Time Travelling to Avoid Trump
A Mathematical Analysis

Vincent Macri David White Aviv Haber

November 22, 2017

For Mr. Bouttell. You asked for rigorous. Here it is.

Contents

Part I

The Twin Paradox

Chapter 1

Stating the Twin Paradox

1.1 The problem

Imagine two people, we will call them Alice[1] and Donald[2].

Donald is currently the president of the United States, and he is doing a bad job. Alice doesn't like this situation, so she is going to to time travel to the future when Donald is no longer president.

1.2 Solution to Trump

Special relativity tells us that time will pass slower for an object in motion than an object at rest (Bruni, Dick, Speijer, & Stewart, 2012; Einstein, 1916). This means that if Alice moves very fast, she will experience less time pass for her than on earth, and will effectively time travel to the future.

1.3 The paradox

The paradox here is that special relativity states that the laws of physics are the same in all inertial frames of reference (Bruni et al., 2012; Einstein, 1916). This means that we could also argue that Alice is not moving, and instead the earth is moving, which would result in Alice aging faster than Donald, which is the opposite of what we want and the opposite of what actually happens (Bruni et al., 2012, pp. 593–594).

This is called the twin paradox.

[1] Of computer security white paper fame.
[2] Of presidential infamy.

1.4 Approach to resolution of the paradox

It is commonly thought that general relativity is needed to resolve this paradox because Alice is in an accelerating frame of reference and special relativity cannot handle accelerating frames of reference. That is not true. Special relativity can indeed handle accelerating frames of reference, but it is more difficult (Gibbs, 1996; Weiss, 2016). However, it is much easier to use special relativity to solve this problem than it is to use general relativity. And in fact, we will later see that the acceleration of Alice is irrelevant to the resolution of the paradox.

1.5 Notation and assumptions

1.5.1 Notation

Our notation

In math, Alice will be referred to as A, and Donald will be referred to as D.

Other physics notation

Following is some common non-trivial physics notation that we will be using:

Velocity

$$v = \|\vec{v}\|$$

We will use v to denote $\|\vec{v}\|$.

Speed of light

$$c_0 = 299\,792\,458\,\mathrm{m/s}$$

We will use c_0 as the speed of light in a vacuum. We are using c_0 instead of c because c_0 is the recommended SI notation (BIPM, 2006).

β notation

$$\beta = \frac{v\,\mathrm{m/s}}{c_0\,\mathrm{m/s}}$$

Where v is velocity and c_0 is the speed of light. Also, since nothing can exceed or meet the speed of light, and the direction is not relevant to the amount of time dilation (Bruni et al., 2012; Einstein, 1916), we will say that: $0 \leq \beta < 1$.

1.5.2 Assumptions

Values

We will say that Alice travels a distance of 4 light years at a speed of $0.8\,c_0$ since these are nice numbers to work with (Kogut, 2012, p. 35). She then turns around and comes back to earth.

Turnaround

We will assume that Alice makes an instantaneous turnaround. Later we will show that this assumption has no effect on the resolution of the paradox.

1.5.3 Other terminology

Alice's trip is split up into two parts. First, she is moving away from earth, which we will call the outbound leg of the trip. After, she is moving towards the earth, which we will call the inbound leg of the trip.

Chapter 2

Setup for the Twin Paradox Resolution

2.1 A naive analysis

We have:

$$v = 0.8\,c_0 \implies \beta = 0.8$$
$$d = 4\,\text{ly}$$

And we know this formula from Bruni et al., 2012, p. 583 (modified to use our notation):

$$\Delta t_D = \frac{\Delta t_A}{\sqrt{1-\beta}}$$

Which can be rearranged into:

$$\Delta t_A = \Delta t_D \sqrt{1-\beta^2} \tag{2.1}$$

We can trivially calculate how much time should pass for Donald:

$$\Delta t_D = \frac{2d}{v} = \frac{2 \times 4\,\text{ly}}{0.8\,c_0} = \frac{8\,\text{ly}}{0.8\,c_0} = 10\,\text{y}$$

And how much time should pass for Alice follows by simply plugging this into (2.1):

$$\Delta t_A = 10\,\text{y} \times \sqrt{1-0.8^2}$$
$$\Delta t_A = 10\,\text{y} \times \sqrt{\frac{9}{25}}$$
$$\Delta t_A = 10\,\text{y} \times \frac{3}{5}$$
$$\Delta t_A = 6\,\text{y}$$

So, Donald ages by 10 years, and Alice ages by 6 years.

