In [1]:
```
import pandas as pd
import numpy as np
import cvxpy as cp
```

Modeling Trees as Truncated Cones using Convex Optmization

The purpose of this notebook is to fit the truncated cone model introduced in the previous notebook using convex optimization.

Model Formulation

$$V = \frac{1}{3}\pi(r_1^2 + r_1 r_2 + r_2^2)h$$

Where V is the observed volume of the tree and h is the observed height. Then, r_1 and r_2 are the radii of the base and top of the cone, respectively.

We will consider the upper radius to be a fraction of the lower radius. So, we introduce a parameter α where $r_2 = \alpha r_1, \text{ for } \alpha \in (0, 1)$. We will use a grid search to find an alpha that minimizes the fitted sum of squares. Let the lower tree radius be r and the upper be r_2. Since $r_2 = \alpha r$, we can write the volume as:

$$V_i = \frac{1}{3} \cdot \pi h_i r_i^2 (1 + \alpha + \alpha^2), \ \alpha \in (0, 1), i = 1, \ldots, N$$

Where V_i is the observed volume of the ith tree, where i equals $1, \ldots, N$ where N is the total number of observations.

Formulating this as an optimization problem, we seek to minimize the sum of squares of the residuals.

$$\min \sum_{i=1}^{N} \left(V_i - \pi \frac{h_i}{3} r_i^2 (1 + \alpha + \alpha^2) \right)^2 \tag{1}$$

$$\text{s.t. } \alpha \in [0, 1] \tag{2}$$

Observed constants radius (R_i), height (h_i), and volume (V_i), for $i = 1, \ldots, N$.

We want to minimize the sum of squared residuals, so our loss function is:

$$\min \sum_{i=1}^{N} \left(V_i - \pi \frac{h_i}{3} R_i^2 (1 + \alpha + \alpha^2) \right)^2 \tag{3}$$

$$\text{s.t. } \alpha \in [0, 1] \tag{4}$$

To represent this in matrix notation, Let $V = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{pmatrix}$ and $hr^2 = \begin{pmatrix} h_1 r_1^2 \\ h_2 r_2^2 \\ \vdots \\ h_N r_N^2 \end{pmatrix}$

Putting this all together, we have:

$$\min \| V - \pi \frac{h}{3} r^2 (1 + \alpha + \alpha^2) \|_2^2 \tag{5}$$

$$\text{s.t. } 0 \leq \alpha \leq 1 \tag{6}$$

Solutions $\alpha \in \mathbb{R}$, with $\alpha \in [0, 1]$

Code-Friendly Formulation

We seek to write the minimation problem in the form $\min(Ax - b)$ to play nice with `cvxpy`.

First, since $1 + \alpha + \alpha^2$ is one-to-one over $[0, 1]$, we can define $a = 1 + \alpha + \alpha^2$ and then use the value of a to uniquely solve for α. Then, if $\alpha \in [0, 1]$, we know that $1 + \alpha + \alpha^2$ is in the interval $[1, 3]$, so we convert the constraint from the previous problem to $a \in [1, 3]$

$$\min \| \pi \frac{h}{3} r^2 a - V \|_2^2 \tag{7}$$

$$\text{s.t. } 1 \leq a \leq 3 \tag{8}$$

Read Data and Set Parameters

In [2]:
```
trees = pd.read_csv("trees.csv")
trees.head()
```

Out[2]:

	d	h	v
0	0.691667	70	10.3
1	0.716667	65	10.3
2	0.733333	63	10.2
3	0.875000	72	16.4
4	0.891667	81	18.8

In [3]:
```
r = (trees.d/2).to_numpy() # DBH Divided by height
h = trees.h.to_numpy()
V = trees.v.to_numpy()
```

In [4]:
```
a = cp.Variable(1, pos=True) # alpha parameter, target of interest
```

Optimization

In [9]:
```
np.random.seed(6596) # NLP CU Denver Course number used as seed

objective=cp.sum_squares(h/3*np.pi*r**2 * a - V) # Objective Function
constraints = [a>= 1, a<=3]
prob = cp.Problem(cp.Minimize(objective), constraints)
prob.solve()
```

Out[9]:
```
180.82911552857735
```

In [10]:
```
a.value
```

Out[10]:
```
array([1.1595388])
```

Final Optimization Solution

Using the optmized value of a, we solve for the unique value of α by first setting up a quadratic root problem:

$$1 + \alpha + \alpha^2 = a$$

Then, we use the quadratic formula and ignore the negative root that would correspond to an α outside of $[0, 1]$.

$$\frac{-1 \pm \sqrt{1 - 4 * 1 * (1 - a)}}{2}$$

We calculate with code below:

In [18]:
```
print((-1+np.sqrt(1-4*(1-a.value)))/2)
print((-1-np.sqrt(1-4*(1-a.value)))/2)
```

```
[0.13995219]
[-1.13995219]
```

Therefore, the final estimate of α is 0.13995219, or roughly 0.14. Connecting this back to the physical problem, that means that if we represent a tree trunk as a truncated cone with a known lower radius r, the volume of the tree will be approximated best by using an upper radius of $0.14 * r$.

