Regression III

More Flexible Fitting Methods

Dave Armstrong

Goals for Today

1. Discuss methods for more flexible fitting.
    - Classification and Regression Trees (CART)
    - Random Forests Regression
    - Multivariate Adaptive Regression Splines (MARS)
    - Adaptive LASSO with Polynomial Expansion (Polywog)
2. Discuss MARS in an inferential context.

Classification and Regression Trees (CART)

CART works in a decision-tree framework.

- Considering all independent variables, which dichotomization on one of them explains the most variance.
- Conditional on the previous *split*, which next dichotomization explains the most variance.
- Loss function is the well-known residual sum of squares.
- Continue until some stopping rule is met.

Notes

Type notes here...

Notation

$$f(X_i) = T(X_i, \Theta) \equiv \sum_{b=1}^{B} c_b I(X_i \in R_b)$$

- $T()$ is a regression tree, with rules Θ regarding tree depth, stopping rules, etc...
- X_i is the data.
- c_b is the predicted value in each of the B regions.
- $I()$ is an indicator function.
- R_b defines the different regions in the space.

Notes

Type notes here...

Stopping Rules

- Candidate splits must increase R^2 by a pre-specified amount (the `cp` parameter, default=.01).
- Each candidate split must have at least `minsplit` observations in it (default=20).
- Each terminal node must have at least `minbucket` observations in it. Defaults to `round(minsplit/3)`.
- Tree depth - starting with the root node (0), what is the maximum depth of any node (defaults to 30).

Notes

Type notes here...

Example

```r
library(rpart)
library(dplyr)
library(car)
data(SLID)
SLID <- SLID %>%
  dplyr::select(wages, age, education) %>%
  na.omit
mod <- rpart(log(wages) ~ age + education, data=SLID)
mod
```

```
## n= 4014
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 4014 1018.09900 2.619255
##    2) age< 23.5 615   64.49511 2.064677 *
##    3) age>=23.5 3399  730.23310 2.719598
##      6) education< 15.65 2536  482.80130 2.641571
##       12) age< 31.5 554   85.64258 2.488905 *
##       13) age>=31.5 1982  380.63750 2.684244
##         26) education< 13.95 1560  288.89010 2.646745 *
##         27) education>=13.95 422   81.44458 2.822866 *
##      7) education>=15.65 863  186.62170 2.948886
##       14) age< 29.5 209   37.94680 2.617102 *
##       15) age>=29.5 654  118.31580 3.054915 *
```

Notes

Type notes here...

Decision Tree

```
plot(mod)
text(mod)
```

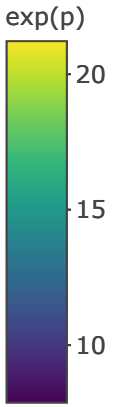
Notes

Type notes here...

Surface Plot

```
age.s <- seq(20, 95, length=25)
educ.s <- seq(9, 20, length=25)
cartpred <- function(x,y){
  predict(mod,
          newdata=data.frame(
            age = x,
            education=y))
}
p <- outer(age.s,
           educ.s,
           cartpred)
```

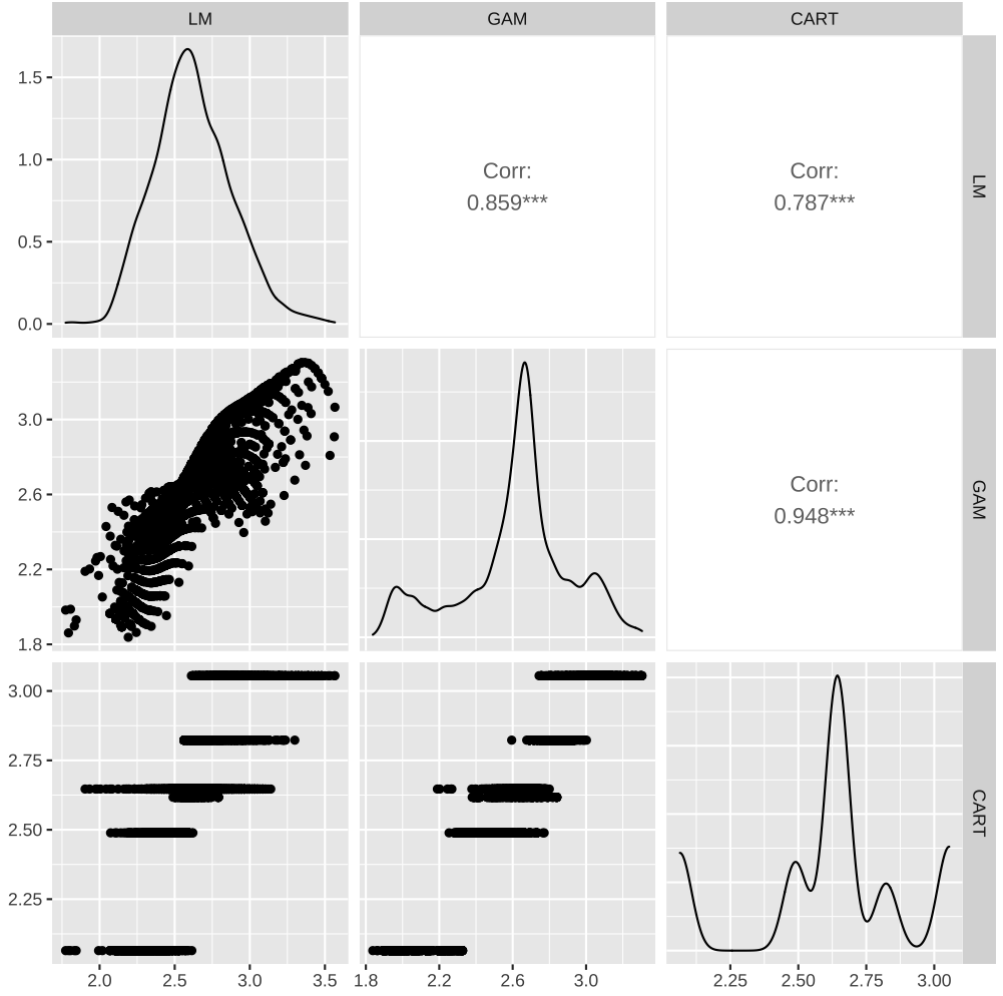
```
library(plotly)
plot_ly() %>%
  add_surface(x=~age.s, y=~educ.s, z=~exp(p)) %>
  layout(
    scene= list(
      xaxis=list(title="Age"),
      yaxis=list(title="Education"),
      zaxis=list(title="Predicted Wage")))
```



Notes

Type notes here...

Compare to Other Models



Notes

Type notes here...

Smoothness of CART Models

Problems with CART

- Not particularly smooth - step functions aren't great approximations for smooth curves (though they can do OK).
- No real means for inference here. Bootstrapping can be problematic because the function is "non-regular" (small data changes can result in wild changes in the model)
- In more complicated models, it's difficult to figure out what effects look like.

Notes

Type notes here...

