Maximum safe altitude for space jumps

Team 355 Problem A

Summary

After comprehensively evaluating the feasibility of practical operation and combining with Felix Baumgartner's skydiving process, this paper limits the possible maximum altitude of skydiving to the intermediate level ($80km$). After fully evaluating the influence of various factors such as oxygen reserve, opening process and friction heat generation, this paper believes that the key to limit the maximum altitude of skydiving is whether the jumper can maintain a good posture under the strong wind resistance at supersonic speeds, so as to avoid losing control of the vehicle under the action of the rolling torque, and ultimately losing consciousness due to the high-speed spinning.

In the first part, this paper firstly makes the $v-h$ images when jumping from different heights through numerical simulation and corrects them by considering the change of drag coefficient of supersonic speed. Through the images, it is found that no matter how high the height of the jump is, the speed is the same when approaching the ground, which shows that the position of the parachute opening can be fixed, which also indicates that the determination of the maximum height is independent of the parachute opening process;

In the second part, with reference to Felix Baumgartner's skydiving video and the data of the process, this paper establishes a physical model about the human falling process which generates rotation and the maximum degree of self-adjustment that can be done, and calculates the conditions under which the skydiver enters into an uncontrolled state and the conditions under which the parachutist rebalances, respectively. Finally, this paper describes the maximum capacity of a skydiver not to fall unconscious after uncontrolled spinning by defining a new physical quantity, the rotational endurance eigenvalue S, and then finds the maximum jumping altitude $h = 57km$.

In the third part, this paper analyzes the effects including the opening process, the landing process and the friction heat generation aspects, and makes a rough estimation of their order of magnitude, proving that these factors do not have an effect at the maximum skydiving altitude $h = 57km$.

Keywords: skydiving, altitude, parachute opening, rotational endurance

Contents

1 Introduction

1.1 Background

Space diving, also known as high-altitude or stratospheric skydiving, refers to the act of jumping from extremely high altitudes, typically from the edge of space, and freefalling towards the Earth before deploying a parachute to slow down and land safely. It is an extreme sport that combines elements of skydiving, high-altitude mountaineering, and aerospace technology.
Space diving usually involves ascending to the desired altitude using a high-altitude balloon or a specialized aircraft.

1.2 Problem Statement

A skydiver rises in a rocket and jumps above the atmosphere. The parachutist encounters a number of difficulties during his descent to the ground, and our task is to identify these difficulties and to find the minimum value of the parachutist's initial height of descent on the basis of no accidents.

1.3 Analysis

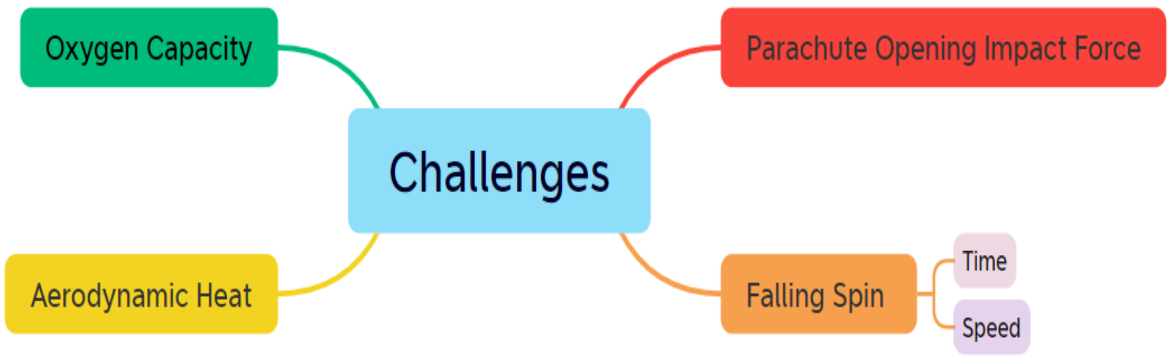

Figure 1: **Possible difficulties encountered by parachutists during their descent**

Space jumpers encounter a variety of hazards during their descent. We have identified five possible sources of danger through a comprehensive analysis of the falling motion of skydivers:
1. The heat generated by the friction between the skydiver and the air during the descent at high speed may damage the spacesuit;
2. The impact force on the parachutist at the moment of opening the parachute is too large;
3 The speed of the parachutist exceeds the speed of sound, breaking through the sound barrier in the instant of air resistance sudden change leads to the parachutist by the impact force is too large;
4 It is difficult for the parachutist to control the body during the descent, resulting in uncontrolled fainting;
5 Excessive speed during landing;
In order to analyze these five sources of danger, we first need to find out the motion characteristics of the falling process before opening the parachute, i.e., when jumping from different heights, the speed about the image of altitude. Based on these images, we can analyze the law of motion of parachute falling, so as to determine when it is most appropriate to open the parachute and the characteristics of the motion after the supersonic speed, so as to have a clearer understanding of the whole process of skydiving.

2 Assumptions

:The parachutist is well-trained and skilled in all the skills of skydiving and does not make any mistakes during the whole process;

Assumption 2: The parachutist has no initial velocity when jumping.

Assumption 3: Ignore the rotation of the earth, and only consider the earth's gravity and air resistance in the process of falling.

Assumption 4: The atmospheric environment is stable during the descent, and the temperature and airflow is only a single-valued function of the altitude.

Assumption 5: There is no collision with other flying objects during the whole process.

Assumption 6: The descent process will be approximated as a cylinder to consider the air resistance.

3 Notations

Table 1: **Notations**

Symbol	**Definition**	**Numeric Value**
F	Gravity between the Earth and the skydiver	N
G	The universal gravitational constant	$6.6710-11N\cdot m^2/kg^2$
M	Earth's mass	$5.965\times10^{24}kg$
m	Total weight of skydiver, parachute and spacesuit	$190kg$
R	Radius of the Earth	$6371000m$
h	Height of Altitude of the skydiver from the surface of the Earth	m
S	The windward area of a skydiver's glide	$0.5m^2$
f	The Air Resistance	N
v	Speed of descent of the skydiver	m/s
c	Coefficient of resistance	
P	The pressure of atmospheric	pa
T	Temperature of the atmosphere	$^{\circ}C$
M_A	Rolling moment	$N\cdot m$

4 Model

4.1 Skydiver gliding before opening parachute

Earth's gravitational pull on skydivers

$$F = \frac{GMm}{(h+R)^2} \tag{1}$$

$$f = \frac{1}{2}c\rho S v^2 \tag{2}$$

Atmospheric density is a function of temperature and pressure, which is a function of altitude. Based on the data provided by NASA, the following equation can be derived that:

$$\rho = \frac{P}{0.2869 \times (T + 273.1)} \tag{3}$$

For $h > 25000\,m$ (Upper Stratosphere):

$$T = -131.21 + 0.00299h \tag{4}$$

$$P = 2.488 \times (\frac{T + 273.1}{216.6})^{-11.388} \tag{5}$$

For $11000\,m < h < 25000\,m$ (Lower Stratosphere):

$$T = -56.46 \tag{6}$$

$$P = 22.65 \times e^{1.73 - 0.000157h} \tag{7}$$

For $h < 11000\,m$ (Troposphere):

