

Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„Calculating timber harvest costs based solely on spatial predictors exemplified by the Colorado State Forest“

vorgelegt von

BSc Ulrich Strötz

102778, UNIGIS MSc Jahrgang 2012

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)“

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

Berlin, 20.10.2014

Acknowledgements

I would like to thank the entire UNIGIS Team for their support during my studies and to Prof. Joseph Strobl for the supervision of this thesis.

In addition, a big thank you to the Colorado State Forest Service for providing me with useful data, which was essential for this research. In particular I wish to thank Hunter Townsend (State Forest Manager) and Russ Gross (Forester).

A very special thanks is dedicated to Ecotrust, which was inspirational in the finding of my research question and which guided me throughout the research process. In particular I would like to thank Matt Perry (Senior Applications Developer) for his technical advice and Mike Mertens (Director of Spatial Analysis) for his patience, creativity and curiosity as we discussed the thesis.

Statement of Authorship

Hereby I, Ulrich Strötz, declare that this master thesis was written without the help of a third party and without the use of sources other than the ones cited in this paper. In addition, I certify that everything in this paper that is not of my own, including; writing, images, figures and tables, are appropriately referenced.

Berlin, 20.10.2014

Abstract

Optimization models for ecological forestry approaches require consideration of a variety of spatial features, including *Harvest Costs*, in order to maximize triple bottom line returns. The models require a pre-generated dataset with the potential *Harvest Costs* for the entire landscape in order to iterate through millions of potential solutions and compare results in terms of an objective function. Since the composition and the structure of the forest systems are usually not available for an entire landscape, a model is required that calculates *Harvest Costs* solely based on *Spatial Predictors*. *Spatial Predictors* can be determined via Geographic Information Systems. Currently no existing study investigates the significance of *Spatial Predictors* on *Timber Harvest Cost*. Therefore it is also not known if the significance of *Spatial Predictors* on *Harvest Cost* is high enough to calculate *Timber Harvest Costs* solely based on *Spatial Predictors*. This study answers these research questions with the following method:

A dataset containing 160,000 test units based on existing harvest data of the Colorado State Forest is created. The dataset contains for each unit the *Spatial* and *Non-Spatial Predictors* of *Timber Harvest Costs*. Each unit is run through a created *Harvest Cost Model*, which is based on existing literature and equations. The *Harvest Cost Model* returns for each unit a *Harvest Cost* per ton. The spatial and non-spatial input data are then used as independent variables in a multiple linear regression model, with the resulting *Harvest Cost* from the model as the dependent variable. From the created regression model, a spatially explicit regression model is derived by excluding the non-spatial variables. The spatially explicit regression model calculates *Harvest Costs* solely based on *Spatial Predictors*. Finally, based on the spatially explicit regression model, a *Cost Surface* is created. The *Cost Surface* contains the *Harvest Cost* for any given location throughout the landscape.

The created spatially explicit regression model has an R-squared of 0.42. Therefore *Spatial Predictors* predict 42% of *Timber Harvest Costs*. Calculating *Timber Harvest Costs* with an accuracy of 42% is not enough to calculate absolute *Harvest Costs* solely based on *Spatial Predictors*. But for optimization models relative *Harvest Costs* are sufficient, since relative *Harvest Cost* allows the comparison of *Costs* of different stands and scenarios. An accuracy of 42% is then enough to estimate relative *Harvest Costs*.

Therefore the results of this research make it possible to include *Harvest Costs* in optimization models for ecological forestry approaches. With their inclusion optimization models are significantly improved.

Table of Contents

Acknowledgements	2
Statement of Authorship	3
Abstract	4
Table of Contents	5
List of Figures	7
List of Tables	9
List of Equations	9
1. Introduction	10
Research Problem	10
Structure of the Paper	14
2. Model Development	15
3. Data Creation	18
3.1. Independent Variables (Input Data)	18
3.2. Dependent Variable (Cost Data)	27
4. Statistical Analysis	28
4.1. Descriptive Statistics of the Independent Variables	28
4.1.1. Individual Variable Description	28
4.1.2. Correlation of the Independent Variables	30
4.2. Regression Model	36
4.2.1. Fitting the Model	36
4.2.2. Validating the Model	44
4.3. Spatial Regression Model	48
4.3.1. Fitting the Model	48
4.3.2. Validating the Model	49
4.4. Comparing the Two Models	52
5. Cost Surface	53
5.1. Data Overview	53
5.2. Spatially explicit Cost Equation	53
5.3. Cost Surface	53
6. Results	57
6.1. Statistical Results	57
6.2. Colorado State Forest Harvest Costs Results	58
6.3. Cost Surface Results	60
7. Application Example	61
7.1. Need of Application	61
7.2. Application Description	61
7.2.1. Application Front-End	62
7.2.2. Application Back-End.....	66
8. Discussion	71
8.1. Discussion of the Statistical Analysis	71
8.2. Discussion of the CSF Harvest Costs	72

9. Conclusion.....	74
Bibliography.....	75
Appendix	80
Appendix I.....	80
Appendix II.....	81
Appendix III.....	89
Appendix IV	96
Appendix V	98
Appendix VI	100
Appendix VII	108

List of Figures

Figure 1: Map of CSF	12
Figure 2: Harvest Process CSF	13
Figure 3: Map of timber sales in the CSF	19
Figure 4: Sample stand.....	20
Figure 5: Sample stand with sub units	20
Figure 6: Sample stand with input data values.....	21
Figure 7: Schema concatenated sub units	21
Figure 8: Sample stand with value for VPT	22
Figure 9: Sample stand with value for VPT, TPA	23
Figure 10: Schema createQuaterCentroids function	23
Figure 11: Schema downloadOSMroad function.....	24
Figure 12: Schema landing function.....	24
Figure 13: Schema Skidding Distance calculation.....	24
Figure 14: Sample stand with values for VPT, TPA, SD.....	25
Figure 15: Sample stand with all input values	25
Figure 16: Schema cost_func function	27
Figure 17: Schema concatenated sub units	27
Figure 18: Histogram Slope	28
Figure 19: Normal Q-Q Plot Slope	28
Figure 20: Histogram Volume per Tree	29
Figure 21: Normal Q-Q Plot Volume per Tree	29
Figure 22: Histogram Slope	29
Figure 23: Normal Q-Q Plot Slope	29
Figure 24: Histogram Skidding Distance	30
Figure 25: Normal Q-Q Plot Skidding Distance	30
Figure 26: Plot Trees per Acre and Volume per Tree.....	31
Figure 27: Plot Trees per Acre and Slope	32
Figure 28: Plot Trees per Acre and Skidding Distance.....	32
Figure 29: Plot Volume per Tree and Slope	33
Figure 30: Plot Volume per Tree and Skidding Distance.....	34
Figure 31: Plot Slope and Skidding Distance	34
Figure 32: Plot Cost and Volume per Tree.....	37
Figure 33: Plot Cost and Trees per Acre.....	37
Figure 34: Plot Cost and Slope	37
Figure 35: Plot Cost and Skidding Distance.....	37
Figure 36: Plot Cost and transformed Volume per Tree	39
Figure 37: Regression Model Residuals vs Fitted Plot	47
Figure 38: Regression Model Normal Q-Q Plot.....	47
Figure 39: Regression Model Residuals vs Leverage Plot	48
Figure 40: Regression Model Scale Location Plot.....	48
Figure 41: Spatial Regression Model Residuals vs Fitted Plot	51
Figure 42: Spatial Regression Model Normal Q-Q Plot	51
Figure 43: Spatial Regression Model Residuals vs Leverage Plot	52
Figure 44: Spatial Regression Model Scale-Location Plot.....	52
Figure 45: Schema rasterize roads	54
Figure 46: Slope sample	55
Figure 47: Cost Surface sample.....	56
Figure 48: Sample Harvest Cost CSF Comaparison	59

Figure 49: Cost Surface Southern Part of CSF	60
Figure 50: Schema Application	62
Figure 51: Front-End Overview	63
Figure 52: Front-End Workflow	65
Figure 53: Back-End Overview	66
Figure 54: Schema cost_func function	68
Figure 55: Schema createQuaterCentroids function	69
Figure 56: Schema landing function.....	69
Figure 57: Schema Skidding Distance Calculation.....	70

List of Tables

Table 1: Harvest Machine and referring Study	16
Table 2: Machine Cost per productive machine hour	17
Table 3: Harvest Costs CSF	59
Table 4: Harvest Cost Statistics CSF	60

List of Equations

Equation 1: Regression Model with all predictors.....	36
Equation 2: Regression Model with all predictors and power transformation.....	40
Equation 3: Regression Model with all predictors, power transformation, and interaction terms	43
Equation 4: Spatially explicit Regression Model.....	48
Equation 5: Spatially explicit Regression Model.....	53
Equation 6: Final Regression Model with all Predicotr.....	58
Equation 7: Final spatially explicit Regression Model.....	58

1. Introduction

The returns from forest management vary dramatically across geographic space. This is to some degree because of the variability in the composition and structure of the forest systems, but also because costs of logging operations vary. Ecological approaches to forestry can be competitive, however such approaches require consideration of a variety of spatial features, making this type of management far more complex than standard industrial approaches (e.g. harvesting 40 acres in a 40 year rotation cycle). There exist tools that facilitate this understanding and help landowners maximize triple bottom line returns on ecological forestry approaches (Ecotrust 2014, Zuuring, Wood and Jones 1995). There also exist models for estimating *Timber Harvest Costs* for specific locations (Tufts, et al. 1985, Hartsough, Zhang and Fight 2001, Loeffler, Calkin and Silverstein 2006, Smidt, Tufts and Gallagher 2009, Becker, et al. 2008). However, when planning operations across even relatively small size woodlots (<100 acres), the optimization of returns requires pre-existing knowledge of the potential *Costs* for the entire landscape. Optimization models require iterating through millions of potential solutions and comparing results in terms of an objective function. *Harvest Cost* is a crucial factor in optimization models. But calculating *Harvest Costs* during the iteration process would slow down the optimization significantly. Therefore a pre-generated dataset with the potential *Harvest Costs* for the entire landscape is required. This dataset is named *Cost Surface* throughout this paper.

Research Problem

As stated, there exist models for estimating *Timber Harvest Costs* for specific locations. These models are based on time-and-motion-studies as well as on the opinion of experts (Keegan III, Fiedler and Stewart 1995, Conner, Adams and Johnson 2009, Hartsough, et al. 1997). Based on these models, research has been conducted to investigate what predictors influence *Harvest Costs* and what their significance in predicting Cost is:

Silverstein, et al. (2006) researched the influence of tree variables (volume and number of trees) on *Timber Harvest Costs*, but assumed fixed values for *Slope* and *Skidding Distance*. Arriagada et al. (2008) researched the influence of *Slope*, *Volume per Tree* and *Trees per Acre* on *Harvest Costs*, but ignored *Skidding Distance* as a variable by keeping it constant. Keegan et al. (2002) and Loeffler et al. (2006) researched the influence of *Volume per Tree*, *Trees per Acre* and *Skidding Distance* on *Timber Harvest Costs*, but kept *Slope* constant.

Therefore no currently existing study, including the above, investigates the influence of all predictors on *Timber Harvest Costs* including all *Spatial Predictors*. *Spatial Predictors*, which are *Slope* and *Skidding Distance*, are predictors that can be determined through Geographic Information Systems. *Non-spatial Predictors*, which

are *Volume per Tree* and *Trees per Acre*, require fieldwork or other methods of determining their value.

Since no research has been conducted on the influence of all predictors on *Timber Harvest Costs*, the significance of *Spatial Predictors* on *Timber Harvest Costs* is unknown. Therefore it is unknown if it is possible to calculate *Harvest Costs* solely based on *Spatial Predictors*. Two major research questions result:

1. What is the significance of *Spatial Predictors* on *Timber Harvest Costs*?
2. Is it possible to calculate *Timber Harvest Costs* solely based on *Spatial Predictors*?

The significance of *Spatial Predictors* on *Harvest Costs* has never been fully researched. Given that the significance of *Spatial Predictors* is high enough to exclude the *Non-Spatial Predictors*, it is possible to calculate *Harvest Costs* exclusively with *Spatial Predictors*. Determining *Harvest Costs* exclusively with *Spatial Predictors* would bring an enormous advantage to the field of forest planning and management.

Determining *Harvest Costs* solely with *Spatial Predictors* would allow determining the *Harvest Costs* for an entire geographic region at once by using Geographic Information Systems. No knowledge of the composition and structure of the forest systems and therefore no fieldwork is necessary.

Knowing the *Harvest Costs* for an entire landscape would allow including that knowledge in the above stated optimization models. The models iterate through millions of potential solutions and compare results in terms of an objective function.

Response to the Problem

This paper will determine the significance of *Spatial Predictors* on *Timber Harvest Costs* and will create a cost equation that enables the calculation of *Timber Harvest Costs* solely based on *Spatial Predictors*.

This will be accomplished by creating a dataset containing 160,000 test units. The dataset contains for each unit the *Spatial* and *Non-Spatial Predictors* of *Timber Harvest Costs*. Each unit is run through a created *Harvest Cost Model*, which is based on existing literature and equations. The *Harvest Cost Model* will return for each unit a *Cost per ton*. The spatial and non-spatial input data are then used as independent variables in a multiple linear regression model, with the resulting *Harvest Cost* from the model as the dependent variable. From the created regression model, a spatially explicit regression model is derived by excluding the non-spatial variables. The spatially explicit regression model calculates *Harvest Costs* based solely on *Spatial Predictors*. Finally, based on the spatially explicit regression model, a *Cost Surface* is created. The *Cost Surface* contains the *Harvest Cost* for any given location throughout the landscape.

To limit potential influences and variations on the results, e.g. through different terrain or species, an area of interest was determined. Because of the availability of data, the homogeneity of the forest, and a fixed area; the Colorado State Forest (CSF) was selected.

Study Area

The CSF is located in North Central Colorado approximately 80 road miles west of Fort Collins, near the city of Walden.

The State Forest stretches north to south for 30 miles and east to west for one mile at it's narrowest point, and 9 miles at the widest (Hubbard 1988). 52,000 of its 71,000 acres are forest land (Colorado Parks and Wildlife 2014). The map below shows the boundary of the CSF with a satellite image (MapBox 2012).



Figure 1: Map of CSF

The tree species covering the forest include subalpine fir (*Abies lasiocarpa*), Englemann spruce (*Picea engelmannii*), lodgepole pine (*Pinus contorta*), Douglas fir (*Pseudotsuga menziesii*), Colorado blue spruce (*Picea pungens*), ponderosa pine (*Pinus ponderosa*), and limber pine (*Pinus flexilis*). About 60% of all trees are lodgepole pines. Spruce and fir community accounts for 23% and aspen, the only deciduous tree, accounts for 17% (Colorado Parks and Wildlife 2014).

With 60% tree cover, lodgepole pine is the dominating tree species in the forest and also the dominating harvest tree. For purposes of simplification, the study therefore focuses exclusively on harvesting lodgepole pine.

The single used harvest method in the CSF is clearcutting, which essentially removes all trees in a stand in one operation (Society of American Foresters 2010).

Commercial timber harvest at the State Forest is for economical reasons, always conducted with ground-based machines, which can operate at slopes equal or less than 40%. The used harvesting system is a ground-based, mechanized-felling, whole tree system. Which means trees are felled and bunched from drive-to-tree machines (which are assumed for flat ground), or swingboom and self-leveling versions (which are assumed for steeper terrain). The latter is shown in the first image of Figure 2 operating in the CSF. Rubber-tired grapple skidders transport bunches to the landing (image 2 of Figure 2). Trees are processed mechanically with stroke or single-grip processors at the landing as shown in image 3 of Figure 2.



Figure 2: Harvest Process CSF

Structure of the Paper

The Introduction is followed by Chapters 2 through 5, which together serve as the Methods section of this paper. Chapter 2 describes the development of the *Cost Model*, which is based on existing literature. In Chapter 3 the process of creating the dataset based on existing data of the Colorado State Forest is described. This dataset is statically analyzed in Chapter 4. Based on the analysis two cost equations are created. One expressing *Timber Harvest Cost* based on all *Predictors* and the other one expressing *Timber Harvest Cost* solely based on the *Spatial Predictors*. The developed spatially explicit cost equation is used in Chapter 5 to create a *Cost Surface* covering the entire Colorado State Forest. After the Methods chapters, there follows in Chapter 6 the summary of the results of this research. In Chapter 7 an application example of the *Cost Surface* is given. Finally, in Chapter 8 the research results will be discussed and the conclusion will be drawn in Chapter 9.

All units, unless otherwise stated, are in United States customary units. All financial values are in United States dollars.

2. Model Development

A harvest model was developed to estimate relative *Harvest Costs* per ton (\$/ton). The model is based on the Fuel Reduction Cost Simulator (FRCS) (Fight, Hartsough and Noordijk 2006) software. Relevant formulas for a ground-based mechanized-felling whole tree system were taken from the FRCS and were written up in a Python script. The script allows iterations over the model in order to analyze it.

The model consists of three processing activities: felling, transportation to the landing, and processing at the landing, as described above in the study area section of the Introduction (Chapter 1).

Costs for loading the trees on to a log-truck, moving expenses for the machines, and management overhead costs are not included in the model.

The model makes the following assumptions as fixed variables:

- Hard wood fraction of 0%
- Wood density of 39 lb./ft³ (lodgepole pine)
- Average log length measures 32 ft.
- Costs for machine and labor reflect Western Colorado standards (Appendix I)
-

For each of the three activities (felling, transporting, and processing), a cost per cubic foot (\$/ft³) is calculated. The costs are dependent on the following input variables:

- *Slope* (in %)
- *Skidding Distance* (in ft.)
- *Trees per Acre* removed
- *Volume per Tree* (in ft³)

The calculated costs per cubic foot of each step are added together, result in a total *Harvest Cost* per cubic foot, and are converted to a *Harvest Cost* per ton.

The code showing the individual cost calculations can be seen in Appendix II (script CostFunc.py), and is summarized in the following paragraphs.

The FRCS software uses more than 100 productivity equations drawn from the literature for machines doing various operations (Fight, Hartsough and Noordijk 2006). Nine studies for felling, seven studies for skidding and seven studies for processing, are relevant for the ground-based harvesting system discussed in this research. Table 1 shows the discussed machine and its citation in the study.

Felling	Skidding	Processing
Melroe Bobcat (Johnson 1979)	Grapple Skidders (Johnson 1988)	Hahn Stroke Processor (Gonsier and Mandzak 1987)
Chainsaw Heads (Greene and McNeel 1991)	Grapple Skidders (Tufts et. al. 1988)	Stroke Processor (MacDonald 1990)
Intermittent Circular Sawheads (Greene and McNeel 1991)	John Deere 748E (Kosicki 2000)	Roger Stroke Processor (Johnson 1988)
Hydro-Ax 211 (Hartsough 2001)	Cat D5H TSK Custom Track (Henderson 2001)	Harricana Stroke Processor (Johnson 1988)
Timbco 2520 and Cat 227 (Johnson 1988)	JD 748_G-II & TJ 560 (Kosicki 2002)	Hitachi EX150/Keto 500 (Schroder and Johnson 1997)
JD 693B&TJ Timbco 2518 (Gingras 1988)	Tigercat 635 (Boswell 1998)	FERIC Generic (Gingras 1996)
Timbco (Gonsier and Mandzak 1987)	Tigercat 635 (Kosicki 2002)	Valmet 546 Woodstar Processor (Holtzschler and Lanford 1997)
FERIC Generic (Gingras 1996, Plamondon 1998)		
Timbco 420 (Hartsough, et. al. 1997)		

Table 1: Harvest Machine and referring Study

Calculation Steps

The following four calculation steps are done to calculate the cost per cubic foot for each activity.

1. Cost per productive machine hour

For each machine a cost per productive machine hour (PMH) is calculated based on the before-tax machine cost approach (Miyata 1980). The assumptions for the machine purchase prices can be found in Appendix I.

The machines used and the associated costs per PMH (\$/PMH) are shown in Table 2.

Machine	Costs (\$/PMH)
Feller-Buncher DriveToTree	181.62
Feller-Buncher SwingBoom	233.61
Feller-Buncher SelfLeveling	238.35
Skidder small	134.24
Skidder big	189.93
Processor small	209.96
Processor big	265.78

Table 2: Machine Cost per productive machine hour

Which Feller-Buncher machine is used depends on the *Slope* and the associated study.

The two skidder and processor types are each combined by using the approach described in (Hartsough, Zhang and Fight 2001). Final costs per PMH for each of the three depend on tree volume. The bigger the trees are, the less the hourly cost of the machines.

2. Relevance weighting

To prevent excessive extrapolation beyond the range of data for a given study, a relevance weighting function is applied to the estimated cost per unit volume (Hartsough, Zhang and Fight 2001). These relevance weights vary from 1.0, where the study is considered to be highly relevant, to 0.0 in portions of the range where the relationships are not likely to be valid (Fight, Hartsough and Noordijk 2006).

3. Production rate per productive machine hour

For each activity a production rate in cubic feet per PMH (ft³/PMH) is calculated for each pertinent study (Hartsough et al. 2001).

4. Final costs for each activity

The costs per PMH (\$/PMH) are divided by the production rates (ft³/PMH) to give costs per cubic feet (\$/ft³) for each pertinent study (Hartsough, Zhang and Fight 2001). These are multiplied by their respective relevance weights, totaled and then divided by the sum of the weights to give the weighted average cost for the activity (Hartsough, Zhang and Fight 2001). Finally the three costs are added together and converted to \$/ton based on the wood density.

3. Data Creation

For the statistical analysis, that will be done in Chapter 4, dependent and independent variables need to be defined and created. In the following the creation of the dataset is explained. The dataset contains the independent variables, which are the input data to the cost model and the dependent variable, which is the actual *Harvest Cost* calculated by the cost model.

The described data creation process is done by a Python script, which can be seen in Appendix III (script Data.py).

3.1. Independent Variables (Input Data)

The model takes in four input variables per timber stand, which are the independent variables of the analysis:

- **Slope (S)**
Slope represents the incline or steepness of the timber stand. The unit of *Slope* is percent.
- **Skidding Distance (SD)**
Skidding Distance is the distance from the location where the tree is felled, to the landing. The landing is located at the closest road from where a log truck picks up the logs. The unit of *Skidding Distance* is feet.
- **Trees per Acre (TPA)**
Trees per Acre are the number of trees removed from the stand. Since clear cutting is the dominating harvesting technique in the CSF, all standing trees are assumed to be cut.
- **Volume per Tree (VPT)**
Volume per Tree represents the average volume per tree, of the stand's trees. The unit of *Volume per Tree* is cubic foot.

In order to investigate the influence of each variable on the *Harvest Cost*, an input dataset is produced. To make sure the created data are reasonable and represent realistic values for stands in the CSF, the input variables are derived from past timber sales of the Colorado State Forest Service (CSFS). Data for 74 timber sales ranging from the years 2005 to 2014 are available from the CSFS as a shapefile in the projection NAD83/ UTM zone 13N (EPSG:26913) (Townsend 2014). The black polygons in the map below show the timber sales stands across the CSF. The majority of the stands are located in the southern part of the state forest. Initial forest stand conditions were identified through inventory cruises by staff of the CSFS (Townsend 2014). The Forest Vegetation Simulator (FVS) (Dixon 2002) was then applied by the CSFS staff to simulate stand growth and development and estimated removed volumes. The results were added to the shapefile as the attributes *TPA* and *VPT*.

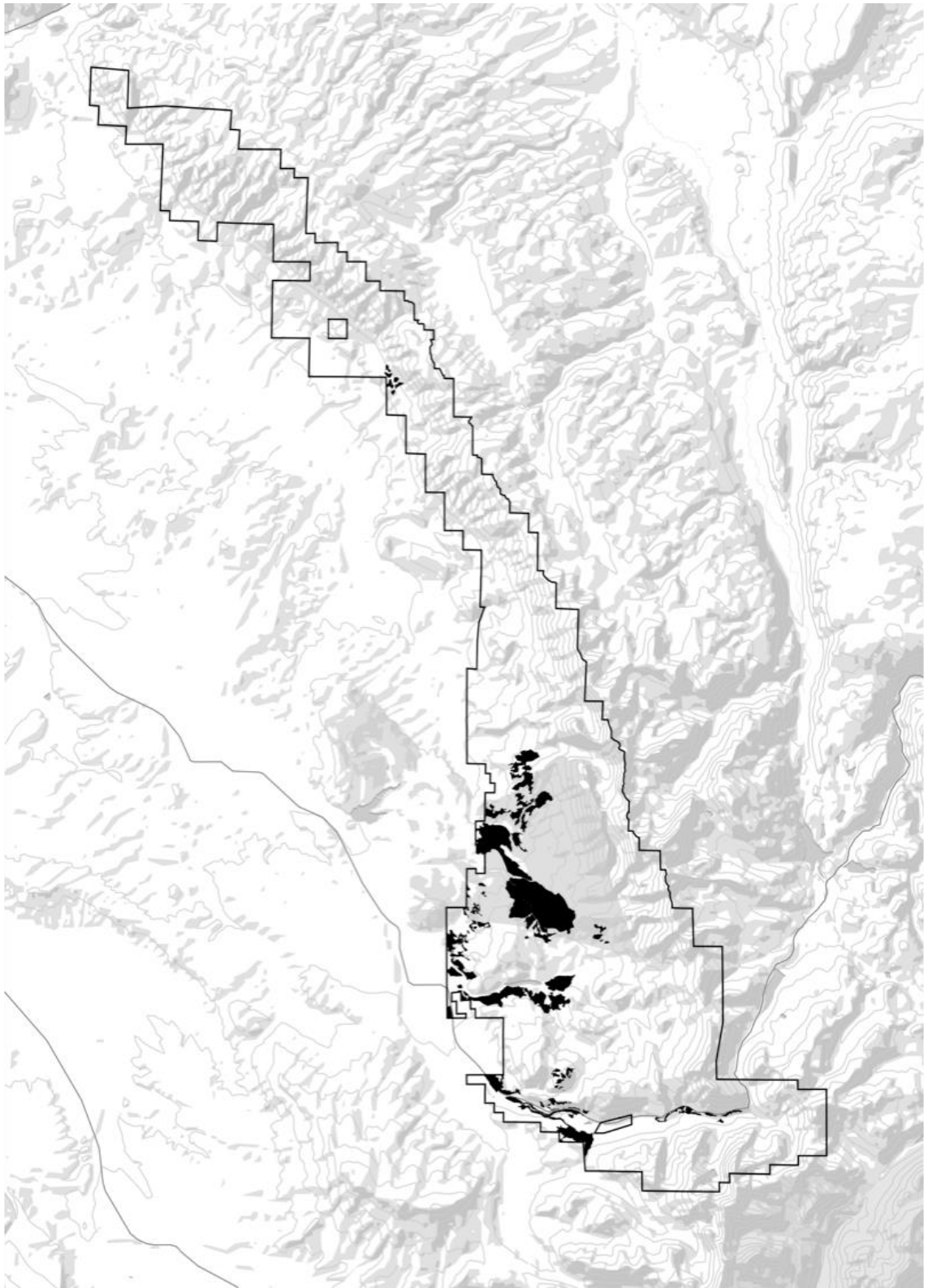


Figure 3: Map of timber sales in the CSF

To demonstrate the process of the data production based on the forest service's data, a sample stand shown in the graphic below is used.

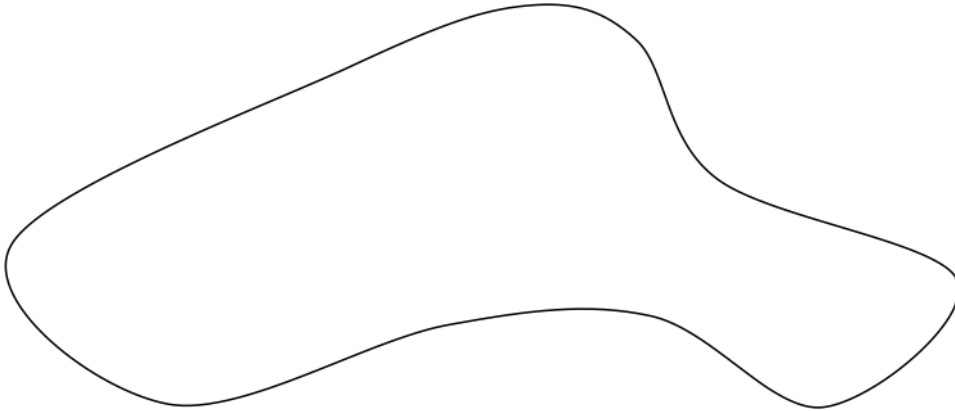


Figure 4: Sample stand

Since *Slope* and *Skidding Distance* vary heavily across the stand, taking average values for the entire stand might bias the results. Therefore the stands are split up into sub units of 10m by 10m.

