Numerical Methods

Jesús Bueren

EUI, 2024

Introduction

- In the lecture we are going to give an overview of different methods for solving recursive problems (not only life cycle).
- In the first part, we are going to cover some tricks to speed up and increase accuracy of the solution when using state-space methods.
- In the second part, we will see projection methods.

Discrete Approximation

Simple Iterative Procedure

- Take the problem faced by an agent in the life cycle model 2 periods before certain death:

$$v_{J-1}(a) = \max_{c,a'} \{u(c) + s_j \beta v_J(a')\}$$
$$\text{s.t. } c + a' = aR + b + T$$

- The function $u(c)$ is strictly concave and twice continuously differentiable.

- The way we solved this in first-year macro is by iterating for each value of $a_i \in \{a_0, ..., a_{nkk}\}$ through all values of $a'_j \in \{a_0, ..., a_{max}\}$ where,

$$a_{max} = a_i R + b + T$$

Discrete Approximation

Exploiting Monotonicity

- As you might have guessed, this is a not very smart algorithm.
- We can do much better if we exploit the structure of the problem.
- We can exploit the monotonicity of the policy function i.e

$$a_i > a_j \Rightarrow g^a_{J-1}(a_i) > g^a_{J-1}(a_j)$$

Discrete Approximation

Exploiting Concavity

- Second, we can shorten the number of computations in the maximization since the function

$$f(a') = u(aR + b + T - a') + s_j \beta v_J(a')$$

is strictly concave.

- A strictly concave function defined over a grid of nkk points either takes its maximum at one of the boundary points or in the interior of the grid.
  - In the first case the function is decreasing (increasing) over the whole grid, if the maximum is the first (last) point of the grid.
  - In the second case the function is first increasing and then decreasing.
- As a consequence we can take the mid-point of the grid a_i and the grid point next to it a_{i+1} and determine whether the maximum is to the right of a_i.

Discrete Approximation

Binary Search Algorithm

- Find the maximum of a strictly concave function $f(x)$ defined over a grid $\{x_1, ..., x_n\}$

1. Select two points: $i_l = \text{floor}\left(\dfrac{i_{min} + i_{max}}{2}\right)$ and $i_u = i_l + 1$
2. If $f(x_{i_u}) > f(x_{i_l})$ set $i_{min} = i_l$ otherwise set $i_{min} = i_u$
3. If $i_{max} - i_{min} = 2$, stop and choose the largest element among $f(x_{i_{min}})$, $f(x_{i_{min}+1})$, and $f(x_{i_{max}})$. Otherwise return to Step 2.

Policy Function Iteration

- In infinite horizon models, we use value function iteration to find the fixed point of the operator.
- Value function iteration is nevertheless a slow procedure since it converges linearly at the rate β:

$$||\boldsymbol{v}^{s+1} - \boldsymbol{v}^*|| \leq \beta||\boldsymbol{v}^s - \boldsymbol{v}^*||$$

- Howard's improvement algorithm or policy function iteration is a method to enhance convergence.
- Each time a policy function is computed, we solve for the value function that would occur, if the policy were followed forever.
- This value function is then used in the next step to obtain a new policy function.

Policy Function Iteration

- What is the value function associated with a given $g^k(k, z)$?

$$v(k_i, z_m) = u(z_m f(k_i) - k_j) + \beta \sum_{z_l} \Pi(z_l|z_m) v(k_j, z_l),$$

where $k_j = g^k(k_i, z_m)$.

- In matrix notation:

$$\operatorname{vec} \boldsymbol{v} = \operatorname{vec} \boldsymbol{u} + \beta Q \operatorname{vec} \boldsymbol{v}$$

with solution:

$$\operatorname{vec} \boldsymbol{v} = [I - \beta Q]^{-1} \boldsymbol{u}$$

- If the state is space is large, computing the inverse of $[I - \beta Q]^{-1}$ can be very expensive.

