

Work after

Sorry I'm still stuck on the spline at least for today.

And I've found the following notes

If 4 points (x_j, y_j) $j=1, 2, 3, 4$ w/

$(-1, 2)$, $(0, 0)$, $(1, -2)$ and $(2, 0)$

then the general form $a + bx + cx^2 + dx^3 = y$

can be written

$$(-1, 2) \quad a - b + c - d = 2$$

$$(0, 0) \quad a = 0$$

$$(1, -2) \quad a + b + c + d = -2$$

$$(2, 0) \quad a + 2b + 4c + 8d = 0$$

Note that this is

$$\begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -2 \\ 0 \end{bmatrix} \Rightarrow \mathbf{a} = \mathbf{M}^{-1} \mathbf{y}$$

$\mathbf{M} \quad \mathbf{a} \quad \mathbf{y}$

Strange - this doesn't line up w/ what I've done so far for the spline.

But what I'm learning nonetheless is that my f -solve sol, while inefficient, should still get me to the sol.

But, using linear algebra, I should be able to get the coefficients. What I'm puzzled about is if I have just as many nodes as observations, then the above fails: E.g.

$$\begin{bmatrix} 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ b \\ c \\ d \end{bmatrix} = 2$$

⇒ The above method needs exactly 4 nodes in each interval! I'm instead thinking that each interval has two nodes adjacent to it.

↳ Can I write the spline objective as a matrix?

$$n+1 = n_{\text{grid}}$$

$$\begin{bmatrix} y_i \\ y_{i+1} \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & x_i & x_i^2 & x_i^3 \\ 1 & x_{i+1} & x_{i+1}^2 & x_{i+1}^3 \\ -1 & -2x_i & -3x_i^2 \end{bmatrix} \begin{bmatrix} a_i \\ b_i \\ c_i \\ d_i \end{bmatrix}$$

here is the problem

that $(a_i, b_i, c_i, d_i)_{i=1}^n$ enter

But maybe the vector can
really be the $\text{vec}(\theta)$?

$$y = M \text{vec}(\theta)$$

$4n \quad 4n \times 4n \quad 4n \times 1$
?

$$\begin{bmatrix} y_i \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & x_i & x_i^2 & x_i^3 & 0 & \dots & 0 \\ 1 & x_{i+1} & x_{i+1}^2 & x_{i+1}^3 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} a_i \\ b_i \\ c_i \\ d_i \end{bmatrix} \quad \left. \begin{array}{l} i=1, \dots, n \\ (2n \text{ eqs}) \end{array} \right\}$$

Shall we try this w/ 3 nodes?

Natural cubic spline w/ 4 nodes (3 intervals)

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 1 & x_2 & x_2^2 & x_2^3 & 0 & \dots \\ 0 & & & & & 0 & 1 & x_3 & x_3^2 & x_3^3 & c_1 \\ 0 & 1 & x_2 & x_2^2 & x_2^3 & 0 & \dots & 0 & d_1 \\ 0 & & 0 & 1 & x_3 & x_3^2 & x_3^3 & 0 & \dots & 0 & a_2 \\ 0 & & & 0 & 1 & x_4 & x_4^2 & x_4^3 & 0 & b_2 \\ 0 & 0 & 1 & 2x_1 & 3x_1^2 & 0 & -1 & -2x_2 & -3x_2^2 & 0 & 0 & 0 & 0 & c_2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2x_2 & 3x_2^2 & 0 & -1 & -2x_3 & -3x_3^2 & d_2 \\ 0 & 0 & 0 & 2 & 6x_1 & 0 & 0 & -2 & -6x_2 & 0 & 0 & 0 & 0 & a_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 6x_2 & 0 & 0 & -2 & -6x_3 & b_3 \\ 0 & 0 & -1 & -2x_1 & -3x_1^2 & 0 & \dots & & & & 0 & & c_3 \\ 0 & 0 & & & & & & & & & 0 & & d_3 \end{bmatrix}$$

$\begin{matrix} 12 \times 1 \\ Y \end{matrix} \quad \begin{matrix} 12 \times 12 \\ M \end{matrix} \quad \begin{matrix} 12 \times 1 \\ \text{vec}(\theta) \end{matrix}$

x_1 derivs at inner nodes x_2 natural spline: derivs at ends

Now I just have to figure out a good way to create the matrices. $\text{vec}(\theta)$ is fine.

$Y = \text{zeros}(4n, 1) \quad (4n, 1)$ Recall: $m=4, n=3$
 for $i=1, \dots, n$

$$Y(i) = y_i; \quad y_1, y_2, y_3$$

$$i = n+1, \dots, 2n, \quad j=1$$

$$j=j+1$$

$$Y(i) = y_j \quad y_2, y_3, y_4$$

$$\text{or } y_{i-(n-1)} = y_{i+1-n}$$

end. Y done!

$$X = \text{zeros}(4n, 4n)$$

for $i=1, \dots, n$ (the intervals)

$$X(i, i \cdot m - n : i \cdot m) = [1 \ x_i \ x_i^2 \ x_i^3]$$

$$\text{if } i=1, 4-3 : 4 \Rightarrow 1:4$$

$$\text{if } i=2 8-3 : 8 \Rightarrow 5:8$$

$$\text{if } i=3 12-3 : 12 \Rightarrow 9:12$$

$$X(i+n, i \cdot m - n : i \cdot m) = [1 \ x_{i+n} \ x_{i+n}^2 \ x_{i+n}^3]$$

end. This has the first 6 rows.

for $i = (2n+1) : (2n+1 + n - 1)$, $j = 1$

$j = j + 1$ (the first dens of the inner nodes.)

$$x(i, \text{start} : \text{end}) = [0 \ 1 \ 2x_j \ 3x_j^2 \ 0 \ -1 \ -2x_{j+1} \ -3x_{j+1}^2]$$

$j=1$ start = 1, end = start + 2m - 1

$j=2$ start = start + m - 1, end (same)

$j=1$ 1:8

$j=2$ 5:12. etc!

$$x(i+2, \text{same}) = [0 \ 0 \ 2 \ 6x_j \ 0 \ 0 \ -2 \ -6x_{j+1}]$$

end Done w/ rows 7, 8, 9, 10.

$$x(\text{end}-1, 1:n) = [0 \ -1 \ -2x_1 \ -3x_1^2]$$

$$x(\text{end}, \text{end}-n : \text{end}) = [0 \ -1 \ -2x_m \ -3x_m^2]$$

Done w/ all!

Now I want to turn back to the implementation of the TC. Recall that we're still just trying to implement the TR when agents don't know the TR, now by treating the consumption expectations eq. as a residual eq.

→ I continue to work in the `input-sequences-for-Ryan` subfolder.

What I'm thinking is that this is the time to get some simple functional form for the smooth anchoring function to work b/c both CUSUM and CEMP are super-cumbersome to work w/. Also, eventually the TC can only be evaluated for the smooth anchoring function, so I have to work w/ that. What I am now finding is that for $\Sigma = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $d=1$, $c=0.5$,

$$k_t = k_{t-1} + d - c(fe^2) \quad \text{works ok.}$$

One problem is that in this formulation, the gain either

is ultimately decreasing; or it turns negative if c is too large.

$$k_t = k_{t-1} + d - c(fe^2) \quad (\text{Alternative 4})$$

Let me try to map this to the inverse gain

$$k_t^{-1} = \frac{1}{k_{t-1} + d - cfe^2}$$

Exactly, this gives you exactly the same thing.

w/ different coeffs

$$k_t^{-1} = \frac{1}{k_{t-1}} - \frac{1}{d} + \frac{1}{C} fe^2 \text{ should work.}$$

But also here negative gains are an issue. A way to fix it is

$$k_t^{-1} = k_{t-1}^{-1} - \underbrace{\delta k_{t-1}^{-1}}_{\text{so that if } k_{t-1}^{-1} = 0, \text{ it doesn't increase}} + \gamma fe^2$$

Alternative 7

$$\text{Or for Alt 4: } k_t = k_{t-1} + d - cfe^2 k_{t-1}^{-1}$$

↳ This ain't cool, dampens too much.

