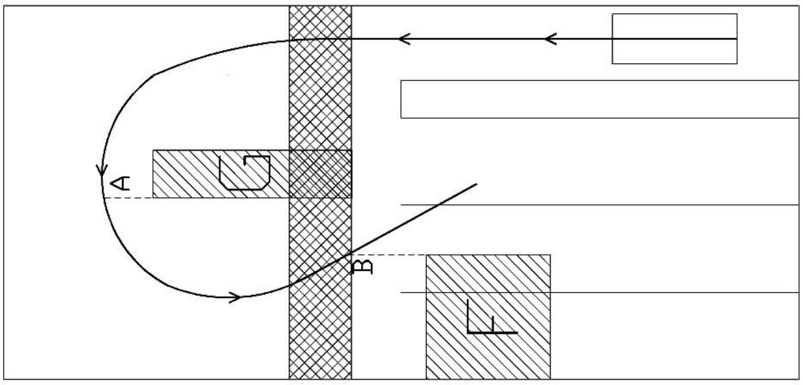


图 8 无人车调头关键判断点示意图

4.5.2 问题五的解答

4.5.2.1 模型的改进与转化

问题五模型是在问题四的人行横道和多障碍物模型基础上引入时间模型，定量化地约束了障碍物移动速度与无人车方向盘转速和无人车行驶速度之间的关系，在存在无人车调头解的情况下，求解出无人车的调头轨迹和轨迹所对应的方向盘转速与无人车行驶速度。

1)时间模型

我们通过时间模型来约束障碍物速度与模型两个变量之间的关系。我们分四种类型讨论障碍物G和障碍物F的移动方向。

如图 8 所示，当两个障碍物同时向右运动，则我们的方案将进入障碍物 F 与 G 间空间前行，它们之间的距离足以满足无人车通行，不会对无人车造成影响；当两个障碍物一个向右，一个向左运动时刻，它们对于无人车到达 A、B 点的时间具有约束作用；当两个障碍物同时向左移动时，两者同时对无人车抵达关键节点的时间具有约束作用。

我们以两者同时向左移动为例，讨论时间节点约束条件下障碍物移动速度与模型的两个变量之间的关系。单一障碍物的向左运动情况，只需要求解单一的约束条件即可。

无人车应该在障碍物G到达 A 之前抵达，我们假设无人车以方向盘转速ω和无人车行驶速度v行驶到 A 点的时间为t_A，障碍物G以速度v_G到达 A 的时间$t_A{}'$。

无人车抵达时间t_A分为三部分组成，即人行横道直线行驶时间，方向盘匀速转动时间和方向盘固定到达 A 点的时间，我们代入部分附件给出的已知数据，可以得到数学关系式：

$$t_A=\frac{20}{v}+\frac{470^{\circ}}{\omega}+\frac{\rho\tau}{v} \tag{9}$$

其中τ表示方向盘固定时间转过的圆心角角度，τ满足数学关系：

$$tan\tau=\cot\left(-\frac{v}{L}\cdot\frac{16}{\omega}\cdot C\right) \tag{10}$$

其中$C=-0.137622$，为一个常量。

障碍物G以速度v_G到达 A 的时间$t_A{}'$可以表示为数学关系：

$$t_A{}'=\frac{y_G}{v_G} \tag{11}$$

其中y_G表示障碍物到 A 点的距离，由于障碍物G自身存在高度，因此我们可以带入附件数据，利用几何关系建立数学方程：

$$\left[6.2-\left(x_1+\rho\cdot\sqrt{\frac{k^2}{k^2+1}}\right)\right]^2+\left[(y_G+8)-\left(y_1+\rho\cdot\sqrt{\frac{1}{k^2+1}}\right)\right]^2=\rho^2 \tag{12}$$

其中(x_1,y_1)为曲线行驶模型中第一步结束时的坐标。

当无人车需要实现调头功能时，应该满足：

$$t_A\le t_A{}' \tag{13}$$

我们可以通过这两者之间的时间约束条件，得到两者的关系，同理可以得到无人车抵达 B 点的时间t_B和障碍物F以速度v_F到达 B 的时间$t_B{}'$之间的关系。

最终我们可以通过时间模型中的约束条件与问题四模型的限制性因素，在无人车调头存在解的情况下，求解出任意障碍物移动速度下，对应能够实现无人车调头功能的无人车方向盘转速和行驶速度，并绘制出调头轨迹。

4.5.2.2 算法求解过程

Step1.通过题目给出的障碍物初始位置信息及两个障碍物的移动速度，求解出满足时间模型情况，即能在障碍物之前抵达关键节点 A、B 位置的所有无人车方向盘转速ω和行驶速度v解；

Step2.将所有的可能解带入问题四模型中人行横道判断的情况，利用问题四的人行横道模型进一步筛选出满足交规，不压人行横道进行调头的无人车两个变量参数的所有可能解；

Step3.之后运用问题三的多障碍物模型，对剩余的无人车解进行分析和筛选，遍历所有解的可能，寻找能够顺利转入切线行驶模型，并在最后阶段，由切线转换到直道上进行直行的无人车方向盘转速ω和行驶速度v解；

Step3.确定第三步中的所有无人车方向盘转速ω和行驶速度v解为最终解后，遍历所

有的解，求解并绘制所有情况下的无人车调头轨迹路线。

4.5.2.3 模型的结果

根据题目要求，我们通过算法求解出了所有可能的方向盘转速ω和无人车曲线行驶速度v，在附件中呈现所有可能的数据。

但在结果中，我们呈现一种特例解，即$\omega = 400^{\circ}/s$和$v = 20km/h$，障碍物速度在约束条件范围内，呈现出我们绘制的无人车调头轨迹，如下所示。

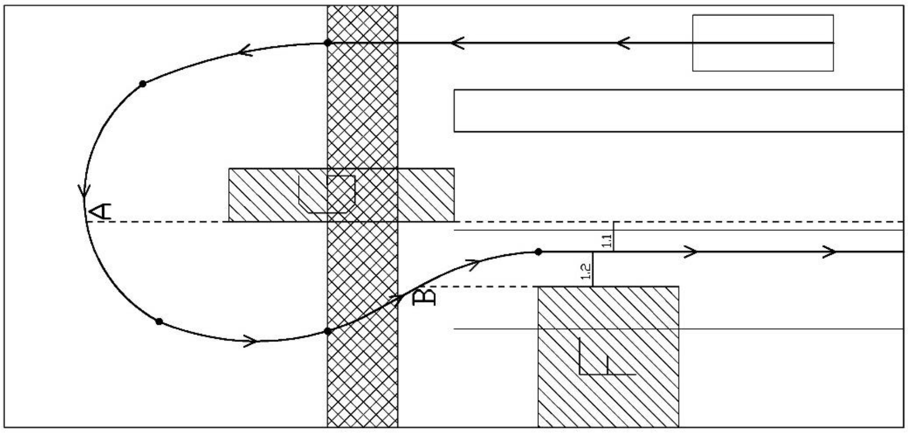


图 9 无人车调头轨迹示意图

4.6 问题六

4.6.1 问题六的分析

问题六提出了模型的优化问题，它指出无人车轨迹的规划算法应该具有较高的求解率，倾向于更收敛、稳定的方案，同时应该采用尽量高的频率根据路况进行实时计算，使其能够适应快速变化的交通环境。

我们充分考虑了无人车行驶的规范性、实际性与安全性问题。对无人车在两种障碍物移动情况下的方案提出了改进。无人车在行驶过程中会结合障碍物和交通路段的具体信息判断是否直接通行或者等待其越过无人车的轨迹后再行进。以降低算法的求解耗时，提高算法的成功率为核心目标。

4.6.2 方案的改进与优化

我们对于无人车算法方案在问题五的结果基础上进行改进。通过对问题五最终结果和无人车调头轨迹的分析，我们结合道路行驶的规范性与实际性，改变原本算法中两处步骤。

1)障碍物行驶速度

在问题五的模型中，当障碍物G和F向上移动时，根据我们原来的模型，无人车必须在障碍物抵达 A 点之前先过 A 点，在其抵达 B 点之前抵达 B 点。但是通过结果分析，我们知道在满足无人车先障碍物一步抵达 A、B 点的情况下，障碍物的速度存在限制，即障碍物的最快速度为$v_G = 0.8m/s$，$v_F = 0.4m/s$，。此速度不符合现实生活中在交通路段上发生的移动速度。因此为了改进算法，使之在求解成功率上提高，我们提出另一种情形，即在障碍物速度比较快的情况下，无人车先等待障碍物经过关键节点 A、B 后，再进行调头。这样不仅能够提升算法的求解成功率，并且符合现实生活。既可以加强算法的应用能力，又可以保障实际道路的交通安全。

2)无人车压线

在问题五的模型中，当障碍物G和F同时向下移动时，我们原本的模型算法提出，无

人车的车宽满足两障碍物G和F之间的距离要求，无人车可以在它们两个障碍物之间的缝隙中行驶。但是考虑到实际的应用性、道路的安全性与规范性。我们可以知道，在原本模型中，无人车行驶在两者中间这个行为，既不符合道路交通的安全性，即无人车不应该压线行驶；又不符合道理交通的安全性，即车辆驾驶在两个障碍物之间，且它们之间的间距很小，非常容易发生交通事故。因此我们对算法模型进行改进。在此情况下，无人车刚完成调头驶入直道行驶后，应选择一个车道，跟在障碍物后方行驶。若两个障碍物本身的移动速度区别较大，则无人车可以进行车道变更，对两个障碍物进行超越；若两个障碍物移动速度相差甚微，基本齐头并进，则无人车应该在调头完成后减慢行驶速度，跟着它们行驶。

通过改进这两种障碍物移动情况下的算法，我们可以使无人车以尽可能高的频率进行实时运算，对路况进行实时判断，优化算法求解成功率和求解耗时两方面之余，还应该提高其安全性，以遵守交通法规的方式，在实际道路上行驶。使之能够应用在更广泛的环境中。

五、模型的综合评价和推广

5.1 模型的优点

我们的模型符合物理规律与现实实际，与道路上的实际行车较为贴合；

我们的模型同时考虑了安全、快速以及舒适性三个方面的综合影响，考虑的角度较多、方面较广；

模型的求解与计算严格符合题给的限制条件。

5.2 模型的缺点分析

我们的模型没有完整地给出解析解，而是一个待积分式；

我们的模型没有给出成熟的算法，即无法提供一套能够直接运用到实际的算法。

六、参考文献

[1] 林巧飞.高速环道几何线形设计与舒适度评价[J].山西交通科技,2020(02):5-8.

[2] 刘晓涛,蔡云飞,王田橙.基于 SVM 的受约束 D*算法在无人车寻路中的应用[J].计算机与数字工程,2017,45(09):1748-1754.

[3] 褚昭明,刘金广.我国关于机动车“限速值”是如何规定的　如何合理设定限速[J].汽车与安全,2018(12):72-75.

[4]龚星衡,吴肖伶,张卫华.基于姿态传感器进行感兴趣区域压缩的园区无人车图传[J].四川大学学报(自然科学版),2021,58(02):90-96.

[5]杨杨,任少杰,杨正才.基于改进型人工势场的无人车局部避障[J].湖北汽车工业学院学报,2020,34(04):5-10.

附录

运行环境：
C++ 版本：TDM-GCC 4.9.2 64-bit Release
操作系统: Microsoft Windows 10 家庭中文版 Version 10.0 (Build 17134)
Bit Server VM mixed mode

附件 1：几何学预处理.cpp
用于对运动学模型的几何规律做基础分解，方便后续调用。

```
#include <algorithm>
#include <cmath>
#include <cstdio>
#include <cstring>
#include <vector>
using namespace std;
#define EPS 1e-11
#define INF 1e18
#define PI 3.14159265358979323846
#define EQ(t1, t2) (abs((t1) - (t2)) < EPS)
#define LE(t1, t2) ((t1) <= (t2)-EPS)
#define LEQ(t1, t2) ((t1) < (t2) + EPS)
#define NEXT(i, n) ((i) + 1 >= (n) ? 0 : (i) + 1)
#define PREV(i, n) ((i) > 0 ? (i)-1 : (n)-1)
inline int SGN(double t) { return LE(t, 0) ? -1 : LE(0, t) ? 1 : 0; }
struct Point {
    double x, y;
    bool operator==(const Point& p) const { return EQ(x, p.x) && 
EQ(y, p.y); }
    bool operator<(const Point& p) const {
        return LEQ(x, p.x) && (LE(x, p.x) || LE(y, p.y));
    }
    Point operator+(Point& p) { return {x + p.x, y + p.y}; }
    Point operator-(Point& p) { return {x - p.x, y - p.y}; }
    double operator*(Point& p) { return x * p.y - y * p.x; }
    Point operator*(double value) { return {x * value, y * value}; }
    Point operator/(double value) { return {x / value, y / value}; }
    double dot(Point& p) { return x * p.x + y * p.y; }
    double r2() { return x * x + y * y; }
    double r() { return sqrt(x * x + y * y); }
    double dis2(Point& p) { return (*this - p).r2(); }
    double dis(Point& p) { return (*this - p).r(); }
    // 1 锛氱偣鍦ㄧ洿绾垮乏杈?-1 锛氱偣鍦ㄧ洿绾垮彸杈? 0 锛氱偣鍦ㄧ洿
绾夸笂
    int direction(Point& p1, Point& p2) {
        return SGN(x * (p1.y - p2.y) + p1.x * (p2.y - y) + p2.x * (y
- p1.y));
    }
    bool onLine(Point& p1, Point& p2) { return direction(p1, p2) ==
0; }
    // 鍒ゆ柇鐐规槸鍚︽湪绾挎   涓?
    bool onLineSeg(Point& p1, Point& p2) {
        return onLine(p1, p2) && inRect(p1, p2);
    }
    /*
        0 锛埄 1p 鍨傜洿 p1p2
        1: p2p 鍨傜洿 p1p2
        (0,1): p 鍦╬1 鐐瑰瀭绾夸笌 p2 鐐瑰瀭绾夸箣闂?
    */
    double lineRelation(Point& p1, Point& p2) {
        Point t = p2 - p1;
        return t.dot(*this - p1) / t.r2();
    }
    Point footPoint(Point& p1, Point& p2) {
        double r = lineRelation(p1, p2);
        return p1 + (p2 - p1) * r;
    }
    // 涓庣洿绾跨殑璺濈
    double lineDis(Point& p1, Point& p2) {
        return abs((p1 - *this) * (p2 - *this)) / p1.dis(p2);
    }
    double lineSegDis(Point& p1, Point& p2, Point& ret);
    double lineSegArrayDis(Point* p, int lineNum, Point& ret);
    // 鍏充簬鐩寸嚎鐨勫   绉扮偣
    Point mirror(Point& p1, Point& p2) {
        Point foot = footPoint(p1, p2);
        return foot * 2 - *this;
    }
    // 闀滃儚闂堟棆杞?
    Point rotate(double angle) {
        Point f = {sin(angle), cos(angle)};
        return {*this * f, dot(f)};
    }
    Point rotate90() { return {-y, x}; }
    double cosAngle(Point& p1, Point& p2) {
        Point t1 = *this - p1, t2 = *this - p2;
        return t1.dot(t2) / sqrt(t1.r2() * t2.r2());
    }
    double sinAngle(Point& p1, Point& p2) {
        Point t1 = *this - p1, t2 = *this - p2;
        return abs(t1 * t2) / sqrt(t1.r2() * t2.r2());
    }
    double tanAngle(Point& o) {
        if (EQ(x, o.x)) return y - o.y >= 0 ? INF : -INF;
        return (y - o.y) / (x - o.x);
    }
    double angle(Point& p1, Point& p2) { return acos(cosAngle(p1,
p2)); }
    double angle(Point& o) { return atan2(y - o.y, x - o.x); }
    bool inRect(Point& p1, Point& p2) {
        return LEQ((p1.x - x) * (p2.x - x), 0) &&
               LEQ((p1.y - y) * (p2.y - y), 0);
    }
    int inPolygon(Point* p, int n);
    int inConvex(Point* p, int n);
    int inCircle(Point& o, double r) {
        double dist = dis2(o);
        return SGN(r * r - dist);
    }
    void pointcut(Point& o, double r, Point& ret1, Point& ret2);
    Point nearnestPoint(Point& o, double r);
};
double Point::lineSegDis(Point& p1, Point& p2, Point& ret) {
    double r = lineRelation(p1, p2);
    if (LEQ(r, 0))
        ret = p1;
    else if (LEQ(1, r))
        ret = p2;
    else
        ret = footPoint(p1, p2);
    return dis(ret);
}
// input lineNum+1 points
double Point::lineSegArrayDis(Point* p, int lineNum, Point& ret) {
    Point tp;
    double td, mind = INF;
    for (int i = 0; i < lineNum; i++) {
        td = lineSegDis(p[i], p[i + 1], tp);
        if (LE(td, mind)) {
            mind = td;
            ret = tp;
        }
    }
    return mind;
}
// donnot include extream points, and donnot include coincidence.
inline bool lineSegLineSegIntersect(Point& p1, Point& p2, Point& q1,
                                    Point& q2) {
    Point pq1 = p1 - q1, p12 = p2 - p1, q12 = q2 - q1;
    return SGN(pq1 * q12) * SGN((p2 - q1) * q12) < 0 &&
           SGN(pq1 * p12) * SGN((p1 - q2) * p12) < 0;
}
// include extream points and coincidence.
inline bool lineSegLineSegIntersect2(Point& p1, Point& p2, Point&
q1,
                                     Point& q2) {
    if (!LEQ(min(q1.x, q2.x), max(p1.x, p2.x)) ||
        !LEQ(min(p1.x, p2.x), max(q1.x, q2.x)) ||
        !LEQ(min(q1.y, q2.y), max(p1.y, p2.y)) ||
        !LEQ(min(p1.y, p2.y), max(q1.y, q2.y)))
        return false;
    Point pq1 = p1 - q1, p12 = p2 - p1, q12 = q2 - q1;
    return SGN(pq1 * q12) * SGN((p2 - q1) * q12) <= 0 &&
           SGN(pq1 * p12) * SGN((p1 - q2) * p12) <= 0;
}
// donot include extream points, and donot include coincidence.
inline bool lineLineSegIntersect(Point& l1, Point& l2, Point& p1,
Point& p2) {
    Point line = l2 - l1;
    return SGN((p1 - l1) * line) * SGN((p2 - l1) * line) < 0;
}
// donnot include coincidence.
inline bool lineLineIntersect(Point& p1, Point& p2, Point& q1, Point&
q2) {
    return !EQ((p2 - p1) * (q2 - q1), 0);
}
inline Point lineLineIntersectPoint(Point& p1, Point& p2, Point& q1,
                                    Point& q2) {
    Point q12 = q2 - q1;
    double k = (p2 - p1) * q12;
    if (EQ(k, 0)) return {INF * INF, INF * INF};
    double r = ((q1 - p1) * q12) / k;
    return p1 + (p2 - p1) * r;
}
// 澶栧績
Point circumcenter(Point& p1, Point& p2, Point& p3) {
    Point t1 = (p1 + p2) * 0.5, t2, t3 = (p2 + p3) * 0.5, t4;
    t2 = t1 + (p1 - p2).rotate90();
    t4 = t3 + (p2 - p3).rotate90();
    return lineLineIntersectPoint(t1, t2, t3, t4);
}
// 鍐呭績
Point incenter(Point& p1, Point& p2, Point& p3) {
    double r12 = p1.dis(p2), r23 = p2.dis(p3), r31 = p3.dis(p1);
    Point t1 = (p2 * r31 + p3 * r12) / (r12 + r31),
          t2 = (p1 * r23 + p3 * r12) / (r12 + r23);
    return lineLineIntersectPoint(p1, t1, p2, t2);
}
// 鍨傚績
Point prepencenter(Point& p1, Point& p2, Point& p3) {
    Point t1 = p1 + (p2 - p3).rotate90();
    Point t2 = p2 + (p1 - p3).rotate90();
    return lineLineIntersectPoint(p1, t1, p2, t2);
}
// 閲嶅績
inline Point barycenter(Point& p1, Point& p2, Point& p3) {
    return (p1 + p2 + p3) / 3;
}
// 鍐呭垏鍦?
inline double apothem(Point& p1, Point& p2, Point& p3) {
    Point p12 = p2 - p1, p13 = p3 - p1, p23 = p3 - p2;
    return abs(p12 * p23) / (p12.r() + p13.r() + p23.r());
}
// 澶栨帴鍦?
inline double circumradius(Point& p1, Point& p2, Point& p3) {
    Point p12 = p2 - p1, p13 = p3 - p1, p23 = p3 - p2;
    return sqrt(p12.r2() * p23.r2() * p13.r2()) / (2 * abs(p12 *
p23));
}
// 閫嗘椂閽?锛岄『鏃堕拡-1
int getPolygonDirection(Point* p, int n) {
    int index = 0;
    for (int i = 1; i < n; i++) {
        if (p[i] < p[index]) index = i;
    }
    return p[index].direction(p[NEXT(index, n)], p[PREV(index, n)]);
}
bool checkConvex(Point* p, int n) {
    int direction = p[0].direction(p[n - 1], p[1]);
    if (direction == 0) return false;
    if (p[n - 1].direction(p[n - 2], p[0]) != direction) return
```

```
false;
    for (int i = n - 2; i > 0; i--) {
        if (p[i].direction(p[i - 1], p[i + 1]) != direction) return
false;
    }
    return true;
}
// 娉儿剩椤烘椂閽堥潰绉  负璐?
double polygonArea(Point* p, int n) {
    double area = 0;
    for (int i = n - 2; i > 0; i--) area += p[i].y * (p[i - 1].x -
p[i + 1].x);
    area += p[0].y * (p[n - 1].x - p[1].x);
    area += p[n - 1].y * (p[n - 2].x - p[0].x);
    return area / 2;
}
// 鍐呴儴杩斿洖 1 锛岃竟鐣?锛屽  閮?1
int Point::inPolygon(Point* p, int n) {
    int i, j = n - 1, odd = -1;
    for (i = 0; i < n; j = i++) {
        if (LE(p[i].y, y) != LE(p[j].y, y)) {
            double tx =
                (y - p[j].y) / (p[i].y - p[j].y) * (p[i].x - p[j].x)
+ p[j].x;
            if (LEQ(tx, x)) {
                if (LE(tx, x))
                    odd = -odd;
                else
                    return 0;
            }
        } else if (onLineSeg(p[i], p[j]))
            return 0;
    }
    return odd;
}
int Point::inConvex(Point* p, int n) {
    int _direction = p[1].direction(p[2], p[0]);
    if (direction(p[0], p[1]) != _direction) {
        if (onLineSeg(p[0], p[1])) return 0;
        return -1;
    }
    if (direction(p[n - 1], p[0]) != _direction) {
        if (onLineSeg(p[n - 1], p[0])) return 0;
        return -1;
    }
    int left = 2, right = n - 1;
    while (left < right) {
        int mid = (left + right) >> 1;
        if (direction(p[0], p[mid]) == _direction)
            left = mid + 1;
        else
            right = mid;
    }
    int ret = direction(p[left - 1], p[left]);
    return ret == _direction ? 1 : ret == 0 ? 0 : -1;
}
// 浠ヤ笅涓夊嚱鏁板弬鏁伴   闂嗘棩闂堟柟鍚?
// angle array size >= 2*n,return offset
int lineConvexIntersectPointInit(Point* p, int n, double angle[]) {
    int ret = 0;
    for (int i = 0, j = n - 1; i < n; j = i++) angle[j] =
p[i].angle(p[j]);
    do
        angle[ret + n] = angle[ret++];
    while (LE(angle[ret - 1], angle[ret]) && ret < n);
    return ret;
}
// ret 鍜完 et2 鍒喟堬涓鸿窛鐩寸嚎鏈€杩戝拰鏈€杩滅殑鐐逛笅鏍?
int lineConvexIntersect(Point& p1, Point& p2, Point* p, int n, double
angle[],
                        int offset, int& ret1, int& ret2) {
    int pos[2];
    double k[2];
    k[0] = p1.angle(p2);
    k[1] = k[0] <= 0 ? k[0] + PI : k[0] - PI;
    for (int i = 0; i < 2; i++) {
        pos[i] = (upper_bound(angle + offset, angle + offset + n,
k[i] - EPS) -
                  angle) %
                 n;
        if (p[pos[i]].onLine(p1, p2))
            return p[NEXT(pos[i], n)].onLine(p1, p2) ? 3 : 1;
    }
    ret1 = pos[0];
    ret2 = pos[1];
    return p[pos[0]].direction(p1, p2) == p[pos[1]].direction(p1,
p2) ? 0 : 2;
}
```

```
void lineConvexIntersectPoint(Point& p1, Point& p2, Point* p, int n,
int i1,
                              int i2, Point& ret1, Point& ret2) {
    for (int i = 0, l, r; i < 2; i++) {
        if (i) {
            l = min(i1, i2);
            r = max(i1, i2);
        } else {
            l = max(i1, i2);
            r = min(i1, i2) + n;
        }
        while (l < r) {
            int mid = (l + r) >> 1;
            if (p[mid % n].direction(p1, p2) == p[r %
n].direction(p1, p2))
                r = mid;
            else
                l = mid + 1;
        }
        l %= n;
        (i ? ret1 : ret2) = lineLineIntersectPoint(p1, p2, p[l],
p[PREV(l, n)]);
    }
}
bool lineSegInPolygon(Point p1, Point p2, Point* p, int n) {
    if (p2 < p1) swap(p1, p2);
    int s1 = p1.inPolygon(p, n), s2 = p2.inPolygon(p, n), id = -1,
pos = 0;
    if (s1 == -1 || s2 == -1) return false;
    while (p[pos].onLine(p1, p2)) pos++;
    int i = pos, j = pos, d = p[j].direction(p1, p2), d1, d2 = d;
    do {
        i = NEXT(i, n);
        d1 = d2;
        d2 = p[i].direction(p1, p2);
        if (d2 * d == -1) {
            if (lineSegLineSegIntersect(p[i], p[j], p1, p2)) return
false;
            if (d1 == 0 && p1 < p[id] && p[id] < p2 && p1 < p[j] &&
p[j] < p2)
                return false;
            d = d2;
        }
        if (d1 == 0 && d2 && p1 < p[j] && p[j] < p2) id = j;
        if (d2 == 0 && d1 && p1 < p[i] && p[i] < p2) id = i;
    } while ((j = i) != pos);
    if (s1 == 0 && s2 == 0) {
        if (id == -1) return ((p1 + p2) * 0.5).inPolygon(p, n) >= 0;
        Point q1 = p1, q2 = p2;
        for (int i = 0; i < n; i++) {
            if (p[i].onLine(p1, p2)) {
                if (p[i] < p[id]) q1 = max(q1, p[i]);
                if (p[id] < p[i]) q2 = min(q2, p[i]);
            }
        }
        return ((q1 + p[id]) * 0.5).inPolygon(p, n) >= 0 &&
               ((q2 + p[id]) * 0.5).inPolygon(p, n) >= 0;
    }
    return true;
}
Point gravityCenter(Point* p, int n) {
    if (n < 3) {
        if (n == 1)
            return p[0];
        else
            return (p[0] + p[1]) * 0.5;
    }
    double area = 0;
    Point ret = {0, 0};
    for (int i = 0, j = n - 1; i < n; j = i++) {
        double t = p[i] * p[j];
        area += t;
        ret.x += (p[i].x + p[j].x) * t;
        ret.y += (p[i].y + p[j].y) * t;
    }
    return ret / (3 * area);
}
// sort p[] first , ret[n] must be available to visit.
int convexHullSorted(Point* p, int n, Point* ret) {
    int j = 0;
    for (int i = 0; i < n; i++) {
        while (j >= 2 && p[i].direction(ret[j - 2], ret[j - 1]) !=
1) j--;
        ret[j++] = p[i];
    }
    int mid = j + 1;
    for (int i = n - 2; i >= 0; i--) {
        while (j >= mid && p[i].direction(ret[j - 2], ret[j - 1]) !=
1) j--;
        ret[j++] = p[i];
    }
    return j - 1;
}
void convexHullSorted(Point* p, int n, Point* up, int& retUp, Point*
down,
                      int& retDown) {
    retUp = retDown = 0;
    for (int i = 0; i < n; i++) {
        while (retUp >= 2 && p[i].direction(up[retUp - 2], up[retUp
- 1]) != -1)
            retUp--;
        while (retDown >= 2 &&
               p[i].direction(down[retDown  -  2],  down[retDown  -
1]) != 1)
            retDown--;
        up[retUp++] = p[i];
        down[retDown++] = p[i];
    }
}
//  p2 缁眃 1 闆嗘棕閻堣浆 90 搴︿唬琛ㄥ钩闈㈠晥閮   紝鑷      澧炲姞 4 涓
  崐骞抽潰鍋氳竟鐣?
#define judge(p, q) (p.direction(q.second[0], q.second[1]) < 0)
#define intersect(p, q) \
    lineLineIntersectPoint(p.second[0],  p.second[1],  q.second[0],
q.second[1])
int halfPlainIntersect(Point (*p)[2], int n, Point* ret) {
    vector<pair<double, Point*> > v(n), line(n);
    for    (int    i    =    0;    i    <    n;    i++)    v[i]    =
make_pair(p[i][1].angle(p[i][0]), p[i]);
    sort(v.begin(), v.end());
    int m = 0, l = 0, r = 0;
    for (int i = 1; i < n; i++) {
        if (!EQ(v[i].first, v[m].first))
            v[++m] = v[i];
        else if (!judge(v[i].second[0], v[m]))
            v[m] = v[i];
    }
    if  (EQ(v[0].first  +  2  *  PI,  v[m].first)  &&  !judge(v[m-
-].second[0], v[0]))
        v[0] = v[m + 1];
    vector<Point> q(n);
    line[0] = v[0];
    for (int i = 1; i <= m; i++) {
        while (l < r && judge(q[r - 1], v[i])) r--;
        while (l < r && judge(q[l], v[i])) l++;
        if (l == r && LEQ(line[l].first + PI, v[i].first)) return 0;
        line[++r] = v[i];
        q[r - 1] = intersect(line[r - 1], line[r]);
    }
    while (l < r && judge(q[r - 1], line[l])) r--;
    if (l == r) return 0;
    q[r] = intersect(line[l], line[r]);
    int num = unique(q.begin() + l, q.begin() + r + 1) - q.begin() -
l;
    memcpy(ret, &q[l], sizeof(Point) * num);
    if (num > 1 && ret[0] == ret[num - 1]) num--;
    return num;
}
// These points must be put counter-clockwise.
int polygonKernel(Point* p, int n, Point* ret) {
    Point(*q)[2] = new Point[n][2];
    for (int i = 0, j = n - 1; i < n; j = i++) q[j][0] = p[j], q[j][1]
= p[i];
    int m = halfPlainIntersect(&q[0], n, ret);
    delete[] q;
    return m;
}
// return two points representing ax+by<=c
void getPlain(double a, double b, double c, Point* p) {
    if (EQ(a, 0))
        p[0] = {0, c / b};
    else
        p[0] = {c / a, 0};
    p[1] = {-b, a};
    p[1] = p[1] + p[0];
}
// These points must be put counter-clockwise.
// Ensure p[n] exists and p[n]=p[0].
double convexDiameter(Point* p, int n, Point& ret1, Point& ret2) {
    double ret = 0;
    for (int i = 0, j = 1; i < n; i++) {
        double t1 = (p[i + 1] - p[i]) * (p[j] - p[i]), t2;
        for (; LE(t1, t2 = (p[i + 1] - p[i]) * (p[j + 1] - p[i]));
t1 = t2) {
            if (++j == n) j = 0;
        }
        double td2 = p[i].dis2(p[j]);
        if (ret < td2) {
            ret = td2;
            ret1 = p[i];
            ret2 = p[j];
        }
        td2 = p[i + 1].dis2(p[j]);
        if (ret < td2) {
            ret = td2;
            ret1 = p[i + 1];
            ret2 = p[j];
        }
    }
    return sqrt(ret);
}
// These points must be put counter-clockwise.
// Ensure p[n] exists and p[n]=p[0].
double convexWidth(Point* p, int n) {
    double ret = INF;
    for (int i = 0, j = 1; i < n; i++) {
        double t1 = (p[i + 1] - p[i]) * (p[j] - p[i]), t2;
        for (; LE(t1, t2 = (p[i + 1] - p[i]) * (p[j + 1] - p[i]));
t1 = t2) {
            if (++j == n) j = 0;
        }
        ret = min(ret, t1 / p[i].dis(p[i + 1]));
    }
    return ret;
}
struct NearestPointsStruct {
    Point p1, p2;
    double d2;
    vector<Point> v;
};
inline bool nearestPointsCmp(Point& p1, Point& p2) {
    return LEQ(p1.y, p2.y) && (LE(p1.y, p2.y) || LE(p1.x, p2.x));
}
void nearestPointsInternal(Point* p, int left, int right,
                           NearestPointsStruct& s) {
    if (right - left < 8) {
        for (int i = left; i < right; i++) {
            for (int j = i + 1; j < right; j++) {
                double td2 = p[j].dis2(p[i]);
                if (td2 < s.d2) {
                    s.d2 = td2;
                    s.p1 = p[i];
                    s.p2 = p[j];
                }
            }
        }
        return;
    }
    int mid = (left + right) >> 1;
    nearestPointsInternal(p, left, mid, s);
    nearestPointsInternal(p, mid, right, s);
    s.v.clear();
    double l = (p[mid - 1].x + p[mid].x) / 2, d = sqrt(s.d2);
    for  (int  i  =  mid  -  1;  i  >=  left  &&  l  -  p[i].x  <  d;  i--)
s.v.push_back(p[i]);
    for  (int  i  =  mid;  i  <  right  &&  p[i].x  -  l  <  d;  i++)
s.v.push_back(p[i]);
    sort(s.v.begin(), s.v.end(), nearestPointsCmp);
    for (unsigned int i = 0; i < s.v.size(); i++) {
        for (unsigned int j = i + 1; j < s.v.size() && s.v[j].y -
s.v[i].y < d;
             j++) {
            double td2 = s.v[j].dis2(s.v[i]);
            if (td2 < s.d2) {
                s.d2 = td2;
                s.p1 = s.v[i];
                s.p2 = s.v[j];
            }
        }
    }
}
double nearestPointsSorted(Point* p, int n, Point& ret1, Point& ret2)
{
    NearestPointsStruct s;
    s.d2 = INF;
    s.v.reserve(n);
    nearestPointsInternal(p, 0, n, s);
    ret1 = s.p1;
    ret2 = s.p2;
    return sqrt(s.d2);
}
double  farthestPointsSorted(Point*  p,  int  n,  Point&  ret1,  Point&
ret2) {
    vector<Point> v(n + 1);
    int cnt = convexHullSorted(p, n, &*v.begin());
    v[n] = v[0];
    return convexDiameter(&*v.begin(), cnt, ret1, ret2);
}
int circleLineRelation(Point& o, double r, Point& p1, Point& p2) {
    double d = o.lineDis(p1, p2);
    if (LE(d, r)) return 1;
    if (LE(r, d)) return -1;
    return 0;
}
int circleCircleRelation(Point& o1, double r1, Point& o2, double r2)
{
    double r = o1.dis(o2);
    if (LE(r1 + r2, r)) return 4;
    if (LEQ(r1 + r2, r)) return 3;
    double sub = abs(r1 - r2);
    if (LE(sub, r)) return 2;
    if (LEQ(sub, r)) return 1;
    return 0;
}
// include extream points.
bool circleLineSegIntersect(Point& o, double r, Point& p1, Point&
p2) {
    int t1 = p1.inCircle(o, r), t2 = p2.inCircle(o, r);
    if (t1 >= 0 || t2 >= 0) return t1 != 1 || t2 != 1;
    double t = o.lineRelation(p1, p2);
    if (t >= 1 || t <= 0) return false;
    Point foot = p1 + (p2 - p1) * t;
    return foot.inCircle(o, r) >= 0;
}
// ret1 is near p1,ret2 is near p2.
void circleLineIntersect(Point& o, double r, Point& p1, Point& p2,
Point& ret1,
                         Point& ret2) {
    Point foot = o.footPoint(p1, p2);
    double t = r * r - o.dis2(foot);
    t = LEQ(t, 0) ? 0 : sqrt(t / p1.dis2(p2));
    ret1 = foot + (p1 - p2) * t;
    ret2 = foot * 2 - ret1;
}
void circleCircleIntersectPoint(Point& o1, double r1, Point& o2,
double r2,
                                Point& ret1, Point& ret2) {
    double d2 = o1.dis2(o2);
    double t1 = (r1 * r1 - r2 * r2) / (2 * d2) + 0.5;
    double t2 = r1 * r1 / d2 - t1 * t1;
    t2 = LEQ(t2, 0) ? 0 : sqrt(t2);
    Point foot = o1 + (o2 - o1) * t1;
    ret1 = foot + (o2 - o1).rotate90() * t2;
    ret2 = foot * 2 - ret1;
}
double circleCircleIntersectArea(Point& o1, double r1, Point& o2,
double r2) {
    int r = circleCircleRelation(o1, r1, o2, r2);
    if (r >= 3) return 0;
    if (r <= 1) return min(r1, r2) * min(r1, r2) * PI;
    Point p1, p2;
    circleCircleIntersectPoint(o1, r1, o2, r2, p1, p2);
    double ret = r1 * r1 * o1.angle(p1, o2) + r2 * r2 * o2.angle(p1,
o1);
    return ret - sqrt(o1.dis2(o2) * p1.dis2(p2)) / 2;
}
void Point::pointcut(Point& o, double r, Point& ret1, Point& ret2) {
    double t1 = r * r / dis2(o);
    Point foot = o + (*this - o) * t1;
    double t2 = t1 - t1 * t1;
    t2 = LEQ(t2, 0) ? 0 : sqrt(t2);
    ret1 = foot + (*this - o).rotate90() * t2;
    ret2 = foot * 2 - ret1;
}
// ret[0] and ret[2] are on circle o1,ret[1] and ret[3] are on circle
o2.
void circleCirclePointcutOuter(Point& o1, double r1, Point& o2,
double r2,
                               Point* ret) {
    Point o12 = o2 - o1;
    double d12 = o12.r2(), r = (r1 - r2) / d12;
    Point foot1 = o1 + o12 * (r * r1), foot2 = o2 + o12 * (r * r2);
    double t = 1 / d12 - r * r;
    t = LEQ(t, 0) ? 0 : sqrt(t);
    Point line = o12.rotate90();
    ret[0] = foot1 + line * (t * r1);
    ret[1] = foot2 + line * (t * r2);
    ret[2] = foot1 * 2 - ret[0];
    ret[3] = foot2 * 2 - ret[1];
}
void circleCirclePointcutInner(Point& o1, double r1, Point& o2,
double r2,
                               Point* ret) {
    Point o12 = o2 - o1;
    double d12 = o12.r2(), r = (r1 + r2) / d12;
    Point foot1 = o1 + o12 * (r * r1), foot2 = o2 - o12 * (r * r2);
    double t = 1 / d12 - r * r;
    t = LEQ(t, 0) ? 0 : sqrt(t);
    Point line = o12.rotate90();
    ret[0] = foot1 + line * (t * r1);
    ret[1] = foot2 - line * (t * r2);
    ret[2] = foot1 * 2 - ret[0];
    ret[3] = foot2 * 2 - ret[1];
}
Point Point::nearestPoint(Point& o, double r) {
    Point p = *this - o;
    double d = p.r();
    if (EQ(d, 0)) return o;
    return o + p * (r / d);
}
// Upset the order before using this function.
double minCoveringCircle(Point* p, int n, Point& ret) {
    if (n == 1) {
        ret = p[0];
        return 0;
    }
    double r2 = p[0].dis2(p[1]) / 4;
    ret = (p[0] + p[1]) * 0.5;
    for (int i = 2; i < n; i++) {
        if (LE(r2, ret.dis2(p[i]))) {
            ret = (p[0] + p[i]) * 0.5;
            r2 = p[0].dis2(p[i]) / 4;
            for (int j = 1; j < i; j++) {
                if (LE(r2, ret.dis2(p[j]))) {
                    ret = (p[i] + p[j]) * 0.5;
                    r2 = p[i].dis2(p[j]) / 4;
                    for (int k = 0; k < j; k++) {
                        if (LE(r2, ret.dis2(p[k]))) {
                            ret = circumcenter(p[i], p[j], p[k]);
                            r2 = ret.dis2(p[k]);
                        }
                    }
                }
            }
        }
    }
    return sqrt(r2);
}
int unitCoveringCircle(Point* p, int n, double r) {
    int ret = 0;
    vector<pair<double, bool> > v;
    v.reserve(2 * n);
    double t = r * r * 4;
    for (int i = 0; i < n; i++) {
        v.clear();
        int value = 0;
        for (int j = 0; j < n; j++) {
            if (LEQ(p[i].dis2(p[j]), t) && i != j) {
                double a = p[j].angle(p[i]);
                double b = acos(p[i].dis(p[j]) / r / 2);
                double t1 = a - b, t2 = a + b;
                if (t1 < -PI / 2) {
                    t1 += 2 * PI;
                    if (t2 < -PI / 2)
                        t2 += 2 * PI;
                    else
                        value++;
                }
                v.push_back(make_pair(t1, true));
                v.push_back(make_pair(t2, false));
            }
        }
        sort(v.begin(), v.end());
        if (value > ret) ret = value;
        for (unsigned int j = 0; j < v.size(); j++) {
            if (v[j].second) {
                value++;
                if (value > ret) ret = value;
            } else
                value--;
        }
    }
    return ret + 1;
}
/*
double circlePolygonAreaIntersect(Point& o,double r,Point* p,int n){
    double area=0;
    Point p1,p2;
    for(int i=0,j=n-1;i<n;j=i++){
        int f1=p[i].inCircle(o,r),f2=p[j].inCircle(o,r);
        if(f1>=0&&f2>=0)area+=(o-p[i])*(o-p[j]);
        else if(f1>=0&&f2<0){
            circleLineIntersect(o,r,p[i],p[j],p1,p2);
            area+=(o-p[i])*(o-p2);
            area+=asin(o.sinAngle(p2,p[j]))*r*r;
        }
        else if(f1<0&&f2>=0){
            circleLineIntersect(o,r,p[i],p[j],p1,p2);
            area+=(o-p1)*(o-p[j]);
            area+=asin(o.sinAngle(p[i],p1))*r*r;
        }
        else if(circleLineSegIntersect(o,r,p[i],p[j])){
            circleLineIntersect(o,r,p[i],p[j],p1,p2);
            area+=(o-p1)*(o-p2);

area+=(asin(o.sinAngle(p[i],p1))+asin(o.sinAngle(p2,p[j])))*r*r;
        }
        else area+=asin(o.sinAngle(p[i],p[j]))*r*r;
    }
    return abs(area/2);
}
*/
```

