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

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

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 fsolve 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.

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

⇒ 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 = ngrid

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

here is the problem that $(a,b,c,d)_{i-1}$ enter

But maybe the vector can really be the vec(Θ) ?

$$\underset{4n}{y} = \underset{4n \times 4n}{M} \underset{4n \times 1}{vec(\Theta)}$$

?

$$\begin{bmatrix} y_i \\ y_{i+1} \\ \\ \end{bmatrix} = \begin{bmatrix} 1 & x_i & x_i^2 & x_i^3 & 0 & \cdots & 0 \\ 1 & x_{i+1} & x_{i+1}^2 & x_{i+1}^3 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} a_i \\ b_i \\ c_i \\ d_i \end{bmatrix} \Big\} \; i = 1, \ldots, n \; (2n \text{ eqs})$$

Shall we try this w/ 3 nodes?

Natural cubic spline w/ 4 nodes (3 intervals)

$$
\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_2 \\ y_3 \\ y_4 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
=
\begin{bmatrix}
1 & x_1 & x_1^2 & x_1^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & x_2 & x_2^2 & x_2^3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_3 & x_3^2 & x_3^3 \\
1 & x_2 & x_2^2 & x_2^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & x_3 & x_3^2 & x_3^3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_4 & x_4^2 & x_4^3 \\
0 & 1 & 2x_1 & 3x_1^2 & 0 & -1 & -2x_2 & -3x_2^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 2x_2 & 3x_2^2 & 0 & -1 & -2x_3 & -3x_3^2 \\
0 & 0 & 2 & 6x_1 & 0 & 0 & -2 & -6x_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 6x_2 & 0 & 0 & -2 & -6x_3 \\
0 & 0 & -1 & -2x_1 & -3x_1^2 & 0 & \cdots & & & & 0 & \\
0 & 0 & \cdots & & & & & & 0 & -1 & -2x_4 & -3x_4^2
\end{bmatrix}
\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ a_2 \\ b_2 \\ c_2 \\ d_2 \\ a_3 \\ b_3 \\ c_3 \\ d_3 \end{bmatrix}
$$

12×1 $\quad$ 12×12 $\quad$ 12×1

Y $\quad$ M $\quad$ $\text{vec}(\Theta)$

$*_1$ derivs at inner nodes $\quad$ $*_2$ natural spline: derivs at ends

Now I just have to figure out a good way to create the matrices. vec(θ) is fine.

$Y = \underset{(4n,1)}{\text{zeros}}(4n, 1)$ Recall: $m=4, n=3$

for $i = 1, \ldots, n$

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

$i = n+1, \ldots, 2n$, $j = 1$

$j = j+1$

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

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

end. Y done!

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

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

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

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

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

if $i=3$ $12-3 : 12 \Rightarrow 9:12$

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

end. this has the first 6 rows.

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

$j = j+1$ (the first derivs 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)

$j=1$ $1:8$

$j=2$ $5:12$ ok!

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

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 7 May 2020
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 consen-
sus expectation 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 consen
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 $d=1$, $c=0.5$,
$k_t = k_{t-1} + d - c(fe^2)$ 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

$$k_t^{-1} = \frac{1}{k_{t-1} + d - c\,fe^2}$$

Exactly, this gives you exactly the same thing.

W/ different coeffs

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

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

or for Alt 4: $k_t = k_{t-1} + d - c\,fe^2\, k_{t-1}^{-1}$

↳ this ain't cool, dampens too much.

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

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

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

Annoying: if you set $\gamma = 0.001$, still learn, 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 param 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 rock it out of appearing to be done 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 $fe < 0$,

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

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 $\log(fe^2) = 2\log(fe)$ ah but it's taking the positive root. But then you could also do $\text{sqrt}(fe^2)$.

Problem is that it's pretty unstable wrt parameter values b/c they determine whether the gain ↓ or ↑ after a fe.

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

Solvers' stopping criteria

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)

StepTolerance / 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: $\sum_i res_i^2$ ('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-Marquardt.

Recall that this is the scalar · eye you add to $s^k = -(J(x)'J(x) + \lambda I)^{-1}(\nabla f(x^k))'$ to estimate the Hessian when determining the step size & direction. (Notes 10, p 67)

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

When $\lambda \to \infty$, 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 $\lambda_{k+1} = \frac{\lambda_k}{10}$.

