12-2023

Evaluation of Vegetative Indices to Determine Canopy Ground Cover for Winter Survival and Hybrid Necrosis in Winter Wheat

Micheal Young
University of Nebraska-Lincoln

Evaluation of Vegetative Indices to Determine Canopy Ground Cover for Winter Survival and Hybrid Necrosis in Winter Wheat

by

Micheal Young

A THESIS

Presented to the Faculty of

The graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Agronomy

Under the Supervision of Professor Stephen Baenziger

Lincoln, Nebraska

December 2023

Evaluation of Vegetative Indices to Determine Canopy Ground Cover for Winter Survival and Hybrid Necrosis in Winter Wheat

Micheal Young, M.S.

University of Nebraska, 2023

Advisor: Stephen Baenziger

The benefit of unmanned aircraft systems and image processing methods in agronomic research across numerous crops has been well documented as has the importance of wheat, *Triticum aestivum* L., on the global food supply. Hence there is great interest in digital solutions applied to aspects of wheat breeding. A major trait of importance to winter wheat breeders in higher latitudes is winter survival, which can result in poor yield and performance if lines do not survive extreme cold. Scoring winter survival is most commonly based on visual score of 0% to 100% with the higher percentage conveying higher winter survival rates. With the increased interest in hybrid wheat lines, it has brought an increased need to screen for hybrid necrotic lines in the field. With both hybrid necrosis and winter kill reducing the stand count of a plot, the advantageous situation arose to be able to investigate digital solutions of measuring wheat stand and their relationship with winter survival and hybrid necrosis. We were able to show that the utilization of multiple vegetative indices and segmentation indices derived from multispectral imagery within the same linear model was able to predict stand with a correlation of $r = 0.836$ ($p < 0.01$; flight date 5/18/2020) to visually scored plot stand data. Using unnormalized RGB model utilizing segmentation indices (an index that is used to separate vegetative pixels from background pixels) was able to achieve a prediction with a correlation of $r = 0.924$ ($p < 0.01$; flight date 5/18/2020) with the visually scored plot stand data, lending to the potential use of segmentation in conjunction with processes where RGB images are not normalized. In lines exhibiting symptoms of hybrid necrosis a clear clustering pattern could be observed as the season progressed for NDVI values of hybrid necrotic lines compared to non-necrotic lines.

Table of Contents

Multimedia Objects

Tables

Figures

Chapter 1: Introduction

1.1 Wheat:

Common wheat (*Triticum aestivum* L.) has been a pivotal crop for the past 10,000 years, originating from a cross between emmer (a tetraploid wheat 2n=4x=28 chromosomes) with wild goat grass, *Aegilops tauschii* Coss. (2n=2x=14; Feldman, 2001). Throughout the centuries wheat has served as a major source of carbohydrates and proteins for the globe at one point providing 55% of the caloric intake (Gustafson, 2009). Historically the improvement of wheat cultivars was accomplished through exploration and exchange of seed allowing the natural selection pressures of new environments to produce new land races (Baenziger, 2009). It was within the past two centuries that wheat cultivars have been developed through controlled hybridization and artificial selection (Baenziger, 2009). It is through these methods that plant breeders have been able to increase wheat productivity in the world with ever increasing demand.

According to the United States Department of Agriculture the global wheat production in 2023 is 781,980,000 metric tons and in the United States alone 46.7 million acres of wheat were planted during 2021. The Food and Agriculture Organization projects global population to reach 9.7 billion by 2050 and that global food production will need to increase by 70%, further increasing the importance of the development of better more productive wheat cultivars.

1.2 Winterkill:

A major aspect of developing better more productive winter wheat cultivars is overcoming challenges presented by harsh environments such as winter kill. Winter wheat being a fall sown crop requires an overwintering or vernalization period to induce flowering allowing for grain to be produced. Due to the vernalization requirement of winter wheat, it can be subject to winterkill in higher latitudes resulting in a significant decrease in yield (Alessi and Power, 1971). Winterkill is the result of cell damage from ice forming within the wheat cells (Lyons et al. 1979). It has been shown that winterkill is caused in varying degrees by both extreme freezing events below -20º C (Taylor and Olsen, 1985) and prolonged mild freezing events below -4º C (Gusta et al. 1997; Roberts, 1985). This understanding of winter kill has been further described through the estimation of freezing degree days, showing a 1% increase in mortality for every 1º C·d increase in freezing degree days (Zheng et al. 2018). In addition to freezing temperature, desiccation has also been shown to have a negative impact on winter survival resulting in differential winterkill of winter hardened cultivars under conditions that did not pass the critical temperature threshold of -20º C (Taylor and Olsen, 1985).

Winterkill can be mitigated through a number of management practices. No-till practices leave residue behind that holds more snow providing better insulation from extreme temperatures (Cox et al. 1986). Planting date (Fowler, 1982), nutrient applications (Pittman and Tipples, 1978) and planting depth (Loeppky et al, 1989) have all shown a positive influence on winter survival in addition to cultivar selection.

The most commonly practiced method of rating wheat lines for winter kill is through the visual method where 0% represents no plants surviving in a plot of winter wheat and 100% representing all plants in a plot surviving the winter (Saulescu and

Braun, 2001). It is through this visual scale the wheat breeders have evaluated breeding populations to select lines that have improved winter hardiness and survival to provide to producers. There have been other methods proposed to rate lines for winter hardiness such as in controlled freezing experiments where the temperature that results in the death of 50% (lethal temperature 50, LT50) of the wheat plants is recorded. LT50 had a high negative correlation of 95% (r = 0.95) with visual ratings of winter survival (Pomeroy and Fowler, 1973). While the LT50 has been shown to correlate strongly with crown freezing tolerance (Brule-Babel and Fowler, 1981) and crown freezing tolerance is accepted as a prerequisite of winter hardiness (Brule-Babel and Fowler, 1989), the LT50 has not been able to prove itself to be a reliable method to measure winter hardiness across a variety of different environments (Bridger et al. 1996; Gusta et al. 2001).

In addition to visual scoring, the use of vegetative indices of ground level imagery has been used to measure winter survival with correlation of 95% (r = 0.95; Chen et al. 2019). The potential to use objective image-based analysis provides the opportunity to alleviate issues with the visual scoring method resulting from human error, bias and limitations in being able to identifying small differences between lines (Poland and Nelson 2011). Image based analysis can also lead to reduced resource costs associated with needing skilled labor to walk the field. It is through on-going research like using a vegetative index to help select for cultivars with better winter hardiness that wheat breeders will be able to consistently provide yield protection from environmental stresses like harsh winters.

1.3 Hybrid Wheat

In addition to protecting from yield loss in harsh environments, hybrid wheat cultivars also have the potential to increase overall production. Hybrid wheat is a category of F_1 wheat cultivars that are developed through a specific cross between two inbred parents that have been identified as good combiners. The goal of hybrids is to take advantage of the phenomenon known as heterosis or hybrid vigor, where the performance of the hybrid exceeds the performance of parental lines used to create it (Koemel et al., 2004). Hybrid durum cultivars have been found to produce up to a 20% increase in yield performance over the highest yielding inbred lines (Gowda et al., 2012). Hybrids also exhibited increased early vigor and height without an increased susceptibility to lodging in addition to increased yield (Longin et al., 2013). In addition to increased trait performance hybrids are expected to have greater stability in performance across a larger breadth of environments (Gowda et al., 2010).

Hybrid wheat showed some early success in commercialization in the United States, Australia, and Europe during the early 1990's (Gupta et al., 2019) however the success was short lived as competition from improved pure lines and the hybrid seed production costs and difficulties, overcame the profitability of hybrids. The difficulty of producing hybrid seed arises from the inflorescence of wheat. Wheat produces spikelets of perfect flowers along a single rachis (Gao et al., 2019) that contain both the male and female structures in tightly packed florets. Due to the enclosed nature of wheat florets, wheat is a predominately self-pollinating crop with a very low incidence of natural cross pollination. To overcome the issue of self-pollination two primary approaches have been explored, the use of chemical hybridizing agents (CHA) and the development of male sterile lines (Gupta et al., 2019). It was through the use of CHA that hybrids first reached

commercialization in the 1990's. Specifically CHA's Genesis and Crosier were seen to produce a sterilization rate of 95-100% (Gupta et al., 2019). Even though hybrids were gaining popularity, they were unable to overcome the production costs and regulatory limitations of the CHA. As commercialization moved away from CHA derived hybrids, hybrids derived from cytoplasmic or genetic male sterile systems became the forefront of hybrid research. The most likely candidate for commercialization are the cytoplasmic male sterile (CMS) system, a three line system with sterile line (A-line; in a sterile cytoplasm and without genes to restore fertility), a maintainer (B-line; an alloplasmic line of the A-line with a fertile cytoplasm) and male restorer (R-line) where the R-line contains restoration genes that allow full restoration of the F_1 hybrid between the A-line and the R-line. The most commonly used CMS system and most reliable uses *T. timopheevii* Zhuk (Mukai and Tsunewaki, 1979; Singh et al., 2010).

Other male sterile systems have also been explored but have not garnered the same popularity as the three-line *timopheevii* CMS system. Photoperiod sensitive cytoplasmic male sterility is a two-line system based in *Ae. Crassa* Boiss. Ex Hohen cytoplasm where the line is sterile under long day conditions and fertile under short day conditions and an insensitive R-line that would restore fertility to the F_1 under long day conditions (Murai and Tsunewaki, 1993). Genetic male sterility is another two-line system that utilizes a dominant male fertile genes and recessive sterile genes. The difficulty with system is the sterility line is maintained in heterogenous state requiring culling of fertile females (Singh et al., 2015). Chromosomal male sterility utilizes a three-line system to develop the sterile female and maintainer lines, involving a deletion on the 4B chromosome that contains the recessive sterile gene. The male line that will be used

for the F_1 seed production must contain an alien 4E chromosome that carries the restoration gene. The attraction of this system is the presence of the blue aleurone gene allowing for the sorting of sterile and self-fertile lines based on color (Zhou et al., 2006; Whitford et al., 2013).

The biggest inhibitor of hybrid seed production is the cost of maintaining the male and female lines in relatively small strips that prevent the use of large combines or require additional resources to shred the male lines. One proposed method to overcome this is the use of the transgenic SeedLink system where the barnase gene, which provides glufosinate resistance, is linked to a sterility gene allowing for males and females to be planted in a mix and the males to be removed via glufosinate application, this method is used in both the barnase-barstar system in canola (Whitford et al., 2013) and the split barnase system (Kempe et al., 2014). Though not an exhaustive list, the exploration of numerous male sterile systems by researchers around the globe has provided great potential for the commercially viable production of hybrid wheat.

1.4 Hybrid Necrosis

A hurdle faced in the production of wheat hybrids is the loss of potential parental lines due to those same parents carrying genes for hybrid necrosis. Hybrid necrosis is a phenomenon observed in F_1 wheat plants where the leaf tissue and sheath tissue become necrotic to a lethal or semi lethal extant (Caldwell, 1943; Tsunewaki, 1992; Tomar et al., 1991). Hybrid necrosis is caused by the combined presence of two dominant genes *Ne1*, located on chromosome arm 5BL and *Ne2*, located on chromosome arm 2BS (Zeven,

1972; Nishikawa, 1974). Hybrid necrosis was first characterized as the delayed necrosis of leaf tissue first appearing only after the first leaf has reached physiological maturity and the second leaf is well grown (Caldwell, 1943). This expression of necrosis has proven to be barrier to introduction of new genes to breeding populations and combining desirable genes within those populations (Bizimungu et al., 1998) due to both genes being widespread throughout breeding populations across the globe (Tsunewaki, 1992). The frequency of the hybrid necrosis genes provides an additional challenge to hybrid programs, as inbred programs discover hybrid necrotic combinations at the early crossing stage of line development while a hybrid program may not discover a necrotic combination until two parental lines have progressed much further into the breeding pipeline consuming substantially more resources. This potential resource loss exacerbates the need for hybrid programs to track the frequency of necrosis genes in their breeding populations and provides the basis of the need for efficient methods to accurately classify hybrids exhibiting hybrid necrosis with vegetative indices derived from aerial imagery.

1.5 Vegetative Indices

It has been widely demonstrated that aerial imagery coupled with image processing has the potential to provide a high throughput and accurate measurement of numerous traits across all crops. One of the most commonly used methods to measure traits that can be associated with chlorophyl content is to extract the normalized difference vegetative index (NDVI) (Tucker, 1979; Roujean and Breon, 1995) from a multispectral image containing the bands green, red and near infrared (NIR) (Stanton et al., 2017; Shafian et al., 2018). NDVI was shown to have a strong relationship with visual

ground cover ($r^2 = 0.93$) and leaf area index ($r^2 = 0.95$) in winter wheat (Shi et al., 2016). Other indices that have been developed to capture similar traits associated with chlorophyl content such as the renormalized difference vegetation index (RDVI) utilizing the red and NIR bands providing a benefit of reduced sensitivity to soil reflectance (Roujean and Breon, 1995; Li et al., 2018). Green normalized difference vegetation index (GNDVI) utilizing the NIR and green bands, provides a better correlation to Chlorophyll-a (Gitelson et al., 1996; Li et al., 2018). Normalized difference red edge index (NDRE) utilizing the NIR and red edge bands, also had a strong correlation with chlorophyll and leaf nitrogen status (Fitzgerald et al., 2006; Li et al., 2018).

It has also been shown among some traits that indices derived from imagery captured in the visual spectrum with commercial true color (RGB) cameras have a similar correlation as those captured from multispectral cameras. For instance the excessive green segmentation index (ExG), which utilizes the red, green and blue bands to delimitate soil from vegetation (Woebbecke et al., 1995), had similar correlation ($r = 0.88$) with ground cover in spring wheat as NDVI ($r = 0.76$) (Rasmussen et al., 2015). The excessive green minus excessive red segmentation index (ExGR) showed improved capabilities to distinguish between vegetative matter and soil background over ExG (Neto, 2004). The color index of vegetation extraction (CIVE) was derived from the RGB color bands to segment vegetative matter from soil background and was shown to have a good correlation ($r = 0.661$) with soybean, *Glycine max*, biomass (Kataoka et al., 2003). ExG, ExGR and CIVE have all shown to be an acceptable method of segmentation of vegetation from clean soil under both cloudy and sunny conditions as was exhibited in images of corn, *Zea mays*, on various soil types (Yang et al., 2015). These segmentation

indices (ExG, ExGR, CIVE) are calculated by determining a cutoff value or threshold that indicates what pixels to classify as soil and as vegetation, the total number of vegetation pixels are then counted to determine the plot value of the segmentation index, also referred to as vegetative fraction (Gitelson et al., 2002). The red green blue vegetative index (RGBVI) is also derived from the RGB bands and has shown good ability to be used in biomass predictions ($r^2 = 0.82$) (Bendig et al., 2015), and good correlation with canopy cover ($r = 0.75$) (Li et al., 2018). An advantage of many of the mentioned indices is the ability to calculate all of them from the five bands present in many multispectral cameras (red, green, blue, near infrared and red edge), allowing for numerous measurements to be performed on the same images.

1.6 Objective

The objective of this study was to explore the relationship between several vegetative indices (NDVI, NDRE, GNDVI, RDVI, RGBVI) and segmentation indices (ExG, ExGR, CIVE) as they relate to visual winter survival scores. It is also of interest to determine the efficacy of predicting winter survival with the use of multiple indices as predictors within the same linear model as opposed to using multiple separate models that include only one index as a predictor. In addition to measuring winter survival, a second objective was to investigate patterns of canopy cover in lines with hybrid necrosis (often expressed after the plant survives the winter) and how changes in canopy cover over time can differentiate between lines with hybrid necrosis and lines with poor winter hardiness.

Chapter 2: Methods

2.1 Field

25 F_1 hybrids from Germany, provided by Dr. Friedrich Longin of Hohenheim University (Table 2.1), were planted in a yield trial setting in fall of 2019 and evaluated in spring 2020 at the University of Nebraska-Lincolns Havelock field. Each plot was 3 meters in length planted with a five-row small plot planter, 0.23 meters between rows, and .115 meter overhang on both end rows for a total plot width of 1.15 meters. The 25 lines were replicated 3 times and randomized across a 2 x 40 grid layout (2 passes, 40 ranges and included five fill plots). Each plot was planted at a seeding rate of 66 kg/ha^{-1}. All plots were walked after emergence to check for germination issues. The following spring during early tillering the lines were visually scored on 0% to 100% score for plot stand which is an indicator of winter survival. This set of hybrids had multiple male parents, hence some hybrids segregated for hybrid necrosis. The plot data was reviewed for accuracy using ariel imagery. It was determined with the aid of aerial imagery that the visually rated plot stand scores for hybrids coded as A and M in replication 2 were swapped. Based on this determination and the data from the other two replications, the visually rated plot stand scores were corrected. Hybrid necrosis was coded as a 1 for presence of necrosis and 0 for no necrosis.

Table 2.1 Pedigrees of Hybrids used in this study

Code	Pedigree	Mother	Father
A	CAPITOLE-VILMORIN x earlymix	CAPITOLE-VILMORIN	Mix of Ferrum, Apache, Porthus
B	KRAJCAR x earlymix	KRAJCAR	Mix of Ferrum, Apache, Porthus

C	NIAB SHW BC 038-10-8-1-1 x earlymix	NIAB SHW BC 038-10-8-1-1	Mix of Ferrum, Apache, Porthus
D	NIAB SHW BC 038-15-8-1-1 x earlymix	NIAB SHW BC 038-15-8-1-1	Mix of Ferrum, Apache, Porthus
E	ROB-173-2-A-17-7 x earlymix	ROB-173-2-A-17-7	Mix of Ferrum, Apache, Porthus
F	MAJOR x earlymix	MAJOR	Mix of Ferrum, Apache, Porthus
G	NIAB SHW BC 038-15-9-1-1 x earlymix	NIAB SHW BC 038-15-9-1-1	Mix of Ferrum, Apache, Porthus
H	RIGOUDI x earlymix	RIGOUDI	Mix of Ferrum, Apache, Porthus
I	NIAB SHW BC 050-6-9-1-1 x earlymix	NIAB SHW BC 050-6-9-1-1	Mix of Ferrum, Apache, Porthus
J	NIAB SHW BC 038-5-4-1-1 x latemix	NIAB SHW BC 038-5-4-1-1	Mix of Hohenheim parents
K	NIAB SHW BC 038-10-1-1-1 x latemix	NIAB SHW BC 038-10-1-1-1	Mix of Hohenheim parents
L	BLEROY x latemix	BLEROY	Mix of Hohenheim parents
M	NIAB SHW BC 038-2-15-1-1 x latemix	NIAB SHW BC 038-2-15-1-1	Mix of Hohenheim parents
N	NIAB SHW BC 045-12-9-1-1 x latemix	NIAB SHW BC 045-12-9-1-1	Mix of Hohenheim parents
O	ROB-173-2-A-17_6 x latemix	ROB-173-2-A-17_6	Mix of Hohenheim parents
P	WW-13019-210-310-402a-1 x latemix	WW-13019-210-310-402a-1	Mix of Hohenheim parents
Q	WW-14010-203-303-6/3 x latemix	WW-14010-203-303-6/3	Mix of Hohenheim parents
R	WW-13023-213-313-405-3 x latemix	WW-13023-213-313-405-3	Mix of Hohenheim parents
S	WW-14008-201-301-15/1 x latemix	WW-14008-201-301-15/1	Mix of Hohenheim parents
T	WW-14008-201-301-2/3 x latemix	WW-14008-201-301-2/3	Mix of Hohenheim parents

U	WW-14008-201-301-21/3 x latemix	WW-14008-201-301-21/3	Mix of Hohenheim parents
V	WW-14008-201-301-4/2 x latemix	WW-14008-201-301-4/2	Mix of Hohenheim parents
W	WW-14009-202-302-14/1 x latemix	WW-14009-202-302-14/1	Mix of Hohenheim parents
X	WW-14009-202-302-4/3 x latemix	WW-14009-202-302-4/3	Mix of Hohenheim parents
Y	WW-14010-203-303-1/3 x latemix	WW-14010-203-303-1/3	Mix of Hohenheim parents

2.2 UAV flights

The multispectral images were captured with Micasense RedEdge multi spectral ± camera (Bands: Blue: 475 ± 20 nm; Green: 560 ± 20 nm; Red: 668 ± 10 nm; RedEdge: nm; Near Infra-Red: 840 ± 40 nm) mounted on a DJI Matrice 600 Pro. The multispectral images were captured on 05/18/2020 (wheat growth stage: jointing), 05/29/2020 (wheat growth stage: early heading) and 06/12/2020 (wheat growth stage: grain filling). True color (RGB) images were captured on 04/25/2020 (wheat growth stage: early tillering), 05/18/2020, 05/29/2020 and 06/12/2020. The true color camera on 04/25/2020 was flown with a Mavic 2 Pro and the remaining true color images were from cameras flown with a DJI Matrice 600 Pro. All flights followed a preprogrammed path ensuring all plots were captured with overlapping images.

2.3 Image processing

All images were stitched together with Pix4D. The create grid tool in ArcPro was used to generate polygons for plot delimitation. All polygon grids overlaying the

multispectral images were the exact same size. Each flight of the true color images needed a different sized grid to effectively capture each plot due to minor bending of the images. The true band images were clipped to the area of interest to reduce processing load. The bands of multispectral images (R, G, B, RE, NIR) had a value range from 0 to 1 while the bands captured with true color camera were left unnormalized and had a value range from 0 to 255. NDVI (Figure 2.1, Image A), NDRE, GNDVI, RDVI, RGBVI, ExG, ExGR and CIVE (Table 2.2) were calculated for all flights of the multi spectral imagery. ExG, ExGR and CIVE were calculated for all flights of the true color imagery.

