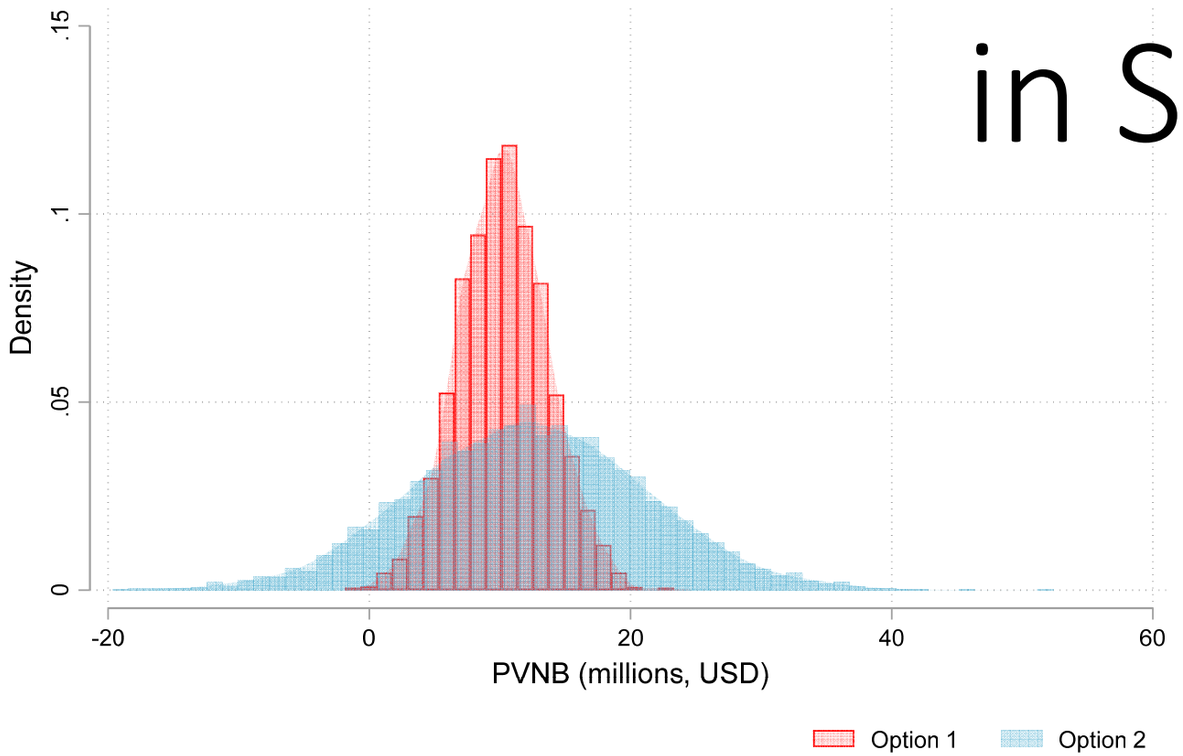
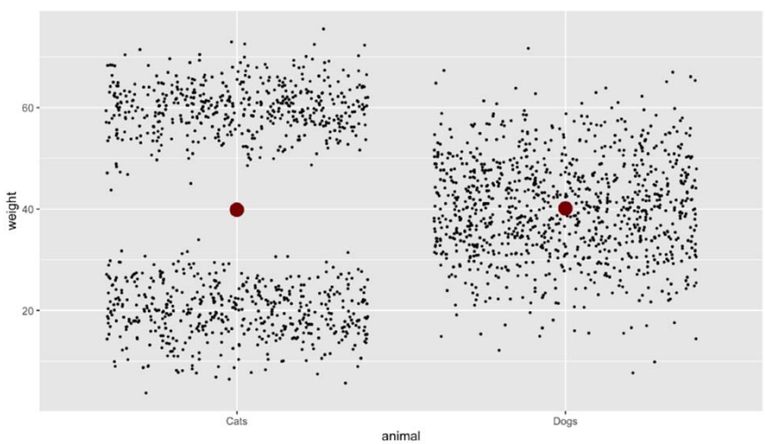
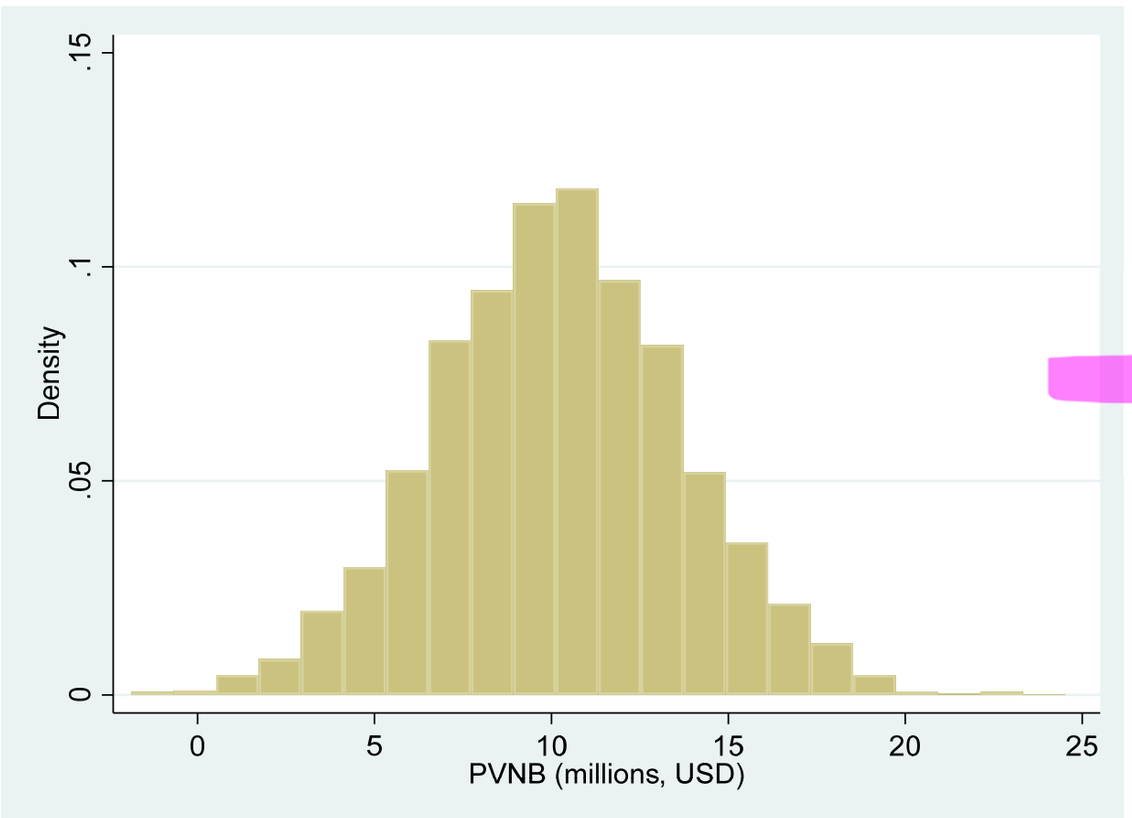
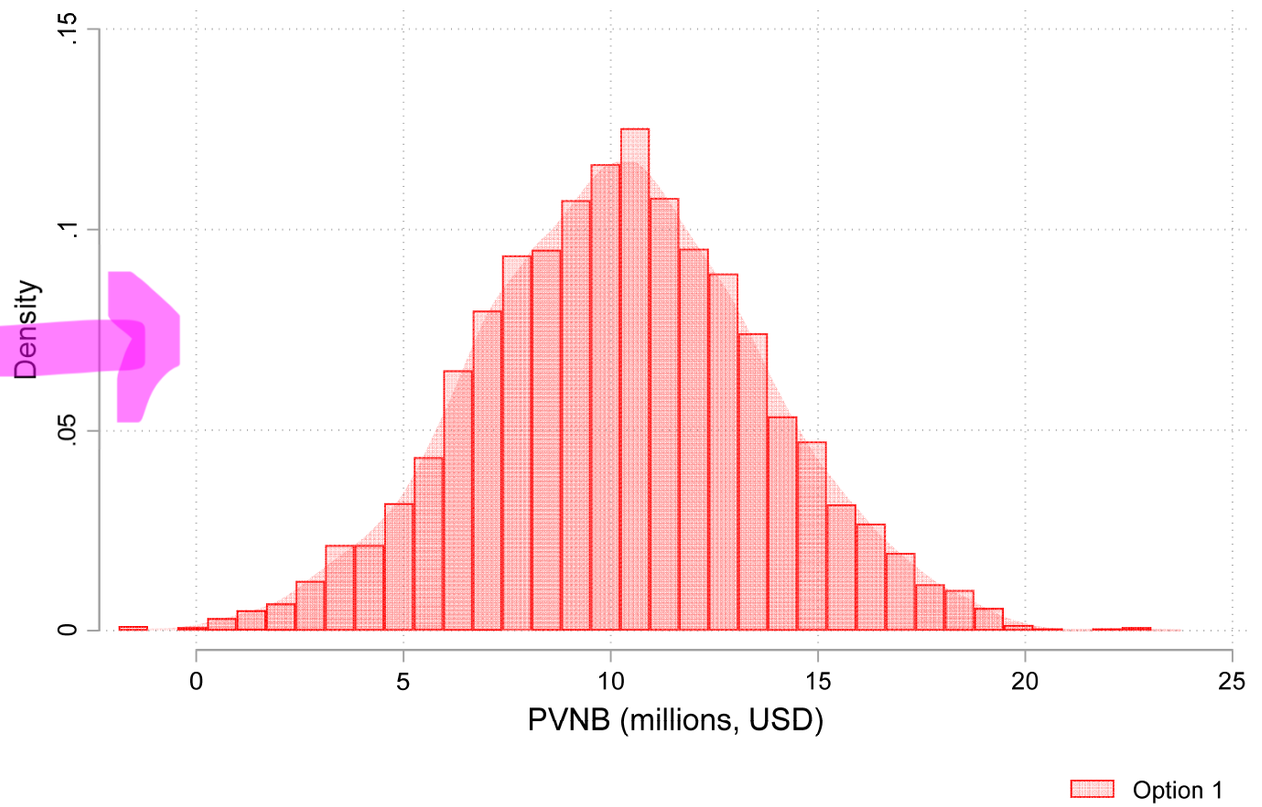
Sensitivity analysis with Monte Carlo in Stata





+ Graphing tips!



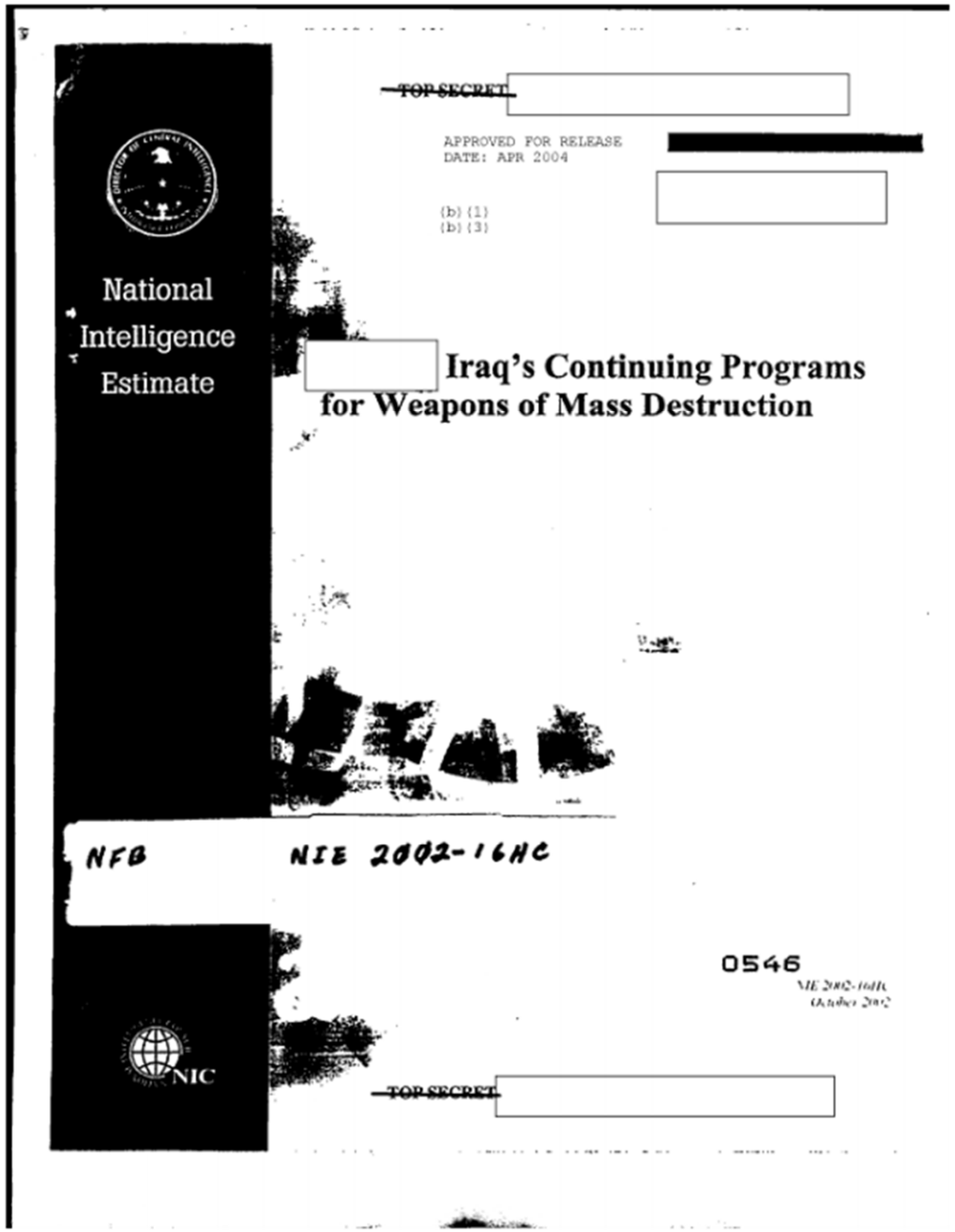


Plan for today

1. Uncertainty matters
2. Stata basics
3. How to set up PVNB in Stata
4. Adding uncertainty (Monte Carlo)
5. + How not to make ugly graphs

Uncertainty matters

It is difficult to communicate uncertainty (and risk)
…but very important!