附件2：coordinate_transfer.cpp
用于坐标转换。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int INF=0x3f3f3f3f;
const long long mod=100000007;
const double e=2.718281828459045;
const double pi=3.1415926535;
#define CK cout<<"OK\n";
int main(){
    freopen("coordinate_transfer_in.txt","r",stdin);
    freopen("coordinate_transfer_out.txt","w",stdout);
    double x[100],y[100];
    double ax[100],ay[100];
    int n=0;
    double k=15.624/27.3244;
    double dx=2.998806,dy=0.220528;
    while(~scanf("%lf%lf",&x[n],&y[n])){
        x[n]-=dx;
        y[n]-=dy;
        n++;
    }
    for(int i=0;i<n;i++){
        double a=k*k+1;
        double b=-2*x[i]-2*k*y[i];
        double c=x[i]*x[i]+y[i]*y[i];
        double xi=-b/2/a;
        ax[i]=-sqrt(
            max(a*xi*xi+b*xi+c,0.0)
        )+15;

        a=1+1/k/k;
        b=2/k*y[i]-2*x[i];
        c=x[i]*x[i]+y[i]*y[i];
        xi=-b/2/a;
        ay[i]=sqrt(
            max(a*xi*xi+b*xi+c,0.0)
        );
    }
    for(int i=0;i<n;i++){
        printf("%.7lf\t%.7lf\n",ax[i],ay[i]);
    }
    return 0;
}
```

附件 3：pic_1.cpp
用于提供图片所需数据。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int INF=0x3f3f3f3f;
const long long mod=100000007;
const double e=2.718281828459045;
const double pi=3.1415926535;
#define CK cout<<"OK\n";
int main(){

    freopen("pic_1_in.txt","r",stdin);
    freopen("pic_1_out.txt","w",stdout);
    double a[100],b[100];
    int n=0;
    while(~scanf("%lf%lf",&a[n],&b[n])){
        n++;
    }
    for(int i=0;i<n;i++){
        printf(″%.5lf,″,a[i]);
    }
    printf(″\n″);
    for(int i=0;i<n;i++){
        printf(″%.5lf,″,b[i]);
    }
    printf(″\n″);
    return 0;
}
```

附件 4：计算圆轨迹.cpp
用于计算匀速圆周运动中的轨迹。

```
/*
    姹傚渾浜?
    Example 锛?
        缁欏畾涓や釜鍦嗗拰锛屽渾蹇冨潗鏍囧強鍗婂緞锛屾眰涓ゅ渾鐨勭浉浜ら潰绉?
鍜屽渾锛屽眰鐩镐氦鍑忛潰绉?
    Solution 锛?
        闈㈢H 涓哄洪ょ鍦嗗氦闈㈢H-2*澶у 鐨勫氦闈㈢H+灏忓渾浜ら潰绉?
*/
using namespace std;
const double EPS = 1e-12;
const double PI = acos(-1.0);
const int inf = ~0U >> 1;
int sgn(double x) {
    if (fabs(x) < EPS) return 0;
    if (x < 0)
        return -1;
    else
        return 1;
}
struct Point {
    double x, y;
    Point() {}
    Point(double x1, double y1) {
        x = x1;
        y = y1;
    }
    Point operator-(const Point &b) const { return Point(x - b.x, y
- b.y); }
```

```
    double operator*(const Point &b) const { return x * b.x + y *
b.y; }  //鐐圭 H
    double operator^(const Point &b) const { return x * b.y - y *
b.x; }  //鍙夌 H
};
struct Line {
    Point u, v;
    Line() {}
    Line(Point u1, Point v1) {
        u = u1;
        v = v1;
    }
};
double dist(Point a, Point b) { return sqrt((a - b) * (a - b)); }
// 涓ゅ渾鐩镐氦闈㈢Н锛H 锛宎锛宆鏈   渾蹇冪紶 r1 锛涜 2 鏈   崐寰?
double Area(Point a, double r1, Point b, double r2) {
    double k = dist(a, b);
    if (k + EPS >= r1 + r2) return 0;  //鐩稿垏鎴栫€呯浉绂?
    if (k <= fabs(r1 - r2) + EPS) {
        double R = min(r1, r2);
        return PI * R * R;
    }
    double x = (k * k + r1 * r1 - r2 * r2) / (2.0 * k);
    double w1 = acos(x / r1);
    double w2 = acos((k - x) / r2);
    return (w1 * r1 * r1 + w2 * r2 * r2 - k * r1 * sin(w1));
}
int main(int T) {
    scanf("%d", &T);
    int cnt = 1;
    while (T--) {
        double x, y, r, R;
        Point p1, p2;
        scanf("%lf%lf", &r, &R);
        scanf("%lf%lf", &x, &y);
        p1 = Point(x, y);
        scanf("%lf%lf", &x, &y);
        p2 = Point(x, y);
        double ans =
            Area(p1, R, p2, R) - 2 * Area(p1, R, p2, r) + Area(p1,
r, p2, r);
        printf("Case #%d: %.6f\n", cnt++, ans);
    }
    return 0;
```

```
}
```

附件 5：角度变换.cpp

用于处理第一阶段和第三阶段中方向盘正在转动的情况。

```
/*
    鏃嬭浆鍗″３
    娉ㄦ剰鍦ㄤ娇鐢ㄥ墠鍏堟眰鍑稿寘
*/
// 鏃嬭浆鍗″３锛屾眰骞抽潰鏈€杩滅偣瀵?
double rotating_calipers(Point p[], int n) {
    double ans = 0;
    Point v;
    int cur = 1, ans1 = 0, ans2 = 0;
    for (int i = 0; i < n; i++) {
        v = p[i] - p[(i + 1) % n];
        while ((v ^ (p[(cur + 1) % n] - p[cur])) < 0) cur = (cur +
1) % n;
        double tmp = dist(p[i], p[cur]);
        if (tmp > ans) ans = tmp, ans1 = i, ans2 = cur;
        tmp = dist(p[(i + 1) % n], p[(cur + 1) % n]);
        if (tmp > ans) ans = tmp, ans1 = (i + 1) % n, ans2 = (cur +
1) % n;
    }
    printf("%.12f  %.12f  %.12f  %.12f\n",  p[ans1].x,  p[ans1].y,
p[ans2].x,
           p[ans2].y);
    return ans;
}
// 鏃嬭浆鍗″３锛屾眰涓夎  褰㈤潰绉  渶澶 y€肩殑涓ゅ€?
double rotating_calipers(Point p[], int n) {
    double ans = 0;
    Point v;
    for (int i = 0; i < n; i++) {
        int j = (i + 1) % n;
        int k = (j + 1) % n;
        while (j != i && k != i) {
            ans = max(ans, abs((p[j] - p[i]) ^ (p[k] - p[i])));
            while (((p[i] - p[j]) ^ (p[(k + 1) % n] - p[k])) < 0)
                k = (k + 1) % n;
            j = (j + 1) % n;
        }
    }
    return ans;
```

```
}
// 鏃婅浆鍗″ 3 锛屾眰涓や釜鍑稿寘鐨勬渶灏忚窛绂?
// 鐐筽 0 鍒扮嚎娈 1p2 鐨勮窛绂?
double pointtoseg(Point p0, Point p1, Point p2) {
    return dist(p0, NearestPointToLineSeg(p0, Line(p1, p2)));
}
// 骞宠   缂挎   p0p1 鍜宲 2p3 鐨勮窛绂?
double dispallseg(Point p0, Point p1, Point p2, Point p3) {
    double ans1 = min(pointtoseg(p0, p2, p3), pointtoseg(p1, p2,
p3));
    double ans2 = min(pointtoseg(p2, p0, p1), pointtoseg(p3, p0,
p1));
    return min(ans1, ans2);
}
// 寰楀埌鍚戦噺 a1a2 鍜宐 1b2 鐨勫緗缃   另绯?
double Get_angle(Point a1, Point a2, Point b1, Point b2) {
    Point t = b1 - (b2 - a1);
    return (a2 - a1) ^ (t - a1);
}
double rotating_calipers(Point p[], int np, Point q[], int nq) {
    int sp = 0, sq = 0;
    for (int i = 0; i < np; i++)
        if (sgn(p[i].y - p[sp].y) < 0) sp = i;
    for (int i = 0; i < nq; i++)
        if (sgn(q[i].y - q[sq].y) > 0) sq = i;
    double tmp;
    double ans = 1e99;
    for (int i = 0; i < np; i++) {
        while (sgn(tmp = Get_angle(p[sp], p[(sp + 1) % np], q[sq],
                                   q[(sq + 1) % nq])) < 0)
            sq = (sq + 1) % nq;
        if (sgn(tmp) == 0)
            ans = min(ans, dispallseg(p[sp], p[(sp + 1) % np], q[sq],
                                      q[(sq + 1) % nq]));
        else
            ans = min(ans, pointtoseg(q[sq], p[sp], p[(sp + 1) %
np]));
        sp = (sp + 1) % np;
    }
    return ans;
}
double solve(Point p[], int n, Point q[], int m) {
    return min(rotating_calipers(p, n, q, m), rotating_calipers(q,
m, p, n));
```

```
}
```

附件 6：

曲线行驶过程结束的 x 坐标与方向盘转速和无人车行驶速度的关系表

	10	11	12	13	14	15	16	17	18	19	20
100	12.64	13.17	13.73	15.12	18.2	21.25	24.24	27.12	29.86	32.42	34.78
101	12.59	13.11	13.66	14.73	17.78	20.81	23.78	26.65	29.39	31.96	34.34
102	12.54	13.05	13.59	14.35	17.37	20.37	23.32	26.18	28.92	31.5	33.89
103	12.49	12.99	13.53	14.1	16.96	19.94	22.87	25.72	28.46	31.04	33.45
104	12.45	12.94	13.47	14.03	16.56	19.51	22.43	25.27	28	30.59	33.01
105	12.4	12.89	13.41	13.96	16.17	19.1	21.99	24.82	27.55	30.14	32.56
106	12.36	12.84	13.35	13.89	15.79	18.69	21.56	24.38	27.1	29.69	32.12
107	12.32	12.79	13.29	13.82	15.41	18.29	21.14	23.94	26.65	29.24	31.69
108	12.28	12.74	13.23	13.76	15.04	17.89	20.72	23.51	26.21	28.8	31.25
109	12.24	12.69	13.18	13.69	14.68	17.5	20.31	23.08	25.77	28.36	30.82
110	12.2	12.64	13.12	13.63	14.32	17.12	19.91	22.66	25.34	27.93	30.38
111	12.16	12.6	13.07	13.57	14.1	16.75	19.51	22.25	24.92	27.5	29.96
112	12.12	12.55	13.02	13.51	14.03	16.38	19.12	21.84	24.5	27.07	29.53
113	12.08	12.51	12.97	13.45	13.97	16.02	18.74	21.44	24.08	26.65	29.11
114	12.05	12.47	12.92	13.4	13.9	15.66	18.36	21.04	23.68	26.23	28.69
115	12.01	12.43	12.87	13.34	13.84	15.31	17.99	20.65	23.27	25.82	28.27
116	11.98	12.39	12.82	13.29	13.78	14.97	17.62	20.27	22.87	25.41	27.86
117	11.95	12.35	12.78	13.23	13.72	14.63	17.26	19.89	22.48	25.01	27.46
118	11.92	12.31	12.73	13.18	13.66	14.3	16.91	19.51	22.09	24.61	27.05
119	11.88	12.27	12.69	13.13	13.6	14.1	16.56	19.15	21.71	24.22	26.65
120	11.85	12.24	12.65	13.08	13.55	14.04	16.22	18.78	21.33	23.83	26.25
121	11.82	12.2	12.6	13.04	13.49	13.97	15.88	18.43	20.95	23.44	25.86
122	11.79	12.16	12.56	12.99	13.44	13.91	15.55	18.08	20.59	23.06	25.47
123	11.77	12.13	12.52	12.94	13.39	13.86	15.22	17.73	20.22	22.68	25.09
124	11.74	12.1	12.49	12.9	13.34	13.8	14.9	17.39	19.87	22.31	24.71
125	11.71	12.07	12.45	12.86	13.29	13.74	14.59	17.05	19.51	21.95	24.33
126	11.68	12.03	12.41	12.81	13.24	13.69	14.28	16.72	19.17	21.59	23.96
127	11.66	12	12.37	12.77	13.19	13.63	14.1	16.4	18.82	21.23	23.59
128	11.63	11.97	12.34	12.73	13.14	13.58	14.04	16.08	18.49	20.88	23.23
129	11.61	11.94	12.3	12.69	13.1	13.53	13.98	15.76	18.15	20.53	22.87
130	11.58	11.91	12.27	12.65	13.05	13.48	13.93	15.45	17.83	20.19	22.52
131	11.56	11.88	12.24	12.61	13.01	13.43	13.87	15.15	17.5	19.85	22.17
132	11.54	11.86	12.2	12.57	12.97	13.38	13.82	14.85	17.18	19.51	21.82
133	11.51	11.83	12.17	12.54	12.92	13.33	13.76	14.55	16.87	19.19	21.48
134	11.49	11.8	12.14	12.5	12.88	13.29	13.71	14.26	16.56	18.86	21.14
135	11.47	11.78	12.11	12.46	12.84	13.24	13.66	14.1	16.26	18.54	20.81
136	11.45	11.75	12.08	12.43	12.8	13.2	13.61	14.04	15.96	18.22	20.48

137	11.43	11.73	12.05	12.4	12.76	13.15	13.56	13.99	15.66	17.91	20.15
138	11.41	11.7	12.02	12.36	12.73	13.11	13.51	13.94	15.37	17.6	19.83
139	11.39	11.68	11.99	12.33	12.69	13.07	13.47	13.88	15.08	17.3	19.51
140	11.37	11.65	11.96	12.3	12.65	13.03	13.42	13.83	14.8	17	19.2
141	11.35	11.63	11.94	12.27	12.62	12.99	13.38	13.78	14.52	16.71	18.89
142	11.33	11.61	11.91	12.24	12.58	12.95	13.33	13.73	14.24	16.42	18.59
143	11.31	11.59	11.88	12.21	12.55	12.91	13.29	13.68	14.1	16.13	18.29
144	11.29	11.56	11.86	12.18	12.51	12.87	13.24	13.64	14.05	15.85	17.99
145	11.27	11.54	11.83	12.15	12.48	12.83	13.2	13.59	13.99	15.57	17.7
146	11.26	11.52	11.81	12.12	12.45	12.8	13.16	13.54	13.94	15.29	17.41
147	11.24	11.5	11.79	12.09	12.42	12.76	13.12	13.5	13.89	15.02	17.12
148	11.22	11.48	11.76	12.06	12.38	12.72	13.08	13.46	13.85	14.75	16.84
149	11.21	11.46	11.74	12.04	12.35	12.69	13.04	13.41	13.8	14.49	16.56
150	11.19	11.44	11.72	12.01	12.32	12.65	13	13.37	13.75	14.23	16.29
151	11.17	11.42	11.69	11.98	12.29	12.62	12.97	13.33	13.71	14.1	16.02
152	11.16	11.4	11.67	11.96	12.26	12.59	12.93	13.29	13.66	14.05	15.75
153	11.14	11.39	11.65	11.93	12.24	12.56	12.89	13.25	13.62	14	15.48
154	11.13	11.37	11.63	11.91	12.21	12.52	12.86	13.21	13.57	13.95	15.22
155	11.11	11.35	11.61	11.88	12.18	12.49	12.82	13.17	13.53	13.91	14.97
156	11.1	11.33	11.59	11.86	12.15	12.46	12.79	13.13	13.49	13.86	14.71
157	11.08	11.32	11.57	11.84	12.13	12.43	12.75	13.09	13.45	13.81	14.46
158	11.07	11.3	11.55	11.81	12.1	12.4	12.72	13.06	13.41	13.77	14.22
159	11.06	11.28	11.53	11.79	12.07	12.37	12.69	13.02	13.37	13.73	14.1
160	11.04	11.27	11.51	11.77	12.05	12.34	12.66	12.98	13.33	13.68	14.05
161	11.03	11.25	11.49	11.75	12.02	12.32	12.63	12.95	13.29	13.64	14
162	11.02	11.24	11.47	11.73	12	12.29	12.59	12.91	13.25	13.6	13.96
163	11	11.22	11.45	11.71	11.98	12.26	12.56	12.88	13.21	13.56	13.91
164	10.99	11.21	11.44	11.69	11.95	12.24	12.53	12.85	13.18	13.52	13.87
165	10.98	11.19	11.42	11.67	11.93	12.21	12.5	12.81	13.14	13.48	13.83
166	10.97	11.18	11.4	11.65	11.91	12.18	12.48	12.78	13.1	13.44	13.78
167	10.96	11.16	11.39	11.63	11.88	12.16	12.45	12.75	13.07	13.4	13.74
168	10.94	11.15	11.37	11.61	11.86	12.13	12.42	12.72	13.03	13.36	13.7
169	10.93	11.13	11.35	11.59	11.84	12.11	12.39	12.69	13	13.32	13.66
170	10.92	11.12	11.34	11.57	11.82	12.08	12.36	12.66	12.97	13.29	13.62
171	10.91	11.11	11.32	11.55	11.8	12.06	12.34	12.63	12.93	13.25	13.58
172	10.9	11.09	11.31	11.53	11.78	12.04	12.31	12.6	12.9	13.22	13.54
173	10.89	11.08	11.29	11.52	11.76	12.01	12.29	12.57	12.87	13.18	13.5
174	10.88	11.07	11.28	11.5	11.74	11.99	12.26	12.54	12.84	13.15	13.47
175	10.87	11.06	11.26	11.48	11.72	11.97	12.24	12.51	12.81	13.11	13.43
176	10.86	11.04	11.25	11.47	11.7	11.95	12.21	12.49	12.78	13.08	13.39
177	10.85	11.03	11.23	11.45	11.68	11.93	12.19	12.46	12.75	13.05	13.36
178	10.84	11.02	11.22	11.43	11.66	11.91	12.16	12.43	12.72	13.01	13.32
179	10.83	11.01	11.21	11.42	11.64	11.88	12.14	12.41	12.69	12.98	13.29
180	10.82	11	11.19	11.4	11.63	11.86	12.12	12.38	12.66	12.95	13.25

