

Work after

26 May 2020

Trying to map VFI results to those of parametric expectations.

The VFI sol is $V(x)$, where $x = (k^1, \bar{\pi}, r^n, u)$

So I'm trying to input x^{pe} at the sol of PE.

Ok, that gives me the value function at the sol.

But I want the policy function, $\{i_t\}_{t=1}^T$.

$$V = L_+(x_+, \pi_+) + \beta V$$

$$\text{So } (1-\beta)V = L_+(x_+, \pi_+)$$

1-period loss given optimal policy

Since I know the form of the function L_+ ($\pi_+^2 + 2x_+^2$),
and I know how π_+ and x_+ depend on i_+), I

can solve for i_+ from $(1-\beta)V = L(i)$

$$L = \pi^2 + 2x^2$$

$$= (kx + s_3 f_a + s_4 s_+)^2 + 2x^2$$

$$= \left(k(-\beta i + s_1 f_b + s_2 s) + s_3 f_a + s_4 s \right)^2 + 2x^2$$

$$L = \left(R(-\omega i + S_1 f_b + S_2 s) + S_3 f_a + S_4 s \right)^2 \\ + 2 \times \left(-\omega i + S_1 f_b + S_2 s \right)^2$$

→ I just need to solve this for i

→ The matlab sol ($\min L$ using fminunc)
works way better
than the analytical sol using Mathematica
(maternls31.nb) Beware!

Note that you also need to initialize the
matlab soln at i^{PE} so it finds that one.

Peter meeting

26 May 2020

- s_{t+1} : the right thing to do: 6-dim state-vector b/c s_{t+1} are separate states
→ which would allow for serial corr in the shadows.
- x^{PE} in $V(x^{PE})$ on RHS in (3)
→ $V(x_t^{PE}) - \beta V(x_{t+1}^{PE}) = L$
- Check whether a finer grid makes a diff w/o estimation, a small grid is fine. Then moment you need quantitative exercises, then you need a larger grid.

$$V(X_t^{\text{pe}}) = L(i) + \beta E V(X_{t+1}^{\text{pe}})$$

If they are iid, then Peter is right to say
that $E X_{t+1}^{\text{pe}} = X_t^{\text{pe}}$. To encompass

The case of AR(1), we can write

$$E X_{t+1}^{\text{pe}} = h_x X_t^{\text{pe}}$$

but that's not cool for the endog. states in X .

Now to compute $E(k_{t+1}^{-1})$ and $E(\bar{\pi}_{t+1})$?

$$\begin{aligned} E_t(k_{t+1}^{-1}) &= E_t[k_t k_t^{-1} + \gamma f e_{t+1|t}] \\ &= \rho_k k_t^{-1} + \gamma E_t[(\bar{\pi}_{t+1} - (\bar{\pi}_t + b_1 s_t))^2] \end{aligned}$$

$$\begin{aligned} E_t(\bar{\pi}_{t+1}) &= E_t[\bar{\pi}_t + k_{t+1}^{-1} f e_{t+1|t}] \\ &= \bar{\pi}_t + E_t[k_{t+1}^{-1} f e_{t+1|t}] \end{aligned}$$

→ It all seems to hinge on $E_t[f e_{t+1|t}]$

$$= E_t[\bar{\pi}_{t+1} - \bar{\pi}_t - b_1 s_t]$$

$$= E_t[\bar{\pi}_{t+1}] - \bar{\pi}_t - b_1 s_t$$

↑ $E[\text{alarm}](1,1)$

You know what better, then:

27 May 2020

$$L = \pi^2 + \lambda_x x^2 = fas$$

$$= (kx + s_3 fa + s_1 s_+)^2 + \lambda_x x^2$$

$$= (k(-\beta i + s_1 f_b + s_2 s) + s_3 fa + s_1 s) + \lambda_x x^2 \\ = fbs$$

$$\Rightarrow L = (k(-\beta i + fbs) + fas)^2 + \lambda_x (-\beta i + fbs)^2$$

$$= k^2(-\beta i + fbs)^2 + fas^2 + 2k(-\beta i + fbs)fas$$

$$+ \lambda_x (-\beta i + fbs)^2$$

$$= (k^2 + \lambda_x)(-\beta i + fbs)^2 + 2k \cdot fas(-\beta i + fbs) + fas^2$$

$$= (k^2 + \lambda_x)(\beta^2 i^2 - 2\beta fbs i + fbs^2) - 2k \cdot fas \cdot i + 2k \cdot fas \cdot fbs$$

$$+ fas^2$$

$$L = (k^2 + \lambda_x)\beta^2 i^2 - (2\beta(k^2 + \lambda_x)fbs + 2k \cdot fas)i$$

$$- 2kfasfbs + fas^2 - (k^2 + \lambda_x)fbs^2$$

$$(k^2 + \lambda_x)\beta^2 i^2 - (2\beta(k^2 + \lambda_x)fbs + 2k \cdot fas)i$$

$$- 2kfasfbs + fas^2 - (k^2 + \lambda_x)fbs^2 - L = 0$$

✓

✓

✓

✓

✓

$$(k^2 + \lambda_x) b^2 i^2 - (23(k^2 + \lambda_x) f_{bs} s + 23k \cdot f_{as}) i$$

$$+ 2k f_{as} f_{bs} s + f_{as}^2 - (k^2 + \lambda_x) f_{bs}^2 - L = 0$$

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$i_{1,2}$

$$= \frac{(23(k^2 + \lambda_x) f_{bs} s + 23k \cdot f_{as}) \pm \Delta}{2(k^2 + \lambda_x) b^2}$$

$$\Delta = (23(k^2 + \lambda_x) f_{bs} s + 23k \cdot f_{as})^2 - 4(k^2 + \lambda_x) b^2 (\epsilon)$$

$$= (23)^2 ((k^2 + \lambda_x) f_{bs} s + k \cdot f_{as})^2 - 4b^2 (k^2 + \lambda_x) c$$

$$= 4b^2 [((k^2 + \lambda_x) f_{bs} s + k \cdot f_{as})^2 - (k^2 + \lambda_x) c]$$

$$i_{1,2} = \frac{23[(k^2 + \lambda_x) f_{bs} s + k \cdot f_{as}] \pm 23\sqrt{\Delta}}{2(k^2 + \lambda_x) b^2}$$

$$= \frac{2(k^2 + \lambda_x) f_{bs} s + k \cdot f_{as} \pm 23\sqrt{\Delta}}{(k^2 + \lambda_x) b^2}$$

✓

$$\Delta = ((k^2 + \lambda_x) f_{bs} s + k \cdot f_{as})^2 - (k^2 + \lambda_x) [2k f_{as} f_{bs} s + f_{as}^2 - (k^2 + \lambda_x) f_{bs}^2 - L]$$

$$\cancel{(k^2 + \lambda_x)^2 f_{bs}^2 s^2} + \cancel{k^2 f_{as}^2 s^2} + \cancel{2k(k^2 + \lambda_x) f_{as} f_{bs}}$$

$$- \cancel{(k^2 + \lambda_x) 2k f_{as} f_{bs}} - (k^2 + \lambda_x) f_{as}^2 - \cancel{(k^2 + \lambda_x)^2 f_{bs}^2} + (k^2 + \lambda_x) L$$

$$\Delta = b^2(k^2 - \lambda_x)L - \lambda_x fas^2 b^2$$

So actually

$$i_{1,2} = \frac{(k^2 + \lambda_x) fbs + k fas \pm \sqrt{(k^2 + \lambda_x)L - \lambda_x fas^2}}{(k^2 + \lambda_x)b}$$

Ok so what have I learned:

1. Since I know there are 2 sols, doing it analytically makes more sense than w/ fsolve.
2. Depending on the values of L, fa, fb, s, some i may be complex, in which case all i show up as complex, some w/ 0 imaginary part. Then one needs a good way to discriminate between the two sols, for every t!

Brian Dombeck macro lunch

Shadow price learning & Expectationally Driven BCs
main diff b/w econ & nat. science: expectations
→ mice!

Evans & McGough : shadow price learning
"SP learning"

SP learning: agents learn about shadow prices
→ using an RLS scheme (so it's a special
case of adaptive learning)

- ↳ ask for slides!

UWash: ppl have been saying $\pi > 2\%$ for a long time! → hard to reconcile w/ RE.

- If new shocks can be incorporated into the info set of agents, then maybe one can drive agents into responding to promises of the gov.

Is "news shock" a proxy for adaptive learning?
Eusepi & Preston AER says that adaptive learning
looks a lot like news-shock-model.

- Nide Bloom "TV volatility is exog"

- news & noise are equiv under RE
→ not obvious if they are outside RE

Brian Pinesky may should be equiv
outside RE b/c they are exog processes.

News shock in RE : can only generate comovement
if $\beta < 0.25$ instead of 1.

- SP-learning emphasizes the distinction b/w
market clearing conditions vs. model egs.
→ so it seems like an improvement over BE learning

SP-learners are less sophisticated than
EE-adaptive-learners.

- Is it really an RPE?

What's a state shadow price?

value to me today of an additional unit
of a state

Ryan: \rightarrow 2 multipliers on intertemporal constraints
(multipliers on static constraints don't
matter b/c you can sub 'em out)

We let them regen on MSV sol.

We simulated $T=500$.

- job market materials?
willing to read?

- These models are usually not treated in their non-linear form!

Ryan meeting

27 May 2020

Can approx the policy fct in the same way b/c you have generated values too; at every point in the state space.

Trick to save time: convex comb of policy iter $\xrightarrow{\text{VFI}}$ faster than VFI but more stable than PI.
Update policy fct every 5th iter.

Keep the policy the same for some iterations

\hookrightarrow saves you the max step.

The distinguishing elements of global methods

- gridpoints

- thing you're applying

- how you find zeros

Value function is also a projection method, but in any case, policy iteration is a projection b/c it works directly w/ FOCs.

Brian meeting

27 Aug 2020

will send job market materials.

ETI 2001: misspecified models & RPE

Chap. 13

Evans & McGough Shadow Price Learning Paper

↳ sends!

End of it is interesting! NK!

fwd guidance as a news shock

Offshoot: currency crisis models un treffen

targeting underpinning it

everything is fine, boom!

→ exog shift in $E(\cdot)$

Schmitt-Grohé & Uribe OpenEcon Textbook
pdf online!

Let's talk periodically! Figure out over mail!

Ok so mixed news:

29 May 2020

- 1) $VFI = PEA \dots$
- 2) ... only if you approx the policy function using a sufficiently large $\bar{\pi}$ -grid: $\bar{\pi} \in [-0.2, 0.2]$ is clearly too small! $\bar{\pi} \in [-4, 4]$ is the smallest that seems approximately ok. The problem is that VFI w/ $\bar{\pi} \in [-10, 10]$ started diverging at one point. But I suppose I should use the same grid for VFI as for i -approx...?
Or not?

Saying that they should have the same grid is to say that we're approximating the value function on the same grid as the policy fct.

I don't know. The good news is that it's not the acceleration that screws up the results. It's the $\bar{\pi}$ -grid. My concern is that the grids should be the same b/c we use i_t -values generated

from a grid X_1 , so $i_t = \hat{V}(X_1)$
and then the coefficients that approximate
 $\hat{i}(\alpha, i_t(X_1), X_2)$, shouldn't be the products of
a different grid.

Let's see if the $\bar{\pi} = [-4, 4]$ grid thus works for
VFI.

Let's try to figure out why my value function
sometimes diverges (it has a kind of spectacular
reversal). In theory (and it looks like, in practice too)
VFF is guaranteed to converge, albeit slowly.
So I'm wondering if there's something about the
learning that can screw things up: for a large
value of $|\bar{\pi}|$, some unpermitted policy is chosen
that causes say a $k_t^{-1} < 0$ or stg. Maybe
it has to do w/ a potential mismatch (or lack thereof)
specification of the problem constraints? But if that's
so then only doing the i -update every j^{th} iter

may actually be helpful b/c you don't do the max step as many times.

$$\text{But } k_{t+1}^{-1} = \rho_k k_{t+1}^{-1} + \gamma f_{t+1}^2$$

definitely keeps $k_t > 0$. What it can do though is that suddenly f_t can become huge: in fact: the bigger & more coarse the grid, the more likely that is to happen b/c $\bar{\alpha}$ changes a lot from one iteration to the next.

On the other hand, since we're on a grid, the result of every iteration is $V(X)$ where X is all combination of all states. Divergence can only happen if somewhere along X ,

$V^{t+1}(X)$ is very different from $V^t(X)$

Can it be that k_t reaches 0? No b/c it's safely constrained by the grid. The only thing that can happen is that $V(X_{t+1})$ is such that an i is chosen which gives a very diff b_i and thus TV .

Ha! what I'm finding from this $\bar{\pi} = [-4, 4]$ VFI is that now I need to use a much bigger grid for \hat{i} (e.g. $\bar{\pi} \in [-40, 40]$ is too small, $\bar{\pi} \in [-100, 100]$ is fine)

for it to have the same shape as PEA.

Let's make the $\bar{\pi}$ grid start at 0.0001 instead of 0.
↳ doesn't change it

Obs:

- The bigger $\bar{\pi}$ grid, the bigger are the changes in V initially. Why? B/c more "data variation?"
- For $\bar{\pi} \in [-4, 4]$, the i^* are huge: $\in [-300, 300]$

I'm wondering whether this is what makes it desirable to scale up the $\bar{\pi}$ grid afterwards for \hat{i} . \rightarrow seems to suggest that VFI suffers from a scaling issue somewhere, especially as the problem is nfd, just the scale off.

Ok so I've set up command-vfi & command-pea
to investigate the two methods; compare-valve-pea-results
compares the two. Now at this point I'm really
wondering if the PEA is wrong.

→ Combine them!

- Initialize PEA at different rand
→ gives the same
- Try $1e-10$ as eps instead of $1e-6$ in PEA
→ gives the same, took 20 min.
- Now input a different sequence of X and see
if the scaling issues remain the same
→ it's diff of course but in the grand scheme of things
it's similar in that
 - i) it has big up/down oscillations initially
 - ii) later in the sample it moves around 0 in the
 $\pm 5\%$ band.

⇒ the scaling issue in VFF seems to be there, no actually,

30 May 2020

no; the $\tilde{\pi}$ -grid used for VFI was $[-4, 4]$,
and loading it for the \hat{i} -approx, I get the
same thing.

value-outputs-server32-accelerated

per-outputs - 30-May-2020-10-18-28

Now try this w/ value-outputs -

w/ a $[-4, 4]$ grid!

→ it works, holy smokes! I must have
used non-compatible grids. Yeah!

To do for First Draft (=Draft4)

31 May 2020

- ✓ • Rewrite the analysis part fundamentally
- ✓ • Once you have that, rewrite the intro in the spirit
of Adam
- ✓ • Once you have that, rewrite the abstract.

- Analysis

- ✓ - Target criterion

- ✓ - Implementing: optimal int. rate sequence vs. a std Taylor rule.

- ✓ • policy fit as a function of X :

- function of $\bar{\pi}$, of b_t^*

- ✗ • reaction fit not observable, π & X ?

- relation to TR

- numbers to the opt. policy response!

- ✓ • Corr b/w b & π and response of i to u and π

$$0.0008981 \times 10^{-4} = 0.0008981$$

- ✓ • App PEA & VFI

- ✓ • Conclusion

- ✓ Intro: Related lit ← 2 June start here!