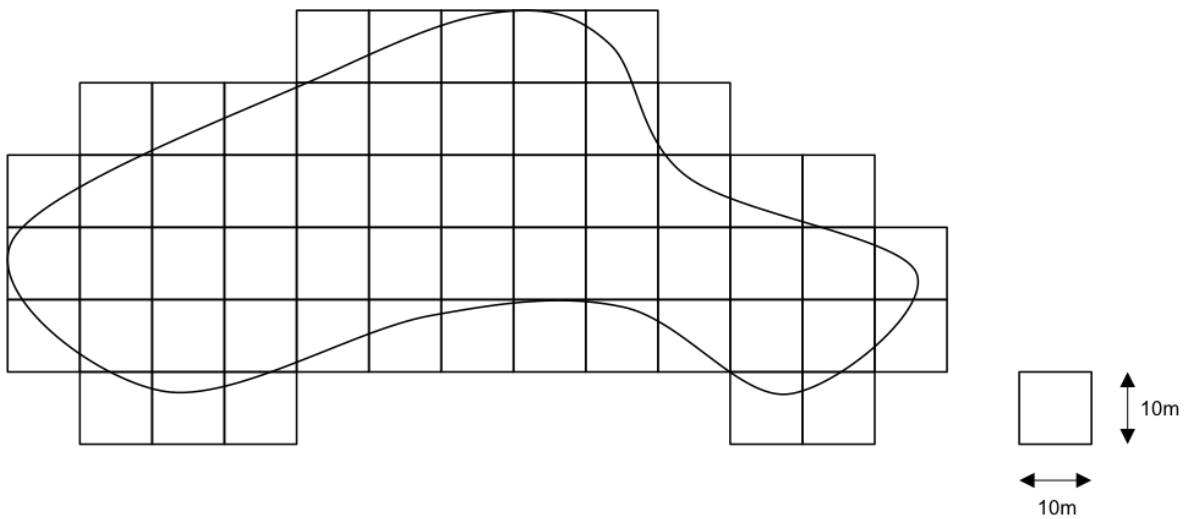


Figure 5: Sample stand with sub units

For each sub unit a value for each of the four variables will be produced and added to the dataset. The shown numbers in the demonstration stand are arbitrary and for visualization purposes only.

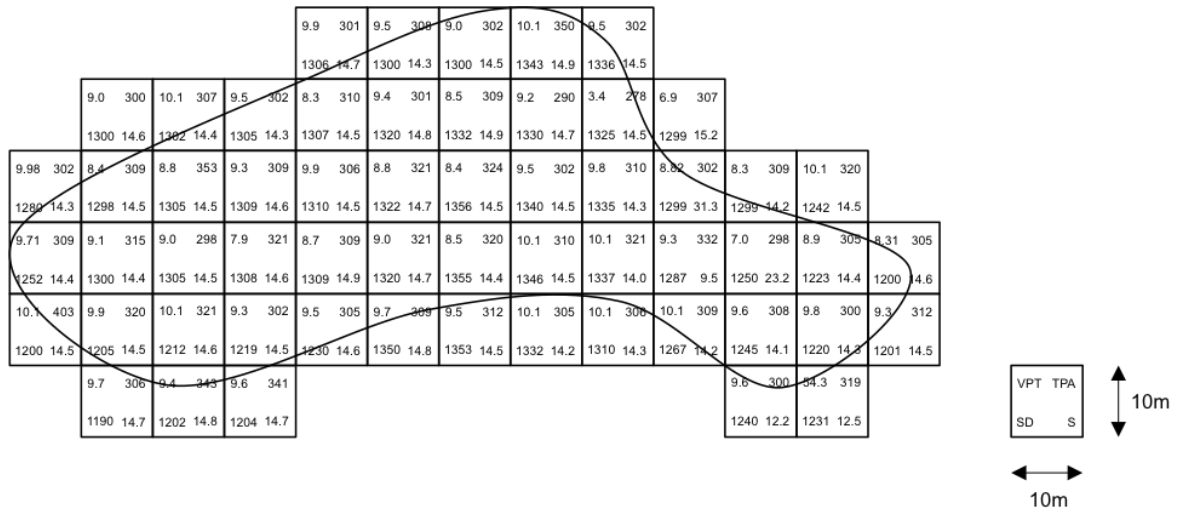


Figure 6: Sample stand with input data values

Overall 161,204 sub units were created with the described procedure covering the 74 stands. The data of the sub units were concatenated and saved to a CSV file. The schema below illustrated the concatenated sub units with its values (5 of the 74 stands are illustrated). The different shades of grey symbolize the different stands, however this is only for visualization purposes. To which stand which sub unit belongs will no longer be relevant, since each sub unit has the relevant data for the analysis assigned.

9.3 312	9.7 306	9.4 343	9.6 341	9.9 301	9.5 308	9.0 302	10.1 350	9.5 302	9.6 300	54.3 319	9.6 300	54.3 319	9.3 312	9.0 300	10.1 307
1201 14.5	1190 14.7	1202 14.8	1204 14.7	1306 14.7	1300 14.3	1300 14.5	1343 14.9	1336 14.5	1240 12.2	1231 12.5	1240 12.2	1231 12.5	1201 14.5	1300 14.6	1302 14.4
9.5 302	8.3 310	9.4 301	8.5 309	9.2 290	3.4 278	6.9 307	9.6 300	54.3 319	9.3 312	9.98 302	8.4 309	8.8 353	9.3 309	9.9 306	8.8 321
1305 14.3	1307 14.5	1320 14.8	1332 14.9	1330 14.7	1325 14.5	1299 15.2	1240 12.2	1231 12.5	1201 14.5	1280 14.3	1298 14.5	1305 14.5	1309 14.6	1310 14.5	1322 14.7
8.4 324	9.5 302	9.8 310	8.82 302	8.3 309	10.1 320	9.3 312	9.71 309	9.1 315	9.0 298	7.9 321	8.7 309	9.0 321	8.5 320	10.1 310	10.1 321
1356 14.5	1340 14.5	1335 14.3	1299 31.3	1299 14.2	1242 14.5	1201 14.5	1252 14.4	1300 14.4	1305 14.5	1308 14.6	1309 14.9	1320 14.7	1355 14.4	1346 14.5	1337 14.0
10.1 403	9.9 320	10.1 321	9.3 302	9.5 305	9.7 309	9.5 312	10.1 305	10.1 306	10.1 309	9.6 308	9.8 300	9.3 332	7.0 298	8.9 305	8.31 305
1200 14.5	1205 14.5	1212 14.6	1219 14.5	1230 14.6	1350 14.8	1353 14.5	1332 14.2	1310 14.3	1267 14.2	1245 14.1	1220 14.3	1287 9.5	1250 23.2	1223 14.4	1200 14.6
9.3 312	9.7 306	9.4 343	9.6 341	9.3 312	9.9 301	9.5 308	9.0 302	10.1 350	9.5 302	9.6 300	54.3 319	9.6 300	54.3 319	9.3 312	9.0 300
1201 14.5	1190 14.7	1202 14.8	1204 14.7	1201 14.5	1306 14.7	1300 14.3	1300 14.5	1343 14.9	1336 14.5	1240 12.2	1231 12.5	1240 12.2	1231 12.5	1201 14.5	1300 14.6
10.1 307	9.5 302	8.3 310	9.4 301	8.5 309	9.2 290	3.4 278	6.9 307	9.6 300	54.3 319	9.3 312	9.98 302	8.4 309	8.8 353	9.3 309	9.9 306
1302 14.4	1305 14.3	1307 14.5	1320 14.8	1332 14.9	1330 14.7	1325 14.5	1299 15.2	1240 12.2	1231 12.5	1201 14.5	1280 14.3	1298 14.5	1305 14.5	1309 14.6	1310 14.5
8.8 321	8.4 324	9.5 302	9.8 310	8.82 302	8.3 309	10.1 320	9.3 312	9.71 309	9.1 315	9.0 298	7.9 321	8.7 309	9.0 321	8.5 320	10.1 310
1322 14.7	1356 14.5	1340 14.5	1335 14.3	1299 31.3	1299 14.2	1242 14.5	1201 14.5	1252 14.4	1300 14.4	1305 14.5	1308 14.6	1309 14.9	1320 14.7	1355 14.4	1346 14.5
10.1 321	9.3 332	7.0 298	8.9 305	8.31 305	1201 14.5	1190 14.7	1202 14.8	1204 14.7	1306 14.7	1300 14.3	1300 14.5	1343 14.9	1336 14.5	1240 12.2	1231 12.5
1337 14.0	1287 9.5	1250 23.2	1223 14.4	1200 14.6	9.3 312	9.0 300	10.1 307	9.5 302	8.3 310	9.4 301	8.5 309	9.2 290	3.4 278	6.9 307	9.6 300
1240 12.2	1231 12.5	1201 14.5	1300 14.6	1302 14.4	1305 14.3	1307 14.5	1320 14.8	1332 14.9	1330 14.7	1325 14.5	1299 15.2	1240 12.2	1231 12.5	1201 14.5	1280 14.3
54.3 319	9.3 312	9.98 302	8.4 309	8.8 353	9.3 309	9.9 306	8.8 321	8.4 324	9.5 302	9.8 310	8.82 302	8.3 309	10.1 320	9.3 312	9.71 309
1298 14.5	1305 14.5	1309 14.6	1310 14.5	1322 14.7	1356 14.5	1340 14.5	1335 14.3	1299 31.3	1299 14.2	1242 14.5	1201 14.5	1252 14.4	1300 14.4	1305 14.5	1308 14.6
9.1 315	9.0 298	7.9 321	8.7 309	9.0 321	8.5 320	10.1 310	10.1 321	9.3 332	7.0 298	8.9 305	8.31 305	10.1 403	9.9 320	10.1 321	9.3 302
1309 14.9	1320 14.7	1355 14.4	1346 14.5	1337 14.0	1287 9.5										
9.5 305	9.7 309	9.5 312	10.1 305	10.1 306	10.1 309										

Figure 7: Schema concatenated sub units

In the following, the production process of the values for each input variable is explained based on the sample stand. The relating functions in the Data.py script can be found in parentheses behind the variable heading.

Volume per Tree (*createVPTarray*)

For each timber stand the average *Volume per Tree* is available from the CSFS data. Since the volume of each tree varies across the stand, a function which produces random normally distributed data around the average value of the stand, is run to produce for each sub unit a value for *VPT* that imitates the natural distribution of the volume across the stand. The *numpy.random.normal* function of the NumPy package is used to create these data. The function draws random samples from a normal (Gaussian) distribution (Bressert 2012). The function takes in a mean value and a standard deviation to create the distribution. The stand's *VPT* value is taken as the mean value and four standard deviations are used to produce the normal distribution ($\frac{VPT}{4}$). In this way each sub unit is assigned a *Volume per Tree*.

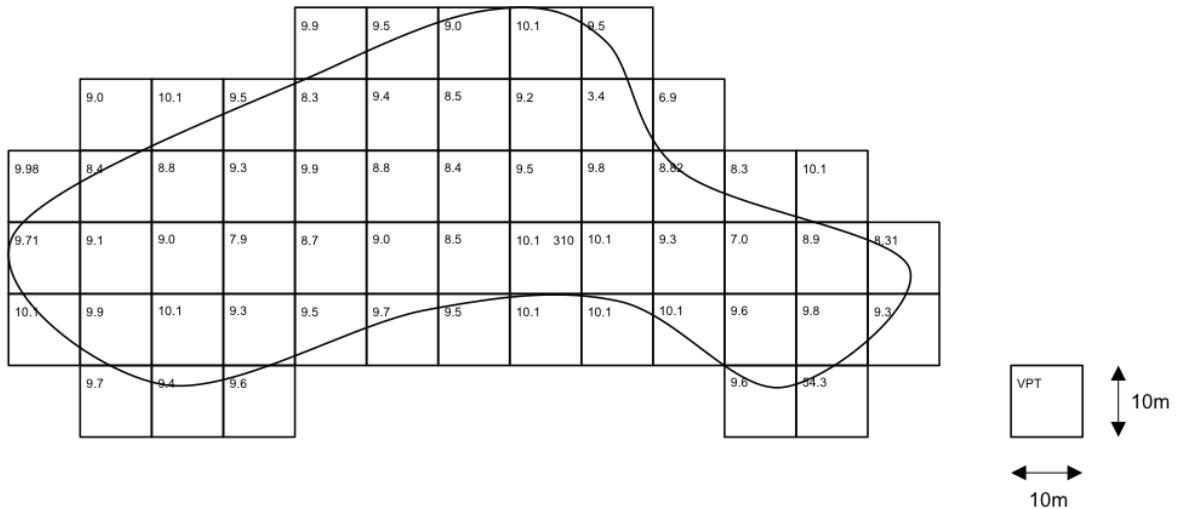


Figure 8: Sample stand with value for VPT

Trees per Acre (*createTPAarray*)

Also the amount of *Trees per Acre* for each timber sale is available from the CSFS data. Like with the variable *Volume per Tree*, the amount of *Trees per Acre* varies across the stand. Therefore the same *numpy.random.normal* function is used to produce for each sub unit a value for *TPA*. Also here the stand's *TPA* value is taken as the mean value and also four standard deviations are used to produce the normal distribution ($\frac{TPA}{4}$). In this way, in addition to the previously assigned *Volume per Tree*, a *Trees per Acre* value is assigned to each sub unit.

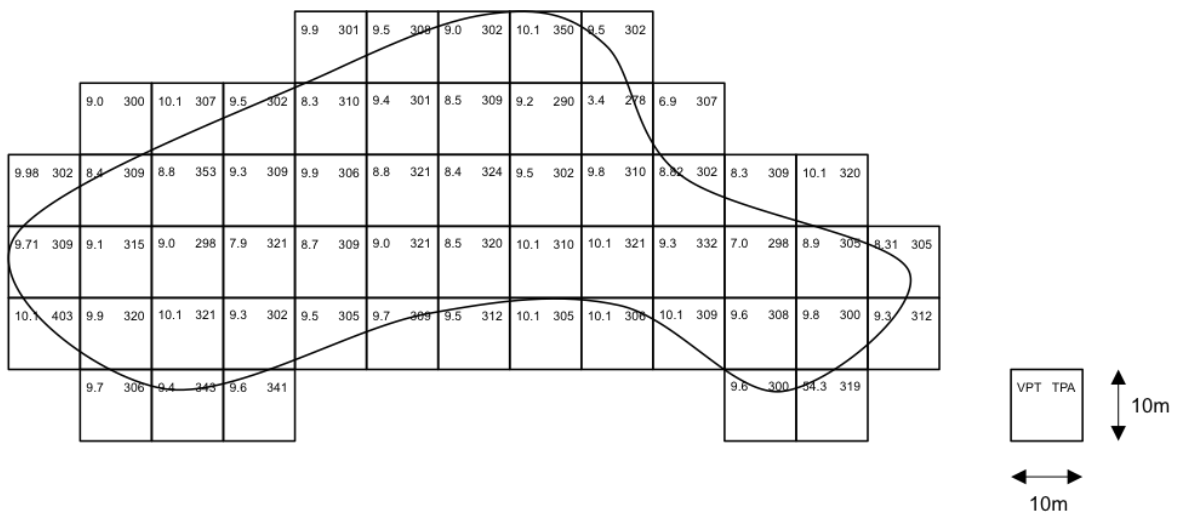


Figure 9: Sample stand with value for VPT, TPA

Skidding Distance (*createSDarray*)

Skidding Distance is the third value that needs to be added to the dataset. As stated above, *Skidding Distance* is the distance from where a tree is felled to the landing. The location of the landing is crucial to determine a reasonable distance. Depending on the size of a stand, usually several landings are created per stand (USDA Forest Service - Northern Research Station Right 2014). In the following the process to determine the average *Skidding Distance* is explained:

1. To imitate the common practice of creating several landings per stand, the stands are divided into four parts. To accomplish this the function *createQuarterCentroids* is applied. First a bounding box around the stand is created. Second the bounding box is split half way north-south and halfway east-west. Third, the four created squares are intersected with the stand, which divides the stand into four parts. Fourth, for each of the four parts of the stand, the centroid is calculated.

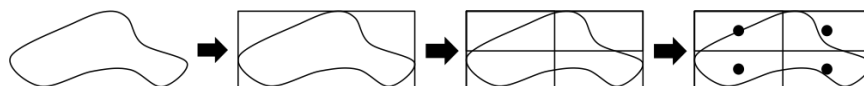


Figure 10: Schema *createQuarterCentroids* function

- Next the function *downloadOSMroad* queries the OpenStreetMap (OSM) Overpass API (OpenStreetMap Wiki contributors 2014) for the road segments surrounding the timber stands. The query is based on the bounding box of the stand. The bounding box is reprojected from NAD83/ UTM zone 13N (EPSG:26913) to WGS 84 (EPSG:4326) in order to query the API. If the queries returns no road segments, the bounding box will be increased and the query is run again.



Figure 11: Schema *downloadOSMroad* function

- For each of the four stand centroids, the Euclidean distance is calculated to each OSM road layer vertices. For this the function *landing* reprojects the downloaded OSM road layer back to NAD83/ UTM zone 13N (EPSG:26913), loops through each vertex, calculates the distance to the stand centroid, and finally returns the geometry of the closest vertex to the stand centroid. In this way four landings on the road are created. In the example below the closest landings for the two eastern centroids are located on the same vertex.

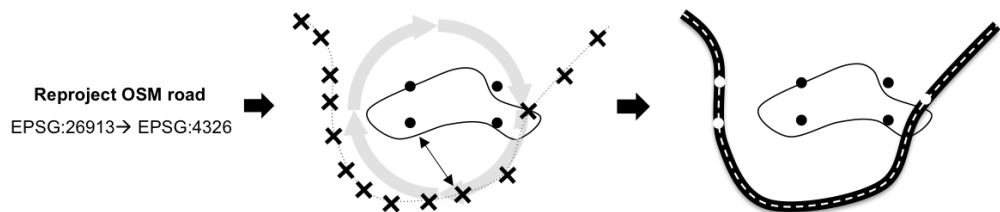


Figure 12: Schema *landing* function

- Last the sub units' centroids are created. The distance to each landing from each centroid is calculated and the shortest distance is assigned as the *Skidding Distance* to the sub unit.

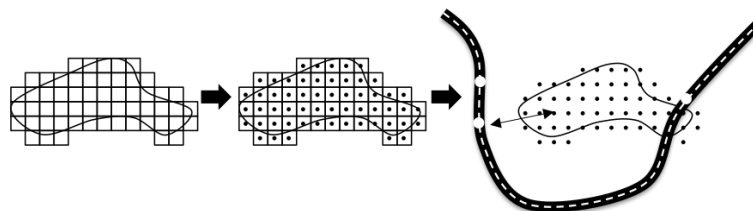


Figure 13: Schema *Skidding Distance* calculation

In this way, in addition to the previously assigned *Volume per Tree* and *Trees per Acre* value, a *Skidding Distance* value is assigned to each sub unit.

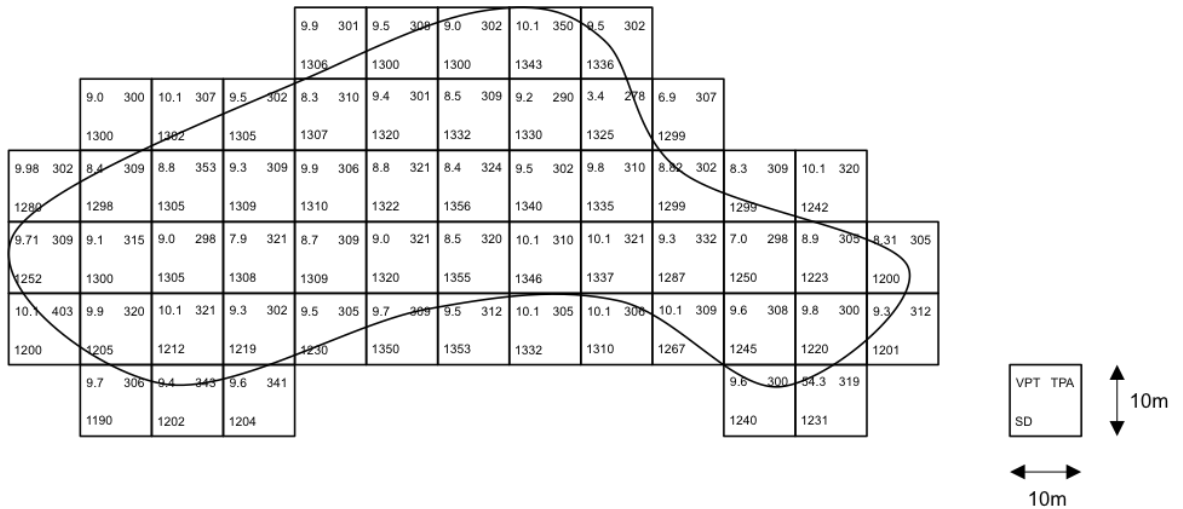


Figure 14: Sample stand with values for VPT, TPA, SD

Slope (*createSarray*)

Last, a value for Slope is added to the dataset. The National Elevation Dataset, a digital elevation raster, is available as a 10m raster from the U.S. Geological Survey (Gesch 2007), which is the same scale the sub units are in. A slope raster in percent is derived from the digital elevation raster. For each sub unit the zonal statistic of the *Slope* is calculated and added to the dataset. Finally the dataset has a value for all four independent variables: *Trees per Acre*, *Volume per Tree*, *Skidding Distance* and *Slope*.

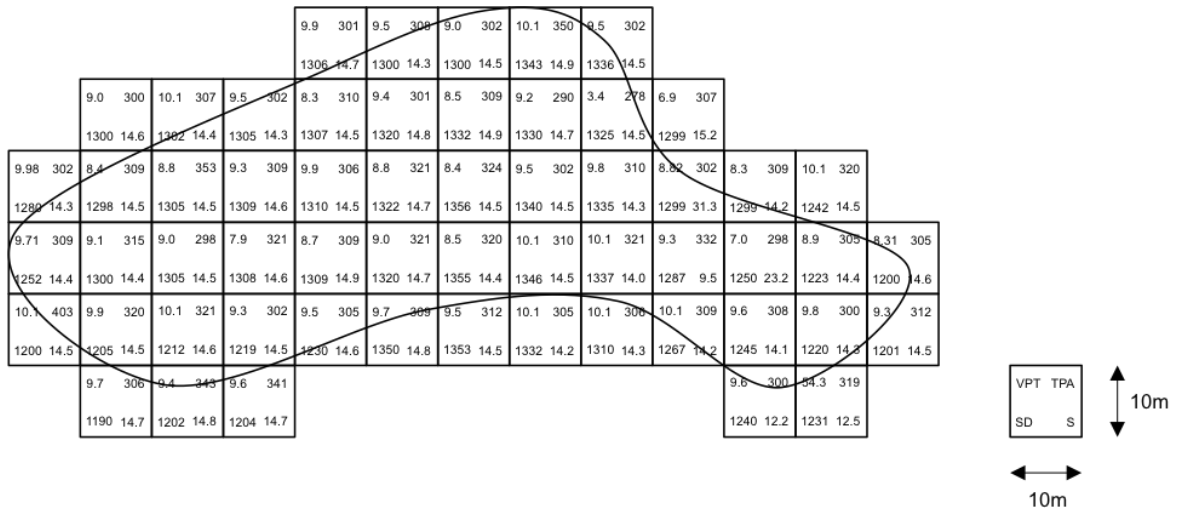


Figure 15: Sample stand with all input values

Input data post-processing (*removeLimits*)

Since *TPA* and *VPT* are created with a function that draws random samples from a normal (Gaussian) distribution (Bressert 2012), it is possible that the generated values are negative, though the values for *TPA* and *VPT* can by nature only be positive.

Therefore sub units containing negative numbers for *TPA* or *VPT* are removed from the dataset. Within the 161,204 sub units this applied to only 14 rows.

4. Statistical Analysis

In Chapter 3, the dataset with the 160,000 sample units with values for each of the four input variables and the associated values for *Cost*, was created. In Chapter 4 the sample units are statistically analyzed. First, in Chapter 4.1 the independent variables are explored. Next, two regression models, one with all four predictors (Chapter 4.2) and the other one with only the two spatial predictors (Chapter 4.3), are developed and validated. Last, both models are compared in Chapter 4.4.

4.1. Descriptive Statistics of the Independent Variables

First, each independent variable is explored in Chapter 4.1.1. This is followed by the investigation of the correlation between them in Chapter 4.1.2.

4.1.1. Individual Variable Description

Trees per Acre

The histogram below shows the distribution of the variable *Trees per Acre*. In average, 305.53 trees per acre with a standard deviation of 109.76 trees per acre cover the stands. The Normal Q-Q plot below shows a straight line, indicating that the *Trees per Acre* are normally distributed across the Colorado State Forest.

The minimum amount of trees is 0.78 per acre and a maximum amount of trees is 1002.79 per acre.

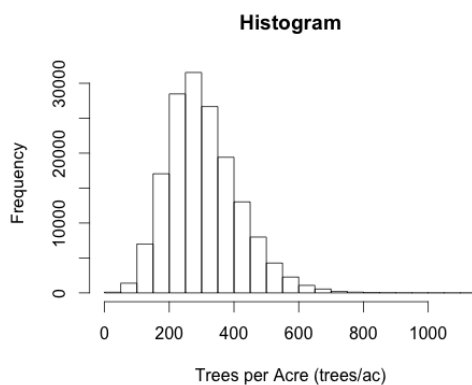


Figure 18: Histogram Slope

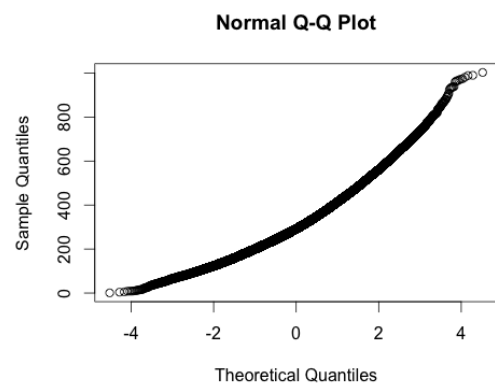


Figure 19: Normal Q-Q Plot Slope

Volume per Tree

The graphic below shows the histogram of the average *Volume per Tree*. An average tree in the given study area has a volume of 9.98 ft³ with a standard deviation of 3.68 ft³. The tree with the least volume has a volume of 0.04 ft³ and the tree with the highest volume has a volume of 49.74 ft³. The Normal Q-Q plot below shows a straight line, indicating that the variable *Volume per Tree* is normally distributed across the Colorado State Forest.

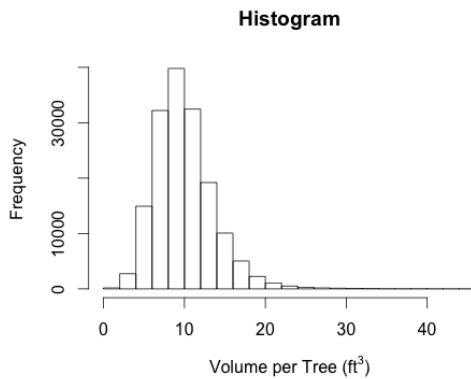


Figure 20: Histogram Volume per Tree

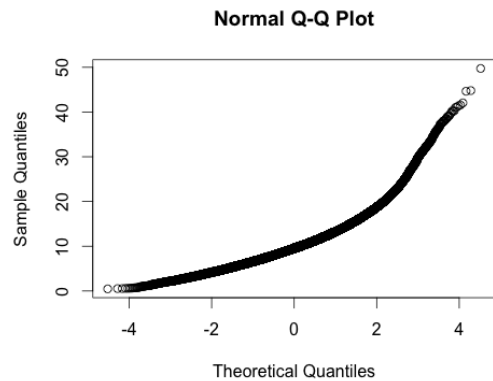


Figure 21: Normal Q-Q Plot Volume per Tree

Slope

The graphic below shows the histogram of the variable *Slope*. The majority of the *Slope* values are spread out across the range between the minimum value of 0% and the maximum value of 40%, which is the operable slope for ground based harvesting machines. The high standard deviation of 10.55% indicates that *Slope* is spread out across this range. Though an accumulation of the values is visible around the mean of 14.75%. Some values can also be found past the 40% threshold, going up to a maximum value of 81.10%. The Normal Q-Q plot below confirms the histogram's view, that the data is not normally distributed.

