Using Stata 16's lasso features for prediction and inference

Di Liu

StataCorp

Motivation I: Prediction

What is a prediction?

- Prediction is to predict an outcome variable on new (unseen) data
- Good prediction minimizes mean-squared error (or another loss function) on new data

Examples:

- Given some characteristics, what would be the value of a house?
- Given an application of a credit card, what would be the probability of default for a customer?

Question:

Suppose I have many covariates, then which one should I include in my prediction model?

Motivation II: Inference

What we say

- Causal inference
- Somehow, we have a perfect model for both data and theory
- Report point estimates and standard errors

What we do

- Try many functional forms
- Pick up a “good” model that supports our story in mind
- Report the results as if there is no model-selection process

Question:

Suppose I have many potential controls, then which one should I include in my model to perform valid inference on some variables of interest? (Take into account the model-selection process.)

Overview of Stata 16's lasso features

- Lasso toolbox for prediction and model selection
  - **lasso** for lasso
  - **elasticnet** for elastic-net
  - **sqrtlasso** for square-root lasso
  - For linear, logit, probit, and Poisson models
- Cutting-edge estimators for inference after lasso model selection
  - double-selection: **dsregress**, **dslogit**, and **dspoisson**
  - partialing-out: **poregress**, **poivregress**, **pologit**, and **popoisson**
  - cross-fit partialing-out: **xporegress**, **xpoivregress**, **xpologit**, and **xpopoisson**
  - For linear, linear IV, logit, and Poisson models

Part I: Lasso for prediction

Using penalized regression to avoid overfitting

Why not include all potential covariates?

- It may not be feasible if $p > N$
- Even if it is feasible, too many covariates may cause overfitting
- Overfitting is the inclusion of extra parameters that reduce the in-sample loss but increase the out-of-sample loss

Penalized regression

$$\hat{\beta} = argmin_{\beta} \left\{ \sum_{i=1}^{N} L(x_i \beta', y_i) + P(\beta) \right\}$$

where $L()$ is the loss function and $P(\beta)$ is the penalization

estimator	$P(\beta)$
lasso	$\lambda \sum_{j=1}^{p} \vert\beta_j\vert$
elasticnet	$\lambda \left[\alpha \sum_{j=1}^{p} \vert\beta_j\vert + \frac{(1-\alpha)}{2} \sum_{j=1}^{p} \beta_j^2 \right]$

Example: Predicting housing value

Goal: Given some characteristics, what would be the value of a house?

data: Extract from American Housing Survey

characteristics: The number of bedrooms, the number of rooms, building age, insurance, access to Internet, lot size, time in house, and cars per person

variables: Raw characteristics and interactions (more than 100 variables)

Question: Among **OLS**, **lasso**, **elastic-net**, and **ridge** regression, which estimator should be used to predict the house value?

Load data and define potential covariates

```
. /*---------- load data ------------------------*/
.
. use housing, clear
.
. /*----------- define potential covariates ----*/
.
. local vlcont bedrooms rooms bag insurance internet tinhouse vpperson
. local vlfv lotsize bath tenure
. local covars `vlcont' i.(`vlfv')                                  ///
>         (c.(`vlcont') i.(`vlfv'))##(c.(`vlcont') i.(`vlfv'))
```

Step 1: Split data into a training and hold-out sample

Firewall principle

The training dataset used to train the model should not contain information from a hold-out sample used to evaluate prediction performance.

```
. /*---------- Step 1: split data --------------*/
.
. splitsample, generate(sample) split(0.70 0.30)
. label define lbsample 1 "traning" 2 "hold-out"
. label value sample lbsample
```

Step 2: Choose tuning parameter using training data

```
. /*---------- Step 2: run in traing sample ----*/
.
. quietly regress lnvalue `covars' if sample == 1
. estimates store ols
.
. quietly lasso linear lnvalue `covars' if sample == 1
. estimates store lasso
.
. quietly elasticnet linear lnvalue `covars' if sample == 1, alpha(0.2 0.5 0.75
>  0.9)
. estimates store enet
.
. quietly elasticnet linear lnvalue `covars' if sample == 1, alpha(0)
. estimates store ridge
```

- **if sample == 1** restricts the estimator to use training data only
- By default, we choose the tuning parameter by cross-validation
- We use **estimates store** to store lasso results
- In **elasticnet**, option **alpha()** specifies α in penalty term $\alpha||\beta||_1 + [(1-\alpha)/2]||\beta||_2^2$
- Specifying **alpha(0)** is ridge regression

Step 3: Evaluate prediction performance using hold-out sample

```
. /*---------- Step 3: Evaluate prediciton in hold-out sample ----*/
.
. lassogof ols lasso enet ridge, over(sample)
Penalized coefficients
```