Alt 7 actually amounts to an AR(1) w/ TV-mean

$$k_t^{-1} = \rho k_{t-1}^{-1} + \gamma f_t^2$$

If you set $\rho < 1$ and $\gamma = 0$, you get a decreasing gain scheme.

Annoying: if you set $\gamma = 0.001$, still decay, if $\gamma = 0.01$, everything explodes.

You can $\uparrow\gamma$ if you $\downarrow\rho$, but the problem is that it's not behaving as I want it to:

I want it to decrease generally but occasionally go back up. Alt 4 has this feature, but only for very particular parameter values.

I'm also a little worried about the fact that if it generally decreases, then a shock that hits at period 25 isn't able to work it out of appearing to be again learning.

Let me zoom in on Alt 4:

$$k_t = k_{t-1} + d - c(fe^2)$$

It's going in the right direction: when $fe < 0$,

$$k_t = k_{t-1} + d \quad (d=1) \text{ again}$$

When $|fe| > 0$, the increasing process is stopped or reversed. The issue is that sometimes

$$k_{t-1} + d < c fe^2$$

And it seems more of an issue early on in the sample

Taking $\log(fe^2)$ seems to be smoothing a bit, requiring much larger c -values. What I don't get is that $\log(fe^2) = 2 \log(fe)$ but it's taking the positive root. But then you could also do $\sqrt{fe^2}$.

Problem is that it's pretty unstable wrt param values b/c they determine whether the gain \downarrow or \uparrow after a fe .

Let me note some stuff about Matlab's solvers' stopping criteria 8 May 2020

Function Tolerance / TolFun

A solver stops if the change in the function value of the objective function is smaller than TolFun.

i.e. if $|f(x_i) - f(x_{i+1})| < \text{TolFun} \cdot (1 + |f(x_i)|)$
(to make it a relative measure)

Step Tolerance / TolX

A lower bound on the stepsize $|x_i - x_{i+1}|$.

Optimality Tolerance

A lower bound on the first-order optimality measure
↳ usually the gradient,
and it should be zero.

Matlab's solvers' "Iterative Display"

f(x) or Fval or Residual

Current objective function value.

For Fsolve: $\sum_i \text{res}_i^2$ (SSR)

(and it's called "Residual" in fsolve.)

Recall that TolFun refers to the change in this.

First-order optimality

→ first-order optimality measure, usually the gradient, should be close to zero.

Lambda in fsolve: Default: 0.01

λ_k in Levenberg-Maquardt.

Recall that this is the scalar · eye you add to $s^k = -(\mathbf{f}(\mathbf{x})^T \mathbf{f}(\mathbf{x}) + \lambda \mathbf{I})^{-1} (\nabla \mathbf{f}(\mathbf{x}^k))^T$

to estimate the hessian when determining the step size & direction. (Notes 10, p 67).

When $\gamma = 0$, the direction is equal to that of the Gauss-Newton method.

When $\gamma \rightarrow \infty$, the step goes in the steepest descent direction, but also becomes very small.

When the step is successful (delivers a lower obj func value), the algorithm set $\gamma_{k+1} = \frac{\gamma_k}{10}$.

When the step is unsuccessful, the algorithm sets

$$\gamma_{k+1} = \gamma_k \cdot 10.$$

Note: an internal default in fsolve

$$\text{opt tol} = 1e-4 \cdot \text{TolFun}$$

Since I set TolFun to $1e-9$,

$$\text{opt tol} = 1e-4 \cdot 1e-9 = 1e-13$$

\Rightarrow So first-order optimality needs to be $< 1e-13$ for the algorithm to stop.

What I'm finding is that importing k can be solved even if the agents don't know the TR but only if you initialize quite close to the TR.

I think that the reason for this is that importing an exogenous k doesn't dampen f_e and it still allows recursive f_e 's to happen.

→ This is why I'm now importing f_e 's instead. This, on the other hand, seems to dampen too much:

I think that de facto this forces the anchoring equation to be fulfilled exactly.

→ Indeed, if I give it just the TR, NKPC, NKIS as residuals for the inputs π, x, i, f_e , it solves in a minute. But it doesn't give me the TR sol back exactly :S

And it doesn't find a sol if I import $i \& f_e$ for a resid of TR only.

→ I wonder if the gradient is kinda flat in the

neighborhood of the TR so it doesn't quite converge to it?

If I set $\gamma_k = 0$ (i.e. dgnin learning) then it solves when inputting π, x, i, fe for resid NKPC, TR, NKIS.

If I also input as resid the LOM gain, it seems to have difficulty solving even for this dgnin specification.

All shot - I'm evaluating the residual wrongly when inputting fe b/c in this case A.6. is indeed fulfilled by design. It's A.7 that isn't.
→ ok corrected that!

→ Now it IS able to recover the TR alloc but it takes almost 4 min! (128 iter.)

Hell yeah!

Well, replacing the TR w/ an RE-TC
doesn't solve.

$$\underline{g_{\pi} \text{ & } g_{\bar{\pi}}} : \quad k_t^{-1} = \rho k_{t-1}^{-1} + \gamma f_{t+1|t-1}^2$$

$$g(k_{t+1|t-1}) = \gamma k_{t+1|t-1}^2$$

$$\left(g_{\pi} = 2\gamma f_{t+1|t-1} \cdot \frac{\partial f_{t+1|t-1}}{\partial \pi_t} = \underline{2\gamma f_{t+1|t-1}} \right)$$

$$\left(\widehat{g}_{\bar{\pi}} = 2\gamma k_{t+1|t-1} \cdot \frac{\partial k_{t+1|t-1}}{\partial \bar{\pi}_{t-1}} = \underline{-2\gamma f_{t+1|t-1}} \right)$$

$$f_{t+1|t-1} = \pi_t - \bar{\pi} - b_1 s_{t-1}$$

This allows me to implement "anchTCO", which is
the anchTC, setting all the expectation stuff to 1.

Interestingly, anchTCO is always just slightly above
RE-TC, suggesting that $\frac{(1-\alpha)\beta}{1-\alpha\beta} (k_t^{-1} + f_{t+1|t-1} g_{\pi})$ is ≈ 0
for all t .

Of course this doesn't solve either.

$$\frac{(1-\alpha)\beta}{1-\alpha\beta} = 0.9083 \rightarrow \text{so it's really } k^{-1} \leftarrow fe \cdot g_{\pi}$$

that's small, in particular $k^{-1} \ll g_{\pi}$

I think one issue is that I'm starting the search at the TR, but the TR clearly doesn't fulfill the RE-TL or the andTLO.

I'm kinda surprised b/c i & fe should be able to affect andTLO.

I'm also still seeing the recursive use in the errors
→ need to investigate that!

What I've now done is I've taken
the `fsoolve` to the server, not as submitting a
job, but running Matlab from the server. This
way, Matlab's parallel connects to 12 cores instead
of 2 (my Mac has 2 cores)
⇒ it is at least 2 or 4 times faster than
my Mac!

The problem is that even w/ `MaxFunEvals=80 000`
(instead of 40 000), the `fit` value has barely
changed and the first-order optimality is on the
order of $\text{e} \times 10^5$. It suggests to me that the
loss function is very bumpy : you move a
lot in some direction w/o changing the SSR,
but the gradient has very non-zero elements
everywhere.

Recursive build-up of errors:
doesn't happen!

Instead, the early part of the sample has
huge residuals for the TL, but not the late
one.

It seems like the TL residuals are much bigger
if you don't input \bar{x} & $\bar{\pi}$, even the SSR seems
bigger.