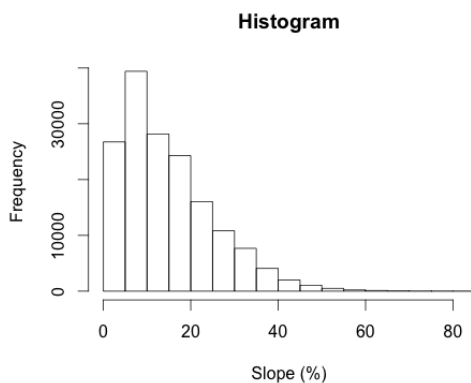


Figure 22: Histogram Slope

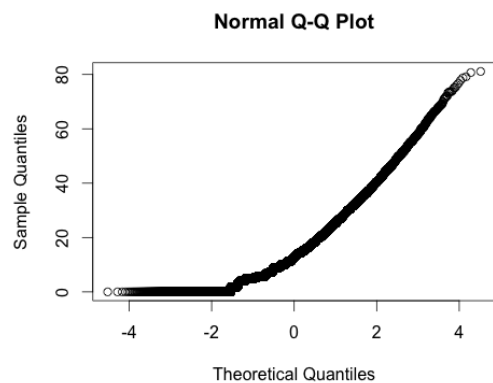


Figure 23: Normal Q-Q Plot Slope

Skidding Distance

The histogram below shows *Skidding Distance* is distributed normally, though at the end of the curve the data varies from a normal distribution. It has a mean of 1303.4 ft. and a standard deviation of 798.8 ft. Also the Normal Q-Q plot below confirms that the data is normally distributed. The minimum *Skidding Distance* is 2.78 ft., which is when the harvest unit is right at the road. The maximum distance is 4919.78 ft.

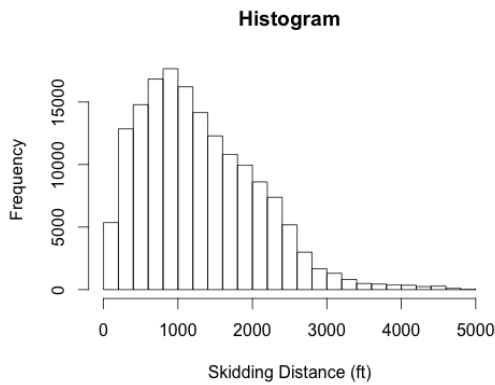


Figure 24: Histogram Skidding Distance

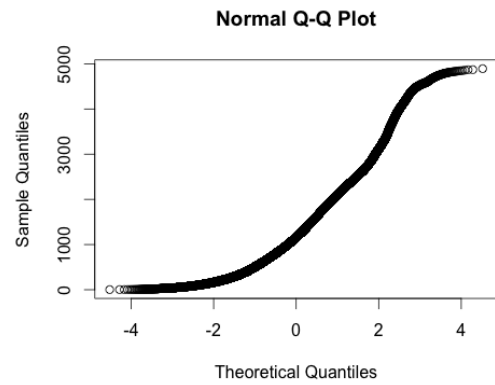


Figure 25: Normal Q-Q Plot Skidding Distance

4.1.2. Correlation of the Independent Variables

In the following the correlation of the independent variables will be investigated. This will be of relevance for the regression model in Chapter 4.2.

The matrix below shows all possible correlations and the correlation coefficients between the variables. Afterwards the correlations are investigated in detail.

	TPA	VPT	S	SD
TPA	1.00000000	-0.36923440	0.034081725	0.056944884
VPT	-0.36923440	1.00000000	-0.003575950	-0.037725313
S	0.03408173	-0.00357595	1.000000000	0.0056265638
SD	0.05694488	-0.03772531	0.005626563	1.000000000

Trees per Acre and Volume per Tree

The correlation coefficient between *Trees per Acre* and *Volume per Tree* is -0.37 with a p-value of less than 0.05. The coefficient of -0.37 indicates a moderate negative linear relationship. The plot of the two variables also supports this. This means the more trees per acre there are, the less the average volume per tree. The p-value is below the conventional threshold of 0.05. This indicates that it is likely that the relationship between the two independent variables is significant. The confidence interval is between -0.373 and -0.365, so the interval excludes zero, which is another indicator that the correlation is significant.

Pearson's product-moment correlation

```
data: costData$TPA and costData$VPT
t = -159.5138, df = 161185, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.3734430 -0.3650106
sample estimates:
      cor
-0.3692344
```

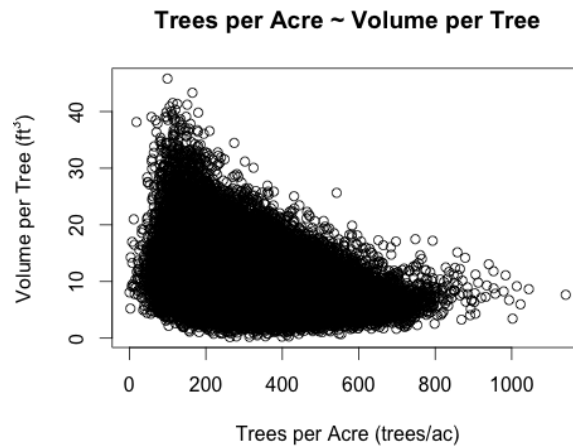


Figure 26: Plot Trees per Acre and Volume per Tree

Trees per Acre and Slope

The correlation coefficient between *Trees per Acre* and *Slope* of 0.03 indicates a weak positive linear relationship. The p-value of less than 0.05 indicates a statistically significant correlation. However the graph does not support a linear relationship, which supports again the low magnitude of the correlation coefficient.

Pearson's product-moment correlation

```
data: costData$TPA and costData$S
t = 13.6912, df = 161185, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.02920482 0.03895701
sample estimates:
      cor
0.03408173
```



Figure 27: Plot Trees per Acre and Slope

Trees per Acre and Skidding Distance

Also the correlation between *Trees per Acre* and *Skidding Distance* with the correlation coefficient of 0.06, indicates a weak positive linear relationship. The p-value of less than 0.05 indicates a statistically significant correlation. However, the graph does not necessary support a linear relationship.

Pearson's product-moment correlation

```

data: costData$TPA and costData$SD
t = 22.8997, df = 161185, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.05207760 0.06180947
sample estimates:
      cor
0.05694488

```

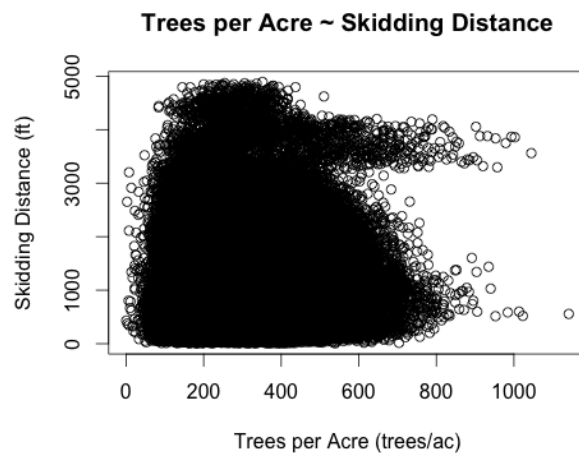


Figure 28: Plot Trees per Acre and Skidding Distance

Volume per Tree and Slope

The correlation coefficient of -0.004 between *Volume per Tree* and *Slope* indicates a weak negative linear relationship. The p-value of less than 0.05 indicates a statistically

significant correlation. However, the graph does not show a linear relation. Also the confidence interval of -0.009 to 0.001 includes zero. Therefore it is possible that the correlation is zero, in which case there would be no correlation.

Pearson's product-moment correlation

```
data: costData$VPT and costData$S
t = -1.4357, df = 161185, p-value = 0.1511
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.008457568  0.001305838
sample estimates:
      cor
-0.00357595
```

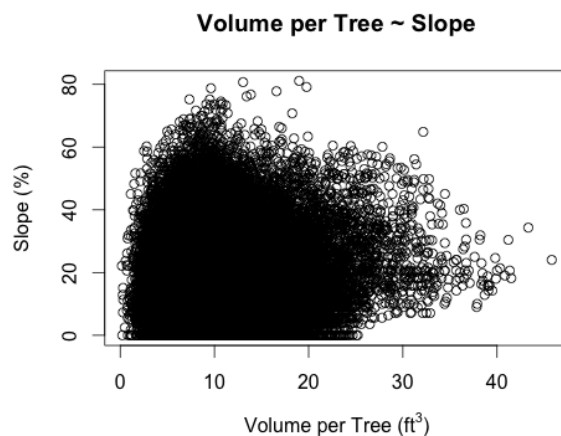


Figure 29: Plot Volume per Tree and Slope

Volume per Tree and Skidding Distance

Also the correlation between *Volume per Tree* and *Skidding Distance* is a weak negative linear relationship. The p-value of less than 0.05 indicates a statistically significant correlation. However, the graph does not indicate a linear relationship.

Pearson's product-moment correlation

```
data: costData$VPT and costData$SD
t = -15.1569, df = 161185, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.04259923 -0.03284960
sample estimates:
      cor
-0.03772531
```

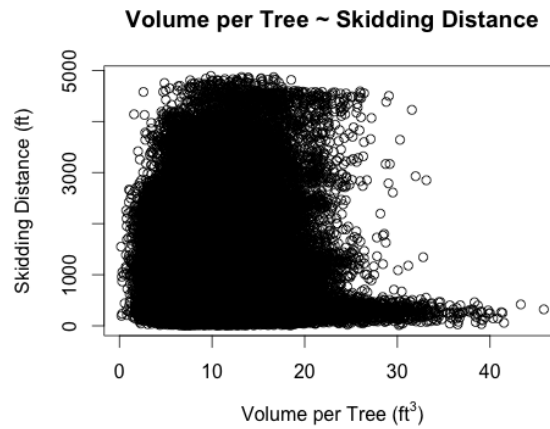


Figure 30: Plot Volume per Tree and Skidding Distance

Slope and Skidding Distance

The correlation coefficient of 0.006 between the two spatial variables *Slope* and *Skidding Distance* indicates a weak positive linear relationship. The p-value of less than 0.05 indicates a statistically significant correlation. However, the graph does not indicate a linear relationship.

Pearson's product-moment correlation

```

data: costData$SD and costData$S
t = 2.259, df = 161185, p-value = 0.02388
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.0007448175 0.0105080400
sample estimates:
cor
0.005626563

```

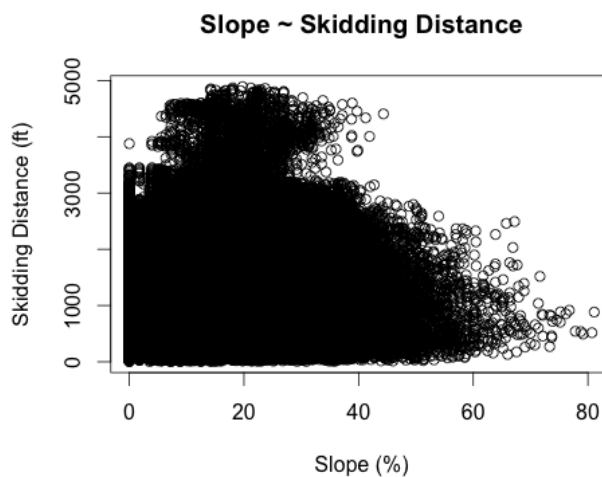


Figure 31: Plot Slope and Skidding Distance

Summary

Overall, it can be said a strong correlation exists between the *Trees per Acre* variable and the *Volume per Tree* variable. Meaning the more trees per acre there are, the less the average volume per tree. Or the other way around; the higher the volume of the trees, the lower the amount *Trees per Acre*

The analysis showed there is a weak correlation between the tree variables and the spatial variables. Also a weak correlation was found among the spatial variables itself.

The results from the correlation analysis will be taken into account in building the regression model.

4.2. Regression Model

A multiple linear regression model is used to model the relationship between the dependent variable *Cost* and the independent input variables. In Chapter 4.2.1 the model is first fitted and then validated in Chapter 4.2.2.

4.2.1. Fitting the Model

The regression respecting all four independent variables as a linear relationship with *Cost*, is the following four-variable-regression:

```
m <- lm(formula = C ~ VPT + TPA + S + SD, data = costData)
```

With the ordinary least-square algorithm (OLS) the following linear model is fitted:

$$C = 42.46 + -1.893 \times VPT + -0.001010 \times TPA + 0.3252 \times S + 0.007242 \times SD + \varepsilon_i$$

Equation 1: Regression Model with all predictors

Since the model is a linear model, the relationships between the independent variables and the dependent variable need to be tested for linearity. Also the model needs to be tested for possible interaction among the independent variables.

4.2.1.1. Testing for Linearity

To better understand if the relationships are linear each independent variable is plotted against *Cost*.

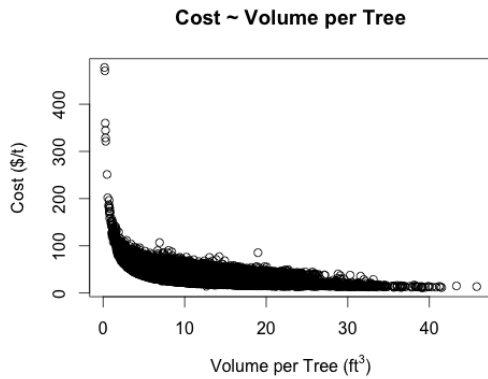


Figure 32: Plot Cost and Volume per Tree

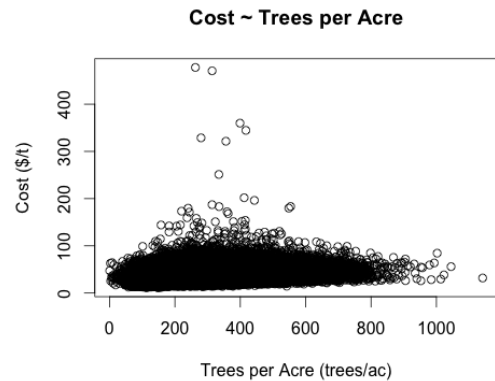


Figure 33: Plot Cost and Trees per Acre

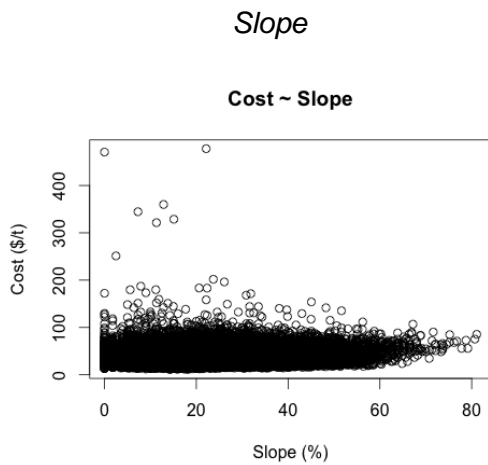


Figure 34: Plot Cost and Slope

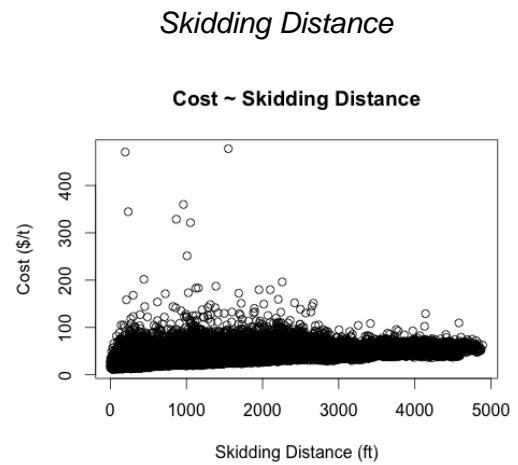


Figure 35: Plot Cost and Skidding Distance

The plots show that *Skidding Distance*, *Slope* and *Trees per Acre* have linear correlation with *Cost*, since a straight line can be created through the plotted values. However, the plot plotting the relationship between *Volume per Tree* and *Cost*, shows that *Volume per Tree* does not have a linear correlation with *Cost*. The plotted values rather show a curve than a line. Since the model is a linear model, all variables need to be linear though. Therefore the variable *Volume per Tree* will be transformed in order to have a linear relationship with *Cost*.

Finding the best power transformation for *Volume per Tree*

Volume per Tree and *Cost* show a non-linear, exponential relationship. *Cost* decreases rapidly as the volume per tree increases. Or in other words, it is more expensive to

harvest small volume trees, than to harvest high volume trees. The curve seems to show a negative exponential correlation.

Several regressions are created to investigate the plotted relationship. To express the exponential relationship between the two variables, exponents between the ranges -2 and 2 are investigated for *Volume per Tree*:

```
R1 <- lm(formula = C ~ VPT, data = costData)
R2 <- lm(formula = C ~ I(VPT^(-1)), data = costData)
R3 <- lm(formula = C ~ I(VPT^(2)), data = costData)
R4 <- lm(formula = C ~ I(VPT^(-2)), data = costData)
R5 <- lm(formula = C ~ I(VPT^(-0.5)), data = costData)
R6 <- lm(formula = C ~ I(VPT^(0.5)), data = costData)
```

Next the Akaike Information Criterion (AIC) is used as a way to select the best fitting model from the set of models (R1 – R6). The model with the lowest AIC score is the best fit in the set of models (Burnham and Anderson, 2002).

	df	AIC
R1	3	1135405
R2	3	1093085
R3	3	1168439
R4	3	1209585
R5	3	1085233
R6	3	1118508

R5 has the lowest AIC score, followed by R2. Therefore the exponent between -1 and -0.5 will be further investigated with additional regressions:

```
R1 <- lm(formula = C ~ I(VPT^(-1)), data = costData)
R2 <- lm(formula = C ~ I(VPT^(-0.9)), data = costData)
R3 <- lm(formula = C ~ I(VPT^(-0.8)), data = costData)
R4 <- lm(formula = C ~ I(VPT^(-0.7)), data = costData)
R5 <- lm(formula = C ~ I(VPT^(-0.6)), data = costData)
R6 <- lm(formula = C ~ I(VPT^(-0.5)), data = costData)
```

Again the Akaike Information Criterion (AIC) is used to select the best fitting models:

	df	AIC
R1	3	1093085
R2	3	1085690
R3	3	1082203
R4	3	1081603
R5	3	1082871
R6	3	1085233

R4 has the lowest AIC score followed by R3. Therefore the exponent between -0.8 and -0.7 will be further investigated with additional regressions.

```
R1 <- lm(formula = C ~ I(VPT^(-0.8)), data = costData)
R2 <- lm(formula = C ~ I(VPT^(-0.79)), data = costData)
R3 <- lm(formula = C ~ I(VPT^(-0.78)), data = costData)
R4 <- lm(formula = C ~ I(VPT^(-0.77)), data = costData)
```

```

R5 <- lm(formula = C ~ I(VPT^(-0.76)), data = costData)
R6 <- lm(formula = C ~ I(VPT^(-0.75)), data = costData)
R7 <- lm(formula = C ~ I(VPT^(-0.74)), data = costData)
R8 <- lm(formula = C ~ I(VPT^(-0.73)), data = costData)
R9 <- lm(formula = C ~ I(VPT^(-0.72)), data = costData)
R10 <- lm(formula = C ~ I(VPT^(-0.71)), data = costData)
R11 <- lm(formula = C ~ I(VPT^(-0.7)), data = costData)

```

Again the Akaike Information Criterion (AIC) is used to select the best fitting models:

	df	AIC
R1	3	1082203
R2	3	1082032
R3	3	1081888
R4	3	1081770
R5	3	1081678
R6	3	1081610
R7	3	1081566
R8	3	1081545
R9	3	1081544
R10	3	1081564
R11	3	1081603

R9 has the lowest AIC score. Therefore the variable *Volume per Tree* needs to be transformed with the exponent of -0.72 in order to have a linear relationship with *Cost*.

To test the assumption the transformed values of *Volume per Tree* will be plotted again against *Cost*.

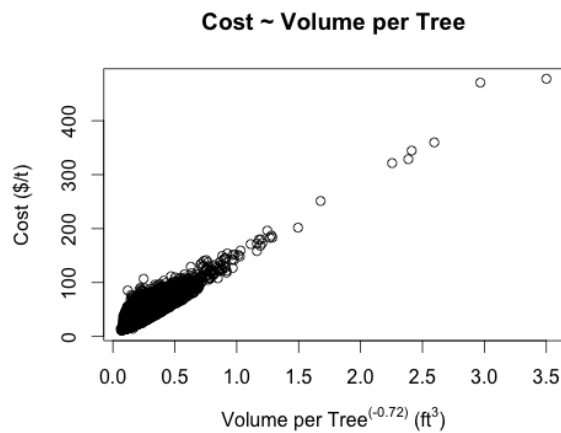


Figure 36: Plot Cost and transformed Volume per Tree

From the plotted values a straight line can be created. Therefore the relationship can now to be considered as a linear relationship. Also the correlation coefficient between *Cost* and *Volume per Tree* went up from -0.66 without the transformation, to 0.77 with the transformation.

To the original regression the exponent of -0.72 will be added to the *Volume per Tree* variable, which results in the new regression:

```
m <- lm(formula = C ~ I(VPT^(-0.72)) + TPA + S + SD, data =
        costData)
```

The new regression also results in a new model:

$$C = -3.667098572 + 133.515209875 \times VPT^{(-0.72)} + -0.003088015 \times TPA \\ + 0.305091203 \times S + 0.007587668 \times SD + \varepsilon_i$$

Equation 2: Regression Model with all predictors and power transformation

By using a power transformation for *Volume per Tree*, all four independent variables have a linear correlation to *Cost*. Therefore a linear regression can be used.

4.2.1.2. Testing for Interaction

When two or more independent variables in combination have an effect on the dependent variable *Cost*, and the effect of these two together differs from the effect of them individually on the dependent *Cost* variable, an interaction effect is occurring in the regression (Jaccard, 2003). The investigated correlations in Chapter 4.1.2 will be used to explore possible interactions within the model for each independent variable. In Chapter 4.1.2 it was shown that the two tree variables *Volume per Tree* and *Tree per Acre* interact the most out of all the interactions among the independent variables. Therefore the two will be investigated for possible interaction within the model. Since this report is ultimately interested in the spatial variables, the interaction between the two spatial variables is investigated. But also the correlation between the spatial variables and the tree data will be investigated. A stepwise forward selection is used for the process.

Volume per Tree and Trees per Acre

Two regression models, one with and one without the interaction term between *Volume per Tree* and *Trees per Acre* are created:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD, costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT,
        costData)
R3 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:I(VPT^(-
0.72)), costData)
```

The Akaike Information Criterion (AIC) is also used here as a way to select the best fitting model from the set of models (R1 – R3). The model with the lowest AIC score is the best fit of the set of models:

	df	AIC
R1	6	574442.8
R2	7	<u>573580.8</u>
R3	7	574442.9

The regression R2 with an interaction between *Volume per Tree* and *Trees per Acre* has the lowest AIC score. Therefore the interaction between the two will be included in the regression.

Slope and Skidding Distance

Also two regressions to identify possible interactions between *Skidding Distance* and *Slope* are created. Again one regression with and one without the interaction are created:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT,
costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD, costData)
```

The AIC is used again to select the better fitting model between the two:

	df	AIC
R1	7	573580.8
R2	8	<u>538015.7</u>

The regression R2 with the interaction term between the two spatial variables has a lower AIC score than without the interaction. Therefore the interaction term will be included in the regression.

Trees per Acre and Slope

Also two regression models, one with and one without the interaction term between *Trees per Acre* and *Slope* are created:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD, costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + S:TPA, costData)
```

The AIC is used again to select the best fitting model:

	df	AIC
R1	8	538015.7
R2	9	<u>534846.3</u>

The regression with the interaction term between *Trees per Acre* and *Slope* has the lower AIC score. Therefore the interaction term will be included in the regression.

Trees per Acre and Skidding Distance

In addition interaction between *Trees per Acre* and *Skidding Distance* will be explored.

Two models, one with the interaction and one without the interaction term are created:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD, costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + SD:TPA, costData)
```

AIC is used again to select the best fitting model:

	df	AIC
R1	8	538015.7
R2	9	<u>537683.6</u>

The regression R2 with the interaction between the *Trees per Acre* and *Skidding Distance* has a lower AIC score. Therefore in the regression the interaction will be included.

Volume per Tree and Slope

Also *Volume per Tree* and *Slope* are investigated. Two models are created to compare the possible interaction:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + SD:TPA + S:TPA, costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + SD:TPA + S:TPA + S:VPT, costData)
```

AIC is used again to select the best fitting model:

	Df	AIC
R1	10	534538.7
R2	11	<u>502358.0</u>

Regression model R2 with the interaction term has the lower AIC score. Therefore, here the interaction term will also be included in the regression.

Volume per Tree and Skidding Distance

Last, the interaction between *Volume per Tree* and *Skidding Distance* will be explored.

Two models are created, one with and one without the interaction term:

```
R1 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + SD:TPA + S:TPA, costData)
R2 <- lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT +
S:SD + SD:TPA + S:TPA + SD:VPT, costData)
```

The AIC is used again to select the best fitting model:

	Df	AIC
R1	10	534538.7
R2	11	524472.9

R2, the regression with the interaction term, has the lower AIC score. Therefore, here the interaction term will also be included in the regression.

The new regression including all the interaction terms is the following:

```
lm(C ~ I(VPT^(-0.72)) + TPA + S + SD + TPA:VPT + S:SD +
SD:TPA + S:TPA + SD:VPT, costData)
```

The new regression also results in a new model:

$$C = -3.556 + 133.6 \times VPT^{(-0.72)} + -0.013 \times TPA + 0.1626 \times S + 0.007395 \times SD + 0.0005438 \times TPA:VPT + 0.00008.250 \times S:SD + -0.0000004.963 \times SD:TPA + 0.0001.673 \times S:TPA + -0.00009.460 \times SD:VPT + S:VPT + \varepsilon_i$$

Equation 3: Regression Model with all predictors, power transformation, and interaction terms

Summary

Due to the minimal increase of the R-squared due to adding interaction terms, three-way interactions will not be tested. Also the minimal increase of the R-squared value of 0.0041 (R-squared of 0.9828 without the interaction terms and 0.9869 with the interaction terms) is reason enough to not include the interaction terms, in order to keep the model simple and understandable.

4.2.2. Validating the Model

After the model is fitted it needs to be validated. The model needs to be tested to assure that it is reasonable and that it is statistically significant. Several indicators are investigated for that. First, the statistical significance of the model is verified. Next, the significances of the coefficients are tested. Then the usefulness of the model and how well the data fits the model is explored. The last step is to explore how well the data satisfies the assumptions of a linear regression.

Statistical significance of the model

The statistical significance of the model can be tested by investigating its F-statistics. The model is significant if any of its coefficients are nonzero (Teetor and Loukide 2011). Or in other words, if all its coefficients are zero; it is not significant. A p-value of less than 0.05 indicates that the model is likely to be significant (Teetor and Loukide 2011). The model in this research has a p-value of smaller than $< 2.2e-16$, which indicates that the model is likely to be significant. Or in other words the probability is $2.2e-16$ that the model is insignificant.

F-statistic: 2.309e+06 on 4 and 161185 DF, p-value: $< 2.2e-16$

Significances of the coefficients

Next, the significances of the coefficients are tested. To check whether the coefficients are significant or not their t-statistics and p-values are investigated. The column estimate shows the estimated regression coefficients. Since they are only estimates, they will never be zero. If they were zero, the variable would have no effect on the dependent variable. To understand statistically, how likely it is that the coefficients are zero; the t-statistics and the p-values need to be explored. The p-value indicates the likelihood that the coefficient is not significant (Teetor und Loukide 2011). Since all the values are below the significance level of $\alpha = 0.05$ ($<2e-16$), it is very likely that all the coefficients are significant. The column t value shows in addition the t-statistic from which the p-value was calculated.

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-3.667e+00	1.546e-02	-237.16	$<2e-16$	***
I (VPT ^{-0.72})	1.335e+02	5.993e-02	2227.67	$<2e-16$	***
TPA	-3.088e-03	3.428e-05	-90.08	$<2e-16$	***
S	3.051e-01	3.362e-04	907.56	$<2e-16$	***
SD	7.588e-03	4.515e-06	1680.67	$<2e-16$	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Usefulness of the model

The R-squared value of the model tells if the model is useful, or what the quality of the model is (Teetor und Loukide 2011). It is the fraction of the variance of *Cost* that is explained by the regression model. The model explains 0.9828 (98.28%) of the variance of *Cost*. The remaining 1.72% is unexplained. This will be discussed in Chapter 8. The adjusted value accounts for the number of variables in the model and is therefore a more realistic assessment of its effectiveness. Since both values are the same here, it is again of no relevance, which one is chosen.

The four predictors and the high sample size of 161185, results in a narrow 95% confidence interval of 0.98259 to 0.98301.

Multiple R-squared: 0.9828, Adjusted R-squared: 0.9828

Fitness of the model to the data

Two things are explored in order to understand how well the model fits the data. First the residuals are explored and second possible outliers of the model are investigated.

The **residuals** are a means to test if the model fits the data well. They are defined as the difference between the observation and the fitted value (Teetor and Loukide 2011).

If the residuals have a normal distribution, it suggests that the model fits the data well.

The positive median sign of the below stated residuals indicates a skew to the right.

The magnitude of 0.042 indicates a low extent, meaning the majority of the residuals are around zero.

The first quartile (1Q) and third quartile (3Q) have about the same magnitude. The first quartile has a magnitude of 0.649, which is only slightly larger then the third quartiles magnitude of 0.626. Considering the residual standard error of 1.422, the distribution can still be considered as a normal distribution, even though the median is not exactly zero. The minimum and maximum values show outliers. The maximum value of 77.691 is a high outlier and will be discussed in Chapter 8.

Residuals:

Min	1Q	Median	3Q	Max
-7.842	-0.649	0.042	0.626	77.691

Residual standard error: 1.422 on 161185 degrees of freedom

To explore the **outliers** is another way to test how well the model fits the data. Hawkins (1980) defines an outlier as an observation which differs so much from the other observations as to assume that it was generated by a different mechanism. The ten rows below show the ten most extreme observations:

ID	rstudent	unadjusted p-value	Bonferonni p
18815	55.52978	0.0000e+00	0.0000e+00
31306	29.68524	3.9645e-193	6.3904e-188
80871	27.80937	8.4454e-170	1.3613e-164
72155	22.37970	9.1652e-111	1.4773e-105
31337	20.97268	1.5741e-97	2.5373e-92
147578	19.79797	3.9368e-87	6.3457e-82
42986	19.60107	1.9051e-85	3.0709e-80
43157	19.10295	2.9361e-81	4.7326e-76
81735	17.70312	4.6233e-70	7.4523e-65
4184	17.53552	8.8929e-69	1.4334e-63

To get a better understanding of them, the first five of the above listed outliers are further explored. The dependent variables and the independent variables are listed:

ID	TPA	VPT	S	SD	C
18815	313.7197	0.2207899	0	195.9101	470.6974
31306	153.6567	18.98797	81.10564	883.7568	85.4359
80871	465.7265	6.923552	67.1984	2492.431	106.9077
72155	290.1743	8.03457	71.63274	1517.892	90.3641
31337	97.83661	13.03708	80.70007	516.5303	75.36923

Investigating the input variables and the resulting *Cost* variable shows two outlier patterns.

The first pattern is the following: An extremely high *Cost* is created when the stand is extremely densely stocked (high *Trees per Acre* value), while at the same time the volume of the trees is very low (low *Volume per Tree* value). This pattern matches the outlier with the ID 18815.

The second pattern is the following: High *Costs* arise when the unit has a high *Slope* value. Though the *Costs* are relatively not as high as with the first pattern. The second pattern matches the other four outliers.

Both patterns will be further discussed in Chapter 8.

Linearity of data

The last point that needs to be checked is whether the data satisfies the assumptions behind a linear regression. The linearity was already investigated in Chapter 4.2.1.1. Now that the model is created, this will be further investigated.

To get a better understanding for that problem, the residuals are plotted below. The plots show few high outliers, but also shows that the majority of the values are located around the zero line, which suggests that the assumption that the data is linear is

reasonable. Also the residuals roughly form a "horizontal band" around the 0 line. This suggests that the variances of the error terms are equal (Teetor and Loukide 2011).

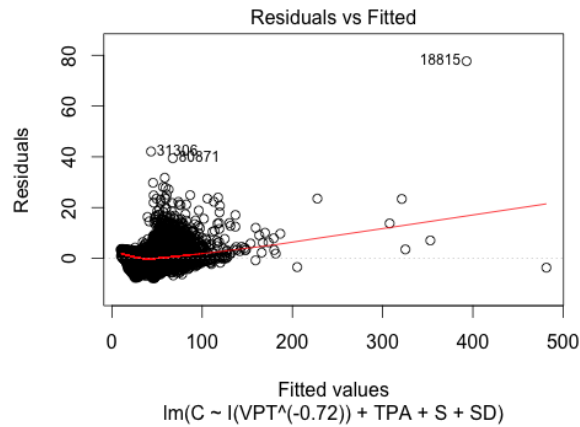


Figure 37: Regression Model Residuals vs Fitted Plot

Next the Normal Q-Q plot will be explored. The points in the Normal Q-Q plot are roughly on the line, indicating that the residuals follow a normal distribution. The graph seems to follow a normal distribution reasonably well, except in the extreme tails. It is a curved pattern with slope increasing from left to right, which indicates the data distribution is skewed to the right (Teetor and Loukide 2011). The individual points on the far end are the above-investigated outliers.

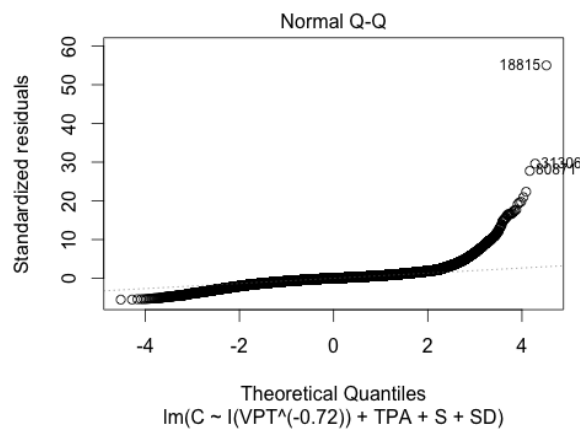


Figure 38: Regression Model Normal Q-Q Plot

The last two plots that are explored are the Residuals vs Leverage plot and the Scale-Location plot. They are explored in order to prove that the data fits the assumption of a linear regression (Teetor and Loukide 2011). In both plots, the majority of the points are in a group with none too far from the center. Some outliers are visible and were already discussed above.

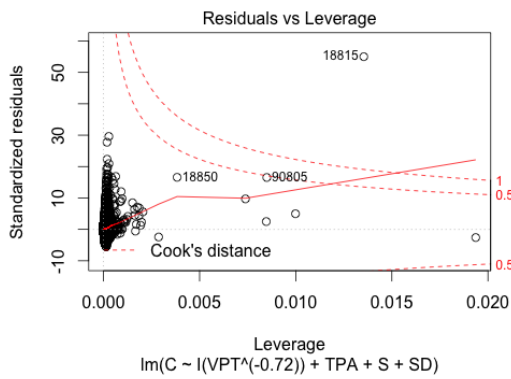


Figure 39: Regression Model Residuals vs Leverage Plot

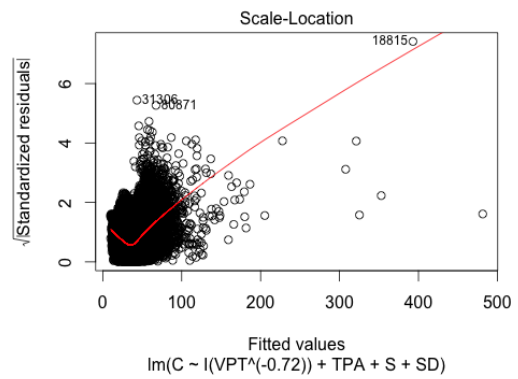


Figure 40: Regression Model Scale Location Plot

After investigating the four plots, it can be said that the data satisfies the assumption of a linear model. Outliers do exist. They will be discussed in Chapter 8.

4.3. Spatial Regression Model

From the developed multiple linear regression model in Chapter 4.2, a multiple-linear-spatially explicit regression model was derived by using a backward selection. The non-spatial tree variables *Trees per Acre* and *Volume per Tree* were removed from the original model. In Chapter 4.3.1 the new model is fitted and thereafter validated in Chapter 4.3.2

4.3.1. Fitting the Model

The regression respecting only the two independent spatial variables as a linear relationship with *Cost*, is the following two-variable-regression:

```
m <-lm(formula = C ~ S + SD, data = costData)
```

With the ordinary least-square algorithm (OLS) the following linear model is fitted:

$$C = 22.8038 + 0.3272 \times S + 0.007578 \times SD + \varepsilon_i$$

Equation 4: Spatially explicit Regression Model

Possible correlation between the two variables, as well as Nonlinearity, has already been tested in Chapter 4.2.1.1. The two independent variables have a linear correlation to *Cost*. Adding an interaction term between the independent variable *Slope* and *Skidding Distance*, adds no further explanatory value to the model.

4.3.2. Validating the Model

Also, the spatially explicit model needs to be validated. The same indicators as used for the original model are used to test if the spatially explicit model is reasonable and if it is statistically significant.

Statistical significance of the model

The spatially explicit model has a p-value of smaller than $< 2.2e-16$, which indicates that the model is likely significant.

F-statistic: 5.475e+04 on 2 and 161187 DF, p-value: $< 2.2e-16$

Significance of the coefficients

All p-values of the two coefficients listed below are under the significance level of $\alpha = 0.05$ ($< 2e-16$). Therefore it is very likely that all the coefficients are significant. The column t value shows in addition the t-statistic from which the p-value was calculated.

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	2.280e+01	4.950e-02	460.7	$< 2e-16$	***
S	3.272e-01	1.979e-03	165.3	$< 2e-16$	***
SD	7.578e-03	2.655e-05	285.4	$< 2e-16$	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Usefulness of the model

The spatially explicit model explains 0.4045 (40.45%) of the variance of *Cost*. The remaining 59.55% are unexplained.

The two predictors and the high sample size of 161187 results in a narrow 95% confidence interval of 0.39981 to 0.40919.

Multiple R-squared: 0.4045, Adjusted R-squared: 0.4045

Fitness of the model to the data

Also for the spatially explicit model the residuals and outliers are explored to understand how well the model fits the data.

The negative median sign of the below stated residuals indicates a skew to the left. The magnitude is 1.19, meaning that not necessary all residuals are around 0.

The first quartile (1Q) and third quartile (3Q) have roughly the same magnitude. The first quartiles magnitude of 4.94 is larger then the third quartiles magnitude of 3.42. Considering the residual standard error of 8.379, the distribution can still be considered as a normal distribution, even though the median is not zero. This will be further discussed in Chapter 8. The minimum value of -24.01 and the maximum value of 446.41 are high outliers and will also be discussed in Chapter 8.

Residuals:

Min	1Q	Median	3Q	Max
-24.01	-4.94	-1.19	3.42	446.41

Next, the **outliers** are explored to get a better understanding how well the spatially explicit model fits the data. The ten rows below show the ten most extreme observations:

	rstudent	unadjusted p-value	Bonferonni p
4324	39.03909	0.0000e+00	0.0000e+00
18815	53.75025	0.0000e+00	0.0000e+00
115797	52.47481	0.0000e+00	0.0000e+00
90805	38.07370	8.9501e-316	1.4427e-310
129763	35.26860	1.9448e-271	3.1349e-266
6350	34.37709	4.8062e-258	7.7471e-253
18850	26.30496	3.5366e-152	5.7006e-147
100984	20.06558	1.8991e-89	3.0612e-84
27331	18.07054	6.4213e-73	1.0350e-67
65002	17.65406	1.1018e-69	1.7760e-64

To get a better understanding of them, the first five of the above listed outliers are further explored. The dependent variables and the independent variables are listed:

ID	TPA	VPT	S	SD	C
4324	399.0791	0.265897	12.86954	960.2507	359.8821
18815	313.7197	0.2207899	0	195.9101	470.6974
115797	262.62	0.1754692	22.15006	1550.183	477.8
90805	417.0828	0.294277	7.28869	235.7349	344.5846
129763	279.9218	0.2988943	15.10381	867.5535	328.7179

Investigating the input variables and the resulting *Cost* variable shows the same outlier pattern observed in the regression model with all four variables.

The pattern is, that an extremely high *Cost* is created when the stand is extremely densely stoked (high *Trees per Acre* value), while at the same time the volume of the trees is very low (low *Volume per Tree* value). This pattern will be further discussed in Chapter 8.

Linearity of data

As in the regression model with all four predictors, the residuals plotted below of the spatially explicit model, show a couple of high outliers, but the majority of the values are located around the 0 line. This suggests that the assumption that the relationship is linear is reasonable. As in the original model the residuals form a "horizontal band" around the 0 line. It suggests also that the variances of the error terms are equal (Teetor and Loukide 2011).

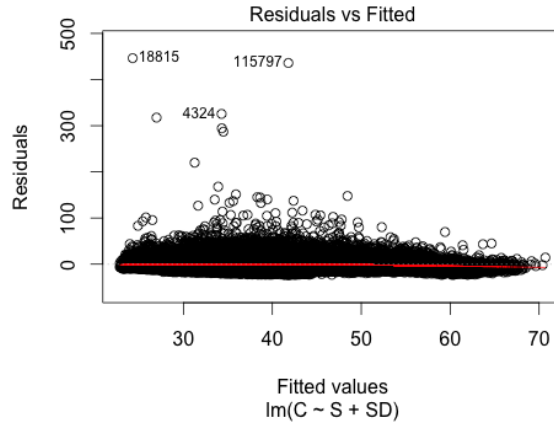


Figure 41: Spatial Regression Model Residuals vs Fitted Plot

The same applies for the Normal Q-Q plot. As in the original model, the points in the Normal Q-Q plot are more-or-less on the line, indicating that the residuals follow a normal distribution (Teetor and Loukide 2011). The graph seems to follow a normal distribution reasonably well, except in the extreme tails. It is a curved pattern and is skewed to the right. The individual points on the far end are also outliers.

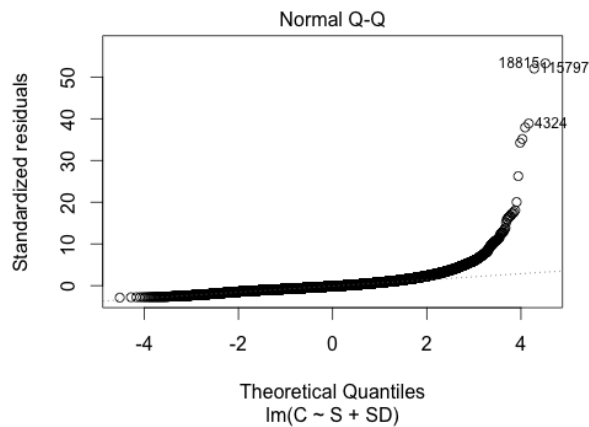


Figure 42: Spatial Regression Model Normal Q-Q Plot

The Scale–Location plot and the Residuals vs Leverage plot are similar to the plot from the four-variable regression model. Again the majority of the points are in a group, with none too far from the center. As in the four-variable regression model, some outliers are visible and were already discussed above.

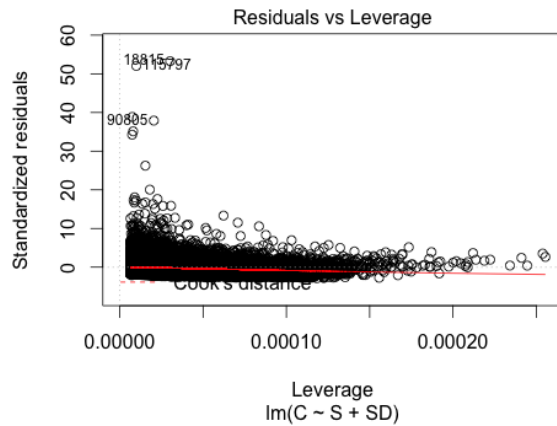


Figure 43: Spatial Regression Model Residuals vs Leverage Plot

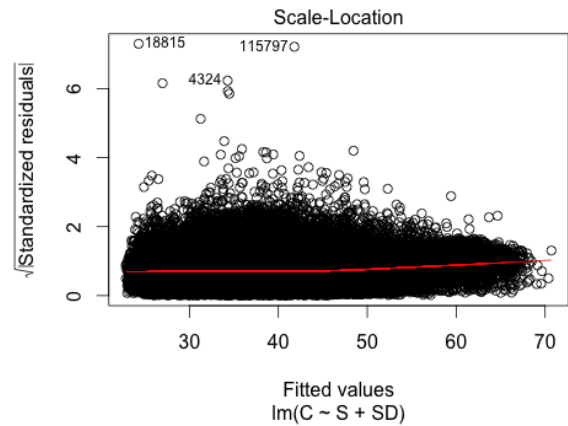


Figure 44: Spatial Regression Model Scale-Location Plot

After investigating the four plots, it can be said that the data satisfies the assumption of a linear model. Outliers do exist. They will be discussed in Chapter 8.

4.4. Comparing the Two Models

The Analysis of variance (ANOVA) function is used to compare the two models. The ANOVA analysis performs an F test between the two models.

Model 1: $C \sim I(VPT^{(-0.72)}) + TPA + S + SD$

Model 2: $C \sim S + SD$

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	161185	325961				
2	161187	11317766	-2	-10991806	2717681	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Comparing the two models yields a p-value of < 2.2e-16 (less than 0.05). This indicates that the two models are significantly different. A detailed discussion of the two models and a detailed comparison can be found in Chapter 8.

5. Cost Surface

Based on the developed spatially explicit linear cost model, a raster *Cost Surface* was generated. The *Cost Surface* describes, for any given location within the Colorado State Forest, the *Harvest Cost* per ton. First an overview of the data that goes into the *Cost Surface* will be given in Chapter 5.1. Next, the spatially explicit equation derived from the model will be discussed in Chapter 5.2. Last, in Chapter 5.3, the production of the *Cost Surface* itself will be described by explaining how the data and the equation are combined.

5.1. Data Overview

Two datasets are crucial for creating the *Cost Surface*. One is a slope raster, and the other a dataset containing the existing roads covering the area of interest.

Slope Dataset

As mentioned in the Data Creation Chapter (Chapter 3), the National Elevation Dataset, a digital elevation raster, is available as a 10 m raster from the U.S. Geological Survey (Gesch 2007). The file is in the projection NAD83/ UTM zone 13N (EPSG:26913). A slope raster in percentages is derived from the digital elevation raster. The raster will be used to derive the values for *Slope*.

Road Dataset

Knowing the location of the roads in the CSF will be necessary in order to calculate the *Skidding Distance*. Roads covering the entire state of Colorado are available from the OSM database via Geofabrik GmbH (2013) as a shapefile. The file is in the projection WGS 84 (EPSG:4326).

5.2. Spatially explicit Cost Equation

From the regression analysis in Chapter 4.3 there follows the following spatially explicit equation:

$$C = 22.8038 + 0.3272 \times S + 0.007578$$

Equation 5: Spatially explicit Regression Model

The two independent variables of the regression are *Slope*, with a coefficient of 0.3272 \$/%, and *Skidding Distance* with a coefficient of 0.007578 \$/ft. The intercept is 22.8038 \$/ton. *Slope* will simply be derived from the slope raster. *Skidding Distance* will be calculated based on the road dataset, with a function explained in the next chapter. The equation results in a *Harvest Cost per ton* in \$.

5.3. Cost Surface

In the following, the process of creating a raster *Cost Surface* is explained. The detailed script of the calculation can be found in Appendix IV (script CostRaster.py). The creation of the *Cost Surface* is done in NumPy arrays (Bressert 2012). Arrays enable calculations for Big Data at a reasonable rate.

Four steps are necessary to create the *Cost Surface*: For the two independent variables, *Slope* and *Skidding Distance*, two separate arrays are created. The *Skidding Distance* array contains the values for the variable *Skidding Distance* (Step 1). The *Slope* array contains the values for the variable *Slope* (Step 2). After the arrays are created, they are placed into the equation, which results in a *Cost* array (Step 3). Last, the *Cost* array is written to a raster file (Step 4). A small sample extract of the State Forest is used to illustrate the process below.

1. *Skidding Distance* Array

The first step in creating a *Skidding Distance* array is to convert the OSM vector roads to raster.

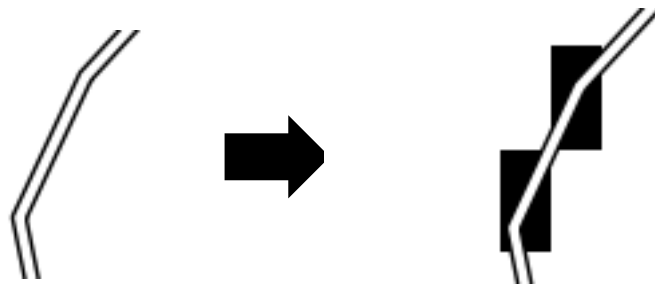


Figure 45: Schema rasterize roads

Next, the created raster is read as an array with the *raster2array* function and results in the array below. The value 0 represents the existence of a road, and the NoData value of 255 represents no existing road at the location.

```
[ [255 255 255 255 0]
  [255 255 255 255 0]
  [255 255 255 0 255]
  [255 255 255 0 255]]
```

The *shp2array* function of the script performs the two described steps above.

Next the function *roadArray2coordDict* converts the array to a dictionary containing the coordinates of all array values of zero. This means the dictionary contains the coordinates of the pixels' upper-left corner where a road exists.

```
[ (417615.43606907444, 4489053.83774992), (417615.43606907444,
4489043.838467314), (417605.4369691394, 4489033.839184708),
(417605.4369691394, 4489023.839902102) ]
```

Last, the function *nonRoadArray2coord* returns the final *Skidding Distance* array. Each array value represents the distance to the closest road segment. The function calculates the Euclidian distance for each pixel in the original slope raster, which is the reference raster, to each coordinate pair in the above dictionary. The shortest distance

results in the *Skidding Distance* for each array element. The distances are calculated in meters and converted to feet. For the array below, 0 feet is where a road is located.

```
[ [ 118.28238627  92.78866378  65.61089406  32.80544703  0.          ]
  [ 103.7401218   73.35547762  46.39433189  32.80544703  0.          ]
  [  98.41634109  65.61089406  32.80544703  0.          32.80544703]
  [  98.41634109  65.61089406  32.80544703  0.          32.80544703]]
```

2. Slope Array

Since the *Cost Surface* raster will have the same extent and pixel size as the *Slope* raster, the *Slope* raster can simply be read as an array. This is done with the *raster2array* function.

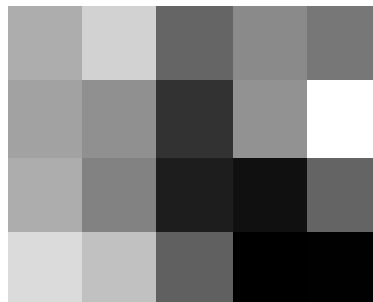


Figure 46: Slope sample

For the sample raster the following array results:

```
[ [ 30.87272263  33.02461243  26.57536507  28.77716064  27.6699295 ]
  [ 30.2076149   29.15476036  23.58495331  29.26174927  36.44345093]
  [ 30.87272263  28.33946037  22.36067963  21.57834625  26.51650429]
  [ 33.58757401  32.01562119  26.2797451   20.61552811  20.69118118]]
```

3. Cost Array

The two created arrays, the *Slope* array and the *Skidding Distance* array, are put into the *Cost* equation from Chapter 5.2:

$$CArray = 22.8038 + SArray \times 0.3272 + SDArray \times 0.007578$$

This results in the final *Cost* array. Each array element represents a *Harvest Cost per ton* for the pixel in the *Cost Surface*:

```
[ [ 33.80170128  34.31260379  31.9964593   32.46828809  31.8574028 ]
  [ 33.47387534  32.89912664  30.87237398  32.62684217  34.72809601]
  [ 33.65115639  32.57367259  30.36881414  29.86423492  31.72859922]
  [ 34.53945443  33.77651149  31.65113416  29.54920197  29.82255521]]
```

4. Cost Raster

Last, the *Cost Array* from Step 3 is written to a raster file by using the function *array2raster*, which results in the final *Cost Surface*. The *Cost* for the sample raster ranges from between 29.55 \$/ton to 34.47 \$/ton. The color range from green to red symbolizes the price range. Red colors symbolize more expensive *Harvest Costs*, while green colors symbolize less expensive *Harvest Costs*.

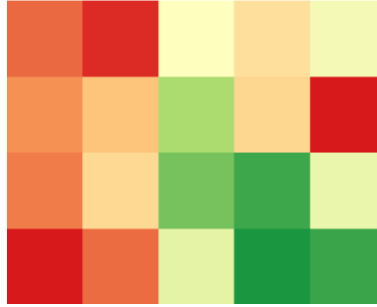


Figure 47: Cost Surface sample

6. Results

After the data was created (Chapter 3) and statistically analyzed (Chapter 4), and based on the analysis a *Cost Surface* was created (Chapter 5), the following results can be presented. First, the statistical results are discussed. This includes the influence of each variable on timber *Harvest Costs* as well as the two developed regression models. Second, the specific *Harvest Costs* for the Colorado State Forest are shown. Third, the created *Cost Surface* based on the spatially explicit regression is shown.

6.1. Statistical Results

Three things result from the statistical analysis: (1) a better understanding of the influence of the input variables on the *Harvest Cost*, (2) the cost equation which enables calculation of the *Harvest Cost* based on all four predictors and (3) the cost equation which enables calculation of the *Harvest Cost* solely based on the spatial predictors.

Input Variables

The generated input variables are all normally distributed except for the variable *Volume per Tree* as outlined in Chapter 4.1.1. Only the two tree variables *Trees per Acre* and *Volume per Tree* have a relevant correlation. The other variables have no significant correlation.

Overall the four variables explain 98.28% of the *Harvest Cost* using the given model.

All of them have an importance in predicting *Cost*. The *Trees per Acre* variable, with 3.2% explanatory value, has the least importance, followed by *Slope* with 9.5%. *Skidding Distance* is the most important spatial variable with 29.9% explanatory value. And *Volume per Tree* is the most important variable out of the four, with 55.7% explanatory value. The two spatial variables taken together explain 39.4% of the *Cost*. Below is again overview of the relative importance of each predictor on *Cost*.