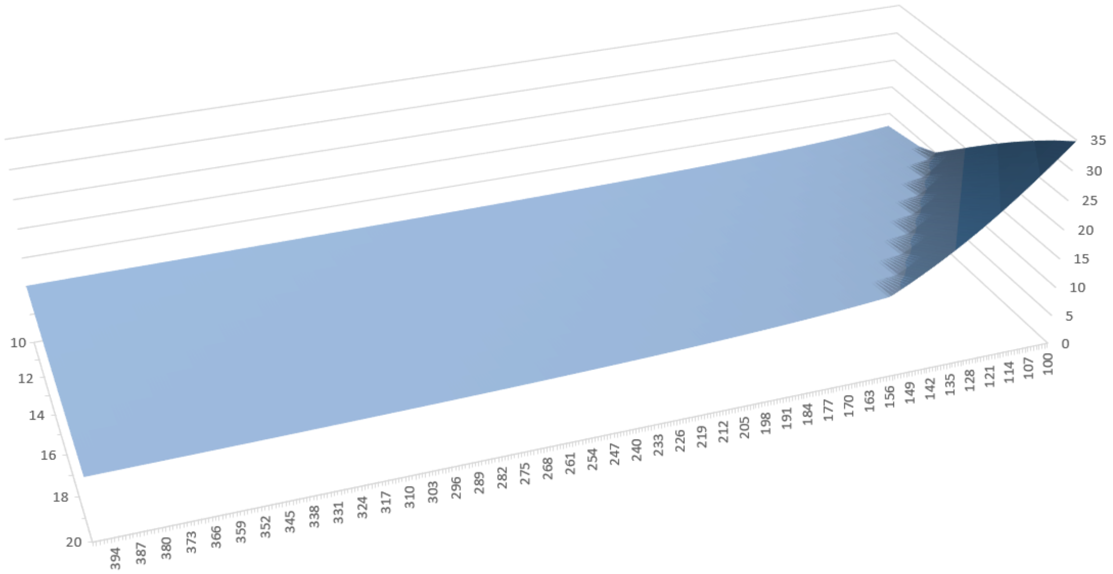
181	10.81	10.99	11.18	11.39	11.61	11.84	12.09	12.36	12.63	12.92	13.22
182	10.8	10.97	11.17	11.37	11.59	11.82	12.07	12.33	12.6	12.89	13.19
183	10.79	10.96	11.15	11.36	11.57	11.8	12.05	12.31	12.58	12.86	13.15
184	10.78	10.95	11.14	11.34	11.56	11.79	12.03	12.28	12.55	12.83	13.12
185	10.77	10.94	11.13	11.33	11.54	11.77	12.01	12.26	12.52	12.8	13.09
186	10.76	10.93	11.11	11.31	11.52	11.75	11.99	12.24	12.5	12.77	13.06
187	10.76	10.92	11.1	11.3	11.51	11.73	11.96	12.21	12.47	12.74	13.03
188	10.75	10.91	11.09	11.28	11.49	11.71	11.94	12.19	12.45	12.72	13
189	10.74	10.9	11.08	11.27	11.48	11.69	11.92	12.17	12.42	12.69	12.97
190	10.73	10.89	11.07	11.26	11.46	11.68	11.9	12.14	12.4	12.66	12.94
191	10.72	10.88	11.06	11.24	11.44	11.66	11.88	12.12	12.37	12.64	12.91
192	10.72	10.87	11.05	11.23	11.43	11.64	11.87	12.1	12.35	12.61	12.88
193	10.71	10.86	11.03	11.22	11.41	11.62	11.85	12.08	12.33	12.58	12.85
194	10.7	10.85	11.02	11.21	11.4	11.61	11.83	12.06	12.3	12.56	12.82
195	10.69	10.85	11.01	11.19	11.39	11.59	11.81	12.04	12.28	12.53	12.8
196	10.68	10.84	11	11.18	11.37	11.58	11.79	12.02	12.26	12.51	12.77
197	10.68	10.83	10.99	11.17	11.36	11.56	11.77	12	12.24	12.48	12.74
198	10.67	10.82	10.98	11.16	11.34	11.54	11.76	11.98	12.21	12.46	12.71
199	10.66	10.81	10.97	11.14	11.33	11.53	11.74	11.96	12.19	12.44	12.69
200	10.66	10.8	10.96	11.13	11.32	11.51	11.72	11.94	12.17	12.41	12.66
201	10.65	10.79	10.95	11.12	11.3	11.5	11.7	11.92	12.15	12.39	12.64
202	10.64	10.79	10.94	11.11	11.29	11.48	11.69	11.9	12.13	12.37	12.61
203	10.64	10.78	10.93	11.1	11.28	11.47	11.67	11.88	12.11	12.34	12.59
204	10.63	10.77	10.92	11.09	11.27	11.45	11.65	11.87	12.09	12.32	12.56
205	10.62	10.76	10.91	11.08	11.25	11.44	11.64	11.85	12.07	12.3	12.54
206	10.62	10.75	10.9	11.07	11.24	11.43	11.62	11.83	12.05	12.28	12.52
207	10.61	10.75	10.9	11.06	11.23	11.41	11.61	11.81	12.03	12.26	12.49
208	10.6	10.74	10.89	11.05	11.22	11.4	11.59	11.8	12.01	12.24	12.47
209	10.6	10.73	10.88	11.04	11.21	11.39	11.58	11.78	11.99	12.21	12.45
210	10.59	10.72	10.87	11.03	11.19	11.37	11.56	11.76	11.97	12.19	12.42
211	10.59	10.72	10.86	11.02	11.18	11.36	11.55	11.75	11.96	12.17	12.4
212	10.58	10.71	10.85	11.01	11.17	11.35	11.53	11.73	11.94	12.15	12.38
213	10.57	10.7	10.84	11	11.16	11.33	11.52	11.71	11.92	12.13	12.36
214	10.57	10.7	10.84	10.99	11.15	11.32	11.5	11.7	11.9	12.12	12.34
215	10.56	10.69	10.83	10.98	11.14	11.31	11.49	11.68	11.88	12.1	12.32
216	10.56	10.68	10.82	10.97	11.13	11.3	11.48	11.67	11.87	12.08	12.3
217	10.55	10.68	10.81	10.96	11.12	11.28	11.46	11.65	11.85	12.06	12.28
218	10.55	10.67	10.8	10.95	11.11	11.27	11.45	11.64	11.83	12.04	12.26
219	10.54	10.66	10.8	10.94	11.1	11.26	11.44	11.62	11.82	12.02	12.24
220	10.54	10.66	10.79	10.93	11.09	11.25	11.42	11.61	11.8	12	12.22
221	10.53	10.65	10.78	10.92	11.08	11.24	11.41	11.59	11.79	11.99	12.2
222	10.52	10.64	10.77	10.92	11.07	11.23	11.4	11.58	11.77	11.97	12.18
223	10.52	10.64	10.77	10.91	11.06	11.22	11.39	11.56	11.75	11.95	12.16
224	10.51	10.63	10.76	10.9	11.05	11.21	11.37	11.55	11.74	11.93	12.14

225	10.51	10.63	10.75	10.89	11.04	11.19	11.36	11.54	11.72	11.92	12.12
226	10.51	10.62	10.75	10.88	11.03	11.18	11.35	11.52	11.71	11.9	12.1
227	10.5	10.61	10.74	10.87	11.02	11.17	11.34	11.51	11.69	11.88	12.08
228	10.5	10.61	10.73	10.87	11.01	11.16	11.33	11.5	11.68	11.87	12.07
229	10.49	10.6	10.73	10.86	11	11.15	11.31	11.48	11.66	11.85	12.05
230	10.49	10.6	10.72	10.85	10.99	11.14	11.3	11.47	11.65	11.84	12.03
231	10.48	10.59	10.71	10.84	10.98	11.13	11.29	11.46	11.64	11.82	12.01
232	10.48	10.59	10.71	10.84	10.97	11.12	11.28	11.45	11.62	11.81	12
233	10.47	10.58	10.7	10.83	10.97	11.11	11.27	11.43	11.61	11.79	11.98
234	10.47	10.58	10.69	10.82	10.96	11.1	11.26	11.42	11.59	11.77	11.96
235	10.46	10.57	10.69	10.81	10.95	11.09	11.25	11.41	11.58	11.76	11.95
236	10.46	10.57	10.68	10.81	10.94	11.08	11.24	11.4	11.57	11.75	11.93
237	10.46	10.56	10.68	10.8	10.93	11.07	11.23	11.39	11.55	11.73	11.92
238	10.45	10.56	10.67	10.79	10.92	11.07	11.22	11.37	11.54	11.72	11.9
239	10.45	10.55	10.66	10.79	10.92	11.06	11.21	11.36	11.53	11.7	11.88
240	10.44	10.55	10.66	10.78	10.91	11.05	11.19	11.35	11.52	11.69	11.87
241	10.44	10.54	10.65	10.77	10.9	11.04	11.18	11.34	11.5	11.67	11.85
242	10.43	10.54	10.65	10.76	10.89	11.03	11.17	11.33	11.49	11.66	11.84
243	10.43	10.53	10.64	10.76	10.89	11.02	11.17	11.32	11.48	11.65	11.82
244	10.43	10.53	10.63	10.75	10.88	11.01	11.16	11.31	11.47	11.63	11.81
245	10.42	10.52	10.63	10.75	10.87	11	11.15	11.3	11.45	11.62	11.79
246	10.42	10.52	10.62	10.74	10.86	11	11.14	11.29	11.44	11.61	11.78
247	10.42	10.51	10.62	10.73	10.86	10.99	11.13	11.28	11.43	11.59	11.77
248	10.41	10.51	10.61	10.73	10.85	10.98	11.12	11.26	11.42	11.58	11.75
249	10.41	10.5	10.61	10.72	10.84	10.97	11.11	11.25	11.41	11.57	11.74
250	10.4	10.5	10.6	10.71	10.83	10.96	11.1	11.24	11.4	11.56	11.72
251	10.4	10.5	10.6	10.71	10.83	10.96	11.09	11.23	11.39	11.54	11.71
252	10.4	10.49	10.59	10.7	10.82	10.95	11.08	11.22	11.37	11.53	11.7
253	10.39	10.49	10.59	10.7	10.81	10.94	11.07	11.21	11.36	11.52	11.68
254	10.39	10.48	10.58	10.69	10.81	10.93	11.06	11.21	11.35	11.51	11.67
255	10.39	10.48	10.58	10.69	10.8	10.92	11.06	11.2	11.34	11.5	11.66
256	10.38	10.47	10.57	10.68	10.79	10.92	11.05	11.19	11.33	11.48	11.65
257	10.38	10.47	10.57	10.67	10.79	10.91	11.04	11.18	11.32	11.47	11.63
258	10.38	10.47	10.56	10.67	10.78	10.9	11.03	11.17	11.31	11.46	11.62
259	10.37	10.46	10.56	10.66	10.78	10.9	11.02	11.16	11.3	11.45	11.61
260	10.37	10.46	10.55	10.66	10.77	10.89	11.02	11.15	11.29	11.44	11.6
261	10.37	10.45	10.55	10.65	10.76	10.88	11.01	11.14	11.28	11.43	11.58
262	10.36	10.45	10.55	10.65	10.76	10.87	11	11.13	11.27	11.42	11.57
263	10.36	10.45	10.54	10.64	10.75	10.87	10.99	11.12	11.26	11.41	11.56
264	10.36	10.44	10.54	10.64	10.75	10.86	10.98	11.11	11.25	11.4	11.55
265	10.36	10.44	10.53	10.63	10.74	10.85	10.98	11.11	11.24	11.39	11.54
266	10.35	10.44	10.53	10.63	10.73	10.85	10.97	11.1	11.23	11.38	11.52
267	10.35	10.43	10.52	10.62	10.73	10.84	10.96	11.09	11.22	11.36	11.51
268	10.35	10.43	10.52	10.62	10.72	10.83	10.95	11.08	11.21	11.35	11.5

269	10.34	10.43	10.51	10.61	10.72	10.83	10.95	11.07	11.21	11.34	11.49
270	10.34	10.42	10.51	10.61	10.71	10.82	10.94	11.06	11.2	11.33	11.48
271	10.34	10.42	10.51	10.6	10.71	10.82	10.93	11.06	11.19	11.32	11.47
272	10.33	10.41	10.5	10.6	10.7	10.81	10.93	11.05	11.18	11.31	11.46
273	10.33	10.41	10.5	10.59	10.69	10.8	10.92	11.04	11.17	11.31	11.45
274	10.33	10.41	10.49	10.59	10.69	10.8	10.91	11.03	11.16	11.3	11.44
275	10.33	10.4	10.49	10.58	10.68	10.79	10.9	11.03	11.15	11.29	11.43
276	10.32	10.4	10.49	10.58	10.68	10.78	10.9	11.02	11.14	11.28	11.42
277	10.32	10.4	10.48	10.57	10.67	10.78	10.89	11.01	11.14	11.27	11.41
278	10.32	10.4	10.48	10.57	10.67	10.77	10.88	11	11.13	11.26	11.4
279	10.32	10.39	10.48	10.57	10.66	10.77	10.88	11	11.12	11.25	11.39
280	10.31	10.39	10.47	10.56	10.66	10.76	10.87	10.99	11.11	11.24	11.38
281	10.31	10.39	10.47	10.56	10.65	10.76	10.86	10.98	11.1	11.23	11.37
282	10.31	10.38	10.46	10.55	10.65	10.75	10.86	10.97	11.09	11.22	11.36
283	10.31	10.38	10.46	10.55	10.64	10.74	10.85	10.97	11.09	11.21	11.35
284	10.3	10.38	10.46	10.54	10.64	10.74	10.85	10.96	11.08	11.21	11.34
285	10.3	10.37	10.45	10.54	10.63	10.73	10.84	10.95	11.07	11.2	11.33
286	10.3	10.37	10.45	10.54	10.63	10.73	10.83	10.95	11.06	11.19	11.32
287	10.3	10.37	10.45	10.53	10.62	10.72	10.83	10.94	11.06	11.18	11.31
288	10.29	10.37	10.44	10.53	10.62	10.72	10.82	10.93	11.05	11.17	11.3
289	10.29	10.36	10.44	10.52	10.62	10.71	10.82	10.93	11.04	11.16	11.29
290	10.29	10.36	10.44	10.52	10.61	10.71	10.81	10.92	11.03	11.16	11.28
291	10.29	10.36	10.43	10.52	10.61	10.7	10.8	10.91	11.03	11.15	11.27
292	10.28	10.35	10.43	10.51	10.6	10.7	10.8	10.91	11.02	11.14	11.26
293	10.28	10.35	10.43	10.51	10.6	10.69	10.79	10.9	11.01	11.13	11.26
294	10.28	10.35	10.42	10.51	10.59	10.69	10.79	10.89	11.01	11.12	11.25
295	10.28	10.35	10.42	10.5	10.59	10.68	10.78	10.89	11	11.12	11.24
296	10.28	10.34	10.42	10.5	10.58	10.68	10.78	10.88	10.99	11.11	11.23
297	10.27	10.34	10.41	10.49	10.58	10.67	10.77	10.87	10.98	11.1	11.22
298	10.27	10.34	10.41	10.49	10.58	10.67	10.77	10.87	10.98	11.09	11.21
299	10.27	10.34	10.41	10.49	10.57	10.66	10.76	10.86	10.97	11.09	11.21
300	10.27	10.33	10.41	10.48	10.57	10.66	10.75	10.86	10.96	11.08	11.2
301	10.26	10.33	10.4	10.48	10.56	10.65	10.75	10.85	10.96	11.07	11.19
302	10.26	10.33	10.4	10.48	10.56	10.65	10.74	10.85	10.95	11.06	11.18
303	10.26	10.33	10.4	10.47	10.56	10.64	10.74	10.84	10.94	11.06	11.17
304	10.26	10.32	10.39	10.47	10.55	10.64	10.73	10.83	10.94	11.05	11.17
305	10.26	10.32	10.39	10.47	10.55	10.64	10.73	10.83	10.93	11.04	11.16
306	10.25	10.32	10.39	10.46	10.54	10.63	10.72	10.82	10.93	11.04	11.15
307	10.25	10.32	10.38	10.46	10.54	10.63	10.72	10.82	10.92	11.03	11.14
308	10.25	10.31	10.38	10.46	10.54	10.62	10.71	10.81	10.91	11.02	11.13
309	10.25	10.31	10.38	10.45	10.53	10.62	10.71	10.81	10.91	11.01	11.13
310	10.25	10.31	10.38	10.45	10.53	10.61	10.7	10.8	10.9	11.01	11.12
311	10.24	10.31	10.37	10.45	10.53	10.61	10.7	10.79	10.9	11	11.11
312	10.24	10.3	10.37	10.44	10.52	10.61	10.69	10.79	10.89	10.99	11.11

313	10.24	10.3	10.37	10.44	10.52	10.6	10.69	10.78	10.88	10.99	11.1
314	10.24	10.3	10.37	10.44	10.51	10.6	10.69	10.78	10.88	10.98	11.09
315	10.24	10.3	10.36	10.43	10.51	10.59	10.68	10.77	10.87	10.98	11.08
316	10.24	10.3	10.36	10.43	10.51	10.59	10.68	10.77	10.87	10.97	11.08
317	10.23	10.29	10.36	10.43	10.5	10.59	10.67	10.76	10.86	10.96	11.07
318	10.23	10.29	10.36	10.43	10.5	10.58	10.67	10.76	10.86	10.96	11.06
319	10.23	10.29	10.35	10.42	10.5	10.58	10.66	10.75	10.85	10.95	11.06
320	10.23	10.29	10.35	10.42	10.49	10.57	10.66	10.75	10.84	10.94	11.05
321	10.23	10.28	10.35	10.42	10.49	10.57	10.65	10.74	10.84	10.94	11.04
322	10.22	10.28	10.35	10.41	10.49	10.57	10.65	10.74	10.83	10.93	11.04
323	10.22	10.28	10.34	10.41	10.48	10.56	10.65	10.73	10.83	10.93	11.03
324	10.22	10.28	10.34	10.41	10.48	10.56	10.64	10.73	10.82	10.92	11.02
325	10.22	10.28	10.34	10.41	10.48	10.56	10.64	10.72	10.82	10.91	11.02
326	10.22	10.27	10.34	10.4	10.47	10.55	10.63	10.72	10.81	10.91	11.01
327	10.22	10.27	10.33	10.4	10.47	10.55	10.63	10.72	10.81	10.9	11
328	10.22	10.27	10.33	10.4	10.47	10.54	10.63	10.71	10.8	10.9	11
329	10.21	10.27	10.33	10.39	10.47	10.54	10.62	10.71	10.8	10.89	10.99
330	10.21	10.27	10.33	10.39	10.46	10.54	10.62	10.7	10.79	10.89	10.99
331	10.21	10.26	10.32	10.39	10.46	10.53	10.61	10.7	10.79	10.88	10.98
332	10.21	10.26	10.32	10.39	10.46	10.53	10.61	10.69	10.78	10.88	10.97
333	10.21	10.26	10.32	10.38	10.45	10.53	10.61	10.69	10.78	10.87	10.97
334	10.21	10.26	10.32	10.38	10.45	10.52	10.6	10.68	10.77	10.86	10.96
335	10.2	10.26	10.32	10.38	10.45	10.52	10.6	10.68	10.77	10.86	10.96
336	10.2	10.26	10.31	10.38	10.44	10.52	10.59	10.68	10.76	10.85	10.95
337	10.2	10.25	10.31	10.37	10.44	10.51	10.59	10.67	10.76	10.85	10.94
338	10.2	10.25	10.31	10.37	10.44	10.51	10.59	10.67	10.75	10.84	10.94
339	10.2	10.25	10.31	10.37	10.44	10.51	10.58	10.66	10.75	10.84	10.93
340	10.2	10.25	10.31	10.37	10.43	10.5	10.58	10.66	10.74	10.83	10.93
341	10.2	10.25	10.3	10.36	10.43	10.5	10.58	10.65	10.74	10.83	10.92
342	10.19	10.25	10.3	10.36	10.43	10.5	10.57	10.65	10.73	10.82	10.92
343	10.19	10.24	10.3	10.36	10.42	10.49	10.57	10.65	10.73	10.82	10.91
344	10.19	10.24	10.3	10.36	10.42	10.49	10.56	10.64	10.73	10.81	10.9
345	10.19	10.24	10.29	10.35	10.42	10.49	10.56	10.64	10.72	10.81	10.9
346	10.19	10.24	10.29	10.35	10.42	10.48	10.56	10.63	10.72	10.8	10.89
347	10.19	10.24	10.29	10.35	10.41	10.48	10.55	10.63	10.71	10.8	10.89
348	10.19	10.23	10.29	10.35	10.41	10.48	10.55	10.63	10.71	10.79	10.88
349	10.18	10.23	10.29	10.35	10.41	10.48	10.55	10.62	10.7	10.79	10.88
350	10.18	10.23	10.29	10.34	10.41	10.47	10.54	10.62	10.7	10.78	10.87
351	10.18	10.23	10.28	10.34	10.4	10.47	10.54	10.62	10.7	10.78	10.87
352	10.18	10.23	10.28	10.34	10.4	10.47	10.54	10.61	10.69	10.77	10.86
353	10.18	10.23	10.28	10.34	10.4	10.46	10.53	10.61	10.69	10.77	10.86
354	10.18	10.23	10.28	10.33	10.4	10.46	10.53	10.6	10.68	10.77	10.85
355	10.18	10.22	10.28	10.33	10.39	10.46	10.53	10.6	10.68	10.76	10.85
356	10.18	10.22	10.27	10.33	10.39	10.46	10.52	10.6	10.68	10.76	10.84

357	10.17	10.22	10.27	10.33	10.39	10.45	10.52	10.59	10.67	10.75	10.84
358	10.17	10.22	10.27	10.33	10.39	10.45	10.52	10.59	10.67	10.75	10.83
359	10.17	10.22	10.27	10.32	10.38	10.45	10.51	10.59	10.66	10.74	10.83
360	10.17	10.22	10.27	10.32	10.38	10.44	10.51	10.58	10.66	10.74	10.82
361	10.17	10.21	10.27	10.32	10.38	10.44	10.51	10.58	10.66	10.73	10.82
362	10.17	10.21	10.26	10.32	10.38	10.44	10.51	10.58	10.65	10.73	10.81
363	10.17	10.21	10.26	10.32	10.37	10.44	10.5	10.57	10.65	10.73	10.81
364	10.17	10.21	10.26	10.31	10.37	10.43	10.5	10.57	10.64	10.72	10.8
365	10.16	10.21	10.26	10.31	10.37	10.43	10.5	10.57	10.64	10.72	10.8
366	10.16	10.21	10.26	10.31	10.37	10.43	10.49	10.56	10.64	10.71	10.8
367	10.16	10.21	10.25	10.31	10.36	10.43	10.49	10.56	10.63	10.71	10.79
368	10.16	10.2	10.25	10.31	10.36	10.42	10.49	10.56	10.63	10.71	10.79
369	10.16	10.2	10.25	10.3	10.36	10.42	10.48	10.55	10.63	10.7	10.78
370	10.16	10.2	10.25	10.3	10.36	10.42	10.48	10.55	10.62	10.7	10.78
371	10.16	10.2	10.25	10.3	10.36	10.42	10.48	10.55	10.62	10.69	10.77
372	10.16	10.2	10.25	10.3	10.35	10.41	10.48	10.54	10.61	10.69	10.77
373	10.16	10.2	10.25	10.3	10.35	10.41	10.47	10.54	10.61	10.69	10.76
374	10.15	10.2	10.24	10.29	10.35	10.41	10.47	10.54	10.61	10.68	10.76
375	10.15	10.2	10.24	10.29	10.35	10.41	10.47	10.53	10.6	10.68	10.76
376	10.15	10.19	10.24	10.29	10.35	10.4	10.47	10.53	10.6	10.67	10.75
377	10.15	10.19	10.24	10.29	10.34	10.4	10.46	10.53	10.6	10.67	10.75
378	10.15	10.19	10.24	10.29	10.34	10.4	10.46	10.53	10.59	10.67	10.74
379	10.15	10.19	10.24	10.29	10.34	10.4	10.46	10.52	10.59	10.66	10.74
380	10.15	10.19	10.23	10.28	10.34	10.39	10.45	10.52	10.59	10.66	10.74
381	10.15	10.19	10.23	10.28	10.34	10.39	10.45	10.52	10.58	10.66	10.73
382	10.15	10.19	10.23	10.28	10.33	10.39	10.45	10.51	10.58	10.65	10.73
383	10.14	10.19	10.23	10.28	10.33	10.39	10.45	10.51	10.58	10.65	10.72
384	10.14	10.18	10.23	10.28	10.33	10.38	10.44	10.51	10.57	10.64	10.72
385	10.14	10.18	10.23	10.28	10.33	10.38	10.44	10.5	10.57	10.64	10.72
386	10.14	10.18	10.23	10.27	10.33	10.38	10.44	10.5	10.57	10.64	10.71
387	10.14	10.18	10.22	10.27	10.32	10.38	10.44	10.5	10.56	10.63	10.71
388	10.14	10.18	10.22	10.27	10.32	10.38	10.43	10.5	10.56	10.63	10.7
389	10.14	10.18	10.22	10.27	10.32	10.37	10.43	10.49	10.56	10.63	10.7
390	10.14	10.18	10.22	10.27	10.32	10.37	10.43	10.49	10.56	10.62	10.7
391	10.14	10.18	10.22	10.27	10.32	10.37	10.43	10.49	10.55	10.62	10.69
392	10.14	10.17	10.22	10.26	10.31	10.37	10.42	10.49	10.55	10.62	10.69
393	10.13	10.17	10.22	10.26	10.31	10.37	10.42	10.48	10.55	10.61	10.68
394	10.13	10.17	10.21	10.26	10.31	10.36	10.42	10.48	10.54	10.61	10.68
395	10.13	10.17	10.21	10.26	10.31	10.36	10.42	10.48	10.54	10.61	10.68
396	10.13	10.17	10.21	10.26	10.31	10.36	10.42	10.47	10.54	10.6	10.67
397	10.13	10.17	10.21	10.26	10.3	10.36	10.41	10.47	10.53	10.6	10.67
398	10.13	10.17	10.21	10.25	10.3	10.36	10.41	10.47	10.53	10.6	10.67
399	10.13	10.17	10.21	10.25	10.3	10.35	10.41	10.47	10.53	10.59	10.66
400	10.13	10.17	10.21	10.25	10.3	10.35	10.41	10.46	10.53	10.59	10.66



附件 7：

曲线行驶过程中最高点的 y 坐标与方向盘转速和无人车行驶速度的关系表

	10	11	12	13	14	15	16	17	18	19	20
100	11.4	11.96	12.51	13.04	13.55	14.03	14.49	14.93	15.34	15.73	16.1
101	11.34	11.9	12.45	12.97	13.48	13.96	14.42	14.86	15.27	15.66	16.03
102	11.28	11.84	12.39	12.91	13.41	13.89	14.35	14.79	15.2	15.59	15.96
103	11.23	11.79	12.32	12.84	13.34	13.82	14.28	14.72	15.13	15.52	15.89
104	11.18	11.73	12.26	12.78	13.28	13.76	14.21	14.65	15.06	15.45	15.82
105	11.12	11.67	12.2	12.72	13.21	13.69	14.14	14.58	14.99	15.38	15.76
106	11.07	11.62	12.15	12.66	13.15	13.62	14.08	14.51	14.92	15.32	15.69
107	11.02	11.56	12.09	12.6	13.09	13.56	14.01	14.44	14.86	15.25	15.62
108	10.97	11.51	12.03	12.54	13.03	13.5	13.95	14.38	14.79	15.18	15.55
109	10.92	11.45	11.97	12.48	12.96	13.43	13.88	14.31	14.72	15.12	15.49
110	10.87	11.4	11.92	12.42	12.9	13.37	13.82	14.25	14.66	15.05	15.42
111	10.82	11.35	11.86	12.36	12.84	13.31	13.76	14.18	14.59	14.99	15.36
112	10.77	11.3	11.81	12.31	12.79	13.25	13.69	14.12	14.53	14.92	15.29
113	10.73	11.25	11.76	12.25	12.73	13.19	13.63	14.06	14.47	14.86	15.23
114	10.68	11.2	11.7	12.2	12.67	13.13	13.57	14	14.4	14.79	15.17
115	10.64	11.15	11.65	12.14	12.61	13.07	13.51	13.94	14.34	14.73	15.1
116	10.59	11.1	11.6	12.09	12.56	13.01	13.45	13.87	14.28	14.67	15.04
117	10.55	11.06	11.55	12.04	12.5	12.96	13.39	13.82	14.22	14.61	14.98
118	10.5	11.01	11.5	11.98	12.45	12.9	13.34	13.76	14.16	14.55	14.92
119	10.46	10.96	11.45	11.93	12.4	12.84	13.28	13.7	14.1	14.49	14.86
120	10.42	10.92	11.41	11.88	12.34	12.79	13.22	13.64	14.04	14.43	14.8
121	10.38	10.87	11.36	11.83	12.29	12.74	13.17	13.58	13.98	14.37	14.74
122	10.34	10.83	11.31	11.78	12.24	12.68	13.11	13.53	13.93	14.31	14.68
123	10.3	10.79	11.27	11.73	12.19	12.63	13.06	13.47	13.87	14.25	14.62
124	10.26	10.74	11.22	11.69	12.14	12.58	13	13.41	13.81	14.19	14.56

125	10.22	10.7	11.18	11.64	12.09	12.53	12.95	13.36	13.76	14.14	14.5
126	10.18	10.66	11.13	11.59	12.04	12.47	12.9	13.31	13.7	14.08	14.45
127	10.14	10.62	11.09	11.54	11.99	12.42	12.84	13.25	13.65	14.03	14.39
128	10.1	10.58	11.04	11.5	11.94	12.37	12.79	13.2	13.59	13.97	14.34
129	10.07	10.54	11	11.45	11.9	12.32	12.74	13.15	13.54	13.92	14.28
130	10.03	10.5	10.96	11.41	11.85	12.28	12.69	13.1	13.49	13.86	14.23
131	9.993	10.46	10.92	11.37	11.8	12.23	12.64	13.04	13.43	13.81	14.17
132	9.958	10.42	10.88	11.32	11.76	12.18	12.59	12.99	13.38	13.76	14.12
133	9.923	10.38	10.84	11.28	11.71	12.13	12.54	12.94	13.33	13.7	14.06
134	9.888	10.35	10.8	11.24	11.67	12.09	12.5	12.89	13.28	13.65	14.01
135	9.854	10.31	10.76	11.2	11.62	12.04	12.45	12.84	13.23	13.6	13.96
136	9.821	10.27	10.72	11.15	11.58	12	12.4	12.8	13.18	13.55	13.91
137	9.787	10.24	10.68	11.11	11.54	11.95	12.36	12.75	13.13	13.5	13.86
138	9.755	10.2	10.64	11.07	11.5	11.91	12.31	12.7	13.08	13.45	13.81
139	9.722	10.17	10.61	11.03	11.45	11.86	12.26	12.65	13.03	13.4	13.76
140	9.69	10.13	10.57	11	11.41	11.82	12.22	12.61	12.98	13.35	13.71
141	9.659	10.1	10.53	10.96	11.37	11.78	12.18	12.56	12.94	13.3	13.66
142	9.628	10.07	10.5	10.92	11.33	11.74	12.13	12.52	12.89	13.25	13.61
143	9.597	10.03	10.46	10.88	11.29	11.69	12.09	12.47	12.84	13.21	13.56
144	9.567	10	10.43	10.84	11.25	11.65	12.04	12.43	12.8	13.16	13.51
145	9.537	9.967	10.39	10.81	11.21	11.61	12	12.38	12.75	13.11	13.46
146	9.508	9.936	10.36	10.77	11.18	11.57	11.96	12.34	12.71	13.07	13.42
147	9.478	9.904	10.32	10.73	11.14	11.53	11.92	12.3	12.66	13.02	13.37
148	9.45	9.873	10.29	10.7	11.1	11.49	11.88	12.25	12.62	12.98	13.32
149	9.421	9.842	10.26	10.66	11.06	11.45	11.84	12.21	12.58	12.93	13.28
150	9.393	9.812	10.22	10.63	11.03	11.42	11.8	12.17	12.53	12.89	13.23
151	9.365	9.781	10.19	10.59	10.99	11.38	11.76	12.13	12.49	12.84	13.19
152	9.338	9.752	10.16	10.56	10.95	11.34	11.72	12.09	12.45	12.8	13.14
153	9.311	9.722	10.13	10.53	10.92	11.3	11.68	12.05	12.41	12.76	13.1
154	9.284	9.693	10.1	10.49	10.88	11.27	11.64	12.01	12.37	12.72	13.06
155	9.258	9.665	10.07	10.46	10.85	11.23	11.6	11.97	12.32	12.67	13.01
156	9.232	9.636	10.04	10.43	10.81	11.19	11.56	11.93	12.28	12.63	12.97
157	9.206	9.608	10.01	10.4	10.78	11.16	11.53	11.89	12.24	12.59	12.93
158	9.18	9.581	9.976	10.36	10.75	11.12	11.49	11.85	12.2	12.55	12.89
159	9.155	9.553	9.946	10.33	10.71	11.09	11.45	11.81	12.17	12.51	12.84
160	9.13	9.526	9.917	10.3	10.68	11.05	11.42	11.78	12.13	12.47	12.8
161	9.106	9.5	9.888	10.27	10.65	11.02	11.38	11.74	12.09	12.43	12.76
162	9.081	9.473	9.86	10.24	10.62	10.98	11.35	11.7	12.05	12.39	12.72
163	9.057	9.447	9.832	10.21	10.58	10.95	11.31	11.67	12.01	12.35	12.68
164	9.033	9.421	9.804	10.18	10.55	10.92	11.28	11.63	11.97	12.31	12.64
165	9.01	9.396	9.776	10.15	10.52	10.89	11.24	11.59	11.94	12.27	12.6
166	8.987	9.37	9.749	10.12	10.49	10.85	11.21	11.56	11.9	12.24	12.56
167	8.964	9.345	9.722	10.09	10.46	10.82	11.18	11.52	11.86	12.2	12.53
168	8.941	9.321	9.696	10.07	10.43	10.79	11.14	11.49	11.83	12.16	12.49