Name	sample	MSE	R-squared	Obs
ols				
	traning	1.104663	0.2256	4,425
	hold-out	1.184776	0.1813	1,884
lasso				
	traning	1.127425	0.2129	4,396
	hold-out	1.183058	0.1849	1,865
enet				
	traning	1.124424	0.2150	4,396
	hold-out	1.180599	0.1866	1,865
ridge				
	traning	1.119678	0.2183	4,396
	hold-out	1.187979	0.1815	1,865

- We choose elastic-net as the best prediction because it has the smallest MSE in the hold-out sample

Step 4: Predict housing value using chosen estimator

```
. /*---------- Step 4: Predict housing value using chosen estimator -*/
.
. use housing_new, clear

. estimates restore enet
(results enet are active now)

.
. predict y_pen
(options xb penalized assumed; linear prediction with penalized coefficients)

.
. predict y_postsel, postselection
(option xb assumed; linear prediction with postselection coefficients)
```

- By default, **predict** uses the penalized coefficients to compute $x_i\beta'$
- Specifying option **postselection** makes **predict** use post-selection coefficients, which are from OLS on variables selected by **elasticnet**
- In the **linear** model, post-selection coefficients tend to be less biased and may have better out-of-sample prediction performance than the penalized coefficients

A closer look at lasso

Lasso (Tibshirani, 1996) is

$$\hat{\beta} = argmin_{\beta} \left\{ \sum_{i=1}^{N} L(x_i\beta', y_i) + \lambda \sum_{j=1}^{p} \omega_j |\beta_j| \right\}$$

where

- λ is the lasso penalty parameter and ω_j is the penalty loading
- We solve the optimization for a set of λ's
- The kink in the absolute value function causes some elements in $\hat{\beta}$ to be zero given some value of λ. Lasso is also a variable-selection technique
  - covariates with $\hat{\beta}_j = 0$ are excluded
  - covariates with $\hat{\beta}_j \neq 0$ are included
- Given a dataset, there exists a λ_{max} that shrinks all the coefficients to zero
- As λ decreases, more variables will be selected

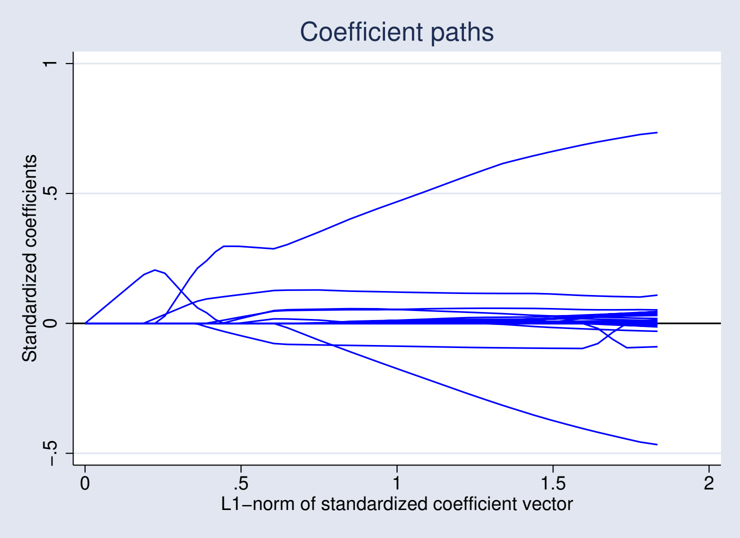
lasso output

```
. estimates restore lasso
(results lasso are active now)

. lasso

Lasso linear model                          No. of obs        =      4,396
                                            No. of covariates =        102
Selection: Cross-validation                 No. of CV folds   =         10

─────────────────────────────────────────────────────────────────────────
         |                                No. of      Out-of-      CV mean
         |                               nonzero       sample   prediction
      ID |     Description      lambda     coef.    R-squared        error
─────────┼───────────────────────────────────────────────────────────────
       1 |    first lambda    .4396153         0       0.0004     1.431814
      39 |   lambda before     .012815        21       0.2041     1.139951
    * 40 | selected lambda    .0116766        22       0.2043     1.139704
      41 |    lambda after    .0106393        23       0.2041     1.140044
      44 |     last lambda    .0080482        28       0.2011     1.144342
─────────────────────────────────────────────────────────────────────────
* lambda selected by cross-validation.
```

- We see the number of nonzero coefficients increases as λ decreases
- By default, **lasso** uses 10-fold cross-validation to choose λ

coefpath: Coefficients path plot

```
. coefpath
```



lassoknots: Display knot table

```
. lassoknots
```

ID	lambda	No. of nonzero coef.	CV mean pred. error	Variables (A)dded, (R)emoved, or left (U)nchanged
2	.4005611	1	1.399934	A 1.bath#c.insurance
7	.251564	2	1.301968	A 1.bath#c.rooms
9	.2088529	3	1.27254	A insurance
13	.1439542	4	1.235793	A internet
(output omitted ...)				
35	.0185924	19	1.143928	A c.insurance#c.tinhouse
37	.0154357	20	1.141594	A 2.lotsize#c.insurance
39	.012815	21	1.139951	A c.bage#c.bage 2.bath#c.bedrooms
39	.012815	21	1.139951	R 1.tenure#c.bage
* 40	.0116766	22	1.139704	A 1.bath#c.internet
41	.0106393	23	1.140044	A c.internet#c.vpperson
42	.0096941	23	1.141343	A 2.lotsize#1.tenure
42	.0096941	23	1.141343	R internet
43	.0088329	25	1.143217	A 2.bath#2.tenure 2.tenure#c.insurance
44	.0080482	28	1.144342	A c.rooms#c.rooms 2.tenure#c.bedrooms 1.lotsize#c.internet

* lambda selected by cross-validation.

