Stand Treelist Imputation

Jacob Beard
jacob.beard@dnr.wa.gov

Kate McBurney, Jeff Ricklefs, Peter Gould, Jacob Strunk, et al.

Sustainable Harvest Calculation | WA

Generating a perpetual supply of revenue on state trust lands for trust beneficiaries requires responsible management with an emphasis on **long-term sustainability**. A major component of DNR's approach to sustainable management is calculation of a sustainable harvest level, which is **the volume of timber to be scheduled for sale during a planning decade** according to applicable laws, policies, and procedures (RCW 79.10.300)(5).

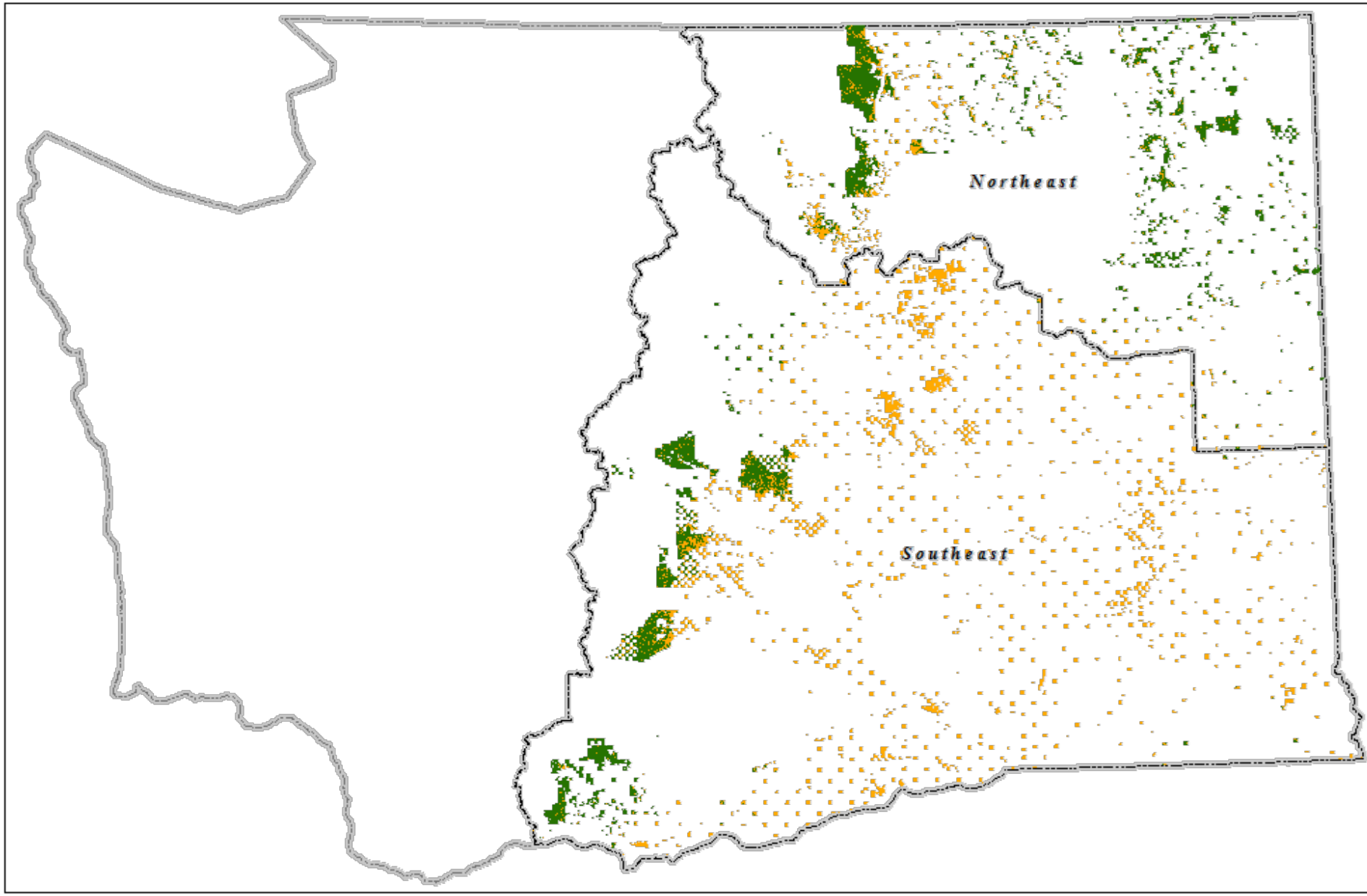
DNR is required to set a sustainable harvest level by Washington state law. Specifically, DNR must periodically adjust acreages designated for inclusion in the sustained yield management program and calculate a sustainable harvest level (RCW 79.10.320). Sustained yield means harvesting on a continual basis without major prolonged curtailment or cessation of harvest (RCW 79.10.310). The **sustainable harvest level is a policy decision** that requires approval from the Board of Natural Resources.

[S]eparate sustainable harvest level[s] for forested state trust lands located east and west of the Cascade Crest because growing conditions and management strategies [differ].

https://www.dnr.wa.gov/shc

DNR Managed EWA Lands

DNR Managed	Acres
Forested	756,000
Non-Forested	803,000
Total	1,559,000



FVS Growth Modeling

[Essential FVS: (usda.gov)]

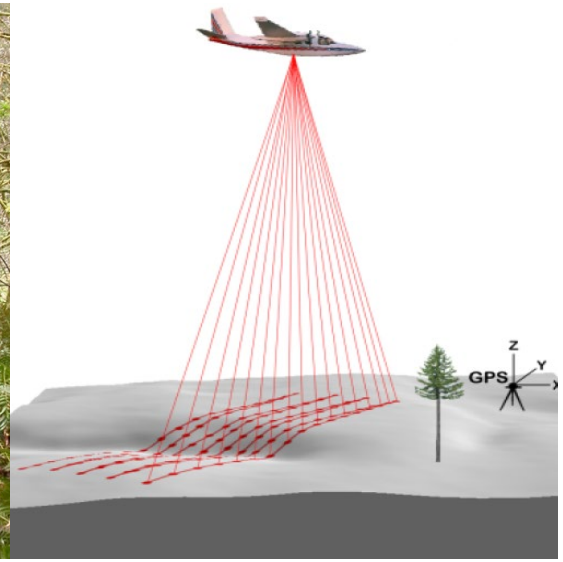
Variables that must be recorded for all trees:

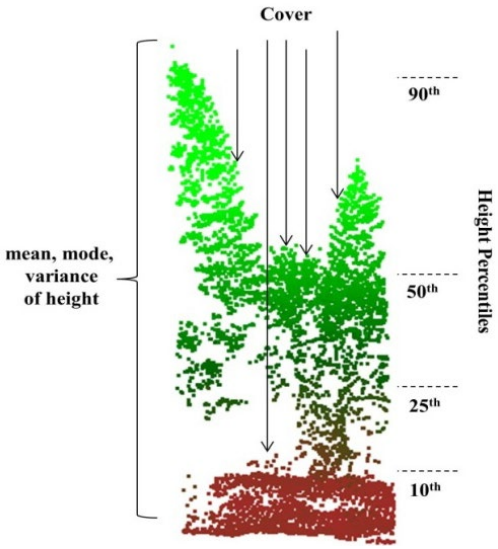
- Plot Identifier
- Species
- Diameter at Breast Height (DBH)

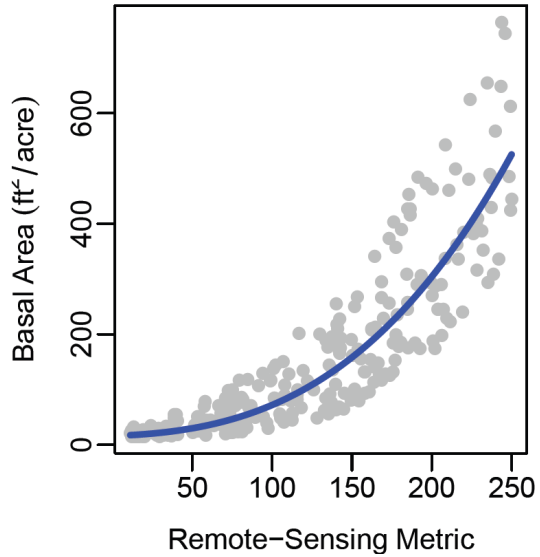
Other variables, however, serve to better describe unique site and tree characteristics and will improve the resolution of the projection.

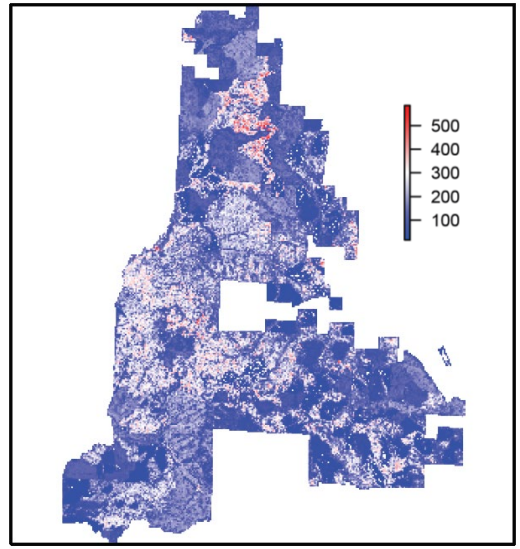
The model will accommodate up to 3000 individual tree records (per stand).

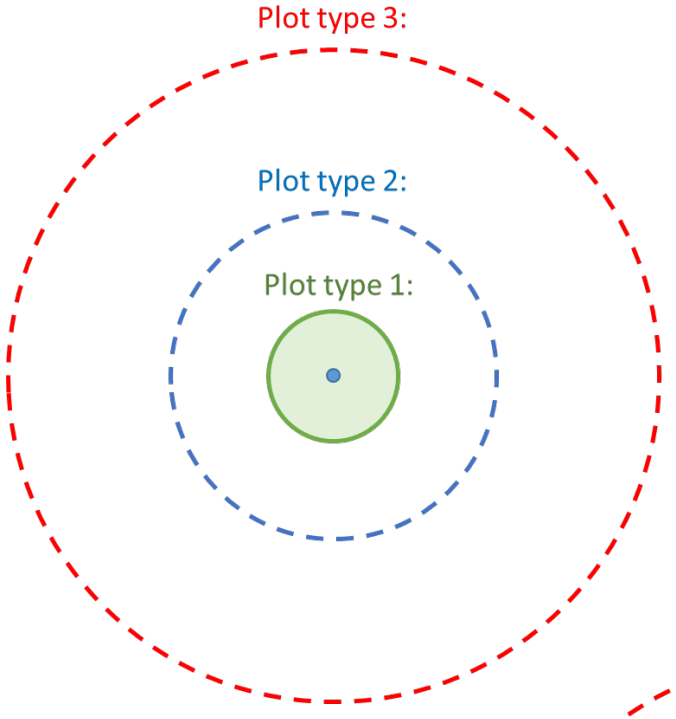
Remote Sensed - Forest Resource Inventory System (RS-FRIS)