Key Judgments [from October 2002 NIE]

Iraq's Continuing Programs for Weapons of Mass Destruction

We judge that Iraq has continued its weapons of mass destruction (WMD) programs in defiance of UN resolutions and restrictions. Baghdad has chemical and biological weapons as well as missiles with ranges in excess of UN restrictions; if left unchecked, it probably will have a nuclear weapon during this decade. (See INR alternative view at the end of these Key Judgments.)

We judge that we are seeing only a portion of Iraq's WMD efforts, owing to Baghdad's vigorous denial and deception efforts. Revelations after the Gulf war starkly demonstrate the extensive efforts undertaken by Iraq to deny information. We lack specific information on many key aspects of Iraq's WMD programs.

Since inspections ended in 1998, Iraq has maintained its chemical weapons effort, energized its missile program, and invested more heavily in biological weapons; in the view of most agencies, Baghdad is reconstituting its nuclear weapons program.

Fall, 2002: intelligence reports in US & UK made a case for invading Iraq

After invasion in 2003: no evidence of active WMDs, no efforts to restart a nuclear program

A US Senate committee investigation expressed strong criticism:

"The Intelligence Community did not accurately or adequately explain to policymakers the uncertainties behind the judgments in the … NIE."

Key Judgments [from October 2002 NIE]

Iraq's Continuing Programs for Weapons of Mass Destruction

We judge that Iraq has continued its weapons of mass destruction (WMD) programs in defiance of UN resolutions and restrictions. Baghdad has chemical and biological weapons as well as missiles with ranges in excess of UN restrictions; if left unchecked, it probably will have a nuclear weapon during this decade. (See INR alternative view at the end of these Key Judgments.)

We judge that we are seeing only a portion of Iraq's WMD efforts, owing to Baghdad's vigorous denial and deception efforts. Revelations after the Gulf war starkly demonstrate the extensive efforts undertaken by Iraq to deny information. We lack specific information on many key aspects of Iraq's WMD programs.

Since inspections ended in 1998, Iraq has maintained its chemical weapons effort, energized its missile program, and invested more heavily in biological weapons; in the view of most agencies, Baghdad is reconstituting its nuclear weapons program.

It is difficult to communicate uncertainty (and risk)
…but very important!

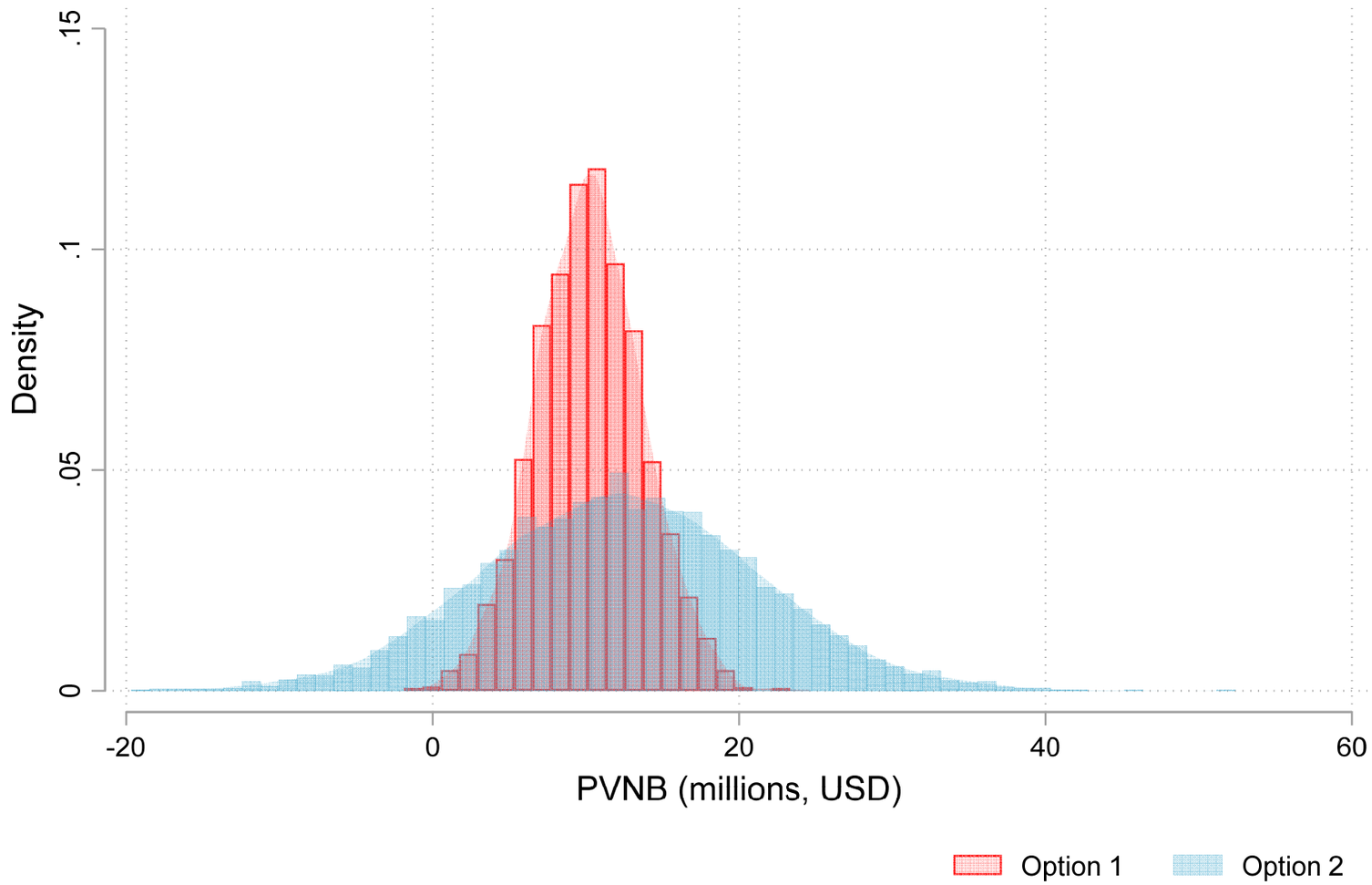
Every CBA has both risks and uncertainty

If you do not communicate these carefully, you may lead decision-makers to put too much weight on your base case

Compare this project summary

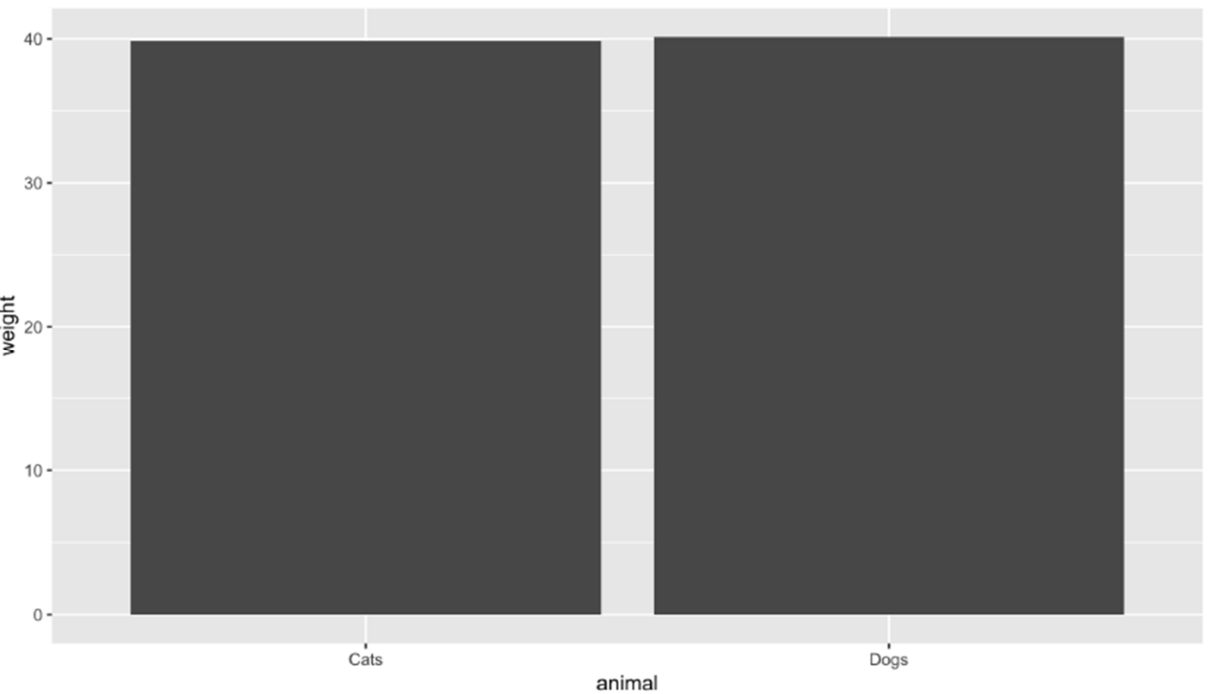
	Alternative	
	Basic alternative	**Bells and whistles**
PV(C)	…	…
PV(B)	…	…
PVNB	**USD 10.2 Million**	**USD 12.3 Million**

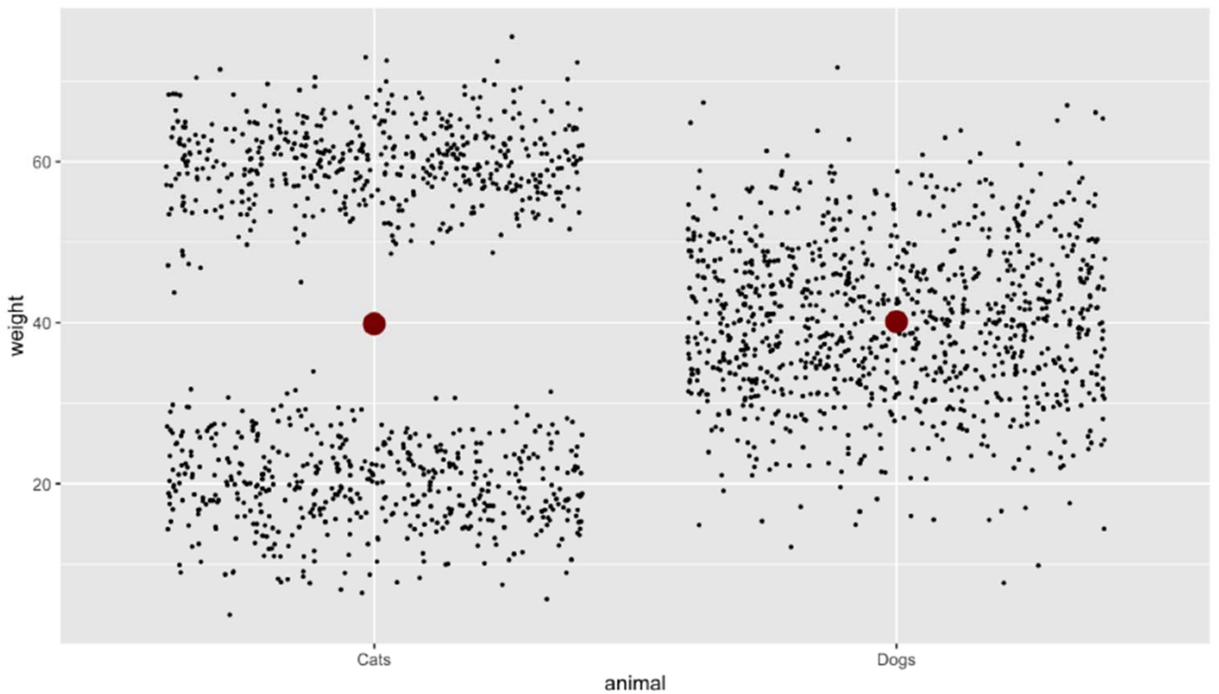
…to this one



Problems with simple numbers

What's wrong with simple summary statistics?





Sensitivity analysis with Monte Carlo

We typically have some components in our analysis that we are willing to believe are certain/fixed (or that are choice variables)

already incurred fixed costs

number of trucks needed to transport construction materials

number of miles of road to rehabilitate

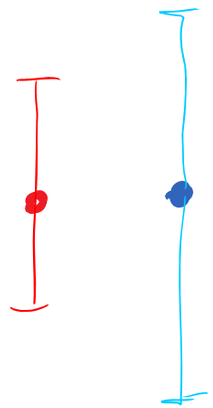
Sensitivity analysis with Monte Carlo

We typically have some components in our analysis that we are willing to believe are certain/fixed (or that are choice variables)

- already incurred fixed costs
- number of trucks needed to transport construction materials
- number of miles of road to rehabilitate

Many other components are uncertain

- price of raw materials
- maintenance costs
- estimated treatment effects
- returns to education



Sensitivity analysis with Monte Carlo

Many other components are uncertain

- price of raw materials
- maintenance costs
- estimated treatment effects
- returns to education

Sensitivity analysis with Monte Carlo

We account for this through Monte Carlo simulations by drawing from ranges of values, based on conditions that we specify

We specify some features of this “range”

In other words, we model uncertainty by assigning a probability distribution to items in the model

Some Stata basics

Basic syntax

The anatomy of a Stata command:

```
[prefix :] command [varlist] [=exp] [if] [in]
[weight] [using filename] [, options]
```

Navigating within Stata …not STATA!

Commands to get around Stata

```
cd
mkdir
copy
pwd
dir/ls
erase
rmdir
shell
```

From the Stata FAQs:

18. What was that comment on Stata, not STATA, about?

The correct spelling is “Stata”, please, not “STATA”. Several of the most active experts on the list can get a little irritated if you get that wrong, although you are free to regard them as pedantic. More importantly, if you write “STATA” you are making it obvious that you didn't read this guide carefully and to the end.

https://www.statalist.org/forums/help#spelling

Don't know how to use these?