- One λ is a knot if a new variable is added or removed from the model
- We can use **lassoselect** to choose a different λ. See lassoselect

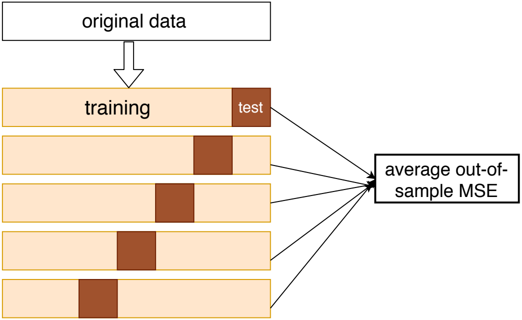
How to choose λ?

For **lasso**, we can choose λ by cross-validation, adaptive lasso, plugin, and customized choice.

- Cross-validation mimics the process of doing out-of-sample prediction. It produces estimates of out-of-sample MSE and selects λ with minimum MSE
- Adaptive lasso is an iterative procedure of cross-validated lasso. It puts more penalty weights on small coefficients than a regular lasso. Covariates with large coefficients are more likely to be selected, and covariates with small coefficients are more likely to be dropped
- Plugin method finds λ that is large enough to dominate the estimation noise

How does cross-validation work?

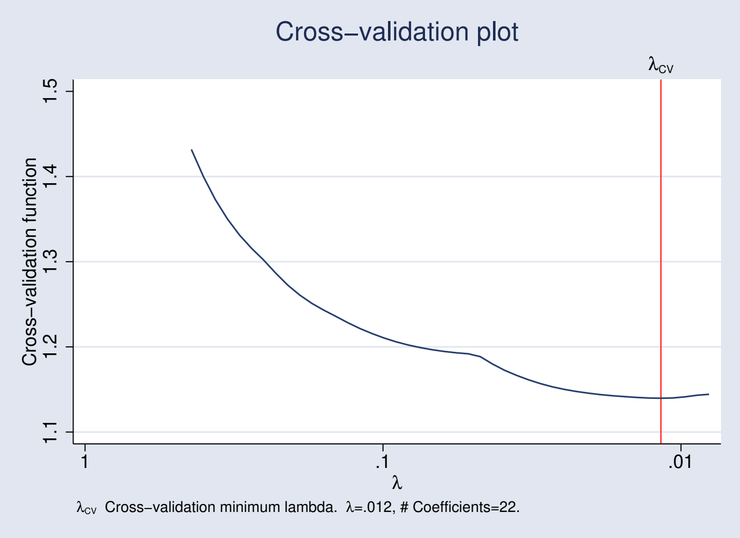
1. Based on data, compute a sequence of λ's as $\lambda_1 > \lambda_2 > \cdots > \lambda_k$. λ_1 set all the coefficients to zero (no variables are selected)
2. For each λ_j, do K-fold cross-validation to get an estimate of out-of-sample MSE



3. Select the λ^* with the smallest estimate of out-of-sample MSE, and refit lasso using λ^* and original data

cvplot: Cross-validation plot

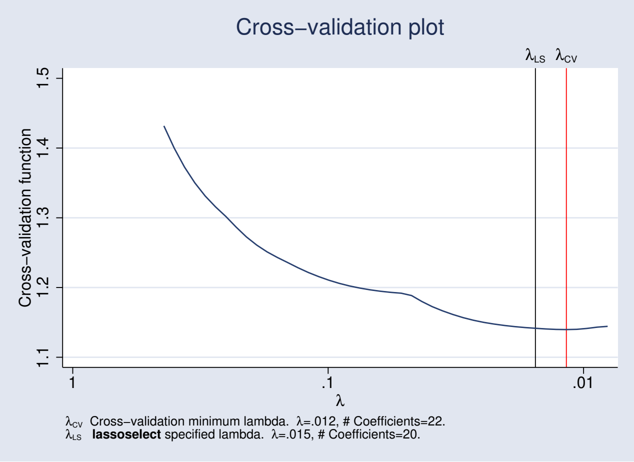
```
. cvplot
```



lassoselect: Manually choose a λ

- First, let's look at output from **lassoknots** lassoknots

```
. estimates restore lasso
(results lasso are active now)

. lassoselect id = 37
ID = 37  lambda = .0154357 selected

.
. cvplot
```