$$T = 15.04 - 0.00649h \tag{8}$$

$$P = 101.29 \times (\frac{T + 273.1}{288.08})^{5.256} \tag{9}$$

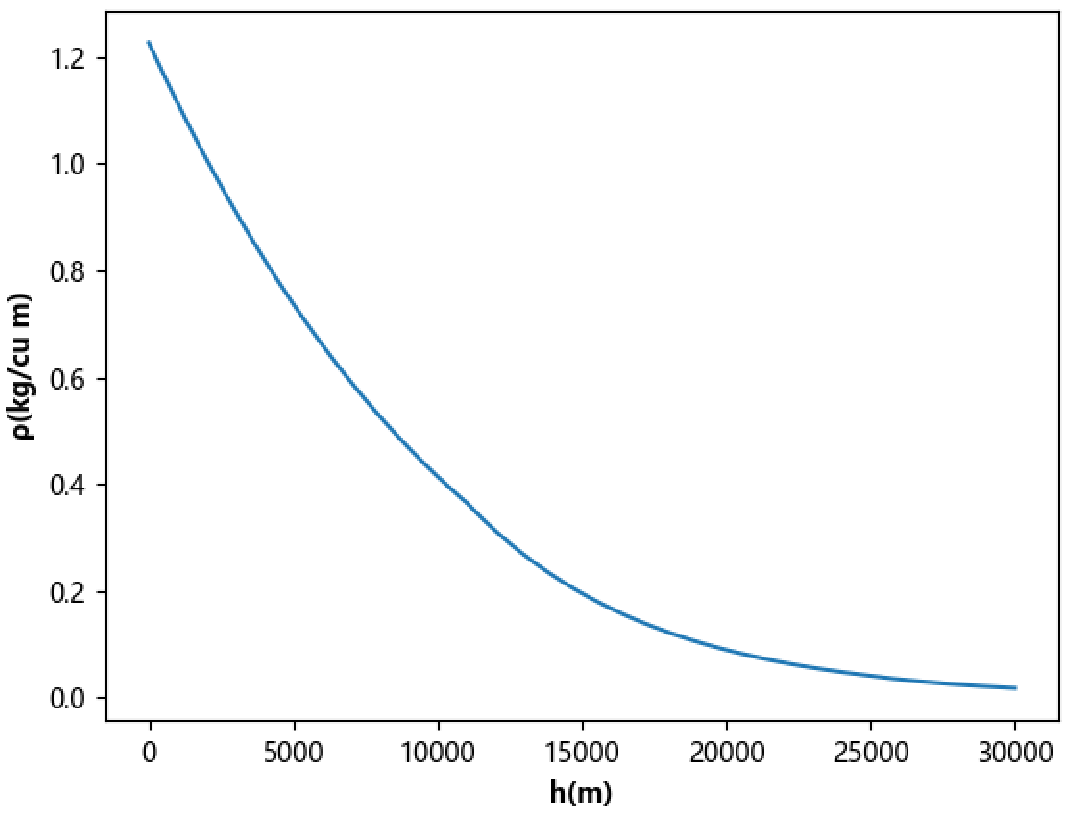

Figure 2: **Atmospheric density versus altitude**

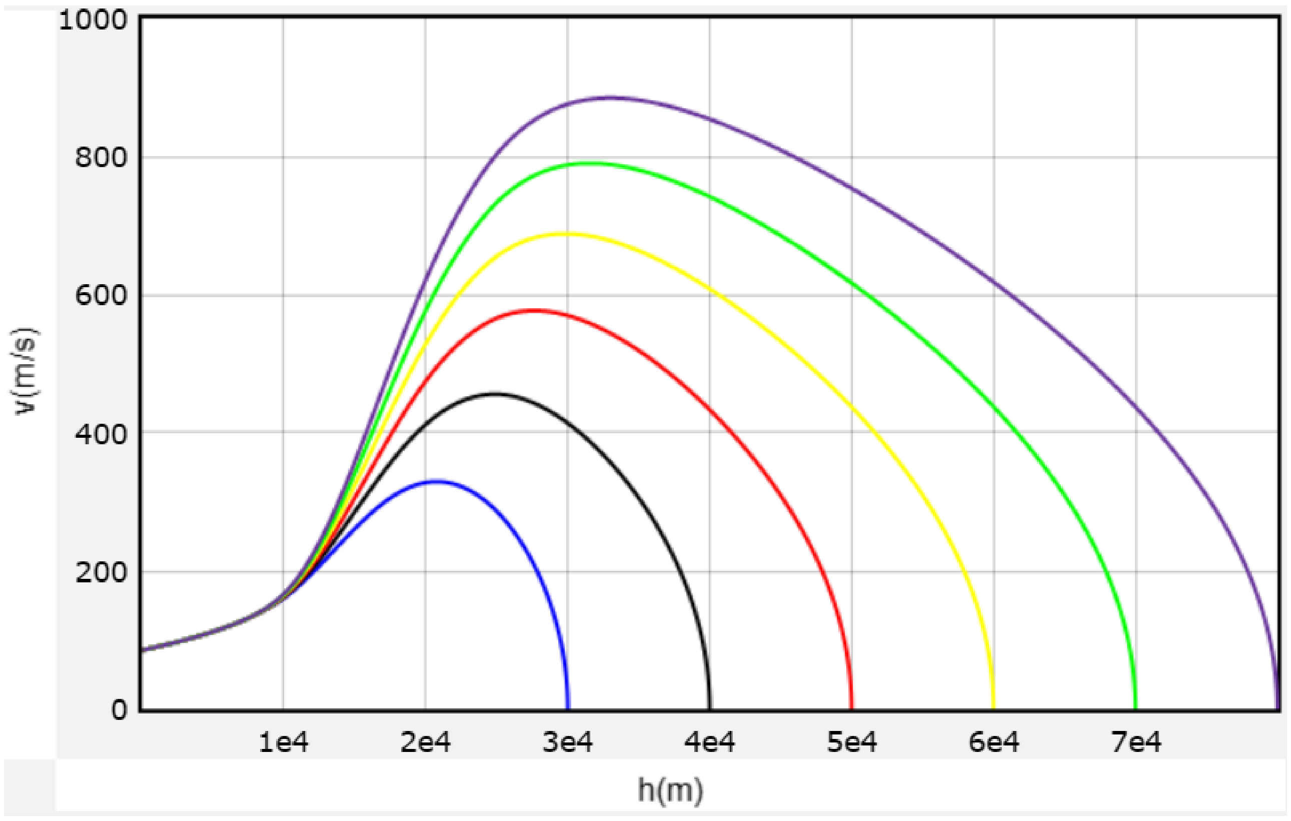

Figure 3: **Plot of skydiver's gliding speed versus altitude**

Note: The blue, black, red, yellow, green, and purple curves represent the drops from 30000m, 40000m, 50000m, 60000, 70000m, and 80000m, respectively

From Figure 3, we can see that no matter how high the skydiver falls from, the motion state tends to be consistent at the final 1km above the ground, while under normal circumstances, parachuting occurs within 1km above the ground. Therefore, the parachuting state is not the main factor limiting the maximum height of parachuting.

We can also find that when the initial falling height reaches a certain value, the falling speed of the parachutist will exceed the speed of sound, and at this time, it will generate a shock wave that will cause a sudden change in the air resistance coefficient. Therefore, we need to correct the data.

The empirical equation for the variation of the speed of sound with altitude:

$$V = 20.05 \times \sqrt{(288 - h \times \frac{0.65}{100})} \tag{10}$$

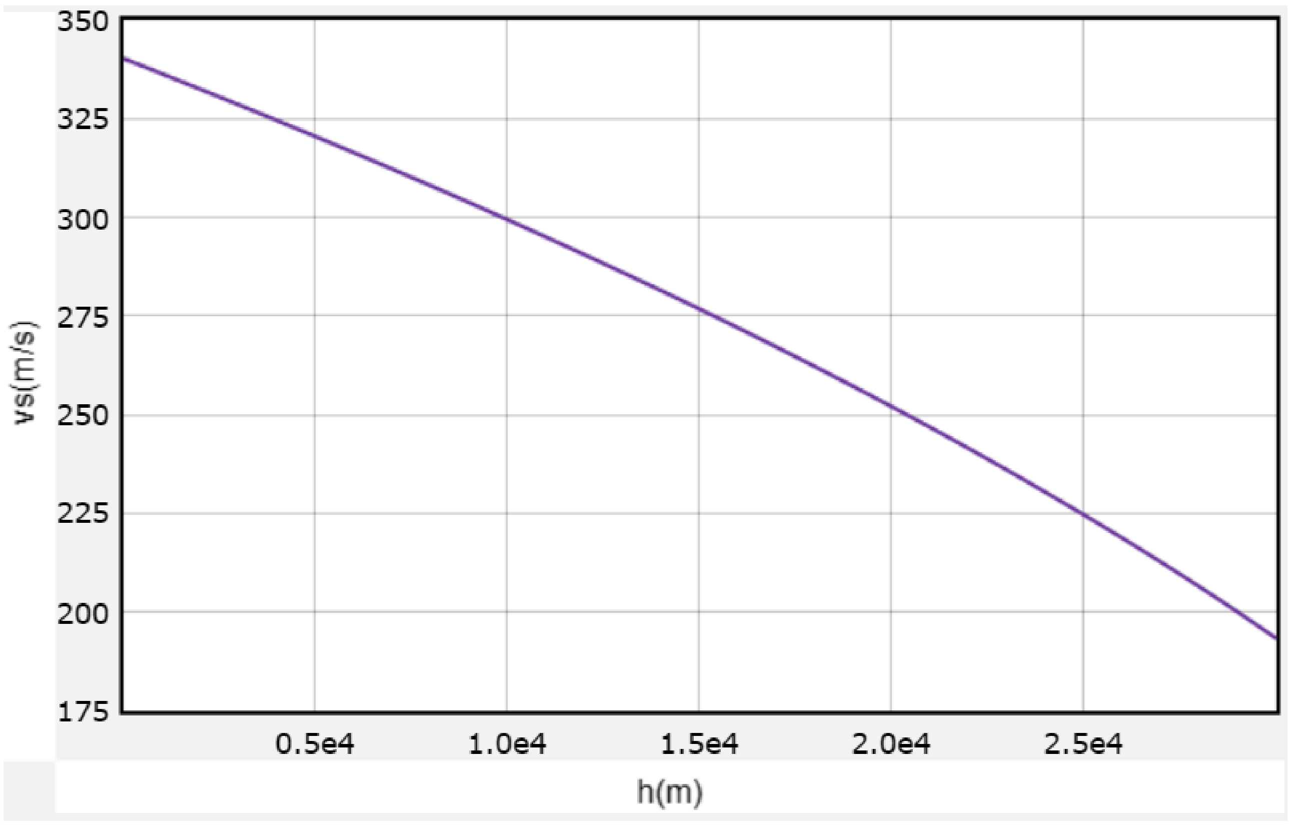

Figure 4: **Velocity of sound plotted against height**

Functional relationship between the coefficient of air resistance and Mach number:[**?**]

$$c = \begin{cases} 0.43 & \text{for } Ma < 0.3 \quad \text{(11a)} \\ 0.516\text{-}0.625\text{Ma}+1.085\text{Ma}^2 & 0.3 \leq Ma < 1.1 \quad \text{(11b)} \\ 1.247\text{-}0.101\text{Ma}+0.008\text{Ma}^2 & 1.1 \leq Ma \leq 6.0 \quad \text{(11c)} \end{cases}$$

where Ma is Mach number.
Force analysis of a falling skydiver considering only gravity and air resistance:

$$m\frac{\partial^2 h}{\partial t^2} = \frac{GMm}{(h+R)^2} - \frac{1}{2}c\rho S(\frac{\partial h}{\partial t})^2 \quad (1)$$

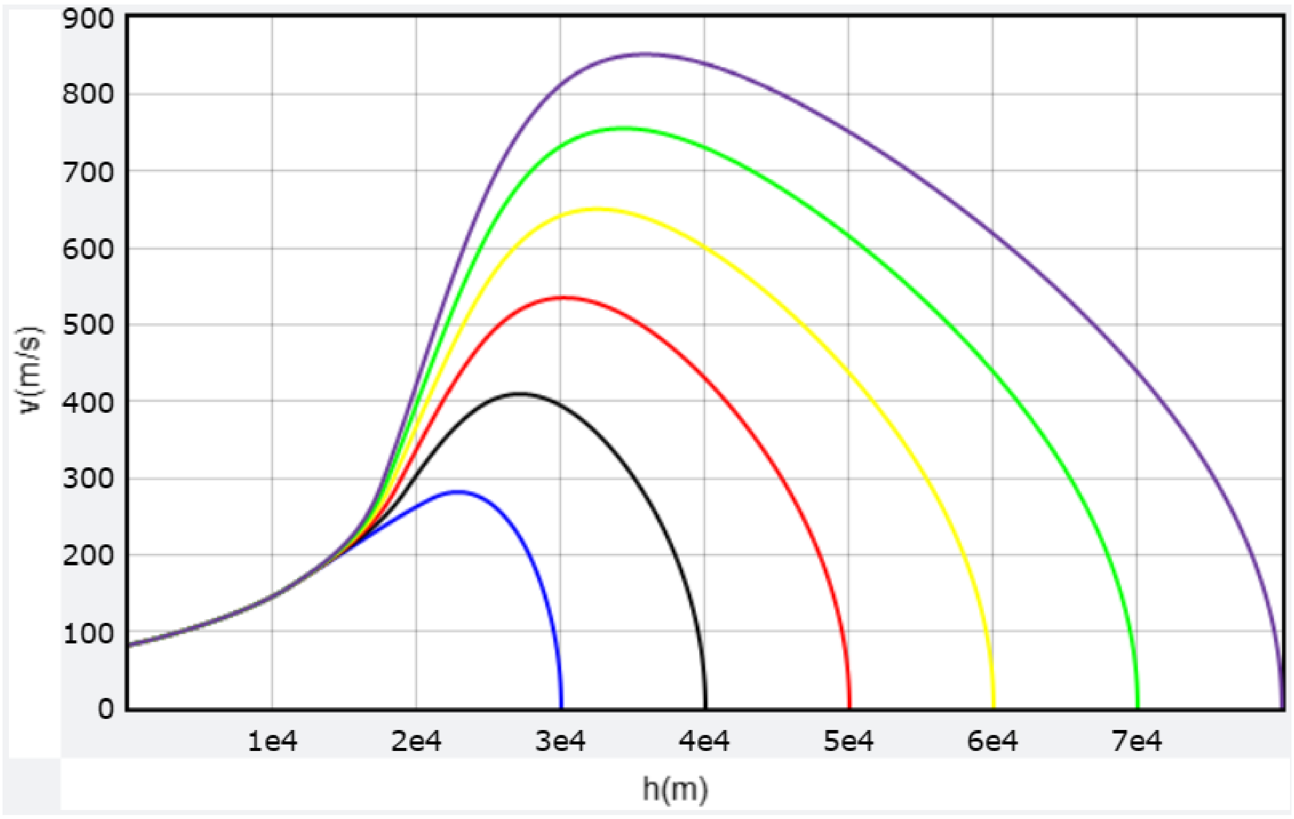

Figure 5: **Plot of corrected gliding speed versus altitude**

Note: The blue, black, red, yellow, green, and purple curves represent the drops from 30000m, 40000m, 50000m, 60000, 70000m, and 80000m, respectively

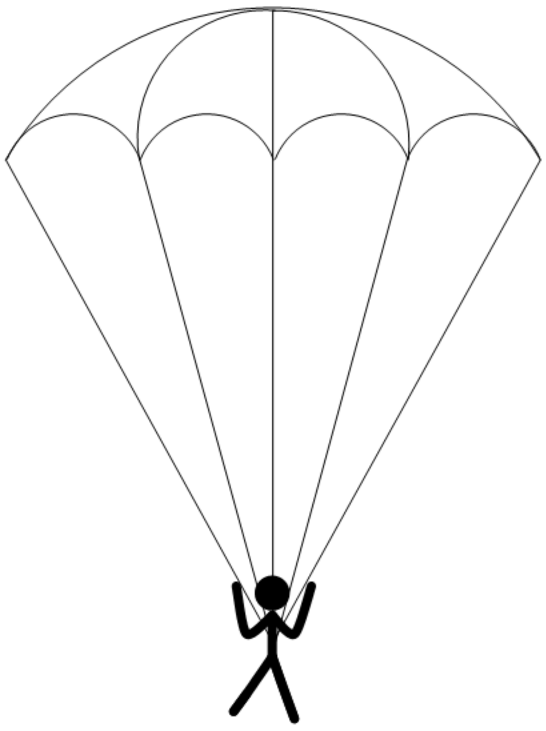

Figure 6: **Schematic diagram of a parachutist after opening his parachute**

4.2 Skydiver's spin

The unevenness and asymmetry of the suit surface causes a roll moment $M_A$when air resistance is applied, causing the skydiver to rotate.
We approximate the skydiver as a cylinder with the following schematic representation of its rotation:

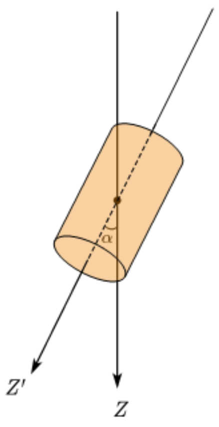

Figure 7: **Schematic of a skydiver's descent**

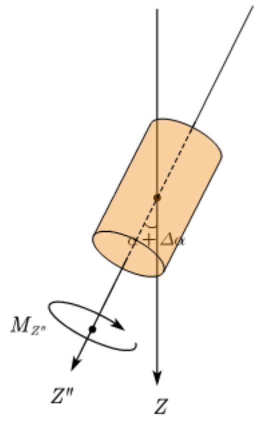

Figure 8: **Schematic of a skydiver's rotation**

Uneven and asymmetric surfaces of spacesuits lead to a roll moment M_A when air resistance is applied:

$$M_A(f, \alpha, S, l) \tag{2}$$

where f is the air resistance during descent, $\alpha$is the angle of approach, S is the windward area, and l is the characteristic length.
When falling, the angle of approach is small, so it can be approximated that M_A and α are linearly related[1]:

$$M_A = kfSl\alpha + M_{A0} \tag{3}$$

where M_{A0} is the rolling moment when the angle of attack is zero.
Consider that S and l are constants during the fall, therefore:

$$C = kSl \tag{4}$$

$$M_A = Cf\alpha + M_{A0} \tag{5}$$

In order to keep themselves from spinning, skydivers need to adjust their angle of approach to meet:

$$M_A = 0 \tag{6}$$

$$\alpha' = -\frac{M_{A0}}{Cf} \tag{7}$$

It can be seen that $\alpha\prime$ will gradually converge to 0 as f continues to increase, but at this time, due to the thin air and very little drag, it is difficult for the skydiver to adjust through movement and skill, and he can only change his angle of approach as much as possible. Therefore, when $\alpha\prime$ gradually converges to 0, it will increase the difficulty of adjustment. At this time if there is a small deflection angle $\Delta\alpha$ relative to $\alpha\prime$, all will lead to a large rolling moment, and

finally lead to the skydiver balance out of control, and begin to rotate at high speed during the descent.
We believe that a skydiver is considered to be off balance if he or she rotates more than 180 degrees in his or her minimum reaction time, so the following derivation can be made:
Let us assume that the minimum reaction time of the jumper is Δt, and the minimum angle of declination that can be maintained with respect to $\alpha\prime$ is $\Delta\alpha$, then the minimum angle of rotation $\Delta\theta$ of the jumper in Δt during the descent is:

$$J\omega = (M_A - M_{A0})\Delta t = Cf\Delta\alpha\Delta t \tag{8}$$

$$\Delta\theta = \frac{1}{2}\omega\Delta t = \frac{Cf\Delta\alpha\Delta t^2}{2J} \tag{9}$$

In order for $\Delta\theta < \pi$ to be satisfied:

$$\frac{f}{J} < \frac{2J\pi}{C\Delta\alpha\Delta t^2} \tag{10}$$

This is the condition of equilibrium during the fall.
However, the parameters in the expression we did not have the time and conditions to test experimentally, so we could only approximate the simulation by Felix Baumgartner's real-world measurements during a skydive.
We assume that the jumper's equipment is roughly the same as Felix Baumgartner's, and the mass distribution of the equipment on his body is similar, so the rotational moment of inertia can be written as $J = km$. It is also assumed that the two men are just as experienced and well-trained, and that the minimum reaction time of Δt and the minimum angle of declination $\Delta\alpha$ can be kept at the same level. Therefore the discriminant of loss of control can be written as:

$$k = \frac{f}{m} \tag{11}$$

And we note that Felix Baumgartner lost control at about $h = 33351.5208m$, $v = 700mph$ (about $312.93m/s$).
Thus by reducing Felix Baumgartner's jump, we can obtain the runaway k_1.

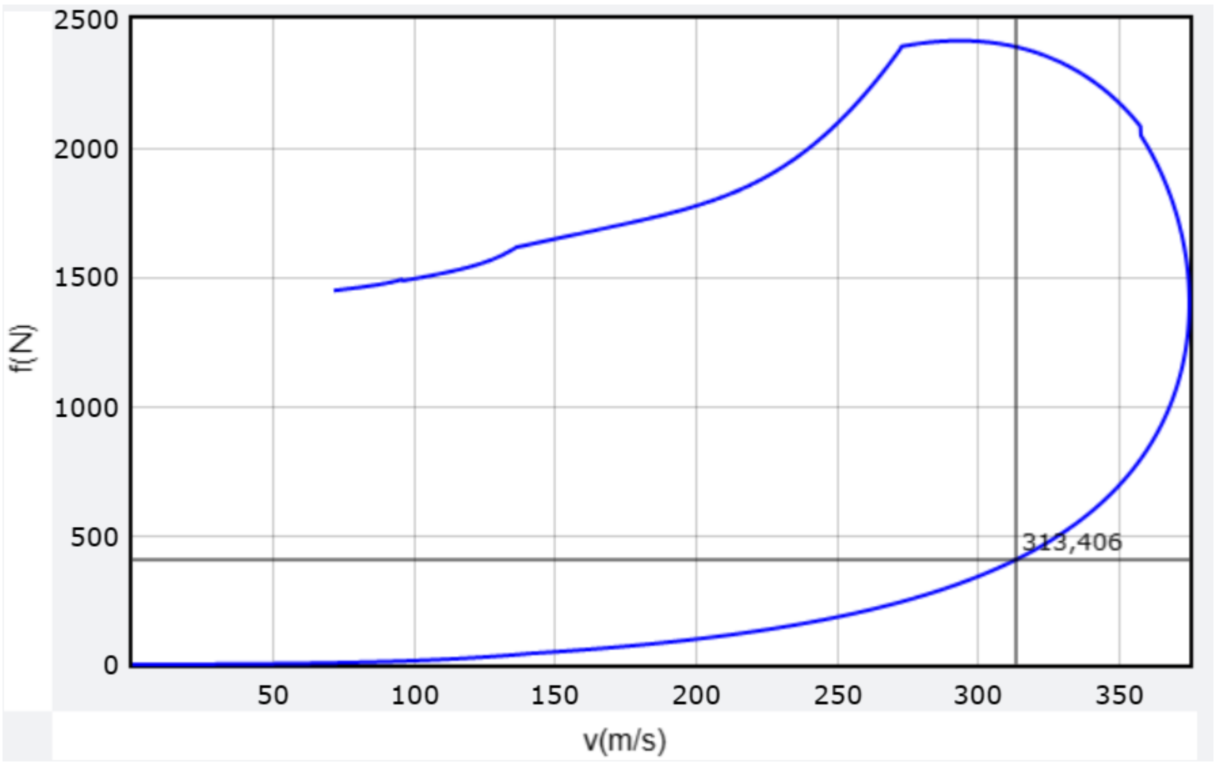

Figure 9: **Simulate the f-v diagram during the FB jump to find k1**

$$K_1 = 2.82N/kg \tag{12}$$

But the experienced Felix Baumgartner does not keep spinning, and shortly after the speed exceeds the maximum value and starts to fall, he regains his balance.
At this point, as the air gradually thickened, the resistance was high enough for the skydiver to adjust it with his own skill. We assume that this skydiver also has the same skill as Felix Baumgartner, and therefore he can also adjust his balance when f is sufficient, so we obtain a second threshold f_2 for returning to balance:
At this point its parameters are $h = 19007.0232m$,$v = 450mph(201.17m/s)$, and from this we get the threshold k_2 to remain stable.