This answer is right, but the problem is that we started by assuming that Donald is stationary and Alice is moving. However, we could have said that Alice is stationary and Donald is moving, and then we would calculate $\Delta t_A = 10$ and $\Delta t_D = 6$, which is wrong. So doing the analysis this way leads to ambiguity. We must develop a more rigorous way to analyze this problem.

Chapter 3

Resolving the Twin Paradox

3.1 The relativistic Doppler effect and the twin paradox

3.1.1 Recall our assumptions

Recall the assumptions and values we decided to use in subsection 1.5.2:

Distance Alice travels a distance of 4 light years, so $d = 4\,\text{ly}$.

Velocity Alice travels at a speed of $0.8\,c_0$, so $v = 0.8\,c_0$ and $\beta = 0.8$.

The actual values we use do not matter for resolving the paradox. These values were chosen because they give nice numbers when we perform the calculations, which makes the analysis easier to follow.

We also assumed an instantaneous turnaround. This too makes the math easier, but we will show that it does not change the resolution to the paradox. We can assume that Alice is very strong and capable of surviving $479\,667\,932.8\,\text{m/s}^2$ of acceleration.[1] If Alice is not that strong, we can instead say that another person, also named Alice, who is travelling at the same speed as the first Alice but in the opposite direction, passes by the first Alice at the turnaround point and syncs up her clock with the first Alice's clock. This would mean that when the second Alice arrives at earth, her clock will read the same thing as the original Alice's would have if she could survive all of that acceleration. Either way of handling Alice's instantaneous turnaround will work, since we will end up with the same reading on Alice's clock.

3.1.2 The Doppler analysis

Analyzing the twin paradox with the relativistic Doppler effect is helpful because it allows us to calculate what each person sees, and show that they are seeing different

[1] $479\,667\,932.8\,\text{m/s} = 1.6\,c_0$

things, which solves the ambiguity stated in section 2.1.

When we derived the relativistic Doppler equations in section 2.2, we said that only Alice is shining a flashlight once per second. We did this to simplify our notation in that section and make the derivation easier to follow. However, this doesn't work for actually solving the paradox. We must have both Alice and Donald flash their lights once per second, as measured by their own proper time. Both Alice and Donald know that the other person is shining their light once per second.

This means that both of (2.6) and (2.7) will apply to both Alice and Donald.

We will use the following notation here:

f_A The frequency Alice shines her light at according to her own time.

f_D The frequency Donald shines his light at according to his own time.

f'_A The frequency Alice sees Donald shine his light at according to her own time. This is calculated with (2.6) when Alice is moving away from Donald, and with (2.7) when Alice is moving towards Donald.

f'_D The frequency Donald sees Alice shine her light at according to his own time. This is calculated with (2.6) when Alice is moving away from Donald, and with (2.7) when Alice is moving towards Donald.

Alice moving away from Donald

What Alice sees

$$f'_A = f_D\sqrt{\frac{1-\beta}{1+\beta}}$$

$$f'_A = f_D \times \sqrt{\frac{1}{9}}$$

$$f'_A = \frac{1}{3}f_D$$

So, Alice sees Donald's clock running slowly as she is moving away from him.

What Donald sees

$$f'_D = f_A\sqrt{\frac{1-\beta}{1+\beta}}$$

$$f'_D = f_A \times \sqrt{\frac{1}{9}}$$

$$f'_D = \frac{1}{3}f_A$$

So, Donald sees Alice's clock running slowly as she is moving away from him.