Use option **selection()** to choose λ

```
. quietly lasso linear lnvalue `covars'
. estimates store cv
.
. quietly lasso linear lnvalue `covars' , selection(adaptive)
. estimates store adaptive
.
. quietly lasso linear lnvalue `covars' , selection(plugin)
. estimates store plugin
```

lassoinfo: Lasso information summary

```
. lassoinfo cv adaptive plugin

    Estimate:  cv
     Command:  lasso
```

Depvar	Model	Selection method	Selection criterion	lambda	No. of selected variables
lnvalue	linear	cv	CV min.	.0034279	36

```
    Estimate:  adaptive
     Command:  lasso
```

Depvar	Model	Selection method	Selection criterion	lambda	No. of selected variables
lnvalue	linear	adaptive	CV min.	.0183654	16

```
    Estimate:  plugin
     Command:  lasso
```

Depvar	Model	Selection method	lambda	No. of selected variables
lnvalue	linear	plugin	.0537642	10

- Adaptive lasso selects fewer variables than regular lasso
- Plugin selects even fewer variables than adaptive lasso

Lasso toolbox summary

- Estimation:
  - **lasso**, **elasticnet**, and **sqrtlasso**
  - cross-validation, adaptive lasso, plugin, and customized
- Graph:
  - **cvplot**: cross-validation plot
  - **coefpath**: coefficient path
- Exploratory tools:
  - **lassoinfo**: summary of lasso fitting
  - **lassoknots**: detailed tabulate table of knots
  - **lassoselect**: manually select a tuning parameter
  - **lassocoef**: display lasso coefficients
- Prediction
  - **splitsample**: randomly divide data into different samples
  - **predict**: prediction for linear, binary, and count data
  - **lassogof**: evaluate in-sample and out-of-sample prediction

inference summary

Part II: Lasso for inference

Example: Air pollution effect

$$htime_i = no2_i\gamma + X_i\beta + \epsilon_i$$

htime	measure of the response time on test of child *i* (hit time)
*no*2	measure of the pollution level in the school of child *i*
X	vector of control variables that might need to be included

- Extract from Sunyer et al. (2017)
- There are 252 controls in *X*, but I only have 1,084 observations
- I cannot reliably estimate γ if I include all 252 controls

Question:

Which controls *X* should I put in my model to get valid inference on γ?

Load data and define controls

```
. /*------------ load data -------------------*/
.
. use breathe7
.
. /*------------ define controls -------------*/
.
. local ccontrols "sev_home sev_sch age ppt age_start_sch  oldsibl "
. local ccontrols "`ccontrols' youngsibl no2_home ndvi_mn noise_sch"
.
. local fcontrols "grade sex lbweight lbfeed smokep "
. local fcontrols "`fcontrols' feduc4 meduc4 overwt_who"
.
. local controls i.(`fcontrols') c.(`ccontrols')  ///
>         i.(`fcontrols')#c.(`ccontrols')
```

Mostly dangerous naive approach

$$htime_i = no2_i\gamma + X_i\beta + \epsilon_i$$

Naive approach

1. Select controls X^*
   - **regress htime** on **no2** and all X. Drop controls that are not significant at 5%
2. **regress htime** on **no2** and X^*
3. Perform inference on **no2** coefficient γ as if we only ran one regression

If you are doing this, the inference you get is mostly wrong.

Mostly dangerous naive approach

$$htime_i = no2_i\gamma + X_i\beta + \epsilon_i$$

Naive approach

1. Select controls X^*
   - **lasso htime** on **no2** and all X. **lasso** chooses the controls
2. **regress htime** on **no2** and X^*
3. Perform inference on **no2** coefficient γ as if we only ran one regression

If you are doing this, the inference you get is mostly wrong.

Things can go wrong even with only one control

- Consider a simple model:

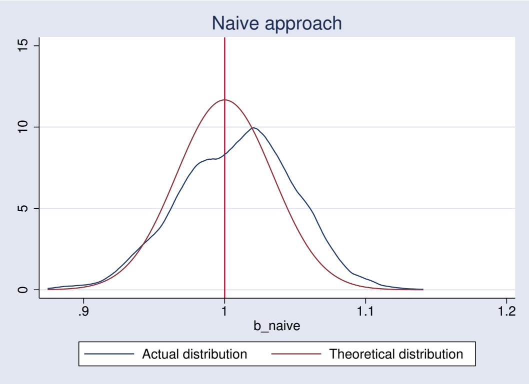
$$y_i = d_i\alpha + x_i\beta + \epsilon$$

- Do the following naive approach:
  1. **regress** y on d and x
  2. Drop x if it is not significant at 5%
  3. Rerun **regress** y on d if x is dropped; otherwise use the results from the first step

Problem:

You will get wrong inference on α if $|\beta|$ is close to zero but not equal to zero.

Why the naive approach fails?