- ✓ • Go over references

- ✓ • Go over App. ←

- ✓ • visual identity for figures ← working on it

- ✓ • Bigger grid search

For after:

- Basu's big. premium correction
- TR w/ smooth crit \rightarrow maybe once 1st is done.
- TV TR
- more relation of policy function to TR
- estimation

A first read on tablet. Notes:

2 June 2020

Nope.

Ryan meeting

5 June 2020

- interpretation of $L(\cdot)$: concave/convex, slope need derivative, so you only need to plot it.
- point of TR: a simple, verifiable description of mon. pol. \Rightarrow a TV-TR has all the complications of a TL, so no need to do.
 \Rightarrow i'm at an inflection point in my work. Ryan's prior was: write up things, and then estimate, but

now it seems like the prob (estimation changes the results & the pitch of the paper) is fairly high. Therefore:

→ spend up to a month estimating
if in 2 weeks I see that I won't get any quick results from estimating, then I can still revert to sending this draft.

NB. "I need 2 weeks of lead time." in the contract means "expect to have 2 weeks until he gives you feedback", not "give him a heads-up 2 weeks before submitting".

Work after Today is June 5.

+ 2 weeks is June 19.

+ 4 weeks is July 3.

Work after

Ok so the draft is done for now. The only thing it will need to do is incorporate Ryan's idea of why a time-invariant TR exercise makes sense.

Don't apologize: instead, say that the whole point of a TR is a simple, easily computable and verifiable implementation of policy.

As for the estimation, I return to the existing code

command GMM_andongy-function.m, but replace the objective function.

- Need to redo bootstrap as a VAR
- Need to redo objective function

↳ incorporating a shape-preserving spline

→ Actually, I decided to leave that old file be. Instead, I moved stuff to materials33.m

1) First step is to make the

7 June 2020

bootstrap and the weighting matrix work

2) Then the GMM will have EXACTLY the same structure as before, except $L\Omega_{\alpha}$ will be replaced by $B \cdot s(x)$ where B are the coeffs and $s(x)$ is the basis (hopefully a shape-preserving spline or simply a piecewise-linear approx).

3) \Rightarrow shape-preserving spline.

1) I'm scrapping bootstrap resample.m b/c it's useless: it's just resampling the errors w/o building a new (VAR) sample. Instead, I'm using the dd code data_boot.m from the IT-project.

So I have dataset-boot, which is the $nboot \times T$ bootstrap-resampled VAR dataset.

\rightarrow Now what I need is a procedure to estimate the ACF of a VAR. Hamilton, p. 280 Mac seems

to provide such an approach.

The j^{th} autocovariance Γ_j of the original process y_t is the first n columns and n rows of

$$\Sigma_j = F \Sigma_{j-1}, \quad j=1,2,\dots \quad [10.2.20]$$

$= F^j \Sigma$

which is

$$\Gamma_j = \Phi_1 \Gamma_{j-1} + \Phi_2 \Gamma_{j-2} + \dots + \Phi_p \Gamma_{j-p}$$

for $j = p, p+1, p+2, \dots \quad [10.2.22]$

The VAR - notation Hamilton uses is

$$y_t = c + \Phi_1 y_{t-1} + \Phi_2 y_{t-2} + \dots + \Phi_p y_{t-p} + \varepsilon_t$$

VAR(p) $[10.1.4]$

Reunite this in terms of deviations from the mean as

$$\mu = (I_n - \Phi_1 - \Phi_2 - \dots - \Phi_p)^{-1} c$$

$$(y_t - \mu) = \Phi_1(y_{t-1} - \mu) + \Phi_2(y_{t-2} - \mu) + \dots + \Phi_p(y_{t-p} - \mu) + \xi_t$$

Use that to reunite as $Q = \text{VC matrix of } v_t$

$$\xi_t = F \xi_{t-1} + v_t \quad \text{VAR}(1) \quad [10.1.11]$$

where $\xi_t = \begin{bmatrix} y_t - \mu \\ y_{t-1} - \mu \\ \vdots \\ y_1 - \mu \end{bmatrix}$ (de-meaned y_t) $[10.1.5]$

The j^{th} Autocovariance Matrix

$$\begin{aligned} \Gamma_j &:= E[(y_t - \mu)(y_{t+j} - \mu)'] \quad [10.2.1] \\ &= E[\xi_t \xi_{t+j}'] \end{aligned}$$

Then $\Sigma := E[\xi_t \xi_t']$

Note that

$$\Sigma = \begin{bmatrix} \Gamma_0 & \Gamma_1 & \dots & \Gamma_{p-1} \\ \Gamma_1' & \Gamma_0 & \dots & \Gamma_{p-2} \\ \vdots & \ddots & \ddots & \vdots \\ \Gamma_{p-1}' & \Gamma_{p-2}' & \dots & \Gamma_0 \end{bmatrix} \quad [10.2.12]$$

Ah I see. I need to rewrite my play VAR(p) as a VAR(1).

Recall from [10.1.11] that

$$\begin{aligned} \xi_t &= F \xi_{t-1} + v_t. \quad \text{Thus } \Sigma = E[\xi_t \xi_t'] \\ &= E[(F \xi_{t-1} + v_t)(F \xi_{t-1} + v_t)'] \\ &= E[(F \xi_{t-1} + v_t)(\xi_{t-1}' F' + v_t')] \\ &= E[F \xi_{t-1} \xi_{t-1}' F' + v_t v_t' + \text{cross terms}] \\ &= F E[\xi_{t-1} \xi_{t-1}'] F' + E[v_t v_t'] \end{aligned}$$

$$\Sigma = F \Sigma F' + Q$$

[10.2.13]

This can be solved for Σ using vec as

$$\text{vec}(\Sigma) = [I_{r^2} - (F \otimes F)]^{-1} \text{vec}(Q)$$

$r^2 \times r^2$

Then, having Σ , we find the ACF_j (Γ_j)

by computing first Σ_j , the ACF of the VAR(1).

$$\Sigma_j = F^j \Sigma .$$

[10.2.21]

Finally, we obtain the ACF of the VAR(p), Γ_j ,

as the first n rows and columns of Σ_j .

What is n, p, r?

p = order of VAR.

F

so n must be the # of variables

$np \times np$

Maybe $r^2 := (np)^2$? Yep!

→ Solved bootstrap and weighting matrix

by using the estimate of the ACF from Hamilton!

A day in Provi - in honor of which 8 June 2020

I'm using Brown ; -)

→ focus on shape-preserving splines

In Matlab: pchip seems to be the way to go

Since, believe it or not, it actually is a one-dimensional object.

I just wanna find out in what relationship
pchip stands w/ Schumaker.

Well that was it ;)

9 June 2020

See Splines and Pchipps from Cleve's Corner

(saved as pdf in numerical-methods is next).

The point is that Hermite cubics and splines
are both piecewise cubic polynomial interpolants.

The difference is that Pchipps (Hermites) is the

1st & 2nd derivative conditions. A spline (at
least in Matlab) imposes 2nd derivative conditions

at the knots and at the endpoints. The Hermite interpolating polynomial only uses 1st derivative conditions. (For this reason, the 2nd derivative may have kinks and jumps, while for a spline it's smooth & continuous. Even the 1st derivative of the Hermite exhibits kinks, while that of the spline is smooth.)

In particular, the Hermite requires that at interior points, the slope is a weighted harmonic mean of the slopes of the piecewise linear interpolant, while at the endpoints, one-sided slope conditions are imposed. That's it, and for this reason, it's less smooth but instead shape-preserving than the spline.

Harmonic mean

$$HM(x_1, \dots, x_n) = \left(\frac{x_1^{-1} + \dots + x_n^{-1}}{n} \right)^{-1}$$

$$\text{Arithmetric mean} = AM(x_1, \dots, x_n) = \frac{x_1 + \dots + x_n}{n}$$

It looks like for ND data, the only 10 June 2020

hope is spapi, where

- 1) you can specify the order of the interpolating polynomial
- 2) it may be that the cubic is shape-preserving

But spapi is in B-form which is, technically,
a pp (piecewise polynomial), w/ some differences.
The coefficients are a $d \times n$ w/ breaks $t(i)$
 $(n+k)+1$

Formally, the spline f is defined as

$$f = \sum_{i=1}^n B_{i,k} a(:, i)$$

↓

i^{th} B-spline
of order k

for the given knot sequence $t(i), \dots, t(i+k)$

→ Need to use spmatrix to construct the B-form / pp.

Gustavo Guballo: Learning by Shopping

Milk Survey of $\pi\text{-E}(\cdot)$:

- large dispersion

$$\text{but } \text{disp}(\pi\text{-E}(\cdot)) < \text{disp}(\text{actual } \pi)$$

i.e. disp (price level
at individual level)

HHS

→ very heterogeneous $\pi\text{-E}(\cdot)$

• very influenced by their limited shopping
experience

Ryan working

10 June 2020

example for ndim_simplex.m! → He sent it.

↳ test_simplex2d.m

use a grid for b_t , $f_{t+1:t-1}$, then use A, b to
get initial values for b_t , $f_{t+1:t-1}$.

Actually, this is important b/c if you simulate,

You may not cover the entire domain of the functions and therefore you may be underidentified (b/c a finite-difference approach needs data from all elements, otherwise it's singular, vs. global interp / spectral methods like Cheby)

The other thing is the inputs to ndm_simplex : x and xx

x = is the whole grid: it stays constant for every evaluation of the pt.

xx = is the current input point; it changes for every evaluation.

Triangulation is about the efficient choice of triangles, once you have a grid. But for Ryan, the choice of a (uniform) grid is tied to the

triangulation b/c he just picks a triangulation that he knows exists for a uniform grid. But it's not efficient, and it seems efficient methods aren't implemented for ND (> 3).

Also note that in the binning step, `ndim_simplex.m` stretches the triangles at the edges so that if you have a data point outside the grid, you still get attributed to that edge-triangle, so you still obtain a value (this is extrapolation).

Work after

I wonder if I should do lag selection for

- 1) all bootstrap samples
- 2) model-implied datasets individually & should I use the lag selected for the initial data?

→ I always (or very often) have singularity issues w/ my regressor and I think it might come from having too many bags
(and here!)

↳ Need to solve singularity issues w/ regressors.

→ Once that is done, maybe the coeffs will update properly. → See Peter's email, saved into notes as Peter-email-11-June-2020.pdf
stochastic singularity

I'm still dropping that issue 11 June 2020

for now and turning to the ndim simplex issue.
I think it's happening b/c the gain goes sufficiently close to -inf.

⇒ How can I tell my approx not to count in negative gains??

→ If I allow the α -coefficients to move too far from α_0 , I get this issue.

Gaetano Gaballo meeting

11 June 2020

- don't fight rational expectation
 - ↳ Be specific about language
 - ⇒ smoother ways of saying you are a member of the other camp
 - produce large deviations in $\bar{\alpha}$ is very hard but we need very small deviations in the model

You use a lot ^{too much!} of the language of the adaptive learning
e.g. the understanding of RE

↳ Guisneve → you may not end up in PEE even if you know the model b/c of coordination failure

Macet & Adam & Dixson & Preston are the ones successful in settling the argument: how large is the deviation that can explain \rightarrow we

want the smallest deviation that can still explain the stuff

"W/ a small departure in modelling I can achieve deviations from \hat{a} from target and make inferences about behavior"

There should be a mapping from b_+ to a Kalman gain

Graph to explain the target criterion

→ relevant graph here: Utility indifference curve of planner

Maybe simplify into 2 periods

Ppl will not be surprised that w/o RE, discretion = commitment. So the contrib. is now the learning changes the problem.

should sell to CB b/c it's "insufficient to a
non-linear world"

→ emphasize not only what is opt pol,
but what's the mistake I make when I follow
a RE model.

- RE is great in LR
- but crisis has shown that things move quickly
- so how costly is it to do plain vanilla RE
Recall that discretion = commitment
But the cost differs depending on the model.
So I do 3 things:
 1. Est. model
 2. solve opt. policy
 3. Assess the mistake I do by ass RE

Add numbers.

A two-period problem w/ an intertemporal price
ppl can learn about, you can already show
the mistake ppl make by using the RE ass.

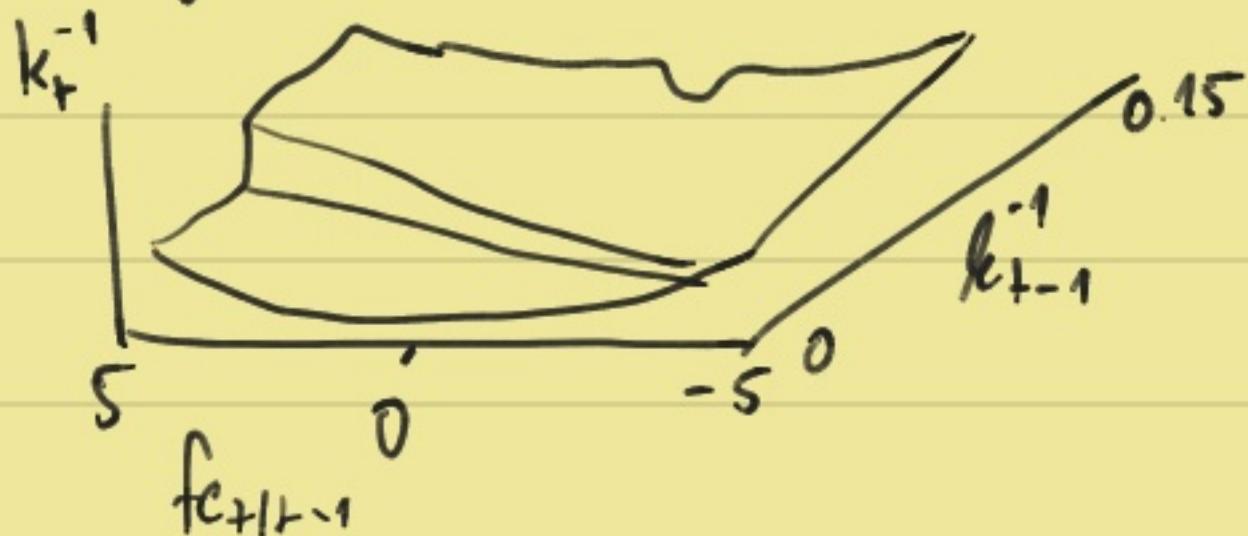
He said it's fine to fall at t₁ and up at t₂.

Work after

12 June 2020

I needed to make sure the simulation doesn't
explode for certain values of α , so
whenever an explosion happens, I set the
GMM loss to $1e+10$. (It's otherwise on an
order of $3e+02$.)

The estimated relationship (when $K > 0$ constrained)
is something like



a) For a small k_t^{-1} , a given k_{t-1}^{-1} is associated w/ a larger k_t^{-1} if the $f_{t+1|t-1} < 0$ then if it is $f_{t+1|t-1} > 0$

$$\text{NB. } f_{t+1|t-1} < 0 \rightarrow \pi_t - \text{fct} < 0 \Rightarrow \pi_t < \text{fct}$$

\Rightarrow surprise deflation / disinflation

"negative inflation surprise"

"overestimated inflation"

2) This is no longer true for high k_t^{-1} : then a given k_{t-1}^{-1} is associated w/ higher k_t^{-1} if $[fe]$ higher except for when fe is small and negative.