Table 2.2 Vegetative and Segmentation Indices Used in this Study

Index	Formula	Application	Source
NDVI	(NIR-Red)/(NIR+Red)	Associated with chlorophyll content	Tucker, 1979
NDRE	(NIR-RedEdge)/(NIR+RedEdge)	Associated with chlorophyll and nitrogen	Fitzgerald et al., 2006
GNDVI	(NIR-Green)/(NIR+Green)	Associated with chlorophyll-a	Gitelson et al., 1996
RDVI	(NIR-Red)/√(NIR+Red)	Reduced sensitivity to soil reflectance	Roujean and Breon, 1995
RGBVI	$(Green^2-Blue\times Red)/(Green^2+Blue\times Red)$	Associated with biomass	Bendig et al., 2015
ExG	2×Green-Red-Blue	Segments soil from vegetation	(Woebbecke et al., 1995

ExGR	(2×Green-Red-Blue)-(1.4×Red-Green)	Segments soil from vegetation	Neto, 2004
CIVE	0.441×Red-0.881×Green+0.385×Blue+18.78745	Segments soil from vegetation	Kataoka et al., 2003

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction

Vegetative indices were not calculated for the RGB images due to unnormalized RGB images being unreliable for measuring the mean reflectance (Woebbecke et al., 1995). The true color images were analyzed as unnormalized since the same bands were covered in the multi spectral imagery and to investigate the efficacy of segmentation indices on unnormalized images.

For the multispectral data sets a threshold was determined for each segmentation index by visually examining pixel values of vegetative and soil pixels. ExG was given a threshold of greater than or equal to 0.029 (Figure 2.1, Image B). ExGR was given a threshold of greater than or equal to 0.01. CIVE was given a threshold of less than or equal to 18.77. It was visually determined by examination of each flight date image that these thresholds were sufficient across all three flights. To generate masks (using one image to isolate areas of a second image), each segmentation index had values outside of their respective threshold set to 0 and values within their threshold set to no data utilizing Python, resulting in a mask that outlined the vegetative pixels and removed the soil background (Figure 2.1, Image C). The masks were then overlayed onto each vegetative index using the Mosaic Data Management tool in Arc Pro (Figure 2.1, Image D).

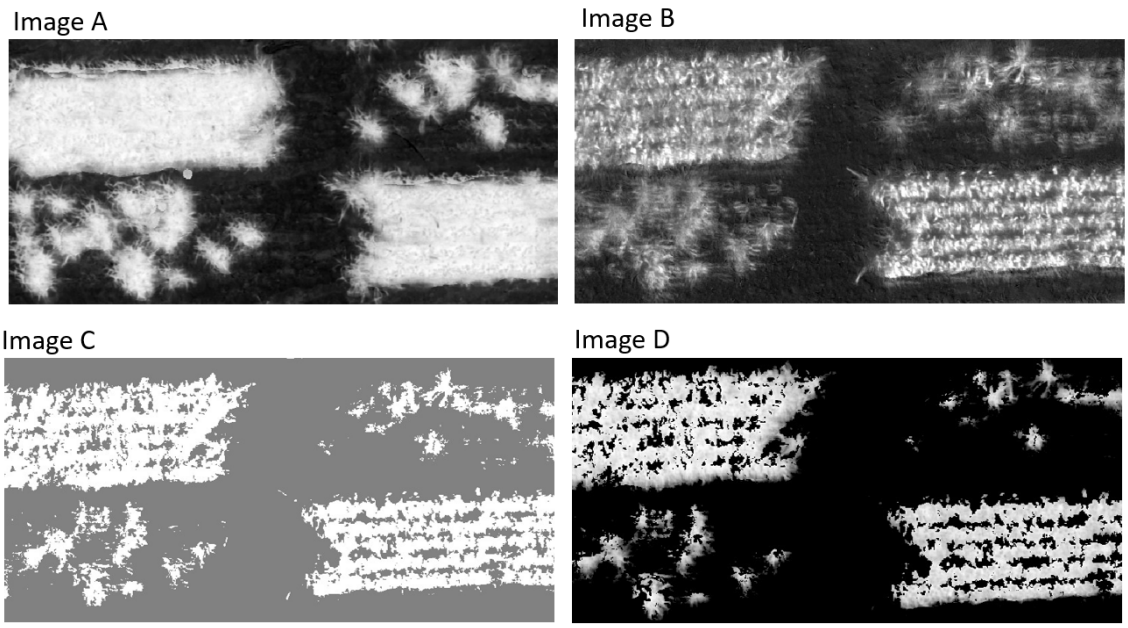

Figure 2.1 ExG Masked NDVI, Flight 5/29, Image A: NDVI on flight 5/29, Image B: ExG on flight 5/29, Image C: ExG segmented mask on flight 5/29, Image D: ExG mask overlayed on NDVI for flight 5/29
NDVI: Normalized Difference Vegetative Index; ExG: Excessive Green

In Python the mean value of all pixels within a plot's respective polygon for the masked and unmasked vegetative indices was calculated for each plot. Values for segmentation indices were calculated by counting the total number of pixels that met the threshold criteria within a plot's respective polygon. For segmentation of the true color images, thresholds were visually determined for each individual flight (Table 2.3). ExGR was not given an upper threshold on flights 5/29 and 6/12 due to mature leaves and shadows of mature leaves having the same or very similar values (Figure 2.2). The values of the segmentation indices for the true color data set were determined by calculating the total number of pixels that met the threshold criterion.

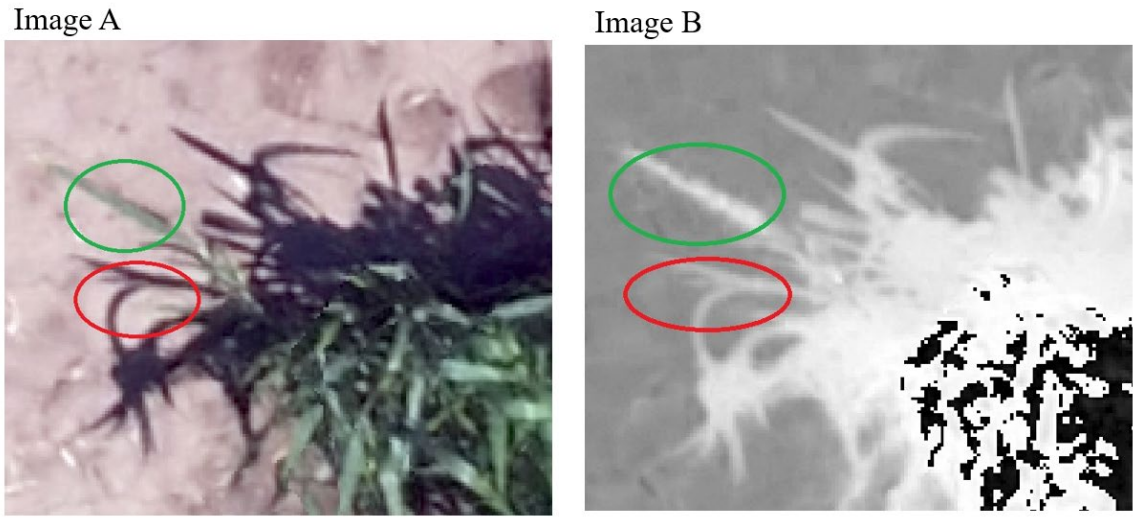

Figure 2.2 True Color and ExGR Leaf Shadow, Image A: RGB image of leaf (circled in green) and shadow (circled in red), Image B: ExGR image of the same leaf (circled in green) and shadow(circled in red). ExGR: Excessive Green Minus Excessive REd

Table 2.3 True Color Segmentation Index Thresholds

Flight	Segmentation Index	Threshold
4/25	ExG	Img ≥ 45; Img ≤ 175
5/18	ExG	Img ≥ 45; Img ≤ 175
5/29	ExG	Img ≥ 30; Img ≤ 175
6/12	ExG	Img ≥ 20; Img ≤ 175
4/25	ExGR	Img ≤ 158
5/18	ExGR	Img ≤ 65; Img ≥ 165
5/29	ExGR*	Img ≤ 65
6/12	ExGR*	Img ≤ 65
4/25	CIVE	Img ≥ 175

5/18	CIVE	Img ≥ 175
5/29	CIVE	Img ≥ 175; Img ≤ 12
6/12	CIVE	Img ≥ 175; Img ≤ 10

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
* Issue with upper threshold distinguishing between leaves and shadows

2.4 Statistical Analysis

All statistical analysis was performed in RStudio (v2023.09.0;R Core Team 2021). A significant difference of plot stands between genotypes was determined using an Analysis of Variance (ANOVA). Additionally, the significance of each index between genotypes was determined with an ANOVA, indices with a non-significant difference between genotypes were removed. All data were analyzed as plot data due to spatial variation observed in the field for lines coded as T and S.

A linear model (initial model) was used to analyze the multispectral data that utilized visual rating of winter survival (plot stand) as the response and all of the indices including an interaction with flight date as the predictors.

Stand ~ NDVImu*Flight + NDREmu*Flight + GNDVImu*Flight + RGBVImu*Flight + ExG_NDVImu*Flight + ExG_NDREmu*Flight + ExG_GNDVImu*Flight + ExG_RDVImu*Flight + (ExG_ExGct/81300)*Flight + ExGR_NDVImu*Flight + ExGR_NDREmu*Flight + ExGR_GNDVImu*Flight + ExGR_RDVImu*Flight + ExGR_RGBVImu*Flight + (ExGR_ExGRct/81300)*Flight + CIVE_NDVImu*Flight + CIVE_NDREmu*Flight + CIVE_GNDVImu*Flight + CIVE_RDVImu*Flight + CIVE_RGBVImu*Flight + (CIVE_CIVEct/81300)*Flight

Masked images were coded as mask_vi (ex. ExG_NDVI is the NDVI image masked with the ExG segmentation index). The segmentation indices were rescaled by dividing them by the total number of pixels in the polygon to alleviate issues with the segmentation indices being on a much different scale than the vegetative indices. Additionally, the data was also separated by flight date and a model containing all indices was generated for each individual flight date. A two-way ANOVA was run on all models.

Pearsons's correlation was calculated between all indices and plot stand within each flight date and across all flight dates. Scatterplots of indices and plot stand were used to identify patterns between hybrid necrotic and non-hybrid necrotic lines that had winterkill, regarding how the lines filled their respective plots over time after their initial plot stand rating. The data used in the scatterplots was separated by flight and labeled based on hybrid necrosis status.

To determine which indices to use in a prediction model a selection criterion of $\alpha = 0.05$ was used. Indices that were shown to be significant ($p < 0.05$) based on the two-way ANOVA performed on the initial model were grouped into a single model (combined prediction model) while the non-significant indices were removed (Table 2.4). Separate individual models were also made containing a single index for each index that was considered significant based on the two-way ANOVA from the initial model.

Plot Stand ~ VI*Flight

All steps statistical analysis steps performed up until this point were repeated on the multispectral data set with the only difference being the data was regrouped based on flight date and the interaction of flight date was removed from the models (Table 2.4).

Table 2.4 Selected Multispectral Combined Prediction Models

Flight Date	Prediction Model
All	Plot Stand ~ NDRE*Flight + RGBVI*Flight + ExG_NDRE*Flight + ExGR_NDRE*Flight + ExGR_ExGR*Flight + CIVE_NDRE*Flight
05/18/2020	Plot Stand ~ NDRE + CIVE_NDVI
05/29/2020	Plot Stand ~ GNDVI + ExG_NDVI + ExG_RGBVI + ExGR_RGBVI + CIVE_NDVI + CIVE_RGBVI
06/12/2020	Plot Stand ~ RGBVI + ExG_GNDVI + ExG_NDRE + CIVE_NDRE + CIVE_GNDVI

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red
Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)

The train function from the Caret package (Kuhn, 2008) was used to generate cross validation and prediction sets, across all selected models. Each prediction model was used to predict plot stand. The predicted plot stands were correlated to the visually scored plot stand.

To analyze the true color data set, a linear model containing all three segmentation indices across all flights was analyzed using a two-way ANOVA. Pearsons's correlation was used to calculate an r value between each index and plot stand. Based on the results on the ANOVA indices with a $p < 0.05$ were selected to used in a prediction model (Table 2.4). Flight date 06/12/2020 only had one index that was significant (CIVE) but was included in Table 2.5 for completeness of the table. Each segmentation index that was significant was also used in an individual model.

Plot Stand ~ SI*Flight

This analysis and selection criterion was repeated on the true color data, but the data were separated by flight date (Table 2.5).

Table 2.5 Selected True Color Combined Prediction Models

Flight Date	Prediction Model
All	Plot Stand ~ ExG*Flight + ExGR*Flight + CIVE*Flight
04/25/2020	Plot Stand ~ ExG + CIVE
05/18/2020	Plot Stand ~ ExGR + CIVE
05/29/2020	Plot Stand ~ ExG + ExGR + CIVE
06/12/2020	Plot Stand ~ CIVE

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red
Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)

Scatterplots of the segmentation indices and plot stand were used to identify patterns between hybrid necrotic and non-hybrid necrotic lines that had winterkill, regarding how the lines filled their respective plots over time after their initial plot stand rating. The data used in the scatterplots was separated by flight and labeled based on hybrid necrosis status.

The train function from the Caret package (Kuhn, 2008) was used to generate cross validation and prediction sets, across all selected true color models. Each prediction model was used to predict plot stand. The predicted plot stands were then correlated to the visually scored plot stand.

The plot data for both multi spectral and true color data sets were examined for numeric patterns of index values across flight dates from NDVI (multispectral) and CIVE

(true color) to comparing non-hybrid necrotic winter killed hybrids, non-winter killed hybrids and hybrid necrotic hybrids.

Chapter 3: Results

3.1 Field

All plots had good germination post planting. 14 plots, consisting of 6 hybrids (hybrids coded as D, G, J, M, S, T), suffered varying degrees of winter kill with winter survival scores ranging from 40% to 80% as estimated by plot stand. Hybrids T and S only had 1 of their 3 replications exhibit winter kill and both of the winterkilled reps were in the central area of the trial. Hybrid necrosis was observed across 9 plots, consisting of 3 hybrids (lines D, G, J). All hybrids exhibiting hybrid necrosis also exhibited winter kill with winter survival scores ranging from 30-60%. The visual plot stand had a significant difference among genotypes ($p < 0.001$). Plot stand was also significantly different among genotypes when the data was separated by flight date ($p < 0.001$).

3.2 Multispectral Image Data and Plot Prediction Results

All indices were determined to be significantly different ($p<0.001$) for genotypes for all data sets, with the exception of ExG_RGBVI in the data set with all flight dates combined, ExG_RGBVI was removed from further analysis. All indices were found to have a good correlation ($r > 0.5$) or strong correlation ($r > 0.8$) when correlated to plot stand across all flight dates (Table 3.1) and within individual flight dates (Table 3.2, 3.3,

and 3.4). However, not all indices were significant ($p < 0.05$). Whether an index was significant varied amongst the models depending on the data set (combined data in Table 3.1, and individual flight date data in Tables 3.2, 3.3, 3.4).

Table 3.1 Multi Spectral p and r Values of Indices from Linear Model Estimating Plot Stand Across All Flight Dates

Index Type	Index	Correlation Between Plot Stand and Index	p value from ANOVA
VI	NDVI	r = 0.831	p = 0.365
VI	NDRE	r = 0.806	p = 0.001
VI	GNDVI	r = 0.833	p = 0.063
VI	RDVI	r = 0.815	p = 0.922
VI	RGBVI	r = 0.788	p = 0.029
MVI	ExG_NDVI	r = 0.805	p = 0.961
MVI	ExG_NDRE	r = 0.775	p = 0.012
MVI	ExG_GNDVI	r = 0.741	p = 0.07
MVI	ExG_RDVI	r = 0.777	p = 0.55
SI	ExG	r = 0.800	p = 0.551
MVI	ExGR_NDVI	r = 0.775	p = 0.728
MVI	ExGR_NDRE	r = 0.721	p = 0.005
MVI	ExGR_GNDVI	r = 0.731	p = 0.077
MVI	ExGR_RDVI	r = 0.742	p = 0.433
MVI	ExGR_RGBVI	r = 0.357	p = 0.168
SI	ExGR	r = 0.862	p = 0.04
MVI	CIVE_NDVI	r = 0.787	p = 0.535
MVI	CIVE_NDRE	r = 0.781	p = 0.005
MVI	CIVE_GNDVI	r = 0.706	p = 0.07
MVI	CIVE_RDVI	r = 0.753	p = 0.654
MVI	CIVE_RGBVI	r = 0.704	p = 0.409
SI	CIVE	r = 0.737	p = 0.353

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)
Index Types, VI: Vegetative Index; MVI: Masked Vegetative Index; SI: Segmentation Index

Table 3.2 Multi Spectral p Values of Indices from Linear Model Estimating Plot Stand for Flight 5/18/2020

Index Type	Index	Correlation Between Plot Stand and Index	p value from ANOVA
VI	NDVI	r = 0.916	p = 0.061
VI	NDRE	r = 0.909	p = 0.002
VI	GNDVI	r = 0.916	p = 0.462
VI	RDVI	r = 0.917	p = 0.367
VI	RGBVI	NA	NA
MVI	ExG_NDVI	r = 0.903	p = 0.2
MVI	ExG_NDRE	r = 0.875	p = 0.514
MVI	ExG_GNDVI	r = 0.882	p = 0.929
MVI	ExG_RDVI	r = 0.905	p = 0.258
MVI	ExG_RGBVI	NA	NA
SI	ExG	r = 0.857	p = 0.962
MVI	ExGR_NDVI	r = 0.908	p = 0.95
MVI	ExGR_NDRE	r = 0.868	p = 0.69
MVI	ExGR_GNDVI	r = 0.877	p = 0.796
MVI	ExGR_RDVI	r = 0.898	p =0.64
MVI	ExGR_RGBVI	r = 0.810	p = 0.659
SI	ExGR	r = 0.914	p = 0.688
MVI	CIVE_NDVI	r = 0.899	p = 0.044
MVI	CIVE_NDRE	r = 0.876	p = 0.341
MVI	CIVE_GNDVI	r = 0.881	p = 0.831
MVI	CIVE_RDVI	r = 0.904	p = 0.151
MVI	CIVE_RGBVI	r = 0.886	p = 0.105
SI	CIVE	r = 0.753	p = 0.781

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction

Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)

Index Types, VI: Vegetative Index; MVI: Masked Vegetative Index; SI: Segmentation Index

Table 3.3 Multi Spectral p Values of Indices from Linear Model Estimating Plot Stand for Flight 5/29/2020

Index Type	Index	Correlation Between Plot Stand and Index	p value from ANOVA
VI	NDVI	r = 0.883	p = 0.971
VI	NDRE	r = 0.883	p = 0.068
VI	GNDVI	r = 0.883	p = 0.048
VI	RDVI	r = 0.881	p = 0.259
VI	RGBVI	r = 0.881	p = 0.976
MVI	ExG_NDVI	r = 0.882	p = 0.016

MVI	ExG_NDRE	r = 0.869	p = 0.42
MVI	ExG_GNDVI	r = 0.859	p = 0.145
MVI	ExG_RDVI	r = 0.884	p = 0.46
MVI	ExG_RGBVI	r = 0.867	p = 0.005
SI	ExG	r = 0.815	p = 0.439
MVI	ExGR_NDVI	r = 0.89	p = 0.34
MVI	ExGR_NDRE	r = 0.859	p = 0.106
MVI	ExGR_GNDVI	r = 0.848	p = 0.269
MVI	ExGR_RDVI	r = 0.865	p = 0.557
MVI	ExGR_RGBVI	r = 0.829	p = 0.048
SI	ExGR	r = 0.885	p = 0.885
MVI	CIVE_NDVI	r = 0.884	p = 0.032
MVI	CIVE_NDRE	r = 0.879	p = 0.399
MVI	CIVE_GNDVI	r = 0.869	p = 0.094
MVI	CIVE_RDVI	r = 0.885	p = 0.506
MVI	CIVE_RGBVI	r = 0.852	p = 0.025
SI	CIVE	r = 0.752	p = 0.774

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)
Index Types, VI: Vegetative Index; MVI: Masked Vegetative Index; SI: Segmentation Index

Table 3.4 Multi Spectral p Values of Indices from Linear Model Estimating Plot Stand for Flight 6/12/2020

Index Type	Index	Correlation Between Plot Stand and Index	p value from ANOVA
VI	NDVI	r = 0.833	p = 0.5
VI	NDRE	r = 0.827	p = 0.238
VI	GNDVI	r = 0.83	p = 0.624
VI	RDVI	r = 0.834	p = 0.106
VI	RGBVI	r = 0.834	p = 0.007
MVI	ExG_NDVI	r = 0.789	p = 0.352
MVI	ExG_NDRE	r = 0.78	p = 0.014
MVI	ExG_GNDVI	r = 0.765	p = 0.023
MVI	ExG_RDVI	r = 0.789	p = 0.105
MVI	ExG_RGBVI	r = 0.726	p = 0.118
SI	ExG	r = 0.775	p = 0.731
MVI	ExGR_NDVI	r = 0.783	p = 0.998
MVI	ExGR_NDRE	r = 0.756	p = 0.201
MVI	ExGR_GNDVI	r = 0.742	p = 0.087

MVI	ExGR_RDVI	r = 0.769	p = 0.084
MVI	ExGR_RGBVI	r = 0.704	p = 0.289
SI	ExGR	r = 0.835	p = 0.095
MVI	CIVE_NDVI	r = 0.783	p = 0.419
MVI	CIVE_NDRE	r = 0.789	p = 0.007
MVI	CIVE_GNDVI	r = 0.766	p = 0.02
MVI	CIVE_RDVI	r = 0.779	p = 0.095
MVI	CIVE_RGBVI	r = 0.693	p = 0.105
SI	CIVE	r = 0.754	p = 0.917

NDVI: Normalized Difference Vegetative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)
Index Types, VI: Vegetative Index; MVI: Masked Vegetative Index; SI: Segmentation Index

To determine the combined prediction model, indices with a $p < 0.05$ were selected. For the data set consisting of all flights NDRE ($p = 0.001$, $r = 0.806$), ExG_NDRE ($p = 0.012$, $r = 0.775$), ExGR_NDRE ($p = 0.005$, $r = 0.721$), ExGR ($p = 0.04$, $r = 0.862$), and CIVE_NDRE ($p = 0.005$, $r = 0.781$) were selected as all other indices were non-significant in the initial model for the all flights data set (Table 3.1). Indices that met the significance criteria for the flight 5/18/2020 flight data set were NDRE ($p = 0.002$, $r = 0.909$) and CIVE_NDVI ($p = 0.044$, $r = 0.899$) (Table 3.2). For the flight 5/18/2020 flight data set, a collinearity problem was found between RGBVI and ExG_RGBVI. The error that caused this collinearity could not be identified nor could it be determined which index had the correct value and which did not, as a result both RGBVI and ExG_RGBVI in the 5/18/2020 flight data set were removed from further analysis. Indices with significance in the 5/29/2020 flight data set were GNDVI ($p = 0.048$, $r = 0.883$), ExG_NDVI ($p = 0.016$, $r = 0.882$), ExG_RGBVI ($p = 0.005$, $r = 0.867$), ExGR_RGBVI ($p= 0.048$, $r = 0.829$), CIVE_NDVI ($p = 0.032$, $r = 0.884$) and CIVE_RGBVI ($p = 0.025$, $r = 0.852$) (Table 3.3). For the flight data set 6/12/2020 the selected indices were RGBVI ($p = 0.007$, $r = 0.834$), ExG_NDRE ($p = 0.014$, $r = 0.78$),

ExG_GNDVI ($p = 0.023$, $r = 0.765$), CIVE_NDRE ($p = 0.007$, $r = 0.789$) and CIVE_GNDVI ($p = 0.02$, $r = 0.766$) (Table 3.4).