- With real data, model-selection techniques inevitably make mistake about missing small β's
- The actual distribution of α is not concentrated (it has multiple modes). (Leeb and Pötscher, 2005) math

Solutions

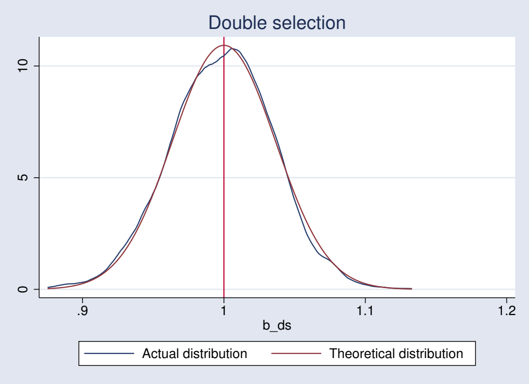
Pseudo-solutions:

- Assuming there is no small β's in the true model. It is known as the **beta-min** condition. (Too restrictive with real data)
- Do not do any selection (not reliable estimates when *p* is large; not feasible when $p > N$)

Realistic solutions: Be robust to model selection mistakes

- Double selection: Belloni et al. (2014), Belloni et al. (2016) (**dsregress**, **dslogit**, and **dspoisson**)
- Partialing-out: Belloni et al. (2016), Chernozhukov et al. (2015) (**poregress**, **poivregress**, **pologit**, and **popoisson**)
- Cross-fit Partialing-out (double machine learning): Chernozhukov et al. (2018) (**xporegress**, **xpoivregress**, **xpologit**, and **xpopoisson**)

Double selection works



Double-selection

1. **lasso** y on X, denote selected X as X_y^*
2. **lasso** d on X, denote selected X as X_d^*
3. **regress** y on d, X_y^*, and X_d^*

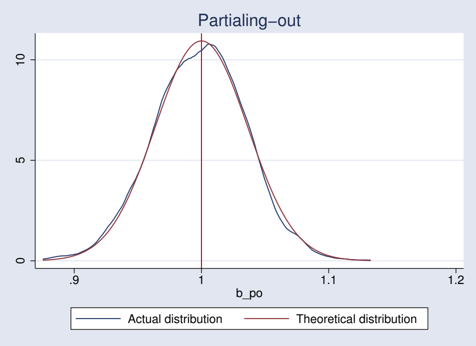
Intuition: The x's that are not selected in both step 1 and 2 have negligible impact on the distribution of α math

dsregress

```
. dsregress htime no2_class, controls(`controls')

Estimating lasso for htime using plugin
Estimating lasso for no2_class using plugin

Double-selection linear model          Number of obs               =      1,036
                                       Number of controls          =        252
                                       Number of selected controls =         11
                                       Wald chi2(1)                =      23.71
                                       Prob > chi2                 =     0.0000

------------------------------------------------------------------------------
             |               Robust
       htime |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   no2_class |   2.370022   .4867462     4.87   0.000     1.416017    3.324027
------------------------------------------------------------------------------
Note: Chi-squared test is a Wald test of the coefficients of the variables
      of interest jointly equal to zero. Lassos select controls for model
      estimation. Type lassoinfo to see number of selected variables in each
      lasso.
```

- **dsregress** selects only 11 controls among 252
- Another microgram of NO2 per cubic meter increases the mean reaction time by 2.37 milliseconds
- No free lunch. We cannot get inference on controls
- By default, lasso with plugin λ is used for all the variables

Partialing-out works



Partialing-out

1. **lasso** y on X, and get post-lasso residuals $\tilde{y} = y - X_y^* \hat{\beta}_y$
2. **lasso** d on X, and get post-lasso residuals $\tilde{d} = d - X_d^* \hat{\beta}_d$
3. **regress** $\tilde{y}$ on $\tilde{d}$

Intuition: Partialing-out is another form of double-selection

$$\tilde{y} = \tilde{d}\gamma + \epsilon \quad \Longrightarrow \quad y - X_y^* \hat{\beta}_y \;\; = d\gamma - X_d^* \hat{\beta}_d \gamma + \epsilon$$

poregress

```
. poregress htime no2_class, controls(`controls')

Estimating lasso for htime using plugin
Estimating lasso for no2_class using plugin

Partialing-out linear model          Number of obs               =      1,036
                                     Number of controls          =        252
                                     Number of selected controls =         11
                                     Wald chi2(1)                =      24.19
                                     Prob > chi2                 =     0.0000

------------------------------------------------------------------------------
             |               Robust
       htime |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   no2_class |   2.354892   .4787494     4.92   0.000     1.416561    3.293224
------------------------------------------------------------------------------
Note: Chi-squared test is a Wald test of the coefficients of the variables
      of interest jointly equal to zero. Lassos select controls for model
      estimation. Type lassoinfo to see number of selected variables in each
      lasso.
```

- **poregress** selects only 11 controls among 252
- Similar point estimate and standard error as in **dsregress**

Cross-fit partialing-out approach

Why cross-fit?