Now I'm wondering if the residual of the and TL0
is potentially exactly the quantity

$$E_{\bar{x}} \sum_{i=1}^{\infty} x_{i+1} \prod_{j=0}^{i-1} (1 - k_{j+1, i+j} - f_{i+1+j, 1} t_{i+j} g_{\bar{x}, i+j}) ?$$

Let's refer to this quantity as E_x . One option to
evaluate it is to simulate N times and take an
average. I kinda feel I wanna do quadrature instead
in order to be faster.

s is either iid $N(0, \Sigma)$

or a function of $\epsilon \sim \text{iid } N(0, \Sigma)$

$\Rightarrow f_e$ is N .

$\Rightarrow k$ is a function of N_s

$\Rightarrow \bar{\pi}$ is a function of N_s

\Rightarrow Gauss-Hermite quadrature.

Gauss-Hermite quadrature, Judd, Numerical, p. 266

Mac, (7.2.11)

$$E\{f(Y)\} = \int_{-\infty}^{\infty} f(y) e^{-\frac{(y-\mu)^2}{2\sigma^2}} dy$$

$n = \# \text{nodes}$

$$\approx \frac{1}{\sqrt{\pi}} \sum_{i=1}^n \omega_i f(\sqrt{2}\beta x_i + \mu)$$

$\uparrow \qquad \qquad \qquad \uparrow$

quadrature weight quadrature nodes

where $Y \sim N(\mu, \beta^2)$

"weights and nodes are kept in tables." p. 259.

↳ e.g. Table 7.3 p. 266 Mac

Ryan's PS6 is doing this interpolated VFI. The logic seems to be

- 1) Use 6K-quadrature to determine the optimal grid w/ weights for the tech shock (5 nodes) and the endog state (capital) individually (25 nodes)
- 2) Use ndgrid.m (inbuilt) to create a 2×125 matrix of each combination of capital (row 1) and technology (row 2) from the gridpoints above. \Rightarrow 125 points on the grid (k_i, tech_i)
 \Rightarrow generate $h = H(k_i, \text{tech}_i)$ $i=1, \dots, 125$ (labor)

This is the initial policy function for hours.

\hookrightarrow N.B. policy pt = jump as a function states.

- 3) \Rightarrow generate coefficients a_i such that $\tilde{f}_i(a_i, x)$ captures those initial policy values.
This is linear interpolation w/ kinks at each 125 points. Use Matlab in-built linear interp programs or Ryan's ndim_simplex.m

4) Generate a residual function (or an objective function) that takes a_i , the gridpoints, params as input and returns the resids to the EE at each of the gridpoints.

- Evaluate $\tilde{H}(a_i, x_i)$ at each gridpoint x_i
- generate the implied k_{t+1} at each x_i
- For each $k_{t+1}(i)$, generate the next 5 gridpoints for technology, $x_{P_{i,j}}$
- evaluate the policy function at each $x_{P_{i,j}}$
 - compute the RHS of the EE and obtain the expectation as

$$\sum_{j=1}^5 P_{ij} \text{rhs}_{i,j}$$

\curvearrowleft I guess there must be quadrature weights? Yea baby!

- return the resids $\frac{1}{c_i} - E[\text{rhs}]$, for each gridpoint i .

5) Use fsolve to find the a_i that zero out the resids.

Ok, given this, what do I do?

To progress on the implementation, I need to compute that damned expectation $E[X]$.

Ryan's PS6 seems helpful.

Note: GM-Quadrature.m is Table 7.4 full!

I've gone back to my VFT spline. Based on Ryan's PS6, I think I don't need to resolve the spline in every fitting step. Instead, you want the coeffs a , to min the residuals $\hat{V}(k_p, a) - \hat{V}(a_{i+1})$ or sthg. Ok, I think this is what full meant when he said that "the fitting step can be an unweighted nonlinear least squares procedure as in

$$\min_{a \in R^n} \sum_{i=1}^n (\hat{V}(x_i, a) - v_i)^2$$

↑
what-spline(x_i, a)

So instead of obj-spline - secant-herniate in the fitting step, use obj-vhat-spline(coffs, x_i, v_i) which for $i = 1, \dots, n_{\text{grid}}$

computes vhat-spline(coffs, x_i)

$$\text{and } \text{resid}(\cdot) = (\hat{v}(\cdot) - v_i)^2$$

$$\text{SSR} = \text{sum}(\text{resid})$$

and min that over coffs!

Nope, just

use

MatLab's
spline in

The fitting step

Adam Green meeting

11 May 2020

Really like the idea

Motivation should not be 2019

"Read policy-makers' statements: they are always nervous about unanchored"

Oliver Blanchard: "4% target", 2008

Reaction: "are you crazy, we had such a hard time getting it to 2% \Rightarrow

→ talked about a lot in ZLB & fwd guidance
↳ "we didn't wanna unanchor!"

Then wanna show results
and lots of intuition

Wanna talk about UB!

- Worrying: deterministic lengths of us
"how much & shall we promise?"
→ he thinks less! b/c we don't wanna unanchor!
- Is then fig asymmetric? go above 26° is really bad? Even if the CB has symmetric polfs, they will look like asymmetric functionally
- What happens when you Δ target?
- What does an estimated PC look like in this model?

→ present history to motivate 70s, 80s

How big does the gain need to be for the CB
to want to change its anchor/target?

→ He thinks huge!

- You need to spend first 30 min of fit talk to motivate anchoring
 - ↳ present evidence from data
- and to use that to motivate how you model anchoring
- You need to do scenarios
 - B2B / fwd guidance
 - changing A target
 - asymmetric behavior even if projs are symmetric.

Work after

- Parametric VFI w/ spline is working in Collard's example
- Stochastic VFI w/ discretization is working in Collard's example

→ Now I'm replicating Ryan's PS6
which is VFI on the optimal growth model
w/ a tech shock (growth rate) that uses
a grid for technology but interpolates
the hours policy function on a grid

$$X = \begin{bmatrix} k_{\text{grid}} \\ \delta_{\text{grid}} \end{bmatrix} \quad 2 \times 125$$

⇒ looking to solve $\hat{H}(a_i, X)$ for a_i .

I've been having some trouble w/ ndim-simplex.m
and would therefore like to use spline or pchip
so that I can construct the coefficients a_i .
(Another idea b/w is to use Cheby polynomials.)

But Matlab's inbuilt spline / pchip are 1D functions, meaning that they are scalar-valued functions (I think).

But I wonder if $H(a_i, X)$ is scalar-valued and I think so! For a level of tech, $\gamma = \gamma_{grid}(i)$ and a level of capital $k = k_{grid}(:)$, i.e.
for the gridpoint $x(i) = \begin{bmatrix} k_{grid}(i) \\ \gamma_{grid}(i) \end{bmatrix}$,
hours takes on one value.

\Rightarrow So need to try whether spline/pchip can handle that!

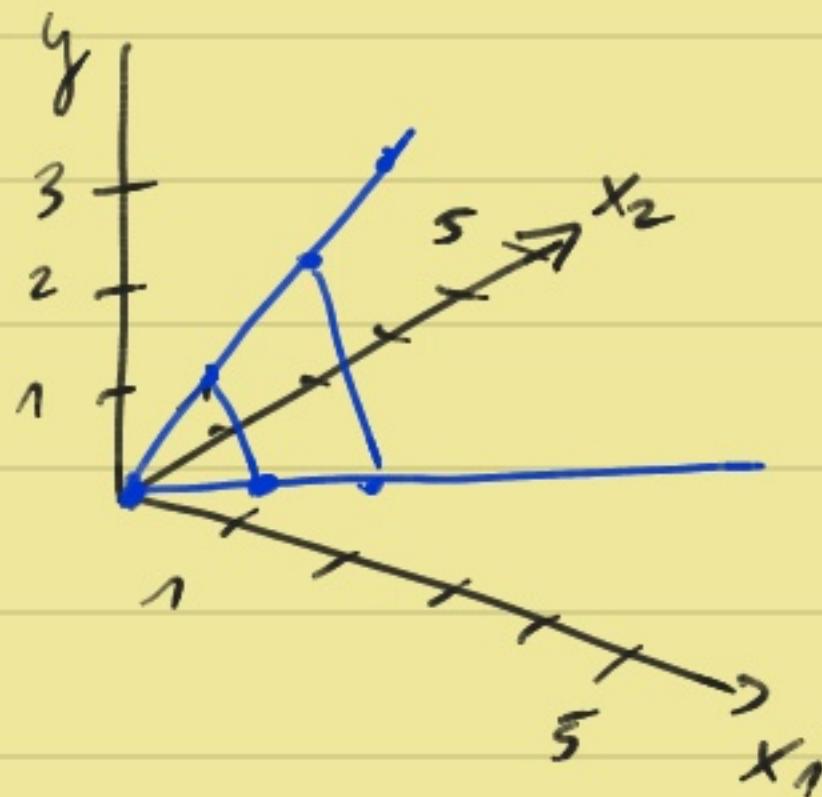
And: no.