Modified Policy Function Iteration

- Instead of computing the inverse, use a value function which is close but not exactly the one associated with the proposed policy function.
- Run a k number of times the following code:

$$
\begin{aligned}
\boldsymbol{w}^1 &= \boldsymbol{v}^0 \\
\operatorname{vec} \boldsymbol{w}^{l+1} &= \operatorname{vec} \boldsymbol{u} + \beta Q \operatorname{vec} \boldsymbol{w}^l \\
\boldsymbol{v}^1 &= \boldsymbol{w}^{k+1}
\end{aligned}
$$

Interpolation Between Grid Points

- In case the relevant state space is large, the computation time on a grid with many points may become a binding constraint.
- We, thus, look for methods that increase precision for a given number of grid-points without a compensating rise in computation time.
- How do we accomplish this?

Interpolation Between Grid Points

- Imagine that using the first year macro code, we found that $a' = a_j$ is optimal for the set of points in the grid.

- Since the value function is increasing and concave, the true maximizer must lie in the interval $[a_{j-1}, a_{j+1}]$.

- If we were able to evaluate the rhs of the Bellman equation at all $a' \in [a_{j-1}, a_{j+1}]$, we could pick the maximizer of the function in this interval.

- Two things are necessary to achieve this goal:

  1. an approximation of the value function over the interval $[a_{j-1}, a_{j+1}]$
  2. a method to locate the maximum of a continuous function.

Interpolation Between Grid Points

Linear Interpolation

- Linear interpolation is simple and shape preserving.
- This property is important, if we use interpolation to approximate the value function, which is known to be concave and increasing.
- Linear interpolation uses the point:

$$\hat{f}(x) := f(x_1) + \frac{f(x_2) + f(x_1)}{x_2 - x_1}(x - x_1)$$

- Thus, f is approximated by the line through $(x1, f(x1))$ and $(x2, f(x2))$.

Interpolation Between Grid Points

Cubic Splines

- Sometimes we are interested in preserving the smoothness of a function.
- Assume that we approximate the function $f(x)$ by a function $s(x)$ over the grid $\boldsymbol{x} = [x_0, x_1, \ldots, x_n]$ with corresponding function values $\boldsymbol{y} = [y_0, y_1, \ldots, y_n]$ with $y_i = f(x_i)$.
- On each subinterval $[x_{i-1}, x_i]$, we will approximate $f(x)$ with a cubic function $s(x) = a_i + b_i x + c_i x^2 + d_i x^3$

Interpolation Between Grid Points

Cubic Splines

- We impose that:

1. The approximation is exact at the grid-points, $y_i = s(x_i)$:

$$y_i = a_i + b_i x_i + c_i x_i^2 + d_i x_i^3, \;\; i = 1, \dots, n$$
$$y_i = a_{i+1} + b_{i+1} x_i + c_{i+1} x_{i+1}^2 + d_{i+1} x_i^3, \;\; i = 0, \dots, n-1$$

2. The first and the second derivatives agree on the nodes:

$$b_i + 2c_i x_i + 3d_i x_i^2 = b_{i+1} + 2c_{i+1} x_i + 3d_{i+1} x_i^2, \;\; i = 1, \dots, n-1$$
$$2c_i + 6d_i x_i = 2c_{i+1} + 6d_{i+1} x_i, \;\; i = 1, \dots, n-1$$

These conditions amount to $4n - 2$ linear equations in the $4n$ unknowns a_i, b_i, c_i, d_i leaving us two conditions short of fixing the coefficients.