When the step is unsuccessful, the algorithm sets $\lambda_{k+1} = \lambda_k \cdot 10$.

Note: an internal default in fsolve

opttol = 1e-4 · TolFun

Since I set TolFun to 1e-9,

opttol = 1e-4 · 1e-9 = 1e-13

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

What I'm finding is that inputting 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 inputting an exog k doesn't dampen fe and it still allows recursive fe's to happen.
→ this is why I'm now inputting fe'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 priors the TR, NKPC, NKIS as residuals for the inputs π, x, i, fe, 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 input i & fe 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. dgain learning) then it solves when inputting a, x, i, fe for resids NKPC, TR, NKIS.
If I also input as resid the LOM gain, it seems to have difficulty solving even for this dgain specification

Ah shit - 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.

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

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

$$g_\pi = 2\gamma fe_{t+1|t-1} \cdot \frac{\partial fe_{t+1|t-1}}{\partial \pi_t} = 2\gamma fe_{t+1|t-1}$$

$$g_{\bar\pi} = 2\gamma fe_{t+1|t-1} \cdot \frac{\partial fe_{t+1|t-1}}{\partial \bar\pi_{t-1}} = -2\gamma fe_{t+1|t-1}$$

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

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 $\frac{(1-\alpha)\beta}{1-\alpha\beta}(k_t^{-1} + fe_{t+1|t-1} g_{\pi_t})$ is ≈ 0 for all t.

Of course this doesn't solve either.

$\frac{(1-\alpha)\beta}{1-\alpha\beta} = 0.9083$ → so it's really $k^{-1} + fc \cdot g_\pi$

that's small, in particular k^{-1} & g_π

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

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

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

What I've now done is I've taken the fsolve 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) 9 May 2020

⇒ 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 fct value has barely changed and the first-order optimality is on the order of e+05. 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 nonzero elements everywhere.

Recursive build-up of errors:

doesn't happen!

Instead, the early part of the sample has huge resids for the TC, but not the late one.

It seems like the TC resids are much bigger

If you don't input x & TI, even the SSR seems bigger.

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

$$E_t \sum_{i=1}^{\infty} x_{t+i} \prod_{j=0}^{i-1} \left(1 - k_{t+1+j}^{-1} - fc_{t+1+j\,|\,t+j}\, g_{\pi,t+j}\right)?$$

Let's refer to this quantity as Ex. 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 $\varepsilon \sim iid\, N(0, \Sigma)$

$\Rightarrow$ fe is N.

$\Rightarrow$ le is a function of Ns

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

$\Rightarrow$ Gauss-Hermite quadrature.

Gauss-Hermite quadrature, Judd, Numerical, p. 266 Mac, (7.2.11)

$$E\{f(Y)\} = \frac{1}{\sqrt{(2\pi b^2)}} \int_{-\infty}^{\infty} f(y)\, e^{-\frac{(y-\mu)^2}{2b^2}}\, dy$$

n = #nodes

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

ω_i: quadrature weights; x_i: quadrature nodes

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

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

↳ e.g. Table 7.9 p. 266 Mac

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

1) Use GH-quadrature to determine the optimal grid w/ weights for the tech shock (5 nodes) and the endog state (kapital) individually (25 nodes)
2) Use <u>ndgrid.m</u> (in-built) to create a 2×125 matrix of each combination of capital (row 1) and technology (row 2) from the gridpoints above. ⇒ 125 points on the grid $(k, tech)$.
⇒ generate $h = H(k_i, tech_i)$ $i = 1, \dots 125$ (labor)
This is the initial policy function for hours.
↳ N.B. policy fct = jump as a function states.
3) ⇒ Generate coefficients a_i such that $\tilde{H}(a_i, X)$ 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 <u>ndim_simplex.m</u>

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, $XP_{i,j}$
- Evaluate the policy function at each $XP_{i,j}$
  - Compute the RHS of the EE and obtain the expectation as

$$\sum_{j=1}^{5} p_{\gamma_j} \, rhs_{i,j}$$

↑ I guess there must be quadrature weights? Yea baby!

- Return the resids $\frac{1}{c_i} - E[rhs]_i$ 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 EX.

Ryan's PS6 seems helpful.

Note: GH-Quadrature.m is Table 7.4 Judd!