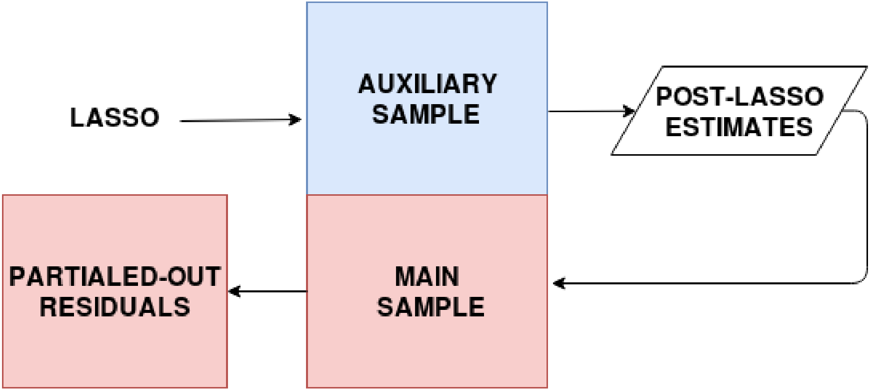
- To weaken sparsity condition
- To have better finite-sample property

Basic idea

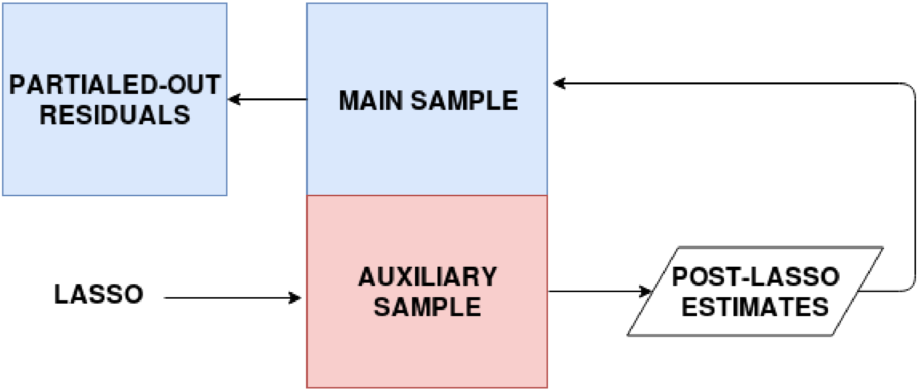
1. Split sample into auxiliary part and main part
2. All the machine-learning techniques are applied to the auxiliary sample
3. All the post-lasso residuals are obtained from the main sample
4. **Switch the role of auxiliary sample and main sample**, and do steps 2 and 3 again
5. Solving the moment equation using the full sample

Cross-fit needs to be combined with partialing-out; otherwise it has no effect.

2-fold cross-fit partialing-out (I)



2-fold cross-fit partialing-out (II)



xporegress

```
. xporegress htime no2_class, controls(`controls')

Cross-fit fold 1 of 10 ...
Estimating lasso for htime using plugin
Estimating lasso for no2_class using plugin

... output omitted

Cross-fit partialing-out            Number of obs                =      1,036
linear model                        Number of controls           =        252
                                    Number of selected controls  =         16
                                    Number of folds in cross-fit =         10
                                    Number of resamples          =          1
                                    Wald chi2(1)                 =      23.59
                                    Prob > chi2                  =     0.0000

------------------------------------------------------------------------------
             |               Robust
       htime |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   no2_class |   2.360406   .4859668     4.86   0.000     1.407928    3.312883
------------------------------------------------------------------------------
Note: Chi-squared test is a Wald test of the coefficients of the variables
      of interest jointly equal to zero. Lassos select controls for model
      estimation. Type lassoinfo to see number of selected variables in each
      lasso.
```

- By default, **xporegress** uses 10-fold cross-fitting
- **xporegress** ran 20 lassos in total (2 variables x 10 folds)
- By default, there is only one sample-splitting (resample = 1)
- We can use option **resample(#)** to get even more stable estimates

lassoinfo after **xporegress**

```
. lassoinfo

    Estimate:  active
     Command:  xporegress
```

Variable	Model	Selection method	No. of selected variables min	median	max
htime	linear	plugin	3	5	6
no2_class	linear	plugin	6	6	7

```
. lassoinfo, each

    Estimate:  active
     Command:  xporegress
```

Depvar	Model	Selection method	xfold no.	lambda	No. of selected variables
htime	linear	plugin	1	.1447945	5
htime	linear	plugin	2	.1448708	4
htime	linear	plugin	3	.1448708	5
(... output omitted)					
no2_class	linear	plugin	8	.1447945	7
no2_class	linear	plugin	9	.1447945	6
no2_class	linear	plugin	10	.1447945	6

- By default, **lassoinfo** displays summary of lassos by variable
- Option **each** displays information of each lasso