Interpolation Between Grid Points

Cubic Splines

- Two possible solutions:
    1. Natural spline:

$$s''(x_0) = s''(x_n) = 0$$

    2. Secant Hermite spline: use the slope of the secant lines over $[x_0, x_1]$ and $[x_{n-1}, x_n]$ respectively:

$$s'(x_0) = \frac{y_1 - y_0}{x_1 - x_0} = b_1 + 2c_1 x_0 + 3d_1 x_0^2$$
$$s'(x_n) = \frac{y_n - y_{n-1}}{x_n - x_{n-1}} = b_n + 2c_n x_n + 3d_n x_n^2$$

Locating the Maximum

- Using these interpolation methods allows us to approximate the rhs of the Bellman equation by a continuous function:

$$\hat{\phi}(K) = u(f(K_i) - K) + \beta \hat{v}(K)$$

- In the interval $[K_{j-1}, K_{j+1}]$ the maximum of $\hat{\phi}(K)$ is located either at the end-points or in the interior.

Locating the Maximum

Golden Section Search

- This method locates the maximum of a single peaked function $f(x)$ in the interval $I = [A, D]$.

- The idea is to shrink the interval around the true maximizer x^* in successive steps until the midpoint of the remaining interval is a good approximation to x^*.

- Imagine we have two other function evaluations at points B and C
  - If $f(B) > f(C) \Rightarrow$ look in $[A, C]$
  - If $f(B) < f(C) \Rightarrow$ look in $[B, D]$

- How to choose B and C?

Locating the Maximum

Golden Section Search

- Choose them such that $\bar{AC} = \bar{BD}$:

$$1.\ \bar{AD} = \bar{AC} + \bar{CD}$$

$$2.\ \frac{\bar{AC}}{\bar{AD}} = \frac{\bar{CD}}{\bar{AC}}$$

substitute 1 in 2:

$$\frac{\bar{AC}}{\bar{AC} + \bar{CD}} = \frac{\bar{CD}}{\bar{AC}}$$

define $p = \dfrac{\bar{CD}}{\bar{AC}}$ solve for p:

$$p = (\sqrt{5} - 1)/2$$

Projection Methods

- When solving DP problems, we are trying to solve for a function f (the value function or policy function) that satisfies a condition (Bellman equation or euler equation).
- Projection methods provide approximate solutions to functional equations.
- Different from $\mathbb{R}^n$, however, function spaces have infinite dimensions.
- Projection methods use a family of polynomials $\mathcal{P} := \{\psi_i\}_{i=0}^{\infty}$ and approximate f by a finite sum of members of this family.

Motivating Example

- Consider the ordinary differential equation:

$$f'(x) + f(x) = 0, \ f(0) = 1,$$

the solution is given by: $f(x) = e^{-x}$.

Definitions

- In order to approximate any function we choose:
  - A basis Ψ.
  - An order of approximation
  - An interval over which we approximate the function

- **Definition:** Basis
  A subset Ψ of a vector space $\boldsymbol{V}$ is a basis if all $\psi \in \Psi$ are linearly independent and all $v \in \boldsymbol{V}$ can be expressed as a linear combination of the elements of Ψ.

Definitions

- Consider the set of all continuous functions that map the interval $[a, b]$ to the real line denoted by $C[a, b]$.

- This set is a vector space and monomials build a base Ψ_m for this space i.e. every element of the set can be represented by:

$$f(x) = \sum_{i=0}^{\infty} \gamma_i x^i$$

- For this reason it is common to use a linear combination of the first p members of this base to approximate a continuous function $f(t)$ in $C[a, b]$

$$f(x) \simeq \sum_{i=0}^{p} \gamma_i x^i$$

Back to Example

- Back to our example, let's approximate $f(x)$ using the basis of monomials with the first 3 members (p=3).

$$\hat{f}(x) = \gamma_0 + \gamma_1 x + \gamma_2 x^2 \ \ (\gamma_0 = 1 \text{ since } f(0) = 1)$$

- Using the differential equation, let us define the residual function:

$$R(\gamma, t) = \gamma_1 + 2\gamma_2 x + 1 + \gamma_1 x + \gamma_2 x^2$$

- This function describes the error that results if we use our guess of the solution instead of the true solution in the functional equation.
- We want γ to minimize $R(\gamma, x)$ for all $x \in [a, b]$ given some metric. This step is known as the projection against a given basis.