I've gone back to my VFI 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_i to min the residuals $\hat{V}(k_p, a_i) - \hat{V}(a_{i+1})$

or sthg. Ok, I think this is what Judd meant when he said that "the fitting step can be an unweighted nonlinear least squares procedure as in

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

"

↑ what_spline (x_i, a)

So instead of obj_spline_secant_hermite in the fitting step, use obj_vhat_spline(coeffs, x_i, v_i)

which for $i = 1, \ldots, ngrid$

computes vhat_spline(coeffs, x_i)

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

SSR = sum(resid)

and sum that over coeffs!

Nope, just use Matlab's spline in the fitting step

Adam Guren meeting — 11 May 2020

- Really like the idea
- Motivation should not be 2019

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

Olivier Blanchard: "4% target", 2000s

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

→ talked about a lot in 2LB & fwd guidance
↳ "we didn't wanna unanchor!"

Then wanna show results
and lots of intuition
Wanna talk about 2LB!

Werning

- Werning: deterministic length of 2LS
  "how much Δ shall we promise?"
  → he thinks less! b/c we don't wanna unanchor!
- is the Eq asymmetric? go above 2% is really bad? Even if the CB has symmetric prefs, they will look like asymmetric functionally
- what happens when you Δ target?
- what does an estimated PC look like in this model?

→ present PC 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 pit talk to motivate anchoring
  - ↳ present evidence from data
  - and to use that to motivate how you model anchoring
- You need to do scenarios
  - ZLB / fwd guidance
  - changing TA target
  - asymmetric behavior even if prefs are symmetric

Work after

- Parametric VFI w/ splines 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

$$X = \begin{bmatrix} k_{grid} \\ g_{grid} \end{bmatrix} \qquad 2 \times 125.$$

⇒ 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 below 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 kapital $k = k_{grid}(i)$, 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 do try whether spline/pchip can handle that!

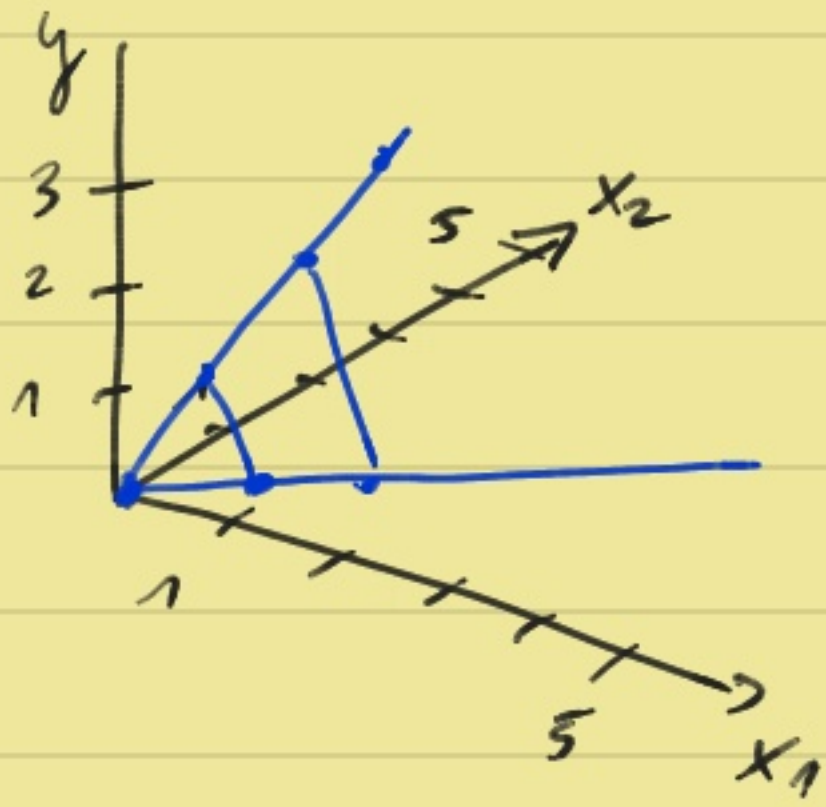
Aaand: no.

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 situation!