The prediction model with the lowest RMSE value across all multispectral flight datasets was the combined prediction model of all significant indices for the data set from flight date 5/18/2020 with an RMSE value of 7.679 (Table 3.5).

Plot Stand ~ NDRE + CIVE_NDVI

The Pearson coefficient for correlation between the predicted stand and the observed stand was $r = 0.836$ ($p < 0.01$) for the model with the lowest RMSE (Table 3.5). The model with the highest correlation ($r = 0.898$, $p < 0.01$) between the visual score of winter survival and the predicted score for winter survival and RMSE value of 10.234 was the combined prediction model from the 6/12/2020 flight data set (Table 3.5)

Plot Stand ~ RGBVI + ExG_GNDVI + ExG_NDRE + CIVE_NDRE + CIVE_GNDVI

Table 3.5 Multispectral Prediction Models Correlation and RMSE with the Observed Data

Flight Date	Prediction Model	Correlation of Plot Stand to Predicted Plot Stand	RMSE
All	Plot Stand ~ NDRE*Flight + RGBVI*Flight + ExG_NDRE *Flight + ExGR_NDRE*Flight + ExGR*Flight + CIVE_NDRE*Flight	r = 0.891** (n = 73)	9.669
All	Plot Stand ~ NDRE*Flight	r = 0.878** (n = 73)	9.64
All	Plot Stand ~ RGBVI*Flight	r = 0.873**	9.723

		(n = 73)	
All	Plot Stand ~ ExG_NDRE*Flight	r = 0.873** (n = 73)	10.456
All	Plot Stand ~ ExGR_NDRE*Flight	r = 0.859** (n = 73)	11.204
All	Plot Stand ~ ExGR*Flight	r = 0.837** (n = 73)	9.342
All	Plot Stand ~ CIVE_NDRE*Flight	r = 0.861** (n = 73)	10.735
05/18/2020	Plot Stand ~ NDRE + CIVE_NDVI	r = 0.836** (n = 23)	7.679
05/18/2020	Plot Stand ~ NDRE	r = 0.845** (n = 23)	8.118
05/18/2020	Plot Stand ~ CIVE_NDVI	r = 0.857** (n = 23)	8.641
05/29/2020	Plot Stand ~ GNDVI + ExG_NDVI + ExG_RGBVI + ExGR_RGBVI + CIVE_NDVI + CIVE_RGBVI	r = 0.84** (n = 23)	8.884
05/29/2020	Plot Stand ~ GNDVI	r = 0.815** (n = 23)	8.468
05/29/2020	Plot Stand ~ ExG_NDVI	r = 0.882** (n = 23)	9.309
05/29/2020	Plot Stand ~ ExG_RGBVI	r = 0.883** (n = 23)	9.918
05/29/2020	Plot Stand ~ ExGR_RGBVI	r = 0.867** (n = 23)	11.176
05/29/2020	Plot Stand ~ CIVE_NDVI	r = 0.884** (n = 23)	8.538
05/29/2020	Plot Stand ~ CIVE_RGBVI	r = 0.852** (n = 23)	9.908
06/12/2020	Plot Stand ~ RGBVI + ExG_GNDVI + ExG_NDRE + CIVE_NDRE + CIVE_GNDVI	r = 0.898** (n = 23)	10.234
06/12/2020	Plot Stand ~ RGBVI	r = 0.845** (n = 23)	10.606
06/12/2020	Plot Stand ~ ExG_GNDVI	r = 0.833** (n = 23)	12.051
06/12/2020	Plot Stand ~ ExG_NDRE	r =0.827** (n = 23)	12.196
06/12/2020	Plot Stand ~ CIVE_NDRE	r = 0.834** (n = 23)	11.847
06/12/2020	Plot Stand ~ CIVE_GNDVI	r = 0.789** (n = 23)	11.721

NDVI: Normalized Difference Veg etative Index; NDRE: Normalized Difference RedEdge; GNDVI: Green Normalized Difference Vegetative Index; RDVI: Renormalized Difference Vegetative Index; RGBVI: Red Green Blue Vegetative Index; ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction

Masked indices are coded as mask_vegetative index (ex. ExG_NDVI: NDVI masked by ExG)
** indicates $p < 0.01$

3.3 True Color Image Data and Plot Stand Prediction Results

All of the segmentation indices were determined to be significantly different for genotypes across all flight dates and within individual flight dates. The true color data showed variable correlation depending on the flight date data set (combined data in Table 3.6, and individual flight date data in Tables 3.7, 3.8, 3.9, 3.10). The highest correlation ($r = 0.918$, $p < 0.001$) with plot stand was ExGR on flight 5/18/2020. As seen with the multispectral data some of the indices were non-significant when analyzed in a combined linear model for each data set (initial model; combined flight date data in Table 3.6, and individual flight date data in Tables 3.7, 3.8, 3.9, 3.10).

Table 3.6 RGB: P Values of Indices from Linear Model Estimating Plot Stand for Flight Across All Flight Dates

Index	Correlation Between Plot Stand and Segmentation index	p value from ANOVA
ExG	r = 0.442	p = 0.005
ExGR	r = 0.468	p < 0.001
CIVE	r = 0.522	p = 0.019

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
*Upper threshold not used on flights 5/29/202 an 6/12/2020 data due to issues delineating between shadows and mature leaves

Table 3.7 RGB: P Values of Indices from Linear Model Estimating Plot Stand for Flight 4/25/2020

Index	Correlation Between Plot Stand and Segmentation index	p value from ANOVA
ExG	r = 0.78	p < 0.001
ExGR	r = 0.773	p = 0.141

CIVE	r = 0.833	p < 0.001

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction

Table 3.8 RGB: P Values of Indices from Linear Model Estimating Plot Stand for Flight 5/18/2020

Index	Correlation Between Plot Stand and Segmentation index	p value from ANOVA
ExG	r = 0.77	p = 0.051
ExGR	r = 0.918	p < 0.001
CIVE	r = 0.899	p = 0.006

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction

Table 3.9 RGB: P Values of Indices from Linear Model Estimating Plot Stand for Flight 5/29/2020

Index	Correlation Between Plot Stand and Segmentation index	p value from ANOVA
ExG	r = 0.666	p < 0.001
ExGR	r = 0.843	p < 0.001
CIVE	r = 0.799	p = 0.007

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
*Upper threshold not used due to issues delineating between shadows and mature leaves

Table 3.10 RGB: P Values of Indices from Linear Model Estimating Plot Stand for Flight 6/12/2020

Index	Correlation Between Plot Stand and Segmentation index	p value from ANOVA
ExG	r = 0.253	p = 0.372
ExGR	r = 0.738	p = 0.393
CIVE	r = 0.79	p = 0.004

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
*Upper threshold not used due to issues delineating between shadows and mature leaves

The results from the two-way ANOVA were used to select indices with a $p < 0.05$ to generate combined prediction models. For the data set that includes all flight dates, the indices ExG ($p = 0.005$, $r = 0.442$), ExGR ($p < 0.001$, $r = 0.468$) and CIVE ($p = 0.019$, $r = 0.522$) were significant (Table 3.6). The 04/25/2020 flight date data set showed significance for indices ExG ($p < 0.001$, $r = 0.78$) and CIVE ($p < 0.001$, $r = 0.833$; Table 3.7). Indices with significance for flight date 05/18/2020 were ExGR ($p < 0.001$, $r = 0.918$) and CIVE ($p = 0.006$, $r = 0.899$; Table 3.8). Flight date 5/29/2020 had significance for indices ExG ($p < 0.001$, $r = 0.666$), ExGR ($p < 0.001$, $r = 0.843$) and CIVE ($p = 0.007$, $r = 0.799$; Table 3.9). Lastly, the data set for flight date 6/12/2020 only CIVE ($p = 0.004$, $r = 0.79$) was significant (Table 3.10).

The prediction model with the lowest RMSE value across all true color datasets was the combined model of all significant indices for the flight date 5/18/2020 with an RMSE value of 7.041 (Table 3.11).

Plot Stand ~ ExGR + CIVE

The Pearson coefficient for correlation between the predicted plot stand and the visually scored plot stand was $r = 0.924$ ($p < 0.01$) for flight date 5/18/2020, the highest r value among the true color predictions and the multispectral predictions (Table 3.11).

Table 3.11 True Color Prediction Models Correlation and RMSE

Flight Date	Prediction Model	Correlation of Plot Stand to	RMSE

		Predicted Plot Stand	
All	Plot Stand ~ ExG*Flight + ExGR*Flight + CIVE*Flight	r = 0.87** (n = 73)	10.234
All	Plot Stand ~ ExG*Flight	r = 0.654** (n = 73)	15.531
All	Plot Stand ~ ExGR*Flight	r = 0.821** (n = 73)	11.544
All	Plot Stand ~ CIVE*Flight	r = 0.831** (n = 73)	11.027
04/25/2020	Plot Stand ~ ExG + CIVE	r = 0.87** (n = 23)	9.644
04/25/2020	Plot Stand ~ ExG	r = 0.78** (n = 23)	12.622
04/25/2020	Plot Stand ~ CIVE	r = 0.833** (n = 23)	11.0
05/18/2020	Plot Stand ~ ExGR + CIVE	r = 0.924** (n = 23)	7.041
05/18/2020	Plot Stand ~ ExGR	r = 0.918** (n = 23)	7.425
05/18/2020	Plot Stand ~ CIVE	r = 0.899** (n = 23)	8.44
05/29/2020	Plot Stand ~ ExG + ExGR + CIVE	r = 0.878** (n = 23)	9.832
05/29/2020	Plot Stand ~ ExG	r = 0.666** (n = 23)	15.548
05/29/2020	Plot Stand ~ ExGR	r = 0.843** (n = 23)	10.417
05/29/2020	Plot Stand ~ CIVE	r = 0.799** (n = 23)	12.0
06/12/2020	Plot Stand ~ CIVE	r = 0.79** (n = 23)	11.699

ExG: Excessive Green; ExGR: Excessive Green Minus Excessive Red; CIVE: Color Index of Vegetative Extraction
** indicates $p < 0.01$

3.4 Evaluation of Observed Patterns with Hybrid Necrosis

In the scatter plots of NDVI it can be observed that the necrotic lines (blue) and non-necrotic lines (red) cluster together much tighter as the flights became later in the season (Fig. 3.1). This same pattern can be observed in the scatter plots of the other indices to varying degrees.

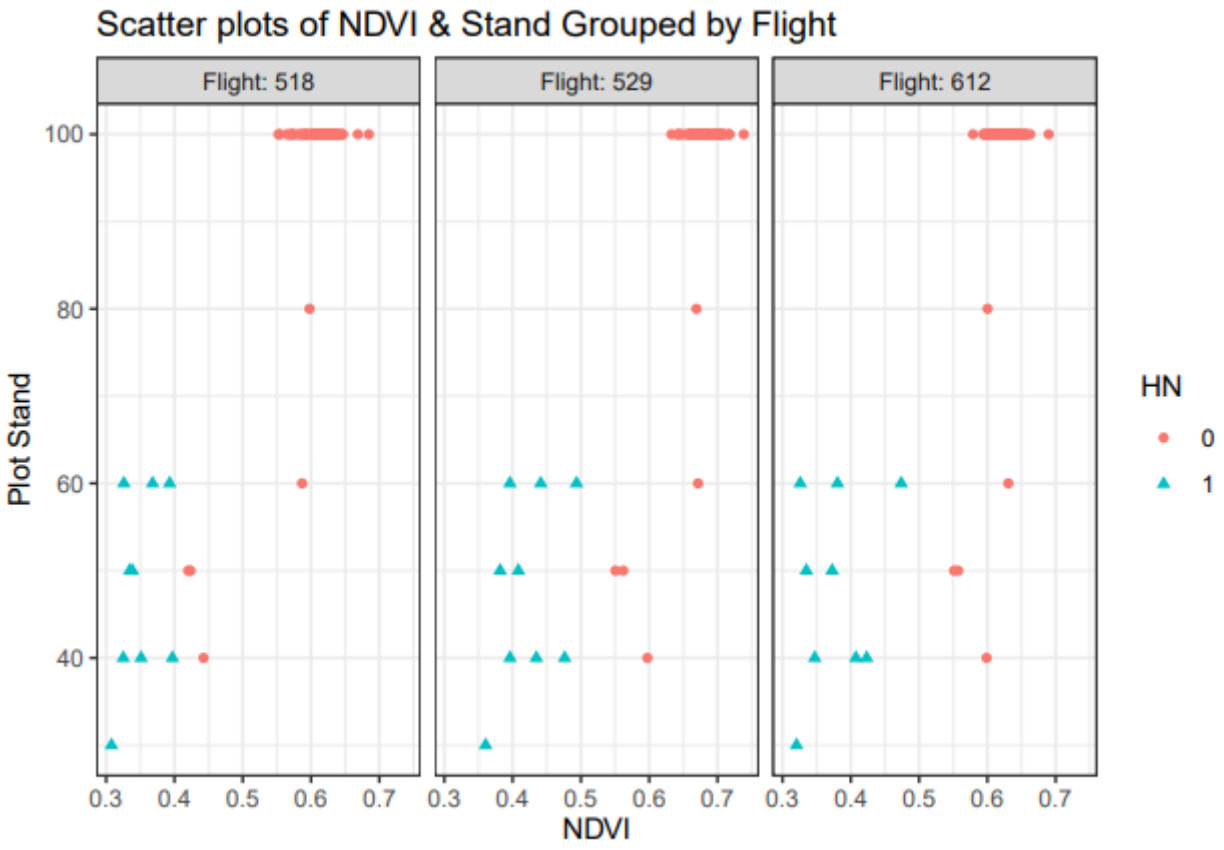

Fig. 3.1 Scatter plots of NDVI & Stand Grouped by Flight

When looking at the numeric data an interesting pattern was observed. The hybrid necrotic hybrids either showed little to no gain in their NDVI value or a moderate increase between the 5/18/2020 flight and 5/29/2020 flight followed by a decrease in NDVI value at the 6/12/2020 flight to relatively similar levels as the 05/18/2020 flight. For example, replication 3 of hybrid D had NDVI value of 0.326 on 5/18/2020, which

went up to 0.396 on 5/29/2020 and then back down to 0.326 on 6/12/2020. This same trend can also be observed in the non-winter killed non-hybrid necrotic hybrids, just with much higher NDVI values. For example: replication 1 of hybrid C had an NDVI value of 0.684 on 5/18/2020, 0.738 on 5/29/2020 and 0.689 on 6/12/2020. This pattern observed in the hybrid necrotic hybrids changes when looking at the hybrid that was not an hybrid necrotic hybrid but did exhibit winter kill, hybrid M, where the NDVI value rises from 5/18/2020 to 5/29/2020 but then plateaus or has relatively small gains in NDVI from 5/29/2020 to 6/12/2020. Example: Replication 2 of hybrid M has a NDVI value of 0.42 on 5/18/2020, 0.551 on 5/29/2020 and 0.557 on 6/12/2020.

In the scatter plot of CIVE and plot stand for the true color data sets a pattern of increased clustering can be observed in the hybrid necrotic lines as compared to the non-hybrid necrotic hybrids (Fig. 3.2) as flight dates progressed in the season, just as was observed with NDVI in the multispectral data (Fig. 3.1). This pattern of clustering was similar to the scatter plot of the ExGR index but was not present in the ExG scatter plot.

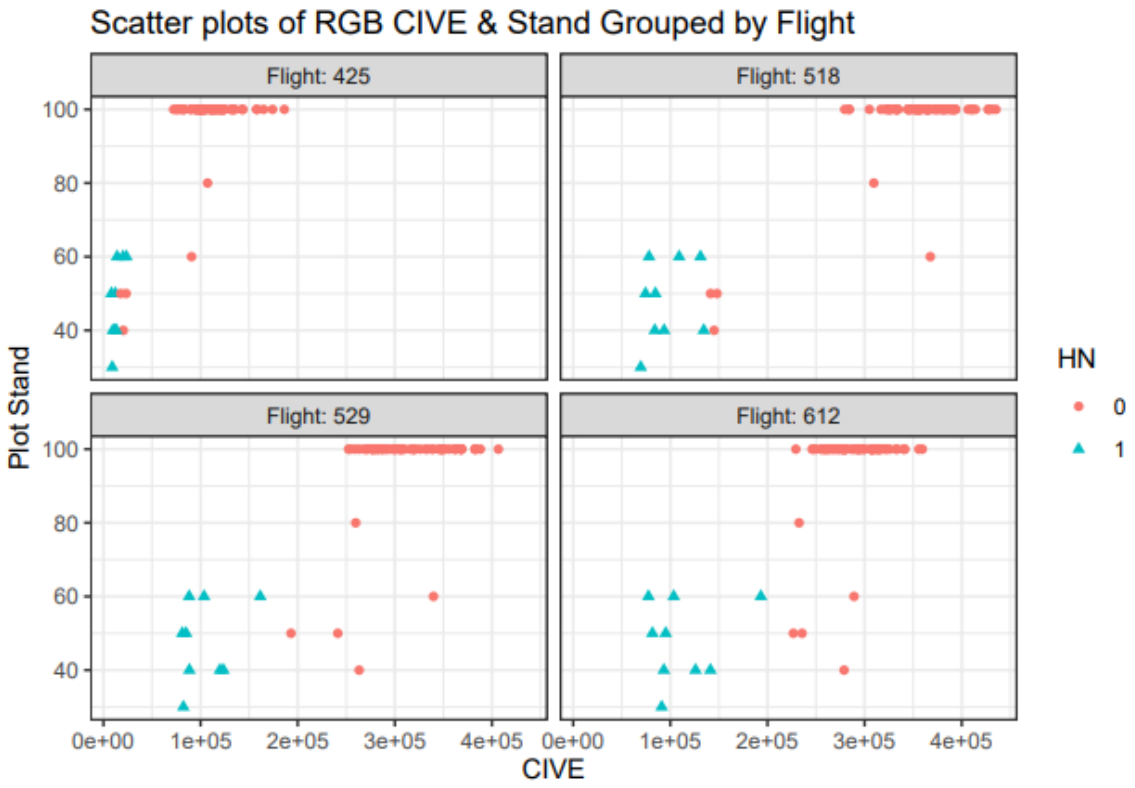

Fig. 3.2 Scatterplots of RGB CIVE & Stand Grouped by Flight

The CIVE data from the true color data set showed a different pattern to the NDVI values from multispectral data with the addition of data from the 04/25/2020 flight date. All hybrids, regardless of hybrid necrosis status or winter kill showed a large increase in CIVE values from 04/25/2020 to 05/18/2020. The non-hybrid necrotic non-winter killed hybrids showed a consistent pattern of 5/18/2020 being the peak for CIVE with slight decrease for the 5/29/2020 and subsequently another small relative decrease in CIVE values for 6/12/2020 For example, replication 1 of hybrid P had CIVE values of 113,168 on 4/25/2020, 435,230 on 5/18/2020, 367,698 on 5/29/2020 and 321,460 on 6/12/2020. The hybrid necrotic hybrids either peaked on 5/29/2020 and plateaued, had a slight decrease on 6/12/2020 or they continued to slightly increase on 6/12/2020, but never reached more than around 50% of the peak CIVE values for non-hybrid necrotic

non-winterkilled plots. For example, replication 2 of Hybrid J had CIVE values of 12,313 on 4/25/2020, 84,622 on 5/18/2020, 85,081 on 5/29/2020, and 81,555 on 6/12/2020 while replicate 1 of hybrid G had CIVE values of 12,499 on 4/25/2020, 93,599 on 5/18/2020, 119,930 on 5/29/2020 and 141,292 on 6/12/2020. The hybrid M that was not hybrid necrotic but did exhibit winter kill in all three replications showed an increase of CIVE values across all dates for replicates 1 and 3 while replicate 2 had a slight decrease in CIVE from 5/29/2020 to 6/12/2020. The CIVE values of hybrid M replicates did increase to over 50% of the peak CIVE value of non-necrotic hybrids that did not exhibit winter kill. Example: Replicate 1 of hybrid M had CIVE value of 20,469 on 4/25/2020, 145,075 on 5/18/2020, 263,243 on 5/29/2020 and 278,895 on 6/12/2020.