Projections Direction

- Generally, we can write a projection as a weighting function $p(x)$ which together with R define an inner product given by:

$$\int_x p(x)R(\gamma, x)dx$$

- We look for a γ such that:

$$\int_x p(x)R(\gamma, x)dx \simeq 0$$

- Depending on the projection that we use we will obtain different results.

Least Squares Projection

- Choose as projection direction the gradient of the loss function:

$$p(x) = \frac{\partial R(\gamma, x)}{\partial \gamma}$$

- The implied problem would be equivalent to solving:

$$\min_{\gamma} \int_x R(\gamma, x)^2 dx$$

- By FOCs using Leibniz integral rule:

$$\int_x \frac{\partial R(\gamma, x)}{\partial \gamma} R(\gamma, x) dx = 0$$

Least Squares Projection

Example

- In the example, for the interval $[0, 2]$, the least squares projection is found by solving:

$$\min_{\gamma} \int_0^2 (1 + \gamma_1(1 + x) + \gamma_2(2x + x^2))^2 dx$$

with FOCs:

$$\int_0^2 (1 + x)(1 + \gamma_1(1 + x) + \gamma_2(2x + x^2))dx = 0$$
$$\int_0^2 (2x + x^2)(1 + \gamma_1(1 + x) + \gamma_2(2x + x^2))dx = 0$$

Dirac Delta Projection / Collocation

- Discretize the state space x into a grid of size p $\tilde{x} = \{\tilde{x}_0, \tilde{x}_1, \ldots, \tilde{x}_{p-1}\}$.
- Projection direction are given by the Dirac deltas:

$$p_i(x) = \begin{cases} 1 \text{ if } x = \tilde{x}_i \\ 0 \text{ otherwise} \end{cases}$$

- This is a fancy way to describe the following: we want the Residual function to be satisfied exactly at p chosen points of x.
- We therefore obtain a system of p equations and p unknowns.

Dirac Delta Projection / Collocation

Example

- We may want that the residual function is equal to zero at a given set of points:
- Suppose we choose $x_1 = 1$ and $x_2 = 2$.
- This gives the linear system:

$$
\begin{aligned}
-1 &= 2\gamma_1 + 3\gamma_2 \\
-1 &= 3\gamma_1 + 8\gamma_2
\end{aligned}
$$

Galerkin Projection

- Projection direction is given by:

$$p_i(x) = \psi_i(x)$$

- There we solve for:

$$\int_x \psi_i(x) R(\gamma, x) dx = 0$$

- Which is also a system of p equations (one for each member of the basis) and p unkowns.

Galerkin Projection

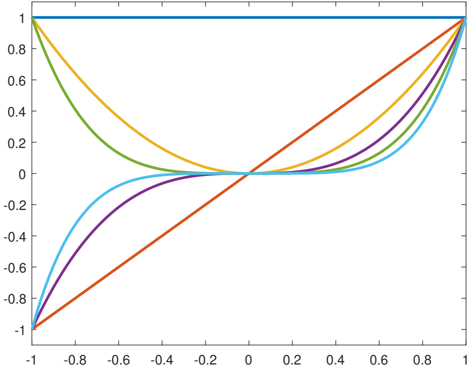
Example

- For our example we would have:

$$\int_0^2 xR(\gamma, x)dx = 0$$
$$\int_0^2 x^2 R(\gamma, x)dx = 0$$

Orthogonal Bases

- This is how the first six elements of the basis of monomials look like.



- The elements of this basis share a lot of information. This makes the numerical solutions suffer in terms of precision.

Orthogonal Bases

Definition

- **Definition:** Orthogonality
  A family of elements $\Psi = \{\psi\} \subset \boldsymbol{V}$ is orthogonal with respect to the inner product $< \cdot, \cdot >$ if $\forall i \neq j, < \psi_i, \psi_j >= 0$

- **Definition:** Orthogonal basis
  A subset Ψ of an inner product vector space $\boldsymbol{V}$ is an orthogonal basis of $\boldsymbol{V}$ if it is a basis and all its elements are orthogonal.