⇒

Judd, Numerical, Alg 6.4 p 243 Mac has a Chebyshev Approx in $\mathbb{R}^2$

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

in one step, w/o evaluating fnld'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 & Zarantonello 1988 12 May 2020

to understand ndim_simplex.m

- Piecewise multilinear interpolant → extension of the piecewise bilinear interpolation on rectangles
- Piecewise linear interpolant → a first-degree multivariate B-spline interpolation

⇒ 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:

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

Both interpolants first find appropriate table intervals

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

such that

$$X_{I_1} \le x_1 \le X_{I_1+1}, \quad \dots, \quad X_{I_N} \le x_n \le X_{I_N+1}$$

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,

$$\Omega = \{[y_1, \dots, y_N): 0 \le y_1 \le 1, \dots, 0 \le y_N \le 1\}$$

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

$$S_{p(1),\ldots p(N)} = \{(y_1,\ldots y_N): 0 \le y_{p(1)} \le \cdots \le y_{p(N)} \le 1\}$$

where $p(1), p(2) \ldots p(N)$ is a permutation of the integers $1 \ldots N$. The particular simplex is found by sorting the coefficients $y_1, \ldots y_N$ of the evaluation point. F_L is of the form

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

in each simplex.

The coeffs $\{a_i\}$ are defined so F_L matches f at the $N+1$ corners $S_0, \ldots, S_N$ 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 Mallat doesn't have a program to give you the coefficients beside!
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: 2LB / 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 TR → 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 A3 w/ a

param · times A3 + (1 − param) B.1.

→ initialize w/ Sol of Rnd!

Making the jump into a continuum.

2. Open up the Mallat invariant 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 $a \cdot TR + (1-a)$ and TC,

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 λ_x is crucial: the higher λ_x, the bigger $\frac{\lambda_x}{k}$, 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 and TC using GH 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 GH-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 EX = weighted mean using the GH weights.
The problem is that for each s_i, I'd need to evaluate all the future paths, so for ns nodes I'd have $(ns^2)^{\#\text{ of future periods}}$ paths to evaluate.
→ 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 $\gamma \sim iid$ → can't ass $\gamma = 0$ on avg?
- Another problem is the fact that the simulation-based evaluation of the arch TC makes the forward problem non-square b/c you need to input sequences of length T + # 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 equation solving

- Bisection
- Newton's Method
- Secant Method (1D-case)
- Fixed-point iteration

↳ 1D method

- Gauss-Jacobi
- Gauss-Seidel

↳ Multivariate methods

- Fixed-point iteration

  a.k.a. function iteration (→VFI?)

  ↳ Newton's method

  ↳ 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 sthg.

Ryan meeting • materials 29 — 13 May 2020

1. initialization
2. $\gamma \sim iid \rightarrow E(\gamma) = 0$?
3. non-square TC problem
4. E in TC $\rightarrow$ GH?
5.

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

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

Homotopy:

solve an easy problem and then advance

↳ $(1-\eta)$ + η

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

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

↳ try swapping k_t for FE

- Try choose i, k and fe

3. "parametric E(·)"
   - take simulation → approx of DGP
   - regress future realizations on current states

   Christiano → email & App. D of Ryan & Ozge
   Marcet — Akinci

Susanto:

- Ryan told him Jaromir's complaint "not a structural model of $\bar{E}(\cdot)$, 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 $\bar{E}(\cdot)$
>
> ↳ meaning they don't revise their $\bar{E}$-formation procedure

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

Christiano & Fisher 2000 — 14 May 2020

2 main approaches to solve nonlinear DSGEs

- parameterizing the policy & multiplier functions
  - → essentially approximating them; their main approach is w/ Cheby polyms & using Gauss-Chebyshev quadrature
- parameterizing the conditional expectation
  - i.e. approximating!
    - Marcet's: "e" → see Eqs 8 & 12
    - Wright & Williams': "v" → see Eq. 14.

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

- Marcet's nonlinear regression approach ["conventional PEA"]
- Judd's 1992 eqb. fct. approx approach

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

$\theta = -3, 3$ (bivariate)

Conventional PEA:

$$\hat{e}_a(k,\theta) = a_\theta' P(y(k)) \quad \text{for } \theta = -b, 2 \qquad (31)$$

a_θ: $N \times 1$ 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 polyms on $[-1,1]$.

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

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 $\{k_1, \dots, k_{M+1}\}$ 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).