Chapter 4: Discussion

The primary objective of this study was to explore the relationship between vegetative indices and segmentation indices as they relate to plot stand which is an indicator of winter survival and to identify a prediction model to determine winter survival scores. We proposed a method of utilizing a combined linear model to predict winter survival of winter wheat using multiple vegetative indices with the goal of a higher correlation of predicted plot stand and visually rated plot stand, over models including only one vegetative index. Previous research has an extensive history using vegetative and segmentation indices to measure canopy cover of different crops, including the comparison of some of the indices (Li et al., 2018; Neto, 2004; Rasmussen et al., 2015). To the best of our knowledge, little research has been done in using multiple indices within the same linear model to improve prediction. Additionally, little research has been done in using segmentation indices to mask other vegetative indices for use in

prediction of plot stand or canopy cover in wheat. When compared to individual indices the masked indices had little to no benefit over their respective indices, however when utilized in combined prediction model some of the masked indices were shown to have a greater level significance and contributed to a better prediction outcome. Overall, we were able to show in both scenarios, unnormalized RGB images and multispectral images, in specific flight dates, the best approach was to use a combined linear model for stand prediction, with an $r = 0.924$ ($p < 0.01$) and $r = 0.836$ ($p < 0.01$) respectively, both of which had strong correlation and the lowest RMSE in comparison to models with individual indices.

In regard to the objective of observing patterns in plot stand over time for hybrid necrosis, it was shown that several vegetative indices were observed to have a strong relationship with hybrid necrosis as the season progressed to later timed flights. This correlation would be expected with the decrease in green biomass that should be observed in more mature necrotic lines (Caldwell, 1943). It was also observed that hybrid necrosis exhibits a clear clustering pattern that separates them from non-necrotic lines for numerous indices, most notably NDVI. The numeric data observation also showed a clear trend of NDVI values decreasing much earlier for hybrids exhibiting hybrid necrosis. Both the scatter plot and numeric patterns showed a clear trend in differences of necrotic hybrids and non-necrotic hybrids, allowing for the potential for necrotic lines to be classified with the use of prediction algorithm or machine learning program. Using the CIVE segmentation index with the true color dataset also showed noticeable differences between necrotic hybrids and non-necrotic hybrids. With the scatterplot of CIVE showing a similar clustering pattern to the NDVI scatter plot and the numeric data showing

increases in plot stand to be substantially smaller in necrotic hybrids than non-necrotic hybrids that exhibited winter kill. The true color images were left unnormalized for the purpose of showing the potential for unnormalized segmentation to be used with machine learning or AI software, that utilize raw RGB images to classify plants exhibiting symptoms of hybrid necrosis, saving on processing time and resources.

Though the excessive green (ExG), color index of vegetation extraction (CIVE) in the true color images and excessive green minus excessive red (ExGR) had good correlation with plot stand depending on the flight, it must be taken into consideration that the ExGR had to drop the upper threshold value due to the inability to segment mature leaves from shadows. It must also be noted that a standard threshold could not be used for the true color data set, and custom thresholds had to be determined for each flight for the respective segmentation indices. The need for a new threshold for each flight date in the true color images indicates the need to be cautious when considering the thresholding in the multispectral imaging was sufficient to be used across all flights, this may have just been due to chance and would need to be further evaluated on whether a standard threshold could be developed for a specific crop.

A limitation of this study that should be considered is the small number of lines observed with limited variability in winter survival and hybrid necrosis and that the data were from one location for a single year. All of the hybrid necrotic lines suffered from winterkill and only 1 hybrid that did not exhibit hybrid necrosis had winter kill in all 3 replications. If repeated, it would be ideal to design a study with checks of known variability in winter survival planted with a greater number of hybrids that are made with only two parental lines, that also produce more combinations with hybrid necrosis. This

data could be done by seed mixtures of winter and spring cultivars to provide a gradient of winter killing. Similarly seed mixtures of known hybrid necrotic and non-hybrid necrotic hybrids could provide a gradient of hybrid necrotic and winter killed data.

Based on the results it appears that the utilization of combined linear models that use more than one index as a predictor provides slightly better results over models that use a single index as a predictor. This was best seen in the non-normalized true color using both ExGR and CIVE ($r = 0.924$, $p < 0.01$) had incremental gain in correlation between predicted plot stand and visually rated plot stand over second highest correlation, the single index ExGR ($r = 0.918$, $p < 0.01$) while also having the lowest RMSE value of 7.041. Even though the difference between the combined model (ExGR + CIVE) and ExGR may not provide a functionally useful significant difference the results indicate the potential for future research to expand and refine the concept of using multiple indices in a single linear model potentially realizing larger more biologically relevant gains in correlation between predicted values and traditionally rated values. As the indices use the same multispectral bands, multiple indices can be readily generated and combined in models using standard computer software. In the multispectral data set, the prediction model with lowest RMSE value of 7.679, which included the vegetative index NDRE and the masked vegetative index CIVE_NDVI ($r = 0.836$, $p < 0.01$), did not have the largest correlation between predicted plot stand and visually rated plot stand, among the prediction models. The multispectral prediction with highest plot stand correlation ($r = 0.898$, $p < 0.01$) was combined model from flight 6/12/2020 that included the indices RGBVI, ExG_GNDVI, ExG_NDRE, CIVE_NDRE and CIVE_GNDVI which had and RMSE value of 10.234. That RMSE was larger than 13 of the other

multispectral prediction models. This provides an interesting scenario where the statistically appropriate model to choose does not predict the trait of interest better than models that should not be chosen based on RMSE. Further research is needed to expand on the idea that utilizing more than one index as a predictor in a single linear model can provide beneficial gains in prediction of agronomic traits. Furthermore, we believe that enough evidence has been shown in the efficacy of using non-normalized RGB values to measure plot stand to provide justification for further research using non-normalized RGB values in prediction of both continuous and categorical traits.

If we were to make recommendations for future research regarding measuring plot stand in winter with a combined model, it would be to utilize a statistical selection method of indices similar to one used in this study to best fit the specific field conditions present in the flight data. In terms of flight dates, it is our opinion that the earlier flight dates of 4/25/2020 and 5/18/2020 would provide the most accurate measurement of plot stand, as these dates occur before the wheat plots are able to “recover” and fill in more of the plot area due to reduced competition. For identifying patterns in hybrid necrosis, it appeared that a minimum of three flights were necessary to capture the changes as the hybrid necrotic hybrids expressed more tissue death. The earlier flight of 4/25/2020 provided little to no benefit in observing hybrid necrosis most likely due to the flight date being before and expression of hybrid necrosis was present.

References

Alessi, J. and Power, J.F. (1971), Influences of Method of Seeding and Moisture on Winter Wheat Survival and Yield1. Agron. J., 63: 81-83. https://doi.org/10.2134/agronj1971.00021962006300010025x

B. Bizimungu, J. Collin, A. Comeau, and C.-A. St-Pierre. 1998. Hybrid necrosis as a barrier to gene transfer in hexaploid winter wheat × triticale crosses. Canadian Journal of Plant Science. 78(2): 239-244. https://doi.org/10.4141/P96-185

Bendig, J., Kang, Y., Helge, A., Andreas, B., Simon, B., Janis, B., et al. (2015). Combining UAV-based plant height from crop surface models, visible, and near infrared vegetation indices for biomass monitoring in barley. Int. J. Appl. Earth Obs. Geoinform. 39, 79–87. doi: 10.1016/j.jag.2015.02.012

Bridger, G. M. (McGill University, et al. "Crown Freezing Tolerance and Field Winter Survival of Winter Cereals in Eastern Canada." Crop Science, vol. 36, no. 1, 1996, pp. 150–57, https://doi.org/10.2135/cropsci1996.0011183X003600010027x.

Caldwell RM, Compton LE (1943) Complementary lethal genes in wheat causing a progressive lethal necrosis of seedlings. J Hered 34:67–70

Camargo Neto, Joao, "A combined statistical-soft computing approach for classification and mapping weed species in minimum -tillage systems" (2004). ETD collection for University of Nebraska - Lincoln. AAI3147135. https://digitalcommons.unl.edu/dissertations/AAI3147135

Chen Y, Sidhu HS, Kaviani M, McElroy MS, Pozniak CJ, Navabi A. Application of image-based phenotyping tools to identify QTL for in-field winter survival of winter wheat (Triticum aestivum L.). Theor Appl Genet. 2019 Sep;132(9):2591-2604. doi: 10.1007/s00122-019-03373-6. Epub 2019 Jun 8. PMID: 31177292.

Cox DJ, Larsen JK, Brun LJ (1986) Winter survival response of winter wheat—tillage and cultivar selection. Agron J 78:795–80

D. W. A. Roberts. 1985. The Effect of Long Exposure to Low Temperatures on the Cold Hardiness of Sprouting Wheat in the Dark. Canadian Journal of Plant Science. 65(4): 893-900. https://doi.org/10.4141/cjps85-115

Fitzgerald, G. J., Rodriguez, D., Christensen, L. K., Belford, R., Sadras, V. O., and Clarke, T. R. (2006). Spectral and thermal sensing for nitrogen and water status in rainfed and irrigated wheat environments. 5th European Conference on Implementation of Precision Agriculture, Uppsala, Sweden, 9-12 June 2005. Secaucus, NJ, United States: Springer. https://doi.org/10.1007/s11119-006-9011-z

Fowler, David. (1982). Date of Seeding, Fall Growth, and Winter Survival of Winter Wheat and Rye. Agronomy Journal - AGRON J. 74. 1060-1063. 10.2134/agronj1982.00021962007400060030x.

Gao XQ, Wang N, Wang XL, Zhang XS. Architecture of Wheat Inflorescence: Insights from Rice. Trends Plant Sci. 2019 Sep;24(9):802-809. doi: 10.1016/j.tplants.2019.06.002. Epub 2019 Jun 27. PMID: 31257155.

Gitelson, A., Kaufman, Y. J., and Merzlyak, M. N. (1996). Use of a green channel in remote sensing of global vegetation from EOS-MODIS. Remote Sens. Environ. 58, 289–298. doi: 10.1016/S0034-4257(96)00072-7

Gitelson, A.A., Kaufman, Y.J., Stark, R. and Rundquist, D. (2002) Novel Algorithms for Remote Estimation of Vegetation Fraction. Remote Sensing of Environment, 80, 76-87. http://dx.doi.org/10.1016/S0034-4257(01)00289-9

Gowda, M., Kling, C., Würschum, T., Liu, W., Maurer, H.P., Hahn, V. and Reif, J.C. (2010), Hybrid Breeding in Durum Wheat: Heterosis and Combining Ability. Crop Sci., 50: 2224-2230. https://doi.org/10.2135/cropsci2009.10.0637

Gupta, P.K., Balyan, H.S., Gahlaut, V. et al. Hybrid wheat: past, present and future. Theor Appl Genet 132, 2463–2483 (2019). https://doi.org/10.1007/s00122-019-03397-y

Li J, Shi Y, Veeranampalayam-Sivakumar A-N and Schachtman DP (2018) Elucidating Sorghum Biomass, Nitrogen and Chlorophyll Contents With Spectral and Morphological Traits Derived From Unmanned Aircraft System. Front. Plant Sci. 9:1406. doi: 10.3389/fpls.2018.01406

Loeppky, Heather, G. Lafond, Fowler, David. (1989). Seeding Depth in Relation to Plant Development, Winter Survival, and Yield of No-Till Winter Wheat. Agronomy Journal - AGRON J. 81. 10.2134/agronj1989.00021962008100010023x.

Longin CF, Gowda M, Mühleisen J, Ebmeyer E, Kazman E, Schachschneider R, Schacht J, Kirchhoff M, Zhao Y, Reif JC. Hybrid wheat: quantitative genetic parameters and consequences for the design of breeding programs. Theor Appl Genet. 2013 Nov;126(11):2791-801. doi: 10.1007/s00122-013-2172-z. Epub 2013 Aug 4. PMID: 23913277.

L. V. Gusta, B. J. O'Connor, and M. G. MacHutcheon. 1997. The selection of superior winter-hardy genotypes using a prolonged freeze test. Canadian Journal of Plant Science. 77(1): 15-21. https://doi.org/10.4141/P95-197

L. V. Gusta, B. J. O'Connor, Y. -P. Gao, and S. Jana. 2001. A re-evaluation of controlled freeze-tests and controlled environment hardening conditions to estimate the winter survival potential of hardy winter wheats. Canadian Journal of Plant Science. 81(2): 241-246. https://doi.org/10.4141/P00-068

J.M. Lyons, J.K. Raison, P.L. Steponkus, 1979: The plant membrane in response to low temperature: an overview. In: Low temperature stress in crop plants: the role of the membrane (LYONS, L. M., GRAHAM, D., RAISON, J. K., eds.), pp. 1–24. New York: Academic Press.

Kataoka, T., Kaneko, T., Okamoto, H., & Hata, S. (2003). Crop growth estimation system using machine vision. Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003), 2, b1079-b1083 vol.2.

Kempe K, Rubtsova M, Gils M (2014) Split-gene system for hybrid wheat seed production. Proc Natl Acad Sci 111:9097–9102 https://doi.org/10.1073/pnas.1402836111

Koichiro Tsunewaki. 1992. Aneuploid analyses of hybrid necrosis and hybrid chlorosis in tetraploid wheats using the D genome chromosome substitution lines of durum wheat. Genome. 35(4): 594-601. https://doi.org/10.1139/g92-089

Kuhn, Max (2008). "Building Predictive Models in R Using the caret Package." Journal of Statistical Software, 28(5), 1–26. doi:10.18637/jss.v028.i05, https://www.jstatsoft.org/index.php/jss/article/view/v028i05.

M. K. Pomeroy and D. B. Fowler. 1973. Use of Lethal Dose Temperature Estimates as Indices of Frost Tolerance for Wheat Cold Acclimated Under Natural and Controlled Environments. Canadian Journal of Plant Science. 53(3): 489-494. https://doi.org/10.4141/cjps73-093

M.P. Reynolds, J.I. Ortiz-Monasterio, and A. McNab (eds.). 2001. Application of Physiology in Wheat Breeding. Mexico, D.F.: CIMMYT

Mukai, Y., Tsunewaki, K. Basic studies on hybrid wheat breeding. Theoret. Appl. Genetics 54, 153–160 (1979). https://doi.org/10.1007/BF00263045

Nishikawa K, Mori T, Takami N, Furuta Y (1974) Mapping of progressive necrosis gene Ne1 and Ne2 of common wheat by the telocentric method. Japan J Breed 24:277–281 https://doi.org/10.1270/jsbbs1951.24.277

Poland JA, Nelson RJ. In the eye of the beholder: the effect of rater variability and different rating scales on QTL mapping. Phytopathology. 2011 Feb;101(2):290-8. doi: 10.1094/PHYTO-03-10-0087. PMID: 20955083.

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rasmussen, Jesper & Ntakos, Georgios & Nielsen, Jon & Svensgaard, Jesper & Poulsen, Robert & Christensen, S.. (2016). Are vegetation indices derived from consumer-grade cameras mounted on UAVs sufficiently reliable for assessing experimental plots?. European Journal of Agronomy. 74. 75-92. 10.1016/j.eja.2015.11.026.

Roujean J-L., Breon, F-M. (1995) Estimating PAR Absorbed by Vegetation from Bi-Directional Reflectance Measurements. Remote Sensing of Environment, 51, 375-384. http://dx.doi.org/10.1016/0034-4257(94)00114-3

Ryan Whitford, Delphine Fleury, Jochen C. Reif, Melissa Garcia, Takashi Okada, Viktor Korzun, Peter Langridge, Hybrid breeding in wheat: technologies to improve hybrid wheat seed production, Journal of Experimental Botany, Volume 64, Issue 18, December 2013, Pages 5411–5428, https://doi.org/10.1093/jxb/ert333

Shafian S, Rajan N, Schnell R, Bagavathiannan M, Valasek J, Shi Y, et al. (2018) Unmanned aerial systems-based remote sensing for monitoring sorghum growth and development. PLoS ONE 13(5): e0196605. https://doi.org/10.1371/journal.pone.0196605

Shi Y, Thomasson JA, Murray SC, Pugh NA, Rooney WL, et al. (2016) Unmanned Aerial Vehicles for High-Throughput Phenotyping and Agronomic Research. PLOS ONE 11(7): e0159781. https://doi.org/10.1371/journal.pone.0159781

Singh, S K & Chatrath, Ravish & Mishra, Bhola. (2010). Perspective of hybrid wheat research: A review. Indian Journal of Agricultural Sciences. 80. 1013-27.

Stanton, Carly & Starek, Michael & Elliott, Norman & Brewer, Michael & Maeda, Murilo & Chu, Tianxing. (2017). Unmanned aircraft system-derived crop height and normalized difference vegetation index metrics for sorghum yield and aphid stress assessment. Journal of Applied Remote Sensing. 11. 026035. 10.1117/1.JRS.11.026035.

Taylor, G. A., & Olsen, R. A. (1985). DESICCATION AS A MAJOR FACTOR IN WINTER INJURY OF WHEAT I. Field Studies. Cereal Research Communications, 13(4), 337–341. http://www.jstor.org/stable/23782950

Tomar SMS, Kochumadhavan M, Nambisan PNN (1991) Hybrid weakness in Triticum dicoccum Schubl. Wheat Inf Serv 72:9–11

Tucker, J. C. (1979), Red and photographic infrared linear combination for monitoring vegetation. Remote Sens. Environ. 8:127-150. https://doi.org/10.1016/0034-4257(79)90013-0

U. J. Pittman and K. H. Tipples. 1978. Survival, Yield, Protein Content, and Baking Quality of Hard Red Winter Wheats Grown Under Various Fertilizer Practices in Southern Alberta. Canadian Journal of Plant Science. 58(4): 1049-1060. https://doi.org/10.4141/cjps78-160

Woebbecke, D. M., Meyer, G. E., Von Bargen, K., & Mortensen, D. A. (1995). Color indices for weed identification under various soil, residue, and lighting conditions. Transactions of the American Society of Agricultural Engineers, 38(1), 259-269.

Yang, Wenzhu & Zhao, Xiaolan & Sile, Wang & Zhang, Jingsi & Feng, Jiaqi. (2015). Greenness identification based on HSV decision tree. Information Processing in Agriculture. 2. 10.1016/j.inpa.2015.07.003.

Zeven AC (1972) Determination of the chromosome and its arm carrying the Ne1-locus of Triticum aestivum L., Chinese Spring and the Ne1-expressivity. Wheat Inf Serv 33–34:4–6

Zheng, Dongxiao & Yang, Xiaoguang & Minguez, M. & Mu, Chenying & He, Qing & Wu, Xia. (2018). Effect of freezing temperature and duration on winter survival and grain yield of winter wheat. Agricultural and Forest Meteorology. 260. 1-8. 10.1016/j.agrformet.2018.05.011.

Zhou, Kuanji & Wang, Shihong & Feng, Yuqin & Liu, Zhongxiang & Wang, Genxuan. (2006). The 4E-System of Producing Hybrid Wheat. Crop Science - CROP SCI. 46. 10.2135/cropsci2005.0029.

