MATLAB programs for Chebyshev projection of multivariate functions
by Shouyong Shi
2012

1. Core programs:
   • chebypol.m: performing step 1 (see the description below)
   • chebybase.m: performing step 2
   • chebyLSR.m: performing step 4.1
   • chebyval.m: performing step 4.2
   • stackgrid.m: re-organizing matrices and cells

2. Test programs:
   • chebyval_test.m: implementing the method with a given function \( f \) and testing the precision of Chebyshev projection.
     funtest.m: a function used in chebyval_test.m.
   • vfun_cheby.m: using Chebyshev projection to compute a one-sector growth model with dynamic programming. In this application, the function \( f \) to be approximated is the value function. Computing the sample for the projection (in step 3) means solving the Bellman equation at all of the grid points. This program also illustrates the precision of the resulted policy functions.
     u.m: the utility function used in vfun_cheby.m.

Chebyshev projection of a multivariate function

The function to be approximated by Chebyshev projection:

\[
f(x_1, \ldots, x_J) \in C(\prod_{j=1}^{J}[x_j, \bar{x}_j]); \quad J \geq 1 \text{ is an integer.}
\]
Step 1. Create the grid points (nodes) for each \( x_j, j = 1, 2, ..., J \):

1.1. Normalize \( x_j, j = 1, 2, ..., J \), where
\[
z(x_j) = \frac{2(x_j - \bar{x}_j)}{\bar{x}_j - \bar{x}_j} - 1,
\]
and so \( x(z_j) = (z_j + 1) \frac{\bar{x}_j - \bar{x}_j}{2} + \bar{x}_j \).

1.2. Choose the number of grid points, \( m_j \), and create the grid for each \( z_j \): The grid points are very specific. They are the zeros of the degree-\( m_j \) Chebyshev polynomial of \( z_j \). For any integer \( m \), the degree-\( m \) Chebyshev polynomial, \( T_m(z) \), is
\[
T_m(z) = \cos(m \cos^{-1} z).
\]
This can be generated recursively as follows:
\[
T_0(z) = 1, \quad T_1(z) = z, \quad T_{i+1}(z) = 2zT_i(z) - T_{i-1}(z), \quad i = 1, 2, ...
\]
The zeros of \( T_m(z) \) are
\[
z_k = -\cos\left(\frac{2k - 1}{2m} \pi\right), \quad k = 1, ..., m.
\]

1.3. The grid points of \( x_j \) are \( \{x(z_k)\}_{k=1}^{m_j} \), where \( \{z_k\} \) are the grid points for \( z_j \) above.

Step 2. Create the basis for Chebyshev projection:

2.1. Choose \( n_j \), the highest degree of the Chebyshev polynomial in the basis for \( x_j \). This choice must satisfy \( n_j \leq m_j - 1 \) for all \( j \).

2.2. For each \( j \in \{1, 2, ..., J\} \), evaluate the Chebyshev polynomials of degree 0 to \( n_j \) at each grid point \( z_{kj} \). That is, compute:
\[
T_{ij}(z_{kj}) = \cos(i_j \cos^{-1}(z_{kj})) \text{ for } i_j = 0, 1, ..., n_j \text{ and for each } z_{kj}.
\]
Again, use the recursion on \( T \) to compute these polynomials.

2.3. Normalize the polynomials computed in step 2.2 for \( z_j \) as
\[
T_{z_{ij}}(z_{kj}) = \frac{T_{ij}(z_{kj})}{\sum_{k=1}^{m_j} (T_{ij}(z_{kj}))^2}.
\]
Use the following property to compute the denominator:
\[
\sum_{k=1}^{m_j} (T_{ij}(z_{kj}))^2 = \begin{cases} m_j, & \text{if } i_j = 0 \\ m_j/2, & \text{if } i_j = 1, ..., n_j. \end{cases}
\]
The matrix \( [T z_{ij}(z_{k_j})]: i_j = 0, 1, ..., n_j; k_j = 1, 2, ..., m_j \) is the Chebyshev basis for \( z_j \).

**Step 3.** Create the sample for the projection:

For each grid point \( x = (x_{k_1}, x_{k_2}, ..., x_{k_J}) \), where \( k_j = 1, ..., m_j \), compute the sample value \( f(x) \). This step depends on the specific problem in hand.

**Step 4.** Estimate the coefficients and interpolate:

4.1. Projection: for \( i_j = 0, 1, ..., n_j \) and \( j = 1, ..., J \), projecting the sample onto the Chebyshev basis yields the following estimate of the coefficients \( a_{i_1...i_j} \):

\[
a_{i_1...i_j} = \sum_{k_1=1}^{m_1} \cdots \sum_{k_J=1}^{m_J} f(x_{k_1}, ..., x_{k_J}) \prod_{j=1}^{J} T z_{ij}(z_{k_j}).
\]

4.2. Interpolation: for any \( x = (x_1, ..., x_J) \) in the interior of \( \prod_{j=1}^{J} [x_j, \bar{x}_j] \), \( f(x) \) is interpolated as

\[
\hat{f}(x_1, ..., x_J) = \sum_{i_1=0}^{n_1} \cdots \sum_{i_J=0}^{n_J} a_{i_1...i_J} \prod_{j=1}^{J} T_i_{ij}(z(x_j)).
\]

For the related description of the method, see