169	8.918	9.296	9.669	10.04	10.4	10.76	11.11	11.45	11.79	12.12	12.45
170	8.896	9.272	9.643	10.01	10.37	10.73	11.08	11.42	11.76	12.09	12.41
171	8.874	9.248	9.618	9.982	10.34	10.7	11.04	11.39	11.72	12.05	12.37
172	8.853	9.225	9.592	9.955	10.31	10.67	11.01	11.35	11.69	12.02	12.34
173	8.831	9.201	9.567	9.928	10.28	10.64	10.98	11.32	11.65	11.98	12.3
174	8.81	9.178	9.542	9.902	10.26	10.61	10.95	11.29	11.62	11.95	12.26
175	8.789	9.155	9.517	9.875	10.23	10.58	10.92	11.25	11.59	11.91	12.23
176	8.768	9.132	9.493	9.849	10.2	10.55	10.89	11.22	11.55	11.88	12.19
177	8.747	9.11	9.469	9.823	10.17	10.52	10.86	11.19	11.52	11.84	12.16
178	8.727	9.088	9.445	9.798	10.15	10.49	10.83	11.16	11.49	11.81	12.12
179	8.707	9.066	9.421	9.772	10.12	10.46	10.8	11.13	11.45	11.77	12.09
180	8.687	9.044	9.398	9.747	10.09	10.43	10.77	11.1	11.42	11.74	12.05
181	8.667	9.023	9.375	9.722	10.07	10.4	10.74	11.07	11.39	11.71	12.02
182	8.647	9.001	9.352	9.698	10.04	10.38	10.71	11.04	11.36	11.68	11.99
183	8.628	8.98	9.329	9.673	10.01	10.35	10.68	11.01	11.33	11.64	11.95
184	8.609	8.96	9.306	9.649	9.988	10.32	10.65	10.98	11.3	11.61	11.92
185	8.59	8.939	9.284	9.626	9.963	10.3	10.62	10.95	11.27	11.58	11.89
186	8.571	8.918	9.262	9.602	9.938	10.27	10.6	10.92	11.24	11.55	11.85
187	8.553	8.898	9.24	9.579	9.913	10.24	10.57	10.89	11.21	11.52	11.82
188	8.534	8.878	9.219	9.555	9.888	10.22	10.54	10.86	11.18	11.48	11.79
189	8.516	8.858	9.197	9.533	9.864	10.19	10.51	10.83	11.15	11.45	11.76
190	8.498	8.839	9.176	9.51	9.84	10.17	10.49	10.8	11.12	11.42	11.73
191	8.48	8.819	9.155	9.487	9.816	10.14	10.46	10.78	11.09	11.39	11.69
192	8.463	8.8	9.134	9.465	9.792	10.12	10.43	10.75	11.06	11.36	11.66
193	8.445	8.781	9.114	9.443	9.769	10.09	10.41	10.72	11.03	11.33	11.63
194	8.428	8.762	9.093	9.421	9.745	10.07	10.38	10.69	11	11.3	11.6
195	8.411	8.744	9.073	9.4	9.722	10.04	10.36	10.67	10.97	11.28	11.57
196	8.394	8.725	9.053	9.378	9.7	10.02	10.33	10.64	10.95	11.25	11.54
197	8.377	8.707	9.033	9.357	9.677	9.993	10.31	10.61	10.92	11.22	11.51
198	8.36	8.689	9.014	9.336	9.655	9.97	10.28	10.59	10.89	11.19	11.48
199	8.344	8.671	8.994	9.315	9.632	9.946	10.26	10.56	10.86	11.16	11.45
200	8.328	8.653	8.975	9.294	9.61	9.923	10.23	10.54	10.84	11.13	11.43
201	8.311	8.635	8.956	9.274	9.589	9.9	10.21	10.51	10.81	11.11	11.4
202	8.295	8.618	8.937	9.254	9.567	9.877	10.18	10.49	10.78	11.08	11.37
203	8.28	8.6	8.918	9.234	9.546	9.854	10.16	10.46	10.76	11.05	11.34
204	8.264	8.583	8.9	9.214	9.524	9.832	10.14	10.44	10.73	11.02	11.31
205	8.248	8.566	8.882	9.194	9.503	9.809	10.11	10.41	10.71	11	11.28
206	8.233	8.549	8.863	9.174	9.483	9.787	10.09	10.39	10.68	10.97	11.26
207	8.218	8.533	8.845	9.155	9.462	9.766	10.07	10.36	10.66	10.94	11.23
208	8.202	8.516	8.827	9.136	9.441	9.744	10.04	10.34	10.63	10.92	11.2
209	8.187	8.5	8.81	9.117	9.421	9.722	10.02	10.31	10.61	10.89	11.18
210	8.173	8.484	8.792	9.098	9.401	9.701	9.998	10.29	10.58	10.87	11.15
211	8.158	8.468	8.775	9.079	9.381	9.68	9.976	10.27	10.56	10.84	11.12
212	8.143	8.452	8.758	9.061	9.361	9.659	9.953	10.24	10.53	10.82	11.1

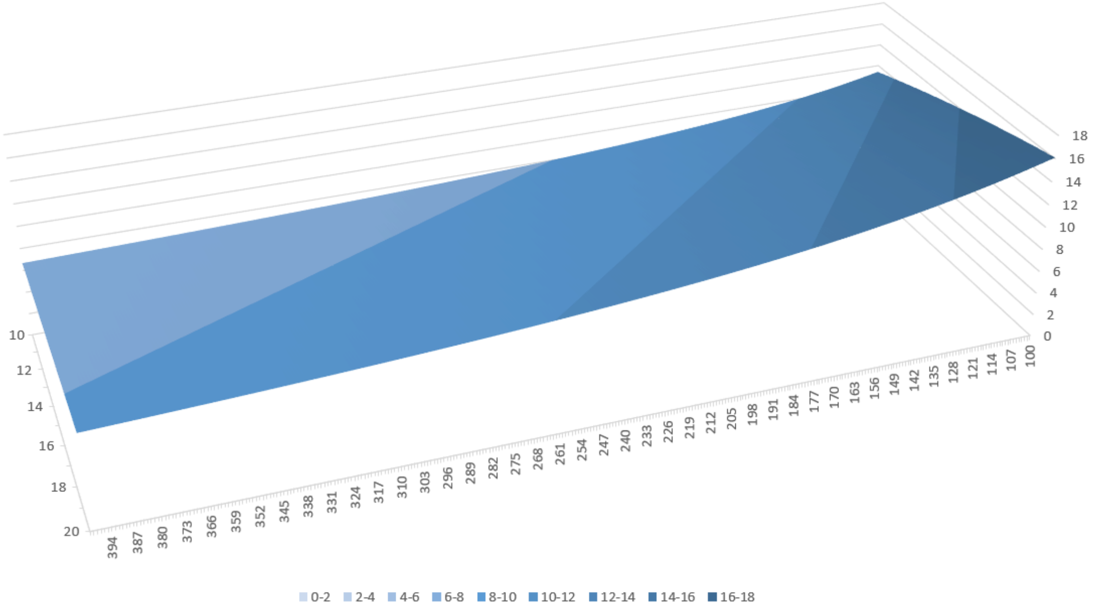
213	8.129	8.436	8.74	9.043	9.342	9.638	9.932	10.22	10.51	10.79	11.07
214	8.114	8.42	8.724	9.024	9.322	9.618	9.91	10.2	10.48	10.77	11.04
215	8.1	8.405	8.707	9.006	9.303	9.597	9.888	10.18	10.46	10.74	11.02
216	8.086	8.389	8.69	8.988	9.284	9.577	9.867	10.15	10.44	10.72	10.99
217	8.072	8.374	8.674	8.971	9.265	9.557	9.846	10.13	10.41	10.69	10.97
218	8.058	8.359	8.657	8.953	9.247	9.537	9.825	10.11	10.39	10.67	10.94
219	8.045	8.344	8.641	8.936	9.228	9.517	9.804	10.09	10.37	10.64	10.92
220	8.031	8.329	8.625	8.918	9.209	9.498	9.783	10.07	10.35	10.62	10.89
221	8.018	8.314	8.609	8.901	9.191	9.478	9.763	10.04	10.32	10.6	10.87
222	8.004	8.3	8.593	8.884	9.173	9.459	9.743	10.02	10.3	10.57	10.85
223	7.991	8.285	8.578	8.868	9.155	9.44	9.722	10	10.28	10.55	10.82
224	7.978	8.271	8.562	8.851	9.137	9.421	9.702	9.981	10.26	10.53	10.8
225	7.965	8.257	8.547	8.834	9.12	9.402	9.683	9.96	10.23	10.51	10.77
226	7.952	8.243	8.531	8.818	9.102	9.384	9.663	9.939	10.21	10.48	10.75
227	7.939	8.229	8.516	8.802	9.085	9.365	9.643	9.919	10.19	10.46	10.73
228	7.926	8.215	8.501	8.785	9.067	9.347	9.624	9.898	10.17	10.44	10.7
229	7.914	8.201	8.486	8.769	9.05	9.329	9.605	9.878	10.15	10.42	10.68
230	7.901	8.187	8.472	8.754	9.033	9.311	9.586	9.858	10.13	10.39	10.66
231	7.889	8.174	8.457	8.738	9.017	9.293	9.567	9.838	10.11	10.37	10.64
232	7.877	8.161	8.442	8.722	9	9.275	9.548	9.819	10.09	10.35	10.61
233	7.865	8.147	8.428	8.707	8.983	9.258	9.53	9.799	10.07	10.33	10.59
234	7.853	8.134	8.414	8.691	8.967	9.24	9.511	9.78	10.05	10.31	10.57
235	7.841	8.121	8.399	8.676	8.951	9.223	9.493	9.76	10.03	10.29	10.55
236	7.829	8.108	8.385	8.661	8.935	9.206	9.475	9.741	10.01	10.27	10.52
237	7.817	8.095	8.371	8.646	8.918	9.189	9.457	9.722	9.985	10.25	10.5
238	7.805	8.082	8.358	8.631	8.903	9.172	9.439	9.704	9.966	10.23	10.48
239	7.794	8.07	8.344	8.616	8.887	9.155	9.421	9.685	9.946	10.2	10.46
240	7.782	8.057	8.33	8.602	8.871	9.138	9.404	9.666	9.927	10.18	10.44
241	7.771	8.045	8.317	8.587	8.856	9.122	9.386	9.648	9.907	10.16	10.42
242	7.76	8.032	8.303	8.573	8.84	9.106	9.369	9.63	9.888	10.14	10.4
243	7.748	8.02	8.29	8.558	8.825	9.089	9.352	9.612	9.869	10.12	10.38
244	7.737	8.008	8.277	8.544	8.81	9.073	9.335	9.594	9.85	10.1	10.36
245	7.726	7.996	8.264	8.53	8.795	9.057	9.318	9.576	9.832	10.09	10.34
246	7.715	7.984	8.251	8.516	8.78	9.041	9.301	9.558	9.813	10.07	10.32
247	7.704	7.972	8.238	8.502	8.765	9.026	9.284	9.541	9.795	10.05	10.3
248	7.694	7.96	8.225	8.489	8.75	9.01	9.268	9.523	9.776	10.03	10.28
249	7.683	7.948	8.213	8.475	8.736	8.994	9.251	9.506	9.758	10.01	10.26
250	7.672	7.937	8.2	8.461	8.721	8.979	9.235	9.489	9.74	9.99	10.24
251	7.662	7.925	8.187	8.448	8.707	8.964	9.219	9.472	9.722	9.971	10.22
252	7.651	7.914	8.175	8.435	8.692	8.948	9.203	9.455	9.705	9.952	10.2
253	7.641	7.903	8.163	8.421	8.678	8.933	9.187	9.438	9.687	9.934	10.18
254	7.631	7.891	8.151	8.408	8.664	8.918	9.171	9.421	9.669	9.916	10.16
255	7.62	7.88	8.138	8.395	8.65	8.904	9.155	9.405	9.652	9.897	10.14
256	7.61	7.869	8.126	8.382	8.636	8.889	9.14	9.388	9.635	9.879	10.12

257	7.6	7.858	8.114	8.369	8.623	8.874	9.124	9.372	9.618	9.861	10.1
258	7.59	7.847	8.103	8.357	8.609	8.86	9.109	9.356	9.601	9.844	10.08
259	7.58	7.836	8.091	8.344	8.595	8.845	9.093	9.34	9.584	9.826	10.07
260	7.571	7.826	8.079	8.331	8.582	8.831	9.078	9.324	9.567	9.808	10.05
261	7.561	7.815	8.068	8.319	8.569	8.817	9.063	9.308	9.55	9.791	10.03
262	7.551	7.804	8.056	8.307	8.555	8.803	9.048	9.292	9.534	9.774	10.01
263	7.542	7.794	8.045	8.294	8.542	8.789	9.033	9.276	9.517	9.756	9.993
264	7.532	7.783	8.033	8.282	8.529	8.775	9.019	9.261	9.501	9.739	9.976
265	7.523	7.773	8.022	8.27	8.516	8.761	9.004	9.245	9.485	9.722	9.958
266	7.513	7.763	8.011	8.258	8.503	8.747	8.99	9.23	9.469	9.706	9.94
267	7.504	7.752	8	8.246	8.491	8.734	8.975	9.215	9.453	9.689	9.923
268	7.495	7.742	7.989	8.234	8.478	8.72	8.961	9.2	9.437	9.672	9.905
269	7.485	7.732	7.978	8.222	8.465	8.707	8.947	9.185	9.421	9.656	9.888
270	7.476	7.722	7.967	8.211	8.453	8.693	8.932	9.17	9.406	9.639	9.871
271	7.467	7.712	7.956	8.199	8.44	8.68	8.918	9.155	9.39	9.623	9.854
272	7.458	7.702	7.946	8.187	8.428	8.667	8.905	9.14	9.375	9.607	9.837
273	7.449	7.693	7.935	8.176	8.416	8.654	8.891	9.126	9.359	9.591	9.821
274	7.44	7.683	7.924	8.165	8.404	8.641	8.877	9.111	9.344	9.575	9.804
275	7.432	7.673	7.914	8.153	8.391	8.628	8.863	9.097	9.329	9.559	9.787
276	7.423	7.664	7.904	8.142	8.379	8.615	8.85	9.083	9.314	9.543	9.771
277	7.414	7.654	7.893	8.131	8.368	8.603	8.836	9.068	9.299	9.528	9.755
278	7.405	7.645	7.883	8.12	8.356	8.59	8.823	9.054	9.284	9.512	9.738
279	7.397	7.635	7.873	8.109	8.344	8.578	8.81	9.04	9.269	9.497	9.722
280	7.388	7.626	7.863	8.098	8.332	8.565	8.797	9.026	9.255	9.481	9.706
281	7.38	7.617	7.853	8.087	8.321	8.553	8.783	9.013	9.24	9.466	9.69
282	7.372	7.608	7.843	8.076	8.309	8.54	8.77	8.999	9.226	9.451	9.675
283	7.363	7.598	7.833	8.066	8.298	8.528	8.758	8.985	9.212	9.436	9.659
284	7.355	7.589	7.823	8.055	8.286	8.516	8.745	8.972	9.197	9.421	9.643
285	7.347	7.58	7.813	8.045	8.275	8.504	8.732	8.958	9.183	9.406	9.628
286	7.339	7.571	7.803	8.034	8.264	8.492	8.719	8.945	9.169	9.392	9.613
287	7.33	7.563	7.794	8.024	8.253	8.48	8.707	8.932	9.155	9.377	9.597
288	7.322	7.554	7.784	8.013	8.242	8.469	8.694	8.918	9.141	9.362	9.582
289	7.314	7.545	7.775	8.003	8.231	8.457	8.682	8.905	9.127	9.348	9.567
290	7.306	7.536	7.765	7.993	8.22	8.445	8.669	8.892	9.114	9.334	9.552
291	7.299	7.528	7.756	7.983	8.209	8.434	8.657	8.879	9.1	9.319	9.537
292	7.291	7.519	7.746	7.973	8.198	8.422	8.645	8.867	9.087	9.305	9.522
293	7.283	7.511	7.737	7.963	8.187	8.411	8.633	8.854	9.073	9.291	9.508
294	7.275	7.502	7.728	7.953	8.177	8.399	8.621	8.841	9.06	9.277	9.493
295	7.268	7.494	7.719	7.943	8.166	8.388	8.609	8.828	9.047	9.263	9.478
296	7.26	7.485	7.71	7.933	8.156	8.377	8.597	8.816	9.033	9.249	9.464
297	7.252	7.477	7.701	7.924	8.145	8.366	8.585	8.804	9.02	9.236	9.45
298	7.245	7.469	7.692	7.914	8.135	8.355	8.574	8.791	9.007	9.222	9.435
299	7.237	7.461	7.683	7.904	8.125	8.344	8.562	8.779	8.994	9.209	9.421
300	7.23	7.452	7.674	7.895	8.114	8.333	8.55	8.767	8.982	9.195	9.407

301	7.223	7.444	7.665	7.885	8.104	8.322	8.539	8.755	8.969	9.182	9.393
302	7.215	7.436	7.657	7.876	8.094	8.311	8.528	8.742	8.956	9.168	9.379
303	7.208	7.428	7.648	7.867	8.084	8.301	8.516	8.73	8.943	9.155	9.365
304	7.201	7.42	7.639	7.857	8.074	8.29	8.505	8.719	8.931	9.142	9.352
305	7.194	7.413	7.631	7.848	8.064	8.28	8.494	8.707	8.918	9.129	9.338
306	7.186	7.405	7.622	7.839	8.054	8.269	8.483	8.695	8.906	9.116	9.324
307	7.179	7.397	7.614	7.83	8.045	8.259	8.472	8.683	8.894	9.103	9.311
308	7.172	7.389	7.605	7.821	8.035	8.248	8.461	8.672	8.882	9.09	9.297
309	7.165	7.381	7.597	7.812	8.025	8.238	8.45	8.66	8.869	9.077	9.284
310	7.158	7.374	7.589	7.803	8.016	8.228	8.439	8.649	8.857	9.065	9.271
311	7.151	7.366	7.58	7.794	8.006	8.218	8.428	8.637	8.845	9.052	9.258
312	7.144	7.359	7.572	7.785	7.997	8.207	8.417	8.626	8.833	9.04	9.245
313	7.138	7.351	7.564	7.776	7.987	8.197	8.407	8.615	8.822	9.027	9.232
314	7.131	7.344	7.556	7.767	7.978	8.187	8.396	8.603	8.81	9.015	9.219
315	7.124	7.336	7.548	7.759	7.969	8.178	8.385	8.592	8.798	9.003	9.206
316	7.117	7.329	7.54	7.75	7.959	8.168	8.375	8.581	8.786	8.99	9.193
317	7.111	7.322	7.532	7.742	7.95	8.158	8.365	8.57	8.775	8.978	9.18
318	7.104	7.314	7.524	7.733	7.941	8.148	8.354	8.559	8.763	8.966	9.168
319	7.097	7.307	7.516	7.725	7.932	8.138	8.344	8.548	8.752	8.954	9.155
320	7.091	7.3	7.508	7.716	7.923	8.129	8.334	8.538	8.74	8.942	9.143
321	7.084	7.293	7.501	7.708	7.914	8.119	8.324	8.527	8.729	8.93	9.13
322	7.078	7.286	7.493	7.699	7.905	8.11	8.313	8.516	8.718	8.918	9.118
323	7.072	7.279	7.485	7.691	7.896	8.1	8.303	8.506	8.707	8.907	9.106
324	7.065	7.272	7.478	7.683	7.887	8.091	8.293	8.495	8.696	8.895	9.093
325	7.059	7.265	7.47	7.675	7.879	8.082	8.284	8.485	8.685	8.883	9.081
326	7.052	7.258	7.463	7.667	7.87	8.072	8.274	8.474	8.674	8.872	9.069
327	7.046	7.251	7.455	7.659	7.861	8.063	8.264	8.464	8.663	8.86	9.057
328	7.04	7.244	7.448	7.651	7.853	8.054	8.254	8.453	8.652	8.849	9.045
329	7.034	7.237	7.44	7.643	7.844	8.045	8.244	8.443	8.641	8.838	9.033
330	7.028	7.231	7.433	7.635	7.836	8.036	8.235	8.433	8.63	8.827	9.022
331	7.021	7.224	7.426	7.627	7.827	8.027	8.225	8.423	8.62	8.815	9.01
332	7.015	7.217	7.418	7.619	7.819	8.018	8.216	8.413	8.609	8.804	8.998
333	7.009	7.211	7.411	7.611	7.81	8.009	8.206	8.403	8.598	8.793	8.987
334	7.003	7.204	7.404	7.603	7.802	8	8.197	8.393	8.588	8.782	8.975
335	6.997	7.197	7.397	7.596	7.794	7.991	8.187	8.383	8.578	8.771	8.964
336	6.991	7.191	7.39	7.588	7.786	7.982	8.178	8.373	8.567	8.76	8.952
337	6.985	7.184	7.383	7.58	7.777	7.974	8.169	8.363	8.557	8.749	8.941
338	6.98	7.178	7.376	7.573	7.769	7.965	8.16	8.354	8.547	8.739	8.93
339	6.974	7.172	7.369	7.565	7.761	7.956	8.151	8.344	8.536	8.728	8.918
340	6.968	7.165	7.362	7.558	7.753	7.948	8.141	8.334	8.526	8.717	8.907
341	6.962	7.159	7.355	7.55	7.745	7.939	8.132	8.325	8.516	8.707	8.896
342	6.956	7.153	7.348	7.543	7.737	7.931	8.123	8.315	8.506	8.696	8.885
343	6.951	7.146	7.341	7.536	7.729	7.922	8.114	8.306	8.496	8.686	8.874
344	6.945	7.14	7.335	7.528	7.721	7.914	8.106	8.296	8.486	8.675	8.863

345	6.939	7.134	7.328	7.521	7.714	7.906	8.097	8.287	8.476	8.665	8.853
346	6.934	7.128	7.321	7.514	7.706	7.897	8.088	8.278	8.467	8.655	8.842
347	6.928	7.122	7.314	7.507	7.698	7.889	8.079	8.268	8.457	8.644	8.831
348	6.923	7.116	7.308	7.5	7.691	7.881	8.07	8.259	8.447	8.634	8.82
349	6.917	7.109	7.301	7.492	7.683	7.873	8.062	8.25	8.438	8.624	8.81
350	6.912	7.103	7.295	7.485	7.675	7.865	8.053	8.241	8.428	8.614	8.799
351	6.906	7.097	7.288	7.478	7.668	7.857	8.045	8.232	8.418	8.604	8.789
352	6.901	7.091	7.282	7.471	7.66	7.849	8.036	8.223	8.409	8.594	8.778
353	6.895	7.086	7.275	7.464	7.653	7.841	8.028	8.214	8.399	8.584	8.768
354	6.89	7.08	7.269	7.457	7.645	7.833	8.019	8.205	8.39	8.574	8.758
355	6.885	7.074	7.262	7.451	7.638	7.825	8.011	8.196	8.381	8.564	8.747
356	6.879	7.068	7.256	7.444	7.631	7.817	8.003	8.187	8.371	8.555	8.737
357	6.874	7.062	7.25	7.437	7.623	7.809	7.994	8.179	8.362	8.545	8.727
358	6.869	7.056	7.244	7.43	7.616	7.801	7.986	8.17	8.353	8.535	8.717
359	6.864	7.051	7.237	7.423	7.609	7.794	7.978	8.161	8.344	8.526	8.707
360	6.858	7.045	7.231	7.417	7.602	7.786	7.97	8.153	8.335	8.516	8.697
361	6.853	7.039	7.225	7.41	7.595	7.778	7.962	8.144	8.326	8.507	8.687
362	6.848	7.034	7.219	7.403	7.587	7.771	7.954	8.136	8.317	8.497	8.677
363	6.843	7.028	7.213	7.397	7.58	7.763	7.946	8.127	8.308	8.488	8.667
364	6.838	7.023	7.207	7.39	7.573	7.756	7.938	8.119	8.299	8.479	8.657
365	6.833	7.017	7.201	7.384	7.566	7.748	7.93	8.11	8.29	8.469	8.647
366	6.828	7.012	7.195	7.377	7.559	7.741	7.922	8.102	8.281	8.46	8.638
367	6.823	7.006	7.189	7.371	7.553	7.734	7.914	8.094	8.273	8.451	8.628
368	6.818	7.001	7.183	7.365	7.546	7.726	7.906	8.085	8.264	8.442	8.619
369	6.813	6.995	7.177	7.358	7.539	7.719	7.898	8.077	8.255	8.432	8.609
370	6.808	6.99	7.171	7.352	7.532	7.712	7.891	8.069	8.247	8.423	8.6
371	6.803	6.984	7.165	7.345	7.525	7.704	7.883	8.061	8.238	8.414	8.59
372	6.798	6.979	7.159	7.339	7.518	7.697	7.875	8.053	8.229	8.405	8.581
373	6.793	6.974	7.154	7.333	7.512	7.69	7.868	8.045	8.221	8.397	8.571
374	6.789	6.968	7.148	7.327	7.505	7.683	7.86	8.037	8.213	8.388	8.562
375	6.784	6.963	7.142	7.321	7.498	7.676	7.853	8.029	8.204	8.379	8.553
376	6.779	6.958	7.136	7.314	7.492	7.669	7.845	8.021	8.196	8.37	8.544
377	6.774	6.953	7.131	7.308	7.485	7.662	7.838	8.013	8.187	8.361	8.534
378	6.77	6.948	7.125	7.302	7.479	7.655	7.83	8.005	8.179	8.353	8.525
379	6.765	6.942	7.12	7.296	7.472	7.648	7.823	7.997	8.171	8.344	8.516
380	6.76	6.937	7.114	7.29	7.466	7.641	7.816	7.989	8.163	8.335	8.507
381	6.756	6.932	7.108	7.284	7.459	7.634	7.808	7.982	8.155	8.327	8.498
382	6.751	6.927	7.103	7.278	7.453	7.627	7.801	7.974	8.146	8.318	8.489
383	6.746	6.922	7.097	7.272	7.447	7.62	7.794	7.966	8.138	8.31	8.48
384	6.742	6.917	7.092	7.266	7.44	7.614	7.787	7.959	8.13	8.301	8.472
385	6.737	6.912	7.087	7.261	7.434	7.607	7.779	7.951	8.122	8.293	8.463
386	6.733	6.907	7.081	7.255	7.428	7.6	7.772	7.944	8.114	8.285	8.454
387	6.728	6.902	7.076	7.249	7.422	7.594	7.765	7.936	8.107	8.276	8.445
388	6.724	6.897	7.07	7.243	7.415	7.587	7.758	7.929	8.099	8.268	8.437

389	6.719	6.892	7.065	7.237	7.409	7.58	7.751	7.921	8.091	8.26	8.428
390	6.715	6.888	7.06	7.232	7.403	7.574	7.744	7.914	8.083	8.252	8.419
391	6.71	6.883	7.055	7.226	7.397	7.567	7.737	7.907	8.075	8.243	8.411
392	6.706	6.878	7.049	7.22	7.391	7.561	7.73	7.899	8.068	8.235	8.402
393	6.702	6.873	7.044	7.215	7.385	7.554	7.723	7.892	8.06	8.227	8.394
394	6.697	6.868	7.039	7.209	7.379	7.548	7.717	7.885	8.052	8.219	8.385
395	6.693	6.864	7.034	7.203	7.373	7.542	7.71	7.878	8.045	8.211	8.377
396	6.689	6.859	7.029	7.198	7.367	7.535	7.703	7.87	8.037	8.203	8.369
397	6.684	6.854	7.024	7.192	7.361	7.529	7.696	7.863	8.03	8.195	8.36
398	6.68	6.85	7.018	7.187	7.355	7.523	7.69	7.856	8.022	8.187	8.352
399	6.676	6.845	7.013	7.181	7.349	7.516	7.683	7.849	8.015	8.18	8.344
400	6.672	6.84	7.008	7.176	7.343	7.51	7.676	7.842	8.007	8.172	8.336



附件 8：

曲线行驶过程中方向盘转动结束时的 x 坐标与方向盘转速和无人车行驶速度的关系表

	10	11	12	13	14	15	16	17	18	19	20
100	5.5571	6.44839	7.36238	8.2878	9.2134	10.1281	11.021	11.8819	12.701	13.4693	14.1788
101	5.4628	6.34214	7.24502	8.16052	9.07771	9.98576	10.8741	11.7327	12.552	13.323	14.0379
102	5.37073	6.23827	7.13014	8.03574	8.94445	9.84573	10.7293	11.5853	12.4042	13.1775	13.897
103	5.28082	6.13673	7.01768	7.91342	8.8136	9.70797	10.5865	11.4395	12.2578	13.0327	13.7564
104	5.19302	6.03745	6.9076	7.7935	8.68512	9.57247	10.4457	11.2955	12.1127	12.8888	13.6161
105	5.10725	5.94038	6.79984	7.67595	8.55898	9.43919	10.307	11.1533	11.969	12.7459	13.4763
106	5.02348	5.84546	6.69433	7.56072	8.43514	9.30813	10.1704	11.0128	11.8267	12.604	13.3369
107	4.94163	5.75263	6.59104	7.44776	8.31356	9.17925	10.0357	10.8741	11.686	12.4632	13.1982
108	4.86165	5.66184	6.4899	7.33702	8.19422	9.05254	9.90312	10.7373	11.5467	12.3235	13.0602
109	4.7835	5.57303	6.39088	7.22845	8.07706	8.92796	9.77252	10.6023	11.409	12.1851	12.923
110	4.70712	5.48616	6.29391	7.12202	7.96206	8.8055	9.64393	10.4691	11.2729	12.0478	12.7866
111	4.63247	5.40118	6.19894	7.01768	7.84917	8.68512	9.51732	10.3377	11.1384	11.9119	12.6512