```
Proportion of variance explained by model: 98.28%
Metrics are not normalized (rela=FALSE).
```

```
Relative importance metrics:
```

```
                                lmg
I (VPT^(-0.72)) 0.55669591
TPA              0.03239262
S               0.09457912
SD              0.29918160
```

Cost equation based on all four variables

The created equation was validated in Chapter 4.2.2. The model and its coefficients are both statistically significant. The model is useful since it has a multiple R-squared

of 0.9828. With the exception of some outliers, the model fits the data well and satisfies the assumptions of a linear regression. The outliers are created when the stands are extremely densely stocked (high *Trees per Acre* values), with a low timber volume (low *Volume per Tree* value), or when the *Slope* variable exceeds the operable slope of ground based harvesting machines of 40%.

The linear equation with the all four variables is the following:

$$C = -6.418 + 129.5 \times VPT^{(-0.72)} + -0.003348 \times TPA + 0.3089 \times S + 0.007537 \times SD + \varepsilon_i$$

Equation 6: Final Regression Model with all Predictors

The individual coefficients of the equation are listed below:

Intercept	-6.418326223
VPT ^(-0.72)	129.509646540
TPA	-0.003347710
S	0.308924486
SD	0.007536668

Cost equation based solely on the spatial variables

From the original equation with all the predictors, a spatially explicit equation with only the spatial predictors was derived. The model was validated in Chapter 4.3.2. It is statistically significant and its coefficients are also statistically significant. The model is useful since it has a multiple R-squared of 0.4212. The degree to which it is useful, will be discussed in Chapter 8. The spatially explicit equation has a similar outlier pattern as the original model. The outliers are created when the stands are extremely densely stocked (high *Trees per Acre* values), with a low timber volume (low *Volume per Tree* value). Also the model fits the data well and satisfies the assumptions of a linear regression.

The linear equation with the two variables is the following:

$$C = 22.83 + 0.3306 \times S + 0.007526 \times SD + \varepsilon_i$$

Equation 7: Final spatially explicit Regression Model

The individual coefficients of the equation are listed below:

Intercept	22.827810392
S	0.330640364
SD	0.007526195

6.2. Colorado State Forest Harvest Costs Results

The Table below gives an exemplary overview comparison of the calculated *Harvest Cost* for 10 stands of the Colorado State Forest. The full Table can be found in Appendix V. The Table shows for each stand the input variables and in addition the calculated *Harvest Costs*. It shows the *Harvest Costs* from the full model (Chapter 2), from the full cost equation, and from the spatially explicit cost equation.

ID	TPA	VPT	S	SD	Full Model	Full Equation	Spatial Equation
1	460.43	6.00	19.57	982.11	47.37	45.17	36.69
2	249.05	10.02	8.11	1273.00	34.59	33.15	35.09
3	134.77	13.98	15.24	617.87	26.49	25.21	32.52
4	465.84	4.04	26.99	267.52	55.49	53.95	33.77
5	156.87	21.79	35.38	341.41	21.60	23.50	37.09
6	210.14	13.79	20.11	324.26	24.27	24.45	31.92
7	357.85	8.00	12.23	1658.22	42.66	41.48	39.35
8	219.54	13.96	1.48	442.08	22.34	19.37	26.65
9	367.15	9.04	16.08	2039.09	44.27	42.96	43.49
10	328.22	6.07	23.11	396.65	42.52	41.99	33.45

Table 3: Harvest Costs CSF

In addition, the three calculated *Harvest Costs* are displayed for the ten stands of the CSF in the bar chart below, in order to compare them:

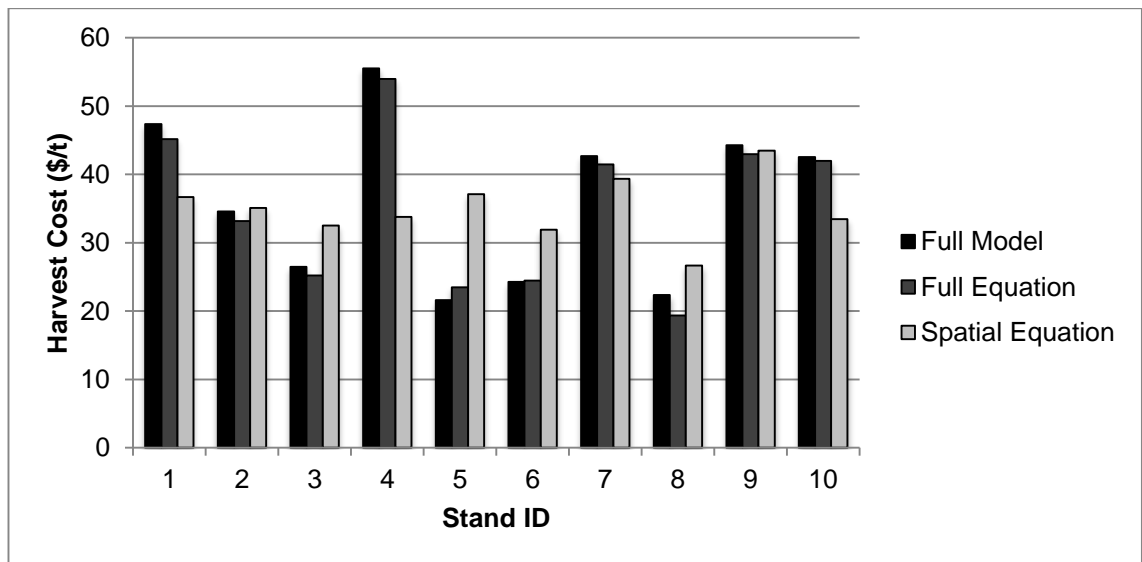


Figure 48: Sample Harvest Cost CSF Comparison

The mean *Harvest Cost* of all stands is 36.38 \$/ton from the full model, 35.62 \$/ton from the full equation, and 39.3 \$/ton from the spatially explicit equation. In addition, Table 4 below gives an overview of the minimum and maximum values of the calculations and of the standard deviations. The full model and the full equation have roughly the same standard deviations, while the spatially explicit equation's standard deviation is significantly lower.

	Full model	Full equation	Spatial equation
MEAN	36.3799	35.6203	39.2976
MIN	16.6285	15.3259	26.2431
MAX	61.5810	60.7428	63.9345
STDEV	10.2949	10.1817	8.6541

Table 4: Harvest Cost Statistics CSF

6.3. Cost Surface Results

Based on the spatially explicit equation produced in Chapter 4.3 a raster *Cost Surface* was produced in Chapter 5. The image below shows the *Cost Surface*. It shows for each pixel (10m x 10m); the potential *Harvest Cost* for the southern part of the Colorado State Forest. The mean *Harvest Cost* is 46.14 \$/ton with a standard deviation of 17.51 \$ /ton. The lowest *Harvest Cost* is 23.38 \$/ton and the maximum *Harvest Cost* is 16816 \$ /ton. The color range from green to red symbolizes the price range. The colors range from green (< 25 \$/ton), to light green (< 40 \$/ton), to orange (< 50 \$/ton), to red (> 50.01 \$/ton). The black lines are roads covering the State Forest.

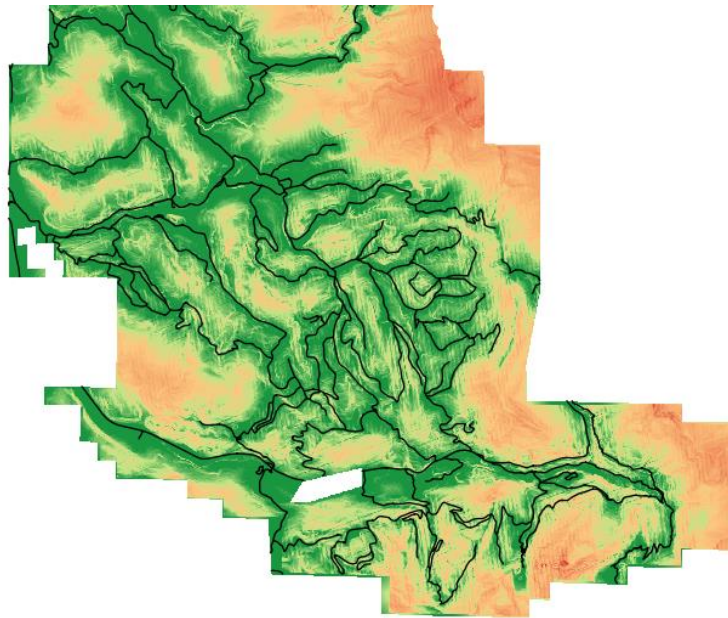


Figure 49: Cost Surface Southern Part of CSF

7. Application Example

The research and the results of this paper were put into practice by developing a forest management web application. It can be accessed via the URL www.wald.io.

First the need of the application will be explained in Chapter 7.1. Followed by Chapter 7.2 where the application itself will be described.

7.1. Need of Application

As outlined in the introduction of this paper, being able to calculate *Timber Harvest Costs* solely with spatial variables brings various advantages. One of the main points is that it helps forest planners to better manage their forests by having the ability to see the potential *Harvest Costs* for an entire area. The application realizes that by providing mainly two functionalities:

1. It visualizes the *Cost Surface* of the CSF (produced in Chapter 5) as a web map. The *Cost Surface* is based on the regression model with the spatial variables. In this way the forester gets a complete overview of the forest's potential *Harvest Cost*. Also the application allows the user to digitize a forest stand and then reports the potential *Harvest Costs* for the digitized stand. The user gets immediate feedback since the *Cost Surface* is pre-generated.
2. In addition, if the user has inventory data of the stand, they can enter these information and the application returns the exact *Harvest Cost* based on the full harvest model.

Besides supporting the user in better decision making by providing crucial *Harvest Cost* information, this application is also a tool to evaluate the quality of this research, since the user can compare the estimated cost (based on the *Cost Surface* and the regression) and the detailed calculated cost (based on the full harvest model).

7.2. Application Description

The application consists of a Front-End and Back-End part. The Front-End shows the map interface as a single web page and is described in Chapter 7.2.1. The Back-End application consists of (1) a Python Flask web application framework, which communicates with the Front-End and runs the Python scripts to calculate the *Harvest Costs*. In addition it consists of (2) a GeoServer, which serves the data as OGC Web Map Services (WMS) and OGC Web Feature Services (WFS). The Back-End application is described in Chapter 7.2.2. The chart schema below gives an overview of the Front-End and Back-End application with its various components.

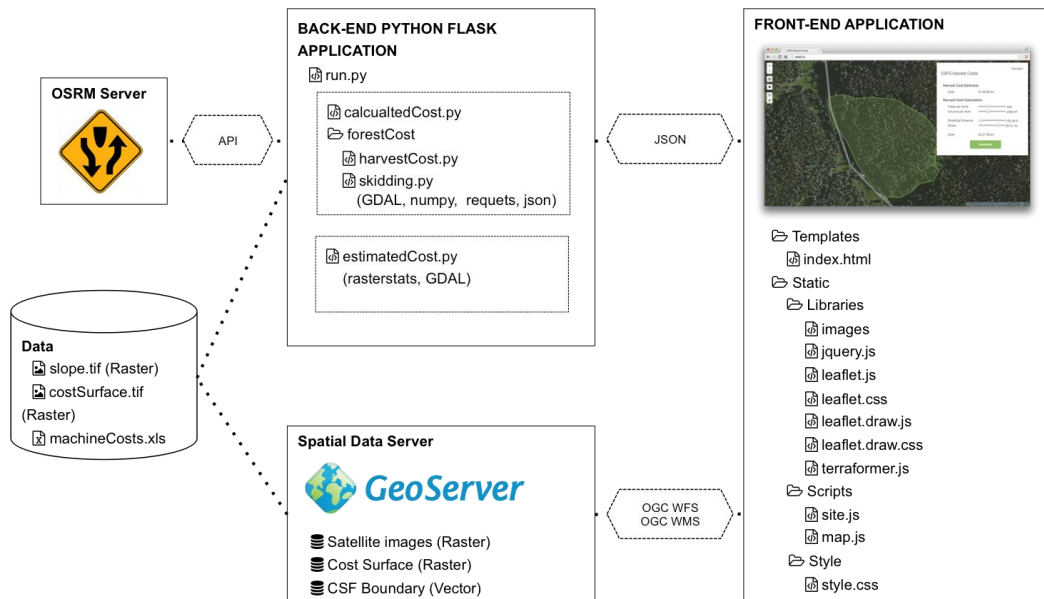


Figure 50: Schema Application

7.2.1. Application Front-End

The Application Front-End is laid out in the following. First the user interface, including the visual appearance and the application workflow, is explained. This is followed by the technical specifications of the application.

7.2.1.1. User Interface

The Front-End consists of a single web page. The individual elements of the page are described below, followed by Figure 51 which shows the Front-End user experience after a stand was digitized and all the cost information are shown in the info container. Last the workflow of the Front-End Application is explained.

Web Page Elements

Map

Besides basic map elements like zooming and panning, the map contains several layers:

- Two base maps are served as OGC Web Map Services (WMS). The base maps are a terrain layer (MapBox 2013) and a satellite layer (MapBox 2012). The two layers can be toggled with a layer switcher.
- In addition to the base maps the *Cost Surface* is served as WMS. The *Cost Surface* can be turned on and off.
- The boundary of the CSF (Colorado Parks and Wildlife 2013) is served as an OGC Web Feature Service (WFS) in GeoJSON format, which can also be turned on and off.
- A draw layer which saves the polygon the user digitizes, is part of the map as well.

Stand Digitizer

The site has a draw control which enables the user to draw, edit and delete a polygon.

Info Sign

The Info Sign informs the user to first digitize a timber stand. The sign disappears after the first stand is drawn.

Info Container

The Info Container informs the user about the application. The top contains the application name, contact information and the legend for the *Cost Surface*. After the user starts digitizing a stand it displays the calculation results and lets the user specify further settings.

Web Page Overview

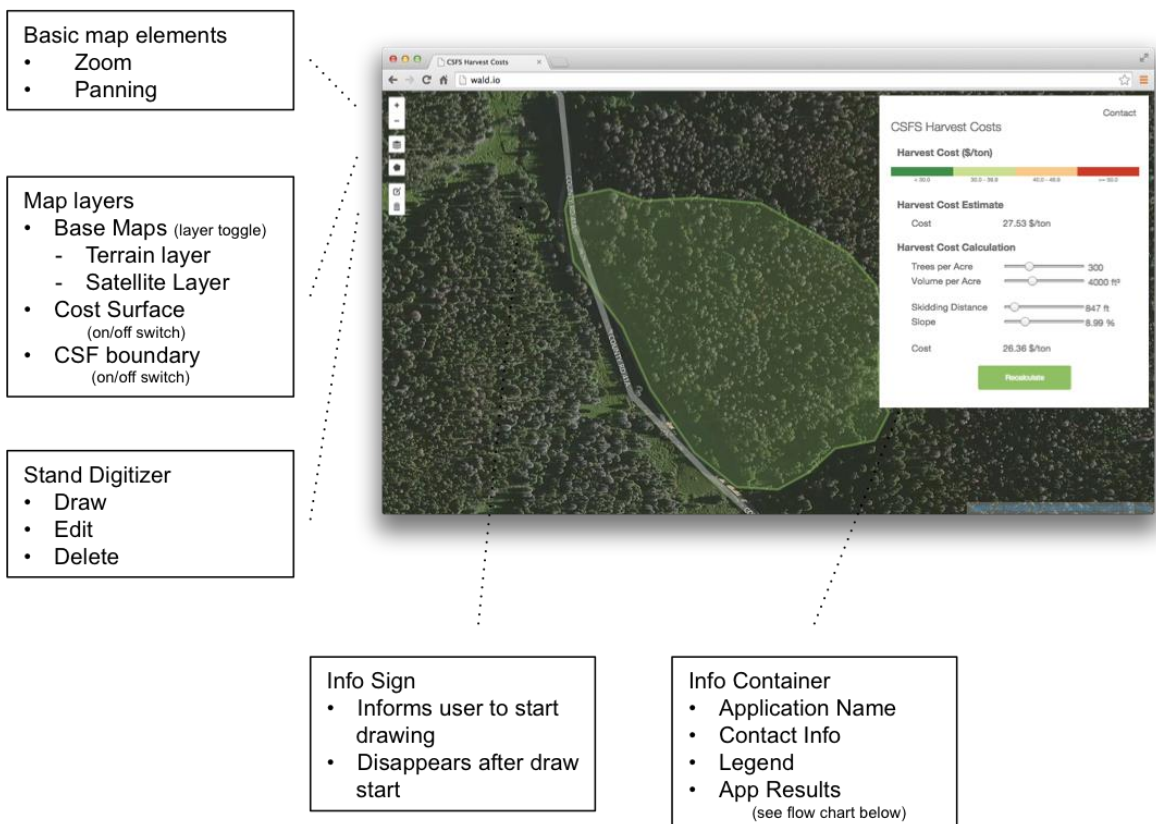


Figure 51: Front-End Overview

Web Page Workflow

In the following the workflow of the application is expounded. The boxes on the right side are the aforementioned Info Containers which show the results of the calculations to the user.

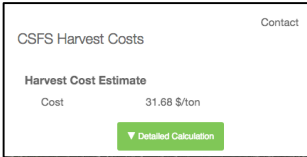
1. The user starts out by digitizing the stand.



2. A *Cost* estimate based on the generated *Cost Surface* is shown to the user.

The user can get a detailed calculation by clicking the “Detailed Calculation” button.

If the user edits the stand during any of these steps, they will always return to this screen.



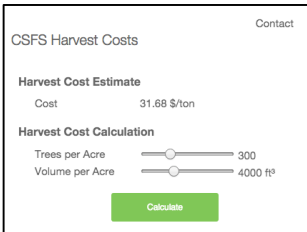
CSFS Harvest Costs Contact

Harvest Cost Estimate

Cost 31.68 \$/ton

[Detailed Calculation](#)

3. Next the user can specify the *Trees per Acre* and the *Volume per Acre* values, or leave the default values of 300 *Trees per Acre* with a *Volume* of 4000ft²/ac.



CSFS Harvest Costs Contact

Harvest Cost Estimate

Cost 31.68 \$/ton

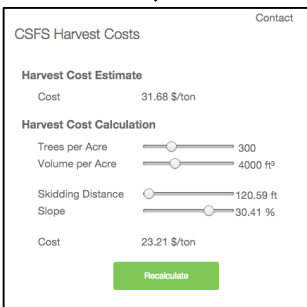
Harvest Cost Calculation

Trees per Acre 300

Volume per Acre 4000 ft³

[Calculate](#)

4. After clicking the “Calculate” button in the previous screen, the *Skidding Distance* to the closest road is determined by the application, as well as the *Slope*. Based on the four variables the actual *Harvest Cost* is calculated from the application and returned to the user.



CSFS Harvest Costs Contact

Harvest Cost Estimate

Cost 31.68 \$/ton

Harvest Cost Calculation

Trees per Acre 300

Volume per Acre 4000 ft³

Skidding Distance 120.59 ft

Slope 30.41 %

Cost 23.21 \$/ton

[Recalculate](#)

5. If the user thinks *Slope*, *Skidding Distance*, or the earlier specified tree data need adjustment; they can adjust the values and recalculate the *Harvest Cost* by clicking the “Recalculate” button.

The screenshot shows a web application titled "CSFS Harvest Costs" with a "Contact" link in the top right corner. The interface is divided into two main sections: "Harvest Cost Estimate" and "Harvest Cost Calculation".

Harvest Cost Estimate
Cost: 31.88 \$/ton

Harvest Cost Calculation

- Trees per Acre: 300
- Volume per Acre: 4000 ft³
- Skidding Distance: 1300 ft
- Slope: 30.41 %

Cost: 34.83 \$/ton

A green "Recalculate" button is located at the bottom of the form.

Figure 52: Front-End Workflow

7.2.1.2. Technical Specifications

The web application is shown by a single HTML page (Appendix VI index.html) with an associated style sheet (Appendix VI style.css).

The map element is created with the open source web mapping library Leaflet (Agafonkin 2014) and adapted with JavaScript (Appendix VI site.js). The Leaflet draw plugin is used to digitize the stand (Toye 2014). The Terraformer plugin (ESRI 2013) is used to convert the Leaflet GeoJSON data format to Well-known text (WKT) format in order to send it to the Back-End application.

JavaScript with jQuery (jQuery Foundation 2014) is used to make the dynamic web elements (e.g. disappearance of Info Sign) and to communicate with the Back-End Python Flask Application via AJAX requests.

7.2.2. Application Back-End

The Back-End of the application consists of two parts; the GeoServer and a Python Flask Application, which is the core part of the application. The Python Flask Application is explained in Chapter 7.2.2.1 and the GeoServer is explained in Chapter 7.2.2.2. The chart below illustrates again the major parts of the application.

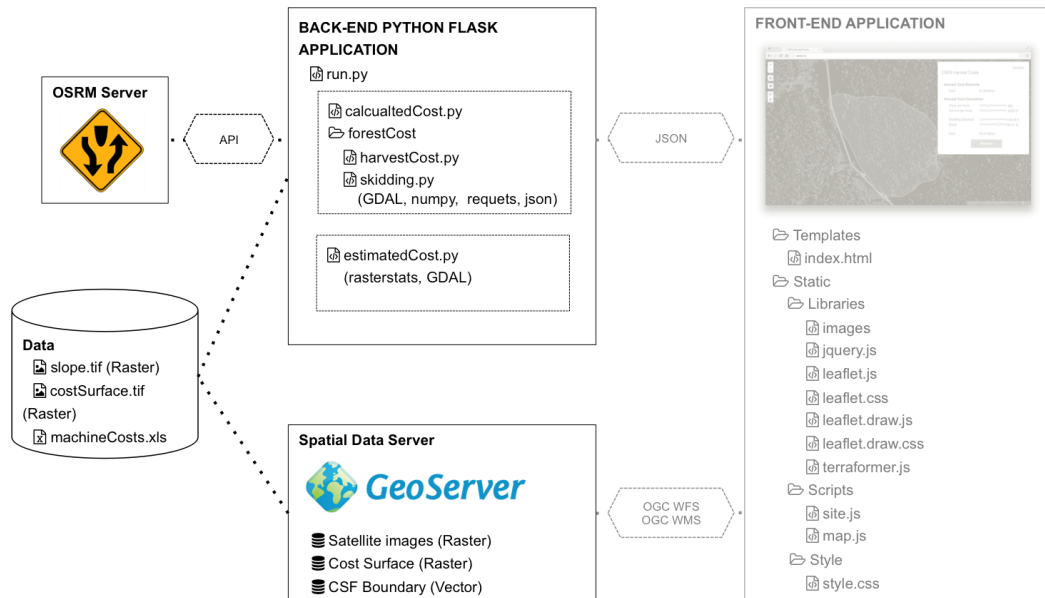


Figure 53: Back-End Overview

7.2.2.1. Flask Application

As a web framework Flask is used. It is a lightweight web application framework written in Python and based on the WSGI toolkit and Jinja2 template engine (Ronacher 2013). The run.py (Appendix VII) script communicates with the Front-End application. The script consists of three functions: (1) It serves the index.html file, (2) it calculates the cost estimate, and (3) it calculates the cost detailed based on the cost model. The latter two are explained in the following.

Estimated Cost

The application gets the stand coordinates in Well-known text (WKT) format in the projection WGS 84 (EPSG:4326) as a JSON string from the Front-End application. Next it calls from the script estimatedCost.py the function `get_zonal_stats` and passes the stand coordinates to the function. The function returns a *Harvest Cost* per ton estimate for the given stand. It does that by calculating the zonal statistics for the stand of the *Cost Surface*, by utilizing the Python library rasterstats (Perry 2014). The Flask application returns the cost estimate value as a JSON to the Front-End application.

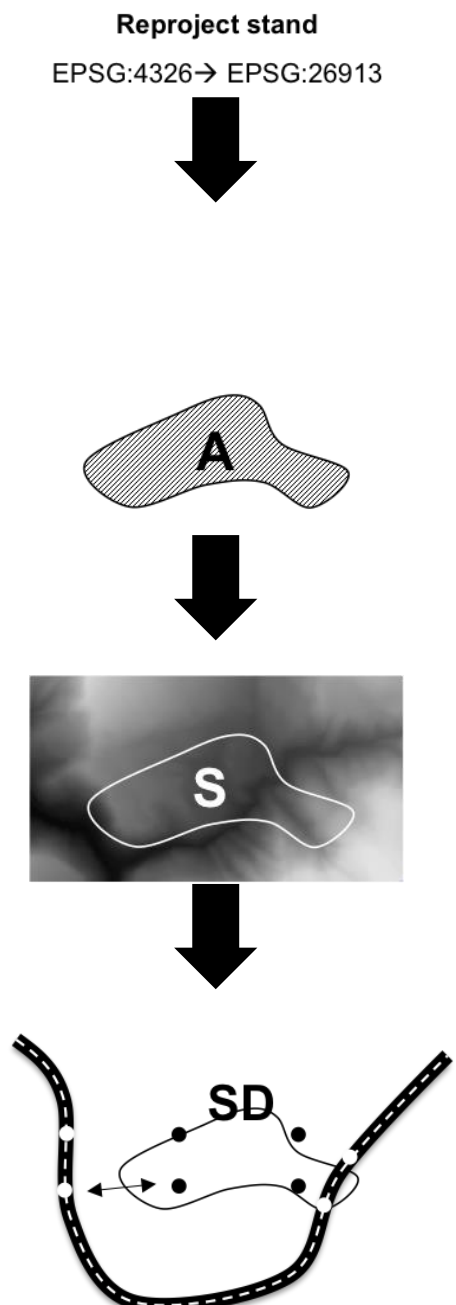
Calculated Cost

The application receives the stand coordinates in Well-known text (WKT) format in the projection WGS 84 (EPSG:4326), as well as the values for *Slope*, *Skidding Distance*,

Trees per Acre and *Volume per Acre* from the Front-End application. The values are also transmitted as a JSON string. After receiving the data, the `run.py` script calls from the `calculatedCost.py` script the function `cost_func`, which is explained below, and passes into that function all the above-mentioned values. The function returns *Slope*, *Skidding Distance*, *Harvest Cost per ton* and *Total Harvest Cost*. These four variables are then again returned to the Front-End application as a JSON string.

The function `cost_func` goes through the following steps:

1. First it reprojects the stand geometry from WGS 84 (EPSG:4326) to NAD83/ UTM zone 13N (EPSG:26913) by using the OSR package from the GDAL library (Rouault 2011). The Front-End application uses the projection WGS 84. In order to do various calculations (e.g. distance calculation) a projection that fits the study area and uses meters as units needs to be used.
2. The area covering the stand is calculated with the OGR package of GDAL.
3. If the *Slope* was not specified from the Front-End application, the script calculates the zonal statistics of the *Slope* for the stand with the `rasterstats` library.
4. Also if the *Skidding Distance* was not specified from the Front-End application, the script calls the function `skidDist` from the `skidding.py` script, which takes in the stand geometry and returns the Skidding Distance. See the flow chart below to see how the



function works.

- Next the Total Volume of the stand is calculated. The calculation is based on the *Volume per Acre* and the Total Acreage. Also the Total Weight is calculated. This is based on the Total Volume and the weight for lodgepole pine (Engineering Toolbox 2014).
- The function *harvestcost* is called from the script *harvesting.py*. It does exactly what is described in the model development section in Chapter 2. It takes in the four arguments *Slope*, *Skidding Distance*, *Volume per Acre*, and *Trees per Acre*. After processing it returns a *Harvest Cost* per ton.
- Finally the script calculates the *Total Harvest Cost* and returns to the Front-End application the following calculated variables: *Slope*, *Skidding Distance*, *Harvest Cost* per ton, and the *Total Harvest Cost*.



total V = VPA x A
weight = total V x 39 lb/ft3



harvestcost(VPA, TPA,SD, S)



return S, SD, Harvest Cost

Figure 54: Schema cost_func function

The *skidDist* function is similar to the function used in the Data Creation Chapter (3), though this process is more accurate, but also more time consuming, since more API calls have to be made. The individual steps of the function are explained below:

- The first part is the same as described in the Data Creation Chapter, but here explained again for the sake of completeness. To imitate the common practice of creating several landings per stand, the stands are divided into four parts. To accomplish this the function *createQuaterCentroids* is applied. First a bounding box around the stand is created. Second the bounding box is split half way north-south and halfway east-west. Third, the four created squares are intersected with the stand, which divides the stand into four parts. Fourth, for each of the four parts of the stand, the centroid is calculated.

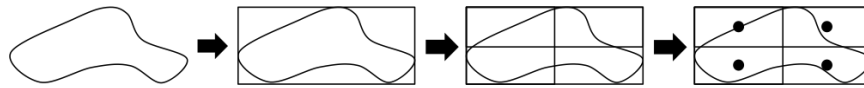


Figure 55: Schema createQuarterCentroids function

2. The second part differs already from the process of the Data Creation Chapter. For each of the four points, the function *landing* is called. Landing reprojects the points from NAD83/ UTM zone 13N (EPSG:26913) to WGS 84 (EPSG:4326). This is necessary to call in the next step the OSRM API nearest function (Luxen 2014). The API call returns the nearest point on an OpenStreetMap street segment. For each centroid an individual call is made. Based on these four calls, four landings on an existing street are created.

Instead of making an API call as described here, the process described in Chapter 3 downloads the surround road segments in order to determine the closest landing. By downloading the road segment, landings could only be created on vertices of the OSM road, while by making an API call to the OSRM server, landings can be created anywhere on the road segment. Though this process requires four API calls per stand, while the process in Chapter 3 only requires one API call. Since this process is only done for one stand, this is justifiable, while the other process has to do it for 160,000 sub units.

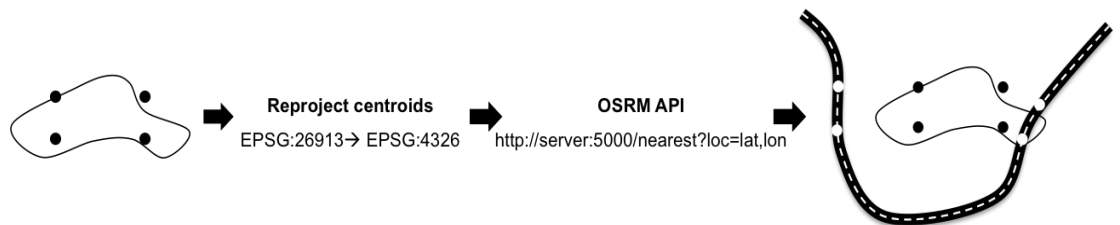


Figure 56: Schema landing function

3. Next the stand and the landings are converted to a raster. For each raster cell in the stand, the Euclidean distance to each of the four landings is calculated. The shortest distance is stored in a list. The conversion to raster is necessary to do the distance calculations in an array, which brings great speed performance benefits. Finally the mean of that list is calculated which results in the final average *Skidding Distance* of that stand.

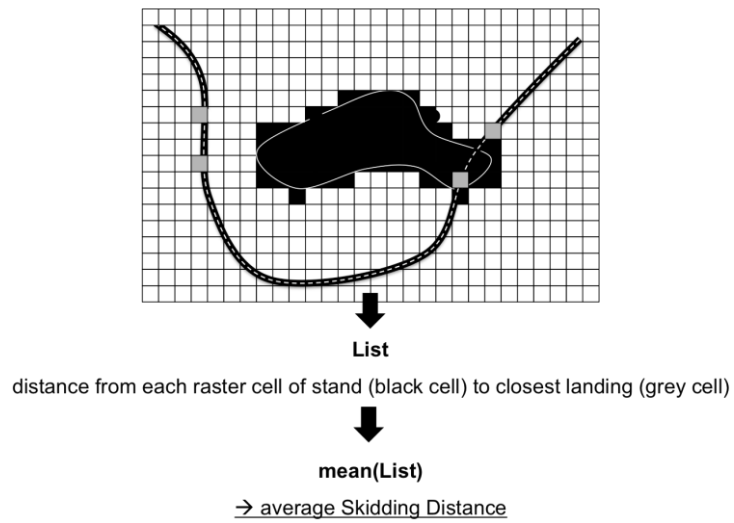


Figure 57: Schema Skidding Distance Calculation

7.2.2.2. GeoServer

Besides the Flask Application a GeoServer (GeoServer 2014) runs and stores three layers; the satellite imagery covering the state forest as a GeoTIFF, the Cost Surface produced in Chapter 5 as a GeoTIFF, and the CSF boundary as a Shapefile. The two raster layers are served as an OGC Web Map Services (WMS) and the CSF boundary is served as an OGC Web Features Service (WFS). These services are called from the Front-End Application via an AJAX request.

8. Discussion

In the following the results from the Statistical Analysis are discussed (Chapter 8.1). This is followed by the discussion of the *Harvest Costs* for the Colorado State Forest, which serves as a validation of this research (Chapter 8.2).

8.1. Discussion of the Statistical Analysis

The created multiple linear regression model explains 98.28% of the *Harvest Cost*. 1.72% of the *Harvest Cost* is unexplained. A possible reason for the unexplained *Cost* is that the input variables are not perfectly normally distributed, as outlined in Chapter 4.1.1. This causes outliers and will be discussed as below.

The spatially explicit regression model without the *Non-Spatial Predictors* explains 40.45% of the *Harvest Cost*. The remaining unexplained variance is explained by the tree variables, which are not part of this regression model. Outliers also occurred in this model, and are discussed below.

Outliers

Two outlier patterns were detected and stated in Chapter 4.2.2 and in Chapter 4.3.2. The first pattern, that high *Harvest Costs* occur in extremely densely stocked stands with a low volume of the trees, is a realistic real world condition and an expected silvicultural behavior. If stands are extremely densely stocked the *Volume per Tree* value is low. Therefore the outliers were not removed from the dataset. But from a practical point of view these stands are not likely to be harvested.

The second pattern is that high *Harvest Costs* arise in stands with a high *Slope* value. This is also an expected behavior; the steeper the slope, the more expensive it is to harvest. But in real world conditions slopes steeper than 40% are not harvested with ground based machinery. Yet the given stands of the CSF are located in this terrain and are assigned by the CSF as harvestable areas. Therefore these outliers were also not removed from the dataset.

Even though both patterns can occur in real world situations, it is likely that they caused the unexplained variance in *Cost*. Future studies should take into account the exclusion of stands that show these outlier patterns.

Interpretation Regression Models

The intercept of the regression with all predictors is -3.66 \$/ton. This means the basic *Cost* to harvest starts out with -3.66 \$/ton. The intercept of the spatially explicit regression is 22.80 \$/ton. This is the *Cost* if the stand is in an absolute flat ground (*Slope* of 0 %), and the stand is located at the road (*Skidding Distance* of 0 ft.).

The coefficient for the transformed variable, *Volume per Tree* (exponent of -0.72) is 133.51 \$/ft³ and for the variable *Trees per Acre* it is -0.0031 \$/tree. This means that if

the tree variables *Volume per Tree* or *Trees per Acre* increase, the *Cost* will decrease. For each additional harvested tree per acre the *Cost* will decrease by 0.3 cent/tree (0.0031 \$/tree). In the case of the variable *Volume per Tree* this is different. Since the variable is transformed with the exponent -0.72 the *Cost* for *Volume per Tree* will never be below 0 \$/ft³. Though the higher the variable *Volume per Tree* is, the closer the value gets to 0 \$/ft³. For instance, with a *Volume per Tree* of 1 ft³, the *Cost* would increase by 133.51 \$/ton. But with a *Volume per Tree* of 100 ft³, the *Cost* will increase by only 4.84 \$/ton. The higher the variable *Volume per Tree* is, the less expensive it is to harvest.

The following example illustrates this behavior. If 1 tree per acre with a volume of 1 ft³ is harvested on a flat ground (*Slope* of 0%) and harvested directly at the road (*Skidding Distance* of 0 ft.), the *Cost* to harvest this single tree will be 129.84 \$/t.

Another more realistic example based on the mean values of the input data is the following: To harvest 305.53 trees per acre with a *Volume per Tree* of 9.98 ft³ directly at the road with a flat ground costs 20.86 \$/ton.

In the regression with all predictors, the coefficient for *Slope* is 0.31 \$/% and for *Skidding Distance* it is 0.0076 \$/ft. The value for the coefficient for the variable *Skidding Distance* is in the spatially explicit regression the same. The value for the *Slope* coefficient is 0.33 \$/%. Since both coefficients are positive, *Harvest Cost* will increase if *Slope* increases or the distance to the road increases. For each percent increase in *Slope*, the *Harvest Cost* will increase by 0.31 \$/ton for the regression with all predictors and 0.33 \$/ton for the spatially explicit regression. The *Harvest Cost* will also increase by 0.0076 \$/ton for each foot increase of the *Skidding Distance* in both models. Or expressed in other units, the *Harvest Cost* will increase by 7.58 \$/ton for each additional 100 feet to skid.

Increasing the *Slope* by 1% in the above stated example (average values for the tree data, *Slope* of 0%, *Skidding Distance* of 0 ft.) will increase the *Harvest Cost* from 20.86 \$/ton to 21.17 \$/ton. In addition *Harvest Cost* will increase to 21.93 \$/ton if the tree needs to be skidded by 100 ft.

The lower coefficient of *Skidding Distance* compared to *Slope* does not contradict the relative importance of the predictors (*Skidding Distance* 29.9%, *Slope* 9.5%), since the range of *Skidding Distance* is between 2.78 ft. and 4919.78 ft. and the range of *Slope* is only between 0% and 81.10%

8.2. Discussion of the CSF Harvest Costs

The calculated *Harvest Costs* for the specific stands of the Colorado State Forest based on the full harvest model, the regression model, and the spatially explicit regression model, serve as a validation of this research and its methods. The regression with all four predictors is useful since it produced almost the identical results

as were produced with the full model. The mean of all stand's *Harvest Costs* differed only by 0.75 \$/ton and the standard deviation differed by 0.11 \$/ton. This confirms again the high R-squared of 0.98, but also confirms the unexplained 1.72% variance of the regression. The spatially explicit regression differs more from these results. The mean of the regression with all predictors to the mean of the spatially explicit regression differs by 3.69 \$/ton. The standard deviation of the spatially explicit regression is with 8.6 \$/ton significantly lower than the full regression model's standard deviation of 10.18 \$/ton. This is because the spatially explicit regression is missing the two *Non-Spatial Predictors*, and assumes therefore a fixed value for the variables. Therefore the variance in *Cost* is smaller.

The produced *Cost Surface* for the southern part of the Colorado State Forest with a mean *Harvest Cost* of 40.83 \$/ton and a standard deviation of 15.75 \$/ton are higher than the other calculations means and standard deviations. The high mean value and the high standard deviation, result from the fact that many stands, or in the case of the *Cost Surface* pixels, are not connected to roads. Also the map (Figure 49) clearly shows that stands close to the road are in the lower price range (green color). Therefore extreme high *Skidding Distance* values result. Therefore the *Cost Surface* is only useful in areas where roads already exist, though usually roads are not created until a stand is to be harvested. This makes planning and cost calculation very difficult. Therefore a way that estimates where potential logging roads will be located, is needed to calculate meaningful *Harvest Costs* for areas without road access.

9. Conclusion

The research showed that *Spatial Predictors* predict 40% of *Timber Harvest Costs*. The remaining 60% are predicted by the variables *Trees per Acre* and *Volume per Tree*. Therefore the research question, which asks what the significance of *Spatial Predictors* on *Timber Harvest Costs* is, can be answered as follows: *Spatial Predictors* have a significance of 40% on *Timber Harvest Costs*.

The gap in literature, which is caused by the lack of studies investigating the influence of all predictors on *Timber Harvest Costs*, including all *Spatial Predictors*, can be closed with the developed methodology and results of this paper.

The second research question, which asks if it is possible to calculate *Timber Harvest Costs* solely based on *Spatial Predictors*, depends on the use case:

It is not possible to calculate with this method an absolute *Harvest Cost*, because an R-squared of 0.4045 of the spatially explicit regression model is too low to calculate *Harvest Costs* solely based on *Spatial Predictors*.

But this study was conducted in order to answer if it is possible to calculate *Timber Harvest Costs* for use in optimization models. Optimization models require iterating through millions of potential solutions and comparing results in terms of an objective function. For this kind of optimization a R-squared of 0.4045 is sufficient because it gives relative *Harvest Costs*. This allows optimization models to compare the *Costs* of different stands and scenarios. These models do not require absolute *Harvest Cost*.

Therefore the results of this research make it possible to include *Harvest Costs* in optimization models for ecological forestry approaches. With their inclusion optimization models are significantly improved.

The study also showed that *Skidding Distance* is a major factor in calculating *Harvest Cost* (29.9%). *Skidding Distance* heavily relies on the location of roads. As stated, roads usually are not created until a stand is harvested; which makes it very difficult to estimate the *Skidding Distance* properly. Therefore further research should be conducted on ways to determine *Skidding Distance* precisely.

Also this research is restricted to a limited study area. The method of this study should be repeated in other areas and the results compared via further analysis.

While the famous myth in GIS, that 80% of all data have a spatial component (Franklin and Hane 1992), remains unverified, this study proves that 40% of all timber harvest data have a spatial component.

Bibliography

Agafonkin, Vladimir. *Leaflet*. 01 08, 2014.

<https://github.com/Leaflet/Leaflet/blob/master/LICENSE> (accessed 02 02, 2014).

Arriagada, Rodrigo A., Frederick W. Cabbage, Karen Lee Abt, and Robert J. Huggett Jr. "Estimating harvest costs for fuel treatments in the west." *Forest Products Journal* 58 (7/8) (2008): 24-30.

Becker, Dennis R., Debra Larson, Eini C. Lowell, and Robert, B. Rummer. *User Guide for HCR Estimator 2.0: Software to Calculate Cost and Revenue Thresholds for Harvesting Small- Diameter Ponderosa Pine*. General Technical Report, Portland, OR: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station., 2008.

Boswell, B. 1998. *Vancouver Island mechanized thinning trials*. Technical Note TN-271, Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.

Bressert, Eli. *SciPy and NumPy*. Beijing Sebastopol: O'Reilly, 2012.

Bruce Ratner, Ph.D. *The Correlation Coefficient: Definition*. 03 02, 2012.

<http://www.dmstat1.com/res/TheCorrelationCoefficientDefined.html> (accessed 08 01, 2014).

Bruce R. Hartsough , Erik S. Drews , Joseph F. McNeel , Thomas A. Durston , and Bryce J. Stokes . "Comparison of mechanized systems for thinning ponderosa pine stands and mixed conifer stands." *Forest Products Journal* 47(11/12) (1997): 59-68.

Burnham, Kenneth P., and David R. Anderson. *Model selection and multimodel inference : a practical information-theoretic approach*. New York, NY: Springer, 2010.

Colorado Parks and Wildlife . "Colorado Parks and Wildlife property boundaries." Colorado Parks and Wildlife, 10 30, 2013.

Colorado Parks and Wildlife . *Colorado Parks and Wildlife State Forest*. 01 01, 2014. <http://cpw.state.co.us/placestogo/parks/StateForest/Pages/default.aspx> (accessed 10 06, 2014).

Conner, Roger C., Tim O. Adams, and Tony G. Johnson. "Assessing the potential for biomass energy development in South Carolina." *Res. Pap. SRS-46* (U.S. Department of Agriculture Forest Service, Southern Research Station), 2009: 19.

Dixon, Gary E. *Essential FVS: A User's Guide to the Forest Vegetation Simulator*. Fort Collins, CO: United States Department of Agriculture Forest Service, 2002.

Ecotrust. *Forest Planner*. 05 01, 2014. <http://forestplanner.ecotrust.org/trees/about-the-forest-planner#about> (accessed 10 10, 2014).

Engineering Toolbox. *Wood Species - Weight at various Moisture Contents*. 05 15, 2014. http://www.engineeringtoolbox.com/weight-wood-d_821.html (accessed 10 10, 2014).

ESRI. *ESRI Terraformer*. 08 15, 2013.

<https://github.com/Esri/Terraformer/blob/master/LICENSE> (accessed 09 02, 2014).

Fight, Roger D., Bruce R. Hartsough, and Peter Noordijk. *Users Guide for FRCS: Fuel Reduction Cost Simulator Software*. General Technical Report, Portland, OR: Pacific Northwest Research Station, 2006.

Franklin, Carl, and Paula Hane. "An introduction to GIS: linking maps to databases." *Database* 15 (2) (1992): 17-22.

Geofabrik GmbH . "GeoFabrik OpenStreetMap Colorado." *GeoFabrik*. 2013. <http://download.geofabrik.de/north-america/us/colorado.html> (accessed 07 23, 2014). GeoServer. *About - GeoServer*. 01 01, 2014. <http://geoserver.org/about/> (accessed 06 02, 2014).

Gesch, D.B. "The National Elevation Dataset." *Digital Elevation Model Technologies and Applications: The DEM Users Manual* (American Society for Photogrammetry and Remote Sensing) 2 (2007): 99-118.

Goodchild, Michael F., Bradley O. Parks, and L. T. Steyae. *Environmental modeling with GIS*. New York, NY: Oxford University Press, 1993.

Greene, W.D. and J.F. McNeel. 1991. *Productivity and cost of sawhead feller-bunchers in the South*. *Forest Products Journal* 41(3): 21-36.

Gingras, J.F. 1988. *The effect of site and stand factors on feller-buncher performance*. Technical Report TR-84, Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.

Gingras, J.F. 1996. *The cost of product sorting during harvesting*. FERIC Technical Note TN-245, Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.

Gonsior, M.J. and J.M. Mandzak. 1987. *Mechanized systems for harvesting small-stem lodgepole pine in mountainous terrain*. IN: R.L. Barger (comp.) *Management of Small-Stem Stands of Lodgepole Pine-Workshop Proceedings*. General Technical Report INT-237. USDA Forest Service, Intermountain Research Station, Ogden, UT: 53-66.

Hartsough, B.R. 2001. *Productivity and cost of harvesting to maintain high structural diversity, and resulting damage to residual trees*. Final report to the USDA Forest Service Pacific Southwest Research Station on Cooperative Agreement USDA-PSW-95-0028CA. University of California, Davis. 29 pages + appendices.

Hartsough, Bruce R., Xiaoshan Zhang, and Roger D. Fight. "Harvesting cost model for small trees in natural stands in the interior northwest." *Forest Products Journal* 51(4) (2001): 54-61.

Hartsough, Bruce R., Erik S. Drews, Joseph F. McNeel, Thomas A. Durston, and Bryce J. Stokes. *Forest Products Journal* 47(11/12) (1997): 59-68.

Henderson, B. 2001. *Roadside harvesting with low ground-pressure skidders in northwestern British Columbia*. FERIC Advantage 2(54), Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.

Holtzschler, M. and B. Lanford 1997. *Tree diameter effects on costs and productivity of cut-to-length systems*. *For. Prod. J.* 47(3):25-30.

- Hubbard, James E. *Colorado State Forest Forest Management Plan*. Management Plan, Walden, CO: Colorado State Forest, 1988.
- Jaccard, James, and Robert Turris. *Interaction effects in multiple regression*. Oaks, CA: Sage Publications, 2003.
- Johnson, L.R. 1988. *Summary of production and time studies of mechanized harvesting equipment in the intermountain west*. Forest Products Department, University of Idaho, Moscow.
- Johnson, L.R. 1979. *Production of wood from small diameter stands: a cost assessment*. Transactions of the American Society of Agricultural Engineers 22(3): 487-493.
- jQuery Foundation. *License | jQuery Foundation*. 07 02, 2014. <https://jquery.org/license/> (accessed 10 01, 2014).
- Keegan III, Charles E., Carl E. Fiedler, and Fred J. Stewart. " <p> Cost of Timber Harvest Under Traditional and "New Forestry" Silvicultural Prescriptions." *Western Journal of Applied Forestry* (Society of American Foresters) 10, no. 1 (01 1995): 36-42(7).
- Keegan, Charles E., III, Michael J. Niccolucci, Carl E. Fiedler, J. Greg Jones, and Roy W. Regel. "Harvest cost collection approaches and associated equations for restoration treatments on national forests. (Management)." *Forest Products Journal*, 2002.
- Kellogga, L.D., and P. P. Bettingera. " Highlight the Article title - Start CrossMark Snippet v1.3 abstract content Thinning Productivity and Cost for a Mechanized Cut-to-Length System in the Northwest Pacific Coast Region of the USA." *Journal of Forest Engineering /abstract content* 5, no. 2 (1994): 43-54.
- Kosicki, K. 2000. *Productivities and costs of two harvesting trials in a western Alberta riparian zone*. FERIC Advantage 1(19), Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.
- Kosicki, K. 2002. *Evaluation of Trans-Gesco TG88C and Tigercat 635 grapple skidders working in central Alberta*. FERIC Advantage 3(37), Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.
- Kosicki, K. 2002. *Productivity and cost of summer harvesting in a central Alberta mixedwood stand*. FERIC Advantage 3(6), Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.
- Loeffler, Dan, David E. Calkin, and Robin P. Silverstein. "Estimating volumes and costs of forest biomass in western Montana using forest inventory and geospatial data." *Forest products journal* 56(6) (2006): 31-37.
- Luxen, Dennis. *Server API Project OSRM*. 08 13, 2014. <https://github.com/Project-OSRM/osrm-backend/wiki/Server-api#nearest-point-on-a-street> (accessed 09 06, 2014).
- MacDonald, A.J. 1990. *A case study of roadside logging in the northern interior of British Columbia*. Technical Report TR-97, Forest Engineering Res. Inst. of Canada, Vancouver, BC.

- MapBox. *Introducing MapBox Satellite*. 12 04, 2012. <https://www.mapbox.com/blog/mapbox-satellite/> (accessed 09 05, 2014).
- MapBox. *Mapbox Terrain*. 01 01, 2013. <https://www.mapbox.com/developers/vector-tiles/mapbox-terrain/> (accessed 10 01, 2014).
- Miyata, E.S. 1980. *Determining fixed and operating costs of logging equipment*. Gen. Tech. Rep. NC-55. St. Paul, MN: U.S. Department of Agriculture, Forest Service, North Central Forest Experiment Station.
- OpenStreetMap Wiki contributors. *Overpass API*. OpenStreetMap Wiki. 09 16, 2014. http://wiki.openstreetmap.org/w/index.php?title=Overpass_API&oldid=1085574 (accessed 10 10, 2014).
- Perry, Matthew. *Rasterstats*. 10 01, 2014. <https://github.com/perrygeo/python-rasterstats> (accessed 10 10, 2014).
- Peter J. Ince, John W. Henley, John B. Grantham, and Douglas L. Hunt. *Costs of harvesting beetle-milled lodgepole pine in eastern Oregon*. General Technical Report PNW-GTR-165, Portland, OR: USDA Forest Service, Pacific Northwest Research Station, 1984.
- Plamondon, J. 1998. *Trials of mechanized tree-length harvesting in eastern Canada*. Technical Note TN-273, Forest Engineering Res. Inst. of Canada, Pointe Claire, PQ.
- Ronacher, Armin. *Flask*. 01 01, 2013. <http://flask.pocoo.org/docs/0.10/> (accessed 07 02, 2014).
- Rouault, Even. *GDAL - Geospatial Data Abstraction Library*. 12 10, 2011. <http://www.gdal.org/python/> (accessed 10 04, 2014).
- Schroder, P.C. and L.R. Johnson. 1997. *Production functions for cut-to-length harvesting in bunched and unbunched material*. IN: J.J. Ball and L.W. Starnes (eds.) Proceedings, Forest Operations for Sustainable Forests and Healthy Economies, 20th COFE Annual Meeting. COFE, Corvallis, OR: 52-61.
- Silverstein, Robin P., Dan Loeffler, J. Greg Jones, Dave E. Calkin, Hans R. Zuuring, and Martin Twer. "Biomass utilization modeling on the Bitterroot National Forest." *Fuels Management-How to Measure Success: Conference Proceedings*. Portland, OR: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, 2006. 673-688.
- Smidt, M. F, R. A. Tufts, and T. V. Gallagher. *Stump to Mill Cost Program (STOMP)*. 2009. http://www.auburn.edu/academic/forestry_wildlife/forestops//stomp/ (accessed 10 15, 2014).
- Society of American Foresters. *SAFnet Dictionary Definition for [clearcut]*. 06 11, 2010. <http://dictionaryofforestry.org/dict/term/clearcut> (accessed 10 03, 2014).
- Teetor, Paul, and Michael K. Loukide. *R cookbook*. Sebastopol, CA: O'Reilly Media, 2011.
- Thompson, Jason. *Forest Residue Trucking Simulator (FoRTS)*. 03 14, 2014. <https://www.frames.gov/rcs/7000/7652.html> (accessed 10 10, 2014).

Townsend, Hunter. "Colorado State Forest Timber Sales." *Shapefile*. Walden, CO: Colorado State Forest Service, 01 01, 2014.

Toye, Jacob. *Leaflet Draw*. 10 03, 2014. <https://github.com/Leaflet/Leaflet.draw> (accessed 10 10, 2014).

Tufts, R.A., B.L. Lanford, W.D. Greene, and J.O. Burrows. "Auburn harvesting analyzer." *The Compiler* 3 (2) (1985): 14-15.

Tufts, R.A., B.J. Stokes and B.L Lanford. 1988. *Productivity of grapple skidders in southern pine*. *Forest Products Journal* 38(10): 24-30.

USDA Forest Service - Northern Research Station Right. "Timber Harvesting." *USDA Forest Service - Northern Research Station*. 01 01, 2014. <http://www.nrs.fs.fed.us/fmg/nfmg/docs/mn/harvesting.pdf> (accessed 09 01, 2014).

Yhat. *Fitting & Interpreting Linear Models in R*. 05 18, 2013. <http://blog.yhathq.com/posts/r-lm-summary.html> (accessed 09 23, 2014).

Zuuring, H R, W L Wood, and J G Jones. *OVERVIEW OF MAGIS: A MULTI-RESOURCE ANALYSIS AND GEOGRAPHIC INFORMATION SYSTEM*. FSRN-INT-427, Ogden, UT: USForest Service, 1995.

Appendix

Appendix I

Costs for machine and labor for Western Colorado (Fight, Hartsough and Noordijk 2006)

	Faller or Bucker	All Others
Wage and benefit rates (\$/person-SH)=	32.22	25.56

Machine description:	FBuncher	FBuncher	FBuncher	Skidder	Skidder	Processor	Processor
1. Input Data:	DriveTo Tree	Swing Boom	SelfLeveling	small	big	small	big
Purchase price as of Dec 02 (P, \$) =	176,670	365,119	365,119	164,892	235,561	353,341	471,121
Machine Horsepower rating (hp) =	150	200	240	120	200	120	200
Machine life (n, years) =	3	5	5	5	4	5	5
Salvage value, percent of purchase price (sv%)	20%	15%	15%	20%	20%	20%	20%
Utilization rate, ph/sh (ut%) =	65%	60%	60%	65%	65%	65%	65%
Repair and maintenance, percent of depreciation (rm%) =	100%	75%	75%	90%	90%	100%	100%
Interest rate, percent of avg yearly investment (in%) =	8%	8%	8%	8%	8%	8%	8%
Insurance and tax rate, percent of avg yearly investment (it%) =	7%	7%	7%	7%	7%	7%	7%
Fuel consumption rate (fcr, gal/hp-ph) =	0.026	0.026	0.026	0.028	0.028	0.022	0.022
Fuel cost per gallon (fcg, \$/gal) =	3.327	3.327	3.327	3.327	3.327	3.327	3.327
Lube and oil, percent of fuel cost (lo%) =	37%	37%	37%	37%	37%	37%	37%
Crew size (persons) =	1	1	1	1	1	1	1
Crew wage and benefits (WB, \$/SH) =	35.46	35.46	35.46	35.46	35.46	35.46	35.46
Scheduled machine hours (SMH, sh/year) =	1600	1600	1600	1600	1600	1600	1600

Appendix II

Script CostFunc.py