\rightarrow Let's rephrase in words

0) El. are always more likely to unanchor for larger FE.

1) When expectations are anchored, they are more

likely to unanchor for negative inflation surprises.

2) When El. are unanchored, then small negative

inflation surprises are associated w/ faster anchoring

\Rightarrow This may be driven by my sample which spans

1959 - 2020 (from 1 to Jan 1), so the Great Inflation is in there, under which $E(\cdot)$ were unanchored, and scary. π one is lower than expected was stabilizing. Once $E(\cdot)$ became anchored in the 90s and especially in the 2010s (after 2008), inflation shortfalls are more likely to unanchor $E(\cdot)$.

\Rightarrow Neat!

Redo PEA w/ this?

For a distance D , we seek

13 June 2020

the stepsize s that splits it into $(n_g - 1)$ parts (n_g knots) where one node is xx .

$$s \cdot (n_g - 1) = D \quad (1)$$

$$y \cdot s = xx \quad (2)$$

$$(1) \text{ in } (2) : y \cdot \frac{D}{n_g - 1} = xx \Rightarrow y = \frac{n_g - 1}{D} xx$$

$$\Rightarrow s = \frac{xx}{y}$$

This is fine but it doesn't lead to $y \in \mathbb{N}$
b/c nothing guarantees that $\frac{n_g}{D} \cdot xx \in \mathbb{N}$.

It'd be better to choose n_g such that $y \in \mathbb{N}$ both
for k^1 and for k^2 .

So I have 2 distances, D_1 and D_2 , which I want
to split into n_g points w/ stepsizes s_1 and s_2 ,
such that $xx(1)$ and $xx(2)$ lie on the grid.

So there must exist $(y_1, y_2) \in \mathbb{N}^2$ such that

$$y_1 \cdot s_1 = xx(1) \quad \text{and} \quad y_2 \cdot s_2 = xx(2).$$

while the stepsizes satisfy

$$(n_g - 1) \cdot s_1 = D_1 \quad \text{and} \quad (n_g - 1) \cdot s_2 = D_2$$

$$\Rightarrow y_1 \frac{D_1}{(n_g - 1)} = xx(1) \quad \text{and} \quad y_2 \frac{D_2}{(n_g - 1)} = xx(2)$$

$$\Leftrightarrow y_1 = \frac{n_g - 1}{D_1} xx(1); \quad y_2 = \frac{n_g - 1}{D_2} xx(2)$$

How to restrict these to be integers?

→ I solved it by simply approximating the derivs
 g_{π} and $g_{\bar{\alpha}}$ at the closest gridpoint.

Now I have the problem that the PEA is stalling
at the pdrc step. More precisely, the
estimation of the jacobian takes long (1 min)
And then the rest seems long too..

I wonder why? The pdrc step shouldn't be so
much more difficult only b/c of the damp estimated
LOM-gain.

→ No it ain't stuck, pdrc is just now having
a hard time to solve. In the 1st iter, it
found no sol. Also, it's sooo slow now
that I can't possibly do this on my Mac.
By contrast, the VFI (accelerated, 4x4x2x1 grid)
takes 30 min.

Let's also pbs it.

Ok, one feeling why it's so slow is b/c of
blk-smooth-approx.m, b/c this involves
computing a bunch of gradients, which is costly.
Can we try to do it outside?

Yes, that speeds up considerably (40 min
on server) The problem is that now f solve isn't
solving, and therefore $\|\beta^{i+1} - \beta\|$ is jumping around
all over the place.

↳ Can I cst. (p_k, γ_k) from est. LGM gain?
Do p_k, γ_k suggest too high values such that
f solve doesn't solve?

I'm concerned b/c technically,
I should have done PEA w/ the PLM "constant-only"
"n only" instead of w/ "constant only".

The reason $k^{-1} < 0$ in PEA is b/c k^{-1} goes outside

The grid I previously used is 6mm. In particular, $h^{-1} \leq 0.001$ which was my lower bound for h1grid.

15 June 2020

Ok, but GMM can't find min for h1grid = $[0, \dots \bar{g}]$
it can for $h1min = 0 + \epsilon$, where ϵ very small.
Maybe more troubling is that GMM can't solve
for PIM = "constant-only, π -only". Since that is
the view of the model on the world, it should be
that it finds a LDM gain for that view.

Discovery: my "copo" PIM sim-learning code
(constant-only - pi-only) diverges in any case.
The constant-only doesn't, slope & constant is ok
too!

⇒ the problem then shows up all over the
sim_learn LH_clean.m - family!

⇒ back to clean-up-Sim-Team.m

I did! It was only that you need to set smat so that agents know the TR! This only becomes a problem for cops b/c ~~themselves~~ they just learn it ;)

What have I learnt today?

- 1) Need to be very careful about smat.
→ may also be important for how you initialize the PEA b/c the TR simulation should be unstable unless you use the CO-PM (constant only) or you have them know the TR. → might records calling smat
- 2) Somehow in the PEA, $b^{-1} < 0$ even if I have an approx w/ only > 0 . So it shouldn't happen. Maybe I'm misunderstanding ndim Simplex after all?

- 3) Estim seems to depend on grid too
→ another reason to understand ndm simplex better. Strange: odd or even matters a lot!
→ more calls to understand how to restrict coeffs α such that $b^{-1} > 0$.

To do's for tomorrow

- rework calling smat
- investigate the behavior of simltt-clean-approx.m to see where b^{-1} might turn negative
same goes for simltt-clean-given-seq-approx.m
- go back to the estimation
understand how ndm-simplex works
understand how to keep $b^{-1} > 0$

analytic estim-
Logarithm

Ok, I think that reworking smat is to June 2020

My time to migrate PEA from the input seq

folder b/c we can't have 2 versions of everything
lying around.

⇒ command-per-main.m is the migrated
file and it works!

⇒ command-per-approx-main.m is the other
migrated one and it also works!

I've changed addpath so that the inputs for
Ryan folder is no longer added so that
there should be no duplicate files
issue

Now the next step is to create a smarter
smat that allows me to select whether
I'm 1) allowing them to know the TR
2) including a monpol shock

→ In fact, I might not have to change smat too
much. It's the ARI codes (ARI.m, ASAI0.m) and

the learning codes (`sim-learnHT-clean.m`,
`sim-learnHT-clean-approx.m`
and their `-given-seq.m` versions)

`ALM.m` & `AGA10.m` (\rightarrow the 4 sim-codes)

get 2 new inputs

`knowTR = 1 or 0`

`mpshock = 1 or 0`

\Rightarrow which they pass on to `smat.m`

(it's the easiest to edit in `smat.m` directly)

For AGA10, since it's for the case when i (at least) is input, there's never a monpol shock.

So I just set `mpshock = 0`, and it doesn't need to be input in AGA10, and thus also not in the `given-seq` codes.

Ok, `command-pca-main` \rightarrow `sim-learnHT-clean`,
`sim-learnHT-clean-given-seq`, ALM & AGA10 work!

command - pea - approx - main, as well as its
children sim... approx & genersg work too!

Peter meeting

16 June 2020

- Maybe the basis $s(x)$ should include $\underline{f_c}^2$
- Am I flipping the sign of the equation?
→ generate IRF: \oplus shock should be the same as a negative shock in the other
- Identification issue: it looks like it blk you don't move much from the original
Lack of ID means: 2 points in the param space generate the eq. model behavior
⇒ the anch. pt. may be overparameterized.
- There could be a tradeoff when fitting two sets of moments: one fits one better, the other

parameters fit the other set of moments better
maybe restricting or coaxes

the GMM to choose one over the other.

- maybe multicoll bmn k_t & k_e

- what I should do is:

do this routine (GMM) from many starting
values \Rightarrow repeat for 100 times.

Or 250 times. Make sure you always get the same.

Nbum:

- One way is to settle on a benchmark
Actually, you could estimate nbum!

Or "numerical experimentation led me to believe that
effect of initialization had died out after t=25
so I can focus on the true effects of E(.)"

• Keeping $k^{-1} > 0$

$\log(k^{-1})$ But shifting units is an issue.
Penalty fit approach is better than taking logs
b/c logs forces you to think of % deviations
instead of absolute deviations.

Instead: impose $k^{-1} > 0$ as a global property
 \Rightarrow calculate k_t^{-1} for all the possible values
of k_{t-1}^{-1}, f_t and impose that to be > 0 .

Work after \rightarrow it is clear that my first step
is to simulate the model for $\hat{\alpha}$ and see
what behavior it implies. Calculate IRFs

I do.

1. Sim model using α_0 :

• know $TR = 1$ is fine, $= 0$ explodes

• import or not doesn't make a diff.

• k^{-1} is never < 0 in any case

but that's fine,

\nearrow PEA can deal w/ that.

I settled for $n_{drop}=5$. If α unrestricted, the picture is fairly consistent, but uninteresting. If I restrict $\alpha > 0$, then ngrid has a huge effect. Grr..

$\alpha > 0 \rightarrow$ implies a gain simulation that stays up, but α unrestricted an oscillatory gain, even worse.

Some thoughts:

1. fe^2 instead of $fe \rightarrow$ doesn't make sense to me b/c if the relationship is a square, then a piecewise linear approx should tell me!
2. Anchoring fit overparameterized would mean that there's one (or more) param too many.
↳ Daa! Wait: α is $(\text{ngrid}, \text{ngrid}) \times 1$ and I have 45 moments... So $\text{ngrid}^2 \leq 45 \Rightarrow$

→ n_{grid} should be 6 at most.
But strange; I seem to have similar problems
across $n_{grid} = 4, 5, 6$.

Tomorrow: try generating IRFs using \hat{d} and
see if you're flipping things.

Maybe generate more moments?

- I've imposed $b_i^{-1} > 0$ as a "global" 17 June 2020
property and it seems to work.
- I don't get the overparameterized issue: if I have
less coeffs than moments, shouldn't I be fine?
Maybe not. Wait: no identification is not the
same as overparameterization.

"No ID": = more coeffs than moments

Overparameterization := you fit more coefficients than
what describes the true DGP. → but I'm

surprised b/c if some α 's are "unnecessary",
then the estimation should set those to 0, no?

I also recall now suddenly that for PEA,
I maintained $k_t^{-1} = g(\mu)$ ~~only~~, i.e. no k_{t+1}^{-1} ,
b/c that's the assumption I used for the TC.
→ so I'd need an estimated fit form in function
of μ only!

↳ So I need to try the 1D estimation anyway!

Also: could it be that I'm forcing overparameterization
to happen by setting $\alpha > 0$? Since I'm not
allowing the useless params to drop out?

→ And indeed, α^* has a 0 if unrestricted.
(at least for $n_g=4$, not for $n_g=6$ or $n_g=5$)

→ No, and in this sense overparametrization
and no ID may be related. Take the cue

from Bryan meeting 22 Jan 2020: overparameterize means to "overfit" a model: you estimate too many things. The problem w/ that is that the model has too many degrees of freedom which means that you'll have a good fit and yet forecast poorly.

→ this makes perfect sense in relation to what Peter said: you have bad ID b/c you find several configs of parameters that fit the moments. But they can imply different underlying model dynamics, which is why you forecast poorly.

Wibi: an overfitted model contains more params than can be justified by the data.

Ok, I'm just confused b/c estimating a functional approx should still kinda tell me when I'm overfitting [i.e. if $k_{t+1}^{(i)}$ doesn't belong in the model]

its coeffs should be set to zero?)

Let's try to understand $\alpha = (\text{ngrid}^2 \cdot 1)$

We could $\alpha^T := \text{reshape}(\alpha, \text{ngrid}, \text{ngrid})$ to get 4 "points", each characterized by 4 values, 2 pertaining to k_{+1} , 2 to k_- .

I'm tempted to interpret these two points as "start" and "end" of the finite-element.

Wait: maybe actually α contains just the heights (z-values) associated w/ each (x, y) -point contained in $\text{ngrid} \times \text{ngrid}$.

\Rightarrow yes! That's it - what proves it is setting

$$\alpha = \text{vec} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0.66 & 0.66 & 0.66 & 0.66 \\ 0.33 & 0.33 & 0.33 & 0.33 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

k_{+1}^{-1} -dimension (y)

k_{+1}^{-1} -dimension (x)

So $\alpha(1,4)$ says that for $(\text{elgrid}(1), \text{egrid}(4))$, $k_{+1}^{-1} = 1$.

\Rightarrow This has several implications:

1) $\alpha \geq 0$ makes sense b/c k_t^{-1} ($= z - \alpha h_t$)

may never be negative, and $\alpha < 1$ too

2) If I write $\text{reshape}(\alpha, ngrid, ngrid)$, I'd expect α^r to be

2.1) increasing in the row dim

(as k_t^{-1} is higher the higher k_{t-1}^{-1})

2.2) a parabola in the column dim

(as k_t^{-1} is higher the higher $|f_{t+1-t}|$)

So in principle, if I restrict $\alpha \in (0,1)$,

• k_t^{-1} should never be negative ✓

• f_t can still be non 0/c k_t^{-1} might still explode

Ryan meeting

17 June 2020

- Need curvature in \mathbf{f} & \mathbf{k} dim, not in \mathbf{k} -dim
→ 2 points in \mathbf{k} -dim may be enough.
- Increase \mathbf{k}^{-1} support → let min \Rightarrow should be band width.
- Plot ACFs at starting point vs. endpoint and see if the source pushes in the right direction
- Legendre better → b/c it uses info on every moment, not just the quadratic form

Stack weighting matrix and multiply residuals one-by-one.

- do \mathbf{f} & \mathbf{k} go off the grid a lot?
Extrapolation isn't a big problem here.
In higher dimensions (3D grids or more)
extrap. could have jumps.
- Simulate model to get data and see if you can match them.

Add to the target vector all the gridpoints at 0
add a weight to x_0 so the only then are
updated when data really wants it to.

When a bump appears, then the cshm
only has iip on that part of the grid!

↳ Only add a weak prior if you think
the AR(1) is a bad description of the
anchoring function.

↳ This is to examine if the cshm only
visits certain areas of the grid and therefore
does whatever it wants in the other areas.

↳ This is the finite elements problem: it needs ip
on points in every element to be identified.

• Ryan also agreed that w/ sufficient # of finite
elements, you can match any jet-al form,
 \mathcal{L}^2 or whatever.

- He also agreed about my interpretation of α : they are the heights (z-values).
- He said one more thing which I seem to have forgotten... :S Ah I know: he said that the ACFs are likely to be correlated at different lags, so even if I have 45 moments, many of those may not be actually contributing (much) info. Therefore, adding more moments by adding more ACF lags won't add much independent info. For this reason, if you have 45 moments, you want to be well under that in terms of params b/c you may have much less truly independent moments.

Work after

→ no more explosions

- ✓ 1) Increase kgrid to $[l_b, u_b]$ on $\alpha = [0, 1]$
- ✓ 2) Should check on fe: what values does it take in a normal simulation? For α_0 , $[-2.6, 3.6]$.
- ✓ 3) Increase gridpoints in fe-dim, decrease in let-dim
 $nfe = 10 \quad nkl = 2 \rightarrow$ more robustness

Experiment w/ nfe

- ✓ 4) Plot ACFs \Rightarrow Pushing in the right direction, just not enough.
- ✓ 5) Implement lsqnonlin
- ✓ 6) Add weights to the prior, α_0 in lsqnonlin.
- ✓ 7) Estimate α on model-simulated data

→ There aren't really bumps, it's 18 June 2020
more like these ridges might be there b/c that's
where we had data? But the flats at the sides
persist.