So there's no way around it, I need to turn to multidimensional approx in full.

Actually, I think my differentiation of scalar-valued or vector-valued functions was mis-

leading. Why? B/c consider the simple

$$y = f(x_1, x_2) = x_1 + x_2$$



This is a scalar-valued function: for any (x_1, x_2) pair, it outputs a scalar. But Matlab's spline and pchip can't handle this kind of sim either!

\Rightarrow

Judd, Numerical, Alg 6.4 p 243 Mac has a
chebyshev approx in \mathbb{R}^2

\rightarrow doesn't quite work b/c I'd like to eval the coeffs

in one step, w/o evaluating Judd's step 4.

Can't figure out how to write the double sums as matrix mult. Also, even if it was working,

I'm doubtful whether it is helpful when you also need to be doing GH quadrature b/c GH quadrature determines the nodes, so they can't be the Cheby zeros.

Reading Weiser & Tarantello 1988 12 May 2020

to understand ndim-simplex.m

- Piccinis multilinear interpolant \rightarrow extension of the piccinis bilinear interpolation on rectangles
- Piccinis linear interpolant \rightarrow a first-degree multivariate B-spline interpolation
 \Rightarrow standard tools in multivariate approx theory and in simplex methods for finding fixed points & sols to nonlinear eqs.

An N -dimensional rectangular table interpolant is an approx $F(x_1, \dots, x_N)$ to a function $f(x_1, \dots, x_N)$ which is computed from table values of f :

$$\{f(x_{i_1}, x_{i_2}, \dots, x_{i_N}): i_1 = 1, \dots, n_1, \dots, i_N = 1, \dots, n_N\}$$

Both interpolants first find appropriate table intervals

$$(x_{I_1}, x_{I_1+1}), \dots, (x_{I_N}, x_{I_N+1})$$

such that

$$x_{I_1} \leq x_1 \leq x_{I_1+1}, \dots, x_{I_N} \leq x_n \leq x_{I_N+1}$$

then you scale each interval to $(0, 1)$ to reduce the problem to finding an interpolant $g(y_1, \dots, y_N)$

which approximates $g(y_1, \dots, y_N)$ in the unit N -cube.

$$\Omega = \{g(y_1, \dots, y_N): 0 \leq y_1 \leq 1, \dots, 0 \leq y_N \leq 1\}$$

using the table values

$$\{g(j_1, \dots, j_N): j_1 = 0 \text{ or } 1, \dots, j_N = 0 \text{ or } 1\}$$

Piecewise linear interpolant F_L breaks the unit N -cube into $N!$ simplices of the form

$$S_{\rho(1), \dots, \rho(N)} = \{(y_1, \dots, y_N) : 0 \leq y_{\rho(1)} \leq \dots \leq y_{\rho(N)} \leq 1\}$$

where $\rho(1), \rho(2), \dots, \rho(N)$ is a permutation of the integers $1 \dots N$. The particular simplex is found by solving the coefficients y_1, \dots, y_N of the evaluation point.

F_L is of the form

$$a_0 + a_1 y_1 + \dots + a_N y_N$$

in each simplex.

The coeffs $\{a_i\}$ are defined so F_L matches f at the $N+1$ corners s_0, \dots, s_N of the simplex

Honestly I'm not sure how much smarter I got. What I'm taking away is that multidimensional approximation is (sort of?) triangulation. In the 3D case, it's like putting triangles on the surface and making sure that the edges, not just the corners,

like up. In higher dimensions, the triangles are simplices.

But I can't believe that Matlab doesn't have a program to give you the coefficients back!

That's it. I'm installing the Curve Fitting Toolbox.

Peter meeting

12 May 2020

- 1 • Ryan: if you input FE, can replicate TR
- 2 • ND spline → what do you use?
- 3 • Adam: 2CB / first guidance, asymmetric target, changing the target
- 4 • Job market extension? | AEA letter

1. Suggestion: supp. you had a model in which B1 applies only at $t=0$ and from then on the TR gets followed → can use the sol of that as

initialization!

→ can also go on w/ B.1 holding for $t=0, 1$,
etc...

Another way:

replace A_3 w/ a

$\text{param} \cdot \text{times } A_3 + (1 - \text{param}) B.1$.

→ Initialize w/ sol of Mat!

Moving the jumps into a continuum.

2. Open up the Matlab intwlt codes

3. All is interesting. Several steps are needed to
get there.

I'm trying what Peter suggested.

13 May 2020

I use as residual $a \cdot TR + (1-a) \text{andTC}$,
and I keep reducing a .

What I find though is that for $a \leq 0.985$, you
can't solve, whether you initialize adaptively or
not. If you initialize adaptively, it gets much
closer to a solution though.

The value of $\frac{\partial x}{k}$ is crucial: the higher $\frac{\partial x}{k}$, the
bigger $\frac{\partial x}{k}$, and the bigger the TC residuals.
But even so it won't solve - for any TC.

So what I'm trying now is to write the EX-term
in the andTC using GLL quadrature. If my
understanding is right, then I need to take
a grid for $s_1 = r^n$ and $s_2 = u$, say for simplicity
5 gridpoints: $n_s = 5$ and evaluate the GLL-nodes

and weights for these two shocks.
Then for each gridpoint (\mathbb{R}^2), simulate the
model to evaluate all the endogenous variables
and get $\bar{X} = \text{weighted mean using the GTI weights}$.
The problem is that for each s_i , I'd need
to evaluate all the future paths, so for n_s nodes
I'd have $(n_s^2)^{\# \text{ of future periods}}$ paths to estimate.

→ unless there's a nice shortcut I can take.

But in this sense, it's almost easier to simply
simulate N future paths and take the cross-sectional
averages as stand-ins for the expectations.

- ask Ryan about this
- ask Ryan about $\gamma \sim \text{iid}$ → Can't ass $\gamma=0$ on avg?
- Another problem is the fact that the simulation -
based evaluation of the anchTC makes the future
problem non-square b/c you need to input
sequences of length $T + \# \text{ of future periods}$.

- Approx methods seem to be used to solve non-linear ODEs numerically → is there a way I could solve the Ramsey problem numerically?

Judd, Numerical, Nonlinear equations solving

- Bisection
 - Newton's method
 - Secant Method (1D-case)
 - Fixed-point iteration
 - Gauss-Jacobi
 - Gauss-Seidel
 - Fixed-point iteration
a.k.a. function iteration (\rightarrow VFI?)
- } 1D methods
- } Multivariate
methods

\hookrightarrow Newton's method

\hookrightarrow Broyden (ND secant)

- continuation methods / homotopy

- Simplicial homotopy methods
eg. piecewise-linear homotopy

My feeling is that parametric VFI in N-dim
is like a simplicial function iteration or stg.