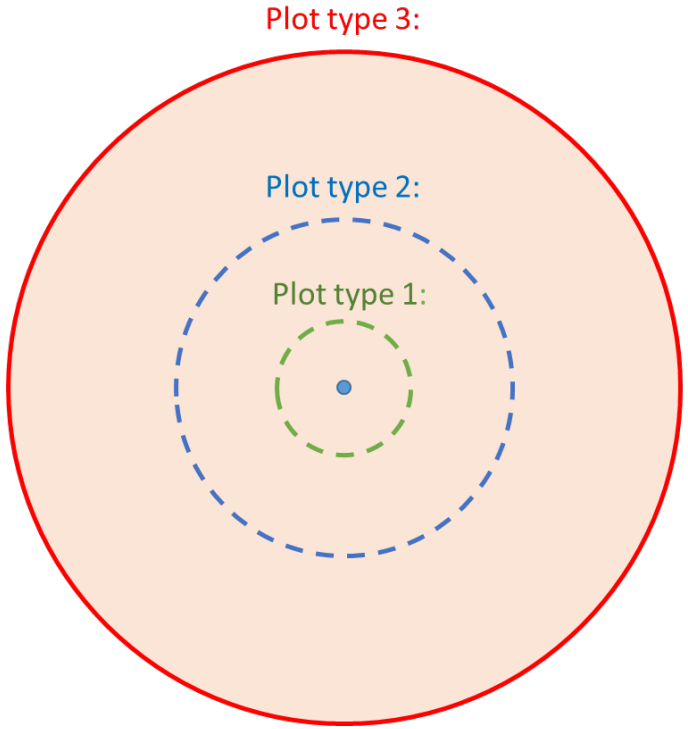






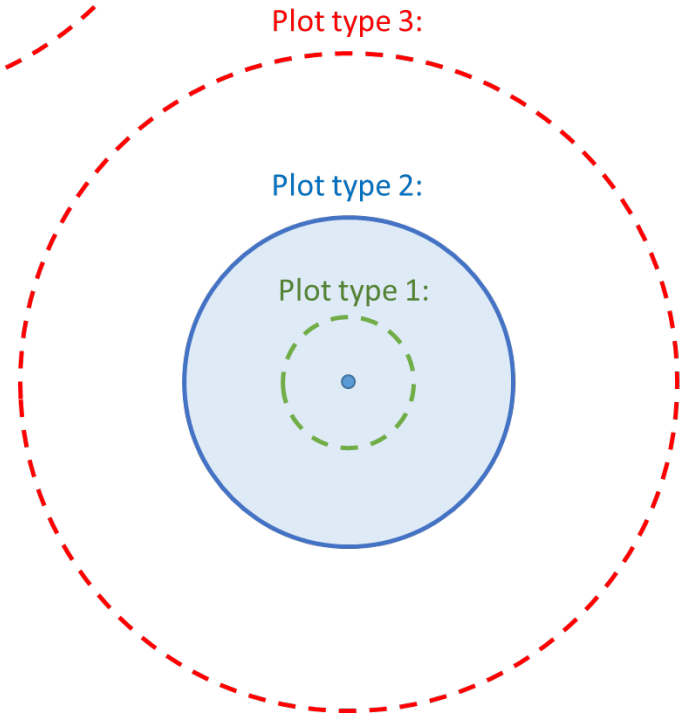
Plot type 1:

Radius	7.4 feet
Area	1/250th ac
Sample Trees	Live trees > 1 foot tall and < 2 inches dbh



Plot type 3:

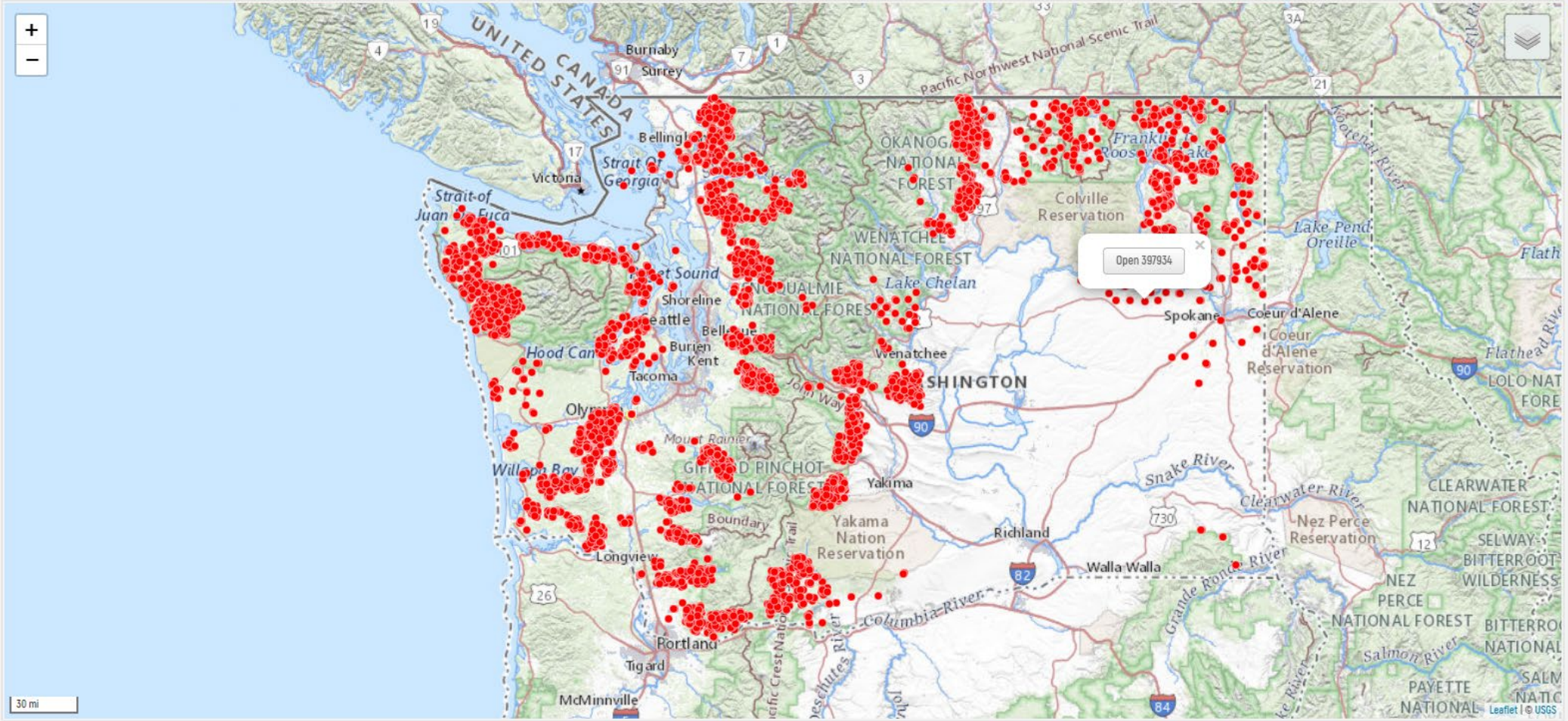
Radius	37.2 feet
Area	1/10th ac
Sample Trees	Live and dead trees ≥ 5.5 inches dbh



Plot type 2:

Radius	18.6 feet
Area	1/40th ac
Sample Trees	Live trees ≥ 2 inches dbh and < 5.5 inches dbh