`help` is your friend

```
help mkdir
```

Macros in Stata

Macros in Stata are very different from macros in Excel

A Stata macro is like a box you put text in for future use
especially useful when you use it to run a single command many times with a *different* bit of text each time (loops)

Local macros: `` `macro1' ``

Global macros: `$macro`

Macros in Stata

Two different ways to define a local macro

```
local name value
local one 1
```

This creates a local macro called `one` and puts the character 1 in it

> Note: this is not the same as the **value** one

To use a macro, put its name inside ` '

Macros in Stata

Macros get processed before they get passed on to Stata

When Stata sees a macro, it replaces it with whatever was in the "box"

```
local one 1
di `one'
```

Stata saw:

```
di 1
```

```
* Try a more complicated macro:
  local two 2 + 2
  display `two'
```

Stata saw

```
  di 2 + 2
```

Macros in Stata

If you want `display` to show something on the screen without evaluating it, put it in quotes

```
* Tell Stata to display as string:
  di "`two'"
```

If you instead want to have a macro that contains *the result of a calculation*, put an equal sign after the macro name

```
  local three = 2 + 2
  display `three'
* Now {three} really contains 4 and not 2 + 2
* ...even if you put it in quotes:
  display "`three'"
```

Macros in Stata

You can also force the processor to evaluate *expressions*:

```
* This is not getting evaluated by display
  display "`= 2+ 2'"
  display "= _N" // this displays the obs number
```

And importantly, macros can contain other macros

So if you have a macro `year` containing 2019...

```
  `=`year'+1'
```

...evaluates to 2020

Ok, but what are they good for?

Macros in Stata

```
* Specify list of control variables
  local controlVars age female occupation
* Run first regression
  reg income treatment `controlVars', r

/* ...Many lines of code later */

* Run another regression
  logit employed treatment `controlVars'
```

Shorter, less error-prone code

Macros in Stata

Consistent subsamples

```
* Define subsamples
  local blackWoman race==1 & female
  local hispMan race==2 & !female
* Run regression
  reg income treatment `controlVars' if `blackWoman', r
  reg income treatment `controlVars' if `hispMan', r
```

Macros in Stata

Loops!

```
foreach macro in list {
    commands
}
```

Key for coding!

```
foreach color in red blue green {
    di "`color'"
}
```

list

color is the name of the macro containing list elements

spaces → new element

Let’s to some CBA!

Basic parameters

```
*************************************************************
* 1 - Set-up
*************************************************************
    clear all
* Set a seed to ensure replicability
    set seed 101010101
* The number of observations equals the number of ...?
  set obs 500
  gen one = 1
  br one
```

Basic parameters

```
* Let's build a pool!
* Define some basic parameters for building a pool:
* One-time costs and benefits:
  gen construction_costs = 14.5
  lab var construction_costs "Construction costs, million USD"

  gen scrap_value = 0.8
  lab var scrap_value "Scrap value, million USD"
```

Basic parameters

```
* Ongoing costs and benefits:
  gen maintenance_costs = 6.2
  lab var maintenance_costs "Maintenance costs, million USD"
  gen revenues = 6.6
  lab var revenues "Yearly revenues, million USD"

  gen consumer_surplus = 1.37
  lab var consumer_surplus "Consumer surplus generated, million USD"

  scalar discount_rate = .07
```

```
* Generate a variable that contains PVNB
* Let's look just at 10 years for simplicity
* Assume costs incurred in year 0, and revenues. maintenance kick in in year 1
    gen pvnb = -construction_costs ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^1 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^2 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^3 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^4 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^5 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^6 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^7 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^8 ///
    + (revenues + consumer_surplus - maintenance_costs)/(1+discount_rate)^9 ///
    + (revenues + consumer_surplus - maintenance_costs + ///
    scrap_value)/(1+discount_rate)^10
```

Look at the base case value

```
* Now examine PVNB
  sum pvnb
  di in red "Our base case results in PVNB = " `r(mean)'
```

Variable	Obs	Mean	Std. Dev.	Min	Max
pvnb	500	-1.661581	0	-1.661581	-1.661581

```
. di in red "Our base case results in PVNB = " pvnb
Our base case results in PVNB = -1.6615806
```

to see what is stored after summarize pvnb, type return list

Look at the base case value

```
* Now examine PVNB
  sum pvnb
  di in red "Our base case results in PVNB = " `r(mean)'
```

scalars:

```
      r(N) =  500
  r(sum_w) =  500
   r(mean) =  -1.661580562591553
    r(Var) =  0
     r(sd) =  0
    r(min) =  -1.661580562591553
    r(max) =  -1.661580562591553
    r(sum) =  -830.7902812957764
```

Variable	Obs	Mean	Std. Dev.	Min	Max
pvnb	500	-1.661581	0	-1.661581	-1.661581

```
. di in red "Our base case results in PVNB = " pvnb
Our base case results in PVNB = -1.6615806
```

Data Editor (Browse) - [Untitled]

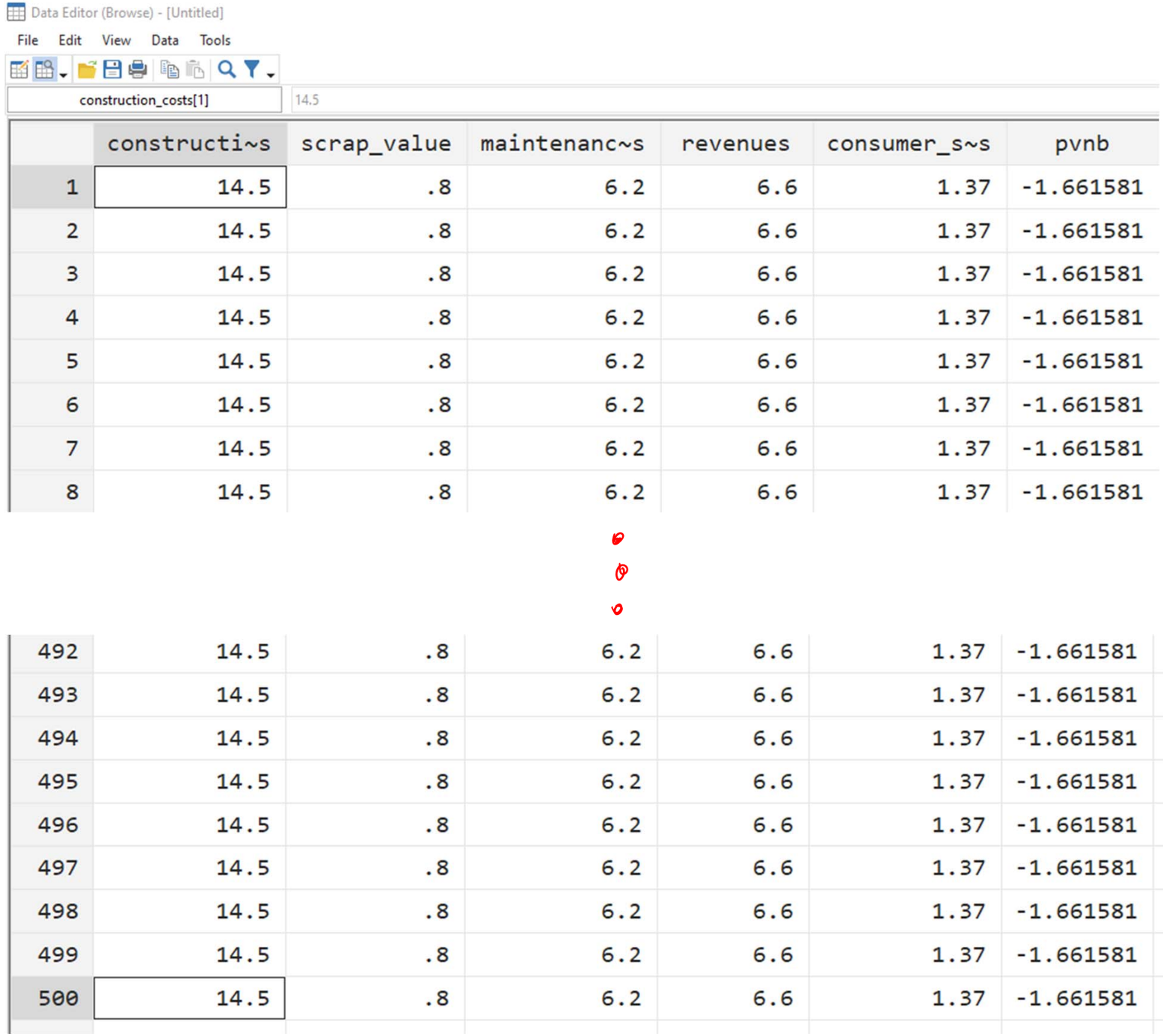
File Edit View Data Tools

construction_costs[1] 14.5

	constructi~s	scrap_value	maintenanc~s	revenues	consumer_s~s	pvnb
1	14.5	.8	6.2	6.6	1.37	-1.661581
2	14.5	.8	6.2	6.6	1.37	-1.661581
3	14.5	.8	6.2	6.6	1.37	-1.661581
4	14.5	.8	6.2	6.6	1.37	-1.661581
5	14.5	.8	6.2	6.6	1.37	-1.661581
6	14.5	.8	6.2	6.6	1.37	-1.661581
7	14.5	.8	6.2	6.6	1.37	-1.661581
8	14.5	.8	6.2	6.6	1.37	-1.661581
⋮						
492	14.5	.8	6.2	6.6	1.37	-1.661581
493	14.5	.8	6.2	6.6	1.37	-1.661581
494	14.5	.8	6.2	6.6	1.37	-1.661581
495	14.5	.8	6.2	6.6	1.37	-1.661581
496	14.5	.8	6.2	6.6	1.37	-1.661581
497	14.5	.8	6.2	6.6	1.37	-1.661581
498	14.5	.8	6.2	6.6	1.37	-1.661581
499	14.5	.8	6.2	6.6	1.37	-1.661581
500	14.5	.8	6.2	6.6	1.37	-1.661581

Kind of boring since they're all the same!

Was there any point in doing this calculation 500 times?



	constructi~s	scrap_value	maintenanc~s	revenues	consumer_s~s	pvnb
1	14.5	.8	6.2	6.6	1.37	-1.661581
2	14.5	.8	6.2	6.6	1.37	-1.661581
3	14.5	.8	6.2	6.6	1.37	-1.661581
4	14.5	.8	6.2	6.6	1.37	-1.661581
5	14.5	.8	6.2	6.6	1.37	-1.661581
6	14.5	.8	6.2	6.6	1.37	-1.661581
7	14.5	.8	6.2	6.6	1.37	-1.661581
8	14.5	.8	6.2	6.6	1.37	-1.661581
⋮						
492	14.5	.8	6.2	6.6	1.37	-1.661581
493	14.5	.8	6.2	6.6	1.37	-1.661581
494	14.5	.8	6.2	6.6	1.37	-1.661581
495	14.5	.8	6.2	6.6	1.37	-1.661581
496	14.5	.8	6.2	6.6	1.37	-1.661581
497	14.5	.8	6.2	6.6	1.37	-1.661581
498	14.5	.8	6.2	6.6	1.37	-1.661581
499	14.5	.8	6.2	6.6	1.37	-1.661581
500	14.5	.8	6.2	6.6	1.37	-1.661581

Kind of boring since they're all the same!

Was there any point in doing this calculation 500 times?

NOPE!!

Introduce uncertainty

```
*************************************************************************
* 2 - Introduce uncertainty
*************************************************************************
* So what about if we feel uncertain about our estimate of consumer surplus?
* Let's make a small Monte Carlo exercise, drawing from a uniform distribution

gen consumer_surplus_uniform = runiform(0, 2.74)
lab var consumer_surplus_uniform "Consumer surplus, drawn from a uniform"
br consumer_surplus_uniform
```

Introduce uncertainty

```
*******************************************************************
* 2 - Introduce uncertainty
*******************************************************************
* So what about if we feel uncertain about our estimate of consumer surplus?
* Let's make a small Monte Carlo exercise, drawing from a uniform distribution
gen consumer_surplus_uniform = runiform(0, 2.74)
lab var consumer_surplus_uniform "Consumer surplus, drawn from a uniform"
br consumer_surplus_uniform
```

generate

random draw from uniform distribution

browse

browse consumer* →

Data Editor (Browse) - [Untitled]

File Edit View Data Tools

var11[1]

	consumer_s~s	consumer_s~m
1	1.37	2.313121
2	1.37	.9222178
3	1.37	.2095526
4	1.37	.6518665
5	1.37	.4423076
6	1.37	2.535007
7	1.37	1.025612
8	1.37	.4793074
9	1.37	1.44813
10	1.37	1.930707
11	1.37	.76757
12	1.37	.7744743
13	1.37	.3206741