SIMULATION STEP

NONLIN REG STEP

2. Find $\tilde{a}$, the new value of a, as the solution to the nonlinear least-squares regression problem

$$\tilde{a} = \underset{a \in \mathbb{R}^{2N}}{\operatorname{argmin}} \frac{1}{M} \sum_{t=0}^{M-1} \left[m_{t+1} - \exp\left(\hat{\epsilon}_a(k_t, \theta_t)\right) \right]^2 \quad (32)$$

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

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

P.T.O.

- unclear: do you iterate or not?
- "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 1)

→ Iterate or not:

p. 28 Mac: They always started by solving

$a^* - S(a^*, N, M) = 0$ numerically

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

instead of "successive approximation"

↳ by which they maybe mean

iterating on it.

→ I think that this is related to why in PS6 Ryan

lets $a = \text{argmin} \frac{1}{C_i} - E[\text{rhs}_i]$ instead

of iterating on $\|V^n - V^{n+1}\|$ or $\|U^n - U^{n+1}\|$.

Am I right in thinking that this possibility to zero out the resids only is there if you do parametric VFI / PFI? → B/C 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 / interp 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 btwn 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 subcategories

- 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 = # gridpoints, N = # of Cheby polynms $T_0, \dots T_{N-1}$)

Reddy 1992

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

Collocation method : 46, 40, 55-57, 157, 581

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

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

so that $R(x^i, y^i, c_j) = 0 \qquad i = 1, 2, \dots, N \qquad (2.113)$

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

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

$$\int_\Omega f(x)\, \delta(x - \xi)\, dx\, dy = f(\xi)$$

Where 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

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

- u_j: values of u at the element nodes
- ψ_j: interpolation functions
- c_j: nodeless coefficients
- ϕ_j: approximation functions

Judd, Numerical

Galerkin method 373, 360-61, 590-92, 594-96

Collocation method 373, 384, 385

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: $y' - y = 0 \qquad y(0) = 1$ (11.1.1)

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

Finite-difference methods' approach to solving this problem: approximate y at a finite number of specified points x_i → 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

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

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 appro,

So we use

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

(a: 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

$$R(x;a) \equiv L\hat{y} = -1 + \sum_{j=1}^{n} a_j \left(j x^{j-1} - x^j \right) \qquad (11.1.4)$$

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 formalize "good" & "nearly".

- least-squares projection:

$$\min_a \int_0^3 R(x,a)^2 \, dx$$

for (11.1.1) and $n = 3$, so $a = a_1, a_2, a_3$.

- subdomain projection
- method of moments projection
- Galerkin projection

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

→ 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 fct (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 Zeros 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

  $$a = \text{argmin} \int R(x,a)\, x^j \, dx \qquad j = 1, \dots N$$

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

  On the computer, we approx this integral as

  $$\sum_{i=1}^{M} R(x_i, a)\, x_i^j$$

  ↑ :S I guess

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

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

  ✓

- It seems like we're going in circles b/c whatever I do, I run into (-partly- 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
   ⇒ numerical integration or functional approx.
2. Solving diff eqs or ODEs ⇒ solving "functional eqs" or "functional problems"
   → we're looking for a function that's the sol to the problem.

⇒ 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 -II- w/ capital & tech shocks (stochastic)

DYNAMIC PROGRAMMING

PROJECTION

⇒ 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 spicying up the problem. there are several things you can do:

- value function iteration where you approximate sthg (the value function or policy fct), 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 fct).

- Now my thinking is that you <u>don't need</u> to write the sol of a DSGE as a value function form i.e. a <u>dynamic programming</u> form - instead, you can write down the functional problem form and approximate sthg: a policy fct and/or multiplier fct (as done in Ryan, PS6!) or policy fct & 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 PS6 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 either, 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.

Pablo, Lect Notes 2 p. 32

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

$$\min \sum_t \left(\pi_t^2 + \lambda x_t^2 \right) \qquad \text{s.t. model eqs}$$

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

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

→ put a grid on s, maybe compute $E(\cdot)$ using Gh-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.e. guess) for each gridpoint using gx from the RE model, i.e. guess.

- Approximate $H(X)$ w/ coeffs a such that $\hat{H}(a) \approx H(X)$. where X = 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; resolve 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 TC, 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/ collocation)

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

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

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

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

1. In our meeting, Ryan suggested forming Ex as the fitted value of a regression of x on the past histories.
2. In Aldrich & Charbonneau, 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 GH-

quadrature to arrive at time t expectations.
Surely 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 your ass, 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 Aleinci & Chahrour, 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 $\hat{E}_t = \beta^i s_t$ coeffs ↑ ↑ 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 Christiano & Fisher did! What's the diff? It's the basis function.