Compare naive with DS, PO, and XPO

```
. /*-------- double selection -------*/
. quietly dsregress htime no2_class, controls(`controls')

. estimates store ds

.
. /*-------- partialing-out -------*/
. quietly poregress htime no2_class, controls(`controls')

. estimates store po

.
. /*-------- cross-fitting partialing-out -------*/
. quietly xporegress htime no2_class, controls(`controls')

. estimates store xpo

.
. /*-------- naive approach-------*/
. quietly naive_regress, depvar(htime) dvar(no2_class) controls(`controls')

. estimates store naive

.
. /*-------- compare naive with ds, po, and xpo-------*/
. estimates table naive ds po xpo, se
```

Variable	naive	ds	po	xpo
no2_class	1.6830394	2.3700223	2.3548921	2.4405325
	.42522548	.48674624	.47874938	.48420429

legend: b/se

Recommendations

1. If you have time, use the cross-fit partialing-out estimator
   - **xporegress**, **xpologit**, **xpopoisson**, **xpoivregress**
2. If the cross-fit estimator takes too long, use either the partialing-out estimator
   - **poregress**, **pologit**, **popoisson**, **poivregress**

   or the double-selection estimator
   - **dsregress**, **dslogit**, **dspoisson**

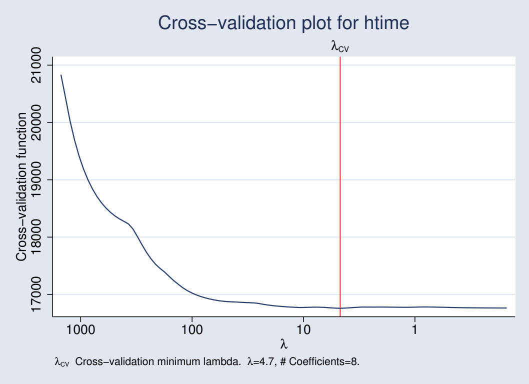
Control individual lasso

```
. /*-------- control lasso individually-------*/
. dsregress htime no2_class, controls(`controls')         ///
>         lasso(htime, selection(adaptive))                ///
>         sqrtlasso(no2_class, selection(cv))

Estimating lasso for htime using adaptive
Estimating square-root lasso for no2_class using cv

Double-selection linear model         Number of obs               =      1,036
                                      Number of controls          =        252
                                      Number of selected controls =         35
                                      Wald chi2(1)                =      23.76
                                      Prob > chi2                 =     0.0000

------------------------------------------------------------------------------
             |               Robust
       htime |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   no2_class |   2.457938   .5042238     4.87   0.000     1.469678    3.446199
------------------------------------------------------------------------------
Note: Chi-squared test is a Wald test of the coefficients of the variables
      of interest jointly equal to zero. Lassos select controls for model
      estimation. Type lassoinfo to see number of selected variables in each
      lasso.

. estimates store ds_cv
```

- Option **lasso()**: we use adaptive lasso for **htime**
- Option **sqrtlasso()**: we use cross-validated square-root lasso for **no2_class**

cvplot for a specified lasso

```
. /*--------- cvplot for htime -----*/
. cvplot, for(htime)
```



- Option **for()**: target the lasso that we want to explore
- The cross-validation function curve is pretty flat for **htime**

Sensitivity analysis (I)

Question: How sensitive is my result to the choice of λ?

```
. /*-------- lassoknots for htime-------*/
. lassoknots, for(htime)
```

ID	lambda	No. of nonzero coef.	CV mean pred. error	Variables (A)dded, (R)emoved, or left (U)nchanged
28	1368.541	1	20437.58	A 1.grade#c.noise_sch
43	338.998	2	18141.23	A 0.sex#c.age
45	281.4421	3	17866.4	A age
51	161.0515	4	17317.3	A 4.feduc4#c.age
66	39.89369	5	16867.32	A 1.sex#c.age_start_sch
70	27.49717	6	16851.58	A 3.grade#c.ndvi_mn
74	18.95273	7	16805.28	A 3.grade#c.noise_sch
83	8.204186	8	16778.24	A 2.meduc4
* 89	4.694737	8	16758.55	U
92	3.551396	9	16771.73	A 1.grade#c.youngsibl
93	3.2359	10	16776.5	A 2.feduc4#c.noise_sch
108	.8015572	11	16781.55	A 1.sex#c.youngsibl
126	.1501972	11	16763.33	U

* lambda selected by cross-validation in final adaptive step.

```
.
. /*-------- select a different lambda for htime-------*/
. lassoselect id = 70, for(htime)
```

ID = 70 lambda = 27.49717 selected

Sensitivity analysis (II)

```
. /*-------- reestimate model ---------------*/
. quietly dsregress, reestimate

. estimates store ds_sen

.
. /*-------- compare with old result ---------------*/
. estimates table ds_cv ds_sen, se
```

Variable	ds_cv	ds_sen
no2_class	2.4579381	2.4739541
	.5042238	.50097675

legend: b/se

- Option **reestimate**: re-estimate the model with changes in some lassos while holding the other part fixed

Big picture

$$E(\underbrace{y}_{\text{outcome}}) = G\left(\underbrace{D}_{\text{variables of interest}} \overbrace{\alpha}^{\text{effect}} + \underbrace{m(x)}_{\text{controls}}\right)$$