- The general idea is that elements of these bases share much less information making the numerical solutions much more precises.

Orthogonal Bases

Chebyshev polynomials

- Define the element i as,

$$\psi_i(x) = \cos(i \arccos(x))$$

- They share much less information than the monomials.
- They can also be generated recursively,

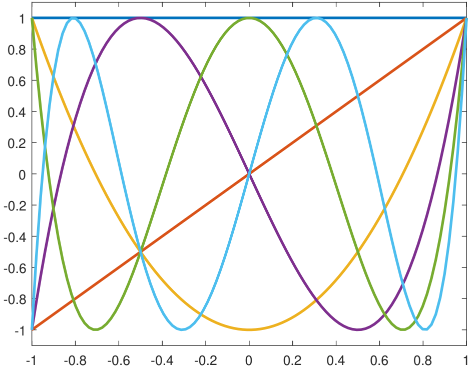
$$\begin{aligned} i = 0,\ \ \psi_i(x) &= 1 \\ i = 1,\ \ \psi_i(x) &= x \\ i \geq 2,\ \ \psi_i(x) &= 2x\psi_{i-1}(x) - \psi_{i-2}(x) \end{aligned}$$

- Chebyshev polynomials are defined in the interval $[-1;1]$, but this is not a limitation as long as the domain of $f(x)$ is bounded.

Orthogonal Bases

Chebyshev polynomials

- This is how the first six elements of the basis of the Chebyshev polynomials look like.



Neoclassical Growth Model

- Let's return to our economic model of reference: the Neoclassical Growth Model.
- And let's go to our preferred equation: the Euler Equation.

$$u_c(c) = \beta f'(f(K) - c)u_c(c')$$

- Assume we want to solve this model in terms of the policy function $C(K)$.

Neoclassical Growth Model

- Letting $\hat{C}(\gamma, K)$ denote the approximate solution, the residual function may be computed from:

$$R(\gamma, K) = \frac{u'(\hat{C}(\gamma, K))}{u'(\hat{C}(\gamma, f(K) - \hat{C}(\gamma, K)))} - \beta f'(f(K) - \hat{C}(\gamma, K))$$

Neoclassical Growth Model

Collocation

- Computing the solution using collocation is relatively straight forward.
- You need to choose p number of points for which to solve the Euler equation.
  - The Chebyshev interpolation theorem shows that Chebyshev zeros minimize the maximal interpolation error (Chebyshev collocation).
- Chebyshev zeros:

$$\tilde{k}_i = \cos\left(\frac{2i-1}{2p}\pi\right),\ i = 1, \ldots, p$$

with, $\tilde{k}_i = \frac{2k}{b-a} - \frac{a+b}{b-a},\ \ \tilde{k}_i \in [0,1], k_i \in [a,b]$

Neoclassical Growth Model

Least Squares

- In case we choose the least squares projection, we need to compute the following integral:

$$\min_{\gamma} \int_a^b R(\gamma, k)^2 dk$$

- How to compute this integral?

Numerical Integration

- There are two ways of computing an integral $\int_a^b f(x)dx$ numerically:
  1. Newton-Cotes Formulas
  2. Gaussian Formulas

Numerical Integration

Newton-Cotes Formulas

- The first idea is to approximate the function $f(x)$ by piecewise polynomials and integrate the polynomials over subdomains of $[a, b]$.

$$\int_a^b f(x)dx \simeq \frac{b-a}{2}[f(a) + f(b)]$$

- If we use higher-order polynomials or a higher number of subdomains, more generally, we derive a Newton-Cotes formula for the approximation of the integral which evaluates the integral at a number of points:

$$\int_a^b f(x)dx \simeq \sum_{i=1}^{n} a_i f(x_i)$$

Numerical Integration

Gaussian Formulas

- In Gaussian formulas, we choose weights and nodes optimally in order to provide a good approximation of $\int_a^b f(x)dx$.