112	4.55948	5.31802	6.10594	6.91539	7.73836	8.56679	9.39268	10.2081	11.0055	11.7773	12.5167
113	4.48813	5.23666	6.01486	6.81509	7.62958	8.45049	9.27	10.0804	10.8741	11.644	12.3832
114	4.41836	5.15704	5.92564	6.71675	7.52282	8.33619	9.14924	9.95445	10.7445	11.5122	12.2508
115	4.35013	5.07911	5.83824	6.62033	7.41801	8.22385	9.0304	9.83032	10.6164	11.3817	12.1196
116	4.28339	5.00284	5.75263	6.52578	7.31513	8.11344	8.91345	9.70797	10.49	11.2526	11.9894
117	4.21812	4.92817	5.66875	6.43306	7.21414	8.00493	8.79836	9.58741	10.3652	11.125	11.8605
118	4.15425	4.85507	5.58657	6.34214	7.115	7.8983	8.68512	9.46862	10.242	10.9989	11.7327
119	4.09177	4.7835	5.50604	6.25296	7.01768	7.7935	8.5737	9.35157	10.1205	10.8741	11.6062
120	4.03063	4.71342	5.42713	6.16551	6.92214	7.69052	8.46407	9.23626	10.0006	10.7509	11.481
121	3.9708	4.64479	5.34979	6.07972	6.82835	7.58931	8.35622	9.12267	9.88236	10.6291	11.357
122	3.91223	4.57758	5.27399	5.99558	6.73627	7.48985	8.2501	9.01078	9.7657	10.5088	11.2343
123	3.8549	4.51174	5.1997	5.91304	6.64586	7.39211	8.14571	8.90057	9.65064	10.39	11.1129
124	3.79877	4.44724	5.12687	5.83207	6.55709	7.29605	8.04301	8.79202	9.53718	10.2727	10.9929
125	3.74382	4.38405	5.05547	5.75263	6.46993	7.20165	7.94198	8.68512	9.42529	10.1568	10.8741
126	3.69001	4.32214	4.98546	5.67469	6.38435	7.10887	7.84259	8.57984	9.31497	10.0424	10.7567
127	3.6373	4.26147	4.91682	5.59821	6.30031	7.01768	7.74482	8.47616	9.2062	9.92948	10.6407
128	3.58568	4.20202	4.84951	5.52316	6.21778	6.92806	7.64863	8.37407	9.09897	9.818	10.5259
129	3.53511	4.14375	4.7835	5.44951	6.13673	6.83998	7.55401	8.27353	8.99326	9.70797	10.4125
130	3.48557	4.08663	4.71876	5.37723	6.05713	6.7534	7.46093	8.17454	8.88906	9.59939	10.3005
131	3.43703	4.03063	4.65526	5.30629	5.97895	6.6683	7.36936	8.07706	8.78636	9.49223	10.1898
132	3.38947	3.97573	4.59296	5.23666	5.90216	6.58466	7.27927	7.98107	8.68512	9.3865	10.0804
133	3.34285	3.9219	4.53185	5.16831	5.82673	6.50243	7.19065	7.88656	8.58534	9.28218	9.97233
134	3.29716	3.86912	4.47189	5.1012	5.75263	6.4216	7.10346	7.7935	8.487	9.17925	9.8656
135	3.25237	3.81735	4.41306	5.03532	5.67984	6.34214	7.01768	7.70187	8.39009	9.07771	9.76019
136	3.20846	3.76658	4.35532	4.97064	5.60832	6.26402	6.93329	7.61165	8.29457	8.97754	9.65609
137	3.1654	3.71677	4.29866	4.90713	5.53806	6.18721	6.85026	7.52282	8.20044	8.87873	9.55329
138	3.12319	3.66791	4.24305	4.84476	5.46902	6.1117	6.76857	7.43534	8.10768	8.78126	9.45179
139	3.08178	3.61998	4.18847	4.7835	5.40118	6.03745	6.6882	7.34921	8.01626	8.68512	9.35157
140	3.04118	3.57294	4.13488	4.72334	5.33451	5.96444	6.60911	7.2644	7.92618	8.5903	9.25263
141	3.00134	3.52679	4.08228	4.66425	5.26899	5.89265	6.53129	7.18089	7.83741	8.49677	9.15495
142	2.96227	3.48149	4.03063	4.60621	5.2046	5.82206	6.45471	7.09866	7.74992	8.40453	9.05852
143	2.92393	3.43703	3.97992	4.54919	5.14132	5.75263	6.37936	7.01768	7.66372	8.31356	8.96334
144	2.88631	3.39339	3.93012	4.49317	5.07911	5.68435	6.30521	6.93795	7.57876	8.22385	8.86939
145	2.8494	3.35055	3.88121	4.43813	5.01796	5.61719	6.23223	6.85942	7.49505	8.13537	8.77665
146	2.81317	3.30849	3.83317	4.38405	4.95785	5.55114	6.16041	6.7821	7.41255	8.04811	8.68512
147	2.77761	3.2672	3.78599	4.33091	4.89874	5.48616	6.08973	6.70594	7.33125	7.96206	8.59478
148	2.7427	3.22665	3.73964	4.27868	4.84064	5.42225	6.02016	6.63095	7.25113	7.87719	8.50562
149	2.70843	3.18682	3.69411	4.22735	4.7835	5.35937	5.95169	6.55709	7.17218	7.7935	8.41763
150	2.67479	3.14771	3.64937	4.1769	4.72732	5.29752	5.88429	6.48435	7.09437	7.71097	8.33079
151	2.64175	3.10929	3.60541	4.12731	4.67207	5.23666	5.81794	6.4127	7.01768	7.62958	8.24509
152	2.60931	3.07156	3.56221	4.07856	4.61774	5.17678	5.75263	6.34214	6.94211	7.54932	8.16052
153	2.57745	3.03448	3.51976	4.03063	4.5643	5.11786	5.68833	6.27263	6.86763	7.47017	8.07706
154	2.54616	2.99806	3.47804	3.98351	4.51174	5.05989	5.62503	6.20417	6.79423	7.39211	7.9947
155	2.51542	2.96227	3.43703	3.93718	4.46003	5.00284	5.56271	6.13673	6.72188	7.31513	7.91342

156	2.48522	2.9271	3.39672	3.89161	4.40917	4.94669	5.50135	6.0703	6.65057	7.23921	7.83321
157	2.45555	2.89253	3.35709	3.84681	4.35914	4.89143	5.44094	6.00485	6.58029	7.16434	7.75406
158	2.42639	2.85856	3.31813	3.80274	4.30991	4.83704	5.38145	5.94038	6.51102	7.0905	7.67595
159	2.39775	2.82517	3.27982	3.7594	4.26147	4.7835	5.32287	5.87686	6.44274	7.01768	7.59888
160	2.36959	2.79234	3.24216	3.71677	4.21381	4.73081	5.26518	5.81428	6.37543	6.94586	7.52282
161	2.34192	2.76007	3.20511	3.67484	4.16692	4.67893	5.20836	5.75263	6.30908	6.87503	7.44776
162	2.31473	2.72835	3.16868	3.63358	4.12076	4.62786	5.15241	5.69188	6.24368	6.80517	7.37368
163	2.28799	2.69715	3.13285	3.59299	4.07534	4.57758	5.0973	5.63202	6.1792	6.73627	7.30059
164	2.26171	2.66647	3.09761	3.55306	4.03063	4.52807	5.04301	5.57303	6.11564	6.6683	7.22845
165	2.23586	2.63631	3.06294	3.51376	3.98663	4.47932	4.98954	5.51491	6.05298	6.60127	7.15727
166	2.21045	2.60663	3.02884	3.47509	3.94331	4.43132	4.93687	5.45763	5.9912	6.53515	7.08701
167	2.18547	2.57745	2.99528	3.43703	3.90067	4.38405	4.88498	5.40118	5.93029	6.46993	7.01768
168	2.16089	2.54874	2.96227	3.39958	3.85868	4.3375	4.83387	5.34554	5.87023	6.4056	6.94926
169	2.13672	2.5205	2.92978	3.36271	3.81735	4.29165	4.7835	5.29071	5.81102	6.34214	6.88174
170	2.11295	2.49272	2.89781	3.32643	3.77665	4.2465	4.73388	5.23666	5.75263	6.27953	6.81509
171	2.08956	2.46538	2.86635	3.2907	3.73658	4.20202	4.68499	5.18339	5.69505	6.21778	6.74932
172	2.06656	2.43848	2.83538	3.25554	3.69712	4.15821	4.63681	5.13088	5.63828	6.15685	6.6844
173	2.04392	2.41201	2.8049	3.22091	3.65825	4.11504	4.58934	5.07911	5.58229	6.09675	6.62033
174	2.02165	2.38596	2.7749	3.18682	3.61998	4.07252	4.54255	5.02808	5.52707	6.03745	6.55709
175	1.99974	2.36032	2.74536	3.15326	3.58228	4.03063	4.49644	4.97777	5.47262	5.97895	6.49467
176	1.97817	2.33508	2.71629	3.1202	3.54514	3.98936	4.451	4.92817	5.41891	5.92123	6.43306
177	1.95695	2.31024	2.68766	3.08765	3.50857	3.94869	4.40621	4.87927	5.36594	5.86427	6.37225
178	1.93606	2.28578	2.65947	3.05559	3.47253	3.90861	4.36206	4.83105	5.3137	5.80808	6.31222
179	1.91549	2.26171	2.63171	3.02401	3.43703	3.86912	4.31854	4.7835	5.26216	5.75263	6.25296
180	1.89525	2.238	2.60437	2.99291	3.40206	3.8302	4.27564	4.73662	5.21133	5.69791	6.19447
181	1.87532	2.21466	2.57745	2.96227	3.36759	3.79184	4.23335	4.69039	5.16119	5.64392	6.13673
182	1.8557	2.19167	2.55093	2.93208	3.33364	3.75404	4.19165	4.64479	5.11172	5.59064	6.07972
183	1.83639	2.16904	2.52482	2.90235	3.30017	3.71677	4.15054	4.59983	5.06292	5.53806	6.02345
184	1.81736	2.14674	2.49909	2.87305	3.2672	3.68004	4.11001	4.55548	5.01477	5.48616	5.96789
185	1.79863	2.12479	2.47374	2.84418	3.2347	3.64383	4.07004	4.51174	4.96727	5.43495	5.91304
186	1.78018	2.10316	2.44877	2.81573	3.20267	3.60813	4.03063	4.46859	4.9204	5.3844	5.85889
187	1.76201	2.08185	2.42417	2.7877	3.17109	3.57294	3.99177	4.42603	4.87416	5.33451	5.80542
188	1.74411	2.06087	2.39993	2.76007	3.13997	3.53824	3.95344	4.38405	4.82853	5.28527	5.75263
189	1.72648	2.04019	2.37605	2.73285	3.10929	3.50403	3.91564	4.34264	4.7835	5.23666	5.7005
190	1.70912	2.01981	2.35251	2.70601	3.07905	3.4703	3.87836	4.30178	4.73907	5.18868	5.64903
191	1.69201	1.99974	2.32931	2.67956	3.04923	3.43703	3.84159	4.26147	4.69522	5.14132	5.59821
192	1.67515	1.97996	2.30645	2.65348	3.01984	3.40423	3.80532	4.2217	4.65195	5.09456	5.54802
193	1.65854	1.96046	2.28392	2.62778	2.99085	3.37187	3.76954	4.18246	4.60924	5.0484	5.49846
194	1.64217	1.94125	2.26171	2.60244	2.96227	3.33997	3.73424	4.14375	4.56709	5.00284	5.44951
195	1.62604	1.92231	2.23981	2.57745	2.93408	3.30849	3.69942	4.10554	4.52548	4.95785	5.40118
196	1.61014	1.90365	2.21823	2.55281	2.90629	3.27745	3.66507	4.06784	4.48442	4.91343	5.35344
197	1.59447	1.88525	2.19695	2.52852	2.87887	3.24683	3.63117	4.03063	4.44388	4.86957	5.30629
198	1.57903	1.86711	2.17597	2.50457	2.85184	3.21662	3.59773	3.99391	4.40387	4.82626	5.25973
199	1.56381	1.84923	2.15528	2.48095	2.82517	3.18682	3.56473	3.95767	4.36437	4.7835	5.21374

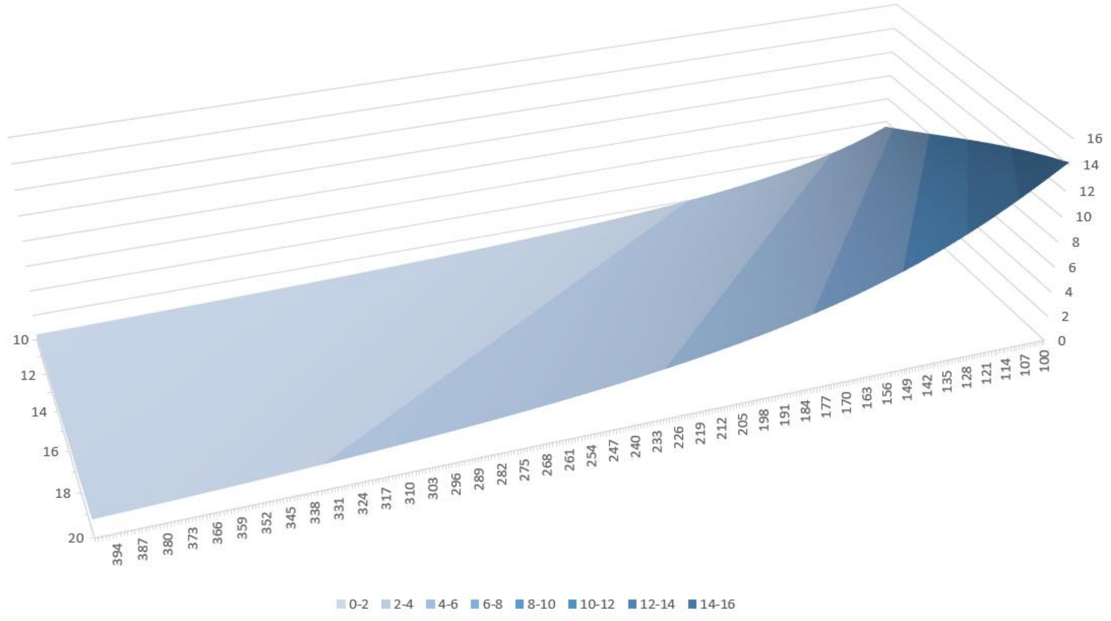
200	1.5488	1.8316	2.13488	2.45765	2.79886	3.15742	3.53217	3.9219	4.32537	4.74128	5.16831
201	1.53401	1.81422	2.11476	2.43467	2.77292	3.12842	3.50004	3.8866	4.28687	4.69958	5.12343
202	1.51942	1.79708	2.09493	2.41201	2.74732	3.0998	3.46833	3.85175	4.24886	4.6584	5.07911
203	1.50504	1.78018	2.07536	2.38965	2.72207	3.07156	3.43703	3.81735	4.21133	4.61774	5.03532
204	1.49086	1.76352	2.05607	2.3676	2.69715	3.04369	3.40614	3.78339	4.17427	4.57758	4.99207
205	1.47688	1.74708	2.03703	2.34585	2.67257	3.01619	3.37566	3.74987	4.13768	4.53791	4.94934
206	1.46309	1.73087	2.01826	2.32439	2.64831	2.98905	3.34557	3.71677	4.10155	4.49873	4.90713
207	1.44949	1.71488	1.99974	2.30321	2.62438	2.96227	3.31586	3.6841	4.06587	4.46003	4.86542
208	1.43608	1.69911	1.98147	2.28232	2.60076	2.93583	3.28654	3.65183	4.03063	4.42181	4.82422
209	1.42285	1.68355	1.96344	2.26171	2.57745	2.90974	3.25759	3.61998	3.99583	4.38405	4.7835
210	1.4098	1.6682	1.94566	2.24136	2.55445	2.88398	3.22901	3.58852	3.96146	4.34676	4.74328
211	1.39693	1.65305	1.92811	2.22129	2.53174	2.85856	3.2008	3.55746	3.92752	4.30991	4.70353
212	1.38423	1.63811	1.9108	2.20148	2.50933	2.83346	3.17294	3.52679	3.89399	4.27351	4.66425
213	1.3717	1.62337	1.89371	2.18193	2.48721	2.80869	3.14543	3.4965	3.86088	4.23755	4.62544
214	1.35934	1.60883	1.87685	2.16263	2.46538	2.78423	3.11827	3.46658	3.82817	4.20202	4.58709
215	1.34714	1.59447	1.86021	2.14359	2.44382	2.76007	3.09145	3.43703	3.79585	4.16692	4.54919
216	1.33511	1.58031	1.84378	2.12479	2.42255	2.73623	3.06497	3.40785	3.76393	4.13223	4.51174
217	1.32323	1.56633	1.82757	2.10623	2.40154	2.71268	3.03881	3.37903	3.7324	4.09796	4.47472
218	1.31151	1.55253	1.81157	2.08791	2.3808	2.68943	3.01298	3.35055	3.70124	4.06409	4.43813
219	1.29995	1.53892	1.79577	2.06982	2.36032	2.66647	2.98747	3.32243	3.67046	4.03063	4.40198
220	1.28853	1.52548	1.78018	2.05197	2.3401	2.6438	2.96227	3.29465	3.64005	3.99756	4.36624
221	1.27726	1.51221	1.76479	2.03434	2.32013	2.62141	2.93738	3.2672	3.61	3.96488	4.33091
222	1.26614	1.49911	1.74959	2.01693	2.30041	2.59929	2.91279	3.24008	3.58031	3.93259	4.29599
223	1.25516	1.48618	1.73459	1.99974	2.28094	2.57745	2.8885	3.21329	3.55097	3.90067	4.26147
224	1.24433	1.47341	1.71977	1.98277	2.26171	2.55588	2.86451	3.18682	3.52198	3.86912	4.22735
225	1.23363	1.46081	1.70515	1.966	2.24271	2.53456	2.84081	3.16067	3.49333	3.83794	4.19362
226	1.22306	1.44837	1.6907	1.94945	2.22395	2.51351	2.81739	3.13483	3.46501	3.80712	4.16028
227	1.21264	1.43608	1.67644	1.9331	2.20542	2.49272	2.79426	3.10929	3.43703	3.77665	4.12731
228	1.20234	1.42395	1.66235	1.91695	2.18712	2.47217	2.7714	3.08406	3.40938	3.74654	4.09472
229	1.19217	1.41196	1.64844	1.901	2.16904	2.45187	2.74882	3.05913	3.38204	3.71677	4.06249
230	1.18213	1.40013	1.6347	1.88525	2.15118	2.43182	2.7265	3.03448	3.35503	3.68734	4.03063
231	1.17221	1.38844	1.62112	1.86969	2.13353	2.41201	2.70444	3.01013	3.32832	3.65825	3.99913
232	1.16242	1.3769	1.60772	1.85432	2.1161	2.39243	2.68265	2.98606	3.30192	3.62949	3.96798
233	1.15275	1.3655	1.59447	1.83913	2.09887	2.37309	2.66111	2.96227	3.27583	3.60106	3.93718
234	1.1432	1.35424	1.58139	1.82412	2.08185	2.35397	2.63983	2.93875	3.25003	3.57294	3.90672
235	1.13376	1.34311	1.56847	1.8093	2.06504	2.33508	2.61879	2.91551	3.22453	3.54514	3.8766
236	1.12444	1.33212	1.5557	1.79465	2.04842	2.31641	2.598	2.89253	3.19932	3.51766	3.84681
237	1.11524	1.32127	1.54309	1.78018	2.032	2.29796	2.57745	2.86982	3.1744	3.49048	3.81735
238	1.10614	1.31054	1.53062	1.76588	2.01578	2.27973	2.55714	2.84737	3.14975	3.46361	3.78822
239	1.09716	1.29995	1.51831	1.75175	1.99974	2.26171	2.53706	2.82517	3.12539	3.43703	3.7594
240	1.08828	1.28948	1.50614	1.73779	1.98389	2.24389	2.51721	2.80322	3.10129	3.41075	3.7309
241	1.07951	1.27913	1.49412	1.72399	1.96823	2.22628	2.49759	2.78153	3.07747	3.38476	3.70272
242	1.07085	1.26891	1.48224	1.71035	1.95274	2.20888	2.47819	2.76007	3.05391	3.35906	3.67484
243	1.06228	1.25881	1.47049	1.69687	1.93744	2.19167	2.45901	2.73886	3.03062	3.33364	3.64726

244	1.05382	1.24883	1.45889	1.68355	1.92231	2.17466	2.44005	2.71789	3.00758	3.30849	3.61998
245	1.04546	1.23896	1.44742	1.67038	1.90736	2.15785	2.4213	2.69715	2.9848	3.28363	3.59299
246	1.0372	1.22921	1.43608	1.65736	1.89258	2.14122	2.40277	2.67664	2.96227	3.25903	3.5663
247	1.02903	1.21957	1.42487	1.64449	1.87796	2.12479	2.38444	2.65636	2.93998	3.2347	3.53989
248	1.02096	1.21005	1.4138	1.63177	1.86352	2.10854	2.36631	2.63631	2.91794	3.21063	3.51376
249	1.01299	1.20063	1.40285	1.6192	1.84923	2.09247	2.34839	2.61647	2.89614	3.18682	3.48791
250	1.0051	1.19133	1.39202	1.60677	1.83511	2.07658	2.33067	2.59685	2.87458	3.16327	3.46234
251	0.99731	1.18213	1.38132	1.59447	1.82115	2.06087	2.31314	2.57745	2.85325	3.13997	3.43703
252	0.98961	1.17304	1.37074	1.58232	1.80734	2.04533	2.29581	2.55826	2.83215	3.11692	3.412
253	0.98199	1.16405	1.36028	1.57031	1.79368	2.02996	2.27866	2.53928	2.81128	3.09412	3.38722
254	0.97446	1.15516	1.34994	1.55843	1.78018	2.01477	2.26171	2.5205	2.79063	3.07156	3.36271
255	0.96702	1.14637	1.33972	1.54668	1.76683	1.99974	2.24493	2.50193	2.77021	3.04923	3.33846
256	0.95966	1.13768	1.32961	1.53506	1.75363	1.98488	2.22835	2.48355	2.75	3.02715	3.31446
257	0.95239	1.12909	1.31961	1.52357	1.74057	1.97017	2.21194	2.46538	2.73	3.00529	3.2907
258	0.94519	1.12059	1.30972	1.51221	1.72765	1.95563	2.1957	2.4474	2.71022	2.98367	3.2672
259	0.93808	1.11219	1.29995	1.50097	1.71488	1.94125	2.17965	2.42961	2.69065	2.96227	3.24394
260	0.93105	1.10389	1.29028	1.48986	1.70224	1.92702	2.16376	2.41201	2.67128	2.94109	3.22091
261	0.92409	1.09567	1.28071	1.47887	1.68974	1.91295	2.14805	2.39459	2.65212	2.92014	3.19813
262	0.91721	1.08755	1.27126	1.46799	1.67738	1.89902	2.1325	2.37737	2.63316	2.8994	3.17558
263	0.91041	1.07951	1.2619	1.45724	1.66515	1.88525	2.11712	2.36032	2.61439	2.87887	3.15326
264	0.90368	1.07156	1.25265	1.4466	1.65305	1.87162	2.1019	2.34345	2.59583	2.85856	3.13116
265	0.89703	1.0637	1.2435	1.43608	1.64109	1.85814	2.08684	2.32676	2.57745	2.83846	3.10929
266	0.89045	1.05593	1.23445	1.42567	1.62925	1.8448	2.07194	2.31024	2.55926	2.81856	3.08765
267	0.88394	1.04824	1.22549	1.41537	1.61753	1.8316	2.05719	2.29389	2.54127	2.79886	3.06622
268	0.8775	1.04063	1.21663	1.40518	1.60594	1.81854	2.0426	2.27772	2.52345	2.77937	3.04501
269	0.87113	1.0331	1.20787	1.3951	1.59447	1.80562	2.02817	2.26171	2.50582	2.76007	3.02401
270	0.86483	1.02566	1.1992	1.38513	1.58313	1.79284	2.01388	2.24586	2.48837	2.74097	3.00322
271	0.8586	1.01829	1.19062	1.37526	1.5719	1.78018	1.99974	2.23018	2.4711	2.72207	2.98264
272	0.85243	1.01101	1.18213	1.3655	1.56079	1.76766	1.98574	2.21466	2.454	2.70335	2.96227
273	0.84633	1.0038	1.17373	1.35584	1.5498	1.75527	1.97189	2.1993	2.43708	2.68482	2.94209
274	0.8403	0.99666	1.16542	1.34628	1.53892	1.74301	1.95819	2.18409	2.42032	2.66647	2.92212
275	0.83433	0.98961	1.1572	1.33682	1.52815	1.73087	1.94462	2.16904	2.40374	2.64831	2.90235
276	0.82842	0.98262	1.14906	1.32746	1.51749	1.71885	1.93119	2.15414	2.38732	2.63033	2.88277
277	0.82257	0.97571	1.14101	1.31819	1.50695	1.70696	1.9179	2.13939	2.37106	2.61253	2.86338
278	0.81679	0.96887	1.13304	1.30902	1.49651	1.6952	1.90474	2.12479	2.35497	2.5949	2.84418
279	0.81107	0.9621	1.12516	1.29995	1.48618	1.68355	1.89171	2.11033	2.33904	2.57745	2.82517
280	0.8054	0.95541	1.11735	1.29096	1.47595	1.67202	1.87882	2.09602	2.32326	2.56017	2.80634
281	0.79979	0.94878	1.10963	1.28207	1.46583	1.6606	1.86605	2.08185	2.30765	2.54306	2.7877
282	0.79425	0.94222	1.10198	1.27328	1.45581	1.6493	1.85342	2.06783	2.29218	2.52611	2.76924
283	0.78875	0.93573	1.09442	1.26457	1.4459	1.63811	1.84091	2.05394	2.27687	2.50933	2.75095
284	0.78332	0.9293	1.08693	1.25594	1.43608	1.62704	1.82852	2.04019	2.26171	2.49272	2.73285
285	0.77794	0.92294	1.07951	1.24741	1.42636	1.61608	1.81625	2.02657	2.24669	2.47626	2.71491
286	0.77262	0.91664	1.07217	1.23896	1.41674	1.60522	1.80411	2.01309	2.23182	2.45996	2.69715
287	0.76735	0.91041	1.06491	1.2306	1.40721	1.59447	1.79209	1.99974	2.2171	2.44382	2.67956

288	0.76213	0.90424	1.05772	1.22232	1.39778	1.58383	1.78018	1.98652	2.20252	2.42784	2.66213
289	0.75697	0.89813	1.0506	1.21412	1.38844	1.5733	1.76839	1.97343	2.18808	2.41201	2.64487
290	0.75186	0.89209	1.04355	1.206	1.3792	1.56286	1.75672	1.96046	2.17377	2.39633	2.62778
291	0.7468	0.8861	1.03657	1.19796	1.37004	1.55253	1.74516	1.94762	2.15961	2.3808	2.61084
292	0.74179	0.88018	1.02966	1.19001	1.36098	1.5423	1.73371	1.93491	2.14558	2.36541	2.59407
293	0.73683	0.87431	1.02282	1.18213	1.352	1.53217	1.72237	1.92231	2.13169	2.35017	2.57745
294	0.73192	0.8685	1.01604	1.17433	1.34311	1.52214	1.71115	1.90984	2.11792	2.33508	2.56099
295	0.72706	0.86275	1.00934	1.1666	1.33431	1.51221	1.70003	1.89749	2.10429	2.32013	2.54468
296	0.72224	0.85705	1.0027	1.15895	1.32559	1.50237	1.68901	1.88525	2.09079	2.30532	2.52852
297	0.71748	0.85141	0.99612	1.15138	1.31696	1.49262	1.67811	1.87313	2.07741	2.29064	2.51252
298	0.71276	0.84583	0.98961	1.14388	1.30841	1.48297	1.6673	1.86112	2.06416	2.27611	2.49666
299	0.70808	0.8403	0.98316	1.13645	1.29995	1.47341	1.6566	1.84923	2.05103	2.26171	2.48095
300	0.70346	0.83482	0.97677	1.12909	1.29156	1.46395	1.646	1.83745	2.03803	2.24744	2.46538
301	0.69887	0.8294	0.97044	1.1218	1.28325	1.45457	1.6355	1.82578	2.02515	2.2333	2.44995
302	0.69434	0.82403	0.96418	1.11458	1.27503	1.44528	1.6251	1.81422	2.01238	2.2193	2.43467
303	0.68984	0.81871	0.95798	1.10744	1.26688	1.43608	1.61479	1.80277	1.99974	2.20542	2.41953
304	0.68539	0.81344	0.95183	1.10035	1.25881	1.42696	1.60459	1.79142	1.98721	2.19167	2.40452
305	0.68098	0.80822	0.94574	1.09334	1.25081	1.41794	1.59447	1.78018	1.9748	2.17805	2.38965
306	0.67662	0.80306	0.93971	1.08639	1.24289	1.40899	1.58446	1.76905	1.9625	2.16455	2.37492
307	0.67229	0.79794	0.93374	1.07951	1.23505	1.40013	1.57453	1.75801	1.95032	2.15118	2.36032
308	0.66801	0.79287	0.92783	1.07269	1.22727	1.39135	1.5647	1.74708	1.93824	2.13792	2.34585
309	0.66377	0.78785	0.92197	1.06594	1.21957	1.38265	1.55496	1.73625	1.92628	2.12479	2.33151
310	0.65956	0.78287	0.91616	1.05925	1.21195	1.37404	1.5453	1.72551	1.91442	2.11177	2.3173
311	0.6554	0.77794	0.91041	1.05262	1.20439	1.3655	1.53574	1.71488	1.90267	2.09887	2.30321
312	0.65128	0.77306	0.90471	1.04606	1.1969	1.35704	1.52626	1.70434	1.89103	2.08609	2.28925
313	0.64719	0.76822	0.89907	1.03955	1.18948	1.34866	1.51687	1.6939	1.87949	2.07342	2.27542
314	0.64314	0.76343	0.89348	1.0331	1.18213	1.34035	1.50757	1.68355	1.86806	2.06087	2.26171
315	0.63913	0.75869	0.88794	1.02672	1.17485	1.33212	1.49835	1.67329	1.85673	2.04842	2.24811
316	0.63516	0.75398	0.88245	1.02039	1.16763	1.32397	1.48921	1.66313	1.8455	2.03609	2.23464
317	0.63123	0.74932	0.87701	1.01412	1.16048	1.31589	1.48015	1.65305	1.83437	2.02386	2.22129
318	0.62733	0.7447	0.87162	1.00791	1.15339	1.30788	1.47118	1.64307	1.82334	2.01175	2.20806
319	0.62346	0.74013	0.86628	1.00175	1.14637	1.29995	1.46229	1.63318	1.81241	1.99974	2.19494
320	0.61963	0.7356	0.86099	0.99565	1.13941	1.29208	1.45347	1.62337	1.80157	1.98783	2.18193
321	0.61584	0.73111	0.85575	0.98961	1.13251	1.28429	1.44474	1.61365	1.79083	1.97603	2.16904
322	0.61208	0.72665	0.85055	0.98362	1.12568	1.27656	1.43608	1.60402	1.78018	1.96434	2.15626
323	0.60836	0.72224	0.8454	0.97768	1.11891	1.26891	1.4275	1.59447	1.76963	1.95274	2.14359
324	0.60467	0.71787	0.8403	0.97179	1.11219	1.26132	1.41899	1.58501	1.75917	1.94125	2.13103
325	0.60101	0.71354	0.83524	0.96596	1.10554	1.2538	1.41056	1.57563	1.7488	1.92985	2.11857
326	0.59739	0.70925	0.83023	0.96018	1.09895	1.24635	1.40221	1.56633	1.73852	1.91856	2.10623
327	0.59379	0.70499	0.82526	0.95446	1.09241	1.23896	1.39392	1.55711	1.72833	1.90736	2.09399
328	0.59023	0.70078	0.82034	0.94878	1.08593	1.23164	1.38571	1.54798	1.71823	1.89626	2.08185
329	0.58671	0.6966	0.81546	0.94315	1.07951	1.22438	1.37757	1.53892	1.70821	1.88525	2.06982
330	0.58321	0.69246	0.81063	0.93757	1.07315	1.21718	1.36951	1.52994	1.69828	1.87434	2.05789
331	0.57974	0.68835	0.80583	0.93205	1.06684	1.21005	1.36151	1.52103	1.68844	1.86352	2.04606