- $G()$ is the link function
- Goal: perform valid inference on α without knowing which controls should be in the model
- X is high-dimensional, and D is low-dimensional
- We are assuming that $m(x)$ can be reasonably approximated by a **sparse** $X\beta$

DS, PO, and XPO in a nutshell

DS, PO, and XPO methods can be summarized as constructing a moment condition

$$E[\psi(\underbrace{W}_{\textbf{data}};\overbrace{\alpha}^{\textit{effect}},\underbrace{\eta}_{\text{nuisance parameter}})] = 0$$

such that

$$\partial_\eta E[\psi(\underbrace{W}_{\textbf{data}};\overbrace{\alpha}^{\textit{effect}},\underbrace{\eta}_{\text{nuisance parameter}})]\Big|_{\eta=\eta_0} = 0$$

- Neyman orthogonality: $\psi()$ is robust to mistakes in estimating nuisance parameters
- A broad class of machine-learning techniques (not just lasso) can be used to estimate the nuisance parameters η (β in lasso case)
- We can get valid inference on α
- No free lunch. We cannot get inference on η

Summary of Stata's lasso inference commands

Estimation:

- **ds***, **po***, and **xpo*** (11 estimation commands)
- Robust to the model-selection mistakes
- Valid inference on some variables of interest
- High-dimensional potential controls
- Partial linear, IV, logit, and Poisson models
- Flexible control of individual lassos

Post-estimation:

- Most post-estimation commands in the lasso toolbox also work here (except **lassogof**) toolbox summary
- Traditional post-estimation commands (**test**, **contrast**, etc.)

Appendix: Why the naive approach fails?

- Let's define M as Model, R as Restricted model ($\beta_0 = 0$), U as Unrestricted model ($\beta_0 \neq 0$)

$$
\begin{aligned}
Pr(\hat{\alpha} < t) &= Pr(\hat{\alpha_R} < t)Pr(M = R) + Pr(\hat{\alpha_U} < t)Pr(M = U) \\
&= Pr(\hat{\alpha_R} < t)Pr(|\hat{\beta}_U/\hat{\sigma_\beta}| \leq c) + Pr(\hat{\alpha_U} < t)Pr(|\hat{\beta}/\hat{\sigma_\beta}| > c)
\end{aligned}
$$

- If $\beta_0 \propto \frac{1}{\sqrt{N}}$, $Pr(|\hat{\beta}_U/\hat{\sigma_\beta}| \leq c) \to 1$ (This means we are going to choose the wrong model!)
- In a finite sample, $Pr(\hat{\alpha} < t)$ is a mixture of two distributions, and neither of them dominates (that's why we see two modes)

back

Appendix: Why double selection works?

- Let's consider this simple model

$$y = d\alpha + x\beta + \epsilon$$
$$d = x\gamma + u$$

- If x is dropped , then

$$\sqrt{n}(\hat{\alpha} - \alpha) = \text{good terms} + \sqrt{n}(d'd)^{-1}(x'x)\beta\gamma$$

- Naive approach drops x if $\beta \propto 1/\sqrt{n}$, so

$$\sqrt{n}(d'd)^{-1}(x'x)\beta\gamma \propto \sqrt{n}(d'd)^{-1}(x'x)1/\sqrt{n}\gamma \neq 0$$

- Double selection drops x if $\beta \propto 1/\sqrt{n}$ and $\gamma \propto 1/\sqrt{n}$

$$\sqrt{n}(d'd)^{-1}(x'x)\beta\gamma \propto \sqrt{n}(d'd)^{-1}(x'x)1/\sqrt{n}1/\sqrt{n} \to 0$$

back

References

Belloni, A., V. Chernozhukov, and C. Hansen. 2014. Inference on treatment effects after selection among high-dimensional controls. *The Review of Economic Studies* 81(2): 608–650.

Belloni, A., V. Chernozhukov, and Y. Wei. 2016. Post-selection inference for generalized linear models with many controls. *Journal of Business & Economic Statistics* 34(4): 606–619.

Chernozhukov, V., D. Chetverikov, M. Demirer, E. Duflo, C. Hansen, W. Newey, and J. Robins. 2018. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* 21(1): C1–C68.

Chernozhukov, V., C. Hansen, and M. Spindler. 2015. Post-selection and post-regularization inference in linear models with many controls and instruments. *American Economic Review* 105(5): 486–90.

Leeb, H., and B. M. Pötscher. 2005. Model selection and inference: Facts and fiction. *Econometric Theory* 21(1): 21–59.

Sunyer, J., E. Suades-González, R. García-Esteban, I. Rivas, J. Pujol, M. Alvarez-Pedrerol, J. Forns, X. Querol, and X. Basagaña. 2017.

Traffic-related air pollution and attention in primary school children: short-term association. *Epidemiology (Cambridge, Mass.)* 28(2): 181.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58(1): 267–288.