- Choosing an orthogonal basis for approximating $f(x)$ is important since it can be shown that we can compute the integral of a polynomial of degree $2n-1$ exactly.

- Gauss-Chebyshev quadrature formula:

$$\int_a^b f(z)dz \simeq \frac{\pi(b-a)}{2n} \sum_{i=1}^{n} f(z_i)\sqrt{1+\tilde{z}_i}\Bigg)$$

where $\tilde{z}_i$ are the Chebyshev zeros.

Neoclassical Growth Model

Least Squares

- Then using the Gauss-Chebyshev quadrature formula we obtain:

$$S(\gamma) = \int_a^b R(\gamma, K)^2 dk \simeq \frac{\pi(b-a)}{2n} \sum_{l=1}^{n} R(\gamma, k_l)^2 \sqrt{1 + \tilde{k}_l}$$

In order to obtain the least squares projection of $C(K)$, we would need to minimize $S(\gamma)$ using a minimization routine.

Neoclassical Growth Model

Galerkin

- With the Galerkin projection method we use again Gauss-Chebyshev quadrature.
- With this, we must solve the system of p non-linear equations:

$$0 = \frac{\pi(b-a)}{2n} \sum_{l=1}^{n} R(\gamma, k_l) T_i(\tilde{k}_l) \sqrt{1 + \tilde{k}_l}$$

Simple RBC

- Euler Equation:

$$u_c(c_t) = \beta E[\exp(z_{t+1}) f_k(k_{t+1}) u_c(c_{t+1})]$$
$$\text{s.t. } z_{t+1} = \rho z_t + \epsilon_{t+1}, \epsilon \sim N(0, \sigma_\epsilon)$$

- Letting $\hat{C}(\gamma, K)$ denote the approximate solution, the residual function can be computed as:

$$\begin{aligned} R(\gamma, K, z) =& E\Bigg[\frac{u_c\Big(\hat{C}(\gamma, K, z)\Big)}{u_c\Big(\hat{C}(\gamma, zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z), z')\Big)} \\ & - \beta\Big(\exp(z') f_k\Big(zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z)\Big) + 1 - \delta\Big)\Bigg] \end{aligned}$$

How to deal with the expectations operator?

$$
\begin{aligned}
R(\gamma, K, z) = \int_{-\infty}^{\infty} \frac{1}{\sigma_\epsilon \sqrt{2\pi}} \exp\Big(-\frac{\epsilon'^2}{2\sigma_\epsilon^2}\Big) \times \\
\Bigg[\frac{u_c\Big(\hat{C}(\gamma, K, z)\Big)}{u_c\Big(\hat{C}(\gamma, zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z), \rho z + \epsilon')\Big)} \\
- \beta\Big(\exp(\rho z + \epsilon') f_k\Big(zf(K) \\
+ (1-\delta)K - \hat{C}(\gamma, K, z)\Big) + 1 - \delta\Big)\Bigg] d\epsilon'
\end{aligned}
$$

How to deal with the expectations operator?

Gauss–Hermite Quadrature

- Apply the following change of variable $x = \dfrac{\epsilon'}{\sqrt{2}\sigma_\epsilon}$ then,

  $dx = \dfrac{1}{\sqrt{2}\sigma_\epsilon} d\epsilon'$,

$$R(\gamma, K, z) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{\pi}} \exp(-x^2) \times$$

$$\Bigg[\frac{u_c\Big(\hat{C}(\gamma, K, z)\Big)}{u_c\Big(\hat{C}(\gamma, zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z), \rho z + \sqrt{2}\sigma_\epsilon x)\Big)}$$

$$- \beta\Big((\rho z + \sqrt{2}\sigma_\epsilon x) f_k\Big(zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z)\Big) + 1 - \delta\Big)\Bigg]$$

$$dx$$

How to deal with the expectations operator?