Ryan meeting • materials 29

13 May 2020

1. initialization
2. $\gamma \sim \text{iid} \rightarrow E(\gamma) = 0?$
3. non-square TC problem
4. E in TC $\rightarrow G+1?$

5

- When $\gamma=0$, what does the model converge to and how does that depend on p .
- Ryan would

$$k_{t+1} = \frac{k_t + 1}{\gamma + k_t^2}$$

Homotopy:

solve an easy problem and then advance

$$\hookrightarrow (1-\alpha) + \alpha$$

\hookrightarrow Does the problem get mitigated if ρ or γ are small/zero?

Is there some inherent incompatibility w/ (4)
and (B.1)

\hookrightarrow try swapping k_+ for FE

- Try choose i, k and f_k

3. "parametric $E(\cdot)$ "

• take simulation \rightarrow approx of DGP

• regress future realizations on current states

Christiansen \rightarrow email & App.D. of Ryan & Ozge
Market Akinti

Susanto:

- Ryan told him Farouk's complaint "not a structural model of $E(\cdot)$, so Lucas critique, how can we do policy?"
- Susanto agrees w/ Ryan
- Susanto said
 - if policy is pretty good, it should implement stable outcomes so in eqb it doesn't give agents reason to revise $E(\cdot)$
 - ↳ meaning they don't revise their E-formation procedure

So the point is that Susanto agrees w/ Ryan that the responses Ryan gave to Farouk's objections are a good defense to such objections.

2 main
approaches
to solve
nonlinear
DSGEs

parameterizing the policy & multiplier functions

→ essentially approximating them; their main approach is w/ Cheby polynomials & using Gauss-Chebyshev quadrature

parameterizing the conditional expectation

L.i.l. approximating!

Marlet's: " e " → see Eqs 8 & 12

Wright & Williams': " v " → see Eq. 14.

⇒ Focus on PEA (parameterized expectation algorithm)
for ℓ , using 2 approximation approaches

Marlet's nonlinear regression approach ["conventional PEA"]

Judd's 1992 exp. fit. approx. approach

Goal: $\hat{e}_a(k, \theta)$
 \uparrow k technology (today)
 \uparrow capital

$$\theta = -\beta, \beta \text{ (bivariate)}$$

Conventional PEA:

$$\hat{e}_a(k, \theta) = a_\theta' P(\gamma(k)) \quad \text{for } \theta = -b, b \quad (31)$$

$N \times 1$ vector of params
to be solved for

$$P(x) = [P_0(x), P_1(x), \dots, P_{N-1}(x)]'$$

where $P_i(x)$, $i = 0, \dots, N-1$ are Legendre polynomials on $[-1, 1]$.

- $\gamma(k)$ is defined in Eq (21) \rightarrow the zeros of Cheby.
- $a = [a_b' \ a_{-b}']'$ $2N \times 1$ (Repeat (31) for both values of θ , the tech should)

Alg.

0. Generate an $M+1$ sequence $\{\theta_0, \theta_1, \dots, \theta_M\}$ using rng.
Formulate an initial guess for a ($2N \times 1$)

1. Compute $\{k_0, \dots, k_{M+1}\}$ recursively from
 $k_{t+1} = \hat{g}_a(k_t, \theta_t)$ $t = 0, 1, \dots, M$ using Eq. 30 and b_0
(i.e. using the policy given the approximated expectation)
and simulate $m_{t+1} = m(\hat{g}_a(k_t, \theta_t), \theta_{t+1}; \hat{g}_a, \hat{h}_a)$ for
 $t = 1, \dots, M$ using (6).

NONLINEAR REG STEP

$\stackrel{1.2}{=}$ Find $\tilde{\alpha}$, the new value of α , as the solution to
the nonlinear least-squares regression problem

$$\tilde{\alpha} = \underset{\text{at } \mathbb{R}^{2N}}{\operatorname{arg\,min}} \frac{1}{M} \sum_{t=0}^{M-1} [m_{t+1} - \exp(\hat{e}_\alpha(k_t, \theta_t))]^2 \quad (32)$$

Denote the mapping from α to $\tilde{\alpha}$ defined by the
above steps $\stackrel{1}{=}$ & $\stackrel{2}{=}$ as $\tilde{\alpha} = S(\alpha; N, M)$.

→ conventional PEA seeks $\alpha^* : \alpha^* - S(\alpha^*; N, M) = 0$
(i.e. iterate on it to find the fixed point)
or solve it, I know now.

- unclear: do you iterate or not? → PTO.
- "all integrals are evaluated using Monte Carlo simulation"
→ I don't see it immediately but I feel it has to do
w/ the fact that given approx. expectations, you
simulate a synthetic time series
- weighting functions emphasize (k, θ, θ') that are assigned
high probability by the model (see step $\stackrel{1}{=}$)

→ iterate or not:

p. 28 Mac: They always started by solving

$$a^* - S(a^*, N, M) = 0 \text{ numerically}$$

(using quasi-Newton w/ Broyden's second update)

instead of "successive approximation"

↳ by which they maybe mean
iterating on it.

→ I think that this is related to why in P56 Ryan

lets $a = \operatorname{argmin}_i \frac{1}{2} \| \cdot - E[\text{rhs}_i] \|^2$ instead
of iterating on $\|V^n - V^{n+1}\|$ or $\|U^n - U^{n+1}\|$.

Am I right in thinking that this probability to zero

out the residuals only is there if you do parametric

VFI/PFI? → BIC conceptually, iter on a

and $a = \text{fsolve}(\cdot)$ is just saying "find a

such that $\hat{V}(a, \cdot)$ is a good approx". W/o

parametric/approx/integ approach, you can't
write this functional fixed point problem.

But: Chebyshev PEA seems to be their preferred method.

Do they only do Chebyshev PEA w/ collocation?

⇒ PEA collocation is the fastest

Ok: What's the diff between the Chebyshev PEA & PEA collocation? What's the diff b/w collocation and Galerkin?

collocation & galerkin are the two schemes for the choice of weighting functions

collocation: Dirac delta functions

Galerkin: The weights of the polynomial family that's the basis.

Chebyshev PEA uses Chebys as basis instead of Legendre. It has 2 sub-categories

PEA collocation: Cheby basis, Dirac weights

PEA galerkin: Cheby basis, Galerkin weights

I think one subtle point that might solve some of these

tensions is that if $M=N$, then X is square and Galerkin reduces to collocation.

($M = \# \text{gridpoints}$, $N = \# \text{of basis polynomials } T_0, \dots, T_{N-1}$)

Reddy 1992

Galerkin method: p. 4, 54, 57, 581-586

Collocation method: 16, 40, 55-57, 157, 581

→ p. 54: Galerkin chooses the weight function ψ_i equal to the approximating function ϕ_i .

→ p. 55 Collocation seeks to zero out the residual at N selected points $\mathbf{x}^i = (x^i, y^i)$ $i=1, \dots, N$

so that $R(x^i, y^i, c_j) = 0 \quad i=1, 2, \dots, N \quad (2.13)$

This amounts to weighting function $\psi_i = \delta(\mathbf{x}-\mathbf{x}^i)$

where $\delta(x)$ is the Dirac delta function, def'd by

$$\int_R f(x) \delta(x - \xi) dx dy = f(\xi)$$

When the general notation of the problem is:

"In the finite element method, we seek an approximate solution u to a differential equation in the form

$$u \approx \sum_{j=1}^n u_j \psi_j + \sum_{j=1}^m c_j \phi_j$$

values of u at the element nodes interpolation functions
nodeless coefficients approximation functions

Judd, Numerical

Galerkin method 373, 380-61, 590-92, 594-96

Collocation method 373, 384, 385

At 369, Judd begins a new chapter, Chapter 11,
Projection methods for functional equations:

a general, regression-based method to solve problems w/ unknown functions.

Example to make it clearer:

Here's an ODE: $y' - y = 0$ $y(0) = 1$. (11.1.1)

It has as solution $y = e^x$. We can use projection methods to solve it over the interval $0 \leq x \leq 3$.

Finite-difference methods' approach to solving this problem: approximate y at a finite number of specified points $x_i \rightarrow$ get (x_i, y_i) , then solve a finite set of eqs. linking the $x_i - y_i$ values.