→ It seems to do a great job initially, but when the

norm is around 0.12, it keeps converging, but very slowly.

It stops prematurely, but it's really close to matching the moments perfectly.

Yet the implied model dynamics are quite diff.

It's not quite matching the moments and it's also not quite at the true coeffs, meaning that $\alpha - \alpha^{\text{true}}$ is a vector w/ sig entries

Let's try to increase MaxFunEvals first to see if w/ more time the solver can get there.

Mm. The fct-val is decreasing but doing so very slowly. What does that mean?

It might mean that the objective function is very flat: since the deriv is close to zero, the solver is only taking small steps but it most likely has a long stretch of almost flat surface ahead of it, so it won't quite get there.

better conclusions to see where potential mistakes fit:

- 1) Close but not close enough, doesn't converge
- 2) ACFs: pushing in the right direction, but not enough
- 3) I don't really see bumps when I assign weights
→ what I do see are ridges
→ it might be that certain f_e -values do not occur in the simulation \Rightarrow check!

↳ It does seem like not all values in the f_e -range do occur: outside of $[-3.8, 3.8]$ seems to maybe not occur.

- Restrict to $f_e \in [-4, 4]$ \Rightarrow it doesn't fare much better, but a little, doesn't converge, loss=0.06.
- Restrict to $f_e \in [-3, 3]$: now you get " f_e was na" error \Rightarrow apparently f_e is exiting the grid.
It's not doing an awful job though.
- Restrict to $f_e \in [-3.9, 3.9]$: also not awful, but not great either.

\Rightarrow I don't think we have the FEM-space that we don't have values, potentially at the outer edges of the fe-grid, but that's ok... or?

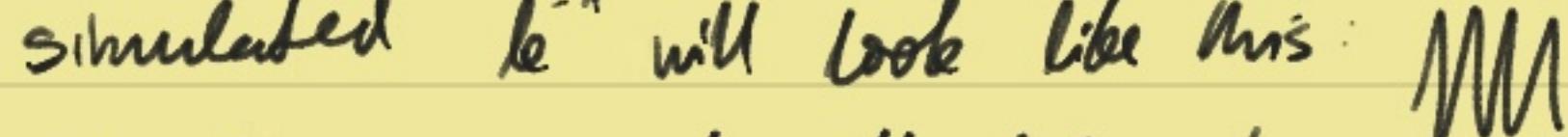
\Rightarrow Can it be that I can't estimate the α 's associated w) the ends of the grid if those simply don't show up?

I'm now thinking that if the fine coefficients α^{true} are 15×1 over fegrid, then maybe I can do better by estimating < 15 coeffs on a smaller fegrid.

$$(\text{nfe_true} = 15)$$

$$\text{loss} = 0.1$$

- $\text{nfe} = 12$: does match moments better. Doesn't converge
- $\text{nfe} = 9$: -it maybe
- $\text{nfe} = 3$: -!!- but the approx is rough and so it delivers a sim gain history that's quite different from the original.
- $\text{nfe} = 12$ and $\text{fe } [-4, 4]$: actually worse

The smaller the (k - or h -) grid, the more the simulated k^1 will look like this: 
b/c the more points will belong to a particular set that generates a particular k^1 value.

Let's now plot α^{true} , α^0 and α^{opt} and then think
Ah wait - looking at this plot, it seems to me that
in many instances, the α 's are moving in the right
direction, and in many, they remain close
to the original. It seems like the solver may
have gotten itself stuck in a local sink. Let's try
random starting points

- same thing
- also, I notice that a random set of α 's already
does a good job at replicating the ACF

a 0.5 starting point: same thing.

\Rightarrow the $\alpha^{\text{true}}, \alpha^0$ and α^{opt} graph is very help-

put here: you can see that things are moving in partly wrong directions. The fact that I'm still close to matching the moments does kinda suggest that I need to reduce the # coeffs b/c I'm under-identified.

- Smaller fe-support? $k \in [-4, 4]$
 - this gets closest to matching the moments, $\text{resnorm} \approx 0.01$ but doesn't converge.
 - and indeed α s move in the right direction
- ... and smaller k-grid: $k^{-1} \in \{0, 0.5\}$
 - nope.
- Smaller fe-support & $n_{fe} = 12$
 - It seems to me like this is closer to α^{true} at the edges (blc fe-support smaller), but of course, it matches moments less.
- $-4 - 8 n_{fe} = 9 \Rightarrow$ finds a min, but it's a higher $\text{resnorm} \approx 0.24$. The problem now

is that at this low nfe, the estim is imprecise.

Let me try a last thing: $nfe = 3$ and $nfe = 11$

- $nfe = 11$ ($fe \in [-4, 4]$)

seems worse than $fe = 12$...

- $nfe = 3$ ($fe \in [-4, 4]$)

Exactly what I thought: it converges to a min in < 10 sec and it seems to span the truth

widely. The issue is that nfe is so small
that the approx is very rough.

- $nfe = 6$ ($fe \in [-4, 4]$)

Finds a min too, same stuff.

- $nfe = 20$ ($fe \in [-4, 4]$)

Let me bet that it can fit moments well
but won't converge. It didn't, $\text{ad resnorm} = 0.08$

\Rightarrow signs are really suggesting that I have too
many params estimated: overparametrization

One more thing: I may be using the ACFs wrong.

Since I'm using for a lag k the entire ACF matrix

$$\gamma_k = \begin{bmatrix} \gamma_k(\pi) & \gamma_k(\pi, x) & \gamma_k(\pi, i) \\ \gamma_k(x, \pi) & \gamma_k(x) & \gamma_k(x, i) \\ \gamma_k(i, \pi) & \gamma_k(i, x) & \gamma_k(i) \end{bmatrix},$$

at each lag I thought I had 9 moments, but actually, I only have 6 independent ones (3 are repeated). So I have $k = 0, 1, \dots, 4$, i.e. 5 ACF matrices, each w/ 6 moments, so at best, I have $5 \cdot 6 = 30$ moments. That's why $n_{kl} \times n_{fe} = 2 \cdot 15$ may be too many params to estimate. And if I only use the "own ACFs", then I only have $5 \cdot 3 = 15$ moments.

→ so at first let's try to only take the upper triangle of the moment matrix.

Interesting: Ω only has the "repeated" 19 June 2020
feature at lag 0. I have a suspicion why that
may be the case. Hamilton, p. 266

$$\Gamma_j \equiv E[(y_t - \mu)(y_{t-j} - \mu)']$$

So e.g. $E[(x_t - \mu_x)(i_{t-j} - \mu_i)']$
 $\neq E[(i_t - \mu_i)(x_{t-j} - \mu_x)]$ if $j \neq 0$.

This is only true at lag 0, i.e. when $j = 0$.

\Rightarrow this means that using k lags, I have $(k \cdot 9) + 6$
moments b/c I need to exclude 3 at lag 0.

$$4 \cdot 9 + 6 = 42$$

But you know, they could still be correlated.

Ok. Now: let's simulate true data using very
few params $nfe = 9$. Now the estimator should
be able to uncover α^{true} perfectly.

- Overfit: $nfe = 15 \rightarrow$ doesn't converge
- $nfe = 9$ doesn't work either!

Try w/ $nfe = 6$. Sim some data w/ BMS and try to fit it.

- estim w/ $nfe = 6$: it still doesn't converge!

→ maybe I do need to take out some moments after all!

→ using the own ACFs does improve the sim.

Let's shrink fe-support to $[-4, 4]$

→ doesn't help a lot. None converge.

See the issue is that already initially, although I have an agnostic initial point of 0.5 everywhere, I match the moments quite well. So in a way,

the moments aren't super informative.

- Estim w/ $nfe = 3$. (back to $\text{fe} [-5, 5]$)

→ same as yesterday: converges, but it's a rough approx and I don't know if it's unique.

Let's go back to truth w/ $nfe = 15$ and see how the smaller moments vector affects it.

→ Sh I can't b/c then a larger moments vector was created. Grr...

Why is the moments vector well matched for any odd α ? $nfe = 6$ so I est. 12 params w/ 15 moments → maybe I'm still overfitting?

The thing though is that the estim behaves the same way for less moments as for more, except I guess the moments are matched better now (w/ fewer moments) (on the order of resnorm ≈ 0.004)
But the behavior is the same: very fast jump initially, then a long, painful, flat objective plot that never converges, unless I underfit considerably, but then the approx is crude.

→ Why are the moments uninformative? Can it be that somehow the gain doesn't make a big diff for model dynamics?

What might be the case is that the moments in the model are very highly corr, since

$$x_t = -\beta i_t + f_p + r_t^n$$

$$\pi_t = \kappa x_t + f_a + u_t$$

$$i_t = \gamma_\pi \pi_t + \gamma_x x_t + \bar{i}_t$$

Recall also that f_p moves a lot more than f_a , and κ is small, so basically the model "ignores" the effects of b^{-1} on x_t , not on π_t , and since γ_x small, i_t doesn't respond either. Let's try 1) a higher γ_x
2) a lower α (but for κ) e.g. 0.5, 0.6 or 0.7

1) $\Psi_X = 1$ Regen true data

→ doesn't converge

- still close to matching moments

- but it doesn't get α right

2) $\Psi_X = 1$, ~~$\alpha = 0.5$~~ , regen true data

↪ sim explodes, try $\alpha = 0.7$

- Now it found a min

- matches moments not bad

- but $\hat{\alpha}$ are kind of uniform

It's interesting: I wonder if this is suggesting that f_k & k_{t-1}^{-1} are multicoll b/c clearly the code sees α^{true}



as equivalent w/ $\hat{\alpha}$

That is, in terms of matching moments, it doesn't matter if k_t^{-1} responds strongly to

- \hat{e}_t , and not at all to \hat{k}_{t-1}^* (or vice versa)
- or medium to both,

→ the same moments are the result

The strange thing though is that $y_{sim}(k^{true})$ has the same moments as $y_{sim}(\hat{k}^*)$, even though \hat{k}^* is basically constant while k^{true} fluctuates a lot and the two are not equal?

→ does that mean that ACFs won't be very informative, b/c even very different y_{sim} s exhibit a similar ACF-structure in the model?

As a next step, I guess I should estimate the univariate anchoring function. If multicoll is the problem, then that should work.

Estimate univariate anchoring function

21 June 2020

• Real data:

- nfe = 6 : finds min

It doesn't move a lot from the initial moments,
but it does move some

- nk = 6, rand start: finds min ($\text{rng}(10)$)

Moves a lot from initial moments towards

the true ones, but it does not seem to get

quite the same α 's, although close.

- nfe = 6, start at 1.1 : doesn't find min

- nfe = 6, start at 0.9 : doesn't find min, even less than before.

start at 0.2 : finds sol and moves towards moments

and it finds pretty much the same sol.

$$\text{Thus } \alpha^* \approx [0.8, 0.13, 0.0002, 0.005, 0.006, 0.08]$$

- nfe = 6, start at another rand ($\text{rng}(5)$) : finds min
Yes! Same sol!

- Sim data: ($nfe = 6$)

- $nfe = 6$, start at α_0 : no sol.

didn't move in the right dir, whether in moments nor in α .

- $nfe = 6$, start at rand mg(10): no sol same.

- $nfe = 6$, start at rand mg(5): local min pinned down pretty very closely!

Both in α and in ACFs.

α_0 was similar to α^{true} .

- $nfe = 6$, start at 1.1: no sol, moved a bit.

- ————— 0.9: local min

going in the right dir, but not close enough

- $nfe = 6$, start at 0.2: no sol, moved a bit

\Rightarrow Interesting that real data vs sim-data behaves so differently, real data is robust, while sim-data not.

Do we see the same for 2D anchoring fit?

Real data: 2D

- nfe = 6, start at α_0 : local min, move in right
- nfe = 6, start at $mg(10)$: local min, -11-
- nfe = 6, start at $mg(5)$: local min, -11-
- nfe = 6, start at 1.1 : local min
moves more in the right dir, get similar thing
- nfe = 6, start at 0.9 : no sol. & barely moved.
loss was also huge: seems like far from sol.
- nfe = 6, start at 0.2 : local min, moved in the right dir. but has big flaps at the edges
Otherwise looks like the others.

Sims data: 2D

- nfe = 6, start at α_0 : no sol, though moved in right dir
- nfe = 6, start at $\text{avg}(10)$: -11- : in both, not enough
- nfe = 6, start at $\text{avg}(5)$: local min
It finds well, b/c α_0 is close to α^{true} .
- nfe = 6, start at 1.1 : no sol, but boy, did it get close! Both in α & in NCF.
- nfe = 6, start at 0.9 : no sol, but again, darn close! Here, as at 1.1, the middle of the fe-space was identified well, while the edges, no.
It's α_1, α_2 and $\alpha_{\text{end}-1}, \alpha_{\text{end}}$ that are badly identified.
- nfe = 6, start at 0.2 : no sol. but again, not bad fit in the middle.

- \Rightarrow Here's my take on things: things work like
 a charm on real data, are robust. Not on
 simulated data. Both for 1D & 2D anchoring fit.
 \rightarrow so this means that the sim data doesn't
 generate as much fe's in the "edges" of the
 fe-support as real data. Therefore the sim data
 aren't D-cd for the "outer" (first & last) α 's.
 \rightarrow There is no diff b/w 1D or 2D anchoring fit.

- 2D, nfe=6, start at 0.2, $fe \in [-4, 4]$: no sol
- $-1t + k1 \in [0, 0.9]$: no sol
- $-1t + k1 \in [0.1, 0.9]$: no sol
- $-1t + k1 \in [0.2, 0.5]$: no sol and this was bad.
- $-1t, k1 \in [0.1, 0.9], fe \in [-4, 4]$: local min but
not a great fit.
- $-1t$ but add $lb = k1_{\min}, ub = k1_{\max}$: same
- $-1t, k1 \in [0.1, 0.8]$: same.

same: but start at 1.1: no sd. Worse fit now.

⇒ Somehow my feeling is that the sim data doesn't generate enough variation in ACFs to pin down the coeffs adequately at the edges. Actually, using 6 coeffs on a smaller fe-support doesn't make sense b/c the fit will by def deteriorate: e.g. $\hat{\alpha}_1$ will never fit α_1 , b/c they pertain to a different fe-space (element)

Starting from 10 diff starting points 22 June 2020

- 1) The conclusion that real data behaves better than sim data is also here: things are D-cl w/ real data, while for sim data they're not, at least in terms of estimated $\hat{\alpha}$.

2) common across all groups is that resids are very similar too, indicating weak identification

3) Also common is that D seems weakest at the edges of the supports, but not strongly.

4) A difference between the 2 cases is that for 1D, the sim-data estimates are very consistent, but more. Is this a local sib?

I increased MaxIter & MaxFunEvals and tol and...

... sim-data 2D converged!

sim-data 2D visually suggests I might be inverting something somewhere...

... sim-data 1D converged!

I'm not sure if this suggests an inverting issue, but it might be

→ just checked thru all my objective pts and simulating pts, and they don't seem to invert anything wonky. The issue is k^{-1} vs k . But it doesn't seem to be happening. So how about the ACF-generating codes?

→ They look totally fine!

(but w/ doing moments for the 2D case!)