RS-FRIS Plot Viewer



[Multiple Fields] | Search | Open Plot Page

RS-FRIS Plots

Date Measured	Plot Name	Region	District	Species (Primary)	BA (Basal Area)	Height: Largest Value	QMD (Quadratic Mean Diame...	DBH: Largest Live	TPA (Trees per Acre)
2018/08/29	116465	Pacific Cascade	Yacolt	DF	192.7	127	16.5	23.7	130.3
2018/08/28	397934	Northeast	Arcadia	PP	93.4	51	5.2	11.3	621.1
2018/08/28	113573	Pacific Cascade	Yacolt	DF	293.1	136	8.1	32.9	824.2
2018/08/28	114535	Pacific Cascade	Yacolt	DF	160.7	85	8.3	15.2	430.8
2018/08/28	127988	Pacific Cascade	Yacolt	DF	69.2	76	12.6	16.3	80.2
2018/08/28	127008	Pacific Cascade	Yacolt	DF	98.3	77.6	7.2	12.7	350.7
2018/08/28	129887	Pacific Cascade	Yacolt	DF	277.9	130	5.3	29.8	1847.1
2018/08/28	133728	Pacific Cascade	Yacolt	DF	1.4	4.7	0.7	1	506.4
2018/08/28	128956	Pacific Cascade	Yacolt	DF	193.7	89.1	3.7	16.6	2579.5

Record ID: 7e99618b-8de3-49b0-b24f-218964522c4b

2,820-2,828 of 4,612

RS-FRIS Plot Viewer

←Return to Statewide View

Plot Metrics

	Field	Value
+	Plot Name	513150
	Sample Date	2020-08-13 12:2...
	X	2444814.2
	Y	1198933.3
	Elevation	3973
+	Species (Primary)	DF
+	BA (Basal Area)	329.6
+	Volume: Board Foot (Gross)	51703.2
+	Canopy: Layer Count	1
+	Height: Largest Value	121
+	QMD (Quadratic Mean Diamet...	9.5
+	RD (Relative Density)	106.9
+	SDI: Sum (Stand Density Index)	561.7
+	Snags: DBH > 20" (Per Acre)	0
+	TPA (Trees per Acre)	671.3

Tree Records

All Fields | N Trees | Measurements | Volumes | BioCarb | Damages

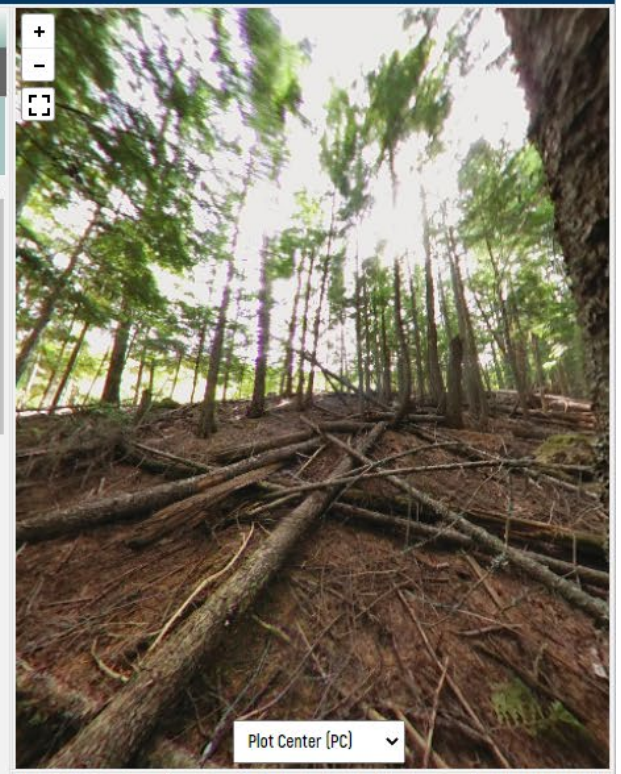
	N	PT	Sts	Sp	N Trees			Measurements				Volumes					Biomass & Carbon			Note
					Count	XFAC	TPA	DBH	Ht	HLC	CR	CFgt...	CFgm	CFnm	BFg	BFn	CFgb	LbAB	LbAC	
	1	3	1	WH	1	10.02	10.02	12.5	94	37	61	34	30	30	150	30	34	1378	689	
	2	2	1	DF	1	10.02	10.02	15.6	99	63	36	48	44	44	220	44	45	1911	955	
	3	2	1	DF	1	10.02	10.02	7.3	90	41	54	11	8							
	4	3	1	WL	1	10.02	10.02	17.6	115	72	37	68	63	6						
	5	3	0	DF	1	10.02	10.02	16.5	20		*0*	11	8	8	10	8	9	375	187	dead
	6	3	1	WL	1	10.02	10.02	15	120	44	63	54	50	50	250	50	50	2127	1063	
	7	3	1	RC	1	10.02	10.02	6.2	35	18	49	3	2	2	10	2	2	74	37	
	8	3	1	WL	1	10.02	10.02	9.7	87	46	47	17	15	15	80	15	15	667	334	
	9	3	1	RC	1	10.02	10.02	5.7	32	14	56	3					2	53	27	
	10	3	1	WL	1	10.02	10.02	15.3	121	82	32	56	52	52	270	52	53	2224	1112	
	11	2	1	WH	1	40.07	40.07	3	17	10	41	0						18	9	
	12	3	1	RC	1	10.02	10.02	10.9	60	13	78	16	13	13	60	13	15	423	212	
	13	3	0	DF	1	10.02	10.02	6.4	52		*0*	5	3	3	20	3	4	158	79	dead
	14	3	1	WL	1	10.02	10.02	13.4	109	75	31	39	36	36	180	36	37	1094	547	conks
	15	3	1	DF	1	10.02	10.02	15.9	110	72	35	55	51	51	270	51	52	2216	1108	
	16	3	1	WH	1	10.02	10.02	8.8	63	24	62	11	9	9	50	9	11	447	224	
	17	3	0	DF	1	10.02	10.02	8.8	25		*0*	4	3	3	10	3	3	147	74	dead
	18	3	0	DF	1	10.02	10.02	10.8	77		*0*	19	16	16	80	16	17	742	371	dead
	19	3	1	RC	1	10.02	10.02	7.5	41	13	68	6	4	4	20	4	5	138	69	
	20	3	1	WH	1	10.02	10.02	6.1	56	23	59	5	3	3	10	3	4	162	81	