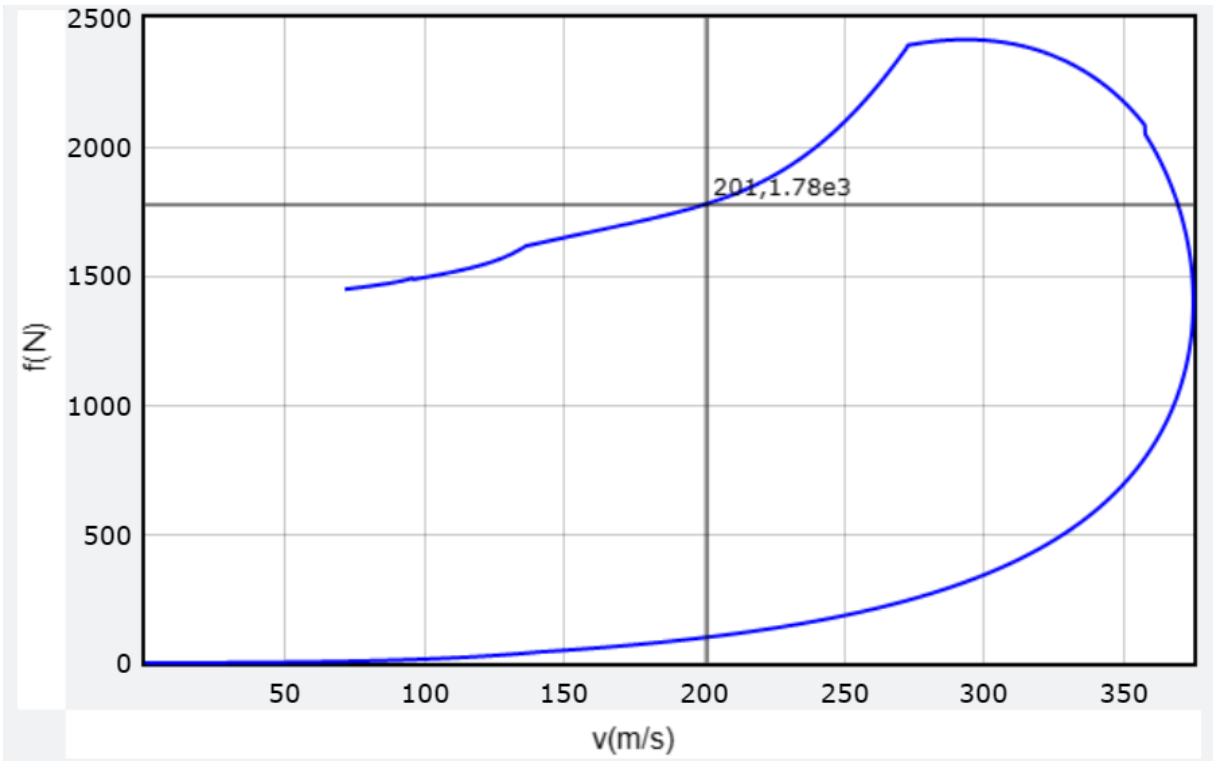

Figure 10: **Simulate the f-v diagram during the FB jump to find k2**

$$K_2 = 12.36N/kg \tag{13}$$

To verify the agreement between the $k1$ and $k2$ models and the actual ones, we bring them back to the $f-t$ simulation plots of F_B and obtain the following two plots:

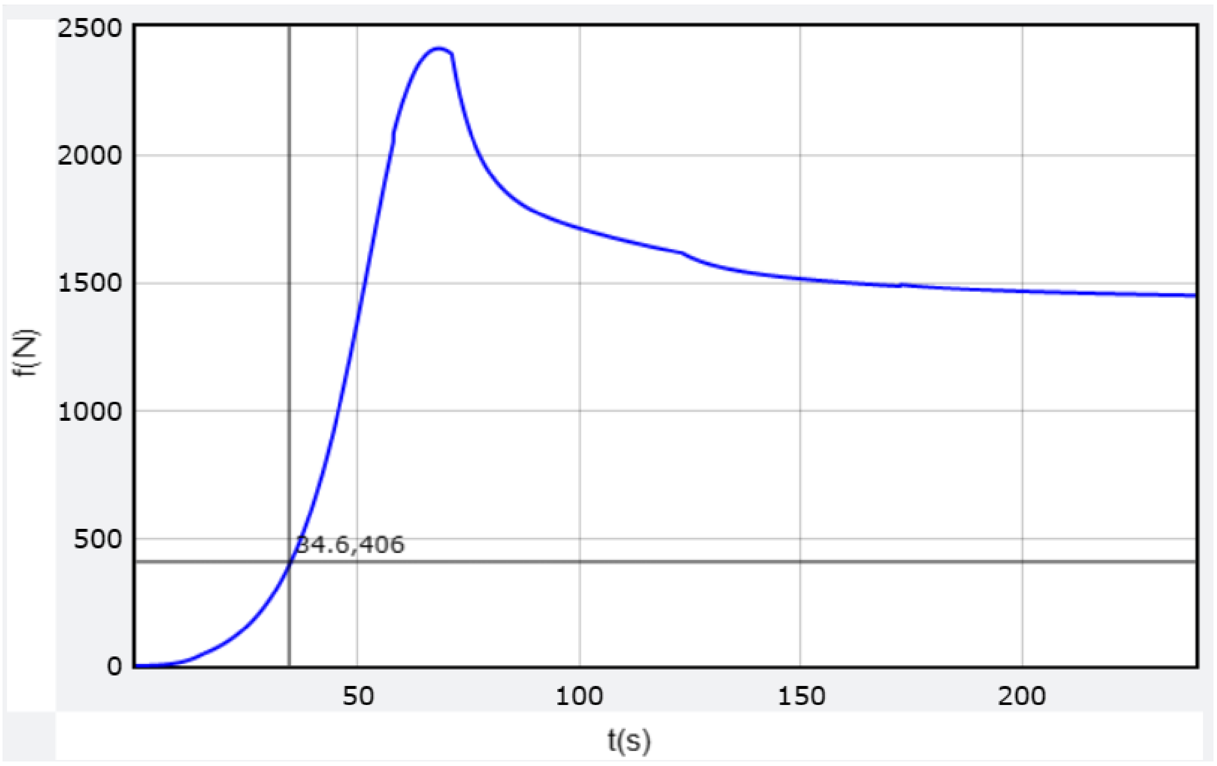

Figure 11: **Simulation of runaway time during F_B skydiving 1**

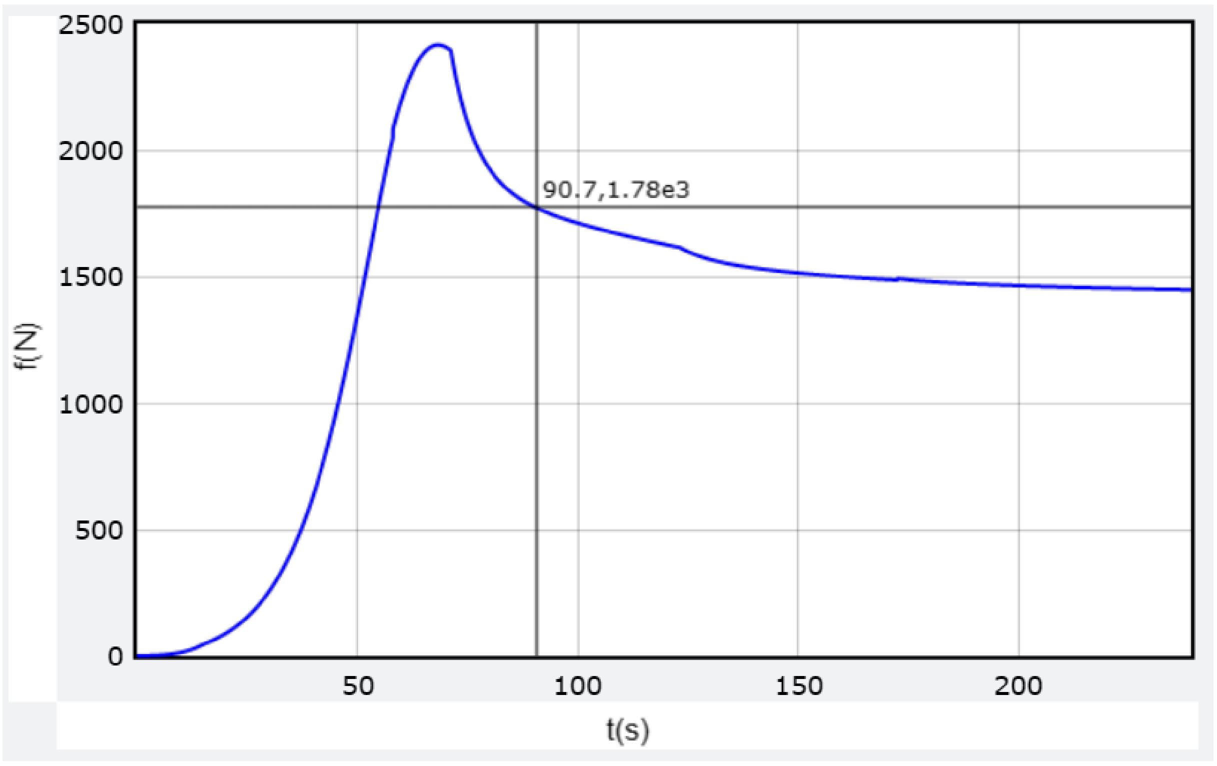

Figure 12: **Simulation of runaway time during F_B skydiving 2**

From the figure, the runaway time of F_B can be calculated to be $56.1s$, which is not much different from the runaway time of $52s$ in the actual process, and therefore the model fits the

actual situation well.
In order to study the limit of runaway rotation that the skydiver can withstand, we simulate the calculation and draw the image of air resistance f about time$t$ when jumping at different heights respectively:

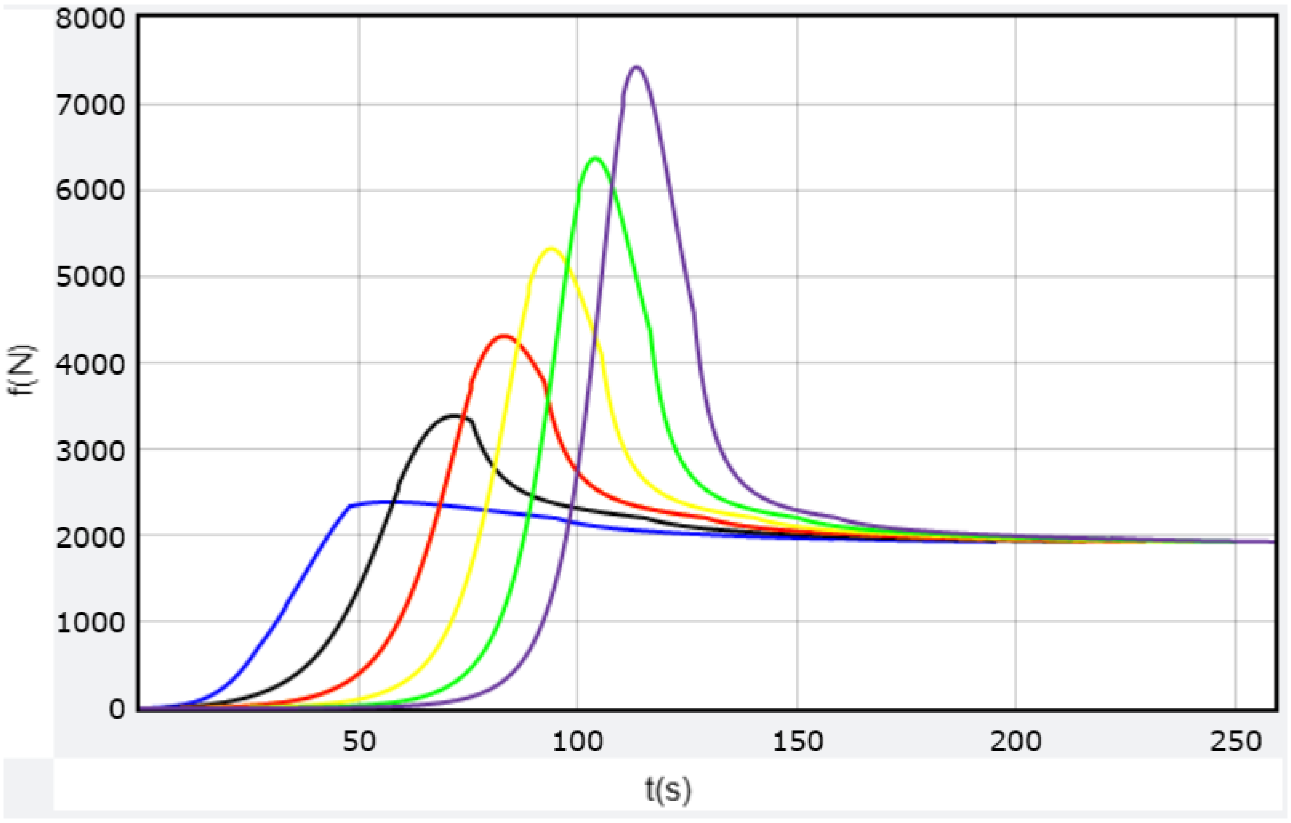

Figure 13: **Plot of air resistance against time**

Note: The blue, black, red, yellow, green, and purple curves represent the drops from 30000m, 40000m, 50000m, 60000, 70000m, and 80000m, respectively
It can be seen that as the jump height increases, the extreme peaks are sharper and the time to loss of control is actually decreasing, so the time to loss of control alone cannot be used to determine whether the tolerance limit is reached.
In order to get the limit of what the skydiver can withstand, we need to define a new physical quantity, which needs to respond to both the size of k and the time of loss of control.
Define the rotational tolerance eigenvalue S:

$$S = \int_{t_1}^{t_2} k dt \tag{14}$$

Where t1 and t2 denote the moments when the skydiver loses control and regains equilibrium, respectively.
Since the air resistance f is proportional to the roll moment M_A, this integral value actually

reflects the angular velocity increment accumulated during the runaway process, i.e., the vigor of the rotation.
Calculate S_0 for the Felix Baumgartner fall process:

$$S_0 = \int_{t_1}^{t_2} \frac{f}{m} dt = 644.68 m/s \tag{15}$$

We consider the skydiver's limit to be:

$$S_m = 1.5 S_0 \tag{16}$$

Based on this, we finally obtain the maximum height when satisfying S by continuously adjusting the jump height:

$$h_0 = 57000m \tag{17}$$

4.3 Calculation of other influencing factors

4.3.1 Landing

The speed at which the jump from a height of 1.5m hits the ground is considered to be the safe speed, from which the parameters are obtained after the parachute is opened:

$$\frac{1}{2}\rho(SC_d){v_m}^2 = mg \tag{18}$$

$$v_m < 5.5m/s \tag{19}$$

Therefore, we take the characteristic value of air resistance of the parachute $(SCd) = 100.33$, which ensures that the skydiver can touch down safely.[2]

4.3.2 The moment a skydiver opens his parachute

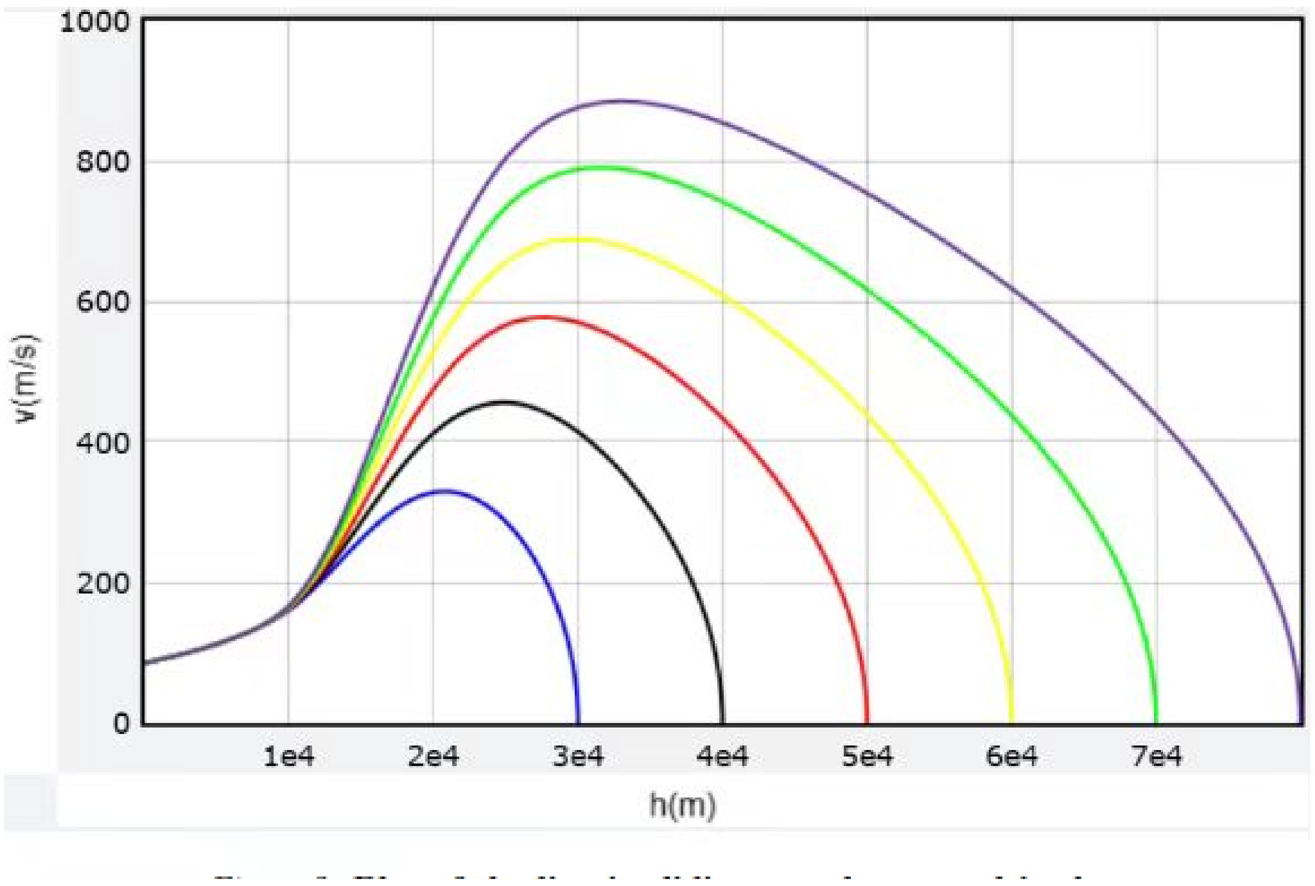

Figure 14: **Plot of velocity as a function of time for a descent from an altitude of 57,000m**