Adding Species

In the absence of data from multiple species, I will generate random species labels for my existing dataset and use that for modeling. In theory, there should be zero effect from

species, since the species labels are generated randomly and have no relation to the outcome variable of volume.

In [21]:
```
# Randomly generate species
np.random.seed(6596) # NLP CU Denver Course number used as seed
trees['species'] = np.random.randint(2, size=len(trees))
trees.head()
```

Out[21]:

	d	h	v	species
0	0.691667	70	10.3	0
1	0.716667	65	10.3	1
2	0.733333	63	10.2	0
3	0.875000	72	16.4	0
4	0.891667	81	18.8	1

We introduce a simple additive effect from species, denoted β. So we modify the original volume formula by adding a parameter by adding β times the species label. Suppose S_i is the species of observation i. Then the volume formula of the truncated cone with an additive constant effect from species, we get:

$$V_i = \frac{1}{3} \cdot \pi h_i r_i^2 (1 + \alpha + \alpha^2) + \beta S_i, \ \alpha \in (0, 1), i = 1, \ldots, N$$

$$\min \| \frac{h}{3} \pi r^2 a + \beta S - V \|_2^2 \tag{9}$$

$$\text{s.t. } 1 \leq a \leq 3 \tag{10}$$

We solve this problem with `cvxpy`. We again transform a back into the parameter α.

In [24]:
```
r = (trees.d/2).to_numpy() # DBH Divided by height
h = trees.h.to_numpy()
V = trees.v.to_numpy()
S = trees.species.to_numpy()
a = cp.Variable(1, pos=True) # alpha parameter, target of interest
b = cp.Variable(1) # beta parameter, additivie species effect
np.random.seed(6596) # NLP CU Denver Course number used as seed

objective=cp.sum_squares(h/3*np.pi*r**2 * a+b*S - V) # Objective Function
constraints = [a>= 1, a<=3]
prob = cp.Problem(cp.Minimize(objective), constraints)
prob.solve()
```

Out[24]:
```
180.0840846549221
```

In [28]:
```
print(f"Alpha Value: {(-1+np.sqrt(1-4*(1-a.value)))/2}")
print(f"Beta value: {b.value}")
```

```
Alpha Value: [0.13535309]
Beta value: [0.28198411]
```

We expected the β parameter value to be zero, since there is no actual effect from species. But this effect is small relative to the average observed volume of a tree and the average fitted volume from the truncated cone model.

In [40]:
```
print(f"Average Observed Volume: {trees.v.mean()}")
print(f"Optimized Species Effect: {b.value[0]}")
print(f"Species Effect, Percent of Average Volume: {100*b.value[0]/trees.v.mean()}"
```

```
Average Observed Volume: 30.17096774193548
Optimized Species Effect: 0.13760595940119233
Species Effect, Percent of Average Volume: 0.4560873240069456
```

So even though the parameter β is nonzero, the optmized value represents less than 1% of the average tree volume.

As a further check, we repeat the randomly species labeling 1,000 times and examine the resulting β values.

In [41]:
```
np.random.seed(6596) # NLP CU Denver Course number used as seed
nreps = 1000
betas = np.zeros(nreps) # initialize vector of beta valuees
for i in range(0, nreps):
    trees['species'] = np.random.randint(2, size=len(trees)) # randomly assign spec
    S = trees.species.to_numpy() # re-extract species label to array
    a = cp.Variable(1, pos=True) # re-declare beta parameter
    b = cp.Variable(1) # re-declare beta parameter
    objective=cp.sum_squares(h/3*np.pi*r**2 * a+b*S - V) # Objective Function
    constraints = [a>= 1, a<=3]
    # Solve problem
    prob = cp.Problem(cp.Minimize(objective), constraints)
    prob.solve()
    # Extract beta value for this iteration
    betas[i]=b.value[0]
```