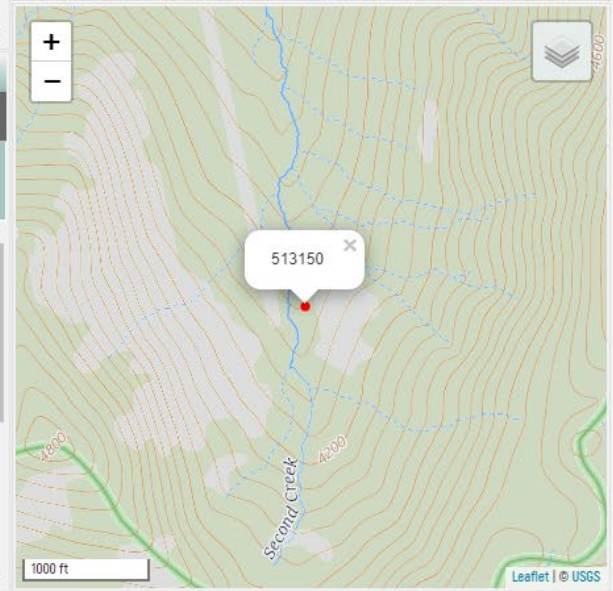
Merchantable cubic-foot gross volume.
This predicted value does not account for defect.
Units = cubic feet / acre.

DDWM Records

All Fields | Diameters

Az	N	Sp	DC	Length	Diameters			Gross Volume CuFt/Acre
					Transect	Small	Large	
120	1	OT	2	15	4	4	6	119
120	2	OT	4	27	7	5	9	366
120	3	OT	3	31	9	8	10	605
120	4	OT	3	14	5	4	6	187
120	5	OT	3	6	9	9	10	605
120	6	RC	4	25	13	11	16	1261
120	7	OT	4	5	6	5	6	269
120	8	OT	3	33	9	6	9	605
120	9	OT	3	16	6	4	6	269
120	10	OT	5	20	8	6	10	478
210	1	OT	5	16	4	4	5	119
210	2	OT	4	21	9	8	11	605





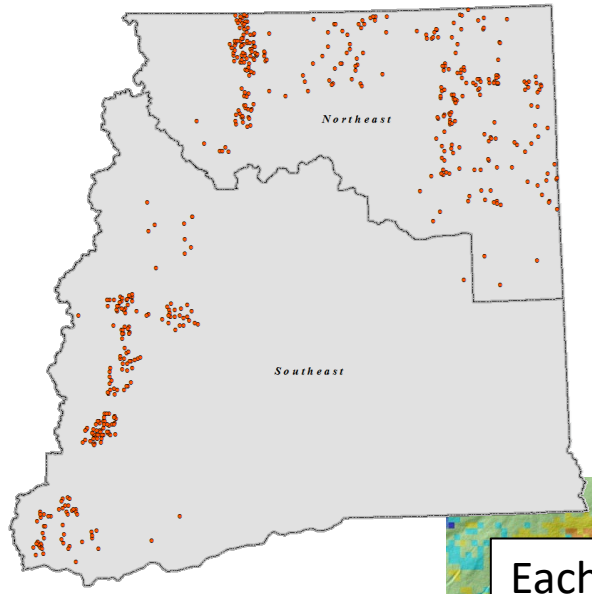
Components involved:

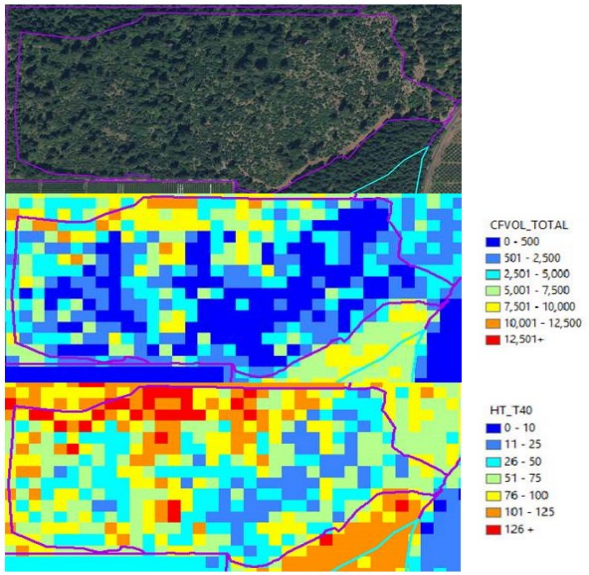
Tree Data from RS-FRIS Plots:

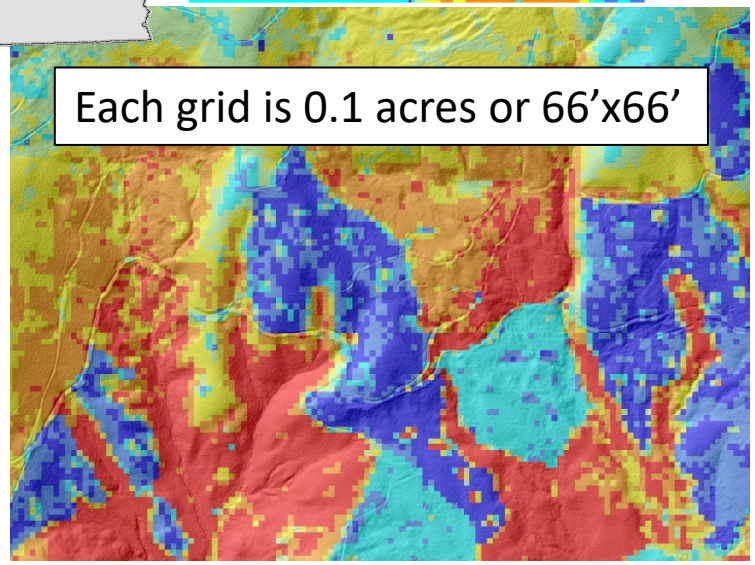
Species, DBH, Height

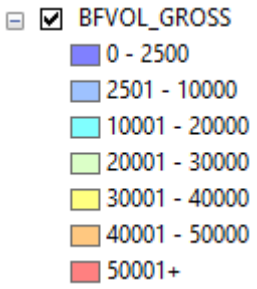
RS-FRIS Raster Variables

1. Total Cubic Volume
2. Heights of the Largest (DBH) Trees
3. Percent Tree Cover









These three raster variables have the least amount of variation when compared to traditional cruise data.