Through the simulation image, we notice that no matter how high the jump is, the trend of the $v - h$ image is always rising first and then falling, and the last section is always overlapped. This is because the atmosphere is thin at the beginning, the resistance is small, and the speed of the descent process can increase rapidly; but when the descent to a certain height, the atmosphere is dense enough, and the air resistance becomes the most important factor affecting the speed. Therefore, no matter how high to jump down, the speed is the same when approaching the ground, and the difficulty of opening a parachute will not increase because of the jumping height.[3]
Check the information that at $1500m$ to open the parachute, the characteristic time of opening the parachute is about $1s$,[4] the speed change is about $\Delta v = 20m/s$.
Then the average impulse force received at the moment of parachute opening is estimated as:

$$\overline{F}\Delta t = m\Delta v \tag{20}$$

$$\overline{F} = 3800N \tag{21}$$

A parachute withstanding above that impulse is sufficient.

4.3.3 Friction-generated heat production

Consider a downward jump from a maximum height of $h_{max} = 57000m$ with a $v - h$ diagram:

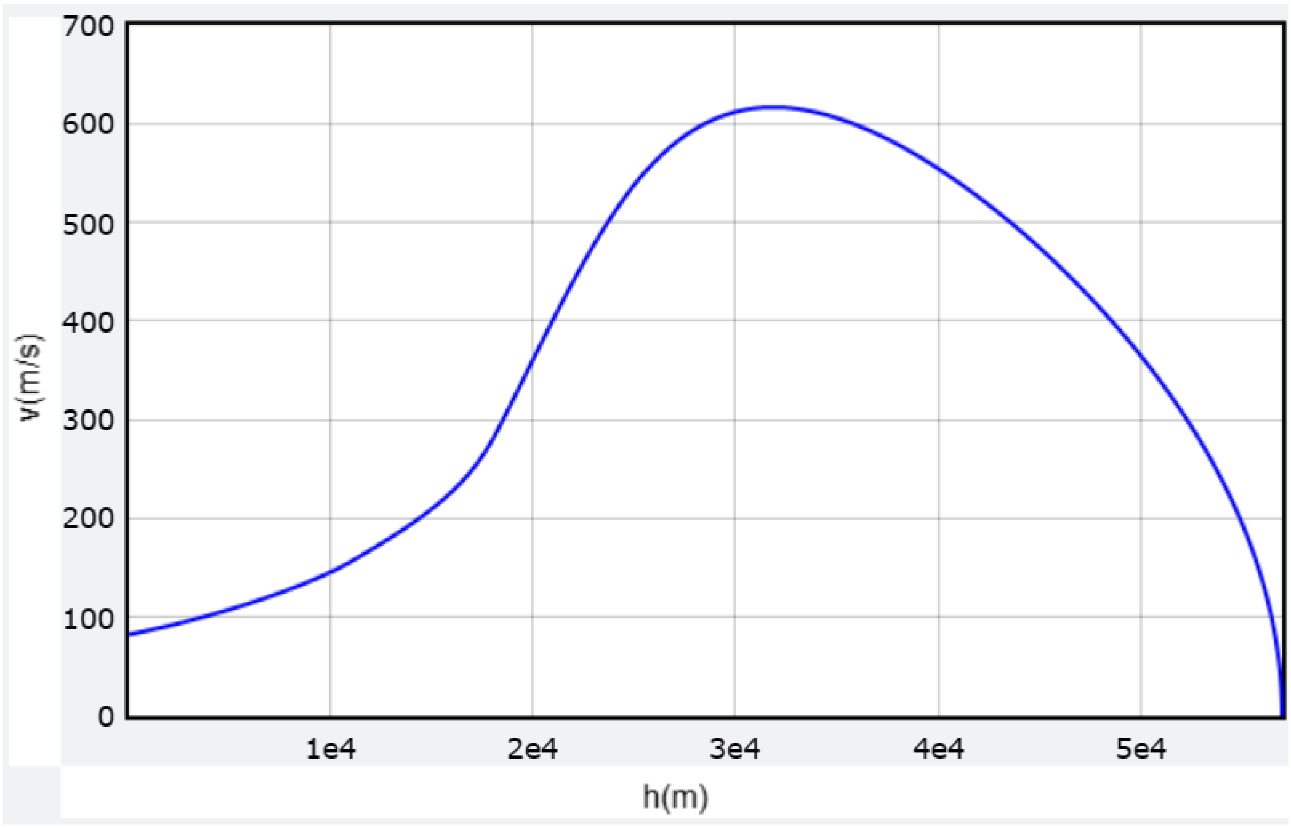

Figure 15: **Plot of velocity as a function of altitude for a descent from an altitude of 57,000m**

To the maximum speed, by the conservation of energy:

$$GMm(\frac{1}{R+h_1} - \frac{1}{R+h_2}) - \frac{1}{2}m{v_{max}}^2 = Q \tag{22}$$

$$\alpha Q = Cm_{suit}\Delta t \tag{23}$$

where α is a correction factor due to the fact that not all of the heat during the fall heats the spacesuit, and a significant portion of it heats the air.
Estimates of spacesuit warming are taken: $H_1 = 30000m$, $h_2 = 57000m$, $v_{max} = 600m/s$, $C = 1000$, $M_{suit} = 50kg$, $\alpha = 0.2$, $\Delta t = 62°C$. This shows that spacesuits don't heat up too much.

5 Discussion

5.1 Advantage

Advantage 1:In this paper, the influence of rotation during the falling process is creatively taken into account, and after comprehensive consideration, it is used as the main factor limiting the maximum jump height. At the same time, in the process of describing the physical phenomenon of rotation, the runaway indicator k and the rotation bearing eigenvalue S are creatively introduced;

Advantage 2:In this paper, after modeling and fitting the results, when the results are compared with the data of Felix Baumgartner's skydiving process, the difference between the results obtained is not too big and is more in line with the actual situation.

5.2 Disadvantage

Disadvantage 1:The value of the maximum rotation bearing eigenvalue Sm in this paper lacks the support of sufficient experimental data, so the final result $h_{max} = 57000m$ is not sufficiently convincing;

Disadvantage 2:The calculation of the other three influencing factors in this paper is only a rough estimation of the order of magnitude and not strictly complete.

References

[1] CHEN Jianping TONG Mingbo XIONG Yuchao, ZHANG Hongying. Simulation analysis of the dynamics of the stabilized deceleration phase of rocket booster recovery. *flight mechanics*, 39(63-70), 2021.

[2] Xu Qian, Guo Fengming, Su Ling, Chen Bin, and Zhang Hongjian. Launch vehicle booster parachute recovery scheme and safety analysis. *total astronautical technology*, (3):7, 2017.

[3] Wang Lirong. *Parachute Theory and Applications*. Parachute Theory and Applications, 1997.

[4] PENG Yong, SONG Xumin, and ZHANG Qingbin. Calculation of parachute inflation time. *Space return and remote sensing*, 25(1):4, 2004.