- In Christiano, it's Legendre, then Cheby
- In Mexico, 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 Christiano, I see how that also has, regression is going on. In fact, it's the same idea. For a guess a_0, think of $\hat{e} = a_0 \cdot \underbrace{P(\psi(\epsilon))}_{\text{Legendre}}$.

1. Given $\hat{e}$, compute simulated histories $m_{t+1}{}_{t=1}^{M}$
2. Find $\tilde{a} = \beta^{OLS}$ of reg m_{t+1} on $\exp(\hat{e}(a))$

(And you can iterate, or fsolve.)

→ they also run into multicollinearity issues.

PEA Chebyshev 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

$$\tilde{\alpha}_\theta = \left[X'X\right]^{-1} X' Y_a(\theta)$$

where I think X is just $X = \left[T(r_1)\, T(r_2) \ldots T(r_m)\right]'$ (M×N)

(I don't think this is multiplication)

and $Y_a(\theta)$ I do think is a simulated history or sty.

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 $\hat{E}_t^i \approx \beta^i S_t$

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 $\hat{E}_t = \beta^i S_t \Rightarrow$ 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 residls 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" EX_t as applying the formula at each t using $\{v_t\}$.

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

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

Repeat until β converges.

16 May 2020

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

$k \times 1 \quad k \times T \; T \times k \quad k \times T \; T \times 1$

$k \times k \qquad k \times 1$

$\Rightarrow$ Sx $T \times k$

E $T \times 1$

β $k \times 1$

Problem: although 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)
- inputting k & fe

Then should go back to NDIM approx:

- Chebypher
- ndim_simplex

Ryan's small questions: (W/o parametric E) → commit TC

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

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

Set $\gamma = 0$ and $\rho = 0.1$ → Solved! And did so fast.

Set $\gamma = 0.004$ and $\rho = 0.1$ → Solved!

Set $\gamma = 0.004$ and $\rho = 0.5$ → Solved!

-11- RE-TC → Solved!

Set $\gamma = 0.001$ and $\rho = 0.8$ ⇒ Now it's having trouble.

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

Set $\gamma = 0.001$ and $\rho = 0.7$ → Nearly solved.

Interesting, the bigger ρ, the harder the solver finds it to decrease the function value. Most likely you get wandering easier in whichever direction it searches.

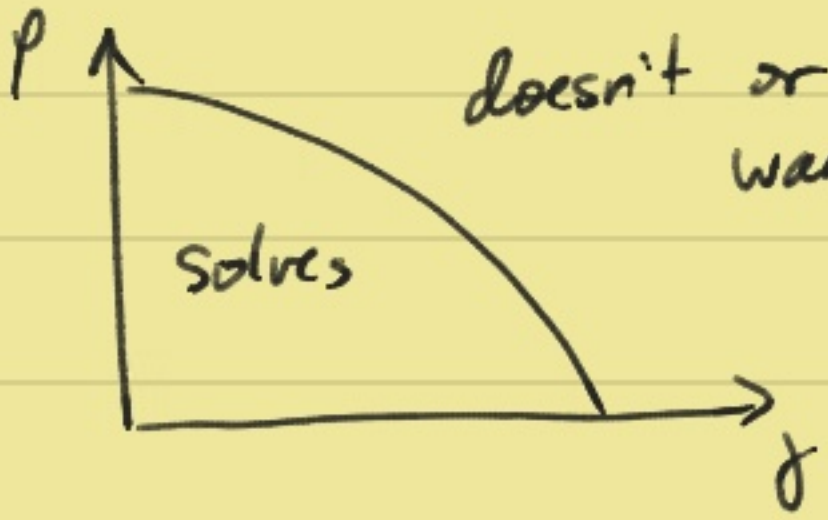
Set $\gamma = 0.001$ and $\rho = 0.6$ → Solved!