332	0.57631	0.68428	0.80108	0.92656	1.06058	1.20298	1.35358	1.51221	1.67868	1.85279	2.03434
333	0.5729	0.68025	0.79637	0.92113	1.05438	1.19597	1.34572	1.50346	1.669	1.84215	2.02271
334	0.56953	0.67625	0.7917	0.91575	1.04824	1.18902	1.33793	1.49478	1.65941	1.8316	2.01117
335	0.56619	0.67229	0.78708	0.91041	1.04215	1.18213	1.3302	1.48618	1.64989	1.82115	1.99974
336	0.56287	0.66836	0.78249	0.90512	1.03611	1.1753	1.32254	1.47765	1.64046	1.81078	1.9884
337	0.55958	0.66447	0.77794	0.89987	1.03012	1.16853	1.31494	1.4692	1.63111	1.80049	1.97715
338	0.55633	0.66061	0.77343	0.89467	1.02418	1.16181	1.30741	1.46081	1.62183	1.79029	1.966
339	0.5531	0.65678	0.76897	0.88951	1.01829	1.15516	1.29995	1.4525	1.61264	1.78018	1.95494
340	0.54989	0.65299	0.76453	0.8844	1.01246	1.14856	1.29254	1.44425	1.60352	1.77015	1.94398
341	0.54672	0.64923	0.76014	0.87933	1.00667	1.14201	1.2852	1.43608	1.59447	1.76021	1.9331
342	0.54357	0.6455	0.75579	0.87431	1.00093	1.13552	1.27792	1.42797	1.58551	1.75035	1.92231
343	0.54045	0.6418	0.75147	0.86933	0.99525	1.12909	1.2707	1.41993	1.57661	1.74057	1.91161
344	0.53736	0.63814	0.74718	0.86438	0.98961	1.12271	1.26355	1.41196	1.56779	1.73087	1.901
345	0.53429	0.6345	0.74294	0.85949	0.98401	1.11638	1.25645	1.40406	1.55905	1.72125	1.89048
346	0.53125	0.6309	0.73873	0.85463	0.97847	1.11011	1.24941	1.39622	1.55037	1.71171	1.88004
347	0.52824	0.62733	0.73456	0.84981	0.97297	1.10389	1.24243	1.38844	1.54177	1.70224	1.86969
348	0.52525	0.62378	0.73042	0.84504	0.96751	1.09772	1.23551	1.38073	1.53324	1.69286	1.85942
349	0.52228	0.62027	0.72631	0.8403	0.9621	1.0916	1.22864	1.37308	1.52477	1.68355	1.84923
350	0.51934	0.61679	0.72224	0.8356	0.95674	1.08553	1.22183	1.3655	1.51638	1.67431	1.83913
351	0.51643	0.61333	0.71821	0.83094	0.95142	1.07951	1.21508	1.35798	1.50805	1.66515	1.82911
352	0.51354	0.60991	0.7142	0.82632	0.94615	1.07354	1.20838	1.35051	1.4998	1.65607	1.81916
353	0.51067	0.60651	0.71024	0.82174	0.94091	1.06762	1.20174	1.34311	1.4916	1.64705	1.8093
354	0.50783	0.60314	0.7063	0.8172	0.93573	1.06175	1.19515	1.33577	1.48348	1.63811	1.79952
355	0.50501	0.5998	0.70239	0.81269	0.93058	1.05593	1.18861	1.32849	1.47542	1.62925	1.78981
356	0.50221	0.59649	0.69852	0.80822	0.92548	1.05015	1.18213	1.32127	1.46742	1.62045	1.78018
357	0.49944	0.5932	0.69468	0.80379	0.92041	1.04442	1.1757	1.3141	1.45949	1.61172	1.77063
358	0.49669	0.58994	0.69088	0.7994	0.91539	1.03874	1.16932	1.307	1.45163	1.60306	1.76115
359	0.49396	0.58671	0.6871	0.79504	0.91041	1.0331	1.16299	1.29995	1.44382	1.59447	1.75175
360	0.49126	0.5835	0.68335	0.79071	0.90547	1.02751	1.15672	1.29295	1.43608	1.58595	1.74242
361	0.48858	0.58032	0.67963	0.78642	0.90057	1.02197	1.15049	1.28601	1.4284	1.5775	1.73317
362	0.48592	0.57717	0.67595	0.78216	0.89571	1.01647	1.14432	1.27913	1.42078	1.56911	1.72399
363	0.48328	0.57404	0.67229	0.77794	0.89089	1.01101	1.13819	1.2723	1.41322	1.56079	1.71488
364	0.48066	0.57093	0.66866	0.77375	0.8861	1.00559	1.13211	1.26553	1.40572	1.55253	1.70584
365	0.47806	0.56785	0.66507	0.7696	0.88136	1.00022	1.12608	1.25881	1.39827	1.54434	1.69687
366	0.47549	0.5648	0.6615	0.76548	0.87665	0.99489	1.1201	1.25214	1.39089	1.53621	1.68797
367	0.47293	0.56177	0.65796	0.76139	0.87198	0.98961	1.11416	1.24552	1.38356	1.52815	1.67914
368	0.4704	0.55877	0.65444	0.75734	0.86734	0.98436	1.10827	1.23896	1.3763	1.52015	1.67038
369	0.46788	0.55578	0.65096	0.75331	0.86275	0.97916	1.10243	1.23245	1.36908	1.51221	1.66168
370	0.46539	0.55283	0.6475	0.74932	0.85819	0.97399	1.09663	1.22599	1.36193	1.50433	1.65305
371	0.46291	0.54989	0.64407	0.74536	0.85366	0.96887	1.09088	1.21957	1.35483	1.49651	1.64449
372	0.46046	0.54698	0.64067	0.74143	0.84917	0.96379	1.08517	1.21321	1.34778	1.48875	1.636
373	0.45802	0.54409	0.63729	0.73753	0.84472	0.95875	1.07951	1.2069	1.34079	1.48105	1.62757
374	0.45561	0.54123	0.63395	0.73367	0.8403	0.95374	1.07389	1.20063	1.33385	1.47341	1.6192
375	0.45321	0.53839	0.63062	0.72983	0.83591	0.94878	1.06832	1.19442	1.32697	1.46583	1.61089

376	0.45083	0.53557	0.62733	0.72602	0.83156	0.94385	1.06278	1.18825	1.32013	1.45831	1.60265
377	0.44847	0.53277	0.62405	0.72224	0.82725	0.93896	1.05729	1.18213	1.31335	1.45084	1.59447
378	0.44613	0.52999	0.62081	0.71849	0.82296	0.93411	1.05185	1.17606	1.30662	1.44343	1.58636
379	0.44381	0.52724	0.61759	0.71478	0.81871	0.9293	1.04644	1.17003	1.29995	1.43608	1.5783
380	0.4415	0.5245	0.61439	0.71108	0.81449	0.92452	1.04108	1.16405	1.29332	1.42878	1.57031
381	0.43921	0.52179	0.61122	0.70742	0.81031	0.91978	1.03575	1.15811	1.28674	1.42154	1.56237
382	0.43694	0.5191	0.60807	0.70379	0.80615	0.91508	1.03047	1.15222	1.28021	1.41435	1.55449
383	0.43469	0.51643	0.60495	0.70018	0.80203	0.91041	1.02522	1.14637	1.27374	1.40721	1.54668
384	0.43246	0.51378	0.60185	0.6966	0.79794	0.90578	1.02002	1.14057	1.26731	1.40013	1.53892
385	0.43024	0.51115	0.59878	0.69305	0.79388	0.90118	1.01486	1.13481	1.26092	1.3931	1.53122
386	0.42804	0.50854	0.59572	0.68952	0.78985	0.89662	1.00973	1.12909	1.25459	1.38612	1.52357
387	0.42585	0.50595	0.59269	0.68602	0.78585	0.89209	1.00464	1.12341	1.2483	1.3792	1.51598
388	0.42369	0.50338	0.58969	0.68255	0.78188	0.88759	0.99959	1.11778	1.24206	1.37232	1.50845
389	0.42154	0.50082	0.58671	0.67911	0.77794	0.88313	0.99458	1.11219	1.23587	1.3655	1.50097
390	0.4194	0.49829	0.58375	0.67569	0.77403	0.8787	0.98961	1.10665	1.22972	1.35873	1.49355
391	0.41728	0.49578	0.58081	0.67229	0.77015	0.87431	0.98467	1.10114	1.22362	1.352	1.48618
392	0.41518	0.49329	0.57789	0.66892	0.7663	0.86995	0.97977	1.09567	1.21756	1.34533	1.47887
393	0.41309	0.49081	0.575	0.66558	0.76248	0.86562	0.9749	1.09024	1.21155	1.3387	1.4716
394	0.41102	0.48835	0.57212	0.66226	0.75869	0.86132	0.97007	1.08486	1.20558	1.33212	1.4644
395	0.40897	0.48592	0.56927	0.65897	0.75492	0.85705	0.96528	1.07951	1.19965	1.32559	1.45724
396	0.40693	0.4835	0.56644	0.6557	0.75118	0.85282	0.96052	1.0742	1.19377	1.31911	1.45013
397	0.4049	0.48109	0.56363	0.65245	0.74747	0.84861	0.9558	1.06893	1.18793	1.31268	1.44308
398	0.40289	0.47871	0.56084	0.64923	0.74379	0.84444	0.95111	1.0637	1.18213	1.30629	1.43608
399	0.4009	0.47634	0.55808	0.64603	0.74013	0.8403	0.94646	1.05851	1.17637	1.29995	1.42913
400	0.39892	0.47399	0.55533	0.64285	0.7365	0.83619	0.94183	1.05336	1.17066	1.29365	1.42222



附件 9：

曲线行驶过程中方向盘转动结束时的 y 坐标与方向盘转速和无人车行驶速度的关系表

	10	11	12	13	14	15	16	17	18	19	20
100	11.8574	12.4949	13.0358	13.479	13.8246	14.0743	14.2309	14.2986	14.2827	14.1896	14.0267
101	11.7829	12.4249	12.9723	13.4239	13.7798	14.0414	14.2111	14.2929	14.2917	14.2136	14.0655
102	11.7089	12.3549	12.9085	13.3682	13.7339	14.0068	14.1893	14.2848	14.2979	14.2345	14.101
103	11.6352	12.2851	12.8444	13.3118	13.6869	13.9708	14.1655	14.2743	14.3014	14.2523	14.1333
104	11.562	12.2154	12.7801	13.2548	13.639	13.9334	14.1399	14.2615	14.3023	14.2673	14.1623
105	11.4892	12.1458	12.7157	13.1973	13.5901	13.8946	14.1126	14.2467	14.3008	14.2794	14.1884
106	11.4169	12.0764	12.6512	13.1394	13.5404	13.8547	14.0836	14.2299	14.2968	14.2889	14.2114
107	11.345	12.0073	12.5865	13.081	13.4899	13.8135	14.0532	14.2111	14.2906	14.2958	14.2317
108	11.2737	11.9384	12.5219	13.0223	13.4387	13.7713	14.0212	14.1905	14.2822	14.3002	14.2492
109	11.2029	11.8698	12.4572	12.9632	13.3869	13.7281	13.9879	14.1682	14.2718	14.3022	14.264
110	11.1325	11.8015	12.3926	12.9039	13.3344	13.684	13.9534	14.1443	14.2593	14.3019	14.2762
111	11.0627	11.7335	12.3281	12.8444	13.2815	13.639	13.9176	14.1188	14.245	14.2995	14.286
112	10.9935	11.6658	12.2636	12.7847	13.228	13.5932	13.8807	14.0918	14.2289	14.2949	14.2935
113	10.9248	11.5985	12.1993	12.7249	13.1742	13.5466	13.8427	14.0635	14.2111	14.2883	14.2986
114	10.8566	11.5316	12.1351	12.665	13.1199	13.4994	13.8037	14.0338	14.1917	14.2798	14.3015
115	10.789	11.465	12.0711	12.605	13.0654	13.4516	13.7638	14.0029	14.1707	14.2694	14.3023
116	10.7219	11.3989	12.0073	12.545	13.0105	13.4031	13.723	13.9708	14.1482	14.2573	14.3011
117	10.6554	11.3331	11.9437	12.4849	12.9553	13.3542	13.6814	13.9376	14.1243	14.2435	14.298
118	10.5894	11.2678	11.8803	12.4249	12.9	13.3047	13.639	13.9034	14.0991	14.2281	14.2929
119	10.5241	11.2029	11.8172	12.3649	12.8444	13.2548	13.5959	13.8681	14.0726	14.2111	14.2861
120	10.4592	11.1384	11.7544	12.305	12.7887	13.2045	13.5521	13.8319	14.0449	14.1927	14.2775
121	10.395	11.0743	11.6918	12.2452	12.7329	13.1539	13.5078	13.7949	14.0161	14.1728	14.2673
122	10.3313	11.0107	11.6296	12.1855	12.677	13.1029	13.4629	13.757	13.9862	14.1516	14.2554
123	10.2682	10.9476	11.5676	12.126	12.621	13.0517	13.4174	13.7184	13.9552	14.1292	14.2421
124	10.2057	10.8849	11.506	12.0665	12.565	13.0002	13.3715	13.679	13.9233	14.1055	14.2273
125	10.1437	10.8227	11.4446	12.0073	12.5089	12.9484	13.3251	13.639	13.8905	14.0807	14.2111
126	10.0823	10.7609	11.3837	11.9482	12.4529	12.8965	13.2783	13.5983	13.8568	14.0547	14.1936
127	10.0214	10.6997	11.323	11.8894	12.3969	12.8444	13.2312	13.557	13.8223	14.0277	14.1748
128	9.96115	10.6388	11.2628	11.8307	12.341	12.7922	13.1837	13.5152	13.787	13.9997	14.1547
129	9.90142	10.5785	11.2029	11.7723	12.2851	12.7399	13.1359	13.4729	13.751	13.9708	14.1336
130	9.84225	10.5186	11.1433	11.7141	12.2293	12.6875	13.0879	13.4301	13.7143	13.941	14.1112
131	9.78363	10.4592	11.0842	11.6562	12.1736	12.635	13.0396	13.3869	13.6769	13.9103	14.0879
132	9.72556	10.4003	11.0254	11.5985	12.118	12.5825	12.991	13.3432	13.639	13.8788	14.0635
133	9.66804	10.3419	10.967	11.5411	12.0626	12.53	12.9423	13.2992	13.6005	13.8465	14.0381
134	9.61106	10.2839	10.909	11.484	12.0073	12.4774	12.8934	13.2548	13.5614	13.8135	14.0119
135	9.55463	10.2265	10.8514	11.4272	11.9522	12.4249	12.8444	13.2101	13.5219	13.7798	13.9847
136	9.49872	10.1694	10.7941	11.3706	11.8972	12.3724	12.7953	13.1652	13.4819	13.7455	13.9567
137	9.44335	10.1129	10.7373	11.3144	11.8424	12.32	12.746	13.1199	13.4414	13.7106	13.9279
138	9.3885	10.0569	10.6809	11.2585	11.7879	12.2676	12.6967	13.0745	13.4006	13.675	13.8984
139	9.33418	10.0013	10.6249	11.2029	11.7335	12.2154	12.6474	13.0288	13.3593	13.639	13.8681
140	9.28038	9.94616	10.5693	11.1476	11.6793	12.1632	12.5979	12.9829	13.3178	13.6024	13.8372
141	9.22709	9.89152	10.514	11.0926	11.6254	12.1111	12.5485	12.9369	13.2759	13.5653	13.8056
142	9.17431	9.83734	10.4592	11.0379	11.5717	12.0591	12.4991	12.8907	13.2337	13.5278	13.7733

143	9.12203	9.78363	10.4048	10.9836	11.5183	12.0073	12.4496	12.8444	13.1912	13.4899	13.7406
144	9.07025	9.73038	10.3508	10.9296	11.465	11.9556	12.4002	12.798	13.1485	13.4516	13.7072
145	9.01897	9.67759	10.2973	10.876	11.4121	11.9041	12.3508	12.7515	13.1056	13.4129	13.6733
146	8.96819	9.62526	10.2441	10.8227	11.3594	11.8527	12.3015	12.705	13.0625	13.3738	13.639
147	8.91788	9.57338	10.1913	10.7697	11.3069	11.8015	12.2523	12.6583	13.0192	13.3344	13.6042
148	8.86806	9.52195	10.1389	10.7171	11.2548	11.7505	12.2031	12.6117	12.9757	13.2948	13.5689
149	8.81871	9.47097	10.087	10.6648	11.2029	11.6996	12.154	12.565	12.9321	13.2548	13.5332
150	8.76984	9.42043	10.0354	10.6129	11.1512	11.649	12.105	12.5183	12.8883	13.2146	13.4971
151	8.72143	9.37034	9.98427	10.5613	11.0999	11.5985	12.0561	12.4716	12.8444	13.1742	13.4607
152	8.67348	9.32068	9.93351	10.5101	11.0488	11.5483	12.0073	12.4249	12.8005	13.1335	13.4239
153	8.62599	9.27146	9.88316	10.4592	10.9981	11.4983	11.9586	12.3783	12.7564	13.0927	13.3869
154	8.57895	9.22267	9.8332	10.4087	10.9476	11.4485	11.9101	12.3316	12.7123	13.0517	13.3495
155	8.53236	9.17431	9.78363	10.3585	10.8974	11.3989	11.8618	12.2851	12.6682	13.0105	13.3118
156	8.48621	9.12637	9.73446	10.3087	10.8475	11.3495	11.8135	12.2386	12.624	12.9691	13.2739
157	8.4405	9.07885	9.68568	10.2592	10.7979	11.3004	11.7655	12.1921	12.5797	12.9277	13.2357
158	8.39522	9.03175	9.6373	10.2101	10.7486	11.2515	11.7176	12.1458	12.5355	12.8861	13.1973
159	8.35037	8.98506	9.58929	10.1613	10.6997	11.2029	11.6698	12.0995	12.4913	12.8444	13.1587
160	8.30594	8.93878	9.54168	10.1129	10.651	11.1545	11.6223	12.0534	12.447	12.8027	13.1199
161	8.26193	8.89291	9.49444	10.0648	10.6026	11.1063	11.5749	12.0073	12.4028	12.7608	13.081
162	8.21834	8.84744	9.44759	10.0171	10.5545	11.0584	11.5277	11.9613	12.3586	12.7189	13.0419
163	8.17515	8.80237	9.40111	9.96973	10.5067	11.0107	11.4807	11.9155	12.3145	12.677	13.0026
164	8.13236	8.75769	9.35501	9.92269	10.4592	10.9633	11.4339	11.8698	12.2704	12.635	12.9632
165	8.08998	8.71341	9.30928	9.87599	10.4121	10.9162	11.3872	11.8242	12.2264	12.593	12.9237
166	8.04799	8.66951	9.26393	9.82964	10.3652	10.8693	11.3408	11.7788	12.1824	12.551	12.8841
167	8.00639	8.62599	9.21893	9.78363	10.3186	10.8227	11.2946	11.7335	12.1385	12.5089	12.8444
168	7.96518	8.58286	9.17431	9.73796	10.2724	10.7763	11.2486	11.6884	12.0947	12.4669	12.8047
169	7.92435	8.5401	9.13004	9.69263	10.2265	10.7302	11.2029	11.6434	12.0509	12.4249	12.7648
170	7.88389	8.49771	9.08613	9.64763	10.1808	10.6844	11.1573	11.5985	12.0073	12.3829	12.7249
171	7.84381	8.45569	9.04258	9.60297	10.1355	10.6388	11.112	11.5539	11.9638	12.341	12.685
172	7.8041	8.41404	8.99938	9.55864	10.0904	10.5935	11.0668	11.5094	11.9203	12.299	12.645
173	7.76475	8.37274	8.95653	9.51464	10.0457	10.5485	11.0219	11.465	11.877	12.2572	12.605
174	7.72576	8.33181	8.91403	9.47097	10.0013	10.5037	10.9773	11.4209	11.8338	12.2154	12.565
175	7.68713	8.29123	8.87187	9.42763	9.95715	10.4592	10.9328	11.3769	11.7907	12.1736	12.525
176	7.64884	8.25099	8.83006	9.38461	9.91332	10.415	10.8886	11.3331	11.7478	12.1319	12.4849
177	7.61091	8.21111	8.78858	9.34191	9.86979	10.371	10.8446	11.2895	11.705	12.0903	12.4449
178	7.57332	8.17157	8.74744	9.29953	9.82657	10.3274	10.8008	11.2461	11.6623	12.0487	12.4049
179	7.53606	8.13236	8.70663	9.25748	9.78363	10.2839	10.7573	11.2029	11.6198	12.0073	12.3649
180	7.49915	8.0935	8.66615	9.21573	9.741	10.2408	10.714	11.1598	11.5774	11.9659	12.325
181	7.46256	8.05496	8.62599	9.17431	9.69865	10.1979	10.671	11.117	11.5351	11.9247	12.2851
182	7.4263	8.01676	8.58616	9.13319	9.6566	10.1553	10.6282	11.0743	11.493	11.8835	12.2452
183	7.39037	7.97887	8.54665	9.09238	9.61485	10.1129	10.5856	11.0319	11.4511	11.8424	12.2054
184	7.35475	7.94131	8.50746	9.05188	9.57338	10.0708	10.5432	10.9896	11.4093	11.8015	12.1656
185	7.31945	7.90407	8.46858	9.01169	9.5322	10.029	10.5011	10.9476	11.3677	11.7607	12.126
186	7.28446	7.86715	8.43002	8.9718	9.49131	9.98745	10.4592	10.9058	11.3262	11.7199	12.0863

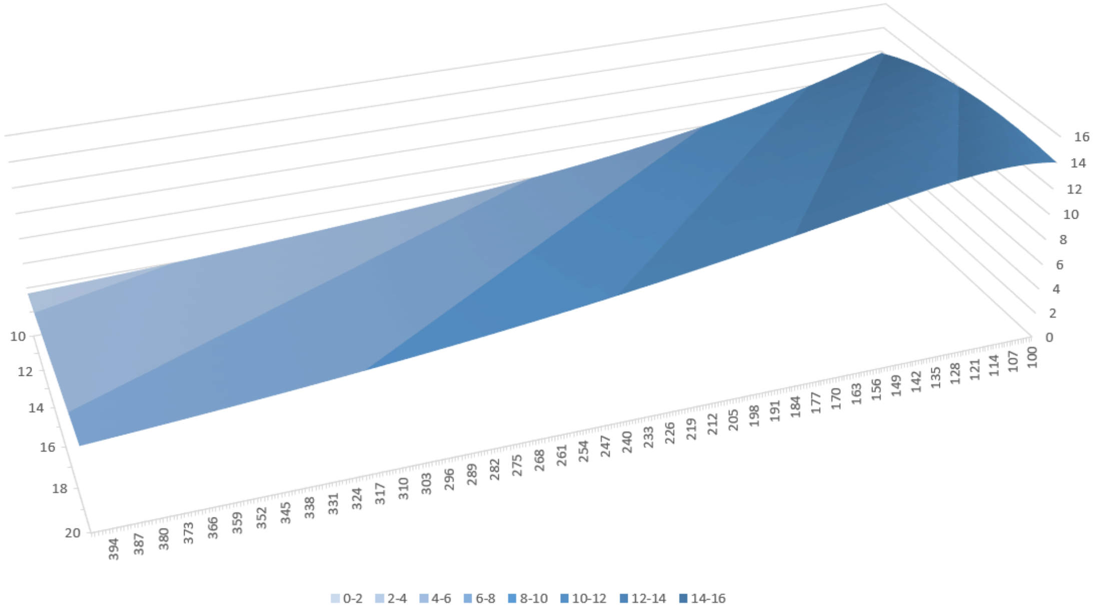
187	7.24979	7.83053	8.39176	8.9322	9.4507	9.94616	10.4176	10.8641	11.2849	11.6793	12.0468
188	7.21541	7.79423	8.35381	8.89291	9.41038	9.90514	10.3762	10.8227	11.2438	11.6389	12.0073
189	7.18134	7.75823	8.31616	8.85391	9.37034	9.86437	10.335	10.7815	11.2029	11.5985	11.9679
190	7.14757	7.72253	8.27881	8.81521	9.33058	9.82387	10.2941	10.7405	11.1621	11.5583	11.9286
191	7.11409	7.68713	8.24176	8.77679	9.2911	9.78363	10.2534	10.6997	11.1215	11.5183	11.8894
192	7.0809	7.65202	8.205	8.73866	9.25189	9.74365	10.213	10.6591	11.0811	11.4783	11.8503
193	7.048	7.61721	8.16854	8.70082	9.21296	9.70393	10.1728	10.6187	11.0408	11.4385	11.8112
194	7.01538	7.58268	8.13236	8.66327	9.17431	9.66447	10.1328	10.5785	11.0007	11.3989	11.7723
195	6.98304	7.54844	8.09647	8.62599	9.13592	9.62526	10.0931	10.5385	10.9609	11.3594	11.7335
196	6.95099	7.51449	8.06087	8.589	9.09781	9.58631	10.0536	10.4988	10.9212	11.32	11.6948
197	6.9192	7.48081	8.02554	8.55228	9.05996	9.54761	10.0143	10.4592	10.8816	11.2808	11.6562
198	6.88769	7.44741	7.9905	8.51583	9.02238	9.50916	9.97528	10.4199	10.8423	11.2418	11.6177
199	6.85644	7.41429	7.95572	8.47966	8.98506	9.47097	9.93649	10.3808	10.8031	11.2029	11.5794
200	6.82546	7.38143	7.92122	8.44375	8.94801	9.43303	9.89793	10.3419	10.7642	11.1641	11.5411
201	6.79475	7.34885	7.88699	8.40812	8.91121	9.39533	9.8596	10.3032	10.7254	11.1255	11.503
202	6.76429	7.31652	7.85303	8.37274	8.87467	9.35788	9.8215	10.2647	10.6868	11.0871	11.465
203	6.73408	7.28446	7.81933	8.33763	8.83839	9.32068	9.78363	10.2265	10.6484	11.0488	11.4272
204	6.70413	7.25266	7.78589	8.30279	8.80237	9.28372	9.746	10.1884	10.6102	11.0107	11.3895
205	6.67443	7.22112	7.75271	8.26819	8.7666	9.24701	9.70859	10.1505	10.5722	10.9728	11.3519
206	6.64498	7.18983	7.71979	8.23386	8.73107	9.21054	9.67141	10.1129	10.5343	10.935	11.3144
207	6.61577	7.15879	7.68713	8.19978	8.6958	9.17431	9.63446	10.0755	10.4967	10.8974	11.2771
208	6.58681	7.128	7.65471	8.16595	8.66077	9.13831	9.59774	10.0383	10.4592	10.86	11.2399
209	6.55808	7.09746	7.62254	8.13236	8.62599	9.10256	9.56124	10.0013	10.422	10.8227	11.2029
210	6.52959	7.06715	7.59062	8.09903	8.59145	9.06704	9.52496	9.96448	10.3849	10.7856	11.166
211	6.50133	7.03709	7.55895	8.06594	8.55716	9.03175	9.48891	9.9279	10.348	10.7486	11.1292
212	6.4733	7.00727	7.52751	8.03309	8.5231	8.99669	9.45308	9.89152	10.3113	10.7119	11.0926
213	6.4455	6.97768	7.49632	8.00048	8.48928	8.96187	9.41747	9.85535	10.2748	10.6753	11.0561
214	6.41792	6.94833	7.46536	7.96811	8.45569	8.92728	9.38209	9.81939	10.2385	10.6388	11.0198
215	6.39057	6.9192	7.43464	7.93597	8.42234	8.89291	9.34692	9.78363	10.2024	10.6026	10.9836
216	6.36344	6.89031	7.40415	7.90407	8.38922	8.85877	9.31196	9.74808	10.1665	10.5665	10.9476
217	6.33653	6.86163	7.37389	7.8724	8.35633	8.82486	9.27723	9.71273	10.1307	10.5306	10.9117
218	6.30983	6.83319	7.34386	7.84096	8.32366	8.79116	9.24271	9.67759	10.0952	10.4948	10.876
219	6.28335	6.80496	7.31405	7.80975	8.29123	8.75769	9.2084	9.64265	10.0598	10.4592	10.8404
220	6.25707	6.77695	7.28446	7.77876	8.25901	8.72444	9.17431	9.60792	10.0246	10.4238	10.805
221	6.23101	6.74915	7.2551	7.748	8.22702	8.69141	9.14042	9.57338	9.98964	10.3886	10.7697
222	6.20515	6.72158	7.22596	7.71745	8.19525	8.65859	9.10675	9.53904	9.95484	10.3535	10.7346
223	6.17949	6.69421	7.19703	7.68713	8.1637	8.62599	9.07329	9.50491	9.92022	10.3186	10.6997
224	6.15404	6.66705	7.16832	7.65702	8.13236	8.59361	9.04003	9.47097	9.8858	10.2839	10.6648
225	6.12879	6.6401	7.13981	7.62712	8.10124	8.56143	9.00698	9.43723	9.85156	10.2494	10.6302
226	6.10373	6.61335	7.11152	7.59744	8.07034	8.52947	8.97414	9.40369	9.8175	10.215	10.5957
227	6.07887	6.58681	7.08344	7.56797	8.03964	8.49771	8.94149	9.37034	9.78363	10.1808	10.5613
228	6.0542	6.56046	7.05556	7.53871	8.00915	8.46616	8.90905	9.33719	9.74995	10.1468	10.5272
229	6.02972	6.53432	7.02789	7.50966	7.97887	8.43482	8.87682	9.30423	9.71645	10.1129	10.4931
230	6.00543	6.50837	7.00042	7.48081	7.9488	8.40368	8.84478	9.27146	9.68313	10.0792	10.4592

231	5.98133	6.48262	6.97315	7.45217	7.91893	8.37274	8.81294	9.23888	9.64999	10.0457	10.4255
232	5.95741	6.45705	6.94608	7.42372	7.88927	8.34201	8.7813	9.2065	9.61704	10.0124	10.3919
233	5.93367	6.43168	6.9192	7.39548	7.8598	8.31147	8.74985	9.17431	9.58426	9.97918	10.3585
234	5.91012	6.4065	6.89252	7.36743	7.83053	8.28114	8.7186	9.1423	9.55167	9.94616	10.3253
235	5.88674	6.3815	6.86603	7.33958	7.80146	8.25099	8.68754	9.11048	9.51926	9.91332	10.2922
236	5.86354	6.35669	6.83973	7.31193	7.77259	8.22105	8.65667	9.07885	9.48702	9.88065	10.2592
237	5.84051	6.33206	6.81362	7.28446	7.74391	8.1913	8.62599	9.0474	9.45496	9.84814	10.2265
238	5.81766	6.30762	6.7877	7.25719	7.71542	8.16173	8.5955	9.01614	9.42308	9.8158	10.1938
239	5.79498	6.28335	6.76196	7.23011	7.68713	8.13236	8.5652	8.98506	9.39138	9.78363	10.1613
240	5.77247	6.25925	6.7364	7.20321	7.65902	8.10318	8.53509	8.95416	9.35985	9.75163	10.129
241	5.75012	6.23534	6.71102	7.1765	7.6311	8.07419	8.50516	8.92345	9.32849	9.71979	10.0969
242	5.72795	6.21159	6.68583	7.14997	7.60336	8.04538	8.47542	8.89291	9.29731	9.68811	10.0648
243	5.70593	6.18802	6.66081	7.12362	7.57581	8.01676	8.44586	8.86255	9.2663	9.6566	10.033
244	5.68408	6.16462	6.63597	7.09746	7.54844	7.98831	8.41648	8.83237	9.23547	9.62526	10.0013
245	5.66239	6.14139	6.6113	7.07147	7.52126	7.96005	8.38728	8.80237	9.2048	9.59408	9.96973
246	5.64085	6.11832	6.58681	7.04566	7.49425	7.93197	8.35826	8.77254	9.17431	9.56306	9.93833
247	5.61948	6.09542	6.56248	7.02002	7.46742	7.90407	8.32941	8.74289	9.14398	9.5322	9.90709
248	5.59826	6.07268	6.53833	6.99456	7.44077	7.87635	8.30074	8.71341	9.11382	9.5015	9.87599
249	5.57719	6.0501	6.51434	6.96927	7.41429	7.8488	8.27225	8.6841	9.08383	9.47097	9.84506
250	5.55628	6.02769	6.49052	6.94415	7.38798	7.82143	8.24393	8.65496	9.05401	9.44059	9.81427
251	5.53552	6.00543	6.46686	6.9192	7.36185	7.79423	8.21578	8.62599	9.02435	9.41038	9.78363
252	5.5149	5.98333	6.44337	6.89442	7.33589	7.7672	8.18781	8.59719	8.99486	9.38032	9.75315
253	5.49444	5.96138	6.42004	6.8698	7.31009	7.74034	8.16	8.56856	8.96553	9.35042	9.72281
254	5.47412	5.93959	6.39686	6.84535	7.28446	7.71365	8.13236	8.5401	8.93636	9.32068	9.69263
255	5.45394	5.91795	6.37385	6.82106	7.259	7.68713	8.10489	8.5118	8.90735	9.2911	9.66259
256	5.43391	5.89646	6.35099	6.79693	7.23371	7.66077	8.07759	8.48366	8.87851	9.26167	9.63271
257	5.41402	5.87512	6.32829	6.77296	7.20857	7.63458	8.05045	8.45569	8.84982	9.23239	9.60297
258	5.39427	5.85392	6.30574	6.74915	7.1836	7.60855	8.02347	8.42788	8.8213	9.20327	9.57338
259	5.37466	5.83288	6.28335	6.7255	7.15879	7.58268	7.99666	8.40023	8.79293	9.17431	9.54394
260	5.35519	5.81197	6.2611	6.70201	7.13414	7.55698	7.97001	8.37274	8.76472	9.14549	9.51464
261	5.33585	5.79122	6.23901	6.67866	7.10965	7.53143	7.94351	8.34541	8.73667	9.11683	9.48549
262	5.31665	5.7706	6.21706	6.65547	7.08531	7.50604	7.91718	8.31824	8.70877	9.08832	9.45648
263	5.29758	5.75012	6.19526	6.63243	7.06112	7.48081	7.891	8.29123	8.68102	9.05996	9.42763
264	5.27864	5.72979	6.1736	6.60955	7.03709	7.45574	7.86498	8.26437	8.65343	9.03175	9.39891
265	5.25984	5.70959	6.15209	6.58681	7.01322	7.43082	7.83912	8.23766	8.62599	9.00369	9.37034
266	5.24116	5.68953	6.13072	6.56421	6.98949	7.40605	7.81341	8.21111	8.59871	8.97577	9.34191
267	5.22261	5.6696	6.10949	6.54177	6.96591	7.38143	7.78785	8.18471	8.57157	8.94801	9.31362
268	5.20419	5.64981	6.08841	6.51947	6.94248	7.35697	7.76245	8.15846	8.54458	8.92039	9.28548
269	5.1859	5.63015	6.06746	6.49731	6.9192	7.33265	7.73719	8.13236	8.51774	8.89291	9.25748
270	5.16773	5.61062	6.04664	6.47529	6.89607	7.30848	7.71208	8.10642	8.49105	8.86558	9.22961
271	5.14968	5.59122	6.02597	6.45342	6.87308	7.28446	7.68713	8.08061	8.46451	8.83839	9.20189
272	5.13175	5.57195	6.00543	6.43168	6.85023	7.26059	7.66231	8.05496	8.43811	8.81135	9.17431
273	5.11395	5.55281	5.98502	6.41009	6.82752	7.23686	7.63765	8.02945	8.41185	8.78445	9.14686
274	5.09626	5.53379	5.96475	6.38863	6.80496	7.21327	7.61313	8.00409	8.38575	8.75769	9.11955