Alice moving towards Donald

What Alice sees

$$f'_A = f_D\sqrt{\frac{1+\beta}{1-\beta}}$$
$$f'_A = f_D \times \sqrt{9}$$
$$f'_A = 3f_D$$

So, Alice sees Donald's clock running quickly as she is moving towards him.

What Donald sees

$$f'_D = f_A\sqrt{\frac{1+\beta}{1-\beta}}$$
$$f'_D = f_A \times \sqrt{9}$$
$$f'_D = 3f_A$$

So, Donald sees Alice's clock running quickly as she is moving towards him.

Alice and Donald see the same thing. So why do we end up with Donald aging more if they both see the other age slowly, then they both see the other age quickly?

The answer lies in how long each person sees the other aging at a different speed.

Part II

The Great Trump Escape

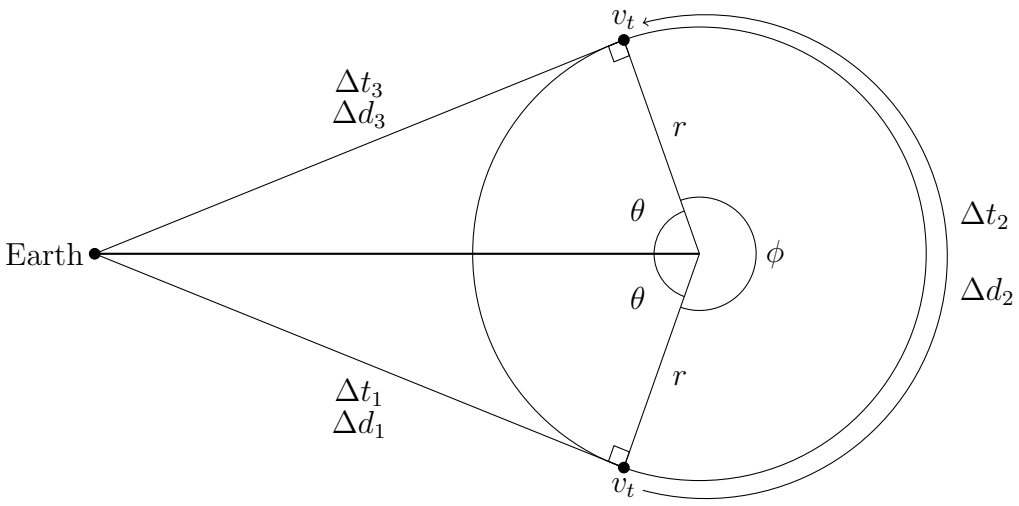
Chapter 4

Deriving the Necessary Formulas

4.1 Turning

4.1.1 Setup

Figure 4.1: Diagram of what happens when Alice turns.



$$\Delta t_1 = \Delta t_3$$
$$\Delta d_1 = \Delta d_3$$

Known values

a The acceleration.

c_0 The speed of light.

T_T The total time.

4.1.2 Calculations

First, recall from Gibbs, 1996 that:

$$v = c_0 \tanh\left(\frac{aT}{c_0}\right)$$

Acceleration distance

$$v = c_0 \tanh\left(\frac{at}{c_0}\right)$$

$$v_t = c_0 \tanh\left(\frac{a\Delta t_1}{c_0}\right)$$

$$\Delta d_1 = \int c_0 \tanh\left(\frac{a\Delta t_1}{c_0}\right) \mathrm{d}\Delta t_1$$

$$\Delta d_1 = \int \frac{c_0}{a} \times \frac{\sinh\left(\frac{a\Delta t_1}{c_0}\right)}{\cosh\left(\frac{a\Delta t_1}{c_0}\right)} \mathrm{d}\Delta t_1$$

$$\Delta d_1 = \frac{{c_0}^2}{a} \int \frac{1}{\frac{\sinh\left(\frac{a\Delta t_1}{c_0}\right)}{\cosh\left(\frac{a\Delta t_1}{c_0}\right)}} \mathrm{d}\Delta t_1$$

$$\Delta d_1 = \frac{{c_0}^2}{a} \ln\left|\cosh\left(\frac{1\Delta t_1}{c_0}\right)\right|$$

$$\frac{a\Delta t_1}{c_0} > 0 \implies \cosh > 0$$

$$\Delta d_1 = \frac{{c_0}^2}{a} \ln\left(\cosh\left(\frac{1\Delta t_1}{c_0}\right)\right)$$