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

$\gamma = 0.001$ solves

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

⇒ there seems to be a "feasibility frontier"



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

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

⇒ To do:

- parameterized E w/ $\rho = 0.5$, $\gamma = 0.001$
- Chebyshev & Ndim_simplex
- VFI

Peter meeting 15 May 2020

11M and Christiano & Fisher 2010

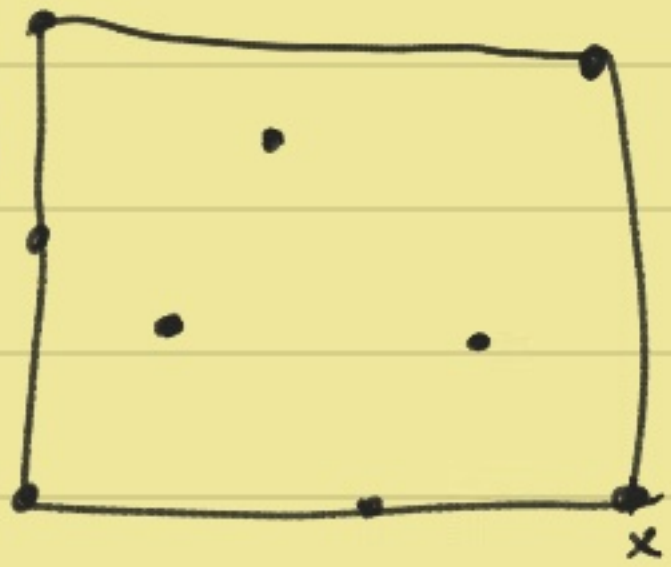
Now I wanna get the 2D interpolation 20 May 2020 night

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)

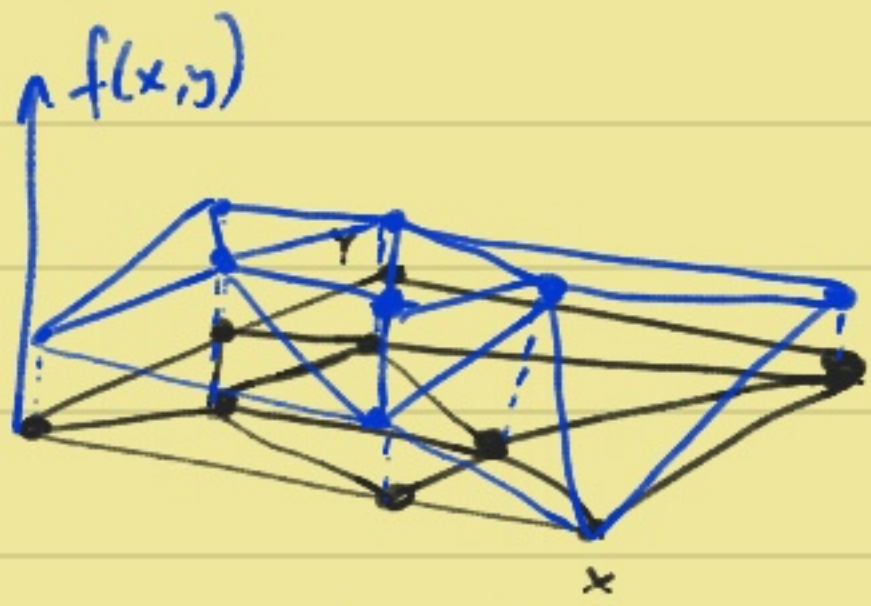


A triangulation:

(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 2020

A plan B: not defend next semester, get RA job a Boston FCU & go on the market as a fresh PhD one year later → only do in communication w) Fratrik.

- Mid-October is the time to decide
- First jobs will be posted in Sept. → we can look at those to get a sense of the thickness → job postings will be held off.

- 3rd power is higher than most people need. 1st & 2nd and all cross-products usually fine. (i.e. w/o multicoll) →

When you get a progressively truncated sample, you fill for the future means from the sample! → that way you avoid trends from the E(·) that are there b/c of the truncation.

→ But higher orders can often lead to instability due to multicoll. A way to choose between orders and cross-terms is multicollinearity tests.

Bob King meeting — 22 May 2020

Fabio Milani: positive implications of TV gains

→ I 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 → trick re Ramsey problem into
   a recursive form w/ commitment
   → he suspects this E(·) doesn't have time-
   inconsistent

- Positive implications of learning models?
  → a positive section to the paper
- Opt. pol w/ private sector adaptive learning
  goes back to the beginning. Sargent
  → Phelps problem of opt. pol.
  ⇒ cutting against the policy design of the
  last 20 yrs
  Sargent is dismissive to rectionalize certain

features of US π-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)