Low Estimate → High Estimate

Stand Treelist Imputation

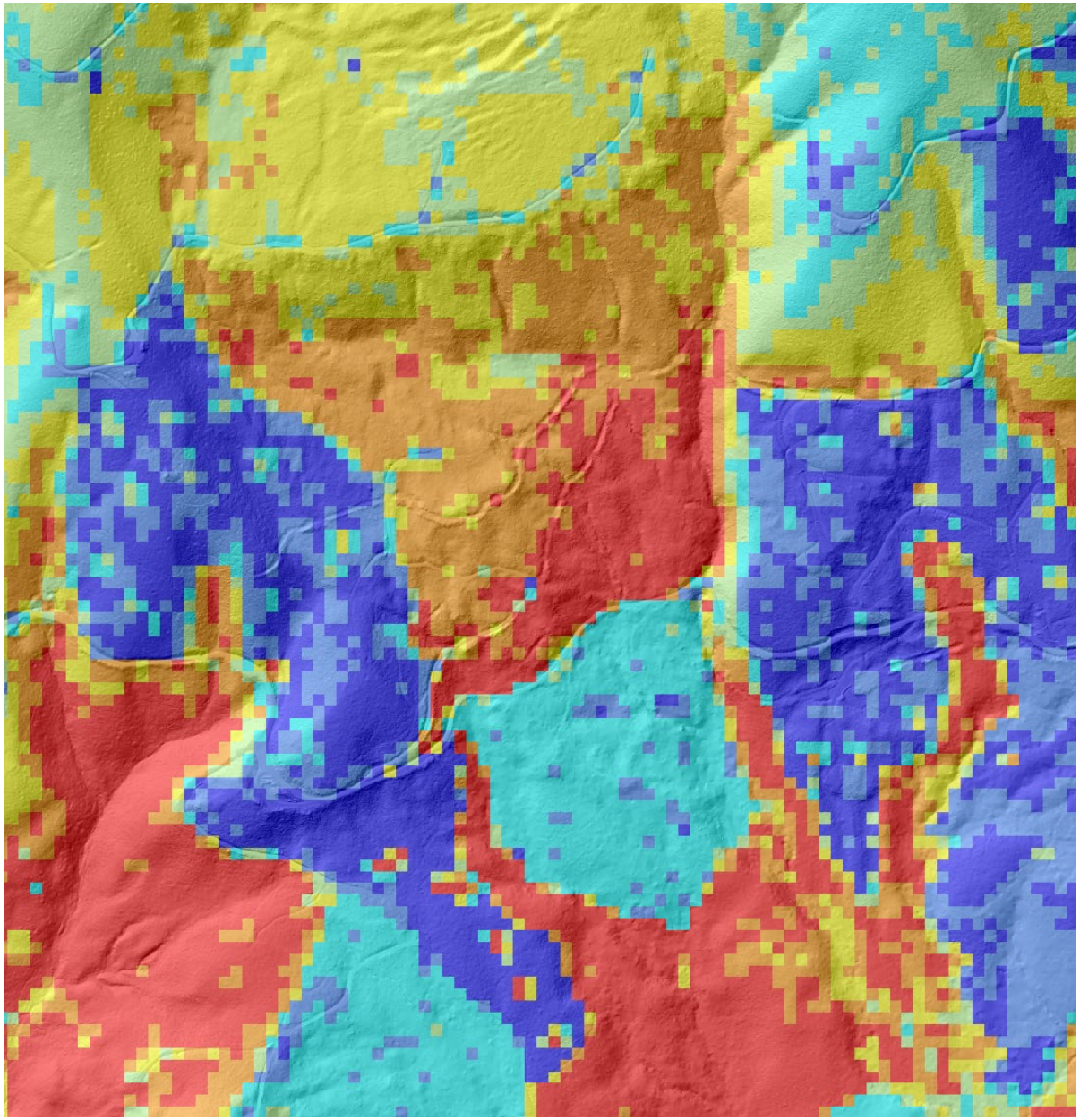
Why? FVS needs lists of trees to represent stands.

1) Gather Data
   a) Normalize RS-FRIS plot data for FVS input

2) Group (*raster*) stand-grids into bins, by volume

3) Match RS-FRIS plots to stand bins by ecotype

4) Impute all stand-grid volume-bins with the ecotype-matched, plot treelists



Plot <-> Stand-Grid (Bin)

Assigns the best matching RS-FRIS plot to a target stand-grid in terms of having the smallest difference between variables:

- Metrics:
  - Heights (*ht_t40*)
  - Cover
  - Volume (*cfvol_total*)
- Climate:
  - Moisture
  - Elevation
  - Temperature

Goal:

Find out what plot treelist is most appropriate for a grid cell.

```
if (eco == 1) {       # SAF-ES-LP Subalpine
  return (
      abs((x$vol1       - y$meanvol2)*2)
    + abs((x$ht1        - y$meanht2))
    + abs((x$cov1       - y$meancov2))
    + abs((x$elevation1 - y$elevation2)*2)
    + abs((x$aet1       - y$aet2))
    + abs((x$td1        - y$td2))
    + abs((x$mat1       - y$mat2)))
}
else if (eco == 2) { # CMMC
  return (
      abs((x$vol1       - y$meanvol2)*2)
    + abs((x$ht1        - y$meanht2))
    + abs((x$cov1       - y$meancov2))
    + abs((x$elevation1 - y$elevation2))
    + abs((x$aet1       - y$aet2))
    + abs((x$mat1       - y$mat2)))
}
else if (eco == 3) { # WMMC
  return (
      abs((x$vol1       - y$meanvol2)*2)
    + abs((x$ht1        - y$meanht2))
    + abs((x$cov1       - y$meancov2))
    + abs((x$aet1       - y$aet2))
    + abs((x$map1       - y$map2))
    + abs((x$shm1       - y$shm2)))
}
```