Visualizing Partial Effects: Partial Dependence Plot

The PDP plots the change in the average predicted value for a subset of features S, averaged over the subset of features C, where C is the complement of S. Formally:

$$f_S = \mathbb{E}_{x_C}\left[f(\mathbf{x}_S, \mathbf{x}_C)\right] = \int f(\mathbf{x}_S, \mathbf{x}_C) dP(\mathbf{x}_C)$$

In words: we are predicting $f()$ with the variables in S averaged over all of the variables in C.

Notes

Type notes here...

ICE Plots

ICE disaggregates the PDP.

- The PDP is obtained by averaging over all of the ICE curves.
- Plots N different curves to enable evaluation of effect heterogeneity.
- Heterogeneity essentially means interactions with variables in C.

$$f_{S_i} = \mathbb{E}_{x_{C_i}} \left[f(\mathbf{x}_S, \mathbf{x}_{C_i}) \right]$$

Notes

Type notes here...

Ice Ice Baby

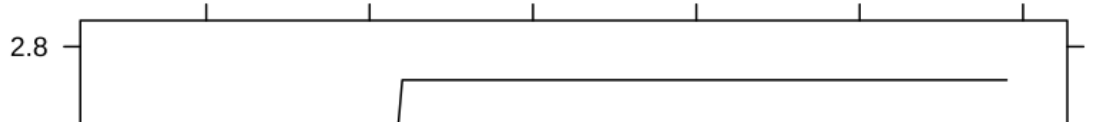
```
library(ICEbox)
library(RColorBrewer)
ice1 <- ice(mod, SLID, y=mod$y,
            predictor="age")
cice1 <- clusterICE(ice1, nClusters=3,
                    plot_legend=TRUE,
                    colorvec=brewer.pal(3,
                                        "Set1"))
```

Notes

Type notes here...

PDPs in R

```
library(pdp)
p1 <- partial(mod, pred.var="age", chull=TRUE)
plotPartial(p1)
```



Notes

Type notes here...

Ensemble Methods

Ensemble methods produce a bunch of (M) trees to better fit $f(X)$ and to prevent overfitting, with general form:

$$f(X_i) = \sum_{m=1}^{M} T_m(X_i, \Theta_m)$$

Tree Bagging (Bootstrap Aggregating)

- Draw lots of random samples from the data
- Fit a *deep* tree to each random sample.
- Average across the trees $\hat{f}_{\text{bag}}(X_i) = \frac{1}{M}\sum_{m=1}^{M} T_m(X_i, \Theta_m)$

Depends on the often dubious assumption of independence across trees to reduce bias and variance.

Notes

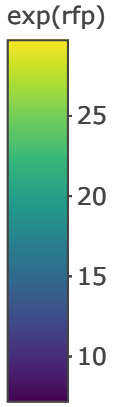
Type notes here...

Random Forests

A tree-bagging algorithm meant to increase independence across trees.

- In each random sample, only a small random subset (a) of the total j covariates is used in the splitting algorithm.
- Reduces bias and variance in the aggregate when a is small.

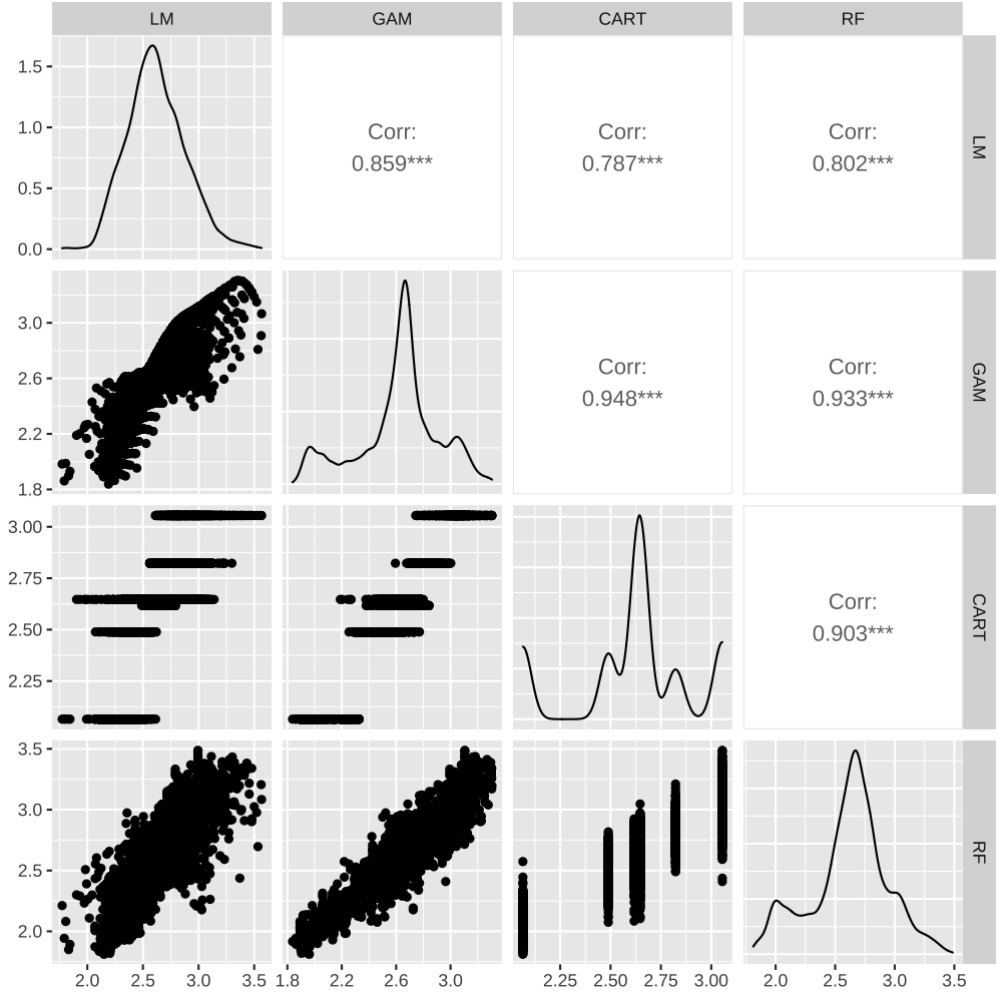
```
library(randomForest)
rfmod <- randomForest(log(wages) ~ age+education,
                      data=SLID,
                      type="regression")
```



Notes

Type notes here...

Compare to Other Models



Notes

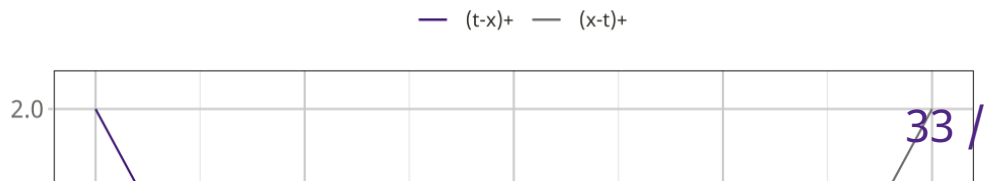
Type notes here...