Turn distance

$$\tan\theta = \frac{\Delta d_1}{r}$$

$$\theta = \arctan\left(\frac{\Delta d_1}{r}\right)$$

$$\phi = 2\pi - 2\theta$$

$$\Delta d_2 = \phi r$$

Turn time

$$\Delta t_2 = \frac{\Delta d_2}{v_t}$$

Turn radius

$$a = \frac{v^2}{r} \qquad r = \frac{v^2}{a}$$

4.1.3 The total time

$$\Delta t_1 = \Delta t_3$$

$$\Delta T_T = \Delta t_1 + \Delta t_2 + \Delta t_s$$

$$\Delta T_T = 2\Delta t_1 + \Delta t_2$$

$$\Delta T_T = 2\Delta t_1 + \frac{\left(2\pi - 2\arctan\left(\frac{\Delta d_1}{r}\right)\right)\frac{v^2}{a}}{v}$$

$$\Delta T_T = 2\Delta t_1 + \frac{\left(2\pi - 2\arctan\left(\frac{\frac{{c_0}^2}{a}\ln\left(\cosh\left(\frac{a\Delta t_1}{c}\right)\right)}{\frac{v^2}{a}}\right)\right)v}{a}$$

$$\Delta T_T = 2\Delta t_1 + \frac{\left(2\pi - 2\arctan\left(\frac{{c_0}^2\ln\left(\cosh\left(\frac{a\Delta t_1}{c_0}\right)\right)}{\left(c_0\tanh\left(\frac{a\Delta t_1}{c_0}\right)\right)^2}\right)\right) \times c_0\tanh\left(\frac{a\Delta t_1}{c_0}\right)}{a}$$

Trying to solve for Δt_1 will lead us to a world of pain, so instead we will use a computer to solve it.

