Finite element solution of a 1D BVP

We’ll use the Galerkin finite element method to solve approximately the BVP

\[ u'' = 1 + e^{2x} \]
\[ u(0) = u(1) = 0. \]

Let \( V_0 \) be the space of admissible functions satisfying the BCs. The weak form of this problem is

\[ \int_0^1 u' v' + f(x) v \, dx = 0 \quad \forall \ v \in V_0 \]

We can form a discrete system by representing \( u \) as a LC of basis functions, then requiring that the equation hold for each \( v \) in that basis.

To demonstrate we’ll use two different sets of basis functions: the 1st-degree Lagrange basis functions (piecewise linear) and the \( M \)-th degree Bernstein polynomial basis.

- Define an expression for the RHS of the equation

```
In[190]:= f[x_] = 1 + Exp[2 x];
```

Setting up the basis functions

- Define an expression for the M-node 1st-degree Lagrange basis functions

You’ve seen the Lagrange basis functions in a homework problem.

```
In[190]:= xNode[M_, n_] = n / (M + 1);

In[191]:= \[\phi\] [M_, n_, x_] = Piecewise[{
}]

Out[191]= Piecewise[{
{\(\frac{x-M \theta}{1+M \theta}\), \(\frac{n-1}{M+1}\) \[LessEqual] x \[LessThan] \(\frac{n}{M+1}\)},
{\(\frac{n+1-M \theta}{1+M \theta}\), \(\frac{n+1}{M+1}\) \[LessEqual] x \[LessThan] \(\frac{n+1}{M+1}\)}
}]
```
Define an expression for the $M+1$-th degree Bernstein basis functions

The Bernstein basis is a set of polynomials having a number of useful properties. For our purposes, they are useful because it’s easy to drop certain members to satisfy BCs.

\[
\begin{align*}
\chi[M_, n_, x_] &= \text{Binomial}[M+1, n] x^n (1-x)^{M+1-n} \\
\end{align*}
\]

First, plot the entire set from $n = 0$ to $M+1$.

These are now a suitable basis for a subspace of $V_0$.

Write functions to compute the matrix and vector

The stiffness matrix for this problem is
\[ K_{ij} = \int_0^1 w_i \cdot w_j \, dx \]

where \( w_i \) is the \( i \)-th basis function.

```
In[195]:= stiffnessMatrix[_, bas_] := Table[
    Integrate[D[bas[M, i, x], x] D[bas[M, j, x], x], {x, 0, 1}],
    {i, 1, M}, {j, 1, M}]
```

The load vector for this problem is

\[ f_i = -\int_0^1 w_i f(x) \, dx \]

where \( w_i \) is the \( i \)-th basis function.

```
In[196]:= loadVector[_, bas_] := Table[
    Integrate[-bas[M, i, x] f[x], {x, 0, 1}],
    {i, 1, M}]
```

### Compute the stiffness matrix for the Lagrange basis, \( M = 6 \)

```
In[197]:= stiffnessMatrix[6, \phi]
```

```
Out[197]=
\[
\begin{pmatrix}
14 & -7 & 0 & 0 & 0 & 0 \\
-7 & 14 & -7 & 0 & 0 & 0 \\
0 & -7 & 14 & -7 & 0 & 0 \\
0 & 0 & -7 & 14 & -7 & 0 \\
0 & 0 & 0 & -7 & 14 & -7 \\
0 & 0 & 0 & 0 & -7 & 14
\end{pmatrix}
\]
```

Notice that this matrix is *sparse*, that is, most of its entries are zero. Why is this? High-performance linear solvers can take advantage of sparsity, so the sparsity of the resulting stiffness matrix is one of the advantages of the Lagrange basis.

### Compute the stiffness matrix for the Bernstein basis, \( M=6 \)

```
In[198]:= stiffnessMatrix[6, \chi]
```

```
Out[198]=
\[
\begin{pmatrix}
294 & 147 & -49 & 49 & 28 & 161 \\
294 & 286 & 147 & 572 & 143 & 1716 \\
286 & 143 & 143 & 429 & 7 & 119 \\
147 & 49 & 70 & 175 & 7 & 147 \\
147 & 143 & 143 & 572 & 286 & 572 \\
572 & 286 & 572 & 143 & 429 & 147 \\
28 & 119 & 7 & 49 & 98 & 147 \\
28 & 143 & 286 & 143 & 143 & 286 \\
161 & 28 & 147 & 49 & 147 & 294 \\
1716 & 143 & 572 & 429 & 286 & 143
\end{pmatrix}
\]
```

Notice that this matrix is *not* sparse.
Solving the problem

- Write helper functions to solve the linear system and sum the linear combination of basis functions

\[
\text{solnCoeffs}[M_\text{, } \text{bas}_\text{, } x_] := \text{Block[}
\{K = \text{stiffnessMatrix}[M, \text{bas}],
    \text{b = loadVector}[M, \text{bas}],
    \text{LinearSolve}[N[K], N[b]]
\}]
\]

\[
\text{solnSum}[M_\text{, } \text{bas}_\text{, } x_] := \text{Block[}
\{u = \text{solnCoeffs}[M, \text{bas}],
    \text{Sum}[u[[i]] \text{bas}[M, i, x], \{i, 1, M\}]\}
\]

- Use DSolve[] to find the exact solution

\[
\text{uEx}[x_] = w[x] /. \text{DSolve[}\{w''[x] == f[x], w[0] == 0, w[1] == 0\}, w[x], x][[1]]
\]

\[\frac{1}{4}(2 x^2 - e^x x - x + e^x - 1)\]

- Solve with \(M=3\) and with both bases

Solve with Lagrange

\[
\text{uf3}[x_] = \text{solnSum}[3, \phi, x];
\]

and with Bernstein

\[
\text{uf3}[x_] = \text{solnSum}[3, \chi, x];
\]

Plot the solutions

\[
\text{Plot[}\{\text{uf3}[x], \text{uf3}[x], \text{uEx}[x]\}, \{x, 0, 1\}, \text{ImageSize} \rightarrow \text{Small}]\]

- Write a function to automate the plot

Making plots for different orders gets tedious quickly, so write a function to do it for us.
Compared to the Lagrange results, the error in the Bernstein results gets smaller much faster as the order increases. The two basis sets represent two different approaches to refinement of the approximation: you can increase the number of nodes while holding the polynomial order constant (h-refinement), or you can increase the degree of the polynomials while holding the grid structure fixed (p-refinement). It can be shown that p-refinement reduces the error as $e^{-CM}$ (where C is some constant) whereas h-refinement decreases the error as $\sim M^{-p+1}$ where $p$ is the order of the basis used on each grid interval ($p=1$ with the first-degree Lagrange basis). In theory, then, p-refinement should be superior. However, in practice the integrations are more complicated (and more expensive) as the polynomial degree increases, and the resulting stiffness matrices are not as sparse which can make the linear solves significantly more expensive.