Multivariate Adaptive Regression Splines (MARS)

The main component of MARS is a pair of piecewise linear (hinge) splines.

$$(x-t)_+ = \begin{cases} x-t & \text{if } x > t \\ 0 & \text{otherwise.} \end{cases}$$

$$(t-x)_+ = \begin{cases} t-x & \text{if } x < t \\ 0 & \text{otherwise.} \end{cases}$$



Notes

Type notes here...

MARS Notation

MARS takes the form:

$$f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(x)$$

where h_m is the pair of hinge functions.

Computationally:

- Forward pass - add pairs of hinge functions by reduction in SSRes until all pairs are in.
- Backward pass - take individual functions out by min increase in SSRes until GCV criterion is satisfied.

Notes

Type notes here...

Interactions

- The `degree` parameter in the R algorithm controls the degree of interaction you want to allow.
- This can make the model really complicated because it's expanding all possible interactions among hinge functions and then pulling them out on the backward pass step.
- This model is more easily constrained (particular w.r.t additivity) than the other models we talked about before.
- You can also identify variables that will enter the model linearly *if they enter the model at all* .

Notes

Type notes here...

MARS Wages

```r
library(earth)
emod <- earth(log(wages) ~ age +
                education, data=SLID,
              degree=3)
```

```r
summary(emod)
```

```
## Call: earth(formula=log(wages)~age+education, data=SLID, degree=3)
##
##                                  coefficients
## (Intercept)                        2.67861807
## h(32-age)                         -0.04945215
## h(age-32)                         -0.03079281
## h(12.6-education)                 -0.16237528
## h(education-12.6)                  0.15308353
## h(age-32) * h(education-17.1)     -0.01150617
## h(age-32) * h(17.1-education)      0.00666845
## h(age-34) * h(education-12.6)      0.00552501
## h(51-age) * h(education-12.6)     -0.00489692
## h(58-age) * h(12.6-education)      0.00426955
## h(age-58) * h(12.6-education)     -0.01536656
##
## Selected 11 of 12 terms, and 2 of 2 predictors
## Termination condition: RSq changed by less than 0.001 at 12 terms
## Importance: age, education
## Number of terms at each degree of interaction: 1 4 6
## GCV 0.166029    RSS 657.835    GRSq 0.3457333    RSq 0.3538597
```

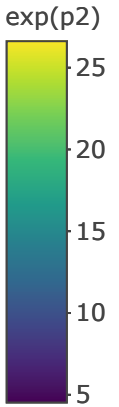
Notes

Type notes here...

Surface Plot

```
marspred <- function(x,y){
  predict(emod,
          newdata=data.frame(age = x,
                             education=y))
}
p2 <- outer(age.s,
            educ.s,
            marspred)
```

```
plot_ly() %>%
  add_surface(x=~age.s, y=~educ.s, z=~exp(p2)) %
  layout(
    scene= list(
      xaxis=list(title="Age"),
      yaxis=list(title="Education"),
      zaxis=list(title="Predicted Wage")
    ))
```



Notes

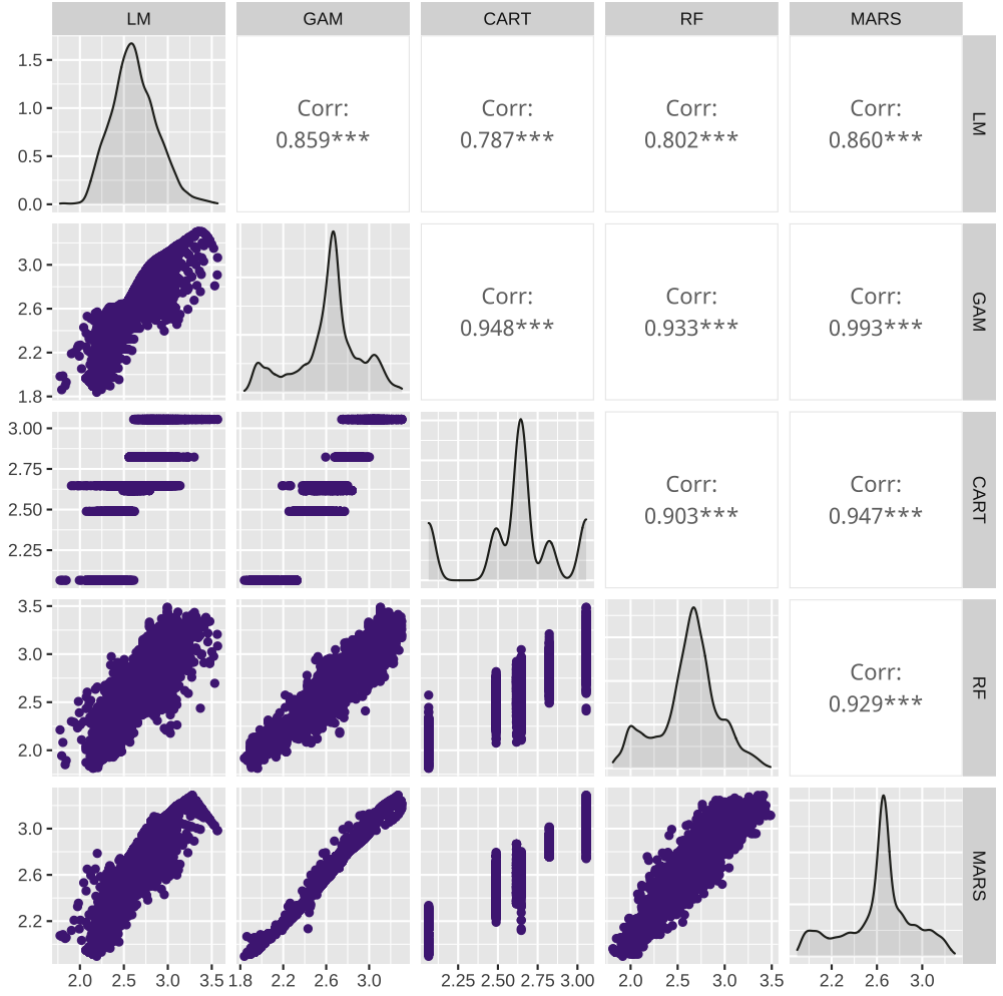
Type notes here...

Plots

Notes

Type notes here...

Compare to Other Models



Notes

Type notes here...

Variance Models

- You can't get confidence intervals from these models because they don't take into account the selection mechanism.
- MARS picks values essentially because they are good predictors, so the items in the model will necessarily have small p-values.
- You can get prediction intervals for the - essentially the variability in future observations predicted by the model.
- The `varmod.method` allows you to model the residual variance by modeling the absolute value of the residuals as a function of the fitted values.
- Prediction variance is:

$$\varepsilon^2_{i,future} = \frac{(y_i - \hat{y}_i)^2}{(1 - h_{ii})} + \text{modvar}_i$$

Notes

Type notes here...