Projection methods' approach: find a function that approximately solves (11.1.1).

1. Formulate (11.1.1) in a functional way. Def L

$$Ly \equiv y' - y \quad (11.1.2)$$

L is an operator: it maps functions to functions.

→ View (11.1.1) as the problem of finding a C^1 function y that satisfies the initial condition $y(0) = 1$ and is mapped by L to the zero function.

2. On the PC, we need to represent functions as an appr,

So we use

$$\hat{y}(x; a) = 1 + \sum_{j=1}^n a_j x^j \quad (11.1.3)$$

|
coefficients

3. the problem reduces to finding a such that

$\hat{y}(x; a)$ "nearly" solves (11.1.1).

To assess this, we compute a residual function

$$R(x; a) = L\hat{y} = -1 + \sum_{j=1}^n a_j (jx^{j-1} - x^j) \quad (11.1.4)$$

A projection method adjusts a until it finds a "good" a which nearly zeroes out $R(x; a)$.

Projection methods differ based on how they form a "good" & "nearly".

- least-squares projection:

$$\min_a \int_0^3 R(x; a)^2 dx \quad \text{for (11.1.1)}$$

and $n=3$, so

$$a = a_1, a_2, a_3.$$

- subdomain projection
- method of moments projection
- Galerkin projection

$$0 = \int_0^3 R(x, a) \times x^j \, dx \quad j = 1, 2, 3$$

→ a weighted-residual scheme in which the same polynomials which were used in the construction of the residual $R(x, a)$ are now used as weights, x^j .

- collocation methods

Do not compute integrals of the residual fit
(do not compute weighted-sums of $R(x, a)$).

Instead, they choose a such that $R(x, a) = 0$ at a finite set of x values.

In this ex, choose 3 collocation points: $0, \frac{3}{2}, 3$

→ 3 eqs in a_1, a_2, a_3 which we solve.

The point is that the choice of points matters: Cheby
nodes is a good choice. ⇒ Chebyshev collocation.

Ok, I'm deep in the rainforest of approximation.

I want to clean up some issues:

- when $M=N$, Galerkin boils down to collocation.

Why? B/c Galerkin does

$$a = \operatorname{argmin}_a \int R(x, a) x^j dx \quad j=1, \dots, N$$

where $N = \# \text{ of coeffs } a$ (in the ex before, $N=3$)

On the computer, we approx this integral as

$$\sum_{i=1}^M R(x_i, a) x_i^j \quad \hat{\Sigma} : \text{S } \text{I guess}$$

on M quadrature nodes. If $M > N$, then we have an overdetermined/overidentified system (as in the method of moments), and so $a = \operatorname{argmin}_a$

When $M=N$, $a = \text{sol to the exactly identified system } R(x_i, a) x_i^j$, i.e. in this case a -coeffs zero out the residuals \Rightarrow collocation.

✓

- It seems like we're going in circles b/c whatever I do, I run into (-party - the same) approximation ideas.

The problems one can/wants to tackle:

1. Compute anything w/ continuous/infinite support
e.g. $E(\cdot)$ or just integrals in general or even just a function w/ continuous support
 \Rightarrow numerical integration or functional approx.
2. Solving diff eqs or ODEs \Rightarrow solving "functional eqs" or "functional problems"
 \rightarrow we're looking for a function that's the sol to the problem.
 \Rightarrow Solving econ. models w/o linearizing them often incorporates both of these elements:
 e.g. solve for policy functions in the optimal growth model w/ capital (deterministic)
 or -!!- w/ capital & tech shocks (stochastic)

⇒ numerical approx ideas can come to bear in solving this problem, which is a nonlinear diff eq / ODE system, if stochastic then w/ a (conditional) expectation splicing up the problem.

there are several things you can do:

- value function iteration where you approximate π^* (the value function or policy fit), in the least as a step function (if you discretize), or you interpolate using Cheby or spline or whatever to have a better approx (not step fit).
- Now my thinking is that you don't need to write the sol of a DSGE as a value function form i.e. a dynamic programming form - instead, you can write down the functional problem form and approximate π^* : a policy fit and/or multidim fit (as done in Ryan, PS6 !) or policy fit & multiplier approximated indirectly by approximating the expectations

as done in Christiano & Fisher 2000

the key to projection methods is to formulate a residual function to the functional eq and drive that close to zero.

⇒ So Ryan's PSB is a projection method solution to the optimal growth problem, in fact, w/ collocation. That's why he doesn't iterate! And that's why Christiano & Fisher don't iterate, unless they have to. I guess you could; it's just a less efficient way of solving the functional eq.

Note: dynamic programming & projection methods are thus both methods that use theories from numerical approx to solve nonlinear ODE-systems. They are global methods. Vs. perturbation, which obtains the linear approx of the policies around the st. st.

Parameterized expectations: a particular example of projections

VFI / PFI : interpreted as an iterative procedure to implement a particular projection method
Pablo prefers to think of it as a different family of problems, though.

⇒ This is all enlightening b/c projections & VFI/PFI are all examples of what Pablo calls the "functional equation approach" (= global methods) which is also an enlightening term b/c it reveals that it's all about the shared perspective of solving a functional equation using approximation theory.

⇒ Let's turn to my own problem and put it in the context of this illuminating discussion.

My general problem is

$$\min \sum_t (\bar{\pi}_t^2 + \gamma x_t^2) \quad \text{s.t. model eqs A1-A8 (w/o A3)}$$

this is a perfect analogue to the optimal growth model where you can write the problem as a value function recursive formulation, or you can take the derivatives of the Lagrangian and just analyze FOCs (which include model equations).

→ Two approaches

- Value function iteration

$$L(k, \bar{\pi}, s) = \min_u u(a, k, k, \bar{\pi}, i) + \beta E L(k', \bar{\pi}', s') \text{ s.t. A1-A8}$$

→ put a grid on s , maybe compute $E(\cdot)$ using GII-quadrature, interpolate $L(k', \bar{\pi}', s')$ and iterate on the coeffs a .

- Projection
 - compute FOCs (we have 'em). Choose a grid for the states.

Using that grid, generate initial policy values (*i*: $\{s\}$ guess) for each grid point using g_x from the RE model, $\{s\}$ guess.

- Approximate $H(x)$ w/ coeffs a such that $\hat{h}_i(a) \approx H(x)$. where $X = \text{state vector}$, H is the policy function for i we're looking for.
- Using the coeffs a , compute the residuals to one of the equations; solve a such that the resids are zero.

⇒ My conclusion is that both approaches are (in principle) feasible. Moreover, what I'm doing now ("guess i -sequences (i.e. the policy function), plug 'em and compute errors, zero 'em out") is like the projection method except it's not making use of any approximation anywhere. It didn't need to, so far, b/c this method can handle nonlinearity

(which is, to be sure, present in the model equations too, not just in the TC), and so far there was no ugly irregularity (like expectations, integrals, etc) to compute. Now that there's an expectation in the and TL, now Ryan is suggesting parameterized expectations: of course, just add the logical next step to projections: the approximation of whatever needs to be approximated.

Ok, I see more clearly now. My two alternatives are
parameterized VFI

· parameterized expectations (projection w/ utilization)

⇒ in each case we need some kind of N-dimensional approx: Chebychev or spline or linear interp/triangularization
(spectral) (finite element)

I need to get those codes to work. Now the only thing I can't

understand is: what's this "synthetic time series" and regression business Ryan is talking about in Akinci & Chalwoux?

Again, there are several things that don't match up.

1. In our meeting, Ryan suggested forming \bar{X} as the fitted value of a regression of x on the past histories.
2. In Akinci & Chalwoux, that's not exactly what happens. First, they explain what a standard parameterized expectations approach would do (which is more or less what I expect it to be given PS6 & my discoveries today about projection). Second, they explain that they don't do that, except at the very last period. Instead, they do what I was thinking (or dreading) to do: forward simulate everything for T periods for all possible grid values for all shocks (OMG!) and use multivariate GMM.

quadrature to arrive at time t expectations.

