AM205: Assignment 1 solutions

Problem 1

We are given the data set $x = (0, 1, 2, 3)$ and $y = (0, 0, 1, 2)$. We want to show that using the monomial basis and using the Lagrange basis give us the same polynomial interpolation.

We first consider the monomial basis. We want to solve for the interpolation coefficients $v = (a, b, c, d)$ in the form of

$$y = a + bx + cx^2 + dx^3.$$  

The Vandermonde matrix is given by

$$A = \begin{pmatrix}
1  & x_0  & x_0^2  & x_0^3 \\
1  & x_1  & x_1^2  & x_1^3 \\
1  & x_2  & x_2^2  & x_2^3 \\
1  & x_3  & x_3^2  & x_3^3
\end{pmatrix} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
1 & 2 & 4 & 8 \\
1 & 3 & 9 & 27
\end{pmatrix}. \quad (1)$$

Solving the linear system $Av = y$ as $v = A^{-1}y$ yields

$$v = \left(0, -\frac{5}{6}, 1, -\frac{1}{6}\right) \quad (2)$$

and hence the polynomial interpolant is $y = -\frac{5}{6}x + x^2 - \frac{1}{6}x^3$. We now consider the Lagrange basis. We interpolate the data using Lagrange polynomials in the form of

$$y(x) = y_0L_0(x) + y_1L_1(x) + y_2L_2(x) + y_3L_3(x), \quad (3)$$

where the Lagrange polynomials are defined as

$$L_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x-x_i}{x_k-x_i}. \quad (4)$$

For the given data, the four polynomials are

$$L_0 = -\frac{1}{6}(x-1)(x-2)(x-3),$$
$$L_1 = \frac{1}{2}(x-0)(x-2)(x-3),$$
$$L_2 = -\frac{1}{2}(x-0)(x-1)(x-3),$$
$$L_3 = \frac{1}{6}(x-0)(x-1)(x-2). \quad (5)$$

After simplification, we obtain $y = -\frac{5}{6}x + x^2 - \frac{1}{6}x^3$, which matches the result for the monomial basis. The left panel of Fig. 1 shows the interpolated function $y(x)$ with the data points superimposed.

*Solutions were written Kevin Chen (TF, Fall 2014), Dustin Tran (TF, Fall 2014), and Chris H. Rycroft. Edited by Chris H. Rycroft.*
Problem 2 – Error bounds with Lagrange Polynomials

Parts (a) and (b)

Figure 2 shows the Lagrange polynomial $p_3(x)$ over the true function $f(x)$ using a slightly modified version of the in-class code example. Running the code, the infinity norm of the error is approximately 5.75191.

Part (c)

The upper bound for the interpolation error is given by

$$\|f(x) - p_{n-1}(x)\|_\infty = \left\| \frac{f^{(n)}(\theta)}{n!} \prod_{i=1}^{n}(x - x_i) \right\|_\infty,$$

where $\theta$ is a specific value within the interval from $-1$ to $1$. Applying the Cauchy–Schwarz inequality gives

$$\|f(x) - p_{n-1}(x)\|_\infty \leq \frac{\|f^{(n)}(\theta)\|_\infty}{n!} \left\| \prod_{i=1}^{n}(x - x_i) \right\|_\infty.$$

Figure 1: Data points and the corresponding interpolating polynomial for problem 1.
Figure 2: The function $f(x) = e^{4x} + e^{-2x}$ and the interpolating polynomial $p_3(x)$ considered in problem 2.

Using the properties of the Chebyshev polynomials, and that the function $f$ achieves its maximum value at $x = 1$,

$$\|f(x) - p_{n-1}(x)\|_\infty \leq \frac{\left\|4^n e^{4\theta} + (-2)^n e^{-2\theta}\right\|_\infty}{n!} \frac{1}{2^{n-1}}$$

$$\leq \frac{4^n e^4 + (-2)^n e^{-2}}{n!} \frac{1}{2^{n-1}}$$

$$\leq \frac{2^{n+1} e^4 + (-1)^n 2 e^{-2}}{n!}. \quad (8)$$

**Part (d)**

There are many ways to find better control points, and this problem highlights that while the Chebyshev points are a good set of points at which to interpolate a general unknown function, it is usually rather straightforward to find a better set of interpolating points for a specific function.

One simple method is to examine where the maximum interpolation error is achieved. This is between the second and third control points; hence if we move the second control point closer to the right, and/or the third control point closer to the left, $p_3(x)$ would have a better approximation of $f(x)$ within this region.

We need only be careful not to move these control points too close together, or else we would achieve an even larger infinity norm as a result of a different region of $|p_3(x) - f(x)|$
getting larger. This also extends to the realization that in order to minimize the infinity norm, we want the maximum interpolation error to be achieved inbetween each control point equally. In other words, we aim to distribute the error as equally as possible.

In this case, we shift the second control point by 0.2, which leads to an infinity norm of 4.76548.

**Problem 3 – The condition number**

**Part (a)**

Throughout this equation, \( || \cdot || \) is taken to mean the Euclidean norm. The first two parts of this problem can be solved using diagonal matrices only. Consider first

\[
B = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}
\] (9)

Then \( ||B|| = 2, ||B^{-1}|| = 1 \) and hence \( \kappa(B) = 2 \). Similarly, \( \kappa(C) = 2 \). Adding the two matrices together gives

\[
B + C = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix} = 3I
\] (10)

and hence \( \kappa(B + C) = ||3I|| ||\frac{1}{3}I|| = 3 \times \frac{1}{3} = 1 \). For these choices of matrices, \( \kappa(B + C) < \kappa(B) + \kappa(C) \).

**Part (b)**