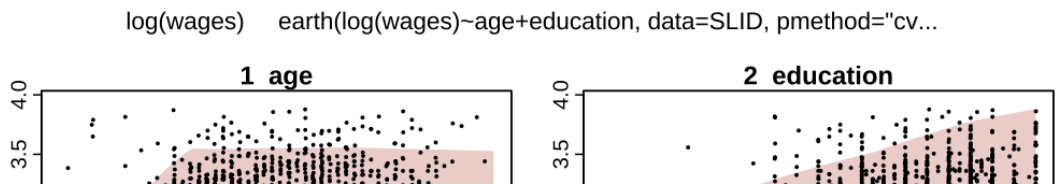
Prediction Variances in earth

```r
library(mgcv)
e2 <- earth(log(wages) ~ age + education,
  data=SLID,
  nfold=10, ncross=10, pmethod="cv",
  degree=2, varmod.meth="gam")
plotmo(e2, pt.col=1, level=.95)
```

Notes

Type notes here...

Plots


log(wages)
earth(log(wages)~age+education, data=SLID, pmethod="cv...
1 age
2 education
4.0
3.5
4.0
3.5

Notes

Type notes here...

Polywog

Polywog is a method developed by Kenkel and Signorino which puts two pieces we've already considered together:

- Polynomial expansion: If the degree = 3 and we have variablex $\{x_1, x_2\}$ in our model, then the following terms would be included in the expansion: $x_1, x_2, x_1^2, x_2^2, x_1^3, x_2^3, x_1x_2, x_1^2x_2, x_2^2x_1$.

- Adaptive Lasso: We use the adaptive LASSO to figure out which of the polynomial expansion terms to keep in the model.

Notes

Type notes here...

Polywog Example

```r
library(polywog)
p1 <- polywog(log(wages) ~ age +
      education, data=SLID, degree=4)
```

```
##
## Call:
## polywog(formula = log(wages) ~ age + education, data = SLID,
##     degree = 4)
##
## Coefficients:
##                      Estimate Std. Error
## (Intercept)         1.438e+00         NA
## age                 1.830e-02         NA
## education           0.000e+00         NA
## age^2               0.000e+00         NA
## age.education       4.063e-03         NA
## education^2        -4.891e-03         NA
## age^3               0.000e+00         NA
## age^2.education    -7.690e-05         NA
## age.education^2     1.129e-04         NA
## education^3         5.197e-05         NA
## age^4               1.172e-08         NA
## age^3.education     0.000e+00         NA
## age^2.education^2   0.000e+00         NA
## age.education^3     0.000e+00         NA
## education^4         0.000e+00         NA
##
## Regularization method: Adaptive LASSO
## Adaptive weights: inverse linear model coefficients
## Number of observations: 4014
## Polynomial expansion degree: 4
## Model family: gaussian
## Bootstrap iterations: 0
```

Notes

Type notes here...

Plots

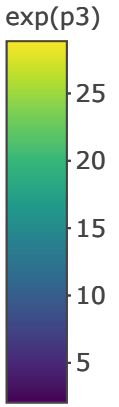
Notes

Type notes here...

Surface Plot

```
pwogpred <- function(x,y){
  predict(p1,
          newdata=data.frame(age = x,
                             education=y))
}
p3 <- outer(age.s,
            educ.s,
            pwogpred)
```

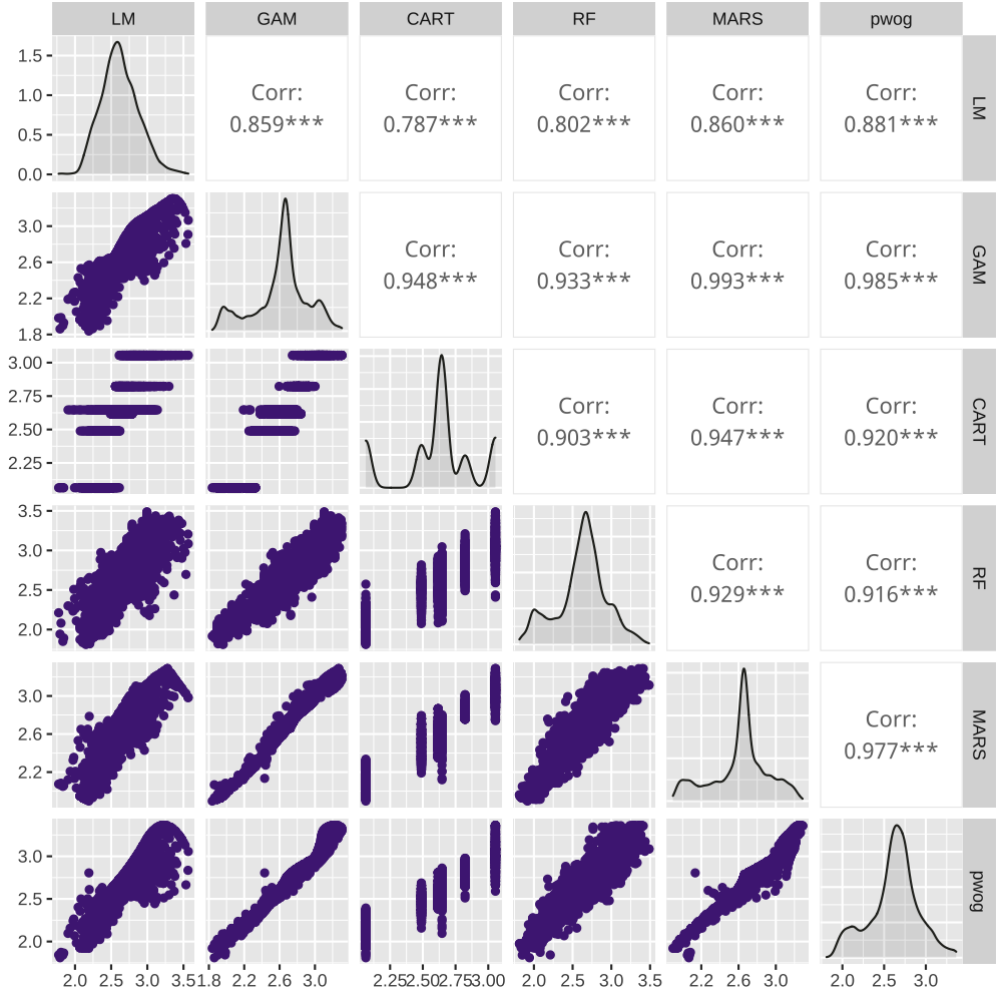
```
plot_ly() %>%
  add_surface(x=~age.s, y=~educ.s, z=~exp(p3)) %
  layout(
    scene= list(
      xaxis=list(title="Age"),
      yaxis=list(title="Education"),
      zaxis=list(title="Predicted Wage")
    ))
```



Notes

Type notes here...

Compare to Other Models



Notes

Type notes here...

Barry and Kleinberg Data

```
load(file("https://quantoid.net/files/reg3/bk.rda"))
bk <-as.data.frame(bk)
model2 <- gamlss(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
```

```
## GAMLSS-RS iteration 1: Global Deviance = 16189.29
## GAMLSS-RS iteration 2: Global Deviance = 16189.29
```

```
model3 <- gamlss(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
```

```
## GAMLSS-RS iteration 1: Global Deviance = 16182.26
## GAMLSS-RS iteration 2: Global Deviance = 16182.26
```

Notes

Type notes here...