275	5.07869	5.5149	5.9446	6.3673	6.78253	7.18983	7.58875	7.97887	8.35978	8.73107	9.09238
276	5.06124	5.49614	5.92459	6.34612	6.76025	7.16653	7.56452	7.9538	8.33395	8.7046	9.06535
277	5.04391	5.4775	5.9047	6.32506	6.7381	7.14337	7.54043	7.92887	8.30827	8.67826	9.03845
278	5.02669	5.45897	5.88495	6.30414	6.71608	7.12034	7.51648	7.90407	8.28273	8.65206	9.01169
279	5.00959	5.44057	5.86532	6.28335	6.69421	7.09746	7.49267	7.87942	8.25732	8.62599	8.98506
280	4.99259	5.42229	5.84581	6.26269	6.67246	7.07471	7.46899	7.85491	8.23206	8.60007	8.95857
281	4.97571	5.40413	5.82643	6.24215	6.65085	7.05209	7.44546	7.83053	8.20693	8.57428	8.9322
282	4.95894	5.38609	5.80717	6.22175	6.62937	7.02962	7.42206	7.80629	8.18194	8.54862	8.90598
283	4.94228	5.36816	5.78804	6.20147	6.60803	7.00727	7.39879	7.78219	8.15708	8.5231	8.87988
284	4.92573	5.35034	5.76902	6.18132	6.58681	6.98506	7.37566	7.75823	8.13236	8.49771	8.85391
285	4.90929	5.33264	5.75012	6.16129	6.56572	6.96298	7.35267	7.73439	8.10778	8.47245	8.82808
286	4.89295	5.31506	5.73135	6.14139	6.54475	6.94102	7.3298	7.71069	8.08332	8.44733	8.80237
287	4.87672	5.29758	5.71269	6.12161	6.52392	6.9192	7.30707	7.68713	8.059	8.42234	8.77679
288	4.8606	5.28022	5.69414	6.10195	6.5032	6.89751	7.28446	7.66369	8.03481	8.39748	8.75134
289	4.84457	5.26296	5.67572	6.08241	6.48262	6.87594	7.26199	7.64038	8.01075	8.37274	8.72602
290	4.82865	5.24582	5.6574	6.06299	6.46215	6.8545	7.23964	7.61721	7.98682	8.34814	8.70082
291	4.81283	5.22878	5.6392	6.04368	6.44181	6.83319	7.21743	7.59416	7.96302	8.32366	8.67576
292	4.79712	5.21185	5.62112	6.0245	6.42159	6.81199	7.19533	7.57124	7.93935	8.29931	8.65081
293	4.7815	5.19503	5.60314	6.00543	6.40149	6.79093	7.17337	7.54844	7.9158	8.27509	8.62599
294	4.76598	5.17831	5.58528	5.98647	6.3815	6.76998	7.15152	7.52578	7.89238	8.25099	8.6013
295	4.75056	5.1617	5.56752	5.96764	6.36164	6.74915	7.12981	7.50323	7.86908	8.22702	8.57673
296	4.73523	5.14518	5.54988	5.94891	6.34189	6.72845	7.10821	7.48081	7.84591	8.20317	8.55228
297	4.72001	5.12878	5.53234	5.93029	6.32226	6.70786	7.08673	7.45852	7.82286	8.17945	8.52795
298	4.70487	5.11247	5.5149	5.91179	6.30275	6.6874	7.06538	7.43634	7.79994	8.15585	8.50374
299	4.68984	5.09626	5.49758	5.8934	6.28335	6.66705	7.04415	7.41429	7.77714	8.13236	8.47966
300	4.67489	5.08015	5.48036	5.87512	6.26406	6.64682	7.02303	7.39235	7.75445	8.109	8.45569
301	4.66004	5.06414	5.46324	5.85694	6.24488	6.6267	7.00203	7.37054	7.73189	8.08576	8.43184
302	4.64528	5.04823	5.44622	5.83888	6.22582	6.6067	6.98115	7.34885	7.70945	8.06264	8.40812
303	4.63061	5.03242	5.42931	5.82092	6.20687	6.58681	6.96039	7.32727	7.68713	8.03964	8.38451
304	4.61603	5.0167	5.4125	5.80306	6.18802	6.56703	6.93974	7.30581	7.66492	8.01676	8.36101
305	4.60154	5.00108	5.39579	5.78531	6.16929	6.54737	6.9192	7.28446	7.64283	7.99399	8.33763
306	4.58714	4.98555	5.37918	5.76767	6.15066	6.52781	6.89878	7.26324	7.62086	7.97134	8.31437
307	4.57283	4.97011	5.36266	5.75012	6.13214	6.50837	6.87847	7.24212	7.599	7.9488	8.29123
308	4.55861	4.95477	5.34625	5.73268	6.11373	6.48904	6.85828	7.22112	7.57726	7.92638	8.26819
309	4.54447	4.93952	5.32993	5.71535	6.09542	6.46981	6.83819	7.20023	7.55563	7.90407	8.24528
310	4.53041	4.92436	5.31371	5.69811	6.07722	6.45069	6.81821	7.17946	7.53411	7.88188	8.22247
311	4.51645	4.90929	5.29758	5.68097	6.05912	6.43168	6.79835	7.15879	7.51271	7.8598	8.19978
312	4.50256	4.89431	5.28155	5.66393	6.04112	6.41278	6.77859	7.13824	7.49142	7.83783	8.1772
313	4.48876	4.87942	5.26561	5.64699	6.02322	6.39398	6.75894	7.11779	7.47024	7.81597	8.15472
314	4.47504	4.86462	5.24977	5.63015	6.00543	6.37528	6.7394	7.09746	7.44916	7.79423	8.13236
315	4.4614	4.8499	5.23401	5.6134	5.98773	6.35669	6.71996	7.07723	7.4282	7.77259	8.11011
316	4.44785	4.83527	5.21835	5.59675	5.97014	6.3382	6.70063	7.05711	7.40735	7.75106	8.08797
317	4.43437	4.82073	5.20278	5.58019	5.95264	6.31982	6.6814	7.03709	7.3866	7.72964	8.06594
318	4.42098	4.80627	5.1873	5.56373	5.93525	6.30153	6.66228	7.01718	7.36596	7.70833	8.04401

319	4.40766	4.7919	5.17191	5.54736	5.91795	6.28335	6.64326	6.99738	7.34543	7.68713	8.02219
320	4.39442	4.77761	5.1566	5.53109	5.90074	6.26526	6.62434	6.97768	7.325	7.66603	8.00048
321	4.38126	4.7634	5.14139	5.5149	5.88363	6.24727	6.60552	6.95809	7.30468	7.64503	7.97887
322	4.36818	4.74928	5.12626	5.49881	5.86662	6.22939	6.58681	6.93859	7.28446	7.62415	7.95737
323	4.35517	4.73523	5.11122	5.48281	5.8497	6.21159	6.56819	6.9192	7.26435	7.60336	7.93597
324	4.34224	4.72127	5.09626	5.4669	5.83288	6.1939	6.54967	6.89991	7.24434	7.58268	7.91468
325	4.32939	4.70739	5.08139	5.45107	5.81614	6.1763	6.53125	6.88072	7.22443	7.5621	7.89349
326	4.3166	4.69359	5.0666	5.43534	5.7995	6.1588	6.51293	6.86163	7.20462	7.54163	7.8724
327	4.3039	4.67986	5.05189	5.41969	5.78295	6.14139	6.49471	6.84264	7.18491	7.52126	7.85142
328	4.29126	4.66622	5.03727	5.40413	5.76649	6.12407	6.47658	6.82375	7.16531	7.50098	7.83053
329	4.2787	4.65265	5.02273	5.38866	5.75012	6.10685	6.45855	6.80496	7.1458	7.48081	7.80975
330	4.26621	4.63916	5.00827	5.37327	5.73384	6.08972	6.44062	6.78626	7.12639	7.46074	7.78907
331	4.25379	4.62574	4.9939	5.35796	5.71765	6.07268	6.42277	6.76766	7.10708	7.44077	7.76848
332	4.24144	4.6124	4.9796	5.34274	5.70155	6.05573	6.40502	6.74915	7.08786	7.42089	7.748
333	4.22917	4.59914	4.96538	5.32761	5.68553	6.03888	6.38737	6.73074	7.06874	7.40111	7.72761
334	4.21696	4.58595	4.95124	5.31255	5.6696	6.02211	6.36981	6.71243	7.04972	7.38143	7.70732
335	4.20482	4.57283	4.93718	5.29758	5.65376	6.00543	6.35233	6.69421	7.0308	7.36185	7.68713
336	4.19275	4.55979	4.9232	5.28269	5.638	5.98884	6.33495	6.67608	7.01196	7.34236	7.66703
337	4.18074	4.54682	4.90929	5.26788	5.62232	5.97233	6.31766	6.65804	6.99323	7.32297	7.64703
338	4.16881	4.53392	4.89546	5.25315	5.60673	5.95592	6.30046	6.6401	6.97458	7.30367	7.62712
339	4.15694	4.52109	4.88171	5.2385	5.59122	5.93959	6.28335	6.62224	6.95603	7.28446	7.60731
340	4.14514	4.50834	4.86803	5.22393	5.57579	5.92334	6.26632	6.60448	6.93757	7.26535	7.5876
341	4.1334	4.49565	4.85442	5.20944	5.56045	5.90718	6.24939	6.58681	6.9192	7.24634	7.56797
342	4.12173	4.48303	4.84089	5.19503	5.54519	5.89111	6.23254	6.56922	6.90092	7.22741	7.54844
343	4.11012	4.47049	4.82743	5.18069	5.53001	5.87512	6.21577	6.55173	6.88274	7.20857	7.52901
344	4.09858	4.45801	4.81405	5.16643	5.5149	5.85921	6.19909	6.53432	6.86464	7.18983	7.50966
345	4.0871	4.4456	4.80073	5.15225	5.49988	5.84338	6.1825	6.517	6.84663	7.17118	7.49041
346	4.07568	4.43325	4.78749	5.13814	5.48494	5.82764	6.16599	6.49976	6.82871	7.15261	7.47124
347	4.06433	4.42098	4.77432	5.12411	5.47007	5.81197	6.14957	6.48262	6.81088	7.13414	7.45217
348	4.05304	4.40877	4.76122	5.11015	5.45528	5.79639	6.13323	6.46555	6.79314	7.11575	7.43318
349	4.04181	4.39662	4.74819	5.09626	5.44057	5.78089	6.11697	6.44858	6.77548	7.09746	7.41429
350	4.03064	4.38455	4.73523	5.08245	5.42594	5.76547	6.10079	6.43168	6.75791	7.07925	7.39548
351	4.01953	4.37253	4.72234	5.06871	5.41138	5.75012	6.0847	6.41487	6.74042	7.06112	7.37676
352	4.00848	4.36058	4.70952	5.05504	5.3969	5.73486	6.06869	6.39815	6.72302	7.04309	7.35813
353	3.99749	4.3487	4.69676	5.04144	5.38249	5.71967	6.05275	6.3815	6.7057	7.02514	7.33958
354	3.98656	4.33688	4.68408	5.02792	5.36816	5.70456	6.0369	6.36494	6.68847	7.00727	7.32113
355	3.97569	4.32512	4.67146	5.01446	5.3539	5.68953	6.02112	6.34846	6.67132	6.98949	7.30275
356	3.96487	4.31342	4.6589	5.00108	5.33971	5.67457	6.00543	6.33206	6.65426	6.97179	7.28446
357	3.95412	4.30179	4.64641	4.98776	5.32559	5.65969	5.98981	6.31574	6.63727	6.95418	7.26626
358	3.94342	4.29021	4.63399	4.97451	5.31155	5.64488	5.97427	6.29951	6.62037	6.93665	7.24814
359	3.93278	4.2787	4.62163	4.96133	5.29758	5.63015	5.95881	6.28335	6.60355	6.9192	7.23011
360	3.9222	4.26725	4.60933	4.94822	5.28368	5.61549	5.94342	6.26727	6.58681	6.90184	7.21215
361	3.91167	4.25586	4.5971	4.93518	5.26985	5.6009	5.92811	6.25126	6.57014	6.88455	7.19429
362	3.90119	4.24452	4.58494	4.9222	5.25609	5.58639	5.91288	6.23534	6.55356	6.86735	7.1765

363	3.89077	4.23325	4.57283	4.90929	5.2424	5.57195	5.89772	6.21949	6.53706	6.85023	7.15879
364	3.88041	4.22204	4.56079	4.89645	5.22878	5.55758	5.88263	6.20372	6.52064	6.83319	7.14117
365	3.8701	4.21088	4.54881	4.88367	5.21523	5.54329	5.86762	6.18802	6.50429	6.81622	7.12362
366	3.85985	4.19978	4.53689	4.87095	5.20175	5.52906	5.85268	6.1724	6.48802	6.79934	7.10616
367	3.84964	4.18874	4.52503	4.8583	5.18833	5.5149	5.83782	6.15686	6.47183	6.78253	7.08877
368	3.83949	4.17775	4.51323	4.84571	5.17498	5.50082	5.82302	6.14139	6.45571	6.7658	7.07147
369	3.8294	4.16683	4.5015	4.83319	5.1617	5.4868	5.8083	6.12599	6.43967	6.74915	7.05424
370	3.81935	4.15595	4.48982	4.82073	5.14848	5.47285	5.79365	6.11067	6.42371	6.73258	7.03709
371	3.80936	4.14514	4.4782	4.80833	5.13533	5.45897	5.77907	6.09542	6.40782	6.71608	7.02002
372	3.79942	4.13438	4.46664	4.796	5.12224	5.44516	5.76456	6.08024	6.39201	6.69966	7.00303
373	3.78953	4.12367	4.45514	4.78372	5.10922	5.43142	5.75012	6.06514	6.37627	6.68332	6.98611
374	3.77969	4.11302	4.44369	4.77151	5.09626	5.41774	5.73575	6.0501	6.3606	6.66705	6.96927
375	3.7699	4.10242	4.43231	4.75936	5.08337	5.40413	5.72145	6.03514	6.345	6.65085	6.95251
376	3.76016	4.09187	4.42098	4.74727	5.07053	5.39059	5.70722	6.02025	6.32948	6.63473	6.93582
377	3.75047	4.08138	4.40971	4.73523	5.05777	5.37711	5.69306	6.00543	6.31403	6.61868	6.9192
378	3.74083	4.07094	4.39849	4.72326	5.04506	5.36369	5.67896	5.99068	6.29865	6.60271	6.90266
379	3.73124	4.06056	4.38733	4.71135	5.03242	5.35034	5.66493	5.97599	6.28335	6.58681	6.8862
380	3.72169	4.05022	4.37622	4.69949	5.01983	5.33706	5.65097	5.96138	6.26811	6.57098	6.8698
381	3.7122	4.03994	4.36517	4.68769	5.00731	5.32384	5.63707	5.94683	6.25294	6.55522	6.85348
382	3.70275	4.02971	4.35418	4.67596	4.99485	5.31068	5.62324	5.93236	6.23785	6.53953	6.83724
383	3.69335	4.01953	4.34323	4.66427	4.98245	5.29758	5.60947	5.91795	6.22282	6.52392	6.82106
384	3.68399	4.0094	4.33235	4.65265	4.97011	5.28455	5.59577	5.9036	6.20786	6.50837	6.80496
385	3.67469	3.99932	4.32151	4.64108	4.95783	5.27158	5.58214	5.88933	6.19297	6.49289	6.78893
386	3.66543	3.98929	4.31073	4.62957	4.94561	5.25867	5.56856	5.87512	6.17815	6.47749	6.77296
387	3.65621	3.97931	4.3	4.61811	4.93344	5.24582	5.55505	5.86097	6.16339	6.46215	6.75707
388	3.64704	3.96937	4.28932	4.60671	4.92134	5.23303	5.54161	5.84689	6.14871	6.44688	6.74125
389	3.63792	3.95949	4.2787	4.59536	4.90929	5.2203	5.52823	5.83288	6.13409	6.43168	6.7255
390	3.62884	3.94965	4.26813	4.58407	4.8973	5.20764	5.5149	5.81893	6.11953	6.41655	6.70982
391	3.6198	3.93987	4.2576	4.57283	4.88537	5.19503	5.50164	5.80504	6.10504	6.40149	6.69421
392	3.61081	3.93013	4.24713	4.56165	4.87349	5.18248	5.48845	5.79122	6.09062	6.38649	6.67866
393	3.60187	3.92044	4.23671	4.55052	4.86167	5.16999	5.47531	5.77746	6.07626	6.37156	6.66319
394	3.59297	3.91079	4.22634	4.53944	4.8499	5.15756	5.46223	5.76376	6.06197	6.35669	6.64778
395	3.58411	3.90119	4.21602	4.52841	4.83819	5.14518	5.44922	5.75012	6.04774	6.34189	6.63243
396	3.57529	3.89164	4.20575	4.51744	4.82654	5.13287	5.43626	5.73655	6.03357	6.32716	6.61716
397	3.56652	3.88213	4.19553	4.50652	4.81494	5.12061	5.42336	5.72304	6.01947	6.31249	6.60195
398	3.55779	3.87267	4.18535	4.49565	4.80339	5.10841	5.41053	5.70959	6.00543	6.29789	6.58681
399	3.5491	3.86326	4.17523	4.48483	4.7919	5.09626	5.39775	5.6962	5.99145	6.28335	6.57173
400	3.54045	3.85389	4.16515	4.47406	4.78046	5.08417	5.38503	5.68287	5.97754	6.26887	6.55672



附件 10：
曲线行驶过程中方向盘转动结束时的斜率与方向盘转速和无人车行驶速度的关系表

	10	11	12	13	14	15	16	17	18	19	20
100	0.21069	0.08393	-0.04021	-0.1656	-0.29628	-0.43696	-0.59384	-0.77579	-0.99666	-1.28014	-1.67118
101	0.22469	0.09863	-0.0244	-0.14818	-0.27656	-0.414	-0.56624	-0.74134	-0.95167	-1.21795	-1.57835
102	0.2385	0.11308	-0.00891	-0.13117	-0.25741	-0.39185	-0.53979	-0.70862	-0.90944	-1.16043	-1.49431
103	0.25212	0.12731	0.00628	-0.11457	-0.23881	-0.37044	-0.51441	-0.67749	-0.86967	-1.10703	-1.41777
104	0.26557	0.14131	0.02118	-0.09834	-0.22071	-0.34972	-0.49001	-0.6478	-0.83213	-1.05727	-1.3477
105	0.27886	0.1551	0.03581	-0.08246	-0.20308	-0.32965	-0.46652	-0.61943	-0.7966	-1.01074	-1.28323
106	0.29199	0.16869	0.05019	-0.06693	-0.1859	-0.31019	-0.44388	-0.59228	-0.76288	-0.96709	-1.22366
107	0.30496	0.18209	0.06431	-0.05171	-0.16914	-0.29129	-0.42201	-0.56624	-0.73083	-0.92602	-1.16838
108	0.31779	0.1953	0.0782	-0.0368	-0.15278	-0.27292	-0.40088	-0.54124	-0.70028	-0.88728	-1.11691
109	0.33048	0.20834	0.09187	-0.02217	-0.13679	-0.25506	-0.38042	-0.51718	-0.67112	-0.85064	-1.06881
110	0.34304	0.22121	0.10533	-0.00782	-0.12116	-0.23766	-0.3606	-0.49401	-0.64324	-0.8159	-1.02373
111	0.35546	0.23391	0.11858	0.00628	-0.10587	-0.22071	-0.34138	-0.47167	-0.61652	-0.7829	-0.98134

112	0.36776	0.24646	0.13164	0.02013	-0.09089	-0.20417	-0.32271	-0.45009	-0.59088	-0.75148	-0.94138
113	0.37994	0.25887	0.14451	0.03374	-0.07621	-0.18802	-0.30457	-0.42922	-0.56624	-0.72151	-0.90361
114	0.39201	0.27113	0.15721	0.04713	-0.06182	-0.17225	-0.28692	-0.40901	-0.54253	-0.69287	-0.86784
115	0.40397	0.28325	0.16973	0.0603	-0.04771	-0.15684	-0.26974	-0.38943	-0.51967	-0.66545	-0.83387
116	0.41581	0.29525	0.18209	0.07327	-0.03385	-0.14175	-0.25299	-0.37044	-0.49762	-0.63916	-0.80156
117	0.42756	0.30711	0.19429	0.08604	-0.02024	-0.12698	-0.23665	-0.35199	-0.47631	-0.61391	-0.77075
118	0.43921	0.31886	0.20634	0.09863	-0.00687	-0.11252	-0.22071	-0.33406	-0.45569	-0.58962	-0.74134
119	0.45076	0.33048	0.21825	0.11103	0.00628	-0.09834	-0.20513	-0.31661	-0.43573	-0.56624	-0.7132
120	0.46221	0.342	0.23002	0.12327	0.01921	-0.08443	-0.1899	-0.29962	-0.41638	-0.54369	-0.68623
121	0.47358	0.3534	0.24166	0.13534	0.03194	-0.07078	-0.17501	-0.28306	-0.3976	-0.52192	-0.66036
122	0.48487	0.3647	0.25316	0.14725	0.04447	-0.05738	-0.16043	-0.26691	-0.37936	-0.50088	-0.63549
123	0.49606	0.3759	0.26454	0.159	0.05681	-0.04422	-0.14615	-0.25115	-0.36163	-0.48051	-0.61155
124	0.50718	0.387	0.27581	0.17062	0.06897	-0.03128	-0.13216	-0.23575	-0.34438	-0.46078	-0.58849
125	0.51822	0.398	0.28696	0.18209	0.08096	-0.01856	-0.11844	-0.22071	-0.32757	-0.44165	-0.56624
126	0.52918	0.40892	0.298	0.19342	0.09278	-0.00604	-0.10497	-0.20599	-0.3112	-0.42309	-0.54475
127	0.54007	0.41974	0.30893	0.20463	0.10444	0.00628	-0.09176	-0.19158	-0.29522	-0.40505	-0.52396
128	0.55089	0.43048	0.31975	0.21571	0.11595	0.01841	-0.07878	-0.17747	-0.27963	-0.38751	-0.50384
129	0.56164	0.44114	0.33048	0.22667	0.12731	0.03036	-0.06603	-0.16365	-0.2644	-0.37044	-0.48434
130	0.57233	0.45172	0.34111	0.23752	0.13853	0.04213	-0.05349	-0.15009	-0.24951	-0.35381	-0.46542
131	0.58295	0.46221	0.35165	0.24825	0.14961	0.05374	-0.04115	-0.13679	-0.23495	-0.3376	-0.44706
132	0.5935	0.47264	0.3621	0.25887	0.16056	0.06519	-0.02902	-0.12374	-0.22071	-0.32179	-0.42922
133	0.604	0.48299	0.37246	0.26939	0.17139	0.07648	-0.01707	-0.11093	-0.20675	-0.30636	-0.41186

134	0.61443	0.49327	0.38274	0.2798	0.18209	0.08763	-0.00531	-0.09834	-0.19308	-0.29129	-0.39497
135	0.62481	0.50348	0.39293	0.29012	0.19267	0.09863	0.00628	-0.08596	-0.17968	-0.27656	-0.37851
136	0.63513	0.51363	0.40305	0.30035	0.20314	0.10949	0.0177	-0.07379	-0.16653	-0.26215	-0.36246
137	0.6454	0.52371	0.41309	0.31048	0.21351	0.12022	0.02896	-0.06182	-0.15363	-0.24804	-0.34681
138	0.65561	0.53373	0.42306	0.32052	0.22376	0.13083	0.04007	-0.05004	-0.14097	-0.23423	-0.33154
139	0.66578	0.54369	0.43295	0.33048	0.23391	0.14131	0.05102	-0.03844	-0.12852	-0.22071	-0.31661
140	0.67589	0.55359	0.44277	0.34036	0.24397	0.15167	0.06184	-0.02702	-0.11629	-0.20744	-0.30202
141	0.68595	0.56343	0.45253	0.35015	0.25393	0.16192	0.07251	-0.01576	-0.10427	-0.19444	-0.28775
142	0.69597	0.57321	0.46221	0.35987	0.26379	0.17206	0.08305	-0.00466	-0.09245	-0.18167	-0.27379
143	0.70594	0.58295	0.47184	0.36951	0.27357	0.18209	0.09347	0.00628	-0.08081	-0.16914	-0.26011
144	0.71587	0.59262	0.4814	0.37908	0.28325	0.19201	0.10376	0.01707	-0.06936	-0.15684	-0.24672
145	0.72575	0.60225	0.49091	0.38858	0.29286	0.20184	0.11393	0.02772	-0.05808	-0.14474	-0.23358
146	0.73559	0.61183	0.50035	0.398	0.30238	0.21157	0.12398	0.03823	-0.04697	-0.13285	-0.22071
147	0.74539	0.62136	0.50974	0.40736	0.31182	0.22121	0.13392	0.0486	-0.03602	-0.12116	-0.20807
148	0.75515	0.63084	0.51907	0.41666	0.32119	0.23075	0.14376	0.05885	-0.02523	-0.10966	-0.19566
149	0.76486	0.64027	0.52834	0.42589	0.33048	0.24021	0.15349	0.06897	-0.01458	-0.09834	-0.18348
150	0.77454	0.64966	0.53757	0.43506	0.3397	0.24958	0.16312	0.07897	-0.00408	-0.08719	-0.17151
151	0.78419	0.65901	0.54674	0.44417	0.34885	0.25887	0.17265	0.08885	0.00628	-0.07621	-0.15974
152	0.79379	0.66831	0.55586	0.45322	0.35793	0.26808	0.18209	0.09863	0.01651	-0.06539	-0.14818
153	0.80336	0.67757	0.56494	0.46221	0.36695	0.27721	0.19143	0.10829	0.0266	-0.05473	-0.13679
154	0.8129	0.68679	0.57396	0.47115	0.3759	0.28626	0.20069	0.11785	0.03658	-0.04422	-0.12559
155	0.82239	0.69597	0.58295	0.48004	0.38478	0.29525	0.20986	0.12731	0.04643	-0.03385	-0.11457

156	0.83186	0.70511	0.59188	0.48887	0.39361	0.30416	0.21895	0.13667	0.05616	-0.02362	-0.10371
157	0.8413	0.71422	0.60077	0.49766	0.40238	0.313	0.22795	0.14593	0.06579	-0.01353	-0.09301
158	0.8507	0.72328	0.60962	0.50639	0.41109	0.32177	0.23688	0.1551	0.0753	-0.00356	-0.08246
159	0.86007	0.73232	0.61843	0.51507	0.41974	0.33048	0.24573	0.16418	0.08471	0.00628	-0.07207
160	0.86941	0.74131	0.62719	0.52371	0.42834	0.33913	0.25451	0.17318	0.09401	0.016	-0.06182
161	0.87872	0.75027	0.63592	0.5323	0.43689	0.34771	0.26321	0.18209	0.10322	0.0256	-0.05171
162	0.88801	0.7592	0.64461	0.54085	0.44538	0.35624	0.27185	0.19092	0.11233	0.03509	-0.04174
163	0.89726	0.76809	0.65326	0.54935	0.45382	0.3647	0.28041	0.19967	0.12135	0.04447	-0.03189
164	0.90649	0.77696	0.66187	0.55781	0.46221	0.37311	0.28891	0.20834	0.13027	0.05374	-0.02217
165	0.91568	0.78579	0.67045	0.56623	0.47056	0.38146	0.29735	0.21693	0.13911	0.06291	-0.01257
166	0.92486	0.79459	0.67899	0.57461	0.47886	0.38976	0.30572	0.22546	0.14787	0.07198	-0.00309
167	0.934	0.80336	0.6875	0.58295	0.48711	0.398	0.31404	0.23391	0.15654	0.08096	0.00628
168	0.94312	0.8121	0.69597	0.59124	0.49532	0.4062	0.32229	0.2423	0.16513	0.08984	0.01554
169	0.95222	0.82081	0.70441	0.5995	0.50348	0.41434	0.33048	0.25062	0.17365	0.09863	0.02469
170	0.96129	0.8295	0.71282	0.60773	0.51161	0.42244	0.33862	0.25887	0.18209	0.10733	0.03374
171	0.97034	0.83816	0.72119	0.61592	0.51969	0.43048	0.3467	0.26706	0.19045	0.11595	0.04269
172	0.97936	0.84679	0.72954	0.62407	0.52773	0.43848	0.35474	0.27519	0.19875	0.12448	0.05154
173	0.98836	0.85539	0.73786	0.63219	0.53573	0.44644	0.36271	0.28325	0.20697	0.13293	0.0603
174	0.99734	0.86397	0.74614	0.64027	0.54369	0.45435	0.37064	0.29126	0.21513	0.14131	0.06897
175	1.0063	0.87252	0.7544	0.64832	0.55161	0.46221	0.37852	0.29922	0.22322	0.14961	0.07755
176	1.01523	0.88105	0.76262	0.65634	0.5595	0.47004	0.38635	0.30711	0.23125	0.15784	0.08604
177	1.02415	0.88955	0.77082	0.66433	0.56735	0.47782	0.39413	0.31495	0.23922	0.16599	0.09445
178	1.03304	0.89803	0.779	0.67228	0.57517	0.48557	0.40187	0.32274	0.24712	0.17407	0.10278
179	1.04191	0.90649	0.78714	0.68021	0.58295	0.49327	0.40956	0.33048	0.25497	0.18209	0.11103
180	1.05076	0.91492	0.79526	0.6881	0.59069	0.50094	0.4172	0.33817	0.26276	0.19004	0.11921
181	1.0596	0.92333	0.80336	0.69597	0.59841	0.50857	0.42481	0.34581	0.27049	0.19792	0.12731
182	1.06841	0.93172	0.81143	0.70381	0.60609	0.51616	0.43237	0.3534	0.27816	0.20574	0.13534
183	1.07721	0.94009	0.81948	0.71162	0.61374	0.52371	0.43989	0.36095	0.28579	0.21351	0.14329
184	1.08598	0.94843	0.8275	0.7194	0.62136	0.53123	0.44737	0.36844	0.29336	0.22121	0.15118
185	1.09474	0.95676	0.83549	0.72716	0.62894	0.53872	0.45481	0.3759	0.30088	0.22885	0.159
186	1.10348	0.96506	0.84347	0.73489	0.6365	0.54617	0.46221	0.38331	0.30835	0.23644	0.16676
187	1.1122	0.97335	0.85142	0.74259	0.64403	0.55359	0.46958	0.39068	0.31578	0.24397	0.17445
188	1.12091	0.98162	0.85935	0.75027	0.65153	0.56097	0.47691	0.398	0.32315	0.25144	0.18209