R6 Class: Imputer

```
initialize = function (debug = FALSE, explore = FALSE, run = FALSE) {
  self$debug = debug
  self$explore = explore
  print("Initializing a new instance of the Imputer class.")
  if (run) {
    t1 = Sys.time()
    self$.impute(debug, explore)
    t2 = Sys.time()
    elapsed = t2 - t1
    print(elapsed)
  }
},
```

```
# Examples of how to call the Imputer R6class written for imputi

#################################################################
# Example 1 -------------------- Run the parLapply() version --
#################################################################

i1 <- Imputer$new(run = TRUE, debug = FALSE)
                # 'run' argument tells it to use parLapply() with

#################################################################
# Example 2 ---------------------- Run the lapply() version ----
#################################################################

i2 <- Imputer$new(run = FALSE, debug = FALSE)
                # 'run' = FALSE allows us to subsequently set make
                # This is done in order to use lapply(), without c

# It is necessary to set 'make_cluster' = FALSE in order to use
i2$make_cluster = FALSE
# The .impute() function runs all of the processes which are par
i2$.impute(i2$debug, i2$explore)
```

```
.impute = function (debug, explore, name_ref=".impute()") {
  t1 = Sys.time()
  if (debug) {
    self$make_cluster = FALSE}
  self$cluster = .fx_set_cluster()
  ptm <- proc.time() # Start the clock!
  self$ptm = ptm
  print("Beginning imputation.")
  print(ptm)
  print("Proceeding to query all trees from the database.")
  # aggregates class methods into one function for general use
  self$trees =        .step_1_query_trees(is_oracle)
  print(proc.time() - ptm) # Stop the clock!
  print("Proceeding to read all stand grids from the R files.")
  self$stands =       .step_2_read_stands(n = 7, directory = dir$inputs[1], name_prefix =
  print(proc.time() - ptm) # Stop the clock!
  print("Proceeding to group all stand-grids in bins according to volume.")
  self$volumes =      .step_3_bin_stand_grids(stands, debug = FALSE)
  print(proc.time() - ptm) # Stop the clock!
  print("Proceeding to match RS-FRIS plots to stand-grid volume-bins by ecotype.")
  self$imputations =  .step_4_match_ecotype(trees, volumes, cluster, plot_file = "plots_er
  print(proc.time() - ptm) # Stop the clock!
  # data exploration
  print("Proceeding to read historic FRIS based numbers from the R file.")
  self$fris =         .fx_read_historic_acreage(debug = FALSE, directory = self$dir$fris[1
  print(proc.time() - ptm) # Stop the clock!
  print("Comparing current to historic data via FRIS.")
  if (explore) {
    self$map_eco_1 =  .fx_explore_acreages(1, fris, plots1, volumes, debug)
    self$map_eco_2 =  .fx_explore_acreages(2, fris, plots2, volumes, debug)
    self$map_eco_3 =  .fx_explore_acreages(3, fris, plots3, volumes, debug)
    self$map_eco_4 =  .fx_explore_acreages(4, fris, plots4, volumes, debug)
    self$map_eco_5 =  .fx_explore_acreages(5, fris, plots5, volumes, debug)
    self$map_eco_6 =  .fx_explore_acreages(6, fris, plots6, volumes, debug)
    self$map_eco_7 =  .fx_explore_acreages(7, fris, plots7, volumes, debug)
    print(proc.time() - ptm) # Stop the clock!
  }
  # imputation processing
  print("Proceeding to impute all stand-grids volume bins with ecotype matched plots.")
  self$impute_map =   .step_5_process_imputations(imputations, volumes, debug)
  print(proc.time() - ptm) # Stop the clock!
  print("Proceeding to map all stand grids for percent difference in volume.")
  self$plot_map =     .step_6_prepare_plot_mapping(impute_map, trees)
  print(proc.time() - ptm) # Stop the clock!
  print(sprintf("Imputation succeeded. Returning from %s function.", name_ref))
  t2 = Sys.time()
  elapsed = t2 - t1
  print(elapsed)
  return(plot_map)
},
```

```r
##############################################################################################
# Example 3 -------------------- Run functions individually ------------------------------ #
##############################################################################################

# - This is particularly useful if:
#   (A) You only need to debug something specific
#   (B) You require intermediary review of data

i3 <- Imputer$new(debug = FALSE)
  # Note that we set 'debug' = FALSE at the R6Class level above
  # Demonstrated further throughout how you can alternate between using the instantiated value or a new Boolean

# It is necessary to set 'make_cluster' and call .fx_set_cluster() when 'manually' running functions
i3$make_cluster = TRUE
i3$cluster = i3$.fx_set_cluster()

trees = i3$.step_1_query_trees(
  i3$is_oracle) # 'is_oracle' gets set to FALSE, by default, during instantiation [Imputer$new()]

stands = i3$.step_2_read_stands(
  n           = 7,
  directory   = i3$dir$inputs[1],
  name_prefix = "vegeco",
  name_suffix = "_dat_environ_standardized_v3rsfris_20210805.rds",
  debug       = FALSE)

volumes = i3$.step_3_bin_stand_grids(
  stands,
  debug = FALSE)

imputations = i3$.step_4_match_ecotype(
  trees, volumes, i3$cluster,
  plot_file = "plots_environ_standardized_domspp_20210805.rds",
  debug = i3$debug)

i3$fris = i3$.fx_read_historic_acreage(
  directory = i3$dir$fris[1],
  name_file = "original_fris_ba_speciesv2.rds",
  debug = FALSE)

impute_map = i3$.step_5_process_imputations(
  imputations = imputations,
  volumes = volumes,
  debug = i3$debug)

i3$plot_map = i3$.step_6_prepare_plot_mapping(
  impute_map,
  trees)
```