Cart Models

```
bk.cart1 <- rpart(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
bk.cart2 <- rpart(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
```

Notes

Type notes here...

CART Model Results

```r
bk.cart1
```

```
## n= 2863
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 2863 61629.460 2.976535
##    2) L_usstock2000_adj< 8.908139 2184 36891.230 1.973858
##      4) L_usstock2000_adj< 6.56151 1169 12158.840 1.038018 *
##      5) L_usstock2000_adj>=6.56151 1015 22529.440 3.051688
##       10) L_growth< 2.675 330  8621.694 1.847877 *
##       11) L_growth>=2.675 685 13199.140 3.631627 *
##    3) L_usstock2000_adj>=8.908139 679 15480.040 6.201639
##      6) L_usstock2000_adj< 10.03128 327  8397.456 4.996460 *
##      7) L_usstock2000_adj>=10.03128 352  6166.405 7.321224 *
```

```r
bk.cart2
```

```
## n= 2863
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 2863 61629.460 2.976535
##    2) L_usstock2000_adj< 8.908139 2184 36891.230 1.973858
##      4) L_usstock2000_adj< 6.56151 1169 12158.840 1.038018 *
##      5) L_usstock2000_adj>=6.56151 1015 22529.440 3.051688
##       10) L_growth< 2.675 330  8621.694 1.847877 *
##       11) L_growth>=2.675 685 13199.140 3.631627 *
##    3) L_usstock2000_adj>=8.908139 679 15480.040 6.201639
##      6) L_usstock2000_adj< 10.03128 327  8397.456 4.996460 *
##      7) L_usstock2000_adj>=10.03128 352  6166.405 7.321224 *
```

Notes

Type notes here...

Random Forests

```
bk.cart1 <- rpart(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
bk.cart2 <- rpart(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000, data = bk)
```

Notes

Type notes here...

Random Forests

```
bk.rf1 <- randomForest(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk, mtry=3)
bk.rf2 <- randomForest(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk, mtry=3)
```

Notes

Type notes here...

Variable Importance

```
imp1 <- importance(bk.rf1)
imp1/max(imp1)
```

```
##                    IncNodePurity
## L_ab_sum              0.13914331
## L_hse_usnum           0.27336546
## L_hse_sanction        0.02201843
## L_growth              0.43484693
## L_lnpercap            0.57810416
## L_lnpop               0.47848065
## L_polity2             0.23685643
## L_durable             0.39365502
## L_civintensity        0.06230792
## L_spending            0.37971906
## L_s_us                0.37797556
## L_lnustrade           0.70098276
## L_us_distance         0.24532186
## L_usstock2000_adj     1.00000000
## allfdi2000            0.33512307
```

```
imp2 <- importance(bk.rf2)
imp2/max(imp2)
```

```
##                    IncNodePurity
## L_sancmeanshare       0.55040707
## L_tradeshare          0.39862913
## L_hse_sanction        0.01993749
## L_growth              0.42616856
## L_lnpercap            0.56690562
## L_lnpop               0.44173764
## L_polity2             0.21967758
## L_durable             0.38148594
## L_civintensity        0.06104718
## L_spending            0.38414386
## L_s_us                0.39166370
## L_lnustrade           0.69009928
## L_us_distance         0.22720748
## L_usstock2000_adj     1.00000000
## allfdi2000            0.35339199
```

Notes

Type notes here...

ICE Plot

```
library(RColorBrewer)
bk.i <- ice(bk.rf1, bk,
            predictor = "L_hse_usnum",
            frac_to_build = 1)
crv <- bk.i$ice_curves
crv <- t(apply(crv, 1, function(x)x-mean(x)))
sapply(2:10, function(i){
  k <- kmeans(crv, centers=i)
  k$betweenss/(k$tot.withinss+k$betweenss)
  })
bk.c <- clusterICE(bk.i, nClusters = 6,
        colorvec=brewer.pal(6, "Set1"),
        plot_legend = TRUE)
```

Prop.

Notes

Type notes here...

ICE Plot

```
bk.i2 <- ice(bk.rf2, bk,
            predictor = "L_sancmeanshare",
            frac_to_build = 1,
            num_grid_pts = 25)
crv <- bk.i2$ice_curves
crv <- t(apply(crv, 1, function(x)x-mean(x)))
sapply(2:10, function(i){
  k <- kmeans(crv, centers=i)
  k$betweenss/(k$tot.withinss+k$betweenss)
  })
bk.c <- clusterICE(bk.i2, nClusters = 5,
        colorvec=brewer.pal(5, "Set1"),
        plot_legend = TRUE)
```

Notes

Type notes here...

Random Forests

```
bk.e1 <- earth(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk, degree=3, pmethod="backward")
bk.e2 <- earth(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk, degree=3, pmethod="backward")
```

Notes

Type notes here...

Results 1

```
summary(bk.e1)
```

```
## Call: earth(formula=usfdi2000_adj~L_ab_sum+L_hse_usnum+L_hse_sanction+...),
##             data=bk, pmethod="backward", degree=3)
##
##                                                                  coefficients
## (Intercept)                                                        0.63392319
## h(17-L_hse_usnum)                                                  0.24534151
## h(L_hse_usnum-17)                                                  0.14932381
## h(5.84698-L_usstock2000_adj)                                      -0.11501938
## h(L_usstock2000_adj-5.84698)                                       1.39884572
## h(17-L_hse_usnum) * h(L_s_us-0.452007)                            -2.07349368
## h(17-L_hse_usnum) * h(L_lnpercap-8.72583) * h(L_s_us-0.452007)     1.32325615
## h(17-L_hse_usnum) * h(8.72583-L_lnpercap) * h(L_s_us-0.452007)     0.60472031
## h(17-L_hse_usnum) * h(L_lnpop-16.1764) * h(L_s_us-0.452007)        0.71042312
## h(17-L_hse_usnum) * h(16.1764-L_lnpop) * h(L_s_us-0.452007)        0.63388703
## h(17-L_hse_usnum) * h(0.452007-L_s_us) * h(6.39192-L_lnustrade)   -0.26949929
## h(L_hse_usnum-17) * h(0.593291-L_s_us) * h(L_lnustrade-9.24532)   -0.51248303
## h(L_hse_usnum-17) * h(0.593291-L_s_us) * h(9.24532-L_lnustrade)   -0.08992691
## h(13-L_growth) * h(L_lnpercap-9.49552) * h(L_usstock2000_adj-5.84698)  -0.05865711
## h(13-L_growth) * h(9.49552-L_lnpercap) * h(L_usstock2000_adj-5.84698)  -0.04877511
## h(13-L_growth) * h(L_lnpop-18.6438) * h(L_usstock2000_adj-5.84698)      0.10606737
## h(13-L_growth) * h(18.6438-L_lnpop) * h(L_usstock2000_adj-5.84698)     -0.00737439
## h(L_growth- -7.84) * h(-9-L_polity2) * h(L_usstock2000_adj-5.84698)    -0.10431613
## h(13-L_growth) * h(21.5-L_spending) * h(L_usstock2000_adj-5.84698)      0.00391730
## h(13-L_growth) * h(L_usstock2000_adj-5.84698) * h(59834.9-allfdi2000)  -0.00000080
##
## Selected 20 of 31 terms, and 10 of 15 predictors
## Termination condition: Reached nk 31
```

Notes

Type notes here...