```

import math
import operator

def costfunc(Slope, SkidDist, TPA, VPT):

    #####
    # General Inputs & Functions #
    #####

    TreeVol = VPT

    # Buncher $/PMH
    costPMHFBDDT = 181.62 # FbuncherDriveToTree $/PMH
    costPMHFBBS = 233.61 # FbuncherSwingBoom $/PMH
    costPMHFBSL = 238.35 # FbuncherSelfLeveling $/PMH
    # Skidder $/PMH
    costPMHSS = 134.24 # Skiddersmall $/PMH
    costPMHSB = 189.93 # Skidderbig $/PMH
    ManualMachineSize = min (1.0, TreeVol/150.0)
    SkidderHourlyCost = round(costPMHSS*(1-
ManualMachineSize)+costPMHSB*ManualMachineSize)
    # Processor $/PMH
    costPMHPS = 209.96 # Processorsmall $/PMH
    costPMHPB = 265.78 # Processorbig $/PMH
    MechMachineSize = min (1.0,TreeVol/80.0)
    ProcessorHourlyCost = round(costPMHPS*(1-
MechMachineSize)+costPMHPB*MechMachineSize)
    # Loader $/PMH
    costPMHLS = 147.07 # Loadersmall $/PMH
    costPMHLB = 180.5 # Loaderbig $/PMH
    LoaderHourlyCost = round(costPMHLS*(1-
ManualMachineSize)+costPMHLB*ManualMachineSize)

    WoodDensity = 39.0 # http://www.engineeringtoolbox.com/weight-wood-d_821.html

    DBH = ((TreeVol+3.675)/0.216)**0.5
    ButtDiam = DBH+3.0

    LogLength = 32.0
    LogsPerTree = max(1.0, (32.0/LogLength)*(-0.43+0.678*(DBH)**0.5))

    # General Functions
    def relevancefunction(cost, relevances, volumes):
        sum_map = sum(map( operator.mul, relevances, volumes))
        if sum_map == 0:
            sum_map = 0.000001
        return cost*sum(relevances)/sum_map

    def volumePMH (vol, ti): # Volume per PMH (Volumte, Time) Function
        return (vol/(ti/60.0))

    #####
    # Fell and Bunch #
    #####
    def fellbunch():
        # General Calculations
        if Slope<15:
            NonSelfLevelCabDummy = 1.0
        elif Slope<35:
            NonSelfLevelCabDummy = 1.75-0.05*Slope
        else :
            NonSelfLevelCabDummy = 0.0

```

```

CSlopeFB_Harv = 0.00015*Slope**2.0+0.00359*NonSelfLevelCabDummy*Slope
CRemovalsFB_Harv = max(0, (0.66-0.001193*TPA*2.47+5.357*10.0**(-
7.0)*(TPA*2.47)**2.0))
costPMHFBSw=costPMHFBS*NonSelfLevelCabDummy+costPMHFBSL*(1-
NonSelfLevelCabDummy)

DistBetweenTrees = (43560.0/ (max(TPA,1)))**0.5

# Relevance
# Drive-to-Tree
# Melroe Bobcat (Johnson, 79)
if DBH<10:
    relevanceFBDJohnson79 = 1.0
elif DBH<15:
    relevanceFBDJohnson79 = 3.0-DBH/5.0
else:
    relevanceFBDJohnson79 = 0

if Slope<10:
    relevanceFBDJohnson79 = relevanceFBDJohnson79
elif Slope<20:
    relevanceFBDJohnson79 = relevanceFBDJohnson79*(2.0-Slope/10.0)
else:
    relevanceFBDJohnson79 = 0

# Chainsaw Heads (Greene&McNeel, 91)
if DBH<15:
    relevanceFBDGreene911 = 1.0
elif DBH<20:
    relevanceFBDGreene911 = 4.0-DBH/5.0
else:
    relevanceFBDGreene911 = 0

if Slope<10:
    relevanceFBDGreene911 = relevanceFBDGreene911
elif Slope<20:
    relevanceFBDGreene911 = relevanceFBDGreene911*(2.0-Slope/10.0)
else:
    relevanceFBDGreene911 = 0

# Intermittent Circular Sawheads (Greene&McNeel, 91)
relevanceFBDGreene912 = relevanceFBDGreene911

# Hydro-Ax 211 (Hartsough, 01)
if DBH<10:
    relevanceFBDHartsough01 = 1.0
elif DBH<15:
    relevanceFBDHartsough01 = 3.0-DBH/5.0
else:
    relevanceFBDHartsough01= 0

if Slope<10:
    relevanceFBDHartsough01 = relevanceFBDHartsough01
elif Slope<20:
    relevanceFBDHartsough01 = relevanceFBDHartsough01*(2.0-
Slope/10.0)
else:
    relevanceFBDHartsough01 = 0

# Swing Boom
# Timbco 2520&Cat 227 (Johnson, 88)
if DBH<15:
    relevanceFBSJohnson88 = 1.0
elif DBH<20:
    relevanceFBSJohnson88 = 4.0-DBH/5.0
else:
    relevanceFBSJohnson88 = 0

```

```

    if Slope<5:
        relevanceFBSJohnson88 = 0
    elif Slope<20:
        relevanceFBSJohnson88 = relevanceFBSJohnson88*(-
1.0/3.0+Slope/15.0)
    else:
        relevanceFBSJohnson88 = relevanceFBSJohnson88

# JD 693B&TJ Timbco 2518 (Gingras, 88)
if DBH<12:
    relevanceFBSGingras88 = 1.0
elif DBH<18:
    relevanceFBSGingras88 = 3.0-DBH/6.0
else:
    relevanceFBSGingras88 = 0

if Slope<5:
    relevanceFBSGingras88 = 0
elif Slope<20:
    relevanceFBSGingras88 = relevanceFBSGingras88*(-
1.0/3.0+Slope/15.0)
else:
    relevanceFBSGingras88 = relevanceFBSGingras88

# Timbco (Gonsier&Mandzak, 87)
if DBH<15:
    relevanceFBSGonsier87 = 1.0
elif DBH<20:
    relevanceFBSGonsier87 = 4.0-DBH/5.0
else:
    relevanceFBSGonsier87 = 0

if Slope<15:
    relevanceFBSGonsier87 = 0
elif Slope<35:
    relevanceFBSGonsier87 = relevanceFBSGonsier87*((-
3.0/4.0)+(Slope/20))
else:
    relevanceFBSGonsier87 = relevanceFBSGonsier87