Gauss-Hermite quadrature

- This integral can be approximated by the Gauss-Hermite quadrature formula:

$$R(\gamma, K,z) \simeq \sum_{l=1}^{n} \frac{1}{\sqrt{\pi}} w_l$$

$$\left[\frac{u_c\Big(\hat{C}(\gamma, K, z)\Big)}{u_c\Big(\hat{C}(\gamma, zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z), \rho z + \sqrt{2}\sigma_\epsilon x_l)\Big)} \right.$$

$$\left. - \beta\Big((\rho z + \sqrt{2}\sigma_\epsilon x_l) f_k\Big(zf(K) + (1-\delta)K - \hat{C}(\gamma, K, z)\Big) + 1 - \delta\Big) \right]$$

- For different n's, the integration nodes x_l and weights w_l can be found in books.

Simple RBC

Function approximation: Examples

- Monomials:

$$\hat{C}(\gamma, K, z) = \gamma_0 + \gamma_1 k + \gamma_2 z + \gamma_3 k^2 + \gamma_4 kz + \gamma_5 z^2$$

- Chebyshev:

$$\hat{C}(\gamma, K, z) = \sum_{i=1}^{p_1} \sum_{j=1}^{p_2} \gamma_{ij} T_i(\tilde{K}) T_j(\tilde{z})$$

Simple RBC

Collocation

- Given the residual function that we have computed, we could solve a on linear system for a number of points in the state space $(z \times k)$ equal to the number of parameters that you need to estimate

Simple RBC

Least Squares

- We need to compute a multidimensional integral:

$$S(\gamma) = \int_{a_k}^{b_k} \int_{a_z}^{b_z} R(K, z; \gamma)^2 dk\, dz$$

- We could again use Gauss-Chebyshev quadrature to approximate it with:

$$S(\gamma) \simeq \frac{\pi^2 (b_k - a_k)(b_z - a_z)}{(2n)^2} \sum_{l=1}^{n} \sum_{m=1}^{n} R(\gamma, k_l, z_m)^2 \sqrt{1 + \tilde{k}_l} \sqrt{1 + \tilde{z}_m}$$

- With Galerkin projection you will have one equation for each element of the family of polynomials.

Accuracy

- A perfect solution will make zero the Euler equation in all the points of the domain.
- This will be typically impossible.
- We have considered different definitions for making the residual small.
- For instance, using state space methods, we chose to make the residual 0 at a certain points and we have paid no attention to the rest of the domain.
- So, how far are we from a zero of the equation in all the domain?

Accuracy

- We could define the residual function as:

$$\mathcal{N}(k, z; \gamma) = \text{abs}\left[\hat{C}(k, z; \gamma) - u_c^{-1}\Big(E\Big[\beta z' f_k(K') u_c(\hat{C}(k', z'; \gamma))\Big]\Big)\right]$$

which measure the numerical error in consumption units.

- People like to report the relative consumption error or even its log,

$$\mathcal{N}_R(k, z; \gamma) = \frac{\mathcal{N}(k, z; \gamma)}{\hat{C}(k, z; \gamma)}; \text{ or } \mathcal{N}_L(k, z; \gamma) = \log(\mathcal{N}_R(k, z; \gamma))$$

Accuracy

- Our accuracy measure is not a real number but a function in $\boldsymbol{z} \times \boldsymbol{k}$.
- There is a lot of information to convey.
  - We can plot the function
  - We can compute some summary statistics. A favorite one would be:

    $$\log \int_{\boldsymbol{z} \times \boldsymbol{k}} \mathcal{N}_R(k, z; \gamma) d\hat{\mu}$$

    This measure gives a sense of the average error, where the average is computed by giving more weight wherever we have more people and just forgetting about accuracy where there is no action.
  - Another typical statistic reported,

    $$\max_{z,k} \log \mathcal{N}_R(k, z; \gamma)$$