Now I want to see:

23 June 2020

- 1) Do we move a lot from initial α 's & moments?
- 2) Do we move in the right direction?
- 3) Can we actually get an exitflag of 1 if we increase things?

Let's always use nfe=6, start at rand, mg(5)

2D Real data

- 1) Yes!
- 2) Yes!
- 3) No.

(If the step tol is stricter,
it iterates out.)

2D sim data

- 1) Yes
- 2) yes, not always
- 3) It initially gets flag = 3

→ No: either the step is too
small or it iterates itself and

→ → My feeling is still that it's the
end where the movement stops

1D Real data

- 1) Yes! Very sim to 2D.
- 2) Yes! -11-
- 3) Well, you can change

Flag 2 to Flag 3 and get
essentially the same result.

1D sim data

- 1) Yes, but α_0 is close
- 2) Yes, but α_0 is close
- 3) → also here you can get 3
instead of 2 by lowering Tolx,
and get the same result.

And no, otherwise it times out.

→ Moments aren't informative: decrease nfe!

But the problem for the sim data is that you're underparameterized so you have a hard time fitting.

Exitflags in lsqnonlin

1 = the 1st order optimality measure was < stg

2 = change in X (stepsize) was < stg

3 = change in obj fn components were < stg

In optimset, you can only set TolX ($\rightarrow 2$)

and TolFun ($\rightarrow 3$). You need optimoptions

to set "Optimality Tolerance" ($\rightarrow 1$).

⇒ You get exitflag = 1 if the first-order opt. measure < crit. This means for me that the gradient is never close enough to zero, even if the fn value is only changing a little bit. ⇒

→ This really means that there are several things you can do to be just as close to the moments, but you can't get any closer. You're stuck at an indifference curve between several α 's, and you can't get to a lower indifference curve.

So let's try on 1D Real data w/ nfe 5

nfe = 4 flag: 2, 3, 0, 0, 0

(I always increase the stepsize tol of func tol
or MaxFunEvals or MaxIter)

$\rightarrow \text{extflag} = 0$

$n_{fe} = 2$ Flag: 3, 2, 3, 2, 2, 2, 0

→ even if f_k isn't enough. So that must mean that there's a combined issue: not only are the moments not informative, maybe the support of f_k is too small/big.

• $nfc = 6$. $fe \in [-6, 6]$ Flag : 3

6 $[-5, 5]$ 3

6 $[-4, 4]$ 3

6 $[-3, 3]$ error

6 $[-3.5, 3.5]$ 3

4 $[-3.5, 3.5]$ 3 but Opt Tol
was lower than
ever. (0.0253)

4 $[-3, 3]$ error.

4 $[-3.2, 3.2]$ 0

4 $[-10, 10]$ 2 0.0235

2 $[-10, 10]$ 2

2 $[-3.5, 3.5]$ 2

But $nfc = 2$ is very-very rough.

Let's try the same thing w/ the sim data w/
a strong weight on the prior, and starting
w/ a diff. rand that's further from the truth

(Btw. from $\text{rng}(5)$, $nfe = 6$ for $f \in [-5, 5]$
has a better fit than $f \in [-3.5, 3.5]$.)

$\text{rng}(8)$: its initial $\text{exitflag} = 0$, b/c it moves
a lot but partly in the wrong dir, although
mostly in the right.

Now $W_{\text{prior}} = 100$, $\text{flag} = 0$.

→ it stays on the prior everywhere.

$W_{\text{prior}} = 1$. $\text{flag} = 0$

→ stays on it almost everywhere.

$W_{\text{prior}} = 0.01$, $\text{flag} = 0$

→ moves pretty much everywhere.

$W_{\text{prior}} = 0.1$, ^(except at 4) $\text{flag} = 0$

→ same

⇒ There aren't really signs of the FEM issue: it's more just that I can't quite match the moments. It seems like the model-simulated moments don't offer as much wiggle-room.

An approach can be to crank up TolX and among the estimates, only keep those whose flag is 3.

- Indeed, cranking up TolX and TolFun, for 1Dsim_data, you get all 0 flags.
- Same TolX & TolFun, real data: all 0 flags.
⇒ set TolX = $1e-15$ but TolFun to stay nice like $1e-9$ or $1e-11$.

Peter meeting

23 June 2020

A DSGE estim w/o a Bayesian prior you'll see the same thing: flat loss portions, ridges, multiple peaks

Suggestions:

- recover true params from sim data
 - ↳ obj-fct evaluated at the true α better be better than its value at any $\hat{\alpha}$.
If not, then the problem has to do w/
the obj fct
 - A consistent estimator requires the loss
to have a peak at α^{true} .
 - coding error
 - or ID problem
- If yes, then the trouble is just coaxing the
solver to the sol. → Then you can just order

along the obj. fit-values associated w/ the candidate
 $\hat{\alpha}$.

Some results: Gamm obj fit is very flat but has curvature. That's fine b/c it's hard to estimate a large # of coeffs w/ the amount of moments.

Or: multiple peaks w/ very flat surfaces

try: fix all α at α^{true} except a single one, α_i ; then eval the obj fit at diff α_i ; and see what the fit looks like.
→ Do this for all the devents α_i .

If there is indeed scale ID, then need to bring in more info. Either by adding ts or El(.) or by restricting the coeffs thru the use of

info outside the model.

Schoemaker & some co: Sequential Monte Carlo
→ they try to replicate Smith-Geweke's
paper on News shocks (Econometrics)
→ Find that MCMC of SG&U got stuck at
a local max

Another case: Monika Piazzesi's paper
In the orig paper, the max was a local one.

Guard against the underestimation better than
SG&U by being honest about it, say
"the likelihood is very flat", run for 2 days
find max and we have the feeling that if
you run it for 2 months, they'd get a better
max, then put in a footnote that explains
how you got it.

Work after

- 1.) obj(α^{true})
- 2) obj(α^{true} except at one)

Let's restrict ourselves to the 1D case.

So any strategy now is to do a bunch of runs and compare fit values. Overnight we running 2 PBS-es w/ $n=1000$ and $n=10\,000$. Let's see what we get!

They are not bad, but not quite 24 June 2020
Here either: the minimum loss I can get is ≈ 30 .
→ I tried a "smart" way by where you search in the neighbourhood of the best results so far, but it's actually not doing great b/c the point is that each local min appears to be a vast plateau; a very mean local sink. You don't find anything

better around there. That's why I think
that a brute force search w/ $n=100\ 000$
(or more? my goodness)

Presentation stuff learned from Alessandro Villa:

- 1.) Nice serif font
- 2) Black header, white slide
- 3) Blue bar that shows progress
- 4) Content: simple model \rightarrow full dynamic model
- 5) Work w/ "pause"
- 6) Emphasize "behavioral"

he uses

OpenMP & Cuda to use GPU computing
and he parallelizes on CPU & GPU

Approximates "max" w/ sigmoids that look like
the kernels in nonparametric stats or like the neurons
in a neural network.

Ryan meeting

29 June 2020

Can do a grid search that's guaranteed to converge

- take a grid for each α
- optimize over just α , keeping the others fixed,
and then do that for each
 \Rightarrow could help to justify whether this is a min.

Other thing: take a 6-dim grid

evaluate the likelihood at the $6^6 = 46\,000$
to see how the likelihood looks

- Fig 9. Lines don't hit 0 \Rightarrow suggests not the truth is a global opt.

Sim data:

- Is the truth too exotic? d^{true} are non-convex.
Try to pick a nicer prior.

Worry: We might punch the PC to get the global optimum
but there are a bunch of points that look similar

↳ Swap that: I don't worry about the local
minimum b/c they are not similar to the global.

When GMM doesn't work:

- add moments
- remove params
- change model

- 1) Warm see $\alpha \in (0, 0.1)$ \rightarrow simulate model or not
- 2) Run $n=100\ 000$, investigate the top 10
 - policy pt
 - residuals
 - loss function for those top ten points

Dream scenario:

- After $n=100\ 000$, you compare to top 10,
then only \hat{t} is close to zero
- \Rightarrow Do for real data, same, get $\hat{\alpha}^{\text{true, data}}$
- \Rightarrow Redo simulation w/ $\hat{\alpha}^{\text{real data, true}}$ as the
truth to prove that estimation works.
 - ↳ Like a proof of concept that my estim strategy works.

Add moments:

- can be calibrated moments
 - or strict priors e.g. function should be
monotonic or convex
- convexity: take α 's
- take diffs twice
 - \Rightarrow should be positive
 - 2nd terms $> 0 \Rightarrow$ penalize
the loss fit for that

Make the penalty convex (increasing in

derivation) so that loss doesn't become
biksy

calibrated moment: e.g. average $b = 0.05$
in the simulation

Work after Materials 35

- 2.1) Evaluate loss on a 6^6 grid $\rightarrow g_{\text{subbed}}$
- 2.2) $a_{\text{true}} \in (0, 0.1)$ and convex \rightarrow whoa! much better!
- 2.3) 100 000 starting points, top 10 \rightarrow been running
on the server since morning
- 2.4) Add moments \rightarrow strict priors: convex
 \rightarrow calibrated moments

\hookrightarrow Only the $f_e = -5$ coefficient seems to be unidentified.

Shrinking $f_e \in [-3.5, 3.5]$ helps get a better fit

Good news: (1D function) 25 June 2020
making the α^{true} i) convex ii) $\in (0, 0.1)$ iii)
pertain to $f \in [-3.5, 3.5]$ makes the estimation
able to nail $\hat{\alpha} = \alpha^{\text{true}}$ exactly.

↳ So the estimation is able to recover the truth.
But for real data, the sim is still "bad".
I have reason to suppose that the truth is
convex, and I can shrink the f -support I
consider to $[-2.5, 2.5]$ [but not further!],
but I still seem to be unidentified
b/c starting points really matter, and the solver
mostly iterates itself out.

⇒ Add moments! Try it out on the sim data,
which now is working brilliantly!

1) convexity is the first moment I should add

$$\hat{\alpha}'' > 0$$

But the moment condition itself should be convex ($:P$) in the sense that the more it's violated, the higher should the penalty be.

→ what I'm trying is

$$\text{sum} \left[(\hat{\alpha}'' < 0)^2 \right]$$

Dam. That doesn't converge.

Try max instead of sum. → Not good!

Doesn't impose convexity. Actually sum doesn't work either! It doesn't impose convexity.

Try to discard $\hat{\alpha}$ immediately if not convex.

Nope that's even worse.

→ Yay! It works! ⇒ moment = sum (seconddiffs(seconddiffs > 0)²)

Try on real data. Still iterates itself out.

Also: a nonconvex one still made it.

Need to penalize more. (Multiplied by 10, checked and it still works for sim-data.)

So try to add the average gain-value calibrated moment: $\text{mean}(k^{-1}) = 0.05$

Now the sim data doesn't quite hit the truth (of course b/c I'm imposing they that ain't true in the "fake one world"). But it's still damn close.

Let's see the real data: it still iterates out. Is it robust at least? No.

And convexity penalty still seems too low.

↳ Try 100. (checked that sim data still works!)

↳ Try 1000 (seems to finally

work on the real data, but now the SIR data iterates out, although it's still at the correct tag.
↳ w/o the calibrated moment, it works.

Now real data can't tolerate $\rho < 3.5$.
Hm... But that's not awful. My problem rather is that whatever I do I still get the same features as before: difficulty in converging, breaking out. I mean, it's ok if the moments aren't well matched if the optimum is unique. But I don't find a lot of optima and they seem not unique.

Cont here tomorrow

16 June 2020

As a first step I'll clean up a bit so whatever we can translate can be tweaked at the same place.

Now I'm seeing that even for the sim-data, not all random starting points are able to find the optimum. The solver iterates itself out and again it's mainly the lowest μ -value that seems unidentified.

In this case, adding the calibrated moment helps getting closer to the truth a lot, but unfortunately you don't see this in the residuals b/c of the penalty.

$\text{rng}(0)$, for example, does really badly!

$\text{resnorm} \approx 75$.

Again, adding the calib. moment helps a lot, even though you still iterate out.

$\text{rng}(4)$ too, calib. moment helps.

$\text{rng}(7)$ gives $b < 0$ and therefore initial point is bad.

$\text{rng}(8)$ too!

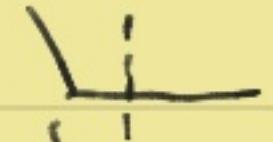
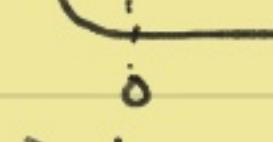
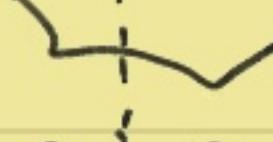
Out of $\text{rng}(0)$ - $\text{rng}(9)$, 4 don't report success! (good to

know for real data b/c when adding the calib moment helps nudge the $\hat{\alpha}$ closer to α^{true}

So this may suggest that for true data we can get there too as long as we nudge enough.

So turn to the real data and play around w/ the things we have one-by-one:

Defaults and:

nfe	flag	resnorm	$\hat{\alpha}$
• $nfe = 6$	0	resnorm = 551	
• $nfe = 5$	0	463	
• $nfe = 4$	2	resnorm = 464	
• $nfe = 3$	2	resnorm = 471	
• $nfe = 8$	0	549	
• $nfe = 12$	0	551	
• $nfe = 30$	0	535	

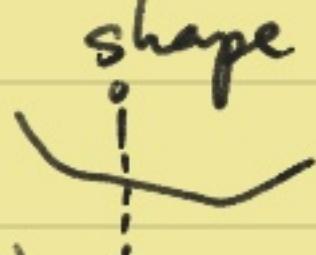
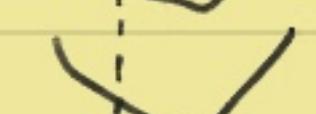
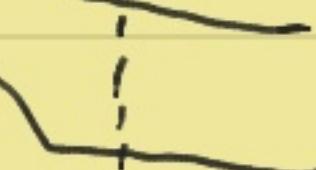
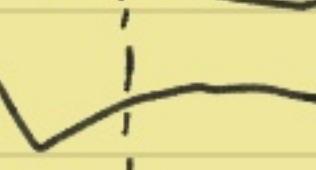
\Rightarrow This seems to suggest that we have about 4 informative

moments. add cabib moment to $nfe = 5$

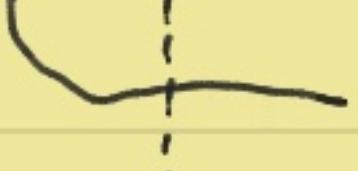
\rightarrow still 0 flag. "

$\uparrow W_{mean}$ from 10 to 100 \rightarrow still 0 flag "

Defaults and fe-support

$k \in (\cdot, \cdot)$	flag	resnom	shape
$[-4, 4]$	0	553	
$[-5, 5]$	0	557	
$[-6, 6]$	0	560	
$[-10, 10]$	0	560	
$[-3, 3]$	0	526	
$[-2, 2]$	0	461	
$[-1, 1]$	0	503	

\hookrightarrow You get some $k = \text{nan}$

* $([-15, 15])$ 2 501 

* $([-18, 18])$ 2 500 

key terms

"flipped classroom": - students do all the talking
→ but you can choose that!