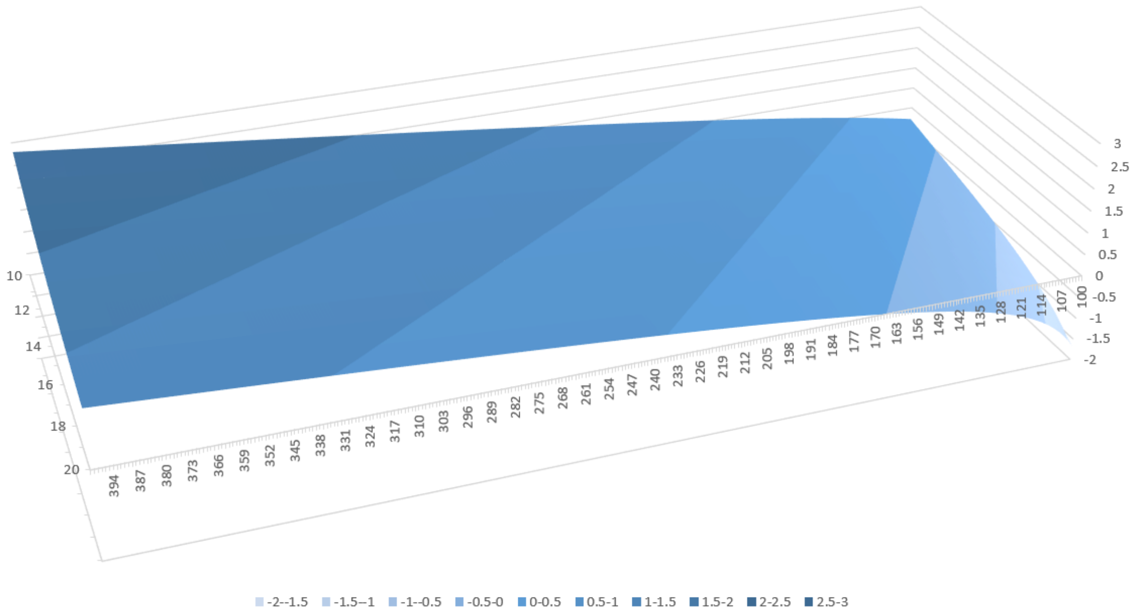
189	1.1296	0.98986	0.86726	0.75793	0.65901	0.56833	0.4842	0.40529	0.33048	0.25887	0.18966
190	1.13827	0.99809	0.87515	0.76556	0.66645	0.57565	0.49146	0.41253	0.33777	0.26624	0.19717
191	1.14692	1.0063	0.88301	0.77316	0.67387	0.58295	0.49869	0.41974	0.34501	0.27357	0.20463
192	1.15556	1.01449	0.89086	0.78075	0.68126	0.59021	0.50588	0.42691	0.3522	0.28084	0.21203
193	1.16419	1.02266	0.89868	0.78831	0.68863	0.59744	0.51304	0.43404	0.35936	0.28807	0.21938
194	1.1728	1.03082	0.90649	0.79584	0.69597	0.60465	0.52016	0.44114	0.36647	0.29525	0.22667
195	1.18139	1.03896	0.91427	0.80336	0.70329	0.61183	0.52726	0.4482	0.37355	0.30238	0.23391
196	1.18997	1.04708	0.92204	0.81085	0.71058	0.61898	0.53432	0.45522	0.38058	0.30947	0.2411
197	1.19853	1.05518	0.92979	0.81833	0.71785	0.6261	0.54135	0.46221	0.38758	0.31652	0.24825
198	1.20708	1.06327	0.93751	0.82578	0.72509	0.6332	0.54835	0.46917	0.39454	0.32352	0.25534
199	1.21561	1.07134	0.94523	0.83321	0.73232	0.64027	0.55533	0.4761	0.40146	0.33048	0.26239
200	1.22414	1.0794	0.95292	0.84062	0.73951	0.64732	0.56227	0.48299	0.40834	0.3374	0.26939
201	1.23264	1.08744	0.96059	0.84802	0.74669	0.65434	0.56919	0.48985	0.41519	0.34429	0.27634
202	1.24114	1.09547	0.96825	0.85539	0.75385	0.66134	0.57608	0.49668	0.42201	0.35113	0.28325
203	1.24962	1.10348	0.97589	0.86274	0.76098	0.66831	0.58295	0.50348	0.42879	0.35793	0.29012
204	1.25809	1.11148	0.98352	0.87008	0.76809	0.67526	0.58978	0.51026	0.43554	0.3647	0.29695
205	1.26654	1.11946	0.99113	0.8774	0.77519	0.68219	0.59659	0.517	0.44226	0.37143	0.30373
206	1.27498	1.12743	0.99872	0.88469	0.78226	0.68909	0.60338	0.52371	0.44894	0.37812	0.31048
207	1.28341	1.13538	1.0063	0.89198	0.78931	0.69597	0.61014	0.5304	0.45559	0.38478	0.31719
208	1.29183	1.14332	1.01386	0.89924	0.79635	0.70283	0.61688	0.53706	0.46221	0.39141	0.32385
209	1.30024	1.15125	1.02141	0.90649	0.80336	0.70967	0.62359	0.54369	0.46881	0.398	0.33048
210	1.30863	1.15916	1.02894	0.91372	0.81036	0.71649	0.63028	0.55029	0.47537	0.40456	0.33708
211	1.31701	1.16706	1.03645	0.92093	0.81733	0.72328	0.63695	0.55687	0.4819	0.41109	0.34363
212	1.32539	1.17494	1.04396	0.92813	0.82429	0.73006	0.64359	0.56343	0.48841	0.41758	0.35015
213	1.33375	1.18282	1.05144	0.93531	0.83123	0.73682	0.65021	0.56996	0.49489	0.42405	0.35664
214	1.34209	1.19068	1.05892	0.94247	0.83816	0.74355	0.65681	0.57646	0.50134	0.43048	0.36309
215	1.35043	1.19853	1.06638	0.94962	0.84506	0.75027	0.66339	0.58295	0.50776	0.43689	0.36951
216	1.35876	1.20637	1.07383	0.95676	0.85195	0.75697	0.66995	0.5894	0.51416	0.44326	0.3759
217	1.36708	1.21419	1.08126	0.96388	0.85882	0.76365	0.67648	0.59584	0.52053	0.44961	0.38225
218	1.37538	1.22201	1.08868	0.97098	0.86568	0.77031	0.683	0.60225	0.52688	0.45593	0.38858
219	1.38368	1.22981	1.09609	0.97808	0.87252	0.77696	0.68949	0.60864	0.53321	0.46221	0.39487
220	1.39196	1.2376	1.10348	0.98515	0.87934	0.78358	0.69597	0.61501	0.5395	0.46848	0.40113
221	1.40024	1.24538	1.11086	0.99221	0.88615	0.79019	0.70243	0.62136	0.54578	0.47472	0.40736
222	1.4085	1.25315	1.11823	0.99926	0.89294	0.79678	0.70887	0.62768	0.55203	0.48093	0.41357
223	1.41676	1.26091	1.12559	1.0063	0.89972	0.80336	0.71528	0.63399	0.55826	0.48711	0.41974
224	1.42501	1.26865	1.13293	1.01332	0.90649	0.80992	0.72169	0.64027	0.56446	0.49327	0.42589
225	1.43324	1.27639	1.14027	1.02033	0.91324	0.81646	0.72807	0.64653	0.57064	0.49941	0.43201
226	1.44147	1.28412	1.14759	1.02733	0.91997	0.82299	0.73443	0.65278	0.57681	0.50552	0.4381
227	1.44969	1.29183	1.1549	1.03431	0.92669	0.8295	0.74078	0.65901	0.58295	0.51161	0.44417
228	1.4579	1.29954	1.1622	1.04128	0.93339	0.83599	0.74711	0.66521	0.58906	0.51767	0.45021
229	1.4661	1.30723	1.16949	1.04824	0.94009	0.84247	0.75343	0.6714	0.59516	0.52371	0.45622
230	1.4743	1.31492	1.17676	1.05518	0.94677	0.84894	0.75972	0.67757	0.60124	0.52973	0.46221
231	1.48248	1.3226	1.18403	1.06212	0.95343	0.85539	0.76601	0.68372	0.6073	0.53573	0.46818
232	1.49065	1.33026	1.19129	1.06904	0.96008	0.86182	0.77227	0.68985	0.61333	0.5417	0.47412

233	1.49882	1.33792	1.19853	1.07595	0.96672	0.86825	0.77852	0.69597	0.61935	0.54766	0.48004
234	1.50698	1.34557	1.20577	1.08285	0.97335	0.87465	0.78475	0.70207	0.62535	0.55359	0.48594
235	1.51513	1.35321	1.21299	1.08974	0.97996	0.88105	0.79097	0.70815	0.63133	0.5595	0.49181
236	1.52327	1.36084	1.22021	1.09661	0.98657	0.88743	0.79717	0.71422	0.6373	0.56539	0.49766
237	1.53141	1.36846	1.22741	1.10348	0.99316	0.89379	0.80336	0.72027	0.64324	0.57126	0.50348
238	1.53954	1.37607	1.23461	1.11033	0.99973	0.90015	0.80953	0.7263	0.64917	0.57711	0.50929
239	1.54765	1.38368	1.24179	1.11718	1.0063	0.90649	0.81569	0.73232	0.65508	0.58295	0.51507
240	1.55577	1.39127	1.24897	1.12401	1.01285	0.91281	0.82184	0.73832	0.66097	0.58876	0.52084
241	1.56387	1.39886	1.25613	1.13084	1.0194	0.91913	0.82797	0.7443	0.66684	0.59455	0.52658
242	1.57197	1.40644	1.26329	1.13765	1.02593	0.92543	0.83408	0.75027	0.6727	0.60033	0.5323
243	1.58006	1.41401	1.27044	1.14445	1.03245	0.93172	0.84019	0.75623	0.67854	0.60609	0.53801
244	1.58814	1.42157	1.27758	1.15125	1.03896	0.938	0.84628	0.76217	0.68437	0.61183	0.54369
245	1.59621	1.42913	1.28471	1.15803	1.04546	0.94426	0.85236	0.76809	0.69018	0.61755	0.54935
246	1.60428	1.43667	1.29183	1.1648	1.05194	0.95052	0.85842	0.77401	0.69597	0.62326	0.555
247	1.61234	1.44421	1.29895	1.17157	1.05842	0.95676	0.86447	0.7799	0.70175	0.62894	0.56062
248	1.6204	1.45174	1.30605	1.17832	1.06489	0.96299	0.87051	0.78579	0.70751	0.63462	0.56623
249	1.62845	1.45927	1.31315	1.18507	1.07134	0.96921	0.87653	0.79166	0.71326	0.64027	0.57182
250	1.63649	1.46679	1.32024	1.1918	1.07779	0.97542	0.88255	0.79752	0.71899	0.64591	0.57739
251	1.64452	1.4743	1.32732	1.19853	1.08423	0.98162	0.88855	0.80336	0.72471	0.65153	0.58295
252	1.65255	1.4818	1.33439	1.20525	1.09065	0.9878	0.89454	0.80919	0.73042	0.65714	0.58848
253	1.66057	1.48929	1.34145	1.21196	1.09707	0.99398	0.90052	0.81501	0.73611	0.66273	0.594
254	1.66859	1.49678	1.34851	1.21866	1.10348	1.00014	0.90649	0.82081	0.74178	0.66831	0.5995
255	1.6766	1.50426	1.35556	1.22535	1.10988	1.0063	0.91244	0.82661	0.74745	0.67387	0.60499
256	1.6846	1.51174	1.3626	1.23204	1.11627	1.01244	0.91839	0.83239	0.75309	0.67942	0.61046
257	1.6926	1.5192	1.36963	1.23871	1.12265	1.01858	0.92432	0.83816	0.75873	0.68495	0.61592
258	1.70059	1.52666	1.37666	1.24538	1.12902	1.0247	0.93024	0.84391	0.76435	0.69047	0.62136
259	1.70857	1.53412	1.38368	1.25204	1.13538	1.03082	0.93615	0.84966	0.76996	0.69597	0.62678
260	1.71655	1.54157	1.39069	1.25869	1.14173	1.03692	0.94205	0.85539	0.77556	0.70146	0.63219
261	1.72452	1.54901	1.39769	1.26533	1.14808	1.04302	0.94794	0.86111	0.78114	0.70694	0.63758
262	1.73249	1.55644	1.40469	1.27197	1.15441	1.04911	0.95382	0.86682	0.78672	0.7124	0.64296
263	1.74045	1.56387	1.41168	1.2786	1.16074	1.05518	0.95969	0.87252	0.79228	0.71785	0.64832
264	1.74841	1.57129	1.41866	1.28522	1.16706	1.06125	0.96555	0.87821	0.79782	0.72328	0.65367
265	1.75636	1.57871	1.42564	1.29183	1.17337	1.06731	0.9714	0.88388	0.80336	0.72871	0.65901
266	1.76431	1.58612	1.43261	1.29844	1.17967	1.07336	0.97724	0.88955	0.80888	0.73412	0.66433
267	1.77225	1.59352	1.43957	1.30504	1.18597	1.0794	0.98307	0.89521	0.8144	0.73951	0.66963
268	1.78018	1.60092	1.44653	1.31163	1.19225	1.08543	0.98889	0.90085	0.8199	0.7449	0.67493
269	1.78811	1.60831	1.45348	1.31821	1.19853	1.09146	0.9947	0.90649	0.82539	0.75027	0.68021
270	1.79604	1.6157	1.46043	1.32479	1.2048	1.09747	1.00051	0.91211	0.83087	0.75563	0.68547
271	1.80395	1.62308	1.46736	1.33136	1.21106	1.10348	1.0063	0.91773	0.83634	0.76098	0.69073
272	1.81187	1.63046	1.4743	1.33792	1.21732	1.10948	1.01208	0.92333	0.84179	0.76632	0.69597
273	1.81978	1.63783	1.48122	1.34448	1.22357	1.11547	1.01786	0.92893	0.84724	0.77164	0.7012
274	1.82768	1.64519	1.48814	1.35103	1.22981	1.12145	1.02362	0.93451	0.85267	0.77696	0.70642
275	1.83558	1.65255	1.49505	1.35757	1.23604	1.12743	1.02938	0.94009	0.8581	0.78226	0.71162
276	1.84348	1.6599	1.50196	1.36411	1.24227	1.13339	1.03513	0.94565	0.86352	0.78755	0.71681

277	1.85137	1.66725	1.50886	1.37064	1.24849	1.13935	1.04087	0.95121	0.86892	0.79283	0.72199
278	1.85925	1.67459	1.51576	1.37716	1.2547	1.1453	1.0466	0.95676	0.87432	0.7981	0.72716
279	1.86713	1.68193	1.52265	1.38368	1.26091	1.15125	1.05232	0.9623	0.8797	0.80336	0.73232
280	1.87501	1.68926	1.52953	1.39019	1.2671	1.15718	1.05804	0.96783	0.88508	0.80861	0.73746
281	1.88288	1.69659	1.53641	1.39669	1.2733	1.16311	1.06375	0.97335	0.89044	0.81385	0.74259
282	1.89075	1.70391	1.54328	1.40319	1.27948	1.16903	1.06945	0.97886	0.8958	0.81907	0.74772
283	1.89861	1.71123	1.55015	1.40968	1.28566	1.17494	1.07514	0.98437	0.90115	0.82429	0.75283
284	1.90647	1.71854	1.55701	1.41617	1.29183	1.18085	1.08082	0.98986	0.90649	0.8295	0.75793
285	1.91432	1.72585	1.56387	1.42265	1.298	1.18675	1.0865	0.99535	0.91182	0.8347	0.76302
286	1.92217	1.73316	1.57072	1.42913	1.30416	1.19264	1.09217	1.00083	0.91714	0.83988	0.76809
287	1.93001	1.74045	1.57757	1.4356	1.31031	1.19853	1.09783	1.0063	0.92245	0.84506	0.77316
288	1.93785	1.74775	1.58441	1.44206	1.31646	1.20441	1.10348	1.01176	0.92775	0.85023	0.77822
289	1.94569	1.75504	1.59125	1.44852	1.3226	1.21028	1.10913	1.01722	0.93304	0.85539	0.78327
290	1.95352	1.76232	1.59808	1.45497	1.32873	1.21615	1.11476	1.02266	0.93833	0.86054	0.78831
291	1.96135	1.7696	1.6049	1.46142	1.33486	1.22201	1.1204	1.0281	0.9436	0.86568	0.79333
292	1.96918	1.77688	1.61172	1.46786	1.34098	1.22786	1.12602	1.03353	0.94887	0.87081	0.79835
293	1.977	1.78415	1.61854	1.4743	1.3471	1.23371	1.13164	1.03896	0.95413	0.87593	0.80336
294	1.98481	1.79141	1.62535	1.48073	1.35321	1.23955	1.13725	1.04437	0.95938	0.88105	0.80836
295	1.99263	1.79868	1.63216	1.48715	1.35931	1.24538	1.14285	1.04978	0.96463	0.88615	0.81335
296	2.00043	1.80593	1.63896	1.49357	1.36541	1.25121	1.14845	1.05518	0.96986	0.89125	0.81833
297	2.00824	1.81319	1.64576	1.49999	1.37151	1.25703	1.15404	1.06058	0.97509	0.89634	0.8233
298	2.01604	1.82044	1.65255	1.5064	1.37759	1.26284	1.15962	1.06596	0.98031	0.90141	0.82826
299	2.02384	1.82768	1.65934	1.5128	1.38368	1.26865	1.1652	1.07134	0.98552	0.90649	0.83321
300	2.03163	1.83492	1.66612	1.5192	1.38975	1.27446	1.17077	1.07672	0.99073	0.91155	0.83816
301	2.03942	1.84216	1.6729	1.5256	1.39583	1.28025	1.17634	1.08208	0.99593	0.9166	0.84309
302	2.04721	1.84939	1.67967	1.53199	1.40189	1.28605	1.18189	1.08744	1.00112	0.92165	0.84802
303	2.05499	1.85662	1.68644	1.53837	1.40795	1.29183	1.18745	1.09279	1.0063	0.92669	0.85293
304	2.06277	1.86385	1.69321	1.54476	1.41401	1.29761	1.19299	1.09814	1.01147	0.93172	0.85784
305	2.07055	1.87107	1.69997	1.55113	1.42006	1.30339	1.19853	1.10348	1.01664	0.93674	0.86274
306	2.07832	1.87829	1.70673	1.5575	1.42611	1.30916	1.20406	1.10881	1.0218	0.94176	0.86764
307	2.08609	1.8855	1.71348	1.56387	1.43215	1.31492	1.20959	1.11414	1.02696	0.94677	0.87252
308	2.09385	1.89271	1.72023	1.57023	1.43818	1.32068	1.21511	1.11946	1.0321	0.95177	0.8774
309	2.10161	1.89992	1.72698	1.57659	1.44421	1.32643	1.22063	1.12477	1.03725	0.95676	0.88226
310	2.10937	1.90712	1.73372	1.58294	1.45024	1.33218	1.22614	1.13008	1.04238	0.96174	0.88712
311	2.11713	1.91432	1.74045	1.58929	1.45626	1.33792	1.23164	1.13538	1.04751	0.96672	0.89198
312	2.12488	1.92152	1.74719	1.59564	1.46228	1.34366	1.23714	1.14067	1.05263	0.97169	0.89682
313	2.13263	1.92871	1.75392	1.60198	1.46829	1.34939	1.24264	1.14596	1.05774	0.97666	0.90166
314	2.14038	1.9359	1.76064	1.60831	1.4743	1.35512	1.24812	1.15125	1.06285	0.98162	0.90649
315	2.14812	1.94308	1.76736	1.61465	1.4803	1.36084	1.25361	1.15652	1.06795	0.98657	0.91131
316	2.15586	1.95026	1.77408	1.62097	1.4863	1.36656	1.25908	1.16179	1.07304	0.99151	0.91612
317	2.16359	1.95744	1.78079	1.6273	1.49229	1.37227	1.26455	1.16706	1.07813	0.99645	0.92093
318	2.17133	1.96461	1.7875	1.63362	1.49828	1.37797	1.27002	1.17232	1.08321	1.00138	0.92573
319	2.17906	1.97178	1.79421	1.63993	1.50426	1.38368	1.27548	1.17757	1.08829	1.0063	0.93052
320	2.18678	1.97895	1.80091	1.64624	1.51024	1.38937	1.28094	1.18282	1.09336	1.01122	0.93531

321	2.19451	1.98612	1.80761	1.65255	1.51622	1.39507	1.28639	1.18806	1.09842	1.01613	0.94009
322	2.20223	1.99328	1.8143	1.65885	1.52219	1.40076	1.29183	1.1933	1.10348	1.02103	0.94486
323	2.20995	2.00043	1.82099	1.66515	1.52816	1.40644	1.29727	1.19853	1.10853	1.02593	0.94962
324	2.21766	2.00759	1.82768	1.67145	1.53412	1.41212	1.30271	1.20376	1.11358	1.03082	0.95438
325	2.22537	2.01474	1.83437	1.67774	1.54008	1.41779	1.30814	1.20898	1.11862	1.0357	0.95913
326	2.23308	2.02189	1.84105	1.68403	1.54603	1.42346	1.31356	1.21419	1.12365	1.04058	0.96388
327	2.24079	2.02903	1.84773	1.69031	1.55198	1.42913	1.31899	1.2194	1.12868	1.04546	0.96862
328	2.2485	2.03618	1.8544	1.69659	1.55793	1.43479	1.3244	1.22461	1.13371	1.05032	0.97335
329	2.2562	2.04332	1.86107	1.70287	1.56387	1.44044	1.32981	1.22981	1.13872	1.05518	0.97808
330	2.26389	2.05045	1.86774	1.70914	1.56981	1.4461	1.33522	1.235	1.14374	1.06004	0.98279
331	2.27159	2.05758	1.8744	1.71541	1.57574	1.45174	1.34062	1.24019	1.14874	1.06489	0.98751
332	2.27928	2.06471	1.88106	1.72168	1.58167	1.45739	1.34602	1.24538	1.15375	1.06973	0.99221
333	2.28697	2.07184	1.88772	1.72794	1.5876	1.46303	1.35141	1.25056	1.15874	1.07457	0.99692
334	2.29466	2.07897	1.89438	1.7342	1.59352	1.46866	1.3568	1.25574	1.16373	1.0794	1.00161
335	2.30235	2.08609	1.90103	1.74045	1.59944	1.4743	1.36218	1.26091	1.16872	1.08423	1.0063
336	2.31003	2.09321	1.90767	1.74671	1.60536	1.47992	1.36756	1.26607	1.1737	1.08905	1.01098
337	2.31771	2.10032	1.91432	1.75295	1.61127	1.48555	1.37294	1.27123	1.17868	1.09386	1.01566
338	2.32539	2.10743	1.92096	1.7592	1.61718	1.49117	1.37831	1.27639	1.18365	1.09867	1.02033
339	2.33306	2.11454	1.9276	1.76544	1.62308	1.49678	1.38368	1.28154	1.18861	1.10348	1.025
340	2.34073	2.12165	1.93424	1.77168	1.62898	1.50239	1.38904	1.28669	1.19357	1.10828	1.02965
341	2.3484	2.12876	1.94087	1.77791	1.63488	1.508	1.3944	1.29183	1.19853	1.11307	1.03431
342	2.35607	2.13586	1.9475	1.78415	1.64077	1.5136	1.39975	1.29697	1.20348	1.11786	1.03896
343	2.36374	2.14296	1.95413	1.79038	1.64666	1.5192	1.4051	1.3021	1.20843	1.12265	1.0436
344	2.3714	2.15005	1.96075	1.7966	1.65255	1.5248	1.41045	1.30723	1.21337	1.12743	1.04824
345	2.37906	2.15715	1.96737	1.80282	1.65843	1.53039	1.41579	1.31236	1.21831	1.1322	1.05287
346	2.38672	2.16424	1.97399	1.80904	1.66431	1.53598	1.42113	1.31748	1.22324	1.13697	1.0575
347	2.39437	2.17133	1.98061	1.81526	1.67019	1.54157	1.42646	1.3226	1.22817	1.14173	1.06212
348	2.40203	2.17841	1.98722	1.82147	1.67606	1.54715	1.43179	1.32771	1.23309	1.14649	1.06673
349	2.40968	2.1855	1.99383	1.82768	1.68193	1.55272	1.43712	1.33282	1.23801	1.15125	1.07134
350	2.41733	2.19258	2.00043	1.83389	1.6878	1.5583	1.44244	1.33792	1.24292	1.15599	1.07595
351	2.42497	2.19966	2.00704	1.84009	1.69366	1.56387	1.44776	1.34302	1.24783	1.16074	1.08055
352	2.43262	2.20673	2.01364	1.8463	1.69952	1.56944	1.45307	1.34812	1.25274	1.16548	1.08515
353	2.44026	2.21381	2.02024	1.85249	1.70538	1.575	1.45838	1.35321	1.25764	1.17021	1.08974
354	2.4479	2.22088	2.02684	1.85869	1.71123	1.58056	1.46369	1.3583	1.26254	1.17494	1.09432
355	2.45554	2.22795	2.03343	1.86488	1.71708	1.58612	1.469	1.36338	1.26743	1.17967	1.0989
356	2.46317	2.23501	2.04002	1.87107	1.72293	1.59167	1.4743	1.36846	1.27232	1.18439	1.10348
357	2.47081	2.24208	2.04661	1.87726	1.72877	1.59722	1.47959	1.37354	1.2772	1.18911	1.10805
358	2.47844	2.24914	2.0532	1.88344	1.73462	1.60277	1.48488	1.37861	1.28208	1.19382	1.11262
359	2.48607	2.2562	2.05978	1.88962	1.74045	1.60831	1.49017	1.38368	1.28696	1.19853	1.11718
360	2.49369	2.26325	2.06636	1.8958	1.74629	1.61385	1.49546	1.38874	1.29183	1.20323	1.12174
361	2.50132	2.27031	2.07294	1.90198	1.75212	1.61939	1.50074	1.3938	1.2967	1.20793	1.12629
362	2.50894	2.27736	2.07951	1.90815	1.75795	1.62493	1.50602	1.39886	1.30156	1.21263	1.13084
363	2.51656	2.28441	2.08609	1.91432	1.76378	1.63046	1.5113	1.40391	1.30642	1.21732	1.13538
364	2.52418	2.29146	2.09266	1.92049	1.7696	1.63598	1.51657	1.40896	1.31128	1.22201	1.13992

365	2.5318	2.2985	2.09923	1.92665	1.77542	1.64151	1.52184	1.41401	1.31613	1.22669	1.14445
366	2.53941	2.30555	2.10579	1.93281	1.78124	1.64703	1.5271	1.41905	1.32098	1.23137	1.14898
367	2.54703	2.31259	2.11236	1.93897	1.78705	1.65255	1.53236	1.42409	1.32583	1.23604	1.15351
368	2.55464	2.31963	2.11892	1.94513	1.79287	1.65806	1.53762	1.42913	1.33067	1.24071	1.15803
369	2.56225	2.32667	2.12548	1.95129	1.79868	1.66358	1.54288	1.43416	1.3355	1.24538	1.16255
370	2.56985	2.3337	2.13203	1.95744	1.80448	1.66909	1.54813	1.43919	1.34034	1.25004	1.16706
371	2.57746	2.34073	2.13859	1.96359	1.81029	1.67459	1.55338	1.44421	1.34517	1.2547	1.17157
372	2.58506	2.34777	2.14514	1.96974	1.81609	1.6801	1.55863	1.44924	1.34999	1.25936	1.17607
373	2.59266	2.35479	2.15169	1.97588	1.82189	1.6856	1.56387	1.45425	1.35482	1.26401	1.18057
374	2.60026	2.36182	2.15824	1.98202	1.82768	1.6911	1.56911	1.45927	1.35963	1.26865	1.18507
375	2.60786	2.36885	2.16478	1.98816	1.83348	1.69659	1.57435	1.46428	1.36445	1.2733	1.18956
376	2.61546	2.37587	2.17133	1.9943	1.83927	1.70208	1.57958	1.46929	1.36926	1.27794	1.19405
377	2.62305	2.38289	2.17787	2.00043	1.84506	1.70757	1.58481	1.4743	1.37407	1.28257	1.19853
378	2.63064	2.38991	2.18441	2.00657	1.85084	1.71306	1.59004	1.4793	1.37888	1.2872	1.20301
379	2.63823	2.39693	2.19094	2.0127	1.85662	1.71854	1.59526	1.4843	1.38368	1.29183	1.20749
380	2.64582	2.40394	2.19748	2.01883	1.8624	1.72403	1.60049	1.48929	1.38848	1.29646	1.21196
381	2.65341	2.41095	2.20401	2.02495	1.86818	1.7295	1.60571	1.49429	1.39327	1.30108	1.21643
382	2.66099	2.41796	2.21054	2.03108	1.87396	1.73498	1.61092	1.49927	1.39806	1.3057	1.22089
383	2.66858	2.42497	2.21707	2.0372	1.87973	1.74045	1.61613	1.50426	1.40285	1.31031	1.22535
384	2.67616	2.43198	2.2236	2.04332	1.8855	1.74592	1.62134	1.50925	1.40763	1.31492	1.22981
385	2.68374	2.43899	2.23012	2.04943	1.89127	1.75139	1.62655	1.51423	1.41242	1.31953	1.23426
386	2.69132	2.44599	2.23664	2.05555	1.89704	1.75686	1.63176	1.5192	1.41719	1.32413	1.23871
387	2.6989	2.45299	2.24316	2.06166	1.9028	1.76232	1.63696	1.52418	1.42197	1.32873	1.24316
388	2.70647	2.45999	2.24968	2.06777	1.90856	1.76778	1.64216	1.52915	1.42674	1.33333	1.2476
389	2.71404	2.46699	2.2562	2.07388	1.91432	1.77324	1.64736	1.53412	1.43151	1.33792	1.25204
390	2.72162	2.47399	2.26271	2.07998	1.92008	1.77869	1.65255	1.53908	1.43628	1.34251	1.25647
391	2.72919	2.48098	2.26922	2.08609	1.92583	1.78415	1.65774	1.54405	1.44104	1.3471	1.26091
392	2.73676	2.48797	2.27573	2.09219	1.93158	1.7896	1.66293	1.54901	1.4458	1.35168	1.26533
393	2.74432	2.49496	2.28224	2.09829	1.93733	1.79505	1.66811	1.55396	1.45056	1.35626	1.26976
394	2.75189	2.50195	2.28875	2.10439	1.94308	1.80049	1.6733	1.55892	1.45531	1.36084	1.27418
395	2.75945	2.50894	2.29525	2.11048	1.94883	1.80593	1.67848	1.56387	1.46006	1.36541	1.2786
396	2.76701	2.51593	2.30176	2.11657	1.95457	1.81137	1.68366	1.56882	1.46481	1.36998	1.28301
397	2.77458	2.52291	2.30826	2.12267	1.96031	1.81681	1.68883	1.57377	1.46955	1.37455	1.28742
398	2.78213	2.52989	2.31476	2.12876	1.96605	1.82225	1.69401	1.57871	1.4743	1.37912	1.29183
399	2.78969	2.53687	2.32125	2.13484	1.97178	1.82768	1.69918	1.58365	1.47903	1.38368	1.29624
400	2.79725	2.54385	2.32775	2.14093	1.97752	1.83311	1.70434	1.58859	1.48377	1.38824	1.30064


3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
10
12
14
16
18
20
394
387
380
373
366
359
352
345
338
331
324
317
310
303
296
289
282
275
268
261
254
247
240
233
226
219
212
205
198
191
184
177
170
163
156
149
142
135
128
121
114
107
100
-2--1.5
-1.5–1
-1--0.5
-0.5-0
0-0.5
0.5-1
1-1.5
1.5-2
2-2.5
2.5-3