Appendix A: Multispectral Image processing Python Code

```
import rasterio
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import fiona
import rasterio.mask
from pandas import DataFrame
import geopandas as gpd

#5\18 Vegetative indices extraction
# Import tif files
green = rasterio.open('518_green.tif')
red = rasterio.open('518_red.tif')
nir = rasterio.open('518_nir.tif')
re = rasterio.open('518_red edge.tif')
blue = rasterio.open('518_blue.tif')

# read in 2d array
nirmap = nir.read(1)
greenmap = green.read(1)
redmap = red.read(1)
remap = re.read(1)
bluemap = blue.read(1)

# correct band to remove negative values
nirc = np.where(nirmap < 0, 0, nirmap)
redc = np.where(redmap < 0, 0, redmap)
rec = np.where(remap < 0, 0, remap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)
```

```
# calculate unmasked VI
ndvimap = (nirc - redc) / (nirc + redc) #vi
ndremap = (nirc - rec) / (nirc + rec) #vi
gndvimap = (nirc - greenc) / (nirc + greenc) #vi
exgmap = (2*greenc) - redc - bluec #mask index
rdvimap = (nirc-redc)/(np.sqrt((nirc+redc))) #vi
rgbvimap = ((greenc**2) - (bluec*redc))/((greenc**2)+(bluec*redc)) #vi
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's
NDVI = np.reshape(ndvimap,(1,ndvimap.shape[0], ndvimap.shape[1]))
NDRE = np.reshape(ndremap,(1, ndremap.shape[0], ndremap.shape[1]))
GNDVI = np.reshape(gndvimap,(1, gndvimap.shape[0], gndvimap.shape[1]))
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
RDVI = np.reshape(rdvimap,(1, rdvimap.shape[0], rdvimap.shape[1]))
RGBVI = np.reshape(rgbvimap,(1, rgbvimap.shape[0], rgbvimap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

#define out_meta using the meta dat from one of the original tif files
out_meta = red.meta

# save reshaped VI's as tif files
with rasterio.open('ndvisave.tif', "w", **out_meta) as dest:
    dest.write(NDVI)
with rasterio.open('ndresave.tif', 'w', **out_meta) as dest:
    dest.write(NDRE)
with rasterio.open('gndvisave.tif', 'w', **out_meta) as dest:
    dest.write(GNDVI)
with rasterio.open('exgsave.tif', 'w', **out_meta) as dest:
    dest.write(ExG)
with rasterio.open('rdvisave.tif', 'w', **out_meta) as dest:
```

```
    dest.write(RDVI)
with rasterio.open('rgbvisave.tif', 'w', **out_meta) as dest:
    dest.write(RGBVI)
with rasterio.open('civesave.tif', 'w', **out_meta) as dest:
    dest.write(CIVE)
with rasterio.open('exgrsave.tif', 'w', **out_meta) as dest:
    dest.write(ExGR)

#Change crs of shp to epsg 32614
data = gpd.read_file('Multi Grid 518-612.shp')
data = data.to_crs(epsg=32614)
data.to_file('CMulti Grid 518-612.shp')

# Calculate 5/18 unmasked VI's
NDVImu = []
NDREmu = []
GNDVImu = []
RDVImu = []
RGBVImu = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('ndvisave.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            NDVImu.append(np.nanmean(out_image))
```

```
    with rasterio.open('ndresave.tif', 'r') as ndresave:
        out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        NDREmu.append(np.nanmean(out_image))

    with rasterio.open('gndvisave.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('rdvisave.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        RDVImu.append(np.nanmean(out_image))

    with rasterio.open('rgbvisave.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        RGBVImu.append(np.nanmean(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, NDVImu, NDREmu, GNDVImu, RDVImu, RGBVImu])
df = df.T
df.columns = ['ID', 'NDVImu', 'NDREmu', 'GNDVImu', 'RDVImu', 'RGBVImu']

# export data frame to excel file
df.to_excel('Output/Unmasked_VI_518.xlsx', index = False)
df

# Check grid polygon size
NDVImu = []

ID = []
```

```
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('ndvisave.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, 225, out_image).astype(float)
            NDVImu.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, NDVImu])
df = df.T
df.columns = ['ID', 'NDVImu']
df

# Correct crs of new shape file to isolate area of interest for mask generation
data = gpd.read_file('MaskShape.shp')
data = data.to_crs(epsg=32614)
data.to_file('CMaskShape.shp')

# Create 518 ExG Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.029, None, 0).astype(float)
        with rasterio.open('exgmask.tif', 'w', **out_meta) as dest:
```

```
            dest.write(out_image)

# Create 518 ExGR Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgrsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.01, None, 0).astype(float)
        with rasterio.open('exgrmask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# Create 518 CIVE Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('civesave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image<=18.77, None, 0).astype(float)
        with rasterio.open('civemask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# calculate mean for ExG Masked 518 VI's
ExG_NDVImu = []
ExG_NDREmu = []
ExG_GNDVImu = []
ExG_ExGct = []
ExG_RDVImu = []
ExG_RGBVImu = []
```

```
ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('518_exg_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExG_NDVImu.append(np.nanmean(out_image))

        with rasterio.open('518_exg_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExG_NDREmu.append(np.nanmean(out_image))



        with rasterio.open('518_exg_gndvi.tif', 'r') as gndvisave:
            out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExG_GNDVImu.append(np.nanmean(out_image))

        with rasterio.open('518_exg_exg.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, 0, out_image).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))

        with rasterio.open('518_exg_rdvi.tif', 'r') as rdvisave:
            out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
```

```
        ExG_RDVImu.append(np.nanmean(out_image))


    with rasterio.open('518_exg_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_RGBVImu.append(np.nanmean(out_image))


#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_NDVImu, ExG_NDREmu, ExG_GNDVImu, ExG_RDVImu, ExG_RGBVImu,
          ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_NDVImu', 'ExG_NDREmu', 'ExG_GNDVImu', 'ExG_RDVImu',
          'ExG_RGBVImu', 'ExG_ExGct']


# export data frame to excel file
df.to_excel('Output/ExG_Masked_VI_518.xlsx', index = False)
df


# calculate mean for ExGR 518 Masked VI's
ExGR_NDVImu = []
ExGR_NDREmu = []
ExGR_GNDVImu = []
ExGR_RDVImu = []
ExGR_RGBVImu = []
ExGR_ExGRct = []


ID = []
for i in range(80):
   with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
      shape = [[feature['geometry'] for feature in shapefile][i]]
      feature = [feature for feature in shapefile][i]
      idx = feature ['properties']['Grid']
      ID.append(idx)
```

```
    with rasterio.open('518_exgr_ndvi.tif', 'r') as ndvisave:
        out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_NDVImu.append(np.nanmean(out_image))

    with rasterio.open('518_exgr_ndre.tif', 'r') as ndresave:
        out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_NDREmu.append(np.nanmean(out_image))



    with rasterio.open('518_exgr_gndvi.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('518_exgr_rdvi.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_RDVImu.append(np.nanmean(out_image))

    with rasterio.open('518_exgr_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_RGBVImu.append(np.nanmean(out_image))

    with rasterio.open('518_exgr_exgr.tif', 'r') as exgrsave:
        out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        ExGR_ExGRct.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, ExGR_NDVImu, ExGR_NDREmu, ExGR_GNDVImu, ExGR_RDVImu,
        ExGR_RGBVImu, ExGR_ExGRct])
```

```
df = df.T
df.columns = ['ID', 'ExGR_NDVImu', 'ExGR_NDREmu', 'ExGR_GNDVImu', 'ExGR_RDVImu',
              'ExGR_RGBVImu', 'ExGR_ExGRct']


# export data frame to excel file
df.to_excel('Output/ExGR_Masked_VI_518.xlsx', index = False)
df


# calculate mean for CIVE 518 Masked VI's
CIVE_NDVImu = []
CIVE_NDREmu = []
CIVE_GNDVImu = []
CIVE_RDVImu = []
CIVE_RGBVImu = []
CIVE_CIVEct = []


ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)


        with rasterio.open('518_cive_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_NDVImu.append(np.nanmean(out_image))


        with rasterio.open('518_cive_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_NDREmu.append(np.nanmean(out_image))
```

```
    with rasterio.open('518_cive_gndvi.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        CIVE_GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('518_cive_rdvi.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        CIVE_RDVImu.append(np.nanmean(out_image))

    with rasterio.open('518_cive_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        CIVE_RGBVImu.append(np.nanmean(out_image))

    with rasterio.open('518_cive_cive.tif', 'r') as civesave:
        out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        CIVE_CIVEct.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, CIVE_NDVImu, CIVE_NDREmu, CIVE_GNDVImu, CIVE_RDVImu,
        CIVE_RGBVImu, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'CIVE_NDVImu', 'CIVE_NDREmu', 'CIVE_GNDVImu', 'CIVE_RDVImu',
        'CIVE_RGBVImu', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/CIVE_Masked_VI_518.xlsx', index = False)
df

#5/29
# Import tif files
```

```
green = rasterio.open('529_green.tif')
red = rasterio.open('529_red.tif')
nir = rasterio.open('529_nir.tif')
re = rasterio.open('529_red edge.tif')
blue = rasterio.open('529_blue.tif')

# read in 2d array
nirmap = nir.read(1)
greenmap = green.read(1)
redmap = red.read(1)
remap = re.read(1)
bluemap = blue.read(1)

# correct band to remove negative values
nirc = np.where(nirmap < 0, 0, nirmap)
redc = np.where(redmap < 0, 0, redmap)
rec = np.where(remap < 0, 0, remap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate unmasked VI
ndvimap = (nirc - redc) / (nirc + redc) #vi
ndremap = (nirc - rec) / (nirc + rec) #vi
gndvimap = (nirc - greenc) / (nirc + greenc) #vi
exgmap = (2*greenc) - redc - bluec #mask index
rdvimap = (nirc-redc)/(np.sqrt((nirc+redc))) #vi
rgbvimap = ((greenc**2) - (bluec*redc))/((greenc**2)+(bluec*redc)) #vi
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's into 3d arrays
NDVI = np.reshape(ndvimap,(1,ndvimap.shape[0], ndvimap.shape[1]))
NDRE = np.reshape(ndremap,(1, ndremap.shape[0], ndremap.shape[1]))
```

```
GNDVI = np.reshape(gndvimap,(1, gndvimap.shape[0], gndvimap.shape[1]))
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
RDVI = np.reshape(rdvimap,(1, rdvimap.shape[0], rdvimap.shape[1]))
RGBVI = np.reshape(rgbvimap,(1, rgbvimap.shape[0], rgbvimap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))


out_meta = red.meta


with rasterio.open('ndvisave.tif', "w", **out_meta) as dest:
   dest.write(NDVI)
with rasterio.open('ndresave.tif', 'w', **out_meta) as dest:
   dest.write(NDRE)
with rasterio.open('gndvisave.tif', 'w', **out_meta) as dest:
   dest.write(GNDVI)
with rasterio.open('exgsave.tif', 'w', **out_meta) as dest:
   dest.write(ExG)
with rasterio.open('rdvisave.tif', 'w', **out_meta) as dest:
   dest.write(RDVI)
with rasterio.open('rgbvisave.tif', 'w', **out_meta) as dest:
   dest.write(RGBVI)
with rasterio.open('civesave.tif', 'w', **out_meta) as dest:
   dest.write(CIVE)
with rasterio.open('exgrsave.tif', 'w', **out_meta) as dest:
   dest.write(ExGR)


# isolate the individual plots within the VI tif files using mask function with shp file
# produce mean of VI pixels within each shp file plot for each VI while generating plot ids associated with
        indices


NDVImu = []
NDREmu = []
GNDVImu = []
RDVImu = []
```

```
RGBVImu = []


ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('ndvisave.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            NDVImu.append(np.nanmean(out_image))

        with rasterio.open('ndresave.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            NDREmu.append(np.nanmean(out_image))

        with rasterio.open('gndvisave.tif', 'r') as gndvisave:
            out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            GNDVImu.append(np.nanmean(out_image))

        with rasterio.open('rdvisave.tif', 'r') as rdvisave:
            out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            RDVImu.append(np.nanmean(out_image))

        with rasterio.open('rgbvisave.tif', 'r') as rgbvisave:
            out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
```

```
        RGBVImu.append(np.nanmean(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, NDVImu, NDREmu, GNDVImu, RDVImu, RGBVImu])
df = df.T
df.columns = ['ID', 'NDVImu', 'NDREmu', 'GNDVImu', 'RDVImu', 'RGBVImu']

# export data frame to excel file
df.to_excel('Output/Unmasked_VI_529.xlsx', index = False)
df

# Create 529 ExG Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.029, None, 0).astype(float)
        with rasterio.open('exgmask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# Create 529 ExGR Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgrsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.01, None, 0).astype(float)
```

```
        with rasterio.open('exgrmask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# Create 529 CIVE Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('civesave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image<=18.77, None, 0).astype(float)
        with rasterio.open('civemask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# calculate mean for ExG Masked 529 VI's
ExG_NDVImu = []
ExG_NDREmu = []
ExG_GNDVImu = []
ExG_ExGct = []
ExG_RDVImu = []
ExG_RGBVImu = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('529_exg_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
```

```
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_NDVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exg_ndre.tif', 'r') as ndresave:
        out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_NDREmu.append(np.nanmean(out_image))



    with rasterio.open('529_exg_gndvi.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exg_exg.tif', 'r') as exgsave:
        out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        ExG_ExGct.append(np.count_nonzero(out_image))

    with rasterio.open('529_exg_rdvi.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_RDVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exg_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_RGBVImu.append(np.nanmean(out_image))

#combine the means of the VI's into one data frame
df = DataFrame([ID, ExG_NDVImu, ExG_NDREmu, ExG_GNDVImu, ExG_RDVImu, ExG_RGBVImu,
        ExG_ExGct])
df = df.T
```

```
df.columns = ['ID', 'ExG_NDVImu', 'ExG_NDREmu', 'ExG_GNDVImu', 'ExG_RDVImu',
              'ExG_RGBVImu', 'ExG_ExGct']


# export data frame to excel file
df.to_excel('Output/ExG_Masked_VI_529.xlsx', index = False)
df


# calculate mean for ExGR 529 Masked VI's
ExGR_NDVImu = []
ExGR_NDREmu = []
ExGR_GNDVImu = []
ExGR_RDVImu = []
ExGR_RGBVImu = []
ExGR_ExGRct = []


ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)


        with rasterio.open('529_exgr_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_NDVImu.append(np.nanmean(out_image))


        with rasterio.open('529_exgr_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_NDREmu.append(np.nanmean(out_image))
```

```
    with rasterio.open('529_exgr_gndvi.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exgr_rdvi.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_RDVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exgr_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExGR_RGBVImu.append(np.nanmean(out_image))

    with rasterio.open('529_exgr_exgr.tif', 'r') as exgrsave:
        out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        ExGR_ExGRct.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, ExGR_NDVImu, ExGR_NDREmu, ExGR_GNDVImu, ExGR_RDVImu,
        ExGR_RGBVImu, ExGR_ExGRct])
df = df.T
df.columns = ['ID', 'ExGR_NDVImu', 'ExGR_NDREmu', 'ExGR_GNDVImu', 'ExGR_RDVImu',
        'ExGR_RGBVImu', 'ExGR_ExGRct']

# export data frame to excel file
df.to_excel('Output/ExGR_Masked_VI_529.xlsx', index = False)
df

# calculate mean for CIVE 529 Masked VI's
CIVE_NDVImu = []
CIVE_NDREmu = []
```

```
CIVE_GNDVImu = []
CIVE_RDVImu = []
CIVE_RGBVImu = []
CIVE_CIVEct = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('529_cive_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_NDVImu.append(np.nanmean(out_image))

        with rasterio.open('529_cive_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_NDREmu.append(np.nanmean(out_image))



        with rasterio.open('529_cive_gndvi.tif', 'r') as gndvisave:
            out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_GNDVImu.append(np.nanmean(out_image))

        with rasterio.open('529_cive_rdvi.tif', 'r') as rdvisave:
            out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            CIVE_RDVImu.append(np.nanmean(out_image))
```

```
    with rasterio.open('529_cive_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        CIVE_RGBVImu.append(np.nanmean(out_image))

    with rasterio.open('529_cive_cive.tif', 'r') as civesave:
        out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        CIVE_CIVEct.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, CIVE_NDVImu, CIVE_NDREmu, CIVE_GNDVImu, CIVE_RDVImu,
        CIVE_RGBVImu, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'CIVE_NDVImu', 'CIVE_NDREmu', 'CIVE_GNDVImu', 'CIVE_RDVImu',
        'CIVE_RGBVImu', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/CIVE_Masked_VI_529.xlsx', index = False)
df

#6/12
# Import tif files
green = rasterio.open('612_green.tif')
red = rasterio.open('612_red.tif')
nir = rasterio.open('612_nir.tif')
re = rasterio.open('612_red edge.tif')
blue = rasterio.open('612_blue.tif')

# read in 2d array
nirmap = nir.read(1)
greenmap = green.read(1)
redmap = red.read(1)
```

```
remap = re.read(1)
bluemap = blue.read(1)

# correct band to remove negative values
nirc = np.where(nirmap < 0, 0, nirmap)
redc = np.where(redmap < 0, 0, redmap)
rec = np.where(remap < 0, 0, remap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate unmasked VI
ndvimap = (nirc - redc) / (nirc + redc) #vi
ndremap = (nirc - rec) / (nirc + rec) #vi
gndvimap = (nirc - greenc) / (nirc + greenc) #vi
exgmap = (2*greenc) - redc - bluec #mask index
rdvimap = (nirc-redc)/(np.sqrt((nirc+redc))) #vi
rgbvimap = ((greenc**2) - (bluec*redc))/((greenc**2)+(bluec*redc)) #vi
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's into 3d arrays
NDVI = np.reshape(ndvimap,(1,ndvimap.shape[0], ndvimap.shape[1]))
NDRE = np.reshape(ndremap,(1, ndremap.shape[0], ndremap.shape[1]))
GNDVI = np.reshape(gndvimap,(1, gndvimap.shape[0], gndvimap.shape[1]))
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
RDVI = np.reshape(rdvimap,(1, rdvimap.shape[0], rdvimap.shape[1]))
RGBVI = np.reshape(rgbvimap,(1, rgbvimap.shape[0], rgbvimap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

out_meta = red.meta
```

```
with rasterio.open('ndvisave.tif', "w", **out_meta) as dest:
    dest.write(NDVI)
with rasterio.open('ndresave.tif', 'w', **out_meta) as dest:
    dest.write(NDRE)
with rasterio.open('gndvisave.tif', 'w', **out_meta) as dest:
    dest.write(GNDVI)
with rasterio.open('exgsave.tif', 'w', **out_meta) as dest:
    dest.write(ExG)
with rasterio.open('rdvisave.tif', 'w', **out_meta) as dest:
    dest.write(RDVI)
with rasterio.open('rgbvisave.tif', 'w', **out_meta) as dest:
    dest.write(RGBVI)
with rasterio.open('civesave.tif', 'w', **out_meta) as dest:
    dest.write(CIVE)
with rasterio.open('exgrsave.tif', 'w', **out_meta) as dest:
    dest.write(ExGR)


# isolate the individual plots within the VI tif files using mask function with shp file
# produce mean of VI pixels within each shp file plot for each VI while generating plot ids associated with
        indices


NDVImu = []
NDREmu = []
GNDVImu = []
RDVImu = []
RGBVImu = []


ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)
```

```
    with rasterio.open('ndvisave.tif', 'r') as ndvisave:
        out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        NDVImu.append(np.nanmean(out_image))

    with rasterio.open('ndresave.tif', 'r') as ndresave:
        out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        NDREmu.append(np.nanmean(out_image))



    with rasterio.open('gndvisave.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('rdvisave.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        RDVImu.append(np.nanmean(out_image))

    with rasterio.open('rgbvisave.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        RGBVImu.append(np.nanmean(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, NDVImu, NDREmu, GNDVImu, RDVImu, RGBVImu])
df = df.T
df.columns = ['ID', 'NDVImu', 'NDREmu', 'GNDVImu', 'RDVImu', 'RGBVImu']

# export data frame to excel file
```

```
df.to_excel('Output/Unmasked_VI_612.xlsx', index = False)
df

# Create 612 ExG Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.029, None, 0).astype(float)
        with rasterio.open('exgmask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# Create 612 ExGR Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        with rasterio.open('exgrsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image>=0.01, None, 0).astype(float)
        with rasterio.open('exgrmask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# Create 612 CIVE Mask

for i in range (1):
    with fiona.open('CMaskShape.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
```

```
        with rasterio.open('civesave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape)
            out_image = np.where(out_image<=18.77, None, 0).astype(float)
        with rasterio.open('civemask.tif', 'w', **out_meta) as dest:
            dest.write(out_image)

# calculate mean for ExG Masked 612 VI's
ExG_NDVImu = []
ExG_NDREmu = []
ExG_GNDVImu = []
ExG_ExGct = []
ExG_RDVImu = []
ExG_RGBVImu = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('612_exg_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExG_NDVImu.append(np.nanmean(out_image))

        with rasterio.open('612_exg_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExG_NDREmu.append(np.nanmean(out_image))
```

```
    with rasterio.open('612_exg_gndvi.tif', 'r') as gndvisave:
        out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_GNDVImu.append(np.nanmean(out_image))

    with rasterio.open('612_exg_exg.tif', 'r') as exgsave:
        out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, 0, out_image).astype(float)
        ExG_ExGct.append(np.count_nonzero(out_image))

    with rasterio.open('612_exg_rdvi.tif', 'r') as rdvisave:
        out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_RDVImu.append(np.nanmean(out_image))

    with rasterio.open('612_exg_rgbvi.tif', 'r') as rgbvisave:
        out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
        out_image = np.where(out_image==0, None, out_image).astype(float)
        ExG_RGBVImu.append(np.nanmean(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_NDVImu, ExG_NDREmu, ExG_GNDVImu, ExG_RDVImu, ExG_RGBVImu,
        ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_NDVImu', 'ExG_NDREmu', 'ExG_GNDVImu', 'ExG_RDVImu',
        'ExG_RGBVImu', 'ExG_ExGct']

# export data frame to excel file
df.to_excel('Output/ExG_Masked_VI_612.xlsx', index = False)
df

# calculate mean for ExGR 612 Masked VI's
ExGR_NDVImu = []
ExGR_NDREmu = []
```

```
ExGR_GNDVImu = []
ExGR_RDVImu = []
ExGR_RGBVImu = []
ExGR_ExGRct = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('612_exgr_ndvi.tif', 'r') as ndvisave:
            out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_NDVImu.append(np.nanmean(out_image))

        with rasterio.open('612_exgr_ndre.tif', 'r') as ndresave:
            out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_NDREmu.append(np.nanmean(out_image))



        with rasterio.open('612_exgr_gndvi.tif', 'r') as gndvisave:
            out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_GNDVImu.append(np.nanmean(out_image))

        with rasterio.open('612_exgr_rdvi.tif', 'r') as rdvisave:
            out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_RDVImu.append(np.nanmean(out_image))
```

```
        with rasterio.open('612_exgr_rgbvi.tif', 'r') as rgbvisave:
            out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, None, out_image).astype(float)
            ExGR_RGBVImu.append(np.nanmean(out_image))

        with rasterio.open('612_exgr_exgr.tif', 'r') as exgrsave:
            out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image==0, 0, out_image).astype(float)
            ExGR_ExGRct.append(np.count_nonzero(out_image))

#combine the means of th VI's into one data frame
df = DataFrame([ID, ExGR_NDVImu, ExGR_NDREmu, ExGR_GNDVImu, ExGR_RDVImu,
            ExGR_RGBVImu, ExGR_ExGRct])
df = df.T
df.columns = ['ID', 'ExGR_NDVImu', 'ExGR_NDREmu', 'ExGR_GNDVImu', 'ExGR_RDVImu',
            'ExGR_RGBVImu', 'ExGR_ExGRct']

# export data frame to excel file
df.to_excel('Output/ExGR_Masked_VI_612.xlsx', index = False)
df

# calculate mean for CIVE 612 Masked VI's
CIVE_NDVImu = []
CIVE_NDREmu = []
CIVE_GNDVImu = []
CIVE_RDVImu = []
CIVE_RGBVImu = []
CIVE_CIVEct = []

ID = []
for i in range(80):
    with fiona.open('CMulti Grid 518-612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
```

```
feature = [feature for feature in shapefile][i]
idx = feature ['properties']['Grid']
ID.append(idx)

with rasterio.open('612_cive_ndvi.tif', 'r') as ndvisave:
    out_image, out_transform = rasterio.mask.mask(ndvisave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, None, out_image).astype(float)
    CIVE_NDVImu.append(np.nanmean(out_image))

with rasterio.open('612_cive_ndre.tif', 'r') as ndresave:
    out_image, out_transform = rasterio.mask.mask(ndresave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, None, out_image).astype(float)
    CIVE_NDREmu.append(np.nanmean(out_image))



with rasterio.open('612_cive_gndvi.tif', 'r') as gndvisave:
    out_image, out_transform = rasterio.mask.mask(gndvisave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, None, out_image).astype(float)
    CIVE_GNDVImu.append(np.nanmean(out_image))

with rasterio.open('612_cive_rdvi.tif', 'r') as rdvisave:
    out_image, out_transform = rasterio.mask.mask(rdvisave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, None, out_image).astype(float)
    CIVE_RDVImu.append(np.nanmean(out_image))

with rasterio.open('612_cive_rgbvi.tif', 'r') as rgbvisave:
    out_image, out_transform = rasterio.mask.mask(rgbvisave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, None, out_image).astype(float)
    CIVE_RGBVImu.append(np.nanmean(out_image))

with rasterio.open('612_cive_cive.tif', 'r') as civesave:
    out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
    out_image = np.where(out_image==0, 0, out_image).astype(float)
```

```
        CIVE_CIVEct.append(np.count_nonzero(out_image))


#combine the means of th VI's into one data frame

df = DataFrame([ID, CIVE_NDVImu, CIVE_NDREmu, CIVE_GNDVImu, CIVE_RDVImu,
        CIVE_RGBVImu, CIVE_CIVEct])

df = df.T

df.columns = ['ID', 'CIVE_NDVImu', 'CIVE_NDREmu', 'CIVE_GNDVImu', 'CIVE_RDVImu',
        'CIVE_RGBVImu', 'CIVE_CIVEct']


# export data frame to excel file

df.to_excel('Output/CIVE_Masked_VI_612.xlsx', index = False)

df
```