↳ sorry it's been a busy day just thinking about
God and teaching and all ...

fe	flag	resnorm
? [-0.5, 0.5]	2	610



(barely moved from initial)

[-2, 2] & add crit. moment:

0	523
---	-----



Dependents and w_b & lb

lb & w_b	flag	resnorm	shape
0 1	0	551	



0	0.5	0	550	-1-
---	-----	---	-----	-----

0	0.1	0	551	-11-
---	-----	---	-----	------

0.001 1

0 559



0.01 2 initial point 438+21
is a min

→ it shifted the initial point to comply w/ lb!

default & play w/ moments.

Default is $W_{prior} = 0$

$W_{convexity} = 1000$

$W_{mean} = 0$

flag 0
res= 551



w/ change the mean moment from $|(\bar{k}^{-1} - 0.05)|$ to
 $(\bar{k}^{-1} - 0.05)^2$

$W_{mean} = 0$ 0 552

$W_{mean} = 100$ 0 567

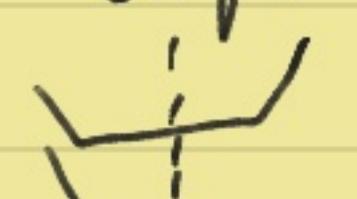
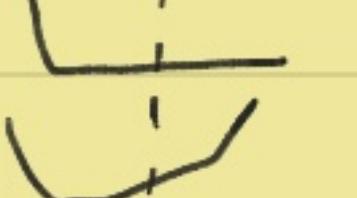
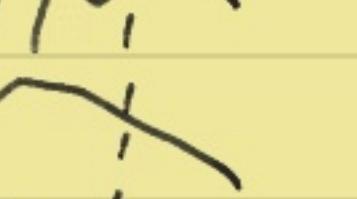
$W_{mean} = 100$, $W_{convexity} = 10000$, 0, 582

$W_{mean} = 1000$, $W_{convexity} = 1000$, 0, 639



Back to default, now change starting point

Default is the U-shaped start by lk-smooth

Start	flag	resnom	shape
0.2	0	491	
0.05	0	521	
rng(0)	0	653	
rng(1)	0	468	
rng(2)	0	563	
rng(3)	0	453	
rng(4)	0	505	
rng(5)	0	528	
rng(6)	0	490	
rng(7)	1	Initial point is min, 4.5+21	
rng(8)	-11-		
rng(9)	0	467	
rng(9) + Wmean=10	0	466	
rng(9) + Wmean=100	0	507	

Some random attempts

$$\text{Default} + W_{\text{mean}} = 100 \quad 0 \quad 567$$

$$\text{Default} + W_{\text{mean}} = 100 + fe \in [-2, 2]$$

0 479



$$\text{Default} + W_{\text{mean}} = 100 + fe \in [-1, 2]$$

2 522



$$\text{Default} + W_{\text{mean}} = 100 + fe \in [-0.5, 2]$$

now it complains fe non

$$\text{Default} + W_{\text{mean}} = 100 + fe \in [-0.8, 2]$$

fe was non again

$$\text{Default} + W_{\text{mean}} = 100 + fe \in [-1, 1.5]$$

fe was non.

But ok, we can go down to $fe \in [-1, 2]$

$$\text{Default} + W_{\text{mean}} = 100, W_{\text{overex}} = 10000, fe \in [-1, 2]$$

0 486



same but $nfe = 5 \quad 0 \quad 485$

-11-

$nfe = 4$

(2)

487

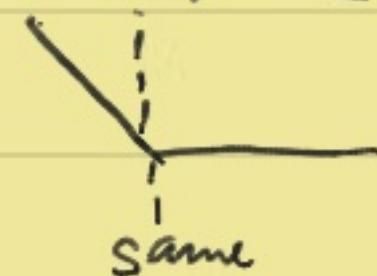
-11-

$nfe = 3$

(3)

482

same



I again conclude that I seem to have 4 informative moments (incl. the convexity moment)

Default, nfe = 4 flag 2

Default, nfe = 4, Wmomx = 0. Gives almost the same thing, so it doesn't seem to constrain a lot. (still flag 2)

Default, nk = 4, Wmomx = 0, Wmean = 200

flag 3 & almost unchanged results

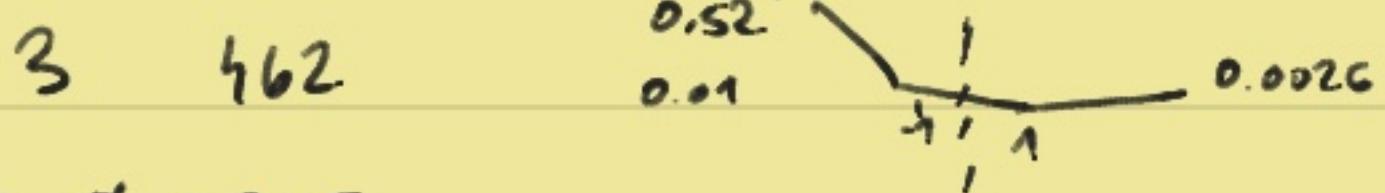
→ So even w/o the extra 2 moments, nfe = 4 solves. So we seem to have 4 moments.

The convexity moment and the mean moment don't add much binding info b/c they seem to be fulfilled by the sol anyway.

But this would in principle mean that nk = 6 (or 5) should solve w/ the 2 extra moments.

Try wfe = 4 w/ diff starting points

Default, $W_{mean} = W_{convex} = 0$, $wf = 4$ $[0.52, 0.01, 0, 0.0026]$



same, but $\alpha_0 = 0.05$

3 464 $[0.43, 0.01, 0, 0.0044]$

same, but $\alpha_0 = 0.2$

3 462 $[0.51, 0.01, 0, 0.0029]$

same, but $\alpha_0 = \text{rng}(9)$

0 467 $[0.37, 0.001, 0.0002, 0.0075]$

same, but $\alpha_0 = \text{mg}(5)$

3 464 $[0.42, 0, 0, 0.0037]$

same, but $\alpha_0 = \text{mg}(6)$

3 464 $[0.42, 0, 0, 0.0057]$

Ok, the estimates are very similar. In all of them, weak convexity holds, so that moment doesn't add more info.

You don't believe it, but

27 June 2020

$n = 100\ 000$ finished on the server after
running for almost 70 hours!

Here's the best it found:

α^{true}

$\hat{\alpha}^{\text{best}}$

0.8147

0.9791

0.9058

1

0.9134

0.8881

0.6324

0.7325

0.0975

0

w/ a resnorm = 34.5273 and min-lik = 41885

(rng('default'))

Not bad but not great. Especially as it
didn't improve on best 10 000, it's the same
point!

↳ I'm gonna return to real data instead!

Real data Default, $n_f = 4$, no additional moments

Now add $W_{convex} = 1000 : 3, 464 [0.43, 0.0005, 0, 0.0041]$

$n_f = 5$, $W_{convex} = W_{mean} = 0 : 2, 471 [0.38, 0.29, 0, 0, 0.0075]$
Ha!

$n_f = 5$, $W_{convex} = 1000 : 0, 463 [0.71, 0.0018, 0.002, 0.005, 0.01]$

$n_f = 6$, $W_c = W_m = 0 : 2, 462 [0.0065, 0.31, 0.05, 0, 0, 0.0025]$

→ This seems like it's
overshooting



$n_f = 6$, $W_{convex} = 1000$ (default) : 0

$n_f = 6$, $W_{convex} = 1000$, $W_{mean} = 1000$: 0



$n_f = 6$, $W_{convex} = 10000$, $W_{mean} = 1000$: 0



Fine but it seems like we have 5 moments
initially b/c we can match $n_f = 5$. What is
surprising is that this convexity & mean moments don't
seem to be very informative.

$n_f = 4$, $W_c = W_m = 0 : 3, 462 [0.52, 0.01, 0, 0.0026]$

$n_f = 4$, $W_c = 0, W_m = 100 : 3, 478 [0.67, 0.03, 0, 0.0006]$

$nfc = 4$, $W_C = 100$, $W_m = 100$: 3, 479 $[0.58, 0.018, 0, 0.0027]$

let's keep nfc and add wf

$nfc = 5$: 3, 478, $[0.88, 0, 0.0008, 0.0061, 0]$

↳ the convexity moment is clearly not penalizing enough.

$nfc = 5$, $W_C = 1000$, $W_m = 100$: 0, 481, $[0.77, 0.0026, 0, 0.005, 0.01]$

↳ and when it is, it no longer converges.

same, $\text{MaxFunEvals} = 1000$ (instead of default = 500):

2, 481, $[0.77, 0.0026, 0, 0.006, 0.01]$

same, $wf = 6$: 0, 563, $[0.07, 0.04, 0.02, 0.01, 0.0038, 0.073]$

↳ given that the two new moments may not always be informative, it may be that we're still only able to estimate $wf = 5$ consistently.

same, $wf = 7$: same.

But how robust is this sol?

$wf = 5$, $W_{\text{convex}} = W_{\text{mean}} = 0$, $\text{MaxFunEvals} < 1000$

different starting points:

α_0	flag	renom	$\hat{\alpha}$
fk-smooth	2	471	[0.99, 0.24, 0, 0, 0.0015]
0.2	3	457	[0.86, 0, 0, 0.0061, 0] ! not convex
0.05	—		—
rug(0)	—		—
mg(2)	—		—
mg(5)	0	461	[0.72, 0.0001, 0, 0.3668, 0.0013] not
rug(8)	3	457	[0.86, 0, 0, 0.0061, 0] not convex

→ If it finds stig, then it's nearly the same thing!

Not Convex = 1000

α_0	flag	renom	$\hat{\alpha}$
fk-smooth	0	463	—
0.2	0	460	—
0.05	0	463	—
rug(0)	3	462	[0.73, 0.0013, 0, 0.0054, 0.01]
mg(2)	0	?	—
mg(5)	3	563	[0.03, 0, 0.03, 0.06, 0.16] (!)
rug(8)	0	460	—

Add Workers = 700

α_0	flag	renom	$\hat{\alpha}$
fk-smooth	2	481	[0.77, 0.0026, 0, 0.0058, 0.0107]
0.2	2	480	[0.78, 0, 0.0005, 0.0052, 0.0082]
0.05	0 ^{but nearly converged}	481	[0.78, 0, 0.002, 0.0005, 0.0058, 0.01]
rng(0)	0-11-	481	[0.73, 0.0024, 0.0009, 0.0058, 0.005]
rng(2)	0-11-	574	[0.0258, 0, 0.03, 0.07, 0.1276] ✓
rng(5)	2	482	[0.76, 0.0025, 0.0007, 0.0058, 0.01]
rng(8)	0	572	[0.0268, 0, 0.03, 0.0653, 0.1742] ✓

like rng(2)

⇒ There are some abominations, but normally it converges to the same thing, either close enough, or its neighborhood.

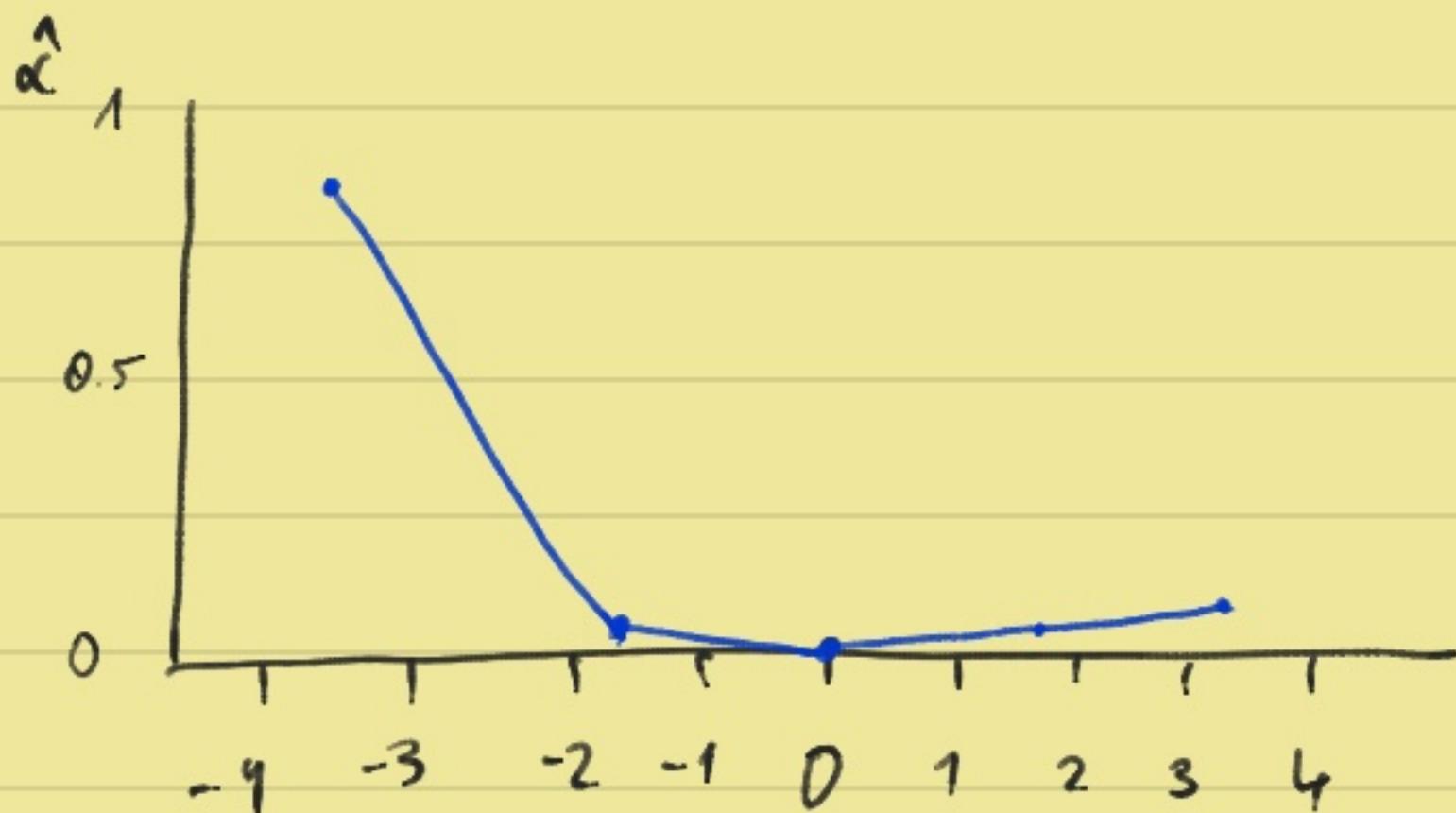
- You need to impose the convexity moment to exclude non-convex sols in the nk=5 case.
- You need to impose the mean moment to make convex sols converged.

→ But w/ these impositions, the following config produces a quite robust result:

$n_k = 5$, $f \in [-3.5, 3.5]$, $[lb, ub] = [0, 1]$, $W_{\text{inner}} = 1000$,
 $V_{\text{mean}} = 100$, starting point = f_k -smooth (or some others)

The sol is for f_k smooth:

$$\hat{\alpha} = [0.77, 0.0026, 0, 0.0058, 0.0107]$$



"a tail-back L"

↳ This would mean that people react extremely strongly to big Θ surprises, but not so strongly to others.

Robust? Trying to set up a server to check!

Peter meeting

30 June 2020

JPE Acemoglu, Robinson & An

econ growth & democracy

Really nice paper! Use a bunch of diff models.