# FERIC Generic (Gingras, J.F., 96. The cost of product sorting during
harvesting. FERIC Technical Note TN-245)
if Slope<5:
    relevanceFBSGingras96 = 0
elif Slope<20:
    relevanceFBSGingras96 = -1.0/3.0+Slope/15.0
else:
    relevanceFBSGingras96 = 1.0

# Plamondon, J. 1998. Trials of mechanized tree-length harvesting in
eastern Canada. FERIC Technical Note TN-273)
if TreeVol<20:
    relevanceFBSPlamondon98 = 1.0
elif TreeVol<50:
    relevanceFBSPlamondon98 = 5.0/3.0-TreeVol/30.0
else:
    relevanceFBSPlamondon98 = 0

if Slope<5:
    relevanceFBSPlamondon98 = 0
elif Slope<20:
    relevanceFBSPlamondon98 = relevanceFBSPlamondon98*(-
1.0/3.0+Slope/15.0)
else:
    relevanceFBSPlamondon98 = relevanceFBSPlamondon98

# Timbco 420 (Hartsough, B., E. Drews, J. McNeel, T. Durston and B.
Stokes. 97.
if DBH<15:

```

```

        relevanceFBSHartsough97 = 1.0
    elif DBH<20:
        relevanceFBSHartsough97 = 4.0-DBH/5.0
    else:
        relevanceFBSHartsough97 = 0

    if Slope<5:
        relevanceFBSHartsough97 = 0
    elif Slope<20:
        relevanceFBSHartsough97 = relevanceFBSHartsough97*(-
1.0/3.0+Slope/15.0)
    else:
        relevanceFBSHartsough97 = relevanceFBSHartsough97

    # Time per Tree
    TPTJohnson79 = 0.204+0.00822*DistBetweenTrees+0.02002*DBH+0.00244*Slope
    TPTGreene911 = (-
0.0368+0.02914*DBH+0.00289*DistBetweenTrees+0.2134*1.1)*(1+CSlopeFB_Harv)
    TPTGreene912 = (-
0.4197+0.01345*DBH+0.001245*DistBetweenTrees+0.7271*1.01)*(1+CSlopeFB_Harv)

    TPAccumHartsough01 = max(1,14.2-2.18*DBH+0.0799*DBH**2)
    TimeAccumHartsough01 =
0.114+0.266+0.073*TPAccumHartsough01+0.00999*TPAccumHartsough01*DBH
    TPPMHHartsough01 = 60*TPAccumHartsough01/TimeAccumHartsough01

    BoomReachJohnson88 = 24
    TreesInReachJohnson88 = TPA*math.pi*BoomReachJohnson88**2/43560
    TreesPCJohnson88 = max(1,TreesInReachJohnson88)
    TPCJohnson88 =
(0.242+0.1295*TreesPCJohnson88+0.0295*DBH*TreesPCJohnson88)*(1+CSlopeFB_Harv)
    TPTJohnson88 = TPCJohnson88/TreesPCJohnson88

    TPTGonsier87 = (0.324+0.00138*DBH**2)*(1+CSlopeFB_Harv+CRemovalsFB_Harv)

    UnmerchMerchGingras88 = min(1.5,(285/(2.47*TPA)))
    TreesInReachGingras88 = TPA*math.pi*24**2/43560
    ObsTreesPerCycleGingras88 = (4.36+9-(0.12+0.34)*DBH+0.00084*2.47*TPA)/2
    TPCGingras88 =
max(1,min(TreesInReachGingras88,ObsTreesPerCycleGingras88))
    TPPMHGingras88 = (127.8+21.2*TPCGingras88-
63.1*UnmerchMerchGingras88+0.033*285)/(1+CSlopeFB_Harv)

    TreesInReachHartsough97 = TPA*math.pi*24**2/43560
    TreesPAccumHartsough97 = max(1,1.81-0.0664*DBH+3.64/DBH-0.0058*20.0)
    MoveFracHartsough97 =
0.5/(math.trunc(TreesInReachHartsough97/TreesPAccumHartsough97)+1)
    MoveHartsough97 = 0.192+0.00779*(24+DistBetweenTrees)
    TimeFellHartsough97 =
0.285+0.126*TreesPAccumHartsough97+0.0176*DBH*TreesPAccumHartsough97
    TPAccumHartsough97 =
MoveFracHartsough97*MoveHartsough97+TimeFellHartsough97
    TPTHartsough97 =
(TPAccumHartsough97*(1+0.0963)/TreesPAccumHartsough97)*(1+CSlopeFB_Harv)

    # Calculate Volume per PMH
    FBDVolumePMHJohnson79 = volumePMH (TreeVol, TPTJohnson79)
    FBDVolumePMHGreene911 = volumePMH (TreeVol, TPTGreene911)
    FBDVolumePMHGreene912 = volumePMH (TreeVol, TPTGreene912)
    FBDVolumePMHHartsough01 = TreeVol * TPPMHHartsough01
    FBSVolumePMHJohnson88 = volumePMH (TreeVol, TPTJohnson88)
    FBSVolumePMHGingras88 = TreeVol*TPPMHGingras88
    FBSVolumePMHGonsier87 = volumePMH (TreeVol, TPTGonsier87)
    FBSVolumePMHGingras96 =
(50.338/0.028317*(TreeVol*0.028317)**0.3011)/(1+CSlopeFB_Harv+CRemovalsFB_Harv)
    FBSVolumePMHPlamondon98
= (5.0/0.028317+57.7*TreeVol)/(1.0+CSlopeFB_Harv+CRemovalsFB_Harv)
    FBSVolumePMHHartsough97 = volumePMH (TreeVol, TPTHartsough97)

```

```

# Felling Cost ($/ ccf)
CostFellBunch = round((
    (costPMHFBDDTT*relevanceFBDJohnson79
    +costPMHFBDDTT*relevanceFBDGreene911
    +costPMHFBDDTT*relevanceFBDGreene912
    +costPMHFBDDTT*relevanceFBDHartsough01
    +costPMHFBFSw*relevanceFBSJohnson88
    +costPMHFBFSw*relevanceFBSGingras88
    +costPMHFBSL*relevanceFBSGonsier87
    +costPMHFBFSw*relevanceFBSGingras96
    +costPMHFBFSw*relevanceFBSPlamondon98
    +costPMHFBFSw*relevanceFBSHartsough97)/
    (relevanceFBDJohnson79*FBDVolumePMHJohnson79
    +relevanceFBDGreene911*FBDVolumePMHGreene911
    +relevanceFBDGreene912*FBDVolumePMHGreene912
    +relevanceFBDHartsough01*FBDVolumePMHHartsough01
    +relevanceFBSJohnson88*FBSVolumePMHJohnson88
    +relevanceFBSGingras88*FBSVolumePMHGingras88
    +relevanceFBSGonsier87*FBSVolumePMHGonsier87
    +relevanceFBSGingras96*FBSVolumePMHGingras96
    +relevanceFBSPlamondon98*FBSVolumePMHPlamondon98
    +relevanceFBSHartsough97*FBSVolumePMHHartsough97)), 4)
return CostFellBunch

#####
# Skidding #
#####
def skidding():

    # General Calculations
    CSlopeSkidForwLoadSize = 1.0-0.000127*Slope**2.0
    TurnVol = 44.87*TreeVol**0.282*CSlopeSkidForwLoadSize

    # Relevance
    # Grapple Skidders (Johnson, 88)
    if ButtDiam < 15:
        relevanceSBJohnson88 = 1
    elif ButtDiam < 20:
        relevanceSBJohnson88 = 4 - ButtDiam/5
    else:
        relevanceSBJohnson88 = 0

    # Grapple Skidders (Tufts et al, 88)
    relevanceSBTufts88 = 0.5

    # John Deere 748E (Kosicki, K. 00. Productivities and costs of two
    harvesting trials in a western Alberta riparian zone. FERIC Advantage 1(19))
    if TreeVol < 5:
        relevanceSBKosicki00 = 0
    elif TreeVol < 10:
        relevanceSBKosicki00 = -1 + TreeVol/5
    elif TreeVol < 50:
        relevanceSBKosicki00 = 1
    elif TreeVol < 100:
        relevanceSBKosicki00 = 2 - TreeVol/50
    else:
        relevanceSBKosicki00 = 0

    # Cat D5H TSK Custom Track (Henderson, B. 01. Roadside harvesting with
    low ground-pressure skidders in northwestern British Columbia. FERIC Advantage 2(54))
    if TreeVol < 5:
        relevanceSBHenderson01 = 0
    elif TreeVol < 10:
        relevanceSBHenderson01 = -1 + TreeVol/5
    elif TreeVol < 50:
        relevanceSBHenderson01 = 1
    elif TreeVol < 100:

```

```

        relevanceSBHenderson01 = 2 - TreeVol/50
    else:
        relevanceSBHenderson01 = 0

    # JD 748_G-II & TJ 560 (Kosicki, K. 02. Productivity and cost of summer
    harvesting in a central Alberta mixedwood stand. FERIC Advantage 3(6))
    if TreeVol < 30:
        relevanceSBKosicki021 = 1
    elif TreeVol < 60:
        relevanceSBKosicki021 = 2 - TreeVol/30
    else:
        relevanceSBKosicki021 = 0

    # Tigercat 635 (Boswell, B. 98. Vancouver Island mechanized thinning
    trials. FERIC Technical Note TN-271)
    if TreeVol < 5:
        relevanceSBBoswell198 = 0
    elif TreeVol < 10:
        relevanceSBBoswell198 = -1 + TreeVol/5
    elif TreeVol < 100:
        relevanceSBBoswell198 = 1
    elif TreeVol < 150:
        relevanceSBBoswell198 = 3 - TreeVol/50
    else:
        relevanceSBBoswell198 = 0

    # Tigercat 635 (Kosicki, K. 02. Evaluation of Trans-Gesco TG88C and
    Tigercat 635 grapple skidders working in central Alberta. FERIC Advantage 3(37))
    if TreeVol < 40:
        relevanceSBKosicki022 = 1
    elif TreeVol < 80:
        relevanceSBKosicki022 = 2 - TreeVol/40
    else:
        relevanceSBKosicki022 = 0

    relevanceSkidB = []
    relevanceSkidB.extend(value for name, value in sorted(locals().items(),
    key=lambda item: item[0]) if name.startswith('relevanceSB'))

    # Trees per Turn
    TreesPerTurnS = TurnVol/TreeVol

    # Turn relevance (lb)
    Turnrelevance = TurnVol*WoodDensity

    # Turn Time Johnson 88
    TravelEmpty = -2.179+0.0362*Slope+0.711*math.log(SkidDist)
    TravelLoaded = -
    0.919+0.00081*SkidDist+0.000062*Slope**3+0.353*math.log(SkidDist)
    LoadTime = max(0,0.882+0.0042*Slope**2-0.000048*(TreesPerTurnS)**3)
    DeckTime = 0.063+0.55*math.log(3)+0.0076*3*TreesPerTurnS
    turnTimeJohnson88 = TravelEmpty + TravelLoaded + LoadTime + DeckTime

    # Turn Time Tufts 88
    skidderHP=50.5+5.74*(TreeVol**0.5)
    treesPerBunch = 6 # F44 = 'Fell&Bunch'!E28 just temporary
    bunchVolume = TreeVol*treesPerBunch
    bunchesPerTurn = max(1,TurnVol/bunchVolume)
    travelEmpty =(0.1905*SkidDist+0.3557*skidderHP-
    0.0003336*SkidDist*skidderHP)/100
    grapple = min(5,(-38.36+161.6*bunchesPerTurn-
    0.5599*bunchesPerTurn*skidderHP+1.398*bunchesPerTurn*treesPerBunch)/100)
    travelLoaded =(-34.52+0.2634*SkidDist+0.7634*skidderHP-
    0.00122*SkidDist*skidderHP+0.03782*SkidDist*bunchesPerTurn)/100
    ungrapple = max(0,(5.177*bunchesPerTurn+0.002508*Turnrelevance-
    0.00007944*Turnrelevance*bunchesPerTurn*treesPerBunch*bunchesPerTurn)/100)
    turnTimeTufts88 = 1.3*(travelEmpty+grapple+travelLoaded+ungrapple)

    # Turn Time Kosicki 00

```

```

turnTimeKosicki00 = 0.65+0.0054*SkidDist+0.244*2.1

# Turn Time Henderson 01
turnTimeHenderson01 = 2.818+0.0109*SkidDist

# Turn Time Kosicki 02-1
turnTimeKosicki021 = 0.649+0.0058*SkidDist+0.581*bunchesPerTurn

# Turn Time Boswell 98
turnTimeBoswell98 = 5.77 + 0.007 * SkidDist

# Turn Time Kosicki 02-2
turnTimeKosicki022 = 2.98+0.006*SkidDist+0.27*TreesPerTurnS

# Skidding Volume/ PMH (ccf)
BSkidVolPMHJohnson88 = volumePMH (TurnVol, turnTimeJohnson88)
BSkidVolPMHTufts88 = volumePMH (TurnVol, turnTimeTufts88)
BSkidVolPMHKosicki00 = volumePMH (TurnVol, turnTimeKosicki00)
BSkidVolPMHHenderson01 = volumePMH (TurnVol, turnTimeHenderson01)
BSkidVolPMHKosicki021 = volumePMH (TurnVol, turnTimeKosicki021)
BSkidVolPMHBoswell98 = volumePMH (TurnVol, turnTimeBoswell98)
BSkidVolPMHKosicki022 = volumePMH (TurnVol, turnTimeKosicki022)

volumeSkidB = []
volumeSkidB.extend(value for name, value in sorted(locals().items(),
key=lambda item: item[0]) if name.startswith('BSkidVolPMH'))

# Skidding cost ($/ ccf)
CostSkidBun = round (relevancefunction(SkidderHourlyCost,
relevanceSkidB, volumeSkidB), 4)

return CostSkidBun

#####
# Process #
#####
def process():
    # Relevance
    # Hahn Stroke Processor (Gonsier&Mandzak, 87)
    if DBH<15:
        relevanceProGonsier87= 1.0
    elif DBH<20:
        relevanceProGonsier87 = 4.0-DBH/5.0
    else:
        relevanceProGonsier87 = 0

    # Stroke Processor (MacDonald, 90)
    if ButtDiam<20:
        relevanceProMacDonald90 = 1.0
    elif ButtDiam<30:
        relevanceProMacDonald90 = 3.0-ButtDiam/10.0
    else:
        relevanceProMacDonald90 = 0

    # Roger Stroke Processor (Johnson, 88)
    relevanceProJohnson881 = 1

    # Harricana Stroke Processor (Johnson, 88)
    relevanceProJohnson882 = 1

    # Hitachi EX150/Keto 500 (Schroder&Johnson, 97)
    if TreeVol<50:
        relevanceProSchroder97 = 1
    elif TreeVol<100:
        relevanceProSchroder97 = 2-TreeVol/50
    else:
        relevanceProSchroder97 = 0

```

```

# FERIC Generic (Gingras, J.F. 96)
relevanceProGingras96 = 1

# Valmet 546 Woodstar Processor (Holtzscher, M. and B. Lanford 1997)
if TreeVol<20:
    relevanceProHoltzscher97 = 1
elif TreeVol<40:
    relevanceProHoltzscher97 = 2-TreeVol/20
else:
    relevanceProHoltzscher97 = 0

relevanceP = []
relevanceP.extend(value for name, value in sorted(locals().items(),
key=lambda item: item[0]) if name.startswith('relevancePro'))

# Time per Tree
TPTGonsier87 = 1.26*(0.232+0.0494*DBH)
TPTMacDonald90 = 0.153+0.0145*ButtDiam
TPTJohnson881 = -0.05+0.6844*LogsPerTree+5*10**(-8)*TreeVol**2
TPTJohnson882 = -0.13+0.001*ButtDiam**2+0.5942*LogsPerTree
TPTSchroder97 = (0.67+0.0116*TreeVol)**2
TPTHoltzscher97 = -0.341+0.1243*DBH

# Skidding Volume/ PMH (ccf)
ProVolPMHGonsier87 = volumePMH (TreeVol, TPTGonsier87)
ProVolPMHMacDonald90 = volumePMH (TreeVol, TPTMacDonald90)
ProVolPMHJohnson881 = volumePMH (TreeVol, TPTJohnson881)
ProVolPMHJohnson882 = volumePMH (TreeVol, TPTJohnson882)
ProVolPMHSchroder97 = volumePMH (TreeVol, TPTSchroder97)
ProVolPMHGingras96 = (41.16/0.02832)*(TreeVol/35.31)**0.4902
ProVolPMHHoltzscher97 = volumePMH (TreeVol, TPTHoltzscher97)

volumeP = []
volumeP.extend(value for name, value in sorted(locals().items(),
key=lambda item: item[0]) if name.startswith('ProVolPMH'))

# Processing cost ($/ ccf)
CostProcess = round(relevancefunction(ProcessorHourlyCost, relevanceP,
volumeP),4)

return CostProcess

#####
# Results #
#####
CostFellBunch = fellbunch() # in $/CF
CostSkid = skidding() # in $/CF
CostProcess = process() # in $/CF
Cost = (CostFellBunch + CostSkid + CostProcess)/WoodDensity*2000 # Convert $/CF
to $/ton

return Cost

```


Appendix III

Script Data.py

```
import ogr, gdal, osr, os
import numpy as np
import requests
import json

def coord2pixelOffset(rasterfn,x,y):
    raster = gdal.Open(rasterfn)
    geotransform = raster.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    xOffset = int((x - originX)/pixelWidth)
    yOffset = int((y - originY)/pixelHeight)
    return xOffset,yOffset

def pixelOffset2coord(rasterfn,xOffset,yOffset):
    raster = gdal.Open(rasterfn)
    geotransform = raster.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    coordX = originX+pixelWidth*xOffset
    coordY = originY+pixelHeight*yOffset
    return coordX, coordY

def stand2standArray(stand_wkt,rasterfn,pixel_size):

    #Define pNoData value of new raster
    NoData_value = 255

    # Write stand_wkt to temp file to read extent
    stand_geom = ogr.CreateGeometryFromWkt(stand_wkt)
    shpDriver = ogr.GetDriverByName('ESRI Shapefile')
    if os.path.exists('temp.shp'):
        shpDriver.DeleteDataSource('temp.shp')
    outDataSource = shpDriver.CreateDataSource('temp.shp')
    tempLayer = outDataSource.CreateLayer('temp.shp', geom_type=ogr.wkbPolygon )
    featureDefn = tempLayer.GetLayerDefn()
    outFeature = ogr.Feature(featureDefn)
    outFeature.SetGeometry(stand_geom)
    tempLayer.CreateFeature(outFeature)
    spatialRef = osr.SpatialReference()
    spatialRef.ImportFromEPSG(26913)
    spatialRef.MorphToESRI()
    file = open('temp.prj', 'w')
    file.write(spatialRef.ExportToWkt())
    file.close()

    x_min, x_max, y_min, y_max = tempLayer.GetExtent()
    xOffset,yOffset = coord2pixelOffset(rasterfn,x_min,y_max)
    x_min,y_max = pixelOffset2coord(rasterfn,xOffset,yOffset)
    outDataSource = None
    tempLayer = None
    outFeature = None
    tempFile = ogr.Open('temp.shp')
    lyr = tempFile.GetLayer()

    # Create temporary raster file and rasterize stand_wkt
    x_res = int((x_max - x_min) / pixel_size)
    y_res = int((y_max - y_min) / pixel_size)
    target_ds = gdal.GetDriverByName('MEM').Create('', x_res, y_res, 1, gdal.GDT_Byte)
    target_ds.SetGeoTransform((x_min, pixel_size, 0, y_max, 0, -pixel_size))
```

```

band = target_ds.GetRasterBand(1)
band.SetNoDataValue(NoData_value)
gdal.RasterizeLayer(target_ds, [1], lyr, burn_values=[1])

# remove temp files
shpDriver.DeleteDataSource('temp.shp')

# Read as array
standArray = band.ReadAsArray()
return standArray, x_min, y_max, y_min

def createTPAarray(standArray, TPA):
    TPAarray = standArray.astype(float)
    TPAarray[TPAarray==1] = np.random.normal(loc=TPA, scale=TPA/4,
size=(TPAarray==1).sum())
    return TPAarray[TPAarray != 0].flatten()

def createVPTarray(standArray, VPT):
    VPTarray = standArray.astype(float)
    VPTarray[VPTarray==1] = np.random.normal(loc=VPT, scale=VPT/4,
size=(VPTarray==1).sum())
    return VPTarray[VPTarray != 0].flatten()

def createSarray(standArray, slopefn, x_min, y_max):
    xOffset,yOffset = coord2pixelOffset(slopefn,x_min,y_max)

    raster = gdal.Open(slopefn)
    band = raster.GetRasterBand(1)
    slopeArray = band.ReadAsArray()
    Sarray = slopeArray[yOffset:(yOffset+standArray.shape[0]),
xOffset:(xOffset+standArray.shape[1])]
    return Sarray[standArray == 1].flatten()

def createSDarray(stand_wkt, standArray,pixel_size,x_min, y_max, y_min):

    def createBBOX(stand_wkt,offsetBbox):
        stand_geom = ogr.CreateGeometryFromWkt(stand_wkt)
        bbox = stand_geom.GetEnvelope()
        xmin,xmax,ymin,ymax = bbox
        xmin -= offsetBbox
        xmax += offsetBbox
        ymin -= offsetBbox
        ymax += offsetBbox
        bbox = xmin,xmax,ymin,ymax
        leftbottom = ogr.Geometry(ogr.wkbPoint)
        leftbottom.AddPoint(bbox[0], bbox[2])
        righttop = ogr.Geometry(ogr.wkbPoint)
        righttop.AddPoint(bbox[1], bbox[3])
        inSpatialRef = osr.SpatialReference()
        inSpatialRef.ImportFromEPSG(26913)
        outSpatialRef = osr.SpatialReference()
        outSpatialRef.ImportFromEPSG(4326)

        coordTransform = osr.CoordinateTransformation(inSpatialRef, outSpatialRef)
        leftbottom.Transform(coordTransform)
        righttop.Transform(coordTransform)

        bboxWGS84 = (leftbottom.GetX(), righttop.GetX(),leftbottom.GetY(),
righttop.GetY())
        return bboxWGS84

    def osmRoadsAPI(bboxWGS84):
        bboxCoords = str(bboxWGS84[0]) + ',' + str(bboxWGS84[2]) + ',' +
str(bboxWGS84[1]) + ',' + str(bboxWGS84[3])
        url = 'http://www.overpass-api.de/api/xapi?way[highway=*][bbox=%s]' % bboxCoords
        osm = requests.get(url)
        file = open(r'OSMroads.osm', 'w')
        file.write(osm.text)
        file.close()

```

```

roadsDs = ogr.Open('OSMroads.osm')
roadsLayer = roadsDs.GetLayer(1) # layer 1 for ways
feature = False
if roadsLayer.GetNextFeature():
    feature = True
return feature

def downloadOSMroad(stand_wkt):
    offsetBbox = 0
    bboxWGS84 = createBBOX(stand_wkt,offsetBbox)
    feature = osmRoadsAPI(bboxWGS84) # creates 'roads.osm'
    while feature == False:
        offsetBbox += 100
        bboxWGS84 = createBBOX(stand_wkt,offsetBbox)
        feature = osmRoadsAPI(bboxWGS84)

def createQuaterCentroids(stand_wkt):
    '''
    This function splits the stand in four quater and returns the centroids of the
    four split quaters
    '''

    geom_poly = ogr.CreateGeometryFromWkt(stand_wkt)

    # Create 4 squares polygons
    geom_poly_envelope = geom_poly.GetEnvelope()
    minX = geom_poly_envelope[0]
    minY = geom_poly_envelope[2]
    maxX = geom_poly_envelope[1]
    maxY = geom_poly_envelope[3]

    '''
    coord0----coord1----coord2
    |           |           |
    |           |           |
    coord3----coord4----coord5
    |           |           |
    |           |           |
    coord6----coord7----coord8
    '''
    coord0 = minX, maxY
    coord1 = minX+(maxX-minX)/2, maxY
    coord2 = maxX, maxY
    coord3 = minX, minY+(maxY-minY)/2
    coord4 = minX+(maxX-minX)/2, minY+(maxY-minY)/2
    coord5 = maxX, minY+(maxY-minY)/2
    coord6 = minX, minY
    coord7 = minX+(maxX-minX)/2, minY
    coord8 = maxX, minY

    ringTopLeft = ogr.Geometry(ogr.wkbLinearRing)
    ringTopLeft.AddPoint_2D(*coord0)
    ringTopLeft.AddPoint_2D(*coord1)
    ringTopLeft.AddPoint_2D(*coord4)
    ringTopLeft.AddPoint_2D(*coord3)
    ringTopLeft.AddPoint_2D(*coord0)
    polyTopLeft = ogr.Geometry(ogr.wkbPolygon)
    polyTopLeft.AddGeometry(ringTopLeft)

    ringTopRight = ogr.Geometry(ogr.wkbLinearRing)
    ringTopRight.AddPoint_2D(*coord1)
    ringTopRight.AddPoint_2D(*coord2)
    ringTopRight.AddPoint_2D(*coord5)
    ringTopRight.AddPoint_2D(*coord4)
    ringTopRight.AddPoint_2D(*coord1)
    polyTopRight = ogr.Geometry(ogr.wkbPolygon)
    polyTopRight.AddGeometry(ringTopRight)

```

```

ringBottomLeft = ogr.Geometry(ogr.wkbLinearRing)
ringBottomLeft.AddPoint_2D(*coord3)
ringBottomLeft.AddPoint_2D(*coord4)
ringBottomLeft.AddPoint_2D(*coord7)
ringBottomLeft.AddPoint_2D(*coord6)
ringBottomLeft.AddPoint_2D(*coord3)
polyBottomLeft = ogr.Geometry(ogr.wkbPolygon)
polyBottomLeft.AddGeometry(ringBottomLeft)

ringBottomRight = ogr.Geometry(ogr.wkbLinearRing)
ringBottomRight.AddPoint_2D(*coord4)
ringBottomRight.AddPoint_2D(*coord5)
ringBottomRight.AddPoint_2D(*coord8)
ringBottomRight.AddPoint_2D(*coord7)
ringBottomRight.AddPoint_2D(*coord4)
polyBottomRight = ogr.Geometry(ogr.wkbPolygon)
polyBottomRight.AddGeometry(ringBottomRight)

# Intersect 4 squares polygons with test polygon
quaterPolyTopLeft = polyTopLeft.Intersection(geom_poly)
quaterPolyTopRight = polyTopRight.Intersection(geom_poly)
quaterPolyBottomLeft = polyBottomLeft.Intersection(geom_poly)
quaterPolyBottomRight = polyBottomRight.Intersection(geom_poly)

# Create centroids of each intersected polygon
centroidTopLeft = quaterPolyTopLeft.Centroid()
centroidTopRight = quaterPolyTopRight.Centroid()
centroidBottomLeft = quaterPolyBottomLeft.Centroid()
centroidBottomRight = quaterPolyBottomRight.Centroid()

return centroidTopLeft,centroidTopRight,centroidBottomLeft,centroidBottomRight

def landing(point_geom):

roadsDs = ogr.Open('OSMroads.osm')
inLayer = roadsDs.GetLayer(1) # layer 1 for ways

# create the input SpatialReference
sourceSR = osr.SpatialReference()
sourceSR.ImportFromEPSG(4326)
# create the output SpatialReference
targetSR = osr.SpatialReference()
targetSR.ImportFromEPSG(26913)
# create transform
coordTrans = osr.CoordinateTransformation(sourceSR,targetSR)

# loop through the input features
inFeature = inLayer.GetNextFeature()
distMin = 9999999
while inFeature:

    # get the input geometry
    road_geom = inFeature.GetGeometryRef()
    # reproject the geometry
    road_geom.Transform(coordTrans)

    for point_road in road_geom.GetPoints():
        point_road_geom = ogr.Geometry(ogr.wkbPoint)
        point_road_geom.AddPoint_2D(point_road[0],point_road[1])
        dist = point_road_geom.Distance(point_geom)
        if dist < distMin:
            distMin = dist
            landing_geom = point_road_geom

```

```

        # destroy the features and get the next input feature
        inFeature = inLayer.GetNextFeature()

    return landing_geom

# Create four stand centroids
centroids = createQuaterCentroids(stand_wkt)

# Downloads OSM road data around stand
downloadOSMroad(stand_wkt)

# Get four landing coordiantes
landing_geom0 = landing(centroids[0])
landing_geom1 = landing(centroids[1])
landing_geom2 = landing(centroids[2])
landing_geom3 = landing(centroids[3])