Results 2

```r
summary(bk.e2)
```

```
## Call: earth(formula=usfdi2000_adj~L_sancmeanshare+L_tradeshare+L_hse_s...),
##             data=bk, pmethod="backward", degree=3)
##
##                                                                    coefficients
## (Intercept)                                                           1.2590580
## h(L_s_us-0.759834)                                                   23.4287388
## h(5.84698-L_usstock2000_adj)                                         -0.1385368
## h(L_usstock2000_adj-5.84698)                                          1.4170450
## h(13-L_growth) * h(L_usstock2000_adj-5.84698)                         0.0553836
## h(L_lnpercap-10.0476) * h(0.759834-L_s_us)                            8.1246653
## h(8.17926-L_lnustrade) * h(L_usstock2000_adj-5.84698)                 0.5108149
## h(L_lnustrade-8.17926) * h(L_usstock2000_adj-5.84698)                -0.2692848
## h(13-L_growth) * h(9.4572-L_lnpercap) * h(L_usstock2000_adj-5.84698) -0.0572491
## h(13-L_growth) * h(L_lnpop-18.6438) * h(L_usstock2000_adj-5.84698)    0.1023436
## h(13-L_growth) * h(18.6438-L_lnpop) * h(L_usstock2000_adj-5.84698)   -0.0263415
## h(13-L_growth) * h(16.9-L_spending) * h(L_usstock2000_adj-5.84698)    0.0032823
## h(1.39-L_growth) * h(L_lnustrade-8.17926) * h(L_usstock2000_adj-5.84698)  0.0617883
## h(L_growth-1.39) * h(L_lnustrade-8.17926) * h(L_usstock2000_adj-5.84698)  0.0421413
## h(13-L_growth) * h(L_usstock2000_adj-5.84698) * h(59834.9-allfdi2000)  -0.0000012
## h(L_lnpercap-10.3123) * h(L_lnustrade-8.17926) * h(L_usstock2000_adj-5.84698)  -1.8565817
## h(-9-L_polity2) * h(8.17926-L_lnustrade) * h(L_usstock2000_adj-5.84698)  -0.9600373
## h(0.333333-L_s_us) * h(8.17926-L_lnustrade) * h(L_usstock2000_adj-5.84698)  -2.9656317
## h(L_s_us-0.333333) * h(8.17926-L_lnustrade) * h(L_usstock2000_adj-5.84698)  -1.6569208
## h(8.17926-L_lnustrade) * h(5425-L_us_distance) * h(L_usstock2000_adj-5.84698)  0.0001429
##
## Selected 20 of 31 terms, and 10 of 15 predictors
## Termination condition: Reached nk 31
```

Notes

Type notes here...

ICE Plot

```
bk.i <- ice(bk.e1, bk,
            predictor = "L_hse_usnum",
            frac_to_build = 1)
crv <- bk.i$ice_curves
crv <- t(apply(crv, 1, function(x)x-mean(x)))
sapply(2:10, function(i){
  k <- kmeans(crv, centers=i)
  k$betweenss/(k$tot.withinss+k$betweenss)
  })
bk.c <- clusterICE(bk.i, nClusters = 4,
        colorvec=brewer.pal(4, "Set1"),
        plot_legend = TRUE)
```

Notes

Type notes here...

Investigating clusters

```r
library(nnet)
bk$cluster <- bk.c$cluster
clust.mod <- multinom(cluster ~ L_ab_sum +
    L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop +
    L_polity2 + L_durable + L_civintensity +
    L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj +
    allfdi2000, data=bk)
```

```
## # weights:  68 (48 variable)
## initial  value 3968.960756
## iter  10 value 3516.858960
## iter  20 value 3460.161139
## iter  30 value 3448.755375
## iter  40 value 3361.707671
## iter  50 value 3283.336646
## final  value 3281.728662
## converged
```

```r
DAMisc::mnlChange(clust.mod, bk)
```

```
##                    [,1]    [,2]    [,3]    [,4]
## L_ab_sum          -0.007  -0.105* -0.058*  0.170*
## L_hse_usnum       -0.049* -0.047*  0.043*  0.054*
## L_hse_sanction     0.013*  0.030* -0.039* -0.003*
## L_growth           0.119* -0.149* -0.011*  0.041*
## L_lnpercap        -0.288*  0.141*  0.245* -0.098*
## L_lnpop           -0.298*  0.038*  0.276* -0.017*
## L_polity2          0.099* -0.016* -0.096*  0.012*
## L_durable          0.250* -0.115*  0.006  -0.141*
## L_civintensity     0.114* -0.086* -0.022* -0.007*
## L_spending        -0.221* -0.018   0.181*  0.058*
## L_s_us            -0.152* -0.002*  0.071*  0.084*
## L_lnustrade        0.294*  0.205* -0.146* -0.354*
## L_us_distance      0.228* -0.177* -0.006  -0.045
## L_usstock2000_adj  0.191* -0.091* -0.395*  0.296*
## allfdi2000         0.060   0.042  -0.016  -0.086*
```

Notes

Type notes here...

Venus

Venus is a project that I am working on with Duncan Murdoch.

- Some variables are subject to a MARS fit.
  - Good way to control for variables.
- Other variables are included in their assumed parametric form.

Notes

Type notes here...

Details

$$\mathbf{y} = \alpha + \mathbf{X}\beta + \mathbf{Z}\mathbf{\Gamma} + \varepsilon$$

We create $e^{(y)}$ with MARS:

$$\mathbf{y} = \lambda H(\mathbf{Z}) + e^{(y)}$$

and $\mathbf{e}^{(x)}$ with

$$\mathbf{X} = \theta H(\mathbf{Z}) + \mathbf{e}^{(X)}$$

Then we regress $e^{(y)}$ on $\mathbf{e}^{(X)}$ to obtain estimates of β controlling for $\mathbf{Z}$ in a flexible way.

Notes

Type notes here...

In R

```
remotes::install_github("dmurdoch/venus")
```

```
library(venus)
bk.v1 <- venus(usfdi2000_adj ~ L_ab_sum + L_hse_usnum,
               usfdi2000_adj~ L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk)
bk.v2 <- venus(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare,
               usfdi2000_adj ~ L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk)
```

Notes

Type notes here...