6 Appendix

```python
   from math import*
import matplotlib.pyplot as plt
import numpy
from scipy import integrate
from vpython import *
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
def T(h):
   if h>25000:
      return -131.21+0.00299*h
   elif 11000<h<25000:
      return -56.46
   else:
      return 15.04-0.00649*h
def P(h):
   if h>25000:
      return 2.488*((T(h)+273.1)/216.6)**(-11.388)
   if 11000<h<25000:
      return 22.65*e**(1.73-0.000157*h)
   else:
      return 101.29*((T(h)+273.1)/288.08)**(5.256)
def p(h):
   return P(h)/(0.2869*(T(h)+273.1))
def vs(h):
   if (288-h*6.5e-3)>0:
      return 20.05*(288-h*6.5e-3)**0.5
   else:
      return 0
def f(h,v):
      return 0.5*0.865*p(h)*0.5*v**2
def G(h):
   return (190*5.965e24/(h+6.371e6)**2)*6.67e-11
h=80000
v=0
dt=0.01
t=0
t1=100000
t2=100000
w=0
#tgraph=graph(xtitle='h(m)',ytitle='v(m/s)')
```

```
f1=gcurve(color=color.purple)
while h>0:
    a=(G(h)-f(h,v))/190
    v=v+a*dt
    h=h-v*dt
    t=t+dt
    f1.plot(h,v)
```

```
from math import*
import matplotlib.pyplot as plt
import numpy
from scipy import integrate
from vpython import *
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
def T(h):
    if h>25000:
        return -131.21+0.00299*h
    elif 11000<h<25000:
        return -56.46
    else:
        return 15.04-0.00649*h
def P(h):
    if h>25000:
        return 2.488*((T(h)+273.1)/216.6)**(-11.388)
    if 11000<h<25000:
        return 22.65*e**(1.73-0.000157*h)
    else:
        return 101.29*((T(h)+273.1)/288.08)**(5.256)
def p(h):
    return P(h)/(0.2869*(T(h)+273.1))
def vs(h):
    if (288-h*6.5e-3)>0:
        return 20.05*(288-h*6.5e-3)**0.5
    else:
        return 0
def f(h,v):
    def Ma(v):
        if vs(h)==0:
            return 6
        else:
            return v/vs(h)
    if Ma(v)<0.3:
        return 0.5*0.93*p(h)*0.5*v**2
    elif 0.3<=Ma(v)<1.1:
        return 0.5*(0.516-0.625*Ma(v)+1.085*Ma(v)**2+0.5)*p(h)*0.5*v**2
    elif 1.1<=Ma(v)<=6:
        return 0.5*(1.247-0.101*Ma(v)+0.008*Ma(v)**2+0.5)*p(h)*0.5*v**2
    else:
        return 0.5*1.429*p(h)*0.5*v**2
def G(h):
    return (190*5.965e24/(h+6.371e6)**2)*6.67e-11
h=100000
v=0
dt=0.01
t=0
f1=gcurve(color=color.blue)
while h>0:
    a=(G(h)-f(h,v))/190
    v=v+a*dt
    h=h-v*dt
    t=t+dt
    f1.plot(t,f(h,v))
    if abs(f(h,v)/190-2.82)<0.01:
        t1=t
    if abs(f(h,v)/190-12.36)<0.01 and t-t1>50:
        t2=t
print(t2-t1)
```

```

from math import*
import matplotlib.pyplot as plt
import numpy
from scipy import integrate
from vpython import *
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
def T(h):
    if h>25000:
        return -131.21+0.00299*h
    elif 11000<h<25000:
        return -56.46
    else:
        return 15.04-0.00649*h
```

```python
def P(h):
    if h>25000:
        return 2.488*((T(h)+273.1)/216.6)**(-11.388)
    if 11000<h<25000:
        return 22.65*e**(1.73-0.000157*h)
    else:
        return 101.29*((T(h)+273.1)/288.08)**(5.256)
def p(h):
    return P(h)/(0.2869*(T(h)+273.1))
def f(h,v):
    return 100.33*p(h)*0.5*v**2
def G(h):
    return (190*5.965e24/(h+6.371e6)**2)*6.67e-11
h=60000
v=0
dt=0.01
f1=gcurve(color=color.yellow)
while h>0:
    a=(G(h)-f(h,v))/190
    v=v+a*dt
    h=h-v*dt
    f1.plot(h,v)
```

```python

from math import*
import matplotlib.pyplot as plt
import numpy
from scipy import integrate
from vpython import *
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
def T(h):
    if h>25000:
        return -131.21+0.00299*h
    elif 11000<h<25000:
        return -56.46
    else:
        return 15.04-0.00649*h
def P(h):
    if h>25000:
        return 2.488*((T(h)+273.1)/216.6)**(-11.388)
    if 11000<h<25000:
        return 22.65*e**(1.73-0.000157*h)
    else:
        return 101.29*((T(h)+273.1)/288.08)**(5.256)
def p(h):
    return P(h)/(0.2869*(T(h)+273.1))
def vs(h):
    if (288-h*6.5e-3)>0:
        return 20.05*(288-h*6.5e-3)**0.5
    else:
        return 0
def f(h,v):
    def Ma(v):
        if vs(h)==0:
            return 6
        else:
            return v/vs(h)
    if Ma(v)<0.3:
        return 0.5*0.93*p(h)*0.5*v**2
    elif 0.3<=Ma(v)<1.1:
        return 0.5*(0.516-0.625*Ma(v)+1.085*Ma(v)**2+0.5)*p(h)*0.5*v**2
    elif 1.1<=Ma(v)<=6:
        return 0.5*(1.247-0.101*Ma(v)+0.008*Ma(v)**2+0.5)*p(h)*0.5*v**2
    else:
        return 0.5*1.429*p(h)*0.5*v**2
def G(h):
    return (190*5.965e24/(h+6.371e6)**2)*6.67e-11
h=57000
v=0
dt=0.01
t=0
t1=100000
t2=100000
w=0
tgraph=graph(xtitle='t(s)',ytitle='v(m/s)')
f1=gcurve(color=color.blue)
while h>0:
    a=(G(h)-f(h,v))/190
    v=v+a*dt
    h=h-v*dt
    t=t+dt
    f1.plot(t,v)

    if abs(f(h,v)/190-2.82)<0.01:
       t1=t
    if abs(f(h,v)/190-12.36)<0.01 and t-t1>50:
       t2=t
    if t>t1:
        w=w+f(h,v)/190*dt
    if t>t2:
        t1=100000
print(w)
```

```

from math import*
import matplotlib.pyplot as plt
import numpy
from scipy import integrate
from vpython import *
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
def T(h):
    if h>25000:
        return -131.21+0.00299*h
    elif 11000<h<25000:
        return -56.46
    else:
        return 15.04-0.00649*h
def P(h):
    if h>25000:
        return 2.488*((T(h)+273.1)/216.6)**(-11.388)
    if 11000<h<25000:
        return 22.65*e**(1.73-0.000157*h)
    else:
        return 101.29*((T(h)+273.1)/288.08)**(5.256)
def p(h):
    return P(h)/(0.2869*(T(h)+273.1))
def vs(h):
    if (288-h*6.5e-3)>0:
        return 20.05*(288-h*6.5e-3)**0.5
    else:
        return 0
def f(h,v):
    def Ma(v):
        if vs(h)==0:
            return 6
        else:
            return v/vs(h)
    if Ma(v)<0.3:
        return 0.5*0.93*p(h)*0.5*v**2
    elif 0.3<=Ma(v)<1.1:
        return 0.5*(0.516-0.625*Ma(v)+1.085*Ma(v)**2+0.5)*p(h)*0.5*v**2
    elif 1.1<=Ma(v)<=6:
        return 0.5*(1.247-0.101*Ma(v)+0.008*Ma(v)**2+0.5)*p(h)*0.5*v**2
    else:
        return 0.5*1.429*p(h)*0.5*v**2
def G(h):
    return (144*5.965e24/(h+6.371e6)**2)*6.67e-11
h=39045
v=0
dt=0.01
t=0
t1=100000
t2=100000
w=0
tgraph=graph(xtitle='t(s)',ytitle='f(N)')
f1=gcurve(color=color.blue)
while h>0:
    a=(G(h)-f(h,v))/144
    v=v+a*dt
    h=h-v*dt
    t=t+dt
    f1.plot(t,f(h,v))
    if abs(f(h,v)-406)<0.5:
        t1=t
    if abs(f(h,v)-1780)<0.5 and t-t1>50:
        t2=t
    if t>t1:
        w=w+f(h,v)/144*dt
    if t>t2:
        t1=100000
print(w)
```