Frankly Ryan doesn't think I can/should do that?

→ it's exactly the thing that would not only be huge, but also make my residuals non-square.

↳ You thought for a second that Galerkin could save you w/o, didn't you? But no, you'd have more unknowns than residuals, not vice versa. Sorry!

⇒ so let's zoom in on points 1 & 2:

What reconciles them is that in Klein & Chaloupka, they say that std parameterized expectations would:

1. Conjecture an initial expectation as $\hat{E}_t^i \approx \beta^i s_t$
2. Solve model equations given $\hat{E}_t = \beta^i s_t$ $\xrightarrow{\text{coeff}} \uparrow \text{basis}$
→ obtain a history $\{v_t\}$.
3. Compute realized expectations E_t given $\{v_t\}$.
4. Update β as $\beta^{i+1} = \text{reg } E_{t+1} \text{ on } s_t$

Ok so Ryan's at least consistent w/ himself. But it didn't feel like this is what Christians & Fisher did! What's the diff? It's the basis function.

- In Christians, it's Legendre, then Cheby
- In Marini, it's s_t (erm... I think.)

Oh wait! s_t isn't the states! It's a basis that spans the states: It's $s_t(x_t)$, and it's an "extended state space":

x_1, x_1^2, x_1^3 and all 2nd & 3rd order cross-terms, or actually, only a subset b/c using the full set would lead to multi-collinearity.

- But going back to Christians, I see how that also their, regression is going on. In fact, it's the same idea. For a guess a_0 , basis of $\hat{e} = a_0 \underbrace{P(p/\epsilon)}$

1. Given \hat{e} , compute simulated histories $m_{t+1} \underset{t=1}{\overset{M}{\text{Legendre}}}$

2. Find $\tilde{\alpha} = \beta^{OLS}$ of reg m_{t+1} on $\exp(\hat{e}(a))$
(And you can iterate, or fsolve.)

→ may also run into multicollinearity issues.
PCA gets around this in a way I don't quite get
since it still has a regression step, except that seems
a little more compact. It is

$$\tilde{\alpha}_\theta = (X'X)^{-1} X' Y_n(\theta) \quad M \times N$$

where I think X is just $X = [T(r_1) T(r_2) \dots T(r_m)]'$
(I don't think this is multiplication)

and $Y_n(\theta)$ I do think is a simulated history or stg.

Before doing spectral or finite element 15 May 2020
approximation in ND (i.e. ND Chebyshev or ND simplices),
let me try to parameterize my expectations using
the extended state space as a basis, like Ryan
does. While that involves the choice of which cross-terms
to use, it still may be easier as a first pass than
having a polynomial basis. Maybe I can even ignore
cross-terms as a first pass?

Let's try to match up my model w/ conventional PEA:

1. Conjecture an initial expectation as $\hat{E}_t^i \sim \beta^i s_t$

For me, $x_t = [k_t, \bar{\pi}_{t-1}, r_t^n, u_t]$, so $s_t(x_t) = x_t + x_t^2 + x_t^3 + \text{cross-terms}$

If I ignore cross-terms, I have $n_x \cdot 3 = 12$ elements. $\beta^0 = \text{ones}(1, 12)$.

2. Solve model equations given $\hat{E}_t^i = \beta^i s_t \rightarrow$ obtain a history $\{v_t\}$.

For a simulation and β^0 , I can compute my $\hat{E}x_t$ at each t as

$\hat{E}x_t = \beta^0 s(x_t)$. \Rightarrow Then I can eval the anchTC for each t.

I think I need to zero out the resids now to arrive at a realized v_t .

3. Compute realized expectations E_t given $\{v_t\}$.

I think given the realized history $\{v_t\}$ I compute "realized" $\hat{E}x_t$ as applying the formula at each t using $\{v_t\}$.

4. Update β as $\beta^{t+1} = \text{reg } E_{t+1} \text{ on } S_t$.

Now I have realized E_t and I can reg it on S_t at each t.

Repeat until β converges.

16 May 2020

$$\beta = (Sx' Sx)^{-1} Sx' E$$

$k \times 1 \quad k \times T \quad T \times k \quad k \times T \quad T \times 1 \quad \Rightarrow Sx \quad T \times k$
 $k \times k \quad k \times 1 \quad E \quad T \times 1$

$$\beta \quad k \times 1$$

Problem: altho the code seems to work, the hiccup I anticipate in materials 30 seems indeed to be a problem. It'd be good to be smarter about it somehow. Need to confirm w/ Ryan that I'm doing it right.

Should first check the small things Ryan suggested:

- (ρ, δ)
- hypothesizing k & f_e

Then should go back to NDIM approx:

- Chebyshev
- ndim-simplex

Ryan's small questions : (w/o parametric E) \rightarrow comment TC

1) Does the problem get mitigated if $\gamma=0$ or p small?

~~→ Doesn't seem like it.~~ Yes! Set $\gamma=0 \rightarrow$ still doesn't solve.

Set $\gamma=0$ and $p=0.1 \rightarrow$ solved! And did so fast.

Set $\gamma=0.004$ and $p=0.1 \rightarrow$ solved!

Set $\gamma=0.004$ and $p=0.5 \rightarrow$ solved!

-11- RE-TC \rightarrow solved!

Set $\gamma=0.002$ and $p=0.8 \Rightarrow$ Now it's having trouble.

But $f(x)$ keeps \downarrow , albeit slowly, so it might solve w/ time.

Set $\gamma=0.001$ and $p=0.7 \rightarrow$ Nearly solved.

Interesting, the bigger p , the harder the solver finds it to decrease the function value. Most likely you get meandering easier in whichever location it searches.

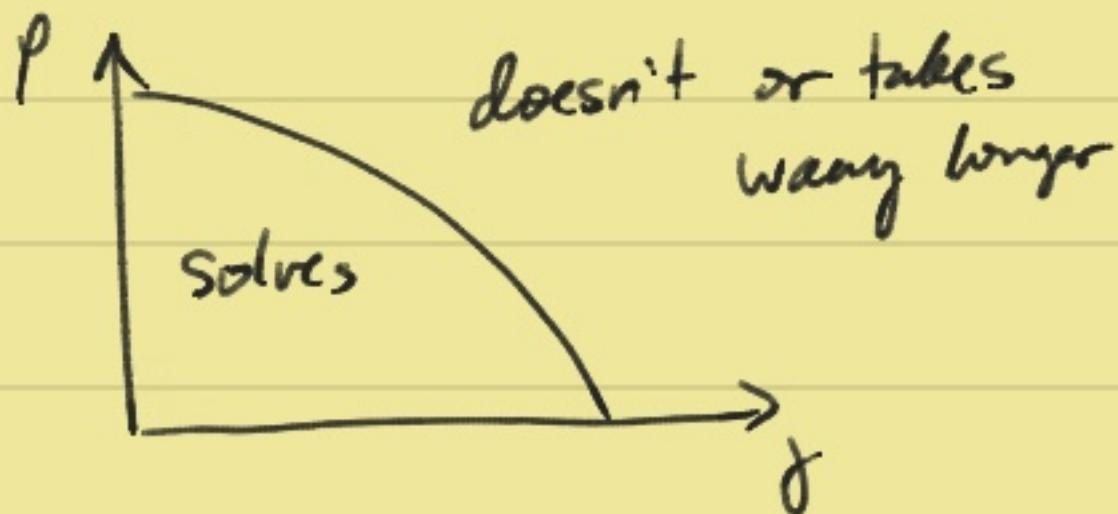
Set $\gamma=0.001$ and $p=0.6 \rightarrow$ solved!

Let's see if setting $\rho=0.1$ and γ impedes solving.

$\gamma=0.001$ solves

$\gamma=0.01 \rightarrow$ a much harder time, but solved!