Preliminary Thoughts:

parLapply

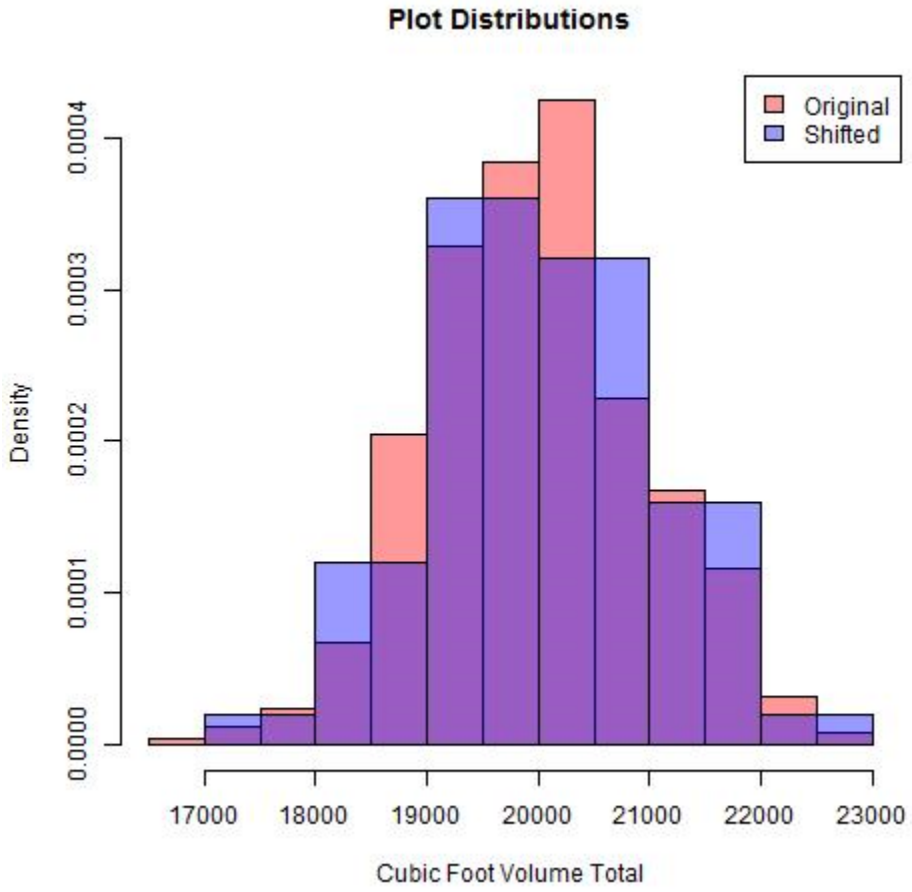
- Functional programming
- OOP R6 Class

```
.fx_run_parallel = function (ecotype, plots, volumes, cluster,
                             debug    = FALSE) {
  library(foreach)
  library(doParallel)
  env1 = environment()
  if (debug) {
    browser()}
  target = .fx_return_bins(ecotype, volumes, debug)
  if (self$make_cluster) {
    # proceed with cluster processing of data
    arguments = c("ecotype", "plots", "debug")
    clusterExport(cluster, arguments, envir = env1)
    registerDoParallel(cluster)
    y_neighbors = parLapply(cl = cluster,
                            X =  target,
                            fun = .fx_match_plots_2_stand_grid_bins,
                            ecotype = ecotype,
                            plots = plots,
                            debug = debug)
    y_df = as.data.frame(rbindlist(y_neighbors))
    return (y_df)
  } else {
```

Known Issue:

Plot to stand-grid(s) matching

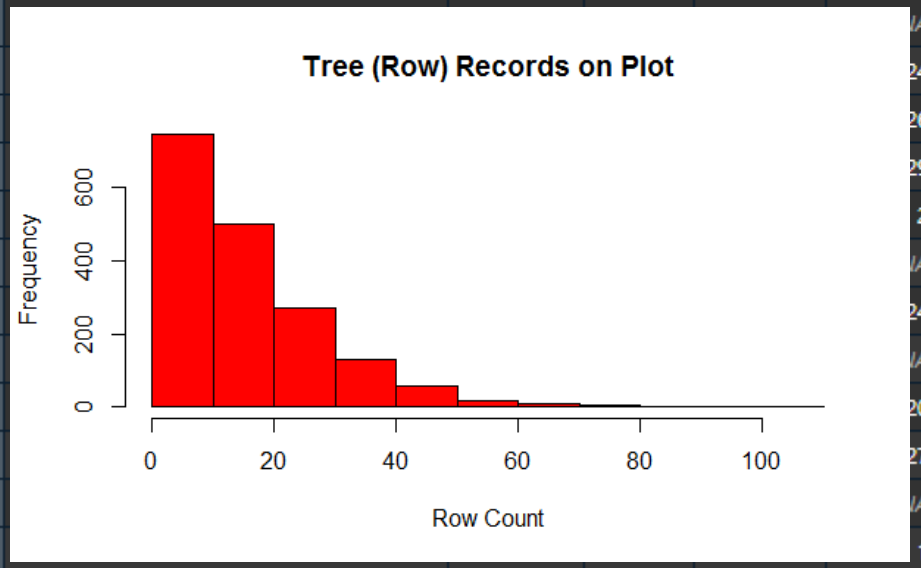
Distribution thinning
and/or shifting around the target metric



Summarization

```
trees %>% group_by(
        plot_id,
        species,
        status, dbh_m, ht_m
) %>% summarise(
        n = n(), # observations
        tpa = sum(tree_acre)
)
```

	plot_id	species	ts	dbh_m	ht_m	n	tpa
1	268803d5-aec4-4bf3-b84a-fcaacc2f9d3a	WA	1	3.0	18	10	400.70
2	6390feb5-29e1-40df-95a1-da416f78cf25	LP	1	2.0	15	8	320.56
3	c6074da0-5eb2-473d-a3cb-ee5bf6cb8cf0	GF	1	1.0	NA	8	2025.60
4	6390feb5-29e1-40df-95a1-da416f78cf25	LP	1	2.0	12	7	280.49
5	df3b926d-154b-444c-a799-7219477034f1	WO	1	4.0	30	6	240.42
6	43ed4603-3f29-4102-997a-816a56a24f09	PP	1	0.0	1	5	1266.05
7	56a546cd-2679-4d58-ba0b-10312df1e4f0	DF	1	0.0	2	5	2025.68
8						4	40.08
9					24	4	586.56
0					26	4	160.28
1					29	4	160.28
2					2	4	1012.84
3						4	40.08
4					24	4	160.28
5						4	1012.84
6					20	4	160.28
7					27	4	160.28
8						4	40.00
9					1	4	26587.05



Thank you for your time!

Jacob Beard
jacob.beard@dnr.wa.gov