Now that's a bit more interesting

Introduce uncertainty

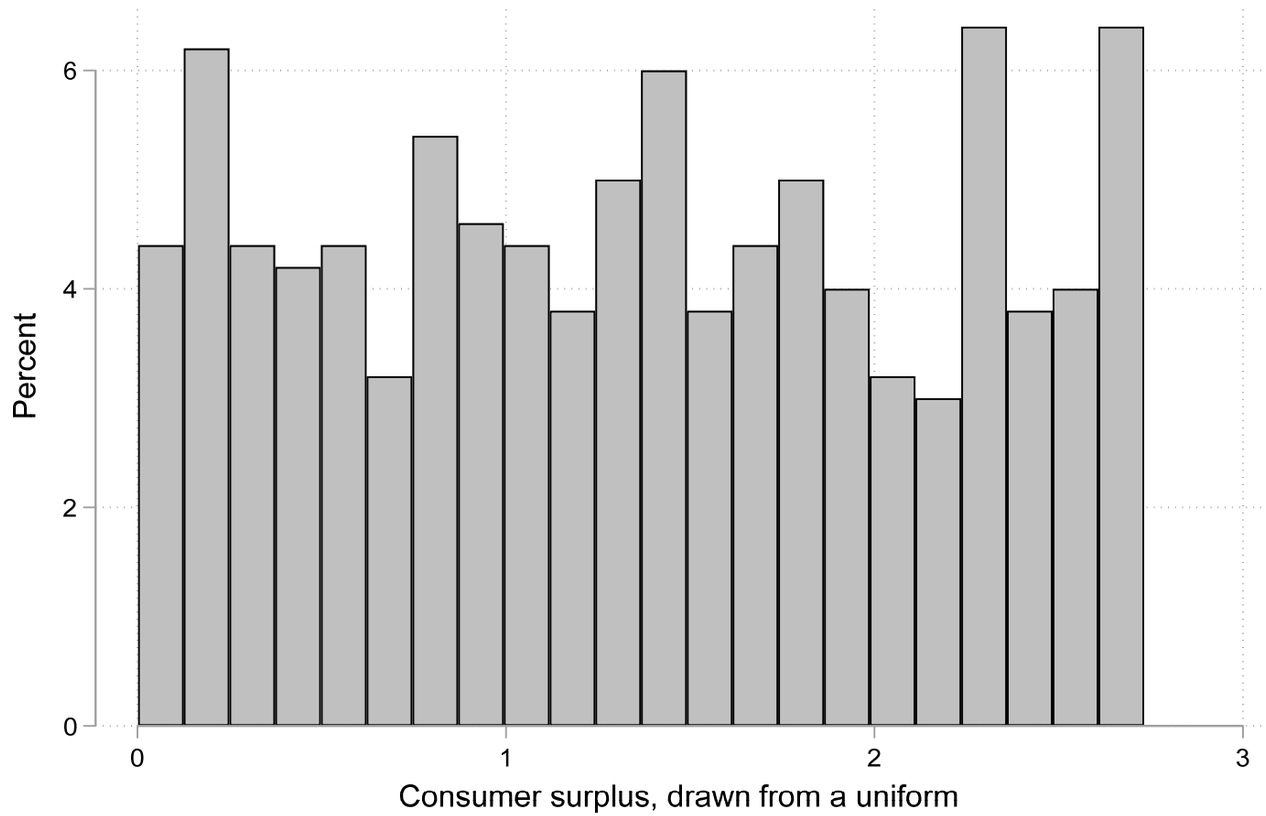
```
*************************************************************************
* 2 - Introduce uncertainty
*************************************************************************
* So what about if we feel uncertain about our estimate of consumer surplus?
* Let's make a small Monte Carlo exercise, drawing from a uniform distribution

gen consumer_surplus_uniform = runiform(0, 2.74)
lab var consumer_surplus_uniform "Consumer surplus, drawn from a uniform"
br consumer_surplus_uniform
hist consumer_surplus_uniform, percent scheme(plotplain)
```

↳ histogram

ssc install blindschemes

Introduce uncertainty

```
hist consumer_surplus_uniform, percent scheme(plotplain)
```



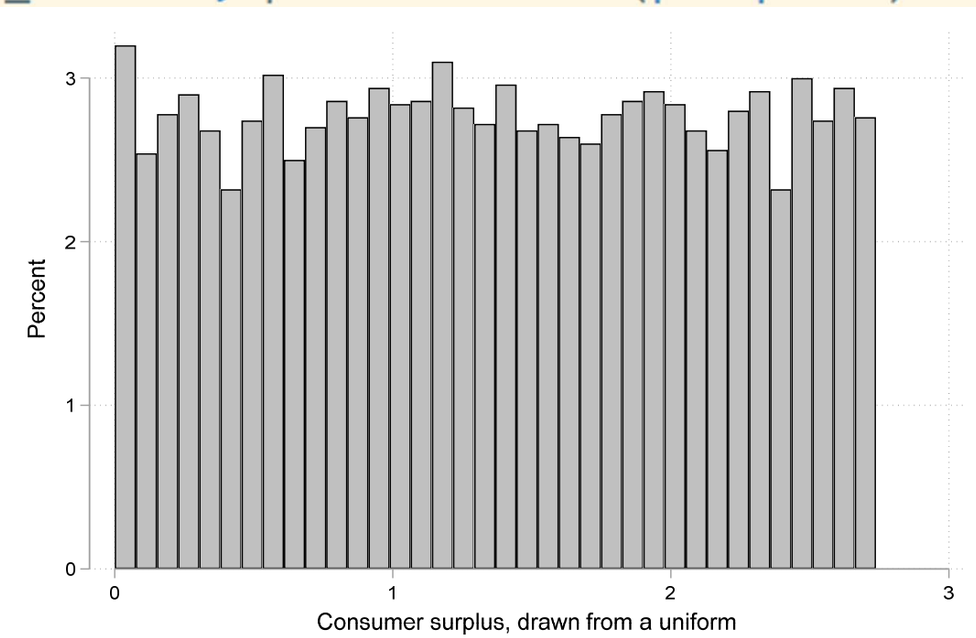
Introduce uncertainty

```
hist consumer_surplus_uniform, percent scheme(plotplain)

* That looks pretty lumpy --> go up to top and set obs to 5000 instead
* set obs 5000  (on line 55)
hist consumer_surplus_uniform, percent scheme(plotplain)
```

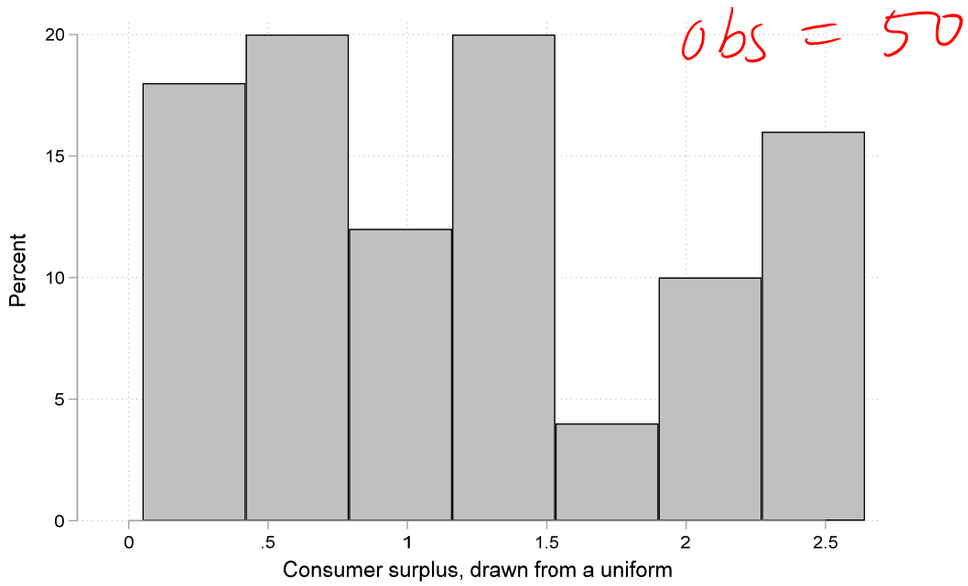
Introduce uncertainty

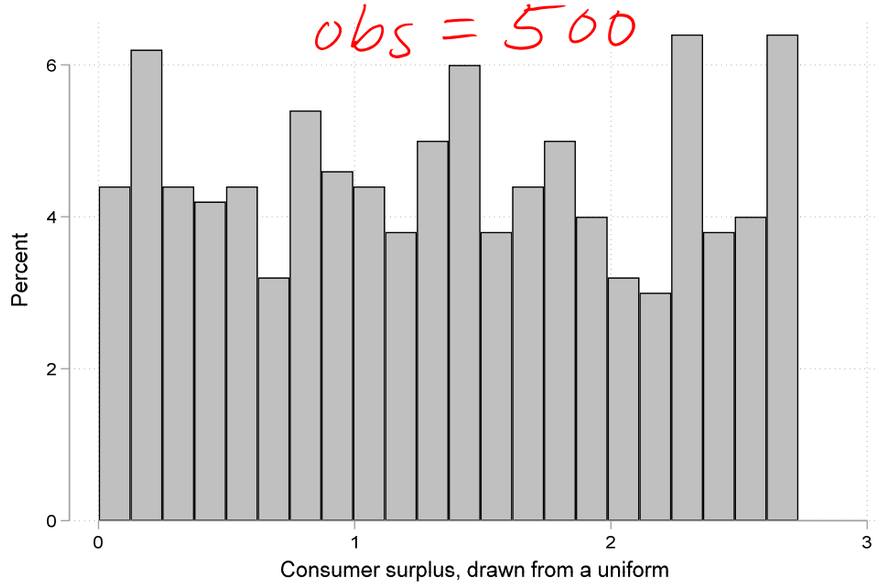
```
hist consumer_surplus_uniform, percent scheme(plotplain)

* That looks pretty lumpy --> go up to top and set obs to 5000 instead
* set obs 5000
hist consumer_surplus_uniform, percent scheme(plotplain)
```

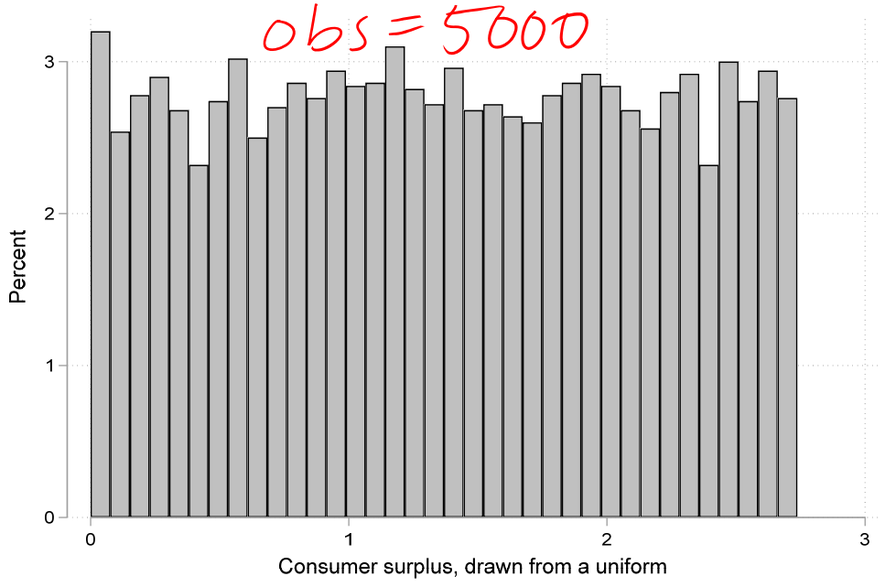
(on line 55)

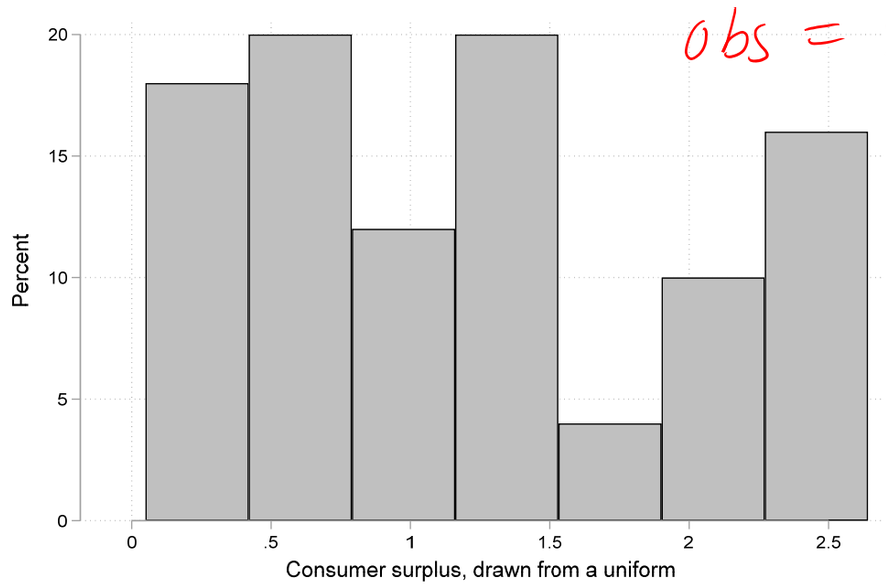


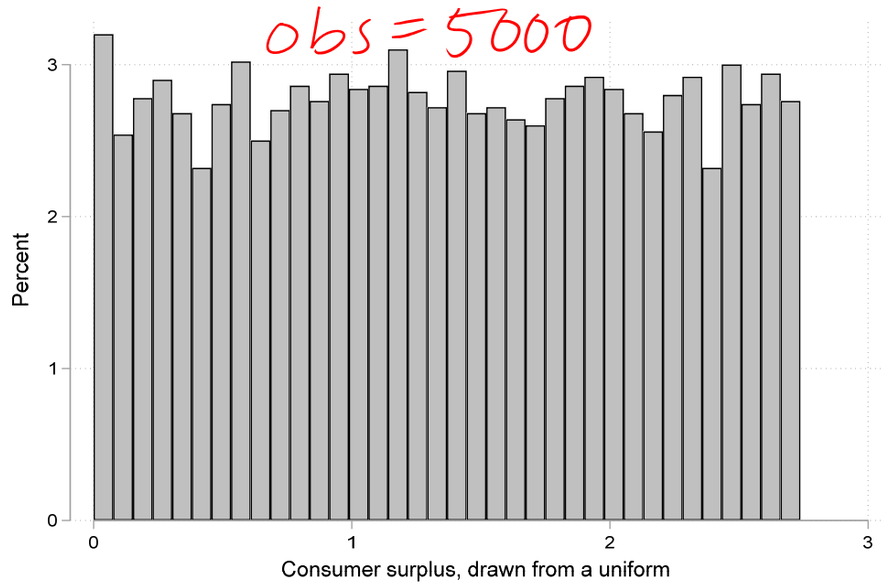
Better!