This program was written by David White, and the source code is on the following pages:

```
1  import java.util.ArrayList;
2  import java.util.Scanner;
3
4  public class TimeDilationCalc
5  {
6      final int TOTAL_TIME = 99779400;
7      //final int TOTAL_TIME = 8*365*24*60*60;
8      final double C = 299792458;
9      final double C_SQ = C*C;
10     final double PI2 = Math.PI*2;
11     final int acceptableError = 5;
12
13     double acceleration;
14     double cSqOverAccel;
15     int arrivalTimeMargin = 10;
16
17     Scanner s;
18
19     public TimeDilationCalc ()
20     {
21         s = new Scanner (System.in);
22     }
23
24     public void getSimData ()
25     {
26         System.out.print("Enter ship acceleration: ");
27         this.acceleration = s.nextDouble();
28         cSqOverAccel = C_SQ/acceleration;
29     }
30
31     public double velocity (int accelTime)
32     {
33         return C*Math.tanh(acceleration*accelTime/C);
34     }
35
36     public double turnTime (int accelTime)
37     {
38         double accelTimeOverC = acceleration*accelTime/C;
39         double velocity = Math.tanh(accelTimeOverC);
40         double theta = accelTimeOverC;
41         theta = Math.cosh(theta);
42         theta = Math.log(theta);
43         theta *= 1/(velocity*velocity);
44         theta = Math.atan(theta);
45         double turnTime = PI2 - 2*theta;
```

```
46          turnTime *= velocity * C;
47          turnTime /= acceleration;
48          return turnTime;
49      }
50
51      public int findAccelTime (int tripTime)
52      {
53          int accelTime = 0;
54          for (int i = 0; i < tripTime; i++)
55          {
56              if (Math.abs(turnTime (i) + 2*i - tripTime) <
    acceptableError)
57              {
58                  accelTime = i;
59                  break;
60              }
61          }
62          return accelTime;
63      }
64
65      public double getObserverTimeTurning (int tripTime)
66      {
67          double observerTime = 0;
68          int accelTime = findAccelTime(tripTime);
69          for (int i = 0; i <= accelTime; i++)
70          {
71              observerTime += getDilatedTime (1, velocity(i));
72          }
73          observerTime *= 2;
74          double topSpeed = velocity(accelTime);
75          observerTime += getDilatedTime(tripTime - 2*accelTime, topSpeed
    );
76          return observerTime;
77      }
78
79      public double getObserverTimeStraight(int tripTime)
80      {
81          double observerTime = 0;
82          int accelTime = tripTime/4;
83          for (int i = 0; i <= accelTime; i++)
84          {
85              observerTime += getDilatedTime (1, velocity(i));
86          }
87          observerTime *= 4;
88          return observerTime;
89      }
```

```
90
91      public double getDilatedTime (double passengerTime, double
    passengerVelocity)
92      {
93          double dilatedTime = Math.pow(passengerVelocity, 2);
94          dilatedTime /= C_SQ;
95          dilatedTime = 1 - dilatedTime;
96          dilatedTime = Math.sqrt(dilatedTime);
97          dilatedTime = passengerTime/dilatedTime;
98          return dilatedTime;
99      }
100
101     public void optimumTurnTime ()
102     {
103         int tripTime = TOTAL_TIME;
104         double dilatedTime = getObserverTimeTurning(tripTime);
105         double diff = dilatedTime - TOTAL_TIME;
106         while (Math.abs(diff) > arrivalTimeMargin || diff <= 0)
107         {
108            tripTime -= diff;
109            dilatedTime = getObserverTimeTurning(tripTime);
110            diff = dilatedTime - TOTAL_TIME;
111            System.out.println("difference: " + Math.round(diff));
112         }
113         System.out.println("difference: " + Math.round(dilatedTime -
    TOTAL_TIME));
114         System.out.println(tripTime);
115         System.out.println(dilatedTime);
116     }
117
118     public void optimumTurnTime2 ()
119     {
120         System.out.println();
121         System.out.println("-------------");
122         System.out.println("|TURNED PATH|");
123         System.out.println("-------------");
124         System.out.println();
125         int tripTime = TOTAL_TIME;
126         double dilatedTime = getObserverTimeTurning(tripTime);
127         double diff = dilatedTime - TOTAL_TIME;
128         int errorMag = (int) Math.ceil(Math.log10(Math.abs(tripTime)));
129         for (int i = errorMag - 1; i >= 0; i --)
130         {
131             double sign = Math.signum(diff);
132             int modif = (int) (sign*Math.pow(10, i));
133             System.out.print(i);
```

```
134             while (sign == Math.signum(diff) && Math.round(diff) != 0)
135             {
136                 tripTime -= modif;
137                 dilatedTime = getObserverTimeTurning(tripTime);
138                 diff = dilatedTime - TOTAL_TIME;
139                 System.out.print(".");
140                 //System.out.println("difference: " + Math.round(diff))
    ;
141             }
142         }
143         System.out.println();
144         System.out.println();
145         System.out.println("Time after presidency end (arrival): " +
    Math.round(dilatedTime - TOTAL_TIME));
146         System.out.println("Trip Time Elapsed: " + tripTime);
147         System.out.println("Earth Time Elapsed: " + dilatedTime);
148         System.out.println("Time Skipped: " + (dilatedTime - tripTime))
    ;
149     }
150 
151     public void optimumStraightTime ()
152     {
153         int tripTime = TOTAL_TIME;
154         double dilatedTime = getObserverTimeTurning(tripTime);
155         double diff = dilatedTime - TOTAL_TIME;
156         while (Math.abs(diff) > arrivalTimeMargin || diff <= 0)
157         {
158            tripTime -= diff;
159            dilatedTime = getObserverTimeStraight(tripTime);
160            diff = dilatedTime - TOTAL_TIME;
161            System.out.println("difference: " + Math.round(diff));
162         }
163         System.out.println("difference: " + Math.round(dilatedTime -
    TOTAL_TIME));
164         System.out.println(tripTime);
165         System.out.println(dilatedTime);
166     }
167 
168     public void optimumStraightTime2 ()
169     {
170         System.out.println();
171         System.out.println("---------------");
172         System.out.println("|STRAIGHT PATH|");
173         System.out.println("---------------");
174         System.out.println();
175         int tripTime = TOTAL_TIME;
```

```
176         double dilatedTime = getObserverTimeStraight(tripTime);
177         double diff = dilatedTime - TOTAL_TIME;
178         int errorMag = (int) Math.ceil(Math.log10(Math.abs(tripTime)));
179         for (int i = errorMag - 1; i >= 0; i --)
180         {
181             double sign = Math.signum(diff);
182             int modif = (int) (sign*Math.pow(10, i));
183             System.out.print(i);
184             while (sign == Math.signum(diff) && Math.round(diff) != 0)
185             {
186                 tripTime -= modif;
187                 dilatedTime = getObserverTimeStraight(tripTime);
188                 diff = dilatedTime - TOTAL_TIME;
189                 System.out.print(".");
190                 //System.out.println("difference: " + Math.round(diff))
    ;
191             }
192         }
193         System.out.println();
194         System.out.println();
195         System.out.println("Time after presidency end (arrival): " +
    Math.round(dilatedTime - TOTAL_TIME));
196         System.out.println("Trip Time Elapsed: " + tripTime);
197         System.out.println("Earth Time Elapsed: " + dilatedTime);
198         System.out.println("Time Skipped: " + (dilatedTime - tripTime))
    ;
199     }
200
201     public void thing ()
202     {
203         double num = s.nextDouble();
204         num = Math.log10(num);
205         num = Math.ceil(num);
206         System.out.println(num);
207     }
208
209     public static void main(String[] args)
210     {
211         TimeDilationCalc t = new TimeDilationCalc();
212         //t.thing();
213         t.getSimData();
214         t.optimumTurnTime2();
215         t.optimumStraightTime2();
216         System.exit(0);
217     }
218 }
```