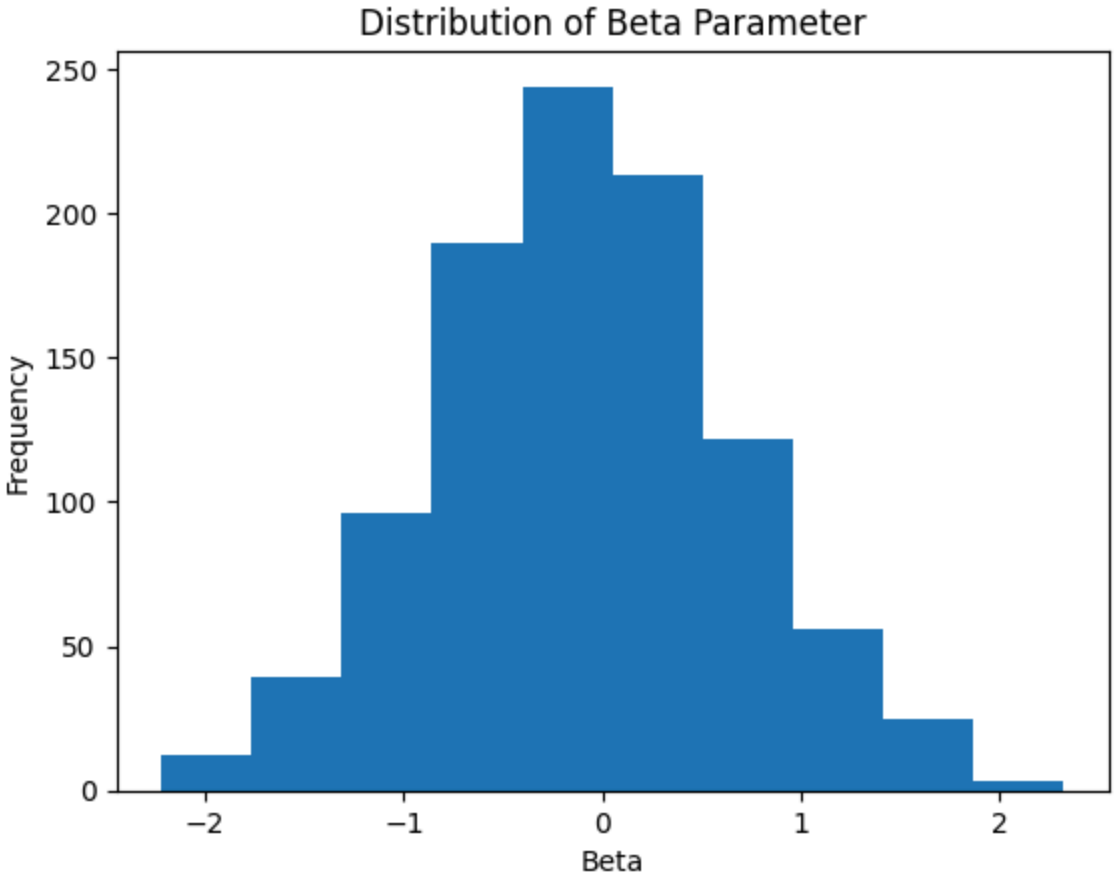
In [45]:
```
print(f"Average Beta Value: {betas.mean()}")
print(f"Average Beta Value as Percent of Mean Volume: {100*betas.mean() / trees.v.m
```

```
Average Beta Value: -0.08348961457711454
Average Beta Value as Percent of Mean Volume: -0.2767216991222657
```

In [50]:
```
import matplotlib.pyplot as plt

plt.hist(betas)
plt.title("Distribution of Beta Parameter")
plt.xlabel("Beta")
plt.ylabel("Frequency")
```

Out[50]:
```
Text(0, 0.5, 'Frequency')
```



The resulting distribution of β values is centered on zero and has a roughly normal distribution. This matches what we would expect theoretically.

Discussion and Further Research

This report shows how to solve a geometrically formulated estimate of tree volume, given the typical forestry field measurements of tree diameter and height. This formulation of a volume model has advantages over standard statistical approaches in a few ways. First, the basic truncated cone model has only one uncertain parameter, while typical linear regression methods would use up degrees of freedom estimating typically 4 or more paramters (one each for height, diameter, random error, and a constant mean effect). Second, the geometrically inspired model gives potentially more physical insight. The truncated cone formulation can be used to get a natural estimate of surface area of the tree, for example. The surface area of a tree might be a useful value to estimate, if for instance the logs were to be painted or wrapped in some material and you wanted an estimate of how much you would need. A standard linear regression model would provide zero estimate of surface area.

This report shows how an effect from species could be added, given more data for different tree species. Numerical experimentation above shows that if species labels are random, and thus have no relationship with volume, the resulting estimates of the additive effect from species vary about the expected value of zero with a roughly normal distribution.

This project could be developed and extended to a scientific contribution to forestry modeling. First, we could collect more data, preferrably the data used to develop models published in the academic literature and cited in the references of this report. Next, we could compare the truncated cone volume model to the various other models in the literature. Preliminary investigation shows that the truncated cone volume model fits this data of 31 cherry trees more accurately and with less uncertainty of relevant parameters.