\Rightarrow There seems to be a "feasibility frontier"



A value pair of $(\rho=0.5, \gamma=0.001)$ solves.

- 2) I know what the model converges to for (ρ, γ) rats
 - \rightarrow it explodes beyond a certain (broader) feasibility frontier, otherwise it converges to $\gamma \cdot E(f_t)$ or so.
 - \rightarrow Maybe this instability in convergence means that for diff input sequences, an otherwise stable (ρ, γ) are unstable.

⇒ To do:

- parameterized E w/ $\rho=0.5$, $\gamma=0.001$
- Chebychev & Ndim simplex
- VFI

Peter meeting

15 May 2020

IM and Christiansen & Fisher 2010

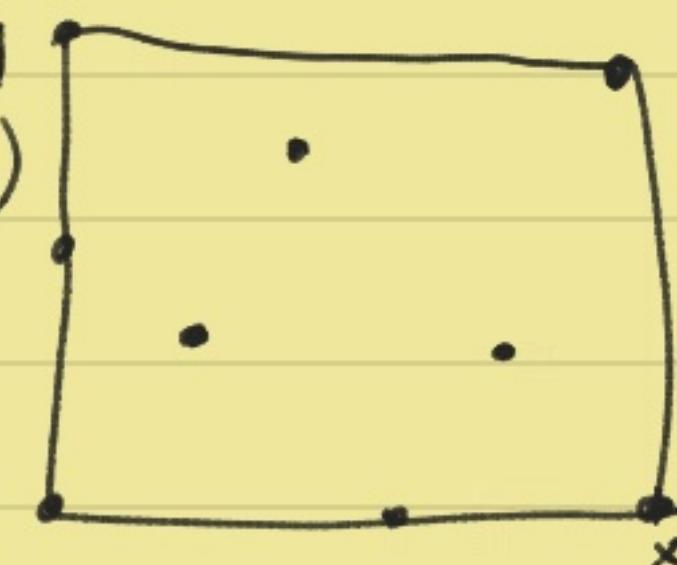
Now I wanna get the 2D interpolation 20 May 2020
right

Reading Berg et al 2008, Triangulation

Triangulation seems to be an approx method of 3D functions (terrains). Idea: you have a set of given function values $f(x, y)$ on a 2D grid. You split the 2D grid into triangles whose vertices are the sample points $f(x, y)$. We then lift each $f(x, y)$ to its correct height and connect the dots (linearly).

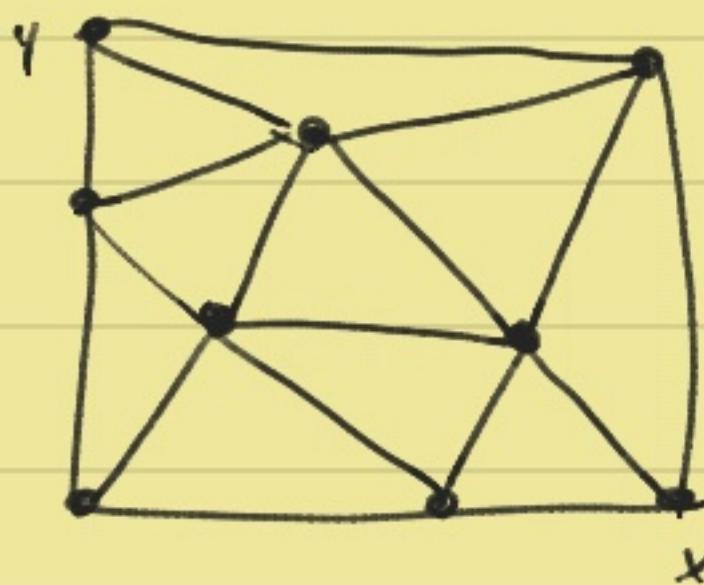
2D grid of surface:

(3 inner sample points)



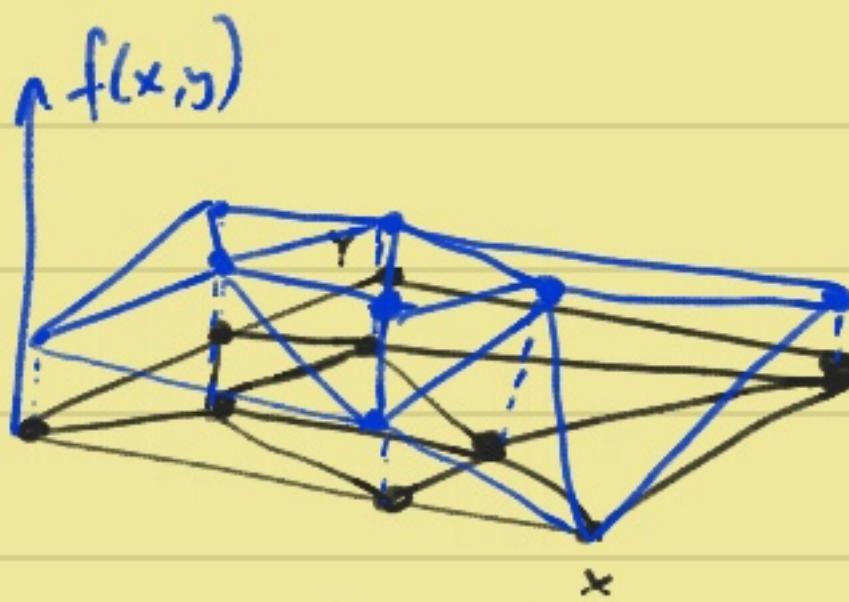
A triangulation:

(One of many possible
ones)



Polyhedral terrain

(the approximation
via raising the points)



Honestly, I think this is exactly what NDIm-Simplicial
interpolation does!

Berg et al p. 10 (Mac) nicely describes how to
compute the triangulation. This seems to be at odds

w/ the interpolation approach where we start out w/ a given grid of interpolation nodes. It seems like the std interpolation is just one "superficial" triangulation of that grid, and I guess it's not as good as the Delaunay triangulation.

Ryan meeting

22 May 2010

A plan B: not defend next summer, x1 RA job a Boston FCI & go on the market as a fresh PhD one year later \rightarrow only do in communication w/ factors.

- Mid-October is the time to decide
- First jobs will be posted in Sept. \rightarrow we can look at those to get a sense of the thickness
 \rightarrow job postings will be held off
- 3rd power is higher than most people need.
1st & 2nd and all cross-products normally fine.
(i.e. w/o multicoll.)

When you get a programmatically truncated sample, you fill for the future means from the sample! \rightarrow that way you avoid trends from the $E(\cdot)$ that are there b/c of the truncation.

\rightarrow but higher orders can often lead to instability due to multicoll. A way to choose between orders and constraints is multicollinearity tests.

Bob King meeting

22 May 2020

Fabio Milani: positive implications of TV gains

\rightarrow 1 do normative implications

Ramsey problems have 2 elements:

1. what is the structure of the CB's problem?

p. 15

2. there's a hierarchy of how to proceed:

1. what's the optimal (π, x) process
 2. how to implement that using i or FX rates or whatever policy rule
 3. In RE, Ramsey is hard b/c of fwd-looking constraints \rightarrow think re Ramsey problem in a recursive form w/ commitment
 \rightarrow he suspects this EL(.) doesn't have time-inconsistent
- Positive implications of learning models?
 \rightarrow a positive section to the paper
 - Opt. pol w/ private sector adaptive learning goes back to the beginning. Sargent
 \rightarrow Phelps problem of opt. pol.
 \Rightarrow cutting against the policy design of the last 20 yrs

Sargent is dismissive to rationalize certain

features of US $\pi-x$ movement

→ ppl might say "hasn't Sargent already told us that it doesn't work?"

→ need to be able to say how my model is diff.

- John Williams & others have explored learning models

- how do we select among these models?
 - if E-stability, then are we constraining ourselves to the transition?

→ need to be able to contrast to std adaptive expectations models
(à la Phelps)