obs = 50
Percent
0
5
10
15
20
0
.5
1
1.5
2
2.5
Consumer surplus, drawn from a uniform


obs = 500
Percent
0
2
4
6
0
1
2
3
Consumer surplus, drawn from a uniform


obs = 5000
Percent
0
1
2
3
0
1
2
3
Consumer surplus, drawn from a uniform



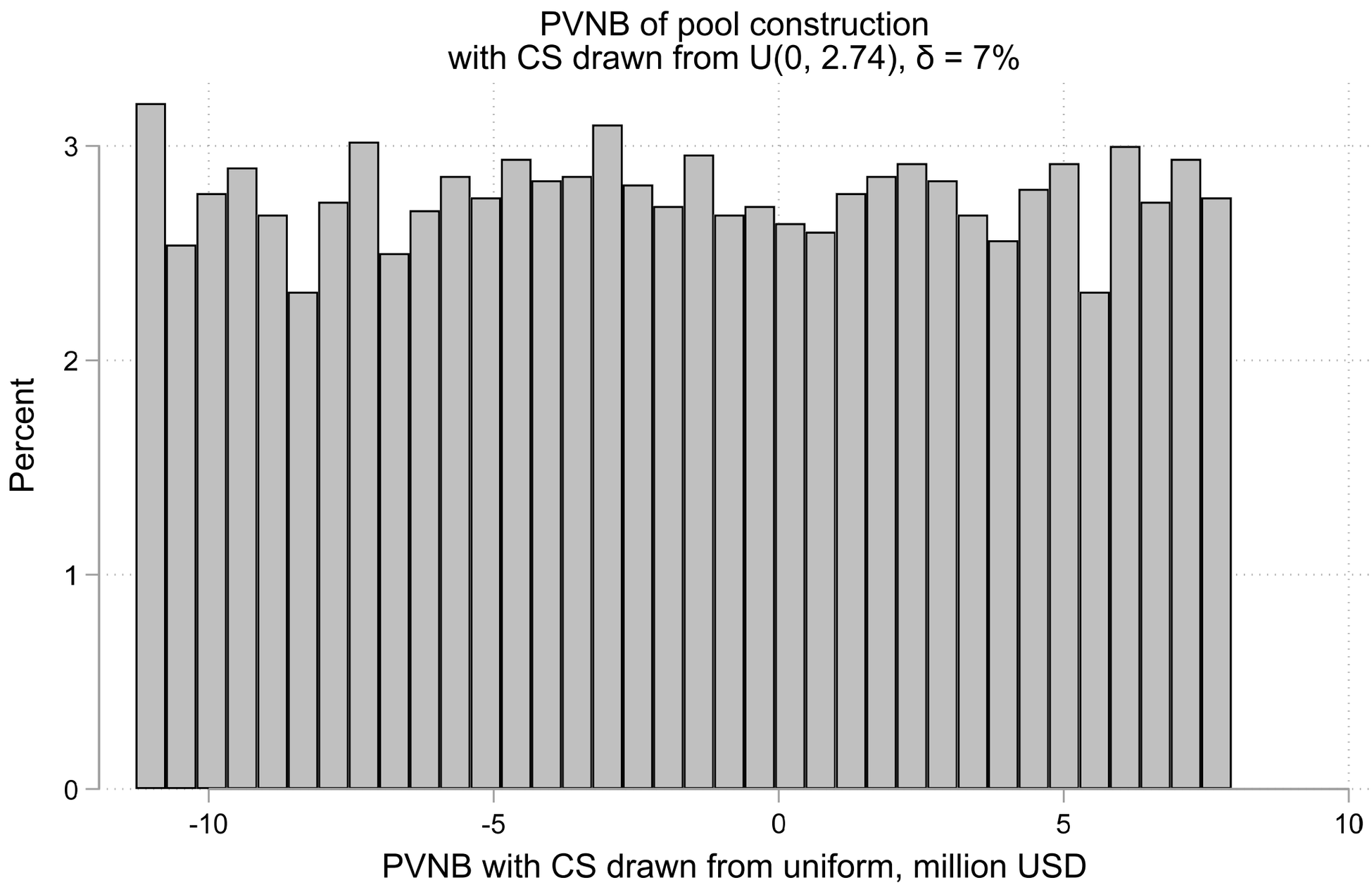


Data Editor (Browse) - [Untitled]

File Edit View Data Tools

var10[1]

	pvnb	pvnb_uniform
1	-1.661581	4.962507
2	-1.661581	-4.806615
3	-1.661581	-9.812078
4	-1.661581	-6.70545
5	-1.661581	-8.177304
6	-1.661581	6.520941
7	-1.661581	-4.080416
8	-1.661581	-7.917433
9	-1.661581	-1.112826
10	-1.661581	2.276593
11	-1.661581	-5.892797
12	-1.661581	-5.844304
13	-1.661581	-9.031607
14	-1.661581	-7.710472
15	-1.661581	6.38758


PVNB of pool construction
with CS drawn from U(0, 2.74), δ = 7%
Percent
0
1
2
3
-10
-5
0
5
10
PVNB with CS drawn from uniform, million USD

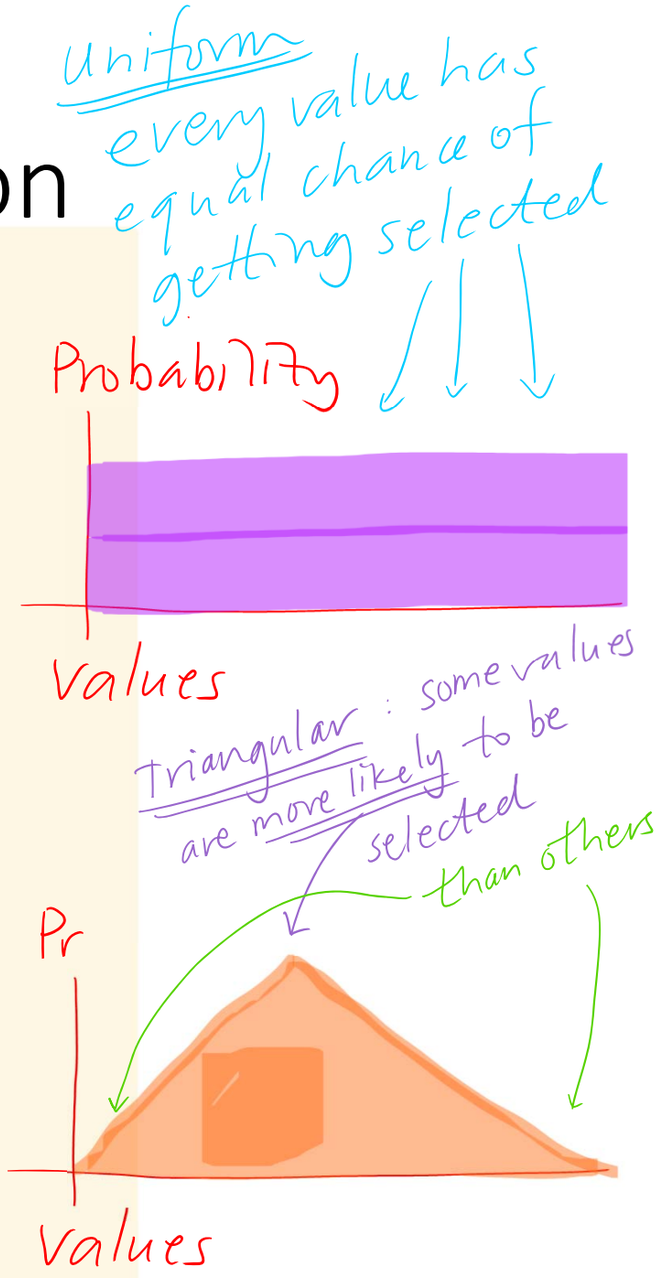
What if uniform seems like a bad idea?

Draw from the triangular distribution

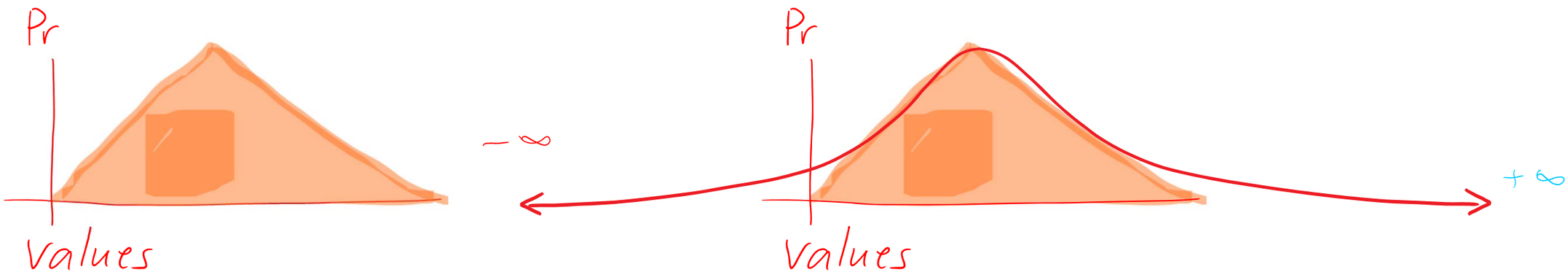
```
* Program to generate triangular distributions *
* 1 = minimum value *
* 2 = mode (peak)   *
* 3 = maximum value *
* 4 = name of triangular variable generated *

quietly: capture program drop Triangular    // (will get error if in memory)
quietly: program define Triangular
local min = `1'
local mode = `2'
local max = `3'
local variable = "`4'"
local cutoff=(`mode'-`min')/(`max'-`min')
generate Tri_temp = uniform()
generate `variable' = `min' + sqrt(Tri_temp*(`mode'-`min')*(`max'-`min')) \\\
if Tri_temp<`cutoff'
replace `variable' = `max' - sqrt((1-Tri_temp)*(`max'-`mode')*(`max'-`min')) \\\
if Tri_temp>=`cutoff'
drop Tri_temp
end
```

Draw from the triangular distribution

```
* Program to generate triangular distributions *
* 1 = minimum value *
* 2 = mode (peak)   *
* 3 = maximum value *
* 4 = name of triangular variable generated *

quietly: capture program drop Triangular    // (will get error if in memory)
quietly: program define Triangular
local min = `1'
local mode = `2'
local max = `3'
local variable = "`4'"
local cutoff=(`mode'-`min')/(`max'-`min')
generate Tri_temp = uniform()
generate `variable' = `min' + sqrt(Tri_temp*(`mode'-`min')*(`max'-`min')) \\\
if Tri_temp<`cutoff'
replace `variable' = `max' - sqrt((1-Tri_temp)*(`max'-`mode')*(`max'-`min')) \\\
if Tri_temp>=`cutoff'
drop Tri_temp
end
```



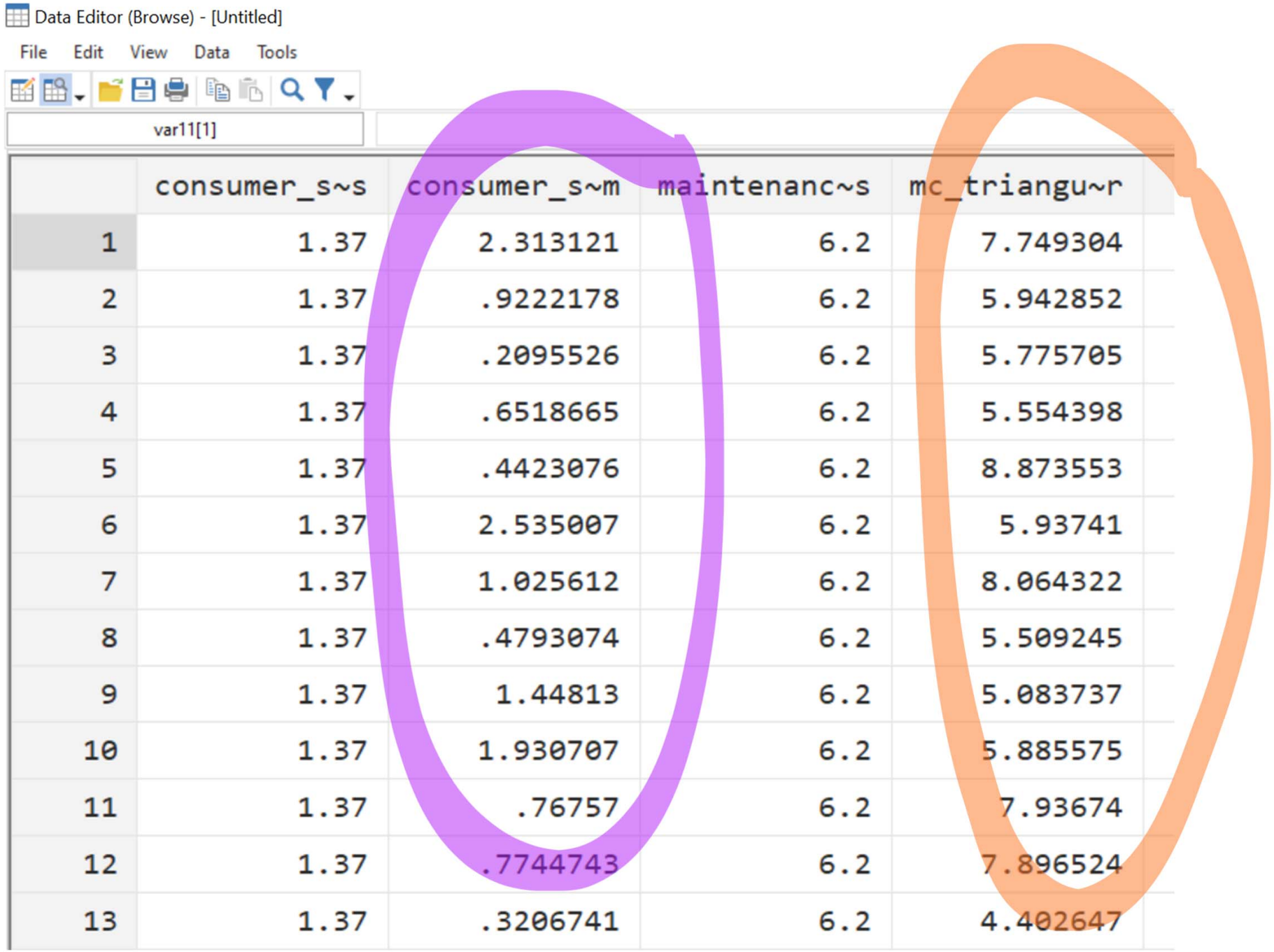
Why not the normal distribution?



Because it never ends!