Summary

```r
summary(bk.v1$mainFit)
```

```
##
## Call:
## lm(formula = yResids ~ mainModelmatrix)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.664  -1.225   1.323   2.558   9.208
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -1.170e-16  7.493e-02   0.000 1.000000
## mainModelmatrixL_ab_sum     2.330e-02  3.994e-02   0.583 0.559609
## mainModelmatrixL_hse_usnum -5.278e-02  1.433e-02  -3.684 0.000234 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.009 on 2860 degrees of freedom
## Multiple R-squared:  0.004733,    Adjusted R-squared:  0.004037
## F-statistic: 6.801 on 2 and 2860 DF,  p-value: 0.001131
```

```r
summary(bk.v2$mainFit)
```

```
##
## Call:
## lm(formula = yResids ~ mainModelmatrix)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.558  -1.157   1.368   2.544   9.225
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -9.826e-17  7.508e-02   0.000    1.000
## mainModelmatrixL_sancmeanshare  5.799e-02  5.111e-02   1.135    0.257
## mainModelmatrixL_tradeshare    -9.313e-03  8.569e-03  -1.087    0.277
##
## Residual standard error: 4.017 on 2860 degrees of freedom
## Multiple R-squared:  0.0008612,    Adjusted R-squared:  0.0001625
## F-statistic: 1.233 on 2 and 2860 DF,  p-value: 0.2917
```

Notes

Type notes here...

In GAMLSS

```
remotes::install_url("https://quantoid.net/files/gamlss.add2_1.0-0.tar.gz")
```

```
library(gamlss)
library(gamlss.add)
bk.g1 <- gamlss(usfdi2000_adj ~ L_ab_sum + L_hse_usnum +
            tr(~L_hse_sanction + L_growth + L_lnpercap +
                L_lnpop + L_polity2 + L_durable + L_civintensity +
                L_spending + L_s_us + L_lnustrade +
                L_us_distance + L_usstock2000_adj + allfdi2000),
            data = bk, trace=FALSE, control=gamlss.control(n.cyc=100))
```

```
## GAMLSS-RS iteration 1: Global Deviance = 16188.54
## GAMLSS-RS iteration 2: Global Deviance = 16188.54
```

```
bk.g2 <- gamlss(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare +
            tr(~L_hse_sanction + L_growth + L_lnpercap +
                L_lnpop + L_polity2 + L_durable + L_civintensity +
                L_spending + L_s_us + L_lnustrade +
                L_us_distance + L_usstock2000_adj + allfdi2000),
            data = bk, trace=FALSE)
```

Notes

Type notes here...

Summary

```
summary(bk.g1)
```

```
## *******************************************************************
## Family:  c("NO", "Normal")
##
## Call:  gamlss(formula = usfdi2000_adj ~ L_ab_sum + L_hse_usnum +
##     tr(~L_hse_sanction + L_growth + L_lnpercap + L_lnpop +
##         L_polity2 + L_durable + L_civintensity + L_spending +
##         L_s_us + L_lnustrade + L_us_distance + L_usstock2000_adj +
##         allfdi2000), data = bk, control = gamlss.control(n.cyc = 100),
##     trace = FALSE)
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  identity
## Mu Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.25029    0.19402   16.75   <2e-16 ***
## L_ab_sum     0.05211    0.04071    1.28   0.2006
## L_hse_usnum -0.01590    0.00935   -1.70   0.0892 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
summary(bk.g2)
```

```
## *******************************************************************
## Family:  c("NO", "Normal")
##
## Call:  gamlss(formula = usfdi2000_adj ~ L_sancmeanshare +
##     L_tradeshare + tr(~L_hse_sanction + L_growth +
##     L_lnpercap + L_lnpop + L_polity2 + L_durable +
##     L_civintensity + L_spending + L_s_us + L_lnustrade +
##     L_us_distance + L_usstock2000_adj + allfdi2000),
##     data = bk, trace = FALSE)
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  identity
## Mu Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.786760   0.103522  26.920  < 2e-16 ***
## L_sancmeanshare  0.364642   0.052158   6.991 3.38e-12 ***
## L_tradeshare    -0.013476   0.008437  -1.597     0.11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

Notes

Type notes here...

MARS

```r
library(gamlss.add2)
bk.g3 <- gamlss(usfdi2000_adj ~ L_ab_sum + L_hse_usnum +
            ma(~L_hse_sanction + L_growth + L_lnpercap +
                L_lnpop + L_polity2 + L_durable + L_civintensity +
                L_spending + L_s_us + L_lnustrade +
                L_us_distance + L_usstock2000_adj + allfdi2000),
            data = bk, trace=FALSE, control=gamlss.control(n.cyc=100))
```

```
## GAMLSS-RS iteration 1: Global Deviance = 16051.26
## GAMLSS-RS iteration 2: Global Deviance = 16057.73
## GAMLSS-RS iteration 3: Global Deviance = 16057.73
```

```r
bk.g4 <- gamlss(usfdi2000_adj ~ L_sancmeanshare + L_tradeshare +
            ma(~L_hse_sanction + L_growth + L_lnpercap +
                L_lnpop + L_polity2 + L_durable + L_civintensity +
                L_spending + L_s_us + L_lnustrade +
                L_us_distance + L_usstock2000_adj + allfdi2000),
            data = bk, trace=FALSE)
```

Notes

Type notes here...

Summary

```r
summary(bk.g3)
```

```
## *******************************************************************
## Family:  c("NO", "Normal")
##
## Call:  gamlss(formula = usfdi2000_adj ~ L_ab_sum + L_hse_usnum +
##     ma(~L_hse_sanction + L_growth + L_lnpercap + L_lnpop +
##         L_polity2 + L_durable + L_civintensity + L_spending +
##         L_s_us + L_lnustrade + L_us_distance + L_usstock2000_adj +
##         allfdi2000), data = bk, control = gamlss.control(n.cyc = 100),
##     trace = FALSE)
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  identity
## Mu Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.564214   0.189639  24.068   <2e-16 ***
## L_ab_sum     0.010584   0.039788   0.266     0.79
## L_hse_usnum -0.082545   0.009139  -9.032   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```r
summary(bk.g4)
```

```
## *******************************************************************
## Family:  c("NO", "Normal")
##
## Call:  gamlss(formula = usfdi2000_adj ~ L_sancmeanshare +
##     L_tradeshare + ma(~L_hse_sanction + L_growth +
##     L_lnpercap + L_lnpop + L_polity2 + L_durable +
##     L_civintensity + L_spending + L_s_us + L_lnustrade +
##     L_us_distance + L_usstock2000_adj + allfdi2000),
##     data = bk, trace = FALSE)
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  identity
## Mu Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.955280   0.100640  29.365  < 2e-16 ***
## L_sancmeanshare  0.149396   0.050706   2.946  0.00324 **
## L_tradeshare    -0.013936   0.008202  -1.699  0.08944 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

Notes

Type notes here...