4.1.4 The linear case

Bibliography

BIPM. (2006). *SI brochure: the international system of units* (Eighth). Bureau international des poids et mesures. Retrieved from http://www.bipm.org/en/si/si_brochure/

Bruni, D., Dick, G., Speijer, J., & Stewart, C. (2012). *Physics 12*. Nelson Education.

Calculating rocket acceleration. (2011).

Crane, L. & Westmoreland, S. (2009). Are black hole starships possible. arXiv: 0908.1803 [gr-qc]

Einstein, A. (1916). Relativity: the special and general theory.

Gibbs, P. (1996). Can special relativity handle acceleration? In D. Koks (Ed.), *Physics FAQ*. (n.p.)

IAAF. (n.d.). Usain bolt.

Jones, P. & Wanex, L. F. (2006). The Clock Paradox in a Static Homogeneous Gravitational Field1. *Foundations of Physics Letters*, *19*, 75–85. doi:10.1007/s10702-006-1850-3. eprint: physics/0604025

Kogut, J. (2012). *Introduction to relativity: for physicists and astronomers*. Complementary Science. Elsevier Science. Retrieved from https://books.google.ca/books?id=9AKPpSxiN4IC

Koks, D. (2015). Does a clock's acceleration affect its timing rate? In D. Koks (Ed.), *Physics FAQ*. (n.p.)

Lockheed Martin. (n.d.). Creating the blackbird.

Nave, C. R. (1998). Relativistic doppler effect. In *Hyperphysics*. Georgia State University.

Weiss, M. (2016). The twin paradox. In D. Koks (Ed.), *Physics FAQ*. (n.p.)