Dem → growth?

Motivate in 2 ways:

1. "Dem is a weakness b/c slow & dissenting"

↳ for this reason we can't absorb this

2. Mixed results in data:

- on avg, growth ↓ before econs become democratic

⇒ coming up w/ a way to control
for that is key for causal effect

⇒ so bunch of ways/models

↳ get around w/o a single graph or two-period model

A couple of annoying news/details 1 July 2020
about the optimal TR w/ estim params. Now I get
a loss that's always below loss^{RE} .

- I can't do Cusum b/c it always explodes (why?)
- I can't do CEMP crit b/c it relies on the ACM
formulated as A_a, A_b, A_s , which I don't have
any more.

↪ I'm changing these plots a bit, so both losses
are next to each other.

Given I should rework Cusum & CEMP :S

Wait: I've uncovered a parameterization issue:

1) What should $\alpha = ?$

2) Is $\alpha = \frac{(1-\alpha)(1-\alpha\beta)}{\alpha} \gamma \approx \frac{1-\alpha\beta}{\alpha} \gamma$?

↪ I'm relatively sure it's the former!

Try to find in experimentation notes!

Rosen et al Exchange Rate Disconnect Redux
seems to offer evidence for expectations
being rational but moving (learning) slowly

Ryan meeting

1 July 2020

- How much does the loss function change as you go from $\eta k = 5$ to 4 to 3?

→ Ryan & Roberts, p. 49 (Mac)

- Show autocorriogram (covariogram)
so we can describe where it succeeds
and where it suffers.

One might be a good fit for it, the other for output

- Try to not generate the same shocks
- Shouldn't it be stationary (i.e. go back to 0)
when the gain is falling? fig.

- dourish: I should be worried if ppl adjust their E(.) downward.
- This Friday to 15th July: he's on vacation.
 - 16th/17th is the earliest we can talk if I send an email
 - can send paper / draft, he won't read it during holiday.
 - can still email him
 - e.g.恭禧gram and some thoughts

As for others' comments: this is a complete paper, with the estimation, it will be. Need to convince ppl externally and internally that it is.

Work after

I see in my Notes for experimentation,
Notes 3 - CEMP-estimation - Preston-Simulation, p. 61

The Calvo param α

$\frac{1}{1-\alpha}$ = avg (=expected) duration of a chosen price
(in periods)

So if $\alpha = 0.9$, then 10 periods (10 quarters)

$\alpha = 0.7$, then 3. $\bar{33}$ periods (3. $\bar{33}$ quarters)

$\alpha = 0.66$, then 2.94 periods (3 quarters)

$\alpha = 0.5$, then 2 periods (2 quarters)

Says who: Prices change every ___ months:

Bils & Klenow (2004) 4.3 months

Klenow & Krusell (2008) mean: 7-9 months
median: 4-7 months

Nakamura & Steinsson (2008) 7-9 months

Klenow & Malin (2010) : adjust for sales 6-9 months

Eichengreen, Fairmonitch & Rebels (2008) 0.6 months

Collected slides, 5, p. 43.

But where do I have this info from? Wiki?

But actually, see Hristov (calvo-hristov.pdf
in next > Literature)

He says: $\Theta := \alpha := \text{Prob(unchanged)}$

Expected lifetime of a price is for how many periods
this price is expected to remain unchanged = $\frac{1}{1-\Theta}$

→ So at least that's correct!

Only Collier is wrong to say that $\xi_p = \alpha = 0.66$
or 0.75 → adjust prices every 3 to 4 months, b/c
it's only months if that's the period of the model.
It's period. Not months.

→ We need stay ranging from 4 to 9 months,
i.e. 2 quarters seems ideal, or 3 at most

→ $\alpha = 0.5$ or $\alpha = 0.66$

Maybe $\alpha = 0.5$ is actually fine: prices change
on average every 6 months. Seems reasonable.

Now to α : finally I've found when I discuss this in my next Notes 2 "after DW", starting on p. 67 (Mac), bottom.

Mean of empirical evidence is 7.56 2 July 2020
months. This is $7.56/3 = 2.52$ quarters

$$\frac{1}{1-\alpha} = 2.52 \text{ quarters} \Rightarrow \frac{1}{2.52} = 1-\alpha$$

$$\alpha = 1 - \frac{1}{2.52} = 1 - 0.397 = 0.603$$

$$\underline{\alpha = 0.6}$$

So we can do $\alpha=0.5$ as a flex price extreme, $\alpha=0.7$ as a sticky price extreme, or we can settle for $\alpha=0.6$ which corresponds to the mean of estimates and to an average duration of ≈ 7.5 months.

$$k = \frac{(1-\alpha)(1-\alpha\beta)}{\alpha} \cdot \gamma$$

and the question is what is γ ?

In COMP, $\gamma = 1$ b/c there's no demand side.

In Baum, $\gamma = \eta + \theta$ b/c there's a demand side but common factor markets

In Preston, $\gamma = \frac{w + \beta^{-1}}{1 + w\theta}$ b/c demand side and specific factor markets

w = elasticity of a firm's real marginal cost function
not its own output, $y_{+}(i)$

θ = elasticity of demand?

Woodford: γ = measure of strat comp. in price setting
when $\gamma \uparrow$, strat comp \downarrow

Ok but the point is that in any case, I forgot the $(1-\alpha)$ up front.

Prop 3.3 in Woodford

$$g = \frac{(1-\mu s_m)(s_y + s_Y)}{1 + \theta [e_p + (1-\mu s_m)s_g]} \quad (14B)$$

So if you are . no intermediate inputs ($s_m=0$)

- common factor markets ($s_y=0$)
- constant desired markups ($e_p=0$)

$$g = s_y \quad \text{and} \quad s_Y = \omega + \beta^{-1}$$

\uparrow
 η I guess.

Supp there are no intermediate inputs ($s_m=0$).

$$\text{then } g = \frac{s_y + s_Y}{1 + \theta [e_p + s_y]}$$

Supp There are constant desired markups w/ levels of output produced ($e_p=0$)

$$\text{then } g = \frac{s_y + s_Y}{1 + \theta s_y}$$

Then the only things we need to understand are

- 1) When specific factor markets, $s_y = \omega$, $s_p = \beta^{-1}$
while for common factor markets, $s_y = 0$, $s_p = \omega + \beta^{-1}$

$$g = \frac{\omega + \beta^{-1}}{1 + \omega \theta} \quad \text{vs.} \quad g = \frac{\omega + \beta^{-1}}{1}$$

specific

common (Basin)

- 2) What is ω and how does it subsume the
Frisch and other stuff?

- 1) Ok, I think I see. Recall that a loglin of the
MC-function is

$$\hat{s}_r = \omega \hat{y}_r'(1) + \beta^{-1} \hat{Y}_r - (\omega + \beta^{-1}) \hat{Y}_r^n \quad (1.15)$$

where $\omega = \text{El of } s \text{ wrt } \hat{y}_r'(1)$

When common factor markets, s_r is the same as

and so it's as if $\hat{y}_r'(1) = \hat{Y}_r$ in this regard,

so $s_y = 0$, $s_p = \omega + \beta^{-1}$. No distinction.

When specific factor markets, $s_y = \omega$ and $s_p = \beta^{-1}$.

- 2) What is ω ? It's $\omega = \omega_w + \omega_p \quad (1.16)$

$$\omega = \omega_p + \omega_w \quad (1.16)$$

↑ ↑

- elasticity of elasticity of work
MPL w.r.t output (Frisch?)

→ price demands → wage demands w.r.t output

Here we have it: Woodford p. 165

3 July 2020

w/ common factor markets, the wage is independent from
the amount firm i produces (that is, from $y_t(i)$)

and so $s_t(\cdot) = p_t s(x_t, Y_t; \xi_t)$

where $x_t = \int' y_t(i) di$. So technically, $x_t = Y_t$
and there's no distinction between $y_t(i)$ and Y_t in
terms of effects on MC , $s_t \rightarrow s_y = 0$, $s_y = \omega + b'$

\Rightarrow this allows me to think of ω as the elasticity
of MC to "some relevant output" coming from
labor.

$$w = w_w + w_p$$

↑
elasticity of disutility
of work (Frisch, see
Basm Sum 1, p. 44 Mac)

elasticity of γ (1.9, p. 148)

$\hookrightarrow = MPL$

$w_p = \text{El MPL wrt output}$

wrt output

$$\text{so } w = \eta + w_p$$

↑
Frisch
elasticity
of disutility
of work

elasticity of MPL

w/ Cobb-Douglas $y_t = A_t h_t^\alpha$
(and K is in fixed supply)

$$MPL = \alpha A_t h_t^{\alpha-1}$$

$$\frac{\partial MPL}{\partial y_t} = 0 \rightarrow w_p = 0$$

$$w = \eta \quad \text{voilà!}$$

One last thing is γ vs γ^{-1}

Basu Shm 1, p. 56 Mac

$$\hat{H}_t = \varepsilon_{HW} (\hat{x}_t + w_t) \quad (\text{log-lin LS})$$

↑ Fresh clothing

Needs to be $\varepsilon_{HW} = \gamma$,

estimates say $\varepsilon_{HW} \approx 1$

Basu Shm 2, p. 46 Mac:

$$\gamma_t H_t^{-2} = \hat{x}_t \frac{w_t}{\hat{P}_t} \quad (\text{LS})$$

$$\Rightarrow \hat{x}_t + \gamma \hat{H}_t = \hat{x}_t + \hat{w}_t - \hat{P}_t$$

Supposing $\hat{x}_t = 0$,

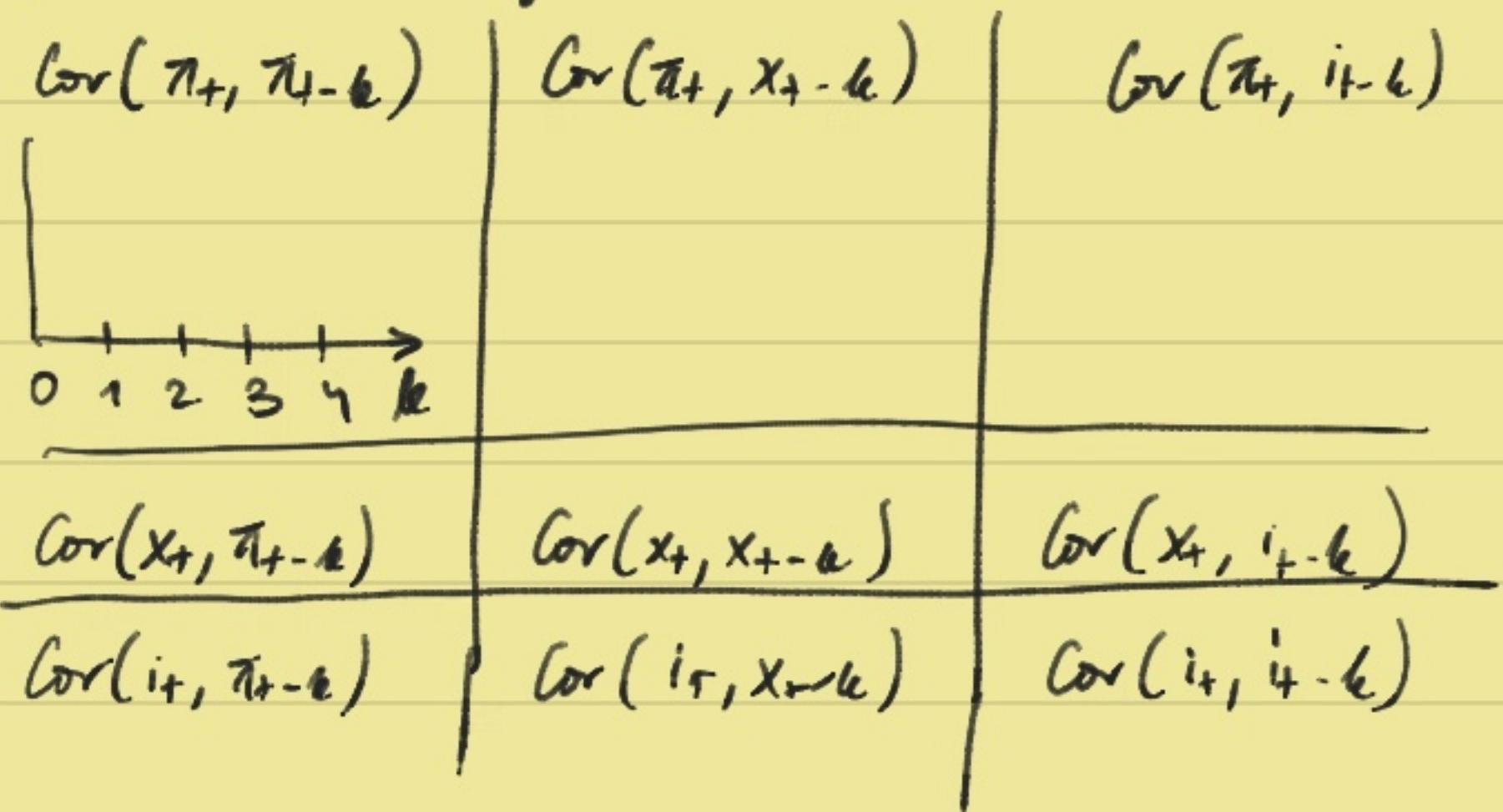
$$\hat{H}_t = \frac{1}{\gamma} [\hat{w}_t - \hat{P}_t]$$

$$\varepsilon_{HW} = \frac{1}{\gamma} = \gamma \Rightarrow \gamma = \frac{1}{\gamma}$$

If I use the number $\omega = 1.25 = \frac{5}{4} = \gamma$,

then $\varepsilon_{HW} = \frac{4}{5} < 1$ ok!

ok, so autocorogram:



Let's try to extract these from $\Omega = 45 \times 1$

$\Omega = \text{vec}(\Gamma)$ where $\Gamma = 3 \times 3 \times (K+1)$

For this purpose, create $A = 3 \times 3 \times 2$

$$\text{with } A(:,:,1) = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} \quad A(:,:,2) = \begin{bmatrix} 10 & 13 & 16 \\ 11 & 14 & 17 \\ 12 & 15 & 18 \end{bmatrix}$$

$$\text{Then } \text{vec}(A) \text{ produces} = \text{vec}(A) = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \end{bmatrix}$$

This means that $\text{reshape}(\text{vec}(A), 3, 3, 2)$

should give you back A: yep!

⇒ So then I can get back $\Gamma = \text{reshape}(\Sigma, 3, 3, k+1)$

The autocovrogram is beautiful. The problem is that I'll need to regenerate the truths if I change calibrated params, and the real data estimation doesn't look great either in response to param change. But resnorm is smaller, so maybe these new params are good?

Now creating truth using the 4 July 2020
new param values → it's not able to solve.
What about w/ convexity moment? Not quite.
We still seem to have a FEM problem, potentially
an under ID problem wr.

Ok so let's be systematic.

Truth: $n_k = 6$, $f \in [-3.5, 3.5]$, convex.

I started out w/ est. 6 α 's, on $f \in [-3.5, 3.5]$
w/o additional moments.

Then w/ the convex moment. They don't find
a sd and aren't convex either.

$W_{\text{convex}} = 0, 1000, 10000$ and 100000 doesn't work.