Tests

```
VC.test(model2, bk.g1)
```

```
##  Vuong's test: -0.733 it is not possible to discriminate between models: model2 and bk.g1
## Clarke's test: 1381 p-value= 0.0616 it is not possible to discriminate between models: model2 and bk.g1
```

```
VC.test(bk.g1, bk.g3)
```

```
##  Vuong's test: -2.898 model bk.g3 is preferred over bk.g1
## Clarke's test: 1232 p-value= 0 bk.g3 is preferred over bk.g1
```

```
VC.test(model3, bk.g2)
```

```
##  Vuong's test: 0.44 it is not possible to discriminate between models: model3 and bk.g2
## Clarke's test: 1474 p-value= 0.1164 it is not possible to discriminate between models: model3 and bk.g2
```

```
VC.test(bk.g2, bk.g4)
```

```
##  Vuong's test: -3.37 model bk.g4 is preferred over bk.g2
## Clarke's test: 1214 p-value= 0 bk.g4 is preferred over bk.g2
```

Notes

Type notes here...

Inference

Venus has good inferential properties.

- Bias, MSE and CI coverage errors go to 0 as N increases.
- Dominates a naive linear model in all but the perfect additive linear case.

Other models:

- The venus result suggests that the GAMLSS model should have decent inferential properties on the parametric terms.
- Trees, MARS and Polywog would need data splitting or something similar to make appropriate inferences.
  - Model building could be done on a training sample and then the appropriate model could be estimated on the other half of the data.

Notes

Type notes here...

Mars Example with Inference

```
set.seed(734)
train.samp <- sample(1:nrow(bk),
                     floor(nrow(bk)*.6),
                     replace=FALSE)
test.samp <- setdiff(1:nrow(bk), train.samp)
bk.e1 <- earth(usfdi2000_adj ~ L_ab_sum + L_hse_usnum + L_hse_sanction +
    L_growth + L_lnpercap + L_lnpop + L_polity2 + L_durable +
    L_civintensity + L_spending + L_s_us + L_lnustrade +
    L_us_distance + L_usstock2000_adj + allfdi2000,
    data = bk[train.samp, ], degree=3, pmethod = "backward")
```

Generating predictions:

```
h <- function(x){
  out <- eval(x)
  out <- out*(out > 0)
  out
}
X <- model.matrix(bk.e1)
hinges <- colnames(X)[-1]
hinges <- gsub("*", ":", hinges, fixed=TRUE)
form <- reformulate(hinges, response="usfdi2000_adj")
lmod1 <- lm(form, data=bk[test.samp, ])
lmod2 <- lm(form, data=bk[train.samp, ])
```

Notes

Type notes here...

Summary

```
summary(lmod1)
```

```
##
## Call:
## lm(formula = form, data = bk[test.samp, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.754  -1.439   1.442   2.656  10.274
##
## Coefficients:
##                                                                          Estimate
## (Intercept)                                                             3.269e-01
## h(L_usstock2000_adj - 6.53771)                                          6.859e-01
## h(6.53771 - L_usstock2000_adj)                                         -1.476e-01
## h(L_hse_usnum - 17)                                                     3.786e-01
## h(17 - L_hse_usnum)                                                     1.763e-01
## h(L_lnpop - 16.6189)                                                    3.585e-01
## h(L_hse_usnum - 20)                                                    -2.383e-01
## h(L_usstock2000_adj - 6.53771):h(L_growth - -3.5)                       6.052e-02
## h(L_usstock2000_adj - 6.53771):h(-3.5 - L_growth)                       7.856e-02
## h(L_usstock2000_adj - 6.53771):h(L_lnpercap - 10.2256)                 -9.912e+00
## h(L_hse_usnum - 17):h(0.814617 - L_s_us)                               -1.885e-01
## h(L_usstock2000_adj - 6.53771):h(8.30968 - L_lnustrade)                 5.019e-01
## h(L_hse_usnum - 20):h(-9 - L_polity2)                                  -3.030e-01
## h(L_usstock2000_adj - 6.53771):h(10.2256 - L_lnpercap):h(9.40286 - L_lnustrade) -1.324e-01
## h(L_usstock2000_adj - 6.53771):h(L_lnpercap - 10.2256):L_lnustrade      9.339e-01
## h(L_usstock2000_adj - 6.53771):h(L_lnustrade - 8.30968):h(21612.3 - allfdi2000)  2.400e-06
## h(L_usstock2000_adj - 6.53771):h(L_lnustrade - 8.30968):h(1.89 - L_growth)       3.078e-02
```

Notes

Type notes here...

Make PDP

```r
seq_range <- function(x, n=25){
  x <- na.omit(x);
  seq(min(x), max(x), length=n)
}
usnum.s <- seq_range(bk$L_hse_usnum)

predfun <- function(x,y){
  z <- bk[x, ]
  z$L_hse_usnum <- y
  predict(lmod1, newdata=z)
}
res <- outer(test.samp, usnum.s, predfun)

res <- t(apply(res, 1, function(x)x-mean(x)))
k <- lapply(2:10, function(k)kmeans(res, centers=k))
sapply(k, function(x)x$betweenss/(x$tot.withinss+x$betweenss))
```

```
## [1] 0.4453257 0.9078453 0.9475397 0.9785455 0.9440876 0.9890874 0.9893255
## [8] 0.9899874 0.9941667
```

```r
res2 <- rbind(k[[4]]$centers, res)
d2 <- dist(res2)
d2 <- as.matrix(d2)
diag(d2) <- max(c(d2))
closest <- apply(d2[1:5, ], 1, which.min)
```

```r
library(purrr)
tmp <- bk[test.samp[closest], ]
tmp$cluster <- 1:5
dats <- map(1:5, ~as.list(tmp[.x, ])) %>%
  map(., ~modify_at(.x, "L_hse_usnum", ~usnum.s)) %>%
  map(., ~do.call(data.frame, .x)) %>%
  bind_rows()

fits <- predict(lmod1, newdata=dats, se.fit=TRUE)
dats <- dats %>%
  mutate(fit = fits$fit) %>%
  group_by(cluster) %>%
  mutate(fit = fit-mean(fit)) %>%
  ungroup %>%
  mutate(lwr = fit - 1.96*fits$se.fit,
         upr = fit + 1.96*fits$se.fit)
```

Notes

Type notes here...

PDP

Notes

Type notes here...

Significant Differences

For how many observations are there significant differences moving across the values of `L_hse_usnum`?

```
sigdiffs <- NULL
combs <- combn(25, 2)
D <- matrix(0, ncol=ncol(combs), nrow=25)
D[cbind(combs[1,], 1:ncol(combs))] <- -1
D[cbind(combs[2,], 1:ncol(combs))] <- 1

for(i in 1:length(test.samp)){
  dk <- as.list(bk[test.samp[i], ])
  dk$L_hse_usnum <- usnum.s
  A <- model.matrix(formula(lmod1),
                    data=do.call(data.frame, dk))
  preds <- A %*% coef(lmod1)
  vpreds <- A %*% vcov(lmod1) %*% t(A)
  diffs <- c(t(D) %*% preds)
  vdiffs <- t(D) %*% vpreds %*% D
  tdiffs <- abs(diffs)/sqrt(diag(vdiffs))
  tdiffs <- ifelse(is.finite(tdiffs), tdiffs, 0)
  sigdiffs <- c(sigdiffs, sum(tdiffs > 1.96))
}
```