Appendix B: True Color Image Processing Python Code

```
import rasterio
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import fiona
import rasterio.mask
from pandas import DataFrame
import geopandas as gpd

# 425 Analysis
#Read in RGB file
image = rasterio.open('425_cRGB.tif')

#read in 2d array
redmap = image.read(1)
greenmap = image.read(2)
bluemap = image.read(3)

imgmeta = rasterio.open('425_cRGB_Blue.tif')

# correct band to remove negative values
redc = np.where(redmap < 0, 0, redmap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate indices
exgmap = (2*greenc) - redc - bluec #mask index
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's so they can be used with rast functions
```

```
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

#define out_meta using the meta dat from one of the original tif files
out_meta = imgmeta.meta

# save reshaped VI's as tif files
with rasterio.open('rgb_exgsave.tif', 'w', **out_meta) as dest:
    dest.write(ExG)
with rasterio.open('rgb_civesave.tif', 'w', **out_meta) as dest:
    dest.write(CIVE)
with rasterio.open('rgb_exgrsave.tif', 'w', **out_meta) as dest:
    dest.write(ExGR)

#Change crs of shp to epsg 32614
data = gpd.read_file('RGB Grid 425.shp')
data = data.to_crs(epsg=32614)
data.to_file('CRGB Grid 425.shp')

# calculate mean for ExG Masked 425 VI's
ExG_ExGct = []
ExGR_ExGRct = []
CIVE_CIVEct = []



ID = []

for i in range(80):
    with fiona.open('CRGB Grid 425.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
```

```
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, out_image, 0).astype(float)
            out_image = np.where(out_image>=45, out_image, 0).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))

        with rasterio.open('rgb_exgrsave.tif', 'r') as exgrsave:
            out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=158, out_image, 0).astype(float)
            ExGR_ExGRct.append(np.count_nonzero(out_image))

        with rasterio.open('rgb_civesave.tif', 'r') as civesave:
            out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
            out_image = np.where(out_image>=175, out_image, 0).astype(float)
            CIVE_CIVEct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct, ExGR_ExGRct, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'ExG_ExGct', 'ExGR_ExGRct', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/RGB_Masked_VI_425.xlsx', index = False)
df

# Check grid size
# calculate ExG Masked 425 VI's
ExG_ExGct = []
```

```
ID = []

for i in range(80):
    with fiona.open('CRGB Grid 425.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, 225, out_image).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_ExGct']

df

#change remaining shape file crs's
data = gpd.read_file('RGB Grid 518.shp')
data = data.to_crs(epsg=32614)
data.to_file('CRGB Grid 518.shp')

data = gpd.read_file('RGB Grid 529.shp')
data = data.to_crs(epsg=32614)
data.to_file('CRGB Grid 529.shp')
```

```
data = gpd.read_file('RGB Grid 612.shp')
data = data.to_crs(epsg=32614)
data.to_file('CRGB Grid 612.shp')

# 518 Analysis
#Read in RGB file
image = rasterio.open('518_cRGB.tif')

#read in 2d array
redmap = image.read(1)
greenmap = image.read(2)
bluemap = image.read(3)

# correct band to remove negative values
redc = np.where(redmap < 0, 0, redmap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate unmasked VI
exgmap = (2*greenc) - redc - bluec #mask index
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's so they can be used with rast functions
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

#define out_meta using the meta dat from one of the original tif files
imgmeta = rasterio.open('518_cRGB_Red.tif')
out_meta = imgmeta.meta

# save reshaped VI's as tif files
```

```
with rasterio.open('rgb_exgsave.tif', 'w', **out_meta) as dest:
    dest.write(ExG)
with rasterio.open('rgb_civesave.tif', 'w', **out_meta) as dest:
    dest.write(CIVE)
with rasterio.open('rgb_exgrsave.tif', 'w', **out_meta) as dest:
    dest.write(ExGR)

# calculate mean for ExG Masked 518 VI's
ExG_ExGct = []
ExGR_ExGRct = []
CIVE_CIVEct = []



ID = []

for i in range(80):
    with fiona.open('CRGB Grid 518.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, out_image, 0).astype(float)
            out_image = np.where(out_image>=45, out_image, 0).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))

        with rasterio.open('rgb_exgrsave.tif', 'r') as exgrsave:
            out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=65, 225, out_image).astype(float)
            out_image = np.where(out_image>=165, out_image, 0).astype(float)
            ExGR_ExGRct.append(np.count_nonzero(out_image))
```

```
        with rasterio.open('rgb_civesave.tif', 'r') as civesave:
            out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=10, 225, out_image).astype(float)
            out_image = np.where(out_image>=175, out_image, 0).astype(float)
            CIVE_CIVEct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct, ExGR_ExGRct, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'ExG_ExGct', 'ExGR_ExGRct', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/RGB_Masked_VI_518.xlsx', index = False)
df

# Check grid size
# calculate ExG Masked 518 VI's
ExG_ExGct = []

ID = []

for i in range(80):
    with fiona.open('CRGB Grid 518.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
```

```
        out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image<=175, 225, out_image).astype(float)
        ExG_ExGct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_ExGct']

df


# 529 Analysis
#Read in RGB file
image = rasterio.open('529_cRGB.tif')

#read in 2d array
redmap = image.read(1)
greenmap = image.read(2)
bluemap = image.read(3)

# correct band to remove negative values
redc = np.where(redmap < 0, 0, redmap)
greenc = np.where(greenmap < 0, 0, greenmap)
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate unmasked VI
exgmap = (2*greenc) - redc - bluec #mask index
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's so they can be used with rast functions
```

```
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

#define out_meta using the meta dat from one of the original tif files
imgmeta = rasterio.open('529_cRGB_Red.tif')
out_meta = imgmeta.meta

# save reshaped VI's as tif files
with rasterio.open('rgb_exgsave.tif', 'w', **out_meta) as dest:
   dest.write(ExG)
with rasterio.open('rgb_civesave.tif', 'w', **out_meta) as dest:
   dest.write(CIVE)
with rasterio.open('rgb_exgrsave.tif', 'w', **out_meta) as dest:
   dest.write(ExGR)

# calculate mean for ExG Masked 529 VI's
ExG_ExGct = []
ExGR_ExGRct = []
CIVE_CIVEct = []



ID = []

for i in range(80):
   with fiona.open('CRGB Grid 529.shp', 'r') as shapefile:
      shape = [[feature['geometry'] for feature in shapefile][i]]
      feature = [feature for feature in shapefile][i]
      idx = feature ['properties']['Grid']
      ID.append(idx)

      with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
         out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
```

```
        out_image = np.where(out_image<=175, out_image, 0).astype(float)
        out_image = np.where(out_image>=30, out_image, 0).astype(float)
        ExG_ExGct.append(np.count_nonzero(out_image))

    with rasterio.open('rgb_exgrsave.tif', 'r') as exgrsave:
        out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
        out_image = np.where(out_image<=65, out_image, 0).astype(float)
        ExGR_ExGRct.append(np.count_nonzero(out_image))

    with rasterio.open('rgb_civesave.tif', 'r') as civesave:
        out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
        out_image = np.where(out_image<=10, 225, out_image).astype(float)
        out_image = np.where(out_image>=175, out_image, 0).astype(float)
        CIVE_CIVEct.append(np.count_nonzero(out_image))



#combine the means of the VI's into one data frame
df = DataFrame([ID, ExG_ExGct, ExGR_ExGRct, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'ExG_ExGct', 'ExGR_ExGRct', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/RGB_Masked_VI_529.xlsx', index = False)
df

# Check grid size
# calculate ExG Masked 529 VI's
ExG_ExGct = []

ID = []
```

```
for i in range(80):
    with fiona.open('CRGB Grid 529.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, 225, out_image).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_ExGct']

df

# 612 Analysis
#Read in RGB file
image = rasterio.open('612_cRGB.tif')

#read in 2d array
redmap = image.read(1)
greenmap = image.read(2)
bluemap = image.read(3)

# correct band to remove negative values
redc = np.where(redmap < 0, 0, redmap)
greenc = np.where(greenmap < 0, 0, greenmap)
```

```
bluec = np.where(bluemap < 0, 0, bluemap)

# calculate unmasked VI
exgmap = (2*greenc) - redc - bluec #mask index
civemap = ((0.441*redc)-(0.881*greenc)+(0.385*bluec)+18.78745) #mask index
exgrmap = (exgmap - ((1.4*redc)-greenc)) #mask index

# reshape VI's so they can be used with rast functions
ExG = np.reshape(exgmap,(1, exgmap.shape[0], exgmap.shape[1]))
CIVE = np.reshape(civemap,(1, civemap.shape[0], civemap.shape[1]))
ExGR = np.reshape(exgrmap,(1, exgrmap.shape[0], exgrmap.shape[1]))

#define out_meta using the meta dat from one of the original tif files
imgmeta = rasterio.open('612_cRGB_Red.tif')
out_meta = imgmeta.meta

# save reshaped VI's as tif files
with rasterio.open('rgb_exgsave.tif', 'w', **out_meta) as dest:
   dest.write(ExG)
with rasterio.open('rgb_civesave.tif', 'w', **out_meta) as dest:
   dest.write(CIVE)
with rasterio.open('rgb_exgrsave.tif', 'w', **out_meta) as dest:
   dest.write(ExGR)

# calculate mean for ExG Masked 612 VI's
ExG_ExGct = []
ExGR_ExGRct = []
CIVE_CIVEct = []


ID = []

for i in range(80):
```

```
    with fiona.open('CRGB Grid 612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, out_image, 0).astype(float)
            out_image = np.where(out_image>=45, out_image, 0).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))

        with rasterio.open('rgb_exgrsave.tif', 'r') as exgrsave:
            out_image, out_transform = rasterio.mask.mask(exgrsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=65, out_image, 0).astype(float)
            ExGR_ExGRct.append(np.count_nonzero(out_image))

        with rasterio.open('rgb_civesave.tif', 'r') as civesave:
            out_image, out_transform = rasterio.mask.mask(civesave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=10, 225, out_image).astype(float)
            out_image = np.where(out_image>=175, out_image, 0).astype(float)
            CIVE_CIVEct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct, ExGR_ExGRct, CIVE_CIVEct])
df = df.T
df.columns = ['ID', 'ExG_ExGct', 'ExGR_ExGRct', 'CIVE_CIVEct']

# export data frame to excel file
df.to_excel('Output/RGB_Masked_VI_612.xlsx', index = False)
```

```
df

# Check grid size
# calculate ExG Masked 612 VI's
ExG_ExGct = []

ID = []

for i in range(80):
    with fiona.open('CRGB Grid 612.shp', 'r') as shapefile:
        shape = [[feature['geometry'] for feature in shapefile][i]]
        feature = [feature for feature in shapefile][i]
        idx = feature ['properties']['Grid']
        ID.append(idx)

        with rasterio.open('rgb_exgsave.tif', 'r') as exgsave:
            out_image, out_transform = rasterio.mask.mask(exgsave, shape, nodata=0, crop=True)
            out_image = np.where(out_image<=175, 225, out_image).astype(float)
            ExG_ExGct.append(np.count_nonzero(out_image))



#combine the means of th VI's into one data frame
df = DataFrame([ID, ExG_ExGct])
df = df.T
df.columns = ['ID', 'ExG_ExGct']

df
```

Appendix C: Statistical Analysis RStudio Code

```{r setup, include=FALSE}
library(knitr)
library(tidyverse)
library(lme4)
library(car)
library(broom)
library(lmerTest)
library(performance)
library(GGally)
library(MuMIn)
library(optimx)
library(caret)
```

```{r}
# Data import

data_multi <- read_csv("Multi Data.csv") %>%
  mutate(name=factor(name),
         bloc=factor(bloc),
         Flight=factor(Flight),
         HN=factor(HN))

data_rgb <- read_csv("RGB Data.csv") %>%
  mutate(name=factor(name),
         bloc=factor(bloc),
         Flight=factor(Flight),
         HN=factor(HN))

data_multi_518 <- read_csv("Multi Data.csv") %>%
  mutate(name=factor(name),
         bloc=factor(bloc),
```

```
        Flight=factor(Flight),
        HN=factor(HN)) %>%
  filter(Flight=="518")

data_rgb_518 <- read_csv("RGB Data.csv") %>%
  mutate(name=factor(name),
        bloc=factor(bloc),
        Flight=factor(Flight),
        HN=factor(HN))%>%
  filter(Flight=="518")

data_multi_529 <- read_csv("Multi Data.csv") %>%
  mutate(name=factor(name),
        bloc=factor(bloc),
        Flight=factor(Flight),
        HN=factor(HN)) %>%
  filter(Flight=="529")

data_rgb_529 <- read_csv("RGB Data.csv") %>%
  mutate(name=factor(name),
        bloc=factor(bloc),
        Flight=factor(Flight),
        HN=factor(HN))%>%
  filter(Flight=="529")

data_multi_612 <- read_csv("Multi Data.csv") %>%
  mutate(name=factor(name),
        bloc=factor(bloc),
        Flight=factor(Flight),
        HN=factor(HN)) %>%
  filter(Flight=="612")

data_rgb_612 <- read_csv("RGB Data.csv") %>%
```

```
  mutate(name=factor(name),
         bloc=factor(bloc),
         Flight=factor(Flight),
         HN=factor(HN))%>%
  filter(Flight=="612")


data_rgb_425 <- read_csv("RGB Data.csv") %>%
  mutate(name=factor(name),
         bloc=factor(bloc),
         Flight=factor(Flight),
         HN=factor(HN))%>%
  filter(Flight=="425")