I'm wondering if I can improve this moment
differently somehow, in a more heuristics fashion.

try $W_{\text{convex}} = 1000$ (defaults but $W_{\text{mean}} = 0$)

	fly	rconom	convex?
default	0	1046	no
$n_k = 5$	3	1082	no
$f \in [-2.5, 2.5]$	2	1037	no
$f \in [-1.5, 1.5]$	2	828	No!
$f \in [-1, 1]$	2	887	No
$f \in [-0.5, 0.5]$	3	1056	No

	fly	econom	convex?
$f \in [-0.1, 0.1]$	0	10.98	No
$n f = 5, f \in [-0.1, 0.1]$	2	1111	No!

Back to default, w/ $W_{\text{mean}} = 0$.

→ Try to fix W_{convex} .

Instead of summing and adding into previous moments, I now add a separate moment
so res is $(45 + \# \text{ of additional moments}) \times 1$.

But it still doesn't seem to be strong enough to impose convexity!

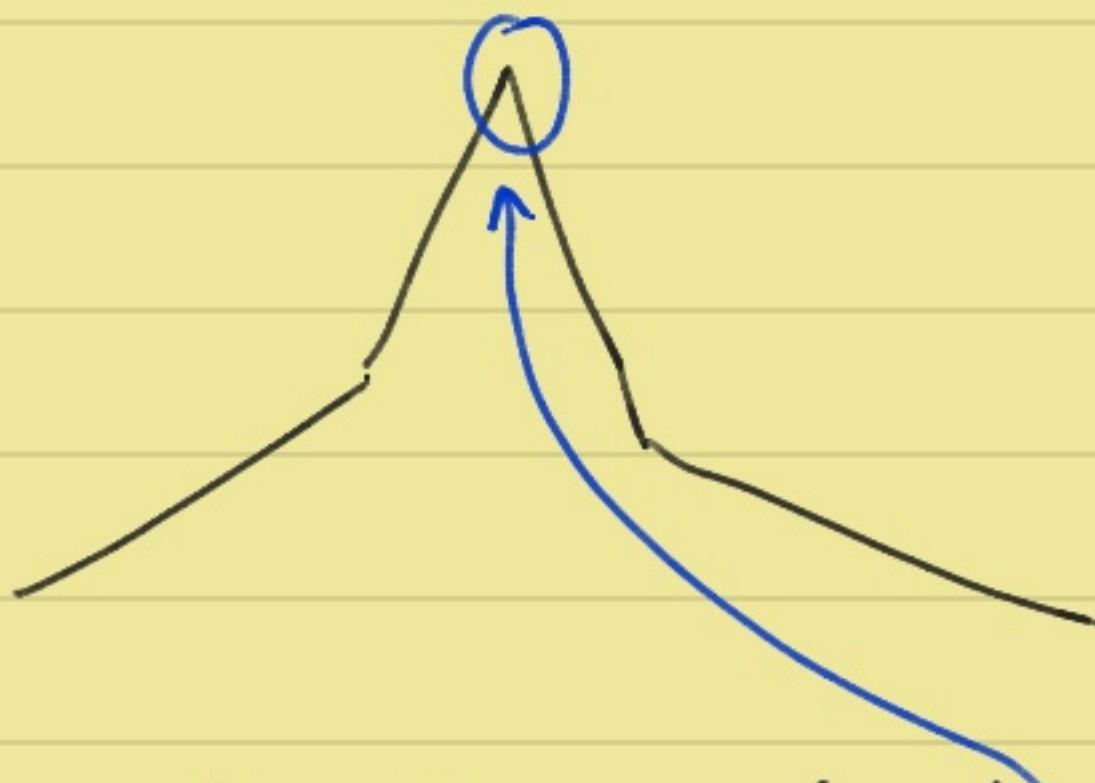
Crunching $W_{\text{convex}} = 10e+10$ is now enough to do it, but bardsy moves from the initial.
I need to change it further.

Maybe I need to add a moment 5 July 2020

that says that the first set of 1st derivs should

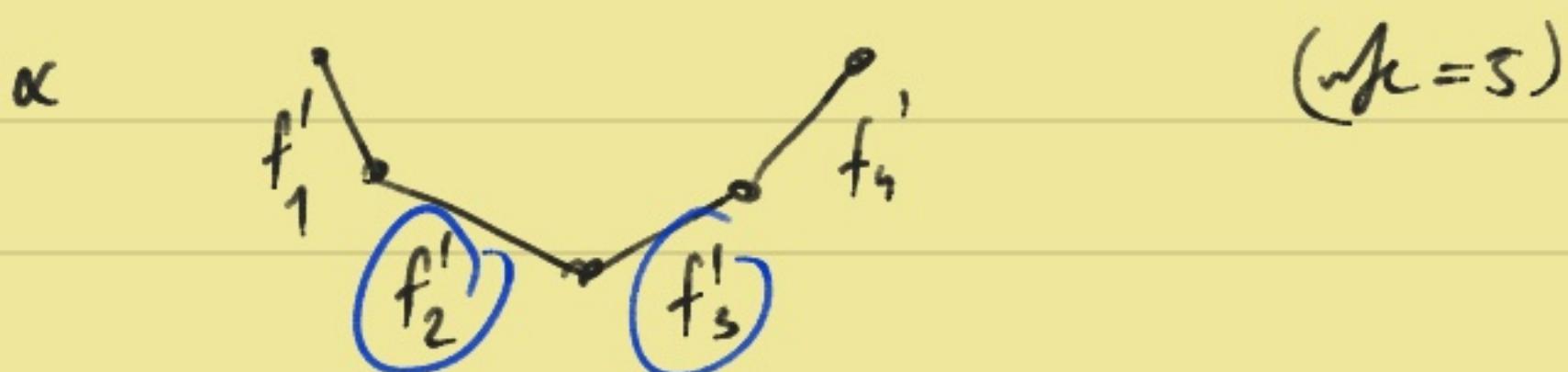
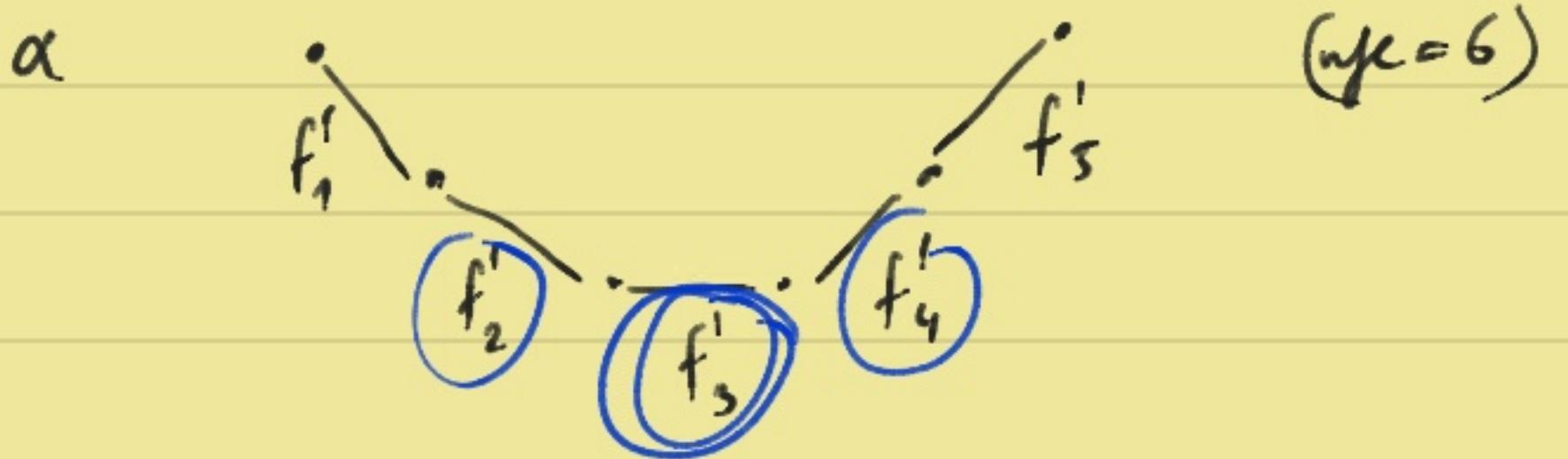
be < 0 , while the 2nd set > 0 .

The problem is that a shape like



is currently only a problem here b/c that's the only point where the 2nd derivative is < 0 . But if I also tell it that the first few 1st derivatives ≤ 0 and the last few ≥ 0 then I can restrict the shape to be at least weakly convex.

If $n_f = 6$, then there are $n_f - 1$ first derivs and $n_f - 2$ second derivs.



I feel that my priors are less strong on what should happen in the middle — in particular, I don't know where the first derivative f' changes sign. So maybe the least restrictive thing to do is to not restrict the middle f' 's. Instead, just restrict $f'_1 \leq 0$ and $f'_{end} \geq 0$.

That doesn't seem to be enough. So let's do the following: $\frac{(nfc-1)}{2}$ is the first half of the first derivative. If $\frac{nfc-1}{2}$ is integer,

then take those. Else take floor($\frac{\alpha^{\text{true}}}{2}$)

Wait - right now $\text{objcrit}(\alpha^{\text{true}}) \neq 0$!

For $\text{rng}(0)$ -generated shocks, $\text{obj}(\alpha^{\text{true}}) = 0$.

Ok, so my (3) generated shocks can actually do a pretty good job, w/o additional moments. $\text{econom} = 865$. $\text{flag} = 0$ though :(

Doing a couple of diff $\text{rng}(\cdot)$'s seems to suggest that the additional moments aren't changing the sol. a lot \rightarrow they're just making it a little prettier, but ultimately, it's the same.

\rightarrow it depends more on the ($\text{rng}(\cdot)$ of the) shocks than on the moments. That's bad news and sounds like lack of ID.

\rightarrow You know what? Maybe each derivative

should be a moment!

So need to be very smart.

6 July 2020

See the problem might not be that we're under-1D b/c if that was so, we should be able to ID smaller up perfectly.

The estimated $\hat{\alpha}$ don't seem robust to up - but that doesn't really mean anything b/c if I initially estimate $w_k = 6$



then for $w_k = 2$ it's not easy to say which fit is better:



Even $n_k > 6$ (the motto for sim-data) converge to
8ty, although I don't think it's robust, but so isn't
the lower n_k estimates.

→ So I kind of wonder if the problem is
not as much a GMM - problem (a moment problem)
as I thought? Could it be that it's more a FEM-
or a gain-value problem? E.g. I see that
the gain in the simulation using α^{true} is always
 < 0.05 , while the α^{true} -values exceed 0.05
3 times! I'm also concerned about the FE-values
which really depend on the specific shades.
Also wondering if \tilde{g} , the initial value of the gain
matters? (Btw, should begin calling it k_0^{-1})

So, looking at the gains k^{-1} and k implied
in the sim by $\hat{\alpha}$ (the one that looks like \wedge)

I see that k^{-1} is bunching around 0.07 with little variance, f_k is mainly b/w $[0.5, 0.5]$. So my feeling is that GMM is choosing, in the f_k -support that's available, the gmm that fits the one that it observes, more or less. That's why the $\hat{\alpha}$ look the opposite of the truth:



while the truth is



\Rightarrow try to create a truth that

- i) is defined on a smaller f_k -support so that this support can actually be visited by most sims
- ii) has $\max(k^{-1})$ values that are actually generated by sims using α^{true}

1) $w_k = b$, α^{true} unchanged, $f_k^{\text{true}} \in [-2, 2]$

Well, that initially doesn't work b/c doing the

estims on anything smaller than $k \in [-3, 3.5]$
yields errors. \rightarrow but it doesn't work
even for $mg(3)$ where the strokes allow me
to go to $k \in [-2, 2]$.

Same w/ $mg(2)$; it doesn't converge at all!

Let's try another drift that's convex, but flatter:
and pushes from 0 in the middle (calculus
suppose is the more frequently visited range)

In particular, let w_f be odd, so 0 is a k.
 $w_f = 5 \quad f \in [-2, 2] \rightarrow -2, -1, 0, 1, 2$

Extremely interesting, you get:



Now, if you mirror $\hat{\alpha}$ (multiply by -1 and add

the lowest value to all to push it up) you
nearly get the truth.

Ok listen here, what I try for this scenario is
plug $\hat{\alpha}$ ^{true}, $\hat{\alpha}$ ^{inv} and $\hat{\alpha}$ into obj.

$$\text{obj}(\hat{\alpha}) = 710$$

$$\text{obj}(\hat{\alpha}^{\text{inv}}) = 1652$$

$$\text{obj}(\alpha^{\text{true}}) = 1514$$

And even if I do $\text{mg}(0)$, $\text{obj}(\alpha^{\text{true}}) > \text{obj}(\hat{\alpha})$ what?! I thought I'd checked that and that it was ole?! \rightarrow Ok, it was b/c I wasn't considering the right fe-space, but now it's good!

So I need to think about the role 7 July 2020

of random shokes. Maybe also IRFs. The nice thing about IRFs is that they don't depend on shokes.

Peter meeting

7 July 2020

Boston Fed Estim "return from tracking entries is the full at a crucial point in my research it would make a huge difference to me"

- generate 100 samples of artificial data and in each case generate the moments and then avg those over the 100 sims
- and match avg of moments
- LLN to wash out the randomness

sim method of moments Anthony
SMM invented by Tony Smith, 1993

→ see the paper "indirect inference"
Journal of applied econometrics

If I had a stationary model, one long ts would also

be good to wash out the contamination of shocks.

In a sense, I was previously controlling for sampling error by my(0)-by everything
→ it's telling me that sampling error is so large that it can't handle.

Put in a footnote: "the consistency properties of the est. I'm using are presented in "xyz".

Work after

It looks like Smith (1993) simulates N=1000 samples, estimates the params on each and then looks at the mean.

Duffie & Singleton 1990: don't mention a cross-section, just that the moments themselves are time-averages

Lee & Ingram (1991): same (1,3,Mac) \Rightarrow shows cons. & asymptotic N.

They all seem to agree that the sim data should be longer than the obs data.

Fuck knows! I have no choice but to try.

But: first, mend the fk-comp criterion using smat.

It works! Verified in cleanup_simLearn.m

and in command-IRFs-approx-polyt.m

I've also "re-implemented" the scalar CUSUM and interestingly, it looks a lot like the COMP-and in terms of IRFs.

The "scalar COMP" of fk-comp-scalar.m implements a criterion where $\theta_1 = \max(\text{diffs}$ b/w E(ALM)-PLM for inflation only). This is in analogy to the CUSUM-scalar that allows for fk coming from wherever to show up and

drive up θ_+ ; also here I allow diff's not just in the constant a , but also the slope b to show up. This may not actually have an impact as long as I consider a "lopo" (constant-only-pi-only) PLM model.

- Yep: in command_IRFs_approx_pretty.m
the COMP-PIs (scalar or vector) looks the same
as the scalar CUSUM. Makes sense b/c
- 1) there's no distinction b/wm scalar & vector COMP
in a lopo-PLM model
 - 2) there still is b/wm scalar & vector CUSUM b/c
firms still make fast errors in x & i as well.

Let's use the vector versions as defaults for
COMP & CUSUM and.

What I'm now looking at is the IRF generated by α^{true} (Ob poly); it looks qualitatively and quantitatively like the one generated by CEMP, but the gain looks like a c-gain, punctuating around a small value.