How to draw from the triangular distribution

```
* Generate variable called mc_triangular with the following features
    * mode: the previous fixed value, i.e. 6.2
    * set min: 6.2 - 3 = 3.2
    * set max: 6.2 + 3 = 9.2

    Triangular 3.2 6.2 9.2 mc_triangular
```



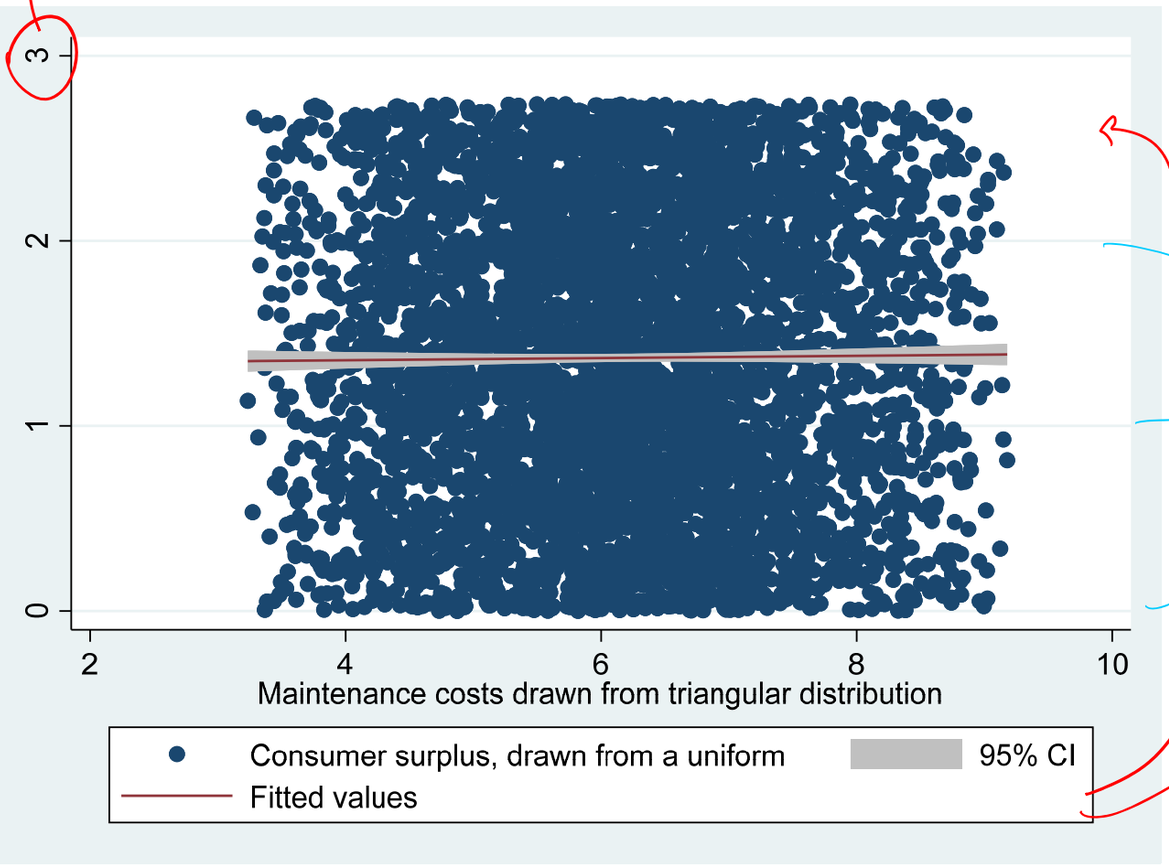
	consumer_s~s	consumer_s~m	maintenanc~s	mc_triangu~r
1	1.37	2.313121	6.2	7.749304
2	1.37	.9222178	6.2	5.942852
3	1.37	.2095526	6.2	5.775705
4	1.37	.6518665	6.2	5.554398
5	1.37	.4423076	6.2	8.873553
6	1.37	2.535007	6.2	5.93741
7	1.37	1.025612	6.2	8.064322
8	1.37	.4793074	6.2	5.509245
9	1.37	1.44813	6.2	5.083737
10	1.37	1.930707	6.2	5.885575
11	1.37	.76757	6.2	7.93674
12	1.37	.7744743	6.2	7.896524
13	1.37	.3206741	6.2	4.402647

From ugly graphs to (at least) ok

(6) numbers make you twist your head

Why make ugly graphs when you don't have to?

Default scheme BAD



(1) Colors are hard to see for colorblind people

(2) Background tinting... ugh, why??

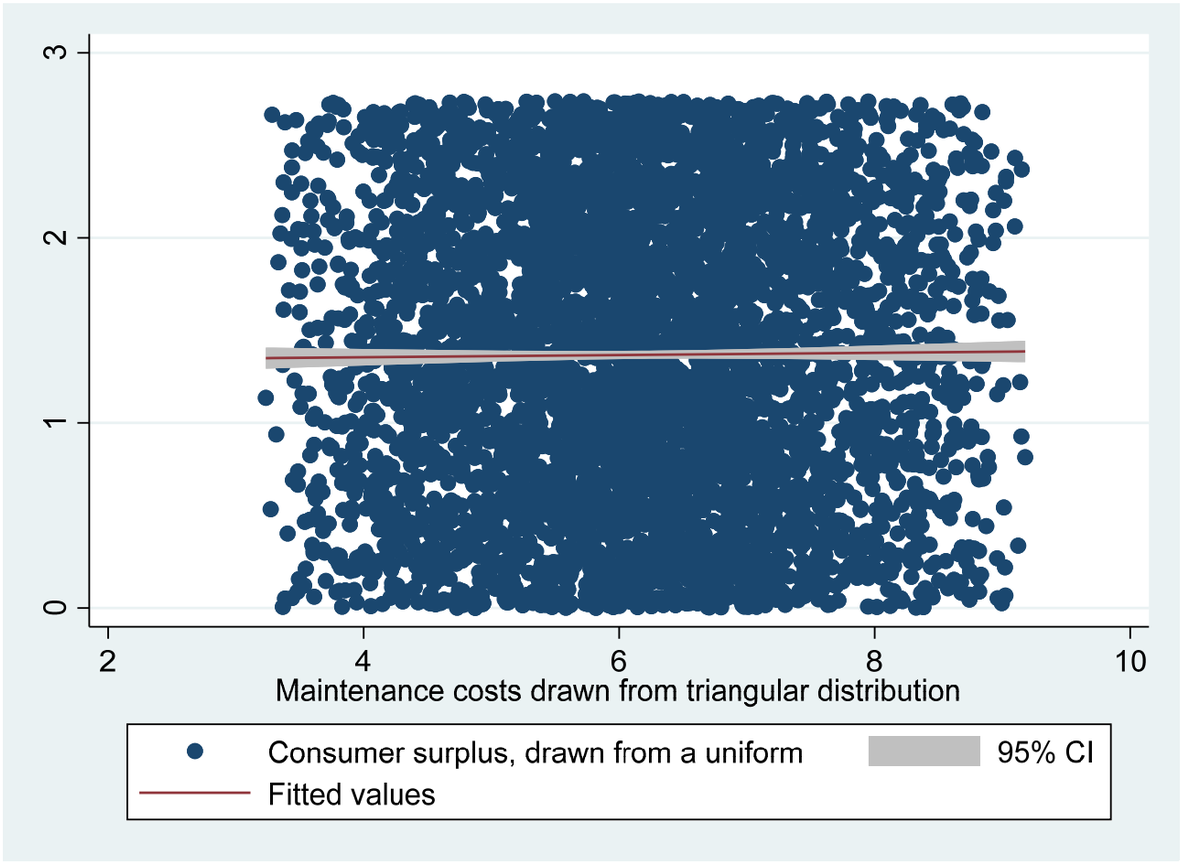
(3) Gridlines only parallel to x-axis

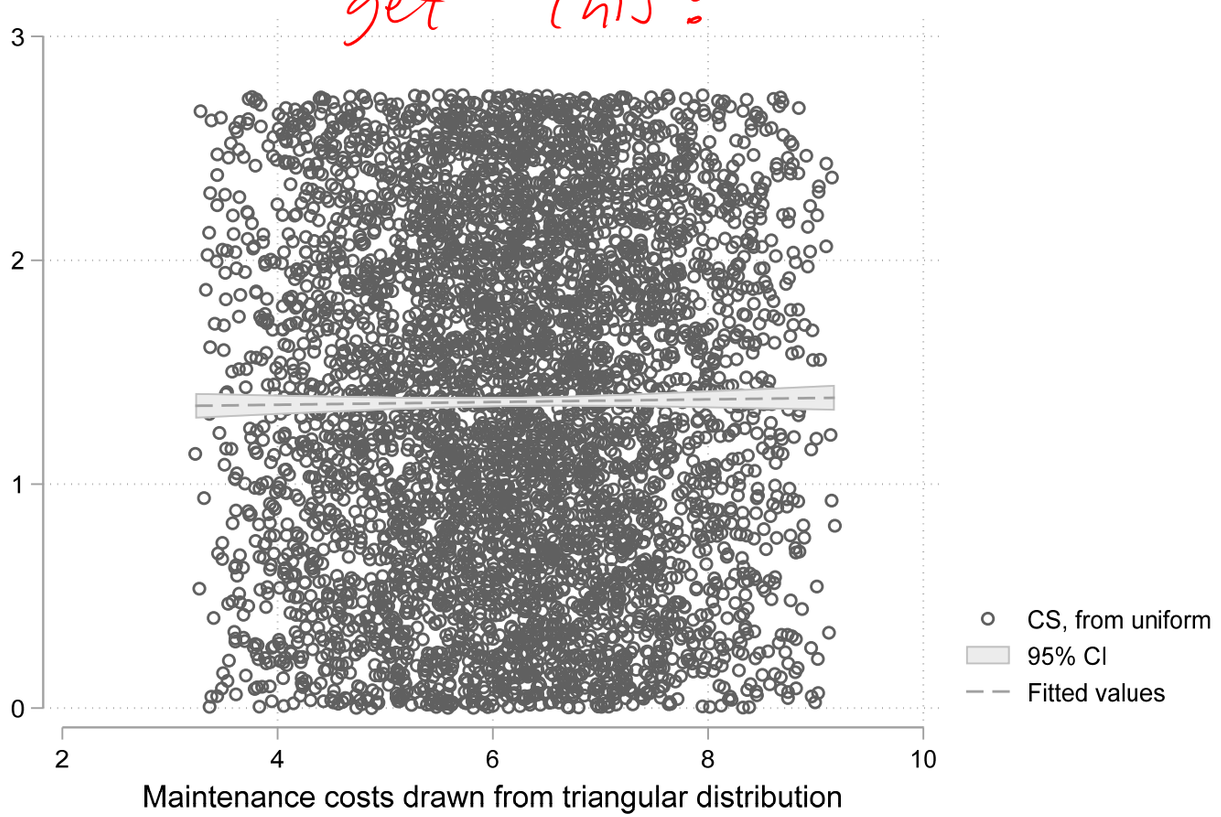
(4) What is this frame good for??

Also, the legend is far from the actual data (5)

Why make ugly graphs when you don't have to?

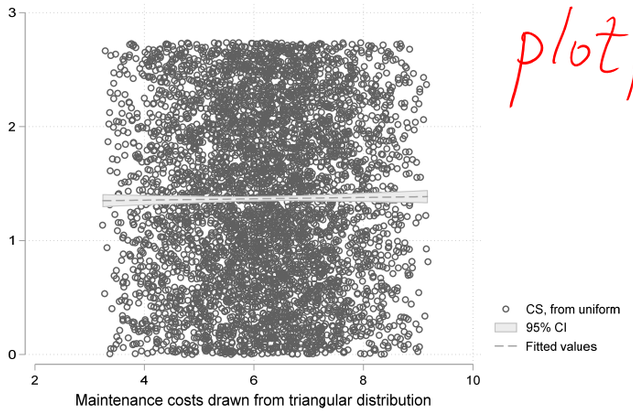
By simply adding , scheme(plotplain) you can get this:



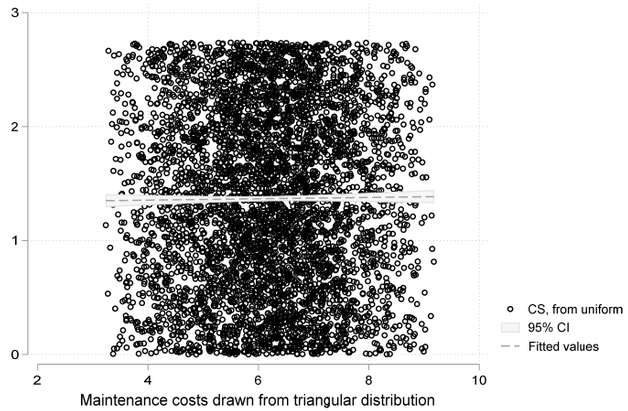


Not perfect but less horrendous

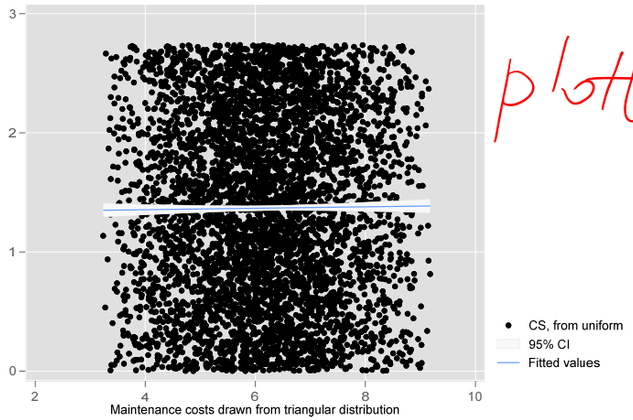
Why make ugly graphs when you don't have to?



plotplain

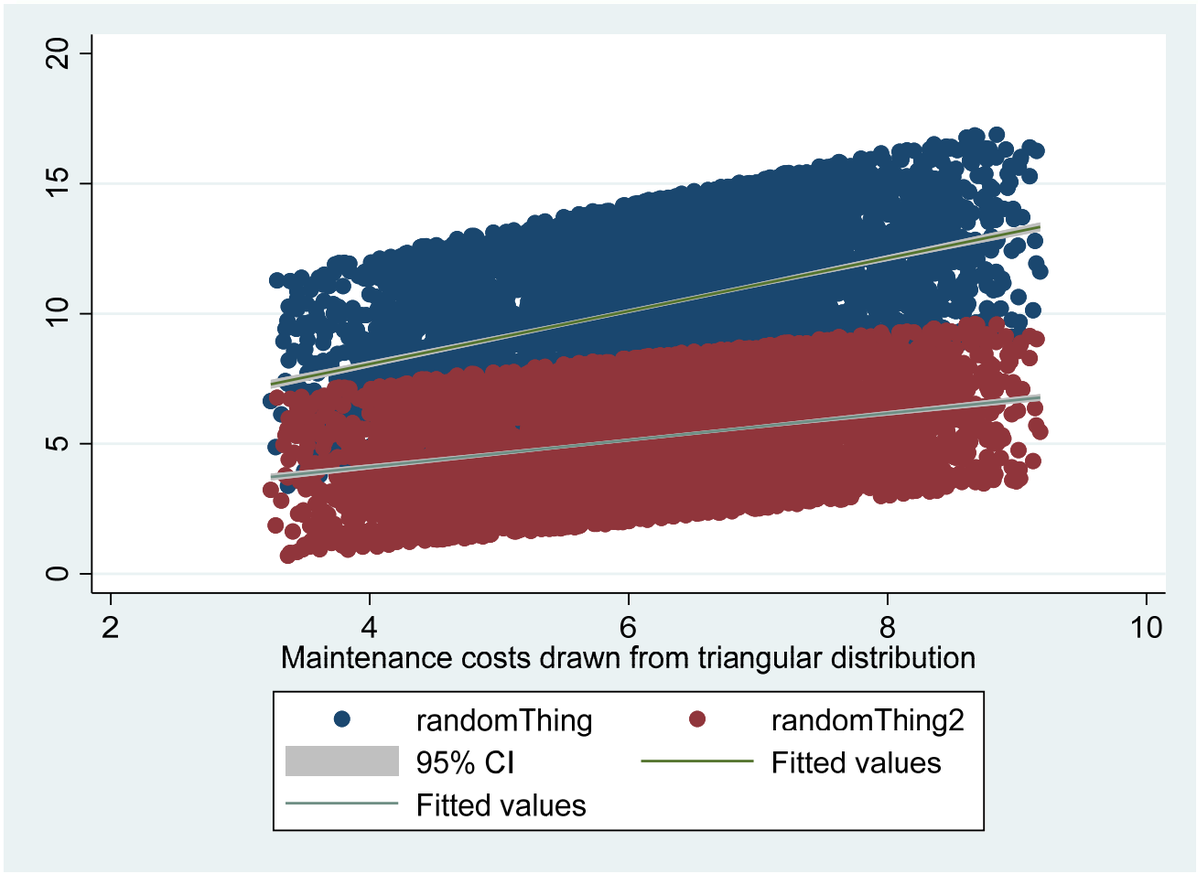


plotplainblind

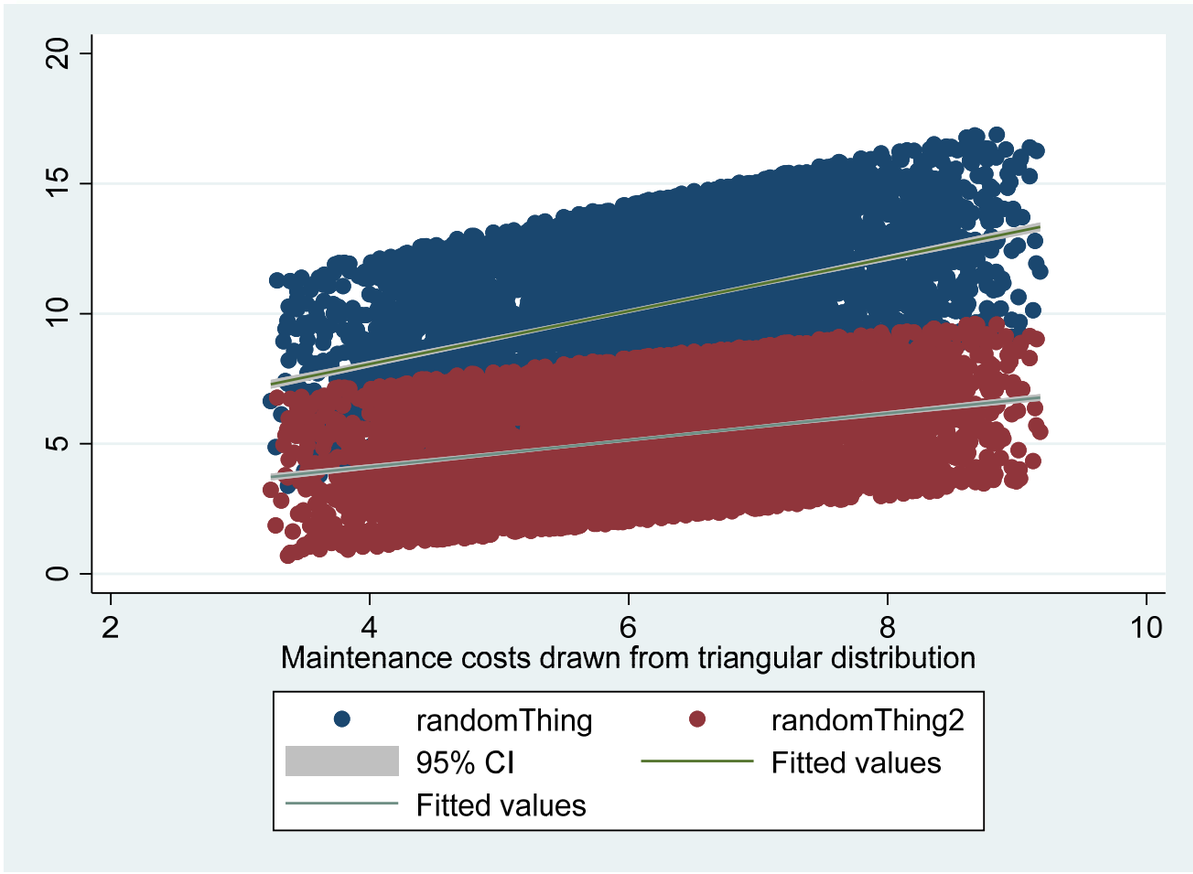


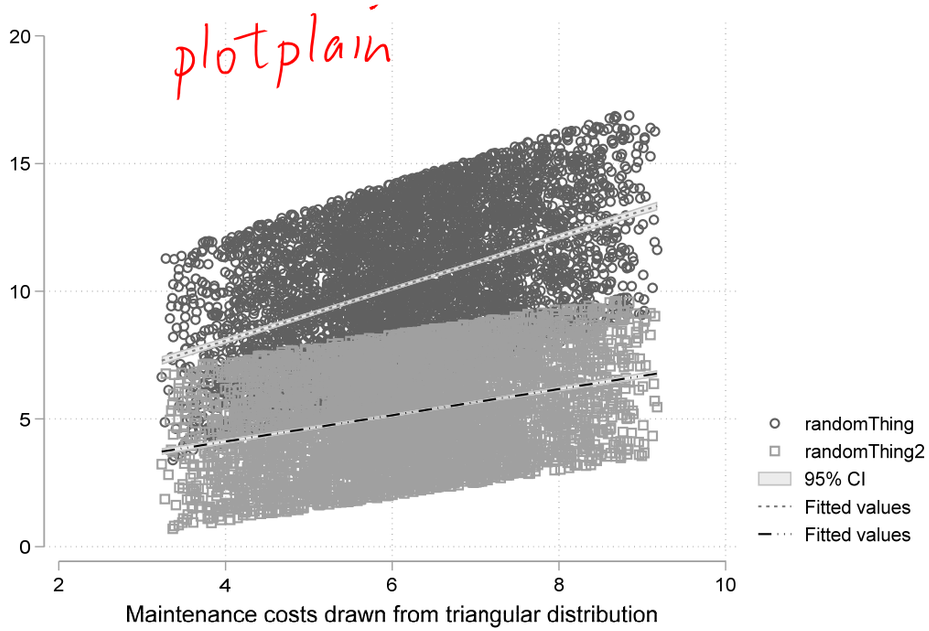
plottig

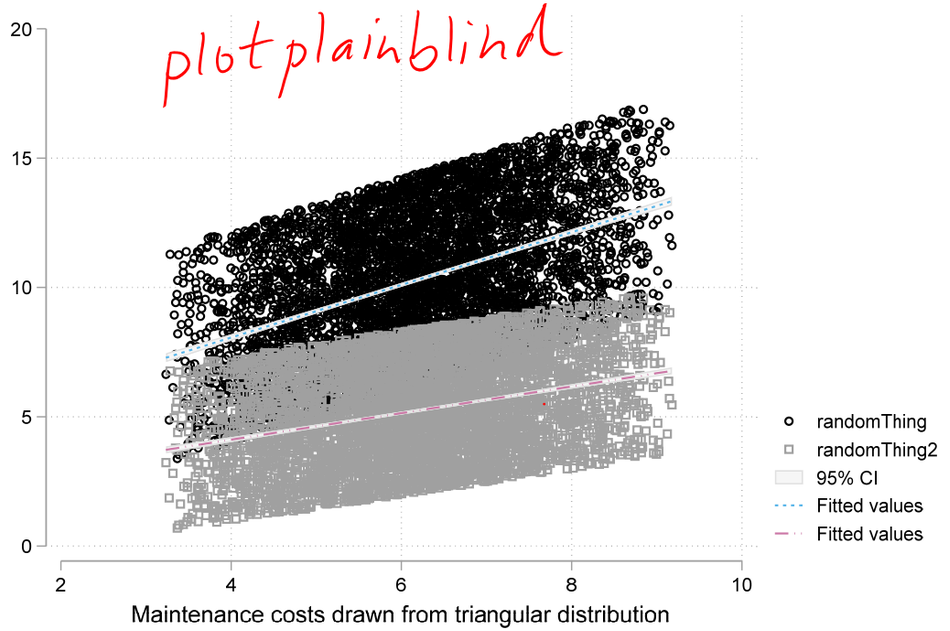
```
* Generate other random thing for graph purposes
    gen randomThing = mc_triangular + 3*cs_uniform
    gen randomThing2 = 0.5*mc_triangular + 2.3*cs_uniform - 1
    twoway (scatter randomThing mc_triangular)  ///
    (scatter randomThing2 mc_triangular)        ///
    (lfitci randomThing mc_triangular)          ///
    (lfitci randomThing2 mc_triangular)
```

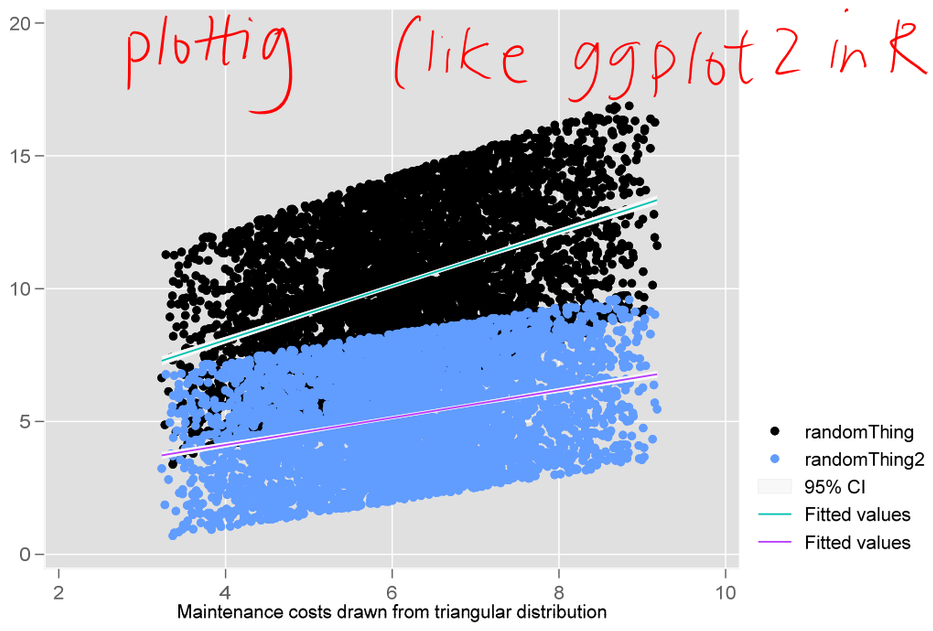


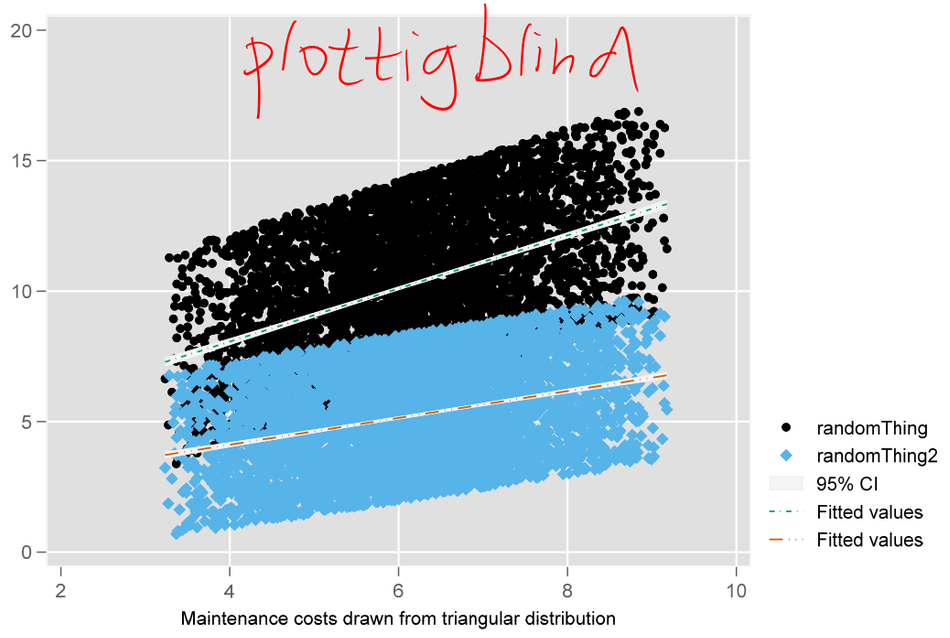
```
* Generate other random thing for graph purposes
    gen randomThing = mc_triangular + 3*cs_uniform
    gen randomThing2 = 0.5*mc_triangular + 2.3*cs_uniform - 1
    twoway (scatter randomThing mc_triangular)  ///
    (scatter randomThing2 mc_triangular)        ///
    (lfitci randomThing mc_triangular)          ///
    (lfitci randomThing2 mc_triangular)
```



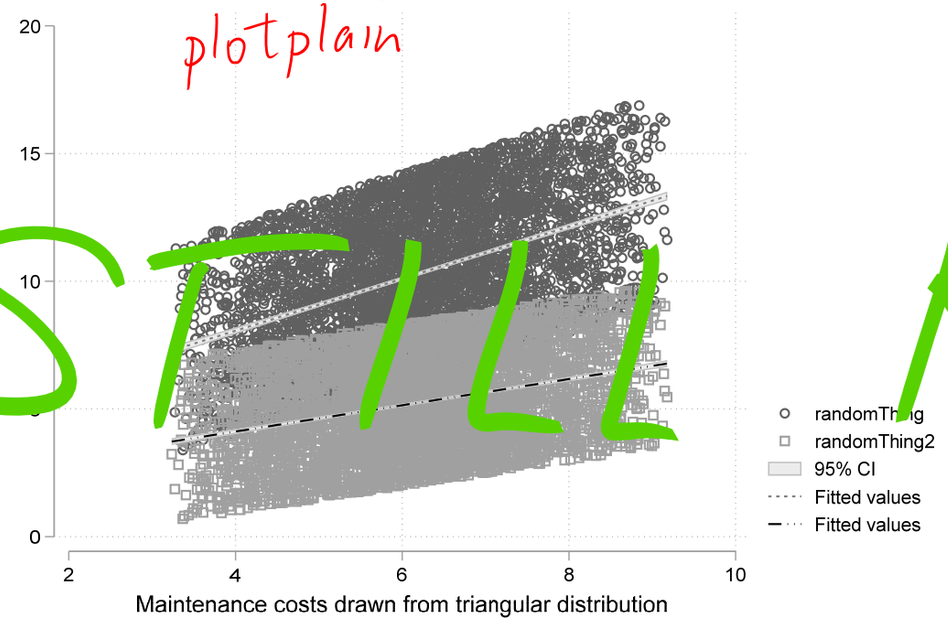

plotplain
20
15
10
5
0
2
4
6
8
10
Maintenance costs drawn from triangular distribution
randomThing
randomThing2
95% CI
Fitted values
Fitted values


plotplainblind
20
15
10
5
0
2
4
6
8
10
Maintenance costs drawn from triangular distribution
randomThing
randomThing2
95% CI
Fitted values
Fitted values

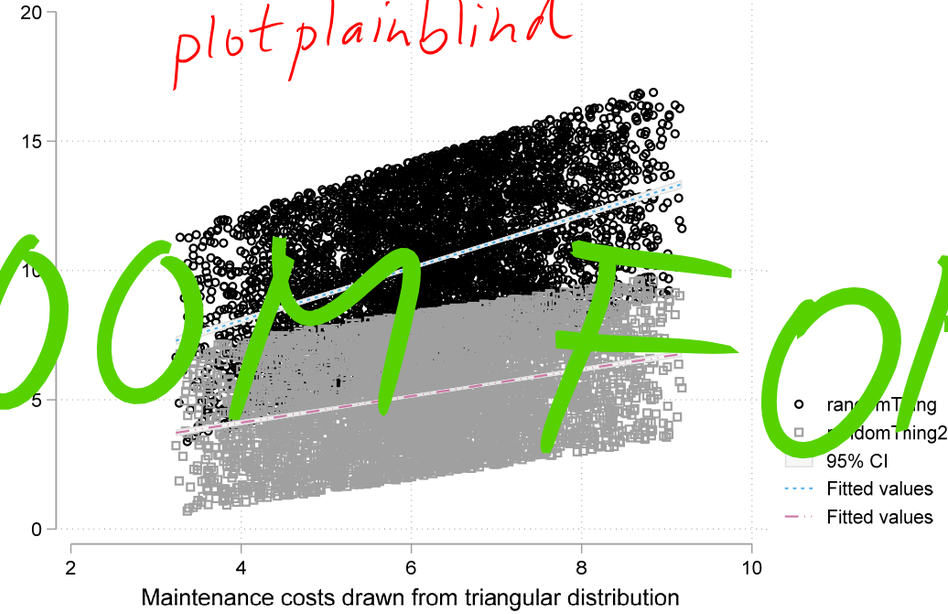

plottig (like ggplot2 in R)
20
15
10
5
0
2
4
6
8
10
Maintenance costs drawn from triangular distribution
randomThing
randomThing2
95% CI
Fitted values
Fitted values


plottigblind
20
15
10
5
0
2
4
6
8
10
Maintenance costs drawn from triangular distribution
randomThing
randomThing2
95% CI
Fitted values
Fitted values

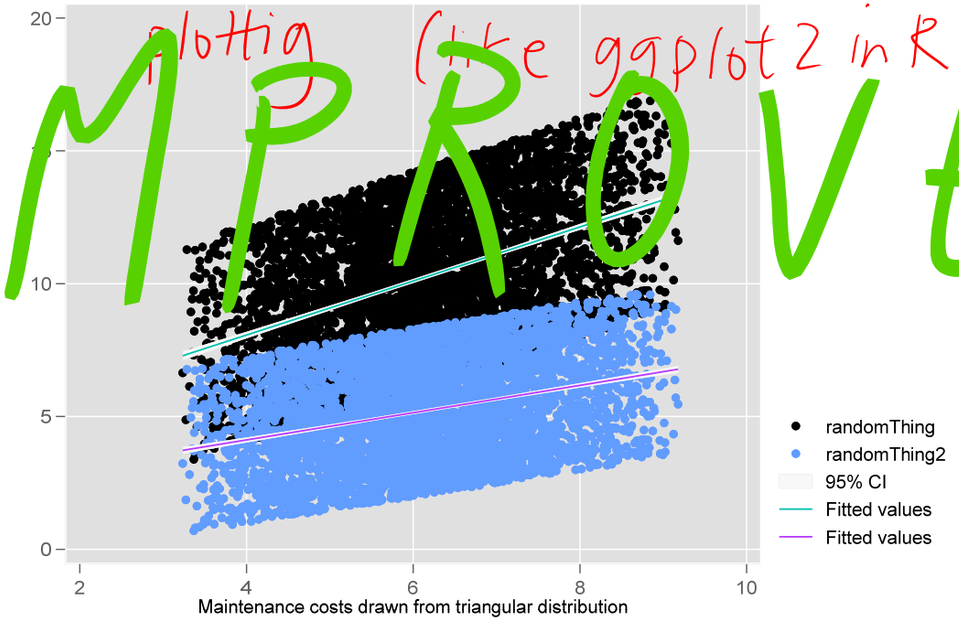
plotplain



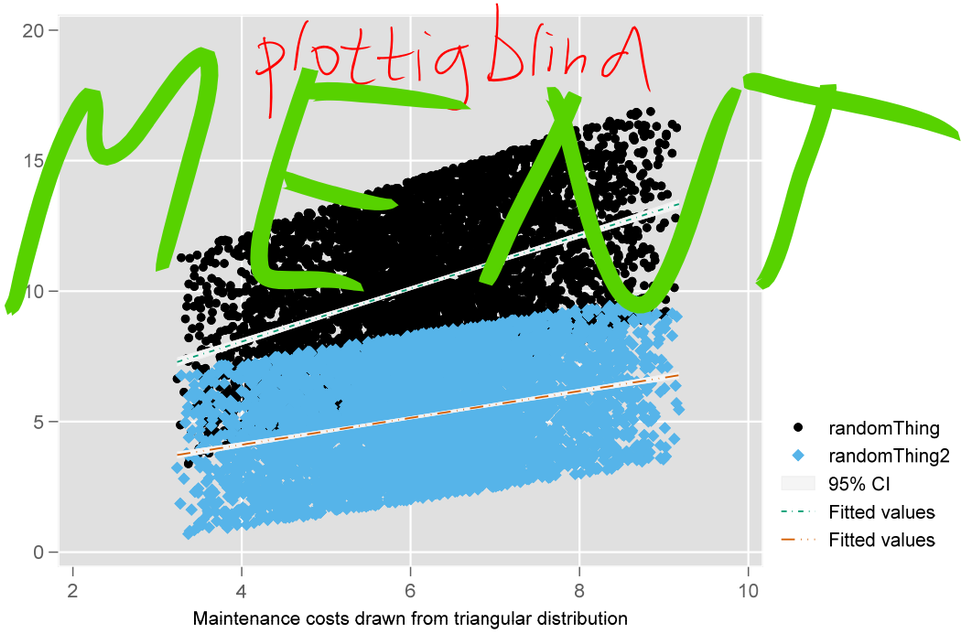
plotplainblind



plottig (like ggplot2 in R)

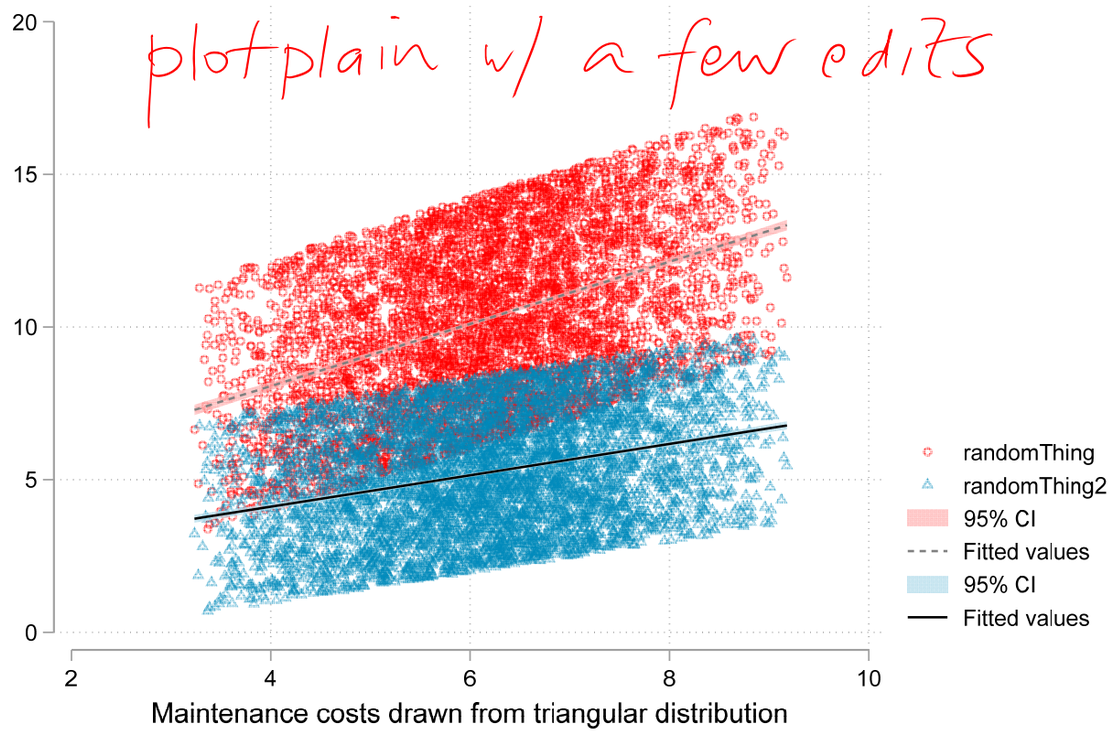


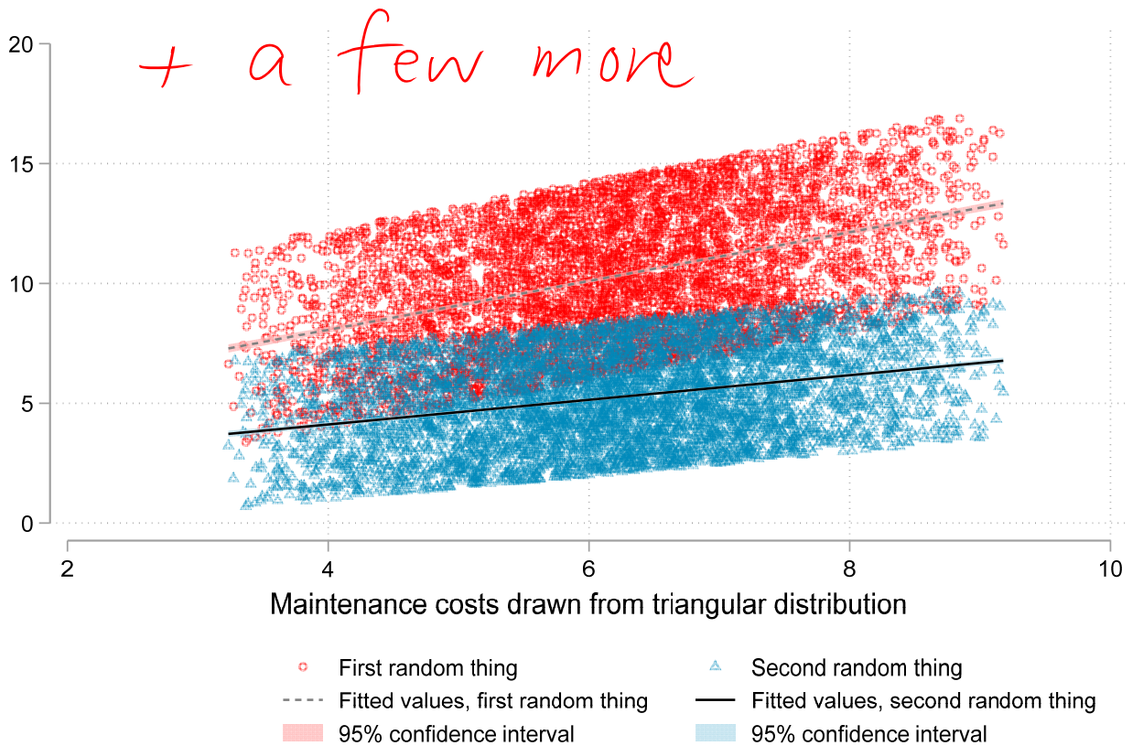
plottigblind



STILL ROOM FOR IMPROVEMENT

Stata has transparency! Use it!





code examples in .do file