# array2skidDistList
count = 0
reversed_standArray = standArray[::-1] # reverse array
standPointArray = np.where(reversed_standArray == 1)
SDarray = reversed_standArray.astype(float)

for indexY in standPointArray[0]:
    indexX = standPointArray[1][count]
    Xcoord = x_min+pixel_size*indexX+pixel_size/2
    Ycoord = y_min+pixel_size*indexY+pixel_size/2
    point_geom = ogr.Geometry(ogr.wkbPoint)
    point_geom.AddPoint_2D(Xcoord, Ycoord)
    distList = []
    distList.append(landing_geom0.Distance(point_geom))
    distList.append(landing_geom1.Distance(point_geom))
    distList.append(landing_geom2.Distance(point_geom))
    distList.append(landing_geom3.Distance(point_geom))

    SDarray[indexY,indexX] = (min(distList)*3.28084)

    # # Create array with coordinates
    # source = osr.SpatialReference()
    # source.ImportFromEPSG(26913)
    # target = osr.SpatialReference()
    # target.ImportFromEPSG(4326)
    # transform = osr.CoordinateTransformation(source, target)
    # point.Transform(transform)
    count += 1
return SDarray[SDarray != 0].flatten()

def removeLimits(inputfn, inputfn_NoLimit):
    import csv

    input_data_NoLimit = csv.writer(open(inputfn_NoLimit,'wb'))
    data = ['TPA', 'VPT', 'S', 'SD']
    input_data_NoLimit.writerow(data)

    input_data = csv.reader(open(inputfn))
    next(input_data, None) # skip the headers
    for row in input_data:
        TPA = float(row[0])
        VPT = float(row[1])
        S = float(row[2])
        SD = float(row[3])

        if TPA > 0 and VPT > 0:
            input_data_NoLimit.writerow(row)
        else:

```

```

        print 'LIMIT'
        print TPA, VPT, S, SD

def CostData(inputfn_NoLimit, outputfn):
    '''
    Open CSV input file and reads input data arguments.
    Runs harvest cost function with input arguments.
    Writes input arguments and returned cost to a new CSV file.
    '''

    import CostFunc as CF
    import csv

    cost_data = csv.writer(open(outputfn,'wb'))
    data = ['TPA', 'VPT', 'S', 'SD', 'C']
    cost_data.writerow(data)

    input_data = csv.reader(open(inputfn_NoLimit))
    next(input_data, None) # skip the headers
    for row in input_data:
        TPA = round(float(row[0]), 7)
        VPT = round(float(row[1]), 7)
        S = round(float(row[2]), 7)
        SD = round(float(row[3]), 7)

        C = CF.costfunc(S, SD, TPA, VPT)
        data = [TPA, VPT, S, SD, C]
        cost_data.writerow(data)

if __name__ == '__main__':
    inputfn = 'data/producedData/inputData.csv'
    inputfn_NoLimit = 'data/producedData/inputData_NoLimit.csv'
    outputfn = 'data/producedData/costData_NoLimit.csv'
    standfn = 'data/CSFS/timber_sales.shp'
    slopefn = 'data/CSFS/Slope.tif'
    stand_shp = ogr.Open(standfn)
    stand_lyr = stand_shp.GetLayer()
    pixel_size = 10.0

    TPAarrayALL = np.array([])
    VPTarrayALL = np.array([])
    SDarrayALL = np.array([])
    SarrayALL = np.array([])

    count = 0
    for stand in stand_lyr:
        print "Stand ", count
        count +=1
        stand_geom = stand.GetGeometryRef()
        TPA = stand.GetField("TPA")
        VPT = stand.GetField("VPT")

        stand_wkt = stand_geom.ExportToWkt()

        standArray, x_min, y_max, y_min = stand2standArray(stand_wkt, slopefn,
pixel_size)

        TPAarray = createTPAarray(standArray, TPA)
        print "TPA array created"

        VPTarray = createVPTarray(standArray, VPT)
        print "VPT array created"

        Sarray = createSarray(standArray, slopefn, x_min, y_max)
        print "S array created"

        SDarray = createSDarray(stand_wkt,standArray, pixel_size, x_min, y_max, y_min)

```

```

print "SD array created"

TPAarrayALL = np.concatenate((TPAarrayALL,TPAarray),1)
VPTarrayALL = np.concatenate((VPTarrayALL, VPTarray),1)
SarrayALL = np.concatenate((SarrayALL, Sarray),1)
SDarrayALL = np.concatenate((SDarrayALL, SDarray),1)
print "Arrays concatenated"

output = np.column_stack((TPAarrayALL,VPTarrayALL,SarrayALL,SDarrayALL))
np.savetxt(inputfn,output,delimiter=',')
print "Array writen to CSV"

removeLimits(inputfn, inputfn_NoLimit)
print "Limits removed"

CostData(inputfn_NoLimit,outputfn)
print "Cost Data created"

```

Appendix IV

Script CostRaster.py

```
from osgeo import ogr, gdal, osr
import numpy as np
from math import sqrt

def shp2array(inputfn,baseRasterfn):
    outputfn = 'rasterized.tif'

    source_ds = ogr.Open(inputfn)
    source_layer = source_ds.GetLayer()

    raster = gdal.Open(baseRasterfn)
    geotransform = raster.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    cols = raster.RasterXSize
    rows = raster.RasterYSize

    target_ds = gdal.GetDriverByName('GTiff').Create(outputfn, cols, rows, 1,
gdal.GDT_Byte)
    target_ds.SetGeoTransform((originX, pixelWidth, 0, originY, 0, pixelHeight))
    band = target_ds.GetRasterBand(1)
    NoData_value = 255
    band.SetNoDataValue(NoData_value)
    gdal.RasterizeLayer(target_ds, [1], source_layer, burn_values=[0])

    # Read as array
    array = band.ReadAsArray()
    return array

def pixelOffset2coord(xOffset,yOffset):
    raster = gdal.Open('rasterized.tif')
    geotransform = raster.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    coordX = originX+pixelWidth*xOffset
    coordY = originY+pixelHeight*yOffset
    return coordX, coordY

def roadArray2coordDict(array):
    count = 0
    roadList = np.where(array == 0)
    roadListCoord = []
    for indexY in roadList[0]:
        indexX = roadList[1][count]
        Xcoord, Ycoord = pixelOffset2coord(indexX,indexY)
        coords = (Xcoord, Ycoord)
        roadListCoord.append(coords)
        count += 1
    return roadListCoord

def distance(coord0X,coord0Y,coord1):
    return sqrt((coord0X-coord1[0])**2+(coord0Y-coord1[1])**2)

def nonRoadArray2coord(array, roadListCoord):
    distArray = (np.copy(array)).astype(float)
    count = 0
    nonRoadList = np.where(array != 0)
    total = len(nonRoadList[0])
    for indexY in nonRoadList[0]:
        indexX = nonRoadList[1][count]
        nonRoadXcoord, nonRoadYcoord = pixelOffset2coord(indexX,indexY)
```



```

        minDist = min([distance(nonRoadXcoord,nonRoadYcoord,roadPoint) for roadPoint in
roadListCoord])
        distArray[indexY,indexX] = minDist*3.28084
        count += 1
        print str(count) + "of" + str(total)
    return distArray

def raster2array(rasterfn):
    raster = gdal.Open(rasterfn)
    band = raster.GetRasterBand(1)
    array = band.ReadAsArray()
    return array

def array2raster(newRasterfn,distArray):
    raster = gdal.Open('rasterized.tif')
    geotransform = raster.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    cols = raster.RasterXSize
    rows = raster.RasterYSize

    driver = gdal.GetDriverByName('GTiff')
    outRaster = driver.Create(newRasterfn, cols, rows, 1, gdal.GDT_Float32)
    outRaster.SetGeoTransform((originX, pixelWidth, 0, originY, 0, pixelHeight))
    outband = outRaster.GetRasterBand(1)
    outband.WriteArray(distArray)
    outRasterSRS = osr.SpatialReference()
    outRasterSRS.ImportFromEPSG(26913)
    outRaster.SetProjection(outRasterSRS.ExportToWkt())

if __name__ == '__main__':

    roads_fn = 'data/CSFS/roads.shp'
    slope_fn = 'data/CSFS/slope_south.tif'
    newRasterfn = 'CostSurface_south.tif'

    roadArray = shp2array(roads_fn, slope_fn)
    print "road array generated"
    roadListCoord = roadArray2coordDict(roadArray)
    print "road list coord generated"
    skidDistArray = nonRoadArray2coord(roadArray,roadListCoord)
    print "skid dist array generated"
    slopeArray = raster2array(slope_fn)
    print "slope array generated"

    costArray = 22.8038 + slopeArray*0.3272 + skidDistArray*0.007578
    print "cost array generated"

    array2raster(newRasterfn,costArray)

```

Appendix V

Colorado State Forest Harvest Cost Results

ID	TPA	VPT	Slope	SD	Full Model	Full Equation	Spatial Equation
1	460.4302	6.0021	19.5747	982.1135	47.3699	45.1708	36.6928
2	249.0462	10.0159	8.1138	1273.0037	34.5947	33.151	35.093
3	134.7744	13.9785	15.2379	617.8667	26.4947	25.2071	32.5177
4	465.8444	4.0354	26.9939	267.516	55.4851	53.9459	33.7675
5	156.8697	21.7927	35.3757	341.4081	21.5972	23.5003	37.0946
6	210.1354	13.7892	20.1087	324.2566	24.267	24.4518	31.9183
7	357.8478	8.0028	12.2345	1658.2219	42.6641	41.4805	39.3545
8	219.535	13.9574	1.4765	442.0764	22.3381	19.3702	26.6452
9	367.1487	9.0445	16.0848	2039.0886	44.269	42.963	43.4938
10	328.2211	6.071	23.1076	396.646	42.5173	41.9945	33.4545
11	268.7183	8.9891	29.995	1561.5041	46.414	44.1129	44.4982
12	240.9274	13.0258	17.1856	583.5776	27.0456	26.2781	32.9036
13	329.8612	8.9679	25.9175	1065.6616	40.3035	38.9588	39.4185
14	366.2662	9.9897	17.6739	1190.7813	36.4473	35.1409	37.6348
15	310.3833	9.9573	19.7382	676.1159	32.6863	32.1475	34.4439
16	205.0352	14.9937	11.8569	950.1592	26.827	25.405	33.9008
17	268.9091	10.012	19.7002	1628.8996	40.771	39.3531	41.602
18	284.1831	11.0182	9.3434	1268.0393	32.9281	31.6492	35.4622
19	275.2289	12.9752	7.673	1550.5676	31.9889	30.5743	37.0363
20	231.313	12.0606	7.7653	933.0576	28.778	27.2748	32.4194
21	213.1377	11.9937	14.5148	1877.2061	37.6894	36.6287	41.7564
22	290.0898	9.0178	18.397	867.1463	36.2387	35.1616	35.4382
23	249.6302	12.0393	9.2603	810.4319	28.2535	26.7803	31.9908
24	369.5147	10.9977	15.9406	3127.0334	47.6201	47.4452	51.634
25	340.3085	14.0709	16.9818	2319.9239	38.3913	37.7876	45.9039
26	389.1626	13.0006	12.0124	1564.9038	32.8474	31.6106	38.5788
27	395.6521	13.1869	14.6395	912.2441	28.723	27.2585	34.5354
28	467.515	6.0087	1.4163	416.282	36.4561	35.2448	26.4312
29	466.7978	6.1219	3.7758	320.2846	35.3682	34.7669	26.4888
30	462.3807	6.0034	2.6832	841.3395	40.2762	38.88	30.049
31	368.3827	8.8897	17.7637	1007.3935	37.4023	36.0485	36.2843
32	377.6028	8.955	8.7725	466.7301	29.8899	29.0177	29.2428
33	378.7436	9.0416	9.4032	212.4343	27.5798	27.0992	27.5375
34	377.5154	9.018	11.9678	547.2131	31.5073	30.4709	30.9049
35	365.087	10.2362	19.892	1663.7448	40.1586	38.9423	41.9276
36	375.6711	8.0542	16.9706	2418.3021	49.7651	48.4741	46.6406
37	264.4257	14.7383	16.017	2229.2954	36.7182	36.3793	44.9029
38	284.8405	12.7595	5.6513	2905.4813	40.3443	40.3951	46.565
39	298.1482	14.1431	6.1054	3090.1121	39.8123	40.2977	48.1046
40	272.6465	15.0177	24.914	2616.0733	42.4105	41.7451	50.7551
41	318.8978	12.9316	11.5865	1969.6663	35.9629	34.8488	41.4842

42	360.0588	12.8798	20.9698	3009.2692	46.1904	45.5083	52.4104
43	164.6568	10.0072	20.0355	2807.9652	50.416	48.7014	50.5865
44	418.7322	8.8327	28.8557	709.0142	37.1972	37.1876	37.7057
45	424.6443	7.2216	40.5827	904.8995	48.4367	46.6379	43.0569
46	465.0691	8.1892	43.1524	1008.5612	47.0112	45.2805	44.6866
47	303.2732	8.9141	18.483	2601.6283	49.8831	48.4492	48.5203
48	436.2883	8.0035	25.5152	2023.2858	50.8914	48.0699	46.4926
49	458.9958	10.2234	14.4782	1861.0478	40.0056	38.4657	41.6227
50	337.6554	13.7931	20.6337	2399.4014	40.657	39.8231	47.7094
51	377.2391	10.06	17.8586	1786.3063	40.9827	39.5188	42.1778
52	147.8706	15.794	33.6006	193.6608	23.6962	25.8797	35.3959
53	155.0838	21.19	38.2203	222.4634	21.0854	23.8051	37.1399
54	163.7545	19.9884	33.3269	128.3577	19.8648	22.2331	34.8139
55	203.5498	11.9849	33.1437	362.4282	29.697	31.0106	36.515
56	233.3125	9.004	34.6886	554.9203	37.5025	38.0615	38.4744
57	218.5937	8.1508	34.472	870.9738	44.7743	42.4878	40.7814
58	227.6901	9.024	31.0235	900.8206	40.4221	39.5109	39.866
59	370.0029	8.6694	28.4167	203.7875	31.6439	33.7884	33.7583
60	250.7138	17.2409	6.847	275.9937	18.9835	16.7139	27.1708
61	615.5792	9.9626	19.1	828.5485	33.7299	32.0675	35.3801
62	358.7774	16.102	15.5778	235.8327	20.5451	19.6588	29.7549
63	159.1713	17.1818	14.3966	348.4863	21.1689	19.9437	30.2122
64	242.5258	17.8642	8.1197	96.8224	17.2353	15.3259	26.2431
65	231.2231	13.7364	12.3181	198.5613	22.143	21.0854	28.3967
66	117.4448	26.6085	18.7192	236.3785	16.6285	15.6038	30.7975
67	248.5987	19.2803	12.8925	1366.2114	26.565	25.3987	37.3744
68	253.7426	14.874	17.9682	1298.4068	30.9036	29.8695	38.5421
69	259.6056	18.3695	18.129	4224.4505	45.8264	49.1181	60.6167
70	218.0891	18.2884	11.1505	3025.3654	37.6017	38.1194	49.2853
71	195.7548	14.1239	17.5114	3474.5579	46.3793	47.0815	54.7688
72	284.3003	12.3205	22.3098	4481.6502	57.1009	57.986	63.9345
73	564.1467	6.938	17.0042	3679.7857	61.581	60.7428	56.1457
74	386.9017	11.9356	24.7562	4008.2231	56.165	55.3526	61.1803

Appendix VI

index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset=utf-8 />
  <meta name='viewport' content='width=device-width, initial-scale=1.0, maximum-
scale=1.0, user-scalable=no' />

  <!-- Title -->
  <title>CSFS Harvest Costs</title>

  <!-- Stylesheet -->
  <link rel='stylesheet' href='static/style/style.css' />

  <!-- Libraries -->
  <script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
  <script src="./static/libraries/simple-expand.min.js"></script>

  <script src="./static/libraries/leaflet.js"></script>
  <link rel='stylesheet' href="./static/libraries/leaflet.css" />

  <script src="./static/libraries/leaflet.draw.js"></script>
  <link rel='stylesheet' href="./static/libraries/leaflet.draw.css" />

  <script src="./static/libraries/terraformer.min.js" type="text/javascript"></script>
  <script src="./static/libraries/terraformer-wkt-parser.min.js"
type="text/javascript"></script>

  <!-- Data -->
  <script src="/static/data/csf.geojson"></script>
</head>

<body>
  <div id="container">
    <div id="map"></div>
    <div id="infoSign" class="the-box arrow_box">Start digitizing a stand</div>
    <div id="infoContainer">

      <h3>CSFS Harvest Costs</h3>
      <a style="color:#636363;text-decoration:
none;position:absolute;top:20px;right:25px"
href="mailto:ustroetz@gmail.com?Subject=CSFS%20Harvest%20Costs">Contact</a>
    <div id='legend'>
      <h4>Harvest Cost ($/ton) </h4>

      <nav class='legend clearfix'>

        <span style='background:#1a9641;'></span>
        <span style='background:#c3e586;'></span>
        <span style='background:#fdc980;'></span>
        <span style='background:#d7191c;'></span>
        <label>< 30.0 </label>
        <label>30.0 - 39.9 </label>
        <label>40.0 - 49.9 </label>
        <label>>= 50.0</label>

      </div>

      <div id="estimate">
        <h4 id="text_estimated_cost"></h4>
        <table style="margin-left:30px">
          <tr>
            <td style="width:143px">Cost</td>
            <td id="estimated_cost"></td>
          </tr>
        </table>
      </div>
    </div>
  </div>
</body>
</html>
```

```

        </table>
        <br>
        <button onclick="this.style.visibility = 'hidden';" id="expander"
href="#">&#9660; Detailed Calculation</a>
    </div>

    <div class="content">
        <h4>Harvest Cost Calculation</h4>
        <table style="margin-left:30px">
            <tr>
                <td style="width:143px">Trees per Acre</td>
                <td>
                    <input type="range" min="10" max="1000" step="10"
value="300" onchange="TPA.value=value"></input>
                    <output id="TPA">300</output>
                </td>
            </tr>
            <tr>
                <td style="width:143px">Volume per Acre</td>
                <td>
                    <input type="range" min="1000" max="10000" step="100"
value="4000" onchange="VPA.value=value"></input>
                    <output id="VPA">4000</output>
                    <output>ft3</output>
                </td>
            </tr>
        </table>
        <br/>
        <button id="calculate" onclick="calculate();"
style="display:block;">Calculate</button>
        <table id="spatial_var" style="margin-left:30px"></table>
        <br/>
        <button id="recalculate" onclick="calculate();"
style="display:none;">Recalculate</button>
    </div>
</div>

    <script src="/static/scripts/site.js"></script>
</body>
</html>

```

style.css

```
html, body, #map, #container {
  height: 100%;
  width: 100%;
  margin: 0px;
  padding: 0px;
  font-family: "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;
  font-size: 15px;
  color: #666666;
}

h3 {
  font-size: 20px;
  font-weight: 200;
  color: #666666;
}

h4 {
  color: rgb(45, 45, 45);
  font-size: 16px;
  font-weight: bold;
  color: #666666;
  margin: 10px 0px 10px 10px;
}

#infoContainer {
  position: absolute;
  padding: 20px;
  background-color: rgba(255, 255, 255, 1);
  top: 10px;
  right: 10px;
  height: 120px;
  width: 400px;
}

#estimate {
  position: absolute;
  top: 160px;
  right: 0px;
  padding: 0px 20px 20px 20px;
  background-color: rgba(255, 255, 255, 1);
  height: 80pt;
  width: 400px;
  visibility: hidden;
}

.content {
  position: absolute;
  top: 230px;
  right: 0px;
  padding: 0px 20px 20px 20px;
  background-color: rgba(255, 255, 255, 1);
  height: 200pt;
  width: 400px;
}

.advanced {
  cursor: pointer;
  font-size: 10px;
}

p {
  padding: 5px 0;
}

button {
  background-color: #80c757;
  text-align: center;
  color: #fff;
}
```

```

    display: inline-block;
    height: 40px;
    width: 150px;
    margin-left: 140px;
    padding: 10px;
    border: none;
    cursor: pointer;
    border-radius: 3px;
    white-space: nowrap;
    text-overflow: ellipsis;
    font-family: 'Open Sans Bold', sans-serif;
    line-height: 20px;
    font-size: 12px;
    text-decoration: none;
}

button:hover {
    background-color: #598b3c;
}

.the-box {
    background: #fabfab;
    padding: 10px;
    width: 200px;
    height: 50px;
    line-height: 50px;
    text-align: center;
}

.arrow_box {
    position: absolute;
    top: 87px;
    left: 74px;
    height: 50px;
    background: rgba(255, 255, 255, 0.9);
}

.arrow_box:after, .arrow_box:before {
    right: 100%;
    top: 50%;
    border: solid transparent;
    content: " ";
    height: 0;
    width: 0;
    position: absolute;
    pointer-events: none;
}

.arrow_box:before {
    border-right-color: rgba(255, 255, 255, 0.9);
    border-width: 20px;
    margin-top: -20px;
}

.legend {
    padding-top: 5px;
}

.legend label,
.legend span {
    display: block;
    float: left;
    height: 15px;
    width: 25%;
    text-align: center;
    font-size: 9px;
    color: #808080;
}

```

site.js

```
// Create map elements
var map = L.map('map').setView([40.545, -105.965], 14);

var csf = L.geoJson(csf, {
  style: {
    opacity: 1.0,
    fill: 0.0,
    color: '#FFF'
  }
});
var satelliteTileLayer = L.tileLayer('https://{s}.tiles.mapbox.com/v3/tmcw.map-
j5fsp01s/{z}/{x}/{y}.png');
var terrainTileLayer = L.tileLayer('https://{s}.tiles.mapbox.com/v3/tmcw.map-
7s15q36b/{z}/{x}/{y}.png');
var costSurfaceTileLayer =
L.tileLayer('https://{s}.tiles.mapbox.com/v3/csfsfc.h/{z}/{x}/{y}.png', {
  opacity: 0.5
});

baseLayers = {
  "Satellite": satelliteTileLayer,
  "Terrain": terrainTileLayer
};

overlays = {
  'Cost Surface': costSurfaceTileLayer,
};

var layerControl = L.control.layers(baseLayers, overlays, {
  position: 'topleft'
});

var drawnItems = new L.FeatureGroup();

var drawControl = new L.Control.Draw({
  draw: {
    polyline: false,
    rectangle: false,
    circle: false,
    marker: false,
    polygon: {
      shapeOptions: {
        color: '#ffffff'
      }
    }
  },
  edit: {
    featureGroup: drawnItems
  }
});

terrainTileLayer.addTo(map);
costSurfaceTileLayer.addTo(map);
csf.addTo(map);
layerControl.addTo(map);
drawControl.addTo(map);
drawnItems.addTo(map)

// Create event listener
map.on('draw:edited', function(e) {
  // after stand is edited: recalcualte estimate, remove detailed calculation

  drawnItems.eachLayer(function(layer) {
    getEstimate(layer);
  });
});
```



```

    if ($('#spatial_var tbody').children().length != 0) {
        $('#spatial_var').empty();
        $('#calculate').css("display", "block");
        $('#recalculate').css("display", "none");
    };
});

map.on('draw:drawstart', function(e) {
    map.removeLayer(terrainTileLayer);
    satelliteTileLayer.addTo(map);

    // after stand drawing started: remove info start box, empty cost estimate, remove
    previous stand layers, remove detailed calculation

    $('#infoSign').remove();

    $('#estimated_cost').empty();
    drawnItems.eachLayer(function(layer) {
        map.removeLayer(layer)
    });

    if ($('#spatial_var tbody').children().length != 0) {
        $('#spatial_var').empty();
        $('#calculate').css("display", "block");
        $('#recalculate').css("display", "none");
    };
});

map.on('draw:created', function(e) {
    // after stand is drawn: calculate estimate

    getEstimate(e.layer)
});

var getEstimate = function(layer) {

    // add layer to map and add 'No Data' default to info box
    drawnItems.addLayer(layer);
    document.getElementById('estimated_cost').innerHTML = 'No Data';

    // convert layer to wkt
    var standGeojson = layer.toGeoJSON();
    var standWKT = Terraformer.WKT.convert(standGeojson.geometry);

    // send wkt to Python App to get cost estimate
    $(function() {
        $.getJSON('/_estimatedCost', {
            data: standWKT
        },
        function(data) {
            $(data.result);

            // Update cost estimate in info box
            var cost = data.result + ' $/ton';
            document.getElementById('text_estimated_cost').innerHTML = 'Harvest Cost
Estimate';

            document.getElementById('estimated_cost').innerHTML = cost;
            document.getElementById('estimate').style.visibility = 'visible';
        });
    });
};

var calculate = function() {

    // Get Input Variables

```

```

var TPA = document.getElementById('TPA').value;
if (!TPA) {
    TPA = 300
} else {
    TPA = parseFloat(TPA);
};

var VPA = document.getElementById('VPA').value;
if (!VPA) {
    VPA = 4000
} else {
    VPA = parseFloat(VPA);
};

var SD = document.getElementById('SD');
if (!SD) {
    SD = null
} else {
    SD = parseFloat(SD.value);
};

var S = document.getElementById('S');
if (!S) {
    S = null
} else {
    S = parseFloat(S.value);
};

// get stand layer
var standWKT;
if ((drawnItems.getLayers().length) != 0) {
    drawnItems.eachLayer(function(layer) {
        standWKT = Terraformer.WKT.convert((layer.toGeoJSON()).geometry);
    });

    var harvestData = {
        TPA: TPA,
        VPA: VPA,
        SD: SD,
        S: S,
        stand_wkt: standWKT
    };
    var harvestDataStr = JSON.stringify(harvestData);

    // send harvest data string to Python App to get cost calculation
    $(function() {
        $.getJSON('/_calculatedCost', {
            harvest_Data: harvestDataStr
        },

        function(data) {
            $(data.result);

            // Update spatial variables and detailed cost
            var tableSpatialData = document.getElementById('spatial_var');

            if ($('#spatial_var tbody').children().length == 0) {
                // Skidding Distance
                var row = tableSpatialData.insertRow(0);
                var cellSD = row.insertCell(0);
                var cellSDvalue = row.insertCell(1);
                var SDvalue = String(parseFloat(data.result[1]));
                cellSD.innerHTML = "Skidding Distance";
                cellSDvalue.innerHTML = "<input type='range' min='0' max='10000'
step='100' value='" + SDvalue + "' onchange='SD.value=value'></input><output id='SD'>" +
SDvalue + "</output><output> ft</output></td>";
                // Slope
                var row = tableSpatialData.insertRow(1);
                var cells = row.insertCell(0);

```

```

        var cellSvalue = row.insertCell(1);
        var Svalue = String(parseFloat(data.result[0]));
        cellS.style.width = "143px";
        cellS.innerHTML = "Slope";
        cellSvalue.innerHTML = "<input type='range' min='0' max='40'
step='1' value='" + Svalue + "' onchange='S.value=value'></input><output id='S'>" +
Svalue + "</output><output> %</output></td>";
        // Cost
        var row = tableSpatialData.insertRow(2);
        var cell1 = row.insertCell(0);
        cell1.innerHTML = "&nbsp;";
        var row = tableSpatialData.insertRow(3);
        var cellC = row.insertCell(0);
        var cellCvalue = row.insertCell(1);
        cellCvalue.innerHTML = String(parseFloat(data.result[2])) + "
$/ton";

        cellC.innerHTML = "Cost";
        // Buttons
        document.getElementById("calculate").style.display = "none";
        document.getElementById("recalculate").style.display = "block";
    }

    // Recalculate
    else {
        document.getElementById("SD").value =
parseFloat(data.result[1]);
        document.getElementById("S").value = parseFloat(data.result[0]);
        tableSpatialData.rows[3].cells[1].innerHTML =
String(parseFloat(data.result[2])) + " $/ton";
    };
    });
} else {
    alert('Please digitize a stand first!')
};
};

// Expand info box
$('#expander').simpleexpand();

```

Appendix VII

run.py

```
from flask import Flask, render_template, request, jsonify
import estimatedCost as eCost
import calculatedCost as cCost
import json

app = Flask(__name__)
app.config['DEBUG'] = True

@app.route('/')

@app.route('/index')
def index():
    return render_template("index.html")

@app.route('/_estimatedCost')
def get_cost_estimate():
    standWKT = request.args.get('data')
    costStats = eCost.get_zonal_stats(standWKT)
    cost = round(costStats[0]['mean'],2)
    return jsonify(result = cost)

@app.route('/_calculatedCost')
def get_cost_detailed():
    harvestData = json.loads(request.args.get('harvest_Data'))
    slope, SkidDist, harvestCostTon, totalHarvestCost =
cCost.cost_func(str(harvestData['stand_wkt']), harvestData['TPA'], harvestData['VPA'],
harvestData['SD'], harvestData['S'])
    totalHarvestCost = round(totalHarvestCost,2)
    harvestCostTon = round(harvestCostTon,2)
    SkidDist = round(SkidDist,2)
    slope = round(slope,2)
    resultData = [slope, SkidDist, harvestCostTon, totalHarvestCost]
    return jsonify(result = resultData)

if __name__ == "__main__":
    app.run()
```