```

```` {r, echo = FALSE}
# Check significance between genotypes
# All flights Anova

Stand_mod <- lm(Stand~name + bloc, data = data_multi)
Anova(Stand_mod) %>%
  kable(digits=5, booktabs=T, caption = "Winter Survival Anova All Flights" )

Stand_mod_518 <- lm(Stand~name + bloc, data = data_multi_518)
Anova(Stand_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 Winter Survival Anova" )

Stand_mod_529 <- lm(Stand~name + bloc, data = data_multi_529)
Anova(Stand_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 Winter Survival Anova" )
```

```
Stand_mod_612 <- lm(Stand~name + bloc, data = data_multi_612)
Anova(Stand_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 Winter Survival Anova" )

ndvi_mod <- lm(NDVImu~name+bloc, data = data_multi)
Anova(ndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "NDVI Anova" )

ndre_mod <- lm(NDREmu~name+bloc, data = data_multi)
Anova(ndre_mod) %>%
  kable(digits=5, booktabs=T, caption = "NDRE Anova" )

gndvi_mod <- lm(GNDVImu~name+bloc, data = data_multi)
Anova(gndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "GNDVI Anova" )

rdvi_mod <- lm(RDVImu~name+bloc, data = data_multi)
Anova(rdvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "RDVI Anova" )

rgbvi_mod <- lm(RGBVImu~name+bloc, data = data_multi)
Anova(rgbvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "RGBVI Anova" )

exg_mod <- lm(ExG_ExGct~name+bloc, data = data_multi)
Anova(exg_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG Anova" )

exg_ndvi_mod <- lm(ExG_NDVImu~name+bloc, data = data_multi)
Anova(exg_ndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG_NDVI Anova" )

exg_ndre_mod <- lm(ExG_NDREmu~name+bloc, data = data_multi)
```

```
Anova(exg_ndre_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG_NDRE Anova" )

exg_gndvi_mod <- lm(ExG_GNDVImu~name+bloc, data = data_multi)
Anova(exg_gndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG_GNDVI Anova" )

exg_rdvi_mod <- lm(ExG_RDVImu~name+bloc, data = data_multi)
Anova(exg_rdvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG_RDVI Anova" )

exg_rgbvi_mod <- lm(ExG_RGBVImu~name+bloc, data = data_multi)
Anova(exg_rgbvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExG_RGBVI Anova" )

exgr_mod <- lm(ExGR_ExGRct~name+bloc, data = data_multi)
Anova(exgr_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExGR Anova" )

exgr_ndvi_mod <- lm(ExGR_NDVImu~name+bloc, data = data_multi)
Anova(exgr_ndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExGR_NDVI Anova" )

exgr_ndre_mod <- lm(ExGR_NDREmu~name+bloc, data = data_multi)
Anova(exgr_ndre_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExGR_NDRE Anova" )

exgr_gndvi_mod <- lm(ExGR_GNDVImu~name+bloc, data = data_multi)
Anova(exgr_gndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExGR_GNDVI Anova" )

exgr_rdvi_mod <- lm(ExGR_RDVImu~name+bloc, data = data_multi)
Anova(exgr_rdvi_mod) %>%
```

```
  kable(digits=5, booktabs=T, caption = "ExGR_RDVI Anova" )

exgr_rgbvi_mod <- lm(ExGR_RGBVImu~name+bloc, data = data_multi)
Anova(exgr_rgbvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "ExGR_RGBVI Anova" )

cive_mod <- lm(CIVE_CIVEct~name+bloc, data = data_multi)
Anova(cive_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE Anova" )

cive_ndvi_mod <- lm(CIVE_NDVImu~name+bloc, data = data_multi)
Anova(cive_ndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE_NDVI Anova" )

cive_ndre_mod <- lm(CIVE_NDREmu~name+bloc, data = data_multi)
Anova(cive_ndre_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE_NDRE Anova" )

cive_gndvi_mod <- lm(CIVE_GNDVImu~name+bloc, data = data_multi)
Anova(cive_gndvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE_GNDVI Anova" )

cive_rdvi_mod <- lm(CIVE_RDVImu~name+bloc, data = data_multi)
Anova(cive_rdvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE_RDVI Anova" )

cive_rgbvi_mod <- lm(CIVE_RGBVImu~name+bloc, data = data_multi)
Anova(cive_rgbvi_mod) %>%
  kable(digits=5, booktabs=T, caption = "CIVE_RGBVI Anova" )

#### 518 Anova

ndvi_mod_518 <- lm(NDVImu~name+bloc, data = data_multi_518)
```

```
Anova(ndvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 NDVI Anova" )

ndre_mod_518 <- lm(NDREmu~name+bloc, data = data_multi_518)
Anova(ndre_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 NDRE Anova" )

gndvi_mod_518 <- lm(GNDVImu~name+bloc, data = data_multi_518)
Anova(gndvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 GNDVI Anova" )

rdvi_mod_518 <- lm(RDVImu~name+bloc, data = data_multi_518)
Anova(rdvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 RDVI Anova" )

rgbvi_mod_518 <- lm(RGBVImu~name+bloc, data = data_multi_518)
Anova(rgbvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 RGBVI Anova" )

exg_mod_518 <- lm(ExG_ExGct~name+bloc, data = data_multi_518)
Anova(exg_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 ExG Anova" )

exg_ndvi_mod_518 <- lm(ExG_NDVImu~name+bloc, data = data_multi_518)
Anova(exg_ndvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = " 518 ExG_NDVI Anova" )

exg_ndre_mod_518 <- lm(ExG_NDREmu~name+bloc, data = data_multi_518)
Anova(exg_ndre_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 ExG_NDRE Anova" )

exg_gndvi_mod_518 <- lm(ExG_GNDVImu~name+bloc, data = data_multi_518)
Anova(exg_gndvi_mod_518) %>%
```

```
 kable(digits=5, booktabs=T, caption = "518 ExG_GNDVI Anova" )

exg_rdvi_mod_518 <- lm(ExG_RDVImu~name+bloc, data = data_multi_518)
Anova(exg_rdvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExG_RDVI Anova" )

exg_rgbvi_mod_518 <- lm(ExG_RGBVImu~name+bloc, data = data_multi_518)
Anova(exg_rgbvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExG_RGBVI Anova" )

exgr_mod_518 <- lm(ExGR_ExGRct~name+bloc, data = data_multi_518)
Anova(exgr_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR Anova" )

exgr_ndvi_mod_518 <- lm(ExGR_NDVImu~name+bloc, data = data_multi_518)
Anova(exgr_ndvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR_NDVI Anova" )

exgr_ndre_mod_518 <- lm(ExGR_NDREmu~name+bloc, data = data_multi_518)
Anova(exgr_ndre_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR_NDRE Anova" )

exgr_gndvi_mod_518 <- lm(ExGR_GNDVImu~name+bloc, data = data_multi_518)
Anova(exgr_gndvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR_GNDVI Anova" )

exgr_rdvi_mod_518 <- lm(ExGR_RDVImu~name+bloc, data = data_multi_518)
Anova(exgr_rdvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR_RDVI Anova" )

exgr_rgbvi_mod_518 <- lm(ExGR_RGBVImu~name+bloc, data = data_multi_518)
Anova(exgr_rgbvi_mod_518) %>%
 kable(digits=5, booktabs=T, caption = "518 ExGR_RGBVI Anova" )
```

```
cive_mod_518 <- lm(CIVE_CIVEct~name+bloc, data = data_multi_518)
Anova(cive_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE Anova" )

cive_ndvi_mod_518 <- lm(CIVE_NDVImu~name+bloc, data = data_multi_518)
Anova(cive_ndvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE_NDVI Anova" )

cive_ndre_mod_518 <- lm(CIVE_NDREmu~name+bloc, data = data_multi_518)
Anova(cive_ndre_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE_NDRE Anova" )

cive_gndvi_mod_518 <- lm(CIVE_GNDVImu~name+bloc, data = data_multi_518)
Anova(cive_gndvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE_GNDVI Anova" )

cive_rdvi_mod_518 <- lm(CIVE_RDVImu~name+bloc, data = data_multi_518)
Anova(cive_rdvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE_RDVI Anova" )

cive_rgbvi_mod_518 <- lm(CIVE_RGBVImu~name+bloc, data = data_multi_518)
Anova(cive_rgbvi_mod_518) %>%
  kable(digits=5, booktabs=T, caption = "518 CIVE_RGBVI Anova" )

### 529 Anova

ndvi_mod_529 <- lm(NDVImu~name+bloc, data = data_multi_529)
Anova(ndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 NDVI Anova" )

ndre_mod_529 <- lm(NDREmu~name+bloc, data = data_multi_529)
Anova(ndre_mod_529) %>%
```

```
  kable(digits=5, booktabs=T, caption = "529 NDRE Anova" )

gndvi_mod_529 <- lm(GNDVImu~name+bloc, data = data_multi_529)
Anova(gndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 GNDVI Anova" )

rdvi_mod_529 <- lm(RDVImu~name+bloc, data = data_multi_529)
Anova(rdvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 RDVI Anova" )

rgbvi_mod_529 <- lm(RGBVImu~name+bloc, data = data_multi_529)
Anova(rgbvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 RGBVI Anova" )

exg_mod_529 <- lm(ExG_ExGct~name+bloc, data = data_multi_529)
Anova(exg_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExG Anova" )

exg_ndvi_mod_529 <- lm(ExG_NDVImu~name+bloc, data = data_multi_529)
Anova(exg_ndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = " 529 ExG_NDVI Anova" )

exg_ndre_mod_529 <- lm(ExG_NDREmu~name+bloc, data = data_multi_529)
Anova(exg_ndre_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExG_NDRE Anova" )

exg_gndvi_mod_529 <- lm(ExG_GNDVImu~name+bloc, data = data_multi_529)
Anova(exg_gndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExG_GNDVI Anova" )

exg_rdvi_mod_529 <- lm(ExG_RDVImu~name+bloc, data = data_multi_529)
Anova(exg_rdvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExG_RDVI Anova" )
```

```
exg_rgbvi_mod_529 <- lm(ExG_RGBVImu~name+bloc, data = data_multi_529)
Anova(exg_rgbvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExG_RGBVI Anova" )

exgr_mod_529 <- lm(ExGR_ExGRct~name+bloc, data = data_multi_529)
Anova(exgr_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR Anova" )

exgr_ndvi_mod_529 <- lm(ExGR_NDVImu~name+bloc, data = data_multi_529)
Anova(exgr_ndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR_NDVI Anova" )

exgr_ndre_mod_529 <- lm(ExGR_NDREmu~name+bloc, data = data_multi_529)
Anova(exgr_ndre_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR_NDRE Anova" )

exgr_gndvi_mod_529 <- lm(ExGR_GNDVImu~name+bloc, data = data_multi_529)
Anova(exgr_gndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR_GNDVI Anova" )

exgr_rdvi_mod_529 <- lm(ExGR_RDVImu~name+bloc, data = data_multi_529)
Anova(exgr_rdvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR_RDVI Anova" )

exgr_rgbvi_mod_529 <- lm(ExGR_RGBVImu~name+bloc, data = data_multi_529)
Anova(exgr_rgbvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 ExGR_RGBVI Anova" )

cive_mod_529 <- lm(CIVE_CIVEct~name+bloc, data = data_multi_529)
Anova(cive_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE Anova" )
```

```
cive_ndvi_mod_529 <- lm(CIVE_NDVImu~name+bloc, data = data_multi_529)
Anova(cive_ndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE_NDVI Anova" )

cive_ndre_mod_529 <- lm(CIVE_NDREmu~name+bloc, data = data_multi_529)
Anova(cive_ndre_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE_NDRE Anova" )

cive_gndvi_mod_529 <- lm(CIVE_GNDVImu~name+bloc, data = data_multi_529)
Anova(cive_gndvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE_GNDVI Anova" )

cive_rdvi_mod_529 <- lm(CIVE_RDVImu~name+bloc, data = data_multi_529)
Anova(cive_rdvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE_RDVI Anova" )

cive_rgbvi_mod_529 <- lm(CIVE_RGBVImu~name+bloc, data = data_multi_529)
Anova(cive_rgbvi_mod_529) %>%
  kable(digits=5, booktabs=T, caption = "529 CIVE_RGBVI Anova" )

### 612 Anova

ndvi_mod_612 <- lm(NDVImu~name+bloc, data = data_multi_612)
Anova(ndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 NDVI Anova" )

ndre_mod_612 <- lm(NDREmu~name+bloc, data = data_multi_612)
Anova(ndre_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 NDRE Anova" )

gndvi_mod_612 <- lm(GNDVImu~name+bloc, data = data_multi_612)
Anova(gndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 GNDVI Anova" )
```

```
rdvi_mod_612 <- lm(RDVImu~name+bloc, data = data_multi_612)
Anova(rdvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 RDVI Anova" )

rgbvi_mod_612 <- lm(RGBVImu~name+bloc, data = data_multi_612)
Anova(rgbvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 RGBVI Anova" )

exg_mod_612 <- lm(ExG_ExGct~name+bloc, data = data_multi_612)
Anova(exg_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExG Anova" )

exg_ndvi_mod_612 <- lm(ExG_NDVImu~name+bloc, data = data_multi_612)
Anova(exg_ndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = " 612 ExG_NDVI Anova" )

exg_ndre_mod_612 <- lm(ExG_NDREmu~name+bloc, data = data_multi_612)
Anova(exg_ndre_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExG_NDRE Anova" )

exg_gndvi_mod_612 <- lm(ExG_GNDVImu~name+bloc, data = data_multi_612)
Anova(exg_gndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExG_GNDVI Anova" )

exg_rdvi_mod_612 <- lm(ExG_RDVImu~name+bloc, data = data_multi_612)
Anova(exg_rdvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExG_RDVI Anova" )

exg_rgbvi_mod_612 <- lm(ExG_RGBVImu~name+bloc, data = data_multi_612)
Anova(exg_rgbvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExG_RGBVI Anova" )
```

```
exgr_mod_612 <- lm(ExGR_ExGRct~name+bloc, data = data_multi_612)
Anova(exgr_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR Anova" )

exgr_ndvi_mod_612 <- lm(ExGR_NDVImu~name+bloc, data = data_multi_612)
Anova(exgr_ndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR_NDVI Anova" )

exgr_ndre_mod_612 <- lm(ExGR_NDREmu~name+bloc, data = data_multi_612)
Anova(exgr_ndre_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR_NDRE Anova" )

exgr_gndvi_mod_612 <- lm(ExGR_GNDVImu~name+bloc, data = data_multi_612)
Anova(exgr_gndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR_GNDVI Anova" )

exgr_rdvi_mod_612 <- lm(ExGR_RDVImu~name+bloc, data = data_multi_612)
Anova(exgr_rdvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR_RDVI Anova" )

exgr_rgbvi_mod_612 <- lm(ExGR_RGBVImu~name+bloc, data = data_multi_612)
Anova(exgr_rgbvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 ExGR_RGBVI Anova" )

cive_mod_612 <- lm(CIVE_CIVEct~name+bloc, data = data_multi_612)
Anova(cive_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE Anova" )

cive_ndvi_mod_612 <- lm(CIVE_NDVImu~name+bloc, data = data_multi_612)
Anova(cive_ndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE_NDVI Anova" )

cive_ndre_mod_612 <- lm(CIVE_NDREmu~name+bloc, data = data_multi_612)
```

```
Anova(cive_ndre_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE_NDRE Anova" )

cive_gndvi_mod_612 <- lm(CIVE_GNDVImu~name+bloc, data = data_multi_612)
Anova(cive_gndvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE_GNDVI Anova" )

cive_rdvi_mod_612 <- lm(CIVE_RDVImu~name+bloc, data = data_multi_612)
Anova(cive_rdvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE_RDVI Anova" )

cive_rgbvi_mod_612 <- lm(CIVE_RGBVImu~name+bloc, data = data_multi_612)
Anova(cive_rgbvi_mod_612) %>%
  kable(digits=5, booktabs=T, caption = "612 CIVE_RGBVI Anova" )

### RGB Anova

Stand_mod_RGB <- lm(Stand~name + bloc, data = data_rgb)
Anova(Stand_mod_RGB) %>%
  kable(digits=5, booktabs=T, caption = "RGB Winter Survival Anova All Flights" )

Stand_mod_RGB_425 <- lm(Stand~name + bloc, data = data_rgb_425)
Anova(Stand_mod_RGB_425) %>%
  kable(digits=5, booktabs=T, caption = "RGB 518 Winter Survival Anova" )

Stand_mod_RGB_518 <- lm(Stand~name + bloc, data = data_rgb_518)
Anova(Stand_mod_RGB_518) %>%
  kable(digits=5, booktabs=T, caption = "RGB 518 Winter Survival Anova" )

Stand_mod_RGB_529 <- lm(Stand~name + bloc, data = data_rgb_529)
Anova(Stand_mod_RGB_529) %>%
  kable(digits=5, booktabs=T, caption = "RGB 529 Winter Survival Anova" )
```

```
Stand_mod_RGB_612 <- lm(Stand~name + bloc, data = data_rgb_612)
Anova(Stand_mod_RGB_612) %>%
  kable(digits=5, booktabs=T, caption = "RGB 612 Winter Survival Anova" )

exg_mod_RGB_425 <- lm(ExG_ExGct~name+bloc, data = data_rgb_425)
Anova(exg_mod_RGB_425) %>%
  kable(digits=5, booktabs=T, caption = "RGB 425 ExG Anova" )

exgr_mod_RGB_425 <- lm(ExGR_ExGRct~name+bloc, data = data_rgb_425)
Anova(exgr_mod_RGB_425) %>%
  kable(digits=5, booktabs=T, caption = "RGB 425 ExGR Anova" )

cive_mod_RGB_425 <- lm(CIVE_CIVEct~name+bloc, data = data_rgb_425)
Anova(cive_mod_RGB_425) %>%
  kable(digits=5, booktabs=T, caption = "RGB 425 CIVE Anova" )


exg_mod_RGB_518 <- lm(ExG_ExGct~name+bloc, data = data_rgb_518)
Anova(exg_mod_RGB_518) %>%
  kable(digits=5, booktabs=T, caption = "RGB 518 ExG Anova" )

exgr_mod_RGB_518 <- lm(ExGR_ExGRct~name+bloc, data = data_rgb_518)
Anova(exgr_mod_RGB_518) %>%
  kable(digits=5, booktabs=T, caption = "RGB 518 ExGR Anova" )

cive_mod_RGB_518 <- lm(CIVE_CIVEct~name+bloc, data = data_rgb_518)
Anova(cive_mod_RGB_518) %>%
  kable(digits=5, booktabs=T, caption = "RGB 518 CIVE Anova" )


exg_mod_RGB_529 <- lm(ExG_ExGct~name+bloc, data = data_rgb_529)
Anova(exg_mod_RGB_529) %>%
  kable(digits=5, booktabs=T, caption = "RGB 529 ExG Anova" )
```

```
exgr_mod_RGB_529 <- lm(ExGR_ExGRct~name+bloc, data = data_rgb_529)
Anova(exgr_mod_RGB_529) %>%
  kable(digits=5, booktabs=T, caption = "RGB 529 ExGR Anova" )

cive_mod_RGB_529 <- lm(CIVE_CIVEct~name+bloc, data = data_rgb_529)
Anova(cive_mod_RGB_529) %>%
  kable(digits=5, booktabs=T, caption = "RGB 529 CIVE Anova" )



exg_mod_RGB_612 <- lm(ExG_ExGct~name+bloc, data = data_rgb_612)
Anova(exg_mod_RGB_612) %>%
  kable(digits=5, booktabs=T, caption = "RGB 612 ExG Anova" )

exgr_mod_RGB_612 <- lm(ExGR_ExGRct~name+bloc, data = data_rgb_612)
Anova(exgr_mod_RGB_612) %>%
  kable(digits=5, booktabs=T, caption = "RGB 612 ExGR Anova" )

cive_mod_RGB_612 <- lm(CIVE_CIVEct~name+bloc, data = data_rgb_612)
Anova(cive_mod_RGB_612) %>%
  kable(digits=5, booktabs=T, caption = "RGB 612 CIVE Anova" )

```


```{r}
# Combined Model for Stand Multi Spec
multi_mixed_mod <- lm(Stand~NDVImu*Flight + NDREmu*Flight + GNDVImu*Flight +
RDVImu*Flight + RGBVImu*Flight + ExG_NDVImu*Flight + ExG_NDREmu*Flight +
ExG_GNDVImu*Flight + ExG_RDVImu*Flight + I(ExG_ExGct/81300)*Flight + ExGR_NDVImu*Flight
+ ExGR_NDREmu*Flight + ExGR_GNDVImu*Flight + ExGR_RDVImu*Flight +
ExGR_RGBVImu*Flight + I(ExGR_ExGRct/81300)*Flight + CIVE_NDVImu*Flight +
CIVE_NDREmu*Flight + CIVE_GNDVImu*Flight + CIVE_RDVImu*Flight + CIVE_RGBVImu*Flight
+ I(CIVE_CIVEct/81300)*Flight,
                      data = data_multi)
```

```
multi_mixed_518_mod <- lm(Stand~NDVImu + NDREmu + GNDVImu + RDVImu + RGBVImu +
ExG_NDVImu + ExG_NDREmu + ExG_GNDVImu + ExG_RDVImu + I(ExG_ExGct/81300) +
ExG_RGBVImu + ExGR_NDVImu + ExGR_NDREmu + ExGR_GNDVImu + ExGR_RDVImu +
ExGR_RGBVImu + I(ExGR_ExGRct/81300) + CIVE_NDVImu + CIVE_NDREmu + CIVE_GNDVImu
+ CIVE_RDVImu + CIVE_RGBVImu + I(CIVE_CIVEct/81300),

                    data = data_multi_518)


multi_mixed_529_mod <- lm(Stand~NDVImu + NDREmu + GNDVImu + RDVImu + RGBVImu +
ExG_NDVImu + ExG_NDREmu + ExG_GNDVImu + ExG_RDVImu + I(ExG_ExGct/81300) +
ExG_RGBVImu + ExGR_NDVImu + ExGR_NDREmu + ExGR_GNDVImu + ExGR_RDVImu +
ExGR_RGBVImu + I(ExGR_ExGRct/81300) + CIVE_NDVImu + CIVE_NDREmu + CIVE_GNDVImu
+ CIVE_RDVImu + CIVE_RGBVImu + I(CIVE_CIVEct/81300),

                    data = data_multi_529)


multi_mixed_612_mod <- lm(Stand~NDVImu + NDREmu + GNDVImu + RDVImu + RGBVImu +
ExG_NDVImu + ExG_NDREmu + ExG_GNDVImu + ExG_RDVImu + I(ExG_ExGct/81300) +
ExG_RGBVImu + ExGR_NDVImu + ExGR_NDREmu + ExGR_GNDVImu + ExGR_RDVImu +
ExGR_RGBVImu + I(ExGR_ExGRct/81300) + CIVE_NDVImu + CIVE_NDREmu + CIVE_GNDVImu
+ CIVE_RDVImu + CIVE_RGBVImu + I(CIVE_CIVEct/81300),

                    data = data_multi_612)


Anova(multi_mixed_mod) %>%

  kable(digits=5, booktabs=T, caption = "Anova Mixed" )


Anova(multi_mixed_518_mod) %>%

  kable(digits=5, booktabs=T, caption = "Anova 518 Mixed" )


Anova(multi_mixed_529_mod) %>%

  kable(digits=5, booktabs=T, caption = "Anova 529 Mixed" )


Anova(multi_mixed_612_mod) %>%

  kable(digits=5, booktabs=T, caption = "Anova 612 Mixed" )
```

```{r}


# Multi-SPec paired plots
```

```
ggpairs(data_multi, columns=c("Stand", "NDVImu", "NDREmu", "GNDVImu", "RDVImu",
"RGBVImu")) +
  ggtitle("Unmasked Pairs Plots")


ggpairs(data_multi, columns=c("Stand","ExG_NDVImu", "ExG_NDREmu", "ExG_GNDVImu",
"ExG_RDVImu", "ExG_RGBVImu", "ExG_ExGct")) +
  ggtitle("ExG Pairs Plots")


ggpairs(data_multi, columns=c("Stand", "ExGR_NDVImu", "ExGR_NDREmu", "ExGR_GNDVImu",
"ExGR_RDVImu", "ExG_RGBVImu", "ExGR_ExGRct")) +
  ggtitle("ExGR Pairs Plots")
ggpairs(data_multi, columns=c("Stand", "CIVE_NDVImu", "CIVE_NDREmu", "CIVE_GNDVImu",
"CIVE_RDVImu", "CIVE_RGBVImu",  "CIVE_CIVEct")) +
  ggtitle("CIVE Pairs Plots")
```

```{r}
# Multi-SPec paired plots by flight


ggpairs(data_multi_518, columns=c("Stand", "NDVImu", "NDREmu", "GNDVImu", "RDVImu",
"RGBVImu")) +
  ggtitle("518 Unmasked Pairs Plots")


ggpairs(data_multi_518, columns=c("Stand", "ExG_NDVImu", "ExG_NDREmu", "ExG_GNDVImu",
"ExG_RDVImu", "ExG_RGBVImu", "ExG_ExGct")) +
  ggtitle("518 ExG Pairs Plots")


ggpairs(data_multi_518, columns=c("Stand", "ExGR_NDVImu", "ExGR_NDREmu",
"ExGR_GNDVImu", "ExGR_RDVImu", "ExGR_RGBVImu", "ExGR_ExGRct")) +
  ggtitle("518 ExGR Pairs Plots")


ggpairs(data_multi_518, columns=c("Stand", "CIVE_NDVImu", "CIVE_NDREmu", "CIVE_GNDVImu",
"CIVE_RDVImu", "CIVE_RGBVImu", "CIVE_CIVEct")) +
  ggtitle("518 CIVE Pairs Plots")


ggpairs(data_multi_529, columns=c("Stand", "NDVImu", "NDREmu", "GNDVImu", "RDVImu",
"RGBVImu")) +
  ggtitle("529 Unmasked Pairs Plots")
```

```
ggpairs(data_multi_529, columns=c("Stand", "ExG_NDVImu", "ExG_NDREmu", "ExG_GNDVImu",
"ExG_RDVImu", "ExG_RGBVImu", "ExG_ExGct")) +

  ggtitle("529 ExG Pairs Plots")


ggpairs(data_multi_529, columns=c("Stand", "ExGR_NDVImu", "ExGR_NDREmu",
"ExGR_GNDVImu", "ExGR_RDVImu", "ExGR_RGBVImu", "ExGR_ExGRct")) +

  ggtitle("529 ExGR Pairs Plots")


ggpairs(data_multi_529, columns=c("Stand", "CIVE_NDVImu", "CIVE_NDREmu", "CIVE_GNDVImu",
"CIVE_RDVImu", "CIVE_RGBVImu", "CIVE_CIVEct")) +

  ggtitle("529 CIVE Pairs Plots")


ggpairs(data_multi_612, columns=c("Stand", "NDVImu", "NDREmu", "GNDVImu", "RDVImu",
"RGBVImu")) +

  ggtitle("612 Unmasked Pairs Plots")


ggpairs(data_multi_612, columns=c("Stand", "ExG_NDVImu", "ExG_NDREmu", "ExG_GNDVImu",
"ExG_RDVImu", "ExG_RGBVImu", "ExG_ExGct")) +

  ggtitle("612 ExG Pairs Plots")


ggpairs(data_multi_612, columns=c("Stand", "ExGR_NDVImu", "ExGR_NDREmu",
"ExGR_GNDVImu", "ExGR_RDVImu", "ExGR_RGBVImu", "ExGR_ExGRct")) +

  ggtitle("612 ExGR Pairs Plots")


ggpairs(data_multi_612, columns=c("Stand", "CIVE_NDVImu", "CIVE_NDREmu", "CIVE_GNDVImu",
"CIVE_RDVImu", "CIVE_RGBVImu", "CIVE_CIVEct")) +

  ggtitle("612 CIVE Pairs Plots")
```

```{r}

# Investigate Colinearity issue

ggpairs(data_multi_518, columns=c("NDVImu", "NDREmu", "GNDVImu", "RDVImu",
"ExG_RGBVImu", "RGBVImu")) +

  ggtitle("Collinearity Pairs Plots")

ggpairs(data_multi_518, columns=c("ExG_NDVImu", "ExG_NDREmu", "ExG_GNDVImu",
"ExG_RDVImu", "ExG_ExGct", "ExG_RGBVImu", "RGBVImu")) +

  ggtitle("Collinearity Pairs Plots")
```

```
ggpairs(data_multi_518, columns=c("ExGR_NDVImu", "ExGR_NDREmu", "ExGR_GNDVImu",
"ExGR_RDVImu", "ExGR_RGBVImu", "ExGR_ExGRct", "ExG_RGBVImu", "RGBVImu")) +
  ggtitle("Collinearity Pairs Plots")
ggpairs(data_multi_518, columns=c( "CIVE_NDVImu", "CIVE_NDREmu", "CIVE_GNDVImu",
"CIVE_RDVImu", "CIVE_RGBVImu", "CIVE_CIVEct", "ExG_RGBVImu", "RGBVImu")) +
  ggtitle("Collinearity Pairs Plots")
```

```
```

```
```{r}
# RGB Models All Flights
rgb_mixed_mod <- lm(Stand~I(ExG_ExGct/638250)*Flight + I(ExGR_ExGRct/638250)*Flight +
I(CIVE_CIVEct/638250)*Flight,
                    data = data_rgb)

# Anova across all extractions
Anova(rgb_mixed_mod) %>%
  kable(digits=5, booktabs=T, caption = "Anova RGB" )
```

```
```{r}
# RGB Models by Flight
rgb_425_mod <- lm(Stand~I(ExG_ExGct/638250)+ I(ExGR_ExGRct/638250) + I(CIVE_CIVEct/638250),
                    data = data_rgb_425)

rgb_518_mod <- lm(Stand~I(ExG_ExGct/638250) + I(ExGR_ExGRct/638250) +
I(CIVE_CIVEct/638250),
                  data = data_rgb_518)

rgb_529_mod <- lm(Stand~I(ExG_ExGct/638250) + I(ExGR_ExGRct/638250) +
I(CIVE_CIVEct/638250),
                    data = data_rgb_529)

rgb_612_mod <- lm(Stand~I(ExG_ExGct/638250) + I(ExGR_ExGRct/638250) +
I(CIVE_CIVEct/638250),
                    data = data_rgb_612)

# Anova across all flights extractions
```

```
Anova(rgb_425_mod) %>%
  kable(digits=9, booktabs=T, caption = "Anova 425 RGB" )
Anova(rgb_518_mod) %>%
  kable(digits=5, booktabs=T, caption = "Anova 518 RGB" )
Anova(rgb_529_mod) %>%
  kable(digits=5, booktabs=T, caption = "Anova 529 RGB" )
Anova(rgb_612_mod) %>%
  kable(digits=5, booktabs=T, caption = "Anova 612 RGB" )
```

```
```{r}
# RGB paired plots

ggpairs(data_rgb, columns=c("Stand", "ExG_ExGct", "ExGR_ExGRct", "CIVE_CIVEct")) +
  ggtitle("RGB Pairs Plots")
```

```{r}
# RGB paired plots by flight

ggpairs(data_rgb_425, columns=c("Stand", "ExG_ExGct", "ExGR_ExGRct", "CIVE_CIVEct")) +
  ggtitle("425 RGB Pairs Plots")

ggpairs(data_rgb_518, columns=c("Stand", "ExG_ExGct", "ExGR_ExGRct", "CIVE_CIVEct")) +
  ggtitle("518 RGB Pairs Plots")

ggpairs(data_rgb_529, columns=c("Stand", "ExG_ExGct", "ExGR_ExGRct", "CIVE_CIVEct")) +
  ggtitle("529 RGB Pairs Plots")

ggpairs(data_rgb_612, columns=c("Stand", "ExG_ExGct", "ExGR_ExGRct", "CIVE_CIVEct")) +
  ggtitle("612 RGB Pairs Plots")
```
```

```
```

```{r}
#Prediction - Stand


#Trainig Models
multi_training_mod1 <- train(Stand~NDREmu*Flight + RGBVImu*Flight + ExG_NDREmu*Flight + ExGR_NDREmu*Flight + I(ExGR_ExGRct/81300)*Flight + CIVE_NDREmu*Flight,
                             data = data_multi,
                             method="lm",
                             trControl = trainControl(method = "cv"))


multi_training_mod2 <- train(Stand~NDREmu*Flight,
                            data = data_multi,
                             method="lm",
                             trControl = trainControl(method = "cv"))


multi_training_mod3 <- train(Stand~RGBVImu*Flight,
                             data = data_multi,
                             method="lm",
                             trControl = trainControl(method = "cv"))


multi_training_mod4 <- train(Stand~ExG_NDREmu*Flight,
                             data = data_multi,
                             method="lm",
                             trControl = trainControl(method = "cv"))


multi_training_mod5 <- train(Stand~ExGR_NDREmu*Flight,
                             data = data_multi,
                             method="lm",
                             trControl = trainControl(method = "cv"))


multi_training_mod6 <- train(Stand~I(ExGR_ExGRct/81300)*Flight,
                             data = data_multi,
                             method="lm",
```

```
                trControl = trainControl(method = "cv"))

multi_training_mod7 <- train(Stand~CIVE_NDREmu*Flight,
                data = data_multi,
                method="lm",
                trControl = trainControl(method = "cv"))

multi_training_mod8 <- train(Stand~NDREmu + CIVE_NDVImu,
                data = data_multi_518,
                method="lm",
                trControl = trainControl(method = "cv"))

multi_training_mod9 <- train(Stand~NDREmu,
                data = data_multi_518,
                method="lm",
                trControl = trainControl(method = "cv"))

multi_training_mod10 <- train(Stand~CIVE_NDVImu,
                data = data_multi_518,
                method="lm",
                trControl = trainControl(method = "cv"))

multi_training_mod11 <- train(Stand~GNDVImu + ExG_NDVImu + ExG_RGBVImu +
ExGR_RGBVImu + CIVE_NDVImu + CIVE_RGBVImu,
                data = data_multi_529,
                method="lm",
                trControl = trainControl(method = "cv"))
multi_training_mod12 <- train(Stand~GNDVImu,
                data = data_multi_529,
                method="lm",
                trControl = trainControl(method = "cv"))

multi_training_mod13 <- train(Stand~ExG_NDVImu,
                data = data_multi_529,
```

```
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod14 <- train(Stand~ExG_RGBVImu,
                    data = data_multi_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod15 <- train(Stand~ExGR_RGBVImu,
                    data = data_multi_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod16 <- train(Stand~CIVE_NDVImu,
                    data = data_multi_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod17 <- train(Stand~CIVE_RGBVImu,
                    data = data_multi_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod18 <- train(Stand~RGBVImu + ExG_GNDVImu + ExG_NDREmu + CIVE_NDREmu
+ CIVE_GNDVImu,
                    data = data_multi_612,
                    method="lm",
                    trControl = trainControl(method = "cv"))

multi_training_mod19 <- train(Stand~RGBVImu,
                    data = data_multi_612,
                    method="lm",
                    trControl = trainControl(method = "cv"))
```

```
multi_training_mod20 <- train(Stand~ExG_GNDVImu,
                      data = data_multi_612,
                      method="lm",
                      trControl = trainControl(method = "cv"))

multi_training_mod21 <- train(Stand~ExG_NDREmu,
                      data = data_multi_612,
                      method="lm",
                      trControl = trainControl(method = "cv"))

multi_training_mod22 <- train(Stand~CIVE_NDREmu,
                      data = data_multi_612,
                      method="lm",
                      trControl = trainControl(method = "cv"))

multi_training_mod23 <- train(Stand~CIVE_GNDVImu,
                      data = data_multi_612,
                      method="lm",
                      trControl = trainControl(method = "cv"))
```

```{r}
#Select Multi Spec Model
print('mod1')
multi_training_mod1
print('mod2')
multi_training_mod2
print('mod3')
multi_training_mod3
print('mod4')
multi_training_mod4
print('mod5')
```

```
multi_training_mod5
print('mod6')
multi_training_mod6
print('mod7')
multi_training_mod7
print('mod8')
multi_training_mod8
print('mod9')
multi_training_mod9
print('mod10')
multi_training_mod10
print('mod11')
multi_training_mod11
print('mod12')
multi_training_mod12
print('mod13')
multi_training_mod13
print('mod14')
multi_training_mod14
print('mod15')
multi_training_mod15
print('mod16')
multi_training_mod16
print('mod17')
multi_training_mod17
print('mod18')
multi_training_mod18
print('mod19')
multi_training_mod19
print('mod20')
multi_training_mod20
print('mod21')
multi_training_mod21
```

```
print('mod22')
multi_training_mod22
print('mod23')
multi_training_mod23
```

```{r}
#Check correlation
# Mod1
multi_flight_data_prediction1 <- predict(multi_training_mod1$finalModel)
cor(multi_flight_data_prediction1, data_multi$Stand, method= c("pearson"))

# Mod2
multi_flight_data_prediction2 <- predict(multi_training_mod2$finalModel)
cor(multi_flight_data_prediction2, data_multi$Stand, method= c("pearson"))

# Mod3
multi_flight_data_prediction3 <- predict(multi_training_mod3$finalModel)
cor(multi_flight_data_prediction3, data_multi$Stand, method= c("pearson"))

# Mod4
multi_flight_data_prediction4 <- predict(multi_training_mod4$finalModel)
cor(multi_flight_data_prediction4, data_multi$Stand, method= c("pearson"))

# Mod5
multi_flight_data_prediction5 <- predict(multi_training_mod5$finalModel)
cor(multi_flight_data_prediction5, data_multi$Stand, method= c("pearson"))

# Mod6
multi_flight_data_prediction6 <- predict(multi_training_mod6$finalModel)
cor(multi_flight_data_prediction6, data_multi$Stand, method= c("pearson"))

# Mod7
multi_flight_data_prediction7 <- predict(multi_training_mod7$finalModel)
```

```
cor(multi_flight_data_prediction7, data_multi$Stand, method= c("pearson"))

# Mod8
multi_flight_data_prediction8 <- predict(multi_training_mod8$finalModel)
cor(multi_flight_data_prediction8, data_multi_529$Stand, method= c("pearson"))

# Mod9
multi_flight_data_prediction9 <- predict(multi_training_mod9$finalModel)
cor(multi_flight_data_prediction9, data_multi_529$Stand, method= c("pearson"))

# Mod10
multi_flight_data_prediction10 <- predict(multi_training_mod10$finalModel)
cor(multi_flight_data_prediction10, data_multi_529$Stand, method= c("pearson"))

# Mod11
multi_flight_data_prediction11 <- predict(multi_training_mod11$finalModel)
cor(multi_flight_data_prediction11, data_multi_529$Stand, method= c("pearson"))

# Mod12
multi_flight_data_prediction12 <- predict(multi_training_mod12$finalModel)
cor(multi_flight_data_prediction12, data_multi_529$Stand, method= c("pearson"))

# Mod13
multi_flight_data_prediction13 <- predict(multi_training_mod13$finalModel)
cor(multi_flight_data_prediction13, data_multi_529$Stand, method= c("pearson"))

# Mod14
multi_flight_data_prediction14 <- predict(multi_training_mod14$finalModel)
cor(multi_flight_data_prediction14, data_multi_529$Stand, method= c("pearson"))

# Mod15
multi_flight_data_prediction15 <- predict(multi_training_mod15$finalModel)
cor(multi_flight_data_prediction15, data_multi_529$Stand, method= c("pearson"))
```

```
# Mod16
multi_flight_data_prediction16 <- predict(multi_training_mod16$finalModel)
cor(multi_flight_data_prediction16, data_multi_529$Stand, method= c("pearson"))

# Mod17
multi_flight_data_prediction17 <- predict(multi_training_mod17$finalModel)
cor(multi_flight_data_prediction17, data_multi_529$Stand, method= c("pearson"))

# Mod18
multi_flight_data_prediction18 <- predict(multi_training_mod18$finalModel)
cor(multi_flight_data_prediction18, data_multi_529$Stand, method= c("pearson"))

# Mod19
multi_flight_data_prediction19 <- predict(multi_training_mod19$finalModel)
cor(multi_flight_data_prediction19, data_multi_529$Stand, method= c("pearson"))

# Mod20
multi_flight_data_prediction20 <- predict(multi_training_mod20$finalModel)
cor(multi_flight_data_prediction20, data_multi_529$Stand, method= c("pearson"))

# Mod21
multi_flight_data_prediction21 <- predict(multi_training_mod21$finalModel)
cor(multi_flight_data_prediction21, data_multi_529$Stand, method= c("pearson"))

# Mod22
multi_flight_data_prediction22 <- predict(multi_training_mod22$finalModel)
cor(multi_flight_data_prediction22, data_multi_529$Stand, method= c("pearson"))

# Mod23
multi_flight_data_prediction23 <- predict(multi_training_mod23$finalModel)
cor(multi_flight_data_prediction23, data_multi_529$Stand, method= c("pearson"))
```

```
```

```{r}
# Prediction for RGB

rgb_training_mod1 <- train(Stand~I(ExG_ExGct/638250)*Flight + I(ExGR_ExGRct/638250)*Flight + I(CIVE_CIVEct/638250)*Flight,
                           data = data_rgb,
                           method="lm",
                           trControl = trainControl(method = "cv"))

rgb_training_mod2 <- train(Stand~I(ExG_ExGct/638250)*Flight,
                           data = data_rgb,
                           method="lm",
                           trControl = trainControl(method = "cv"))

rgb_training_mod3 <- train(Stand~I(ExGR_ExGRct/638250)*Flight,
                           data = data_rgb,
                           method="lm",
                           trControl = trainControl(method = "cv"))

rgb_training_mod4 <- train(Stand~I(CIVE_CIVEct/638250)*Flight,
                           data = data_rgb,
                           method="lm",
                           trControl = trainControl(method = "cv"))

rgb_training_mod5 <- train(Stand~I(ExG_ExGct/638250) + I(CIVE_CIVEct/638250),
                           data = data_rgb_425,
                           method="lm",
                           trControl = trainControl(method = "cv"))

rgb_training_mod6 <- train(Stand~I(ExG_ExGct/638250),
                           data = data_rgb_425,
                           method="lm",
```

```
                trControl = trainControl(method = "cv"))

rgb_training_mod7 <- train(Stand~I(CIVE_CIVEct/638250),
                data = data_rgb_425,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod8 <- train(Stand~I(ExGR_ExGRct/638250) + I(CIVE_CIVEct/638250),
                data = data_rgb_518,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod9 <- train(Stand~I(ExGR_ExGRct/638250),
                data = data_rgb_518,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod10 <- train(Stand~I(CIVE_CIVEct/638250),
                data = data_rgb_518,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod11 <- train(Stand~I(ExG_ExGct/638250) + I(ExGR_ExGRct/638250) +
I(CIVE_CIVEct/638250),
                data = data_rgb_529,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod12 <- train(Stand~I(ExG_ExGct/638250),
                data = data_rgb_529,
                method="lm",
                trControl = trainControl(method = "cv"))

rgb_training_mod13 <- train(Stand~I(ExGR_ExGRct/638250),
```

```
                    data = data_rgb_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

rgb_training_mod14 <- train(Stand~I(CIVE_CIVEct/638250),
                    data = data_rgb_529,
                    method="lm",
                    trControl = trainControl(method = "cv"))

rgb_training_mod15 <- train(Stand~I(CIVE_CIVEct/638250),
                    data = data_rgb_612,
                    method="lm",
                    trControl = trainControl(method = "cv"))
```

```{r}
print('mod1')
rgb_training_mod1
print('mod2')
rgb_training_mod2
print('mod3')
rgb_training_mod3
print('mod4')
rgb_training_mod4
print('mod5')
rgb_training_mod5
print('mod6')
rgb_training_mod6
print('mod7')
rgb_training_mod7
print('mod8')
rgb_training_mod8
```

```
print('mod9')
rgb_training_mod9
print('mod10')
rgb_training_mod10
print('mod11')
rgb_training_mod11
print('mod12')
rgb_training_mod12
print('mod13')
rgb_training_mod13
print('mod14')
rgb_training_mod14
print('mod15')
rgb_training_mod15
```

rgb_training_mod correlation

```{r}
#Check correlation
#1
rgb_flight_data_prediction1 <- predict(rgb_training_mod1$finalModel)
cor(rgb_flight_data_prediction1, data_rgb$Stand, method= c("pearson"))

#2
rgb_flight_data_prediction2 <- predict(rgb_training_mod2$finalModel)
cor(rgb_flight_data_prediction2, data_rgb$Stand, method= c("pearson"))

#3
rgb_flight_data_prediction3 <- predict(rgb_training_mod3$finalModel)
cor(rgb_flight_data_prediction3, data_rgb$Stand, method= c("pearson"))

#4
rgb_flight_data_prediction4 <- predict(rgb_training_mod4$finalModel)
```

```
cor(rgb_flight_data_prediction4, data_rgb$Stand, method= c("pearson"))

#5
rgb_flight_data_prediction5 <- predict(rgb_training_mod5$finalModel)
cor(rgb_flight_data_prediction5, data_rgb_425$Stand, method= c("pearson"))

#6
rgb_flight_data_prediction6 <- predict(rgb_training_mod6$finalModel)
cor(rgb_flight_data_prediction6, data_rgb_425$Stand, method= c("pearson"))

#7
rgb_flight_data_prediction7 <- predict(rgb_training_mod7$finalModel)
cor(rgb_flight_data_prediction7, data_rgb_425$Stand, method= c("pearson"))

#8
rgb_flight_data_prediction8 <- predict(rgb_training_mod8$finalModel)
cor(rgb_flight_data_prediction8, data_rgb_425$Stand, method= c("pearson"))

#9
rgb_flight_data_prediction9 <- predict(rgb_training_mod9$finalModel)
cor(rgb_flight_data_prediction9, data_rgb_425$Stand, method= c("pearson"))

#10
rgb_flight_data_prediction10 <- predict(rgb_training_mod10$finalModel)
cor(rgb_flight_data_prediction10, data_rgb_425$Stand, method= c("pearson"))

#11
rgb_flight_data_prediction11 <- predict(rgb_training_mod11$finalModel)
cor(rgb_flight_data_prediction11, data_rgb_425$Stand, method= c("pearson"))

#12
rgb_flight_data_prediction12 <- predict(rgb_training_mod12$finalModel)
cor(rgb_flight_data_prediction12, data_rgb_425$Stand, method= c("pearson"))
```

```
#13
rgb_flight_data_prediction13 <- predict(rgb_training_mod13$finalModel)
cor(rgb_flight_data_prediction13, data_rgb_425$Stand, method= c("pearson"))

#14
rgb_flight_data_prediction14 <- predict(rgb_training_mod14$finalModel)
cor(rgb_flight_data_prediction14, data_rgb_425$Stand, method= c("pearson"))

#15
rgb_flight_data_prediction15 <- predict(rgb_training_mod15$finalModel)
cor(rgb_flight_data_prediction15, data_rgb_425$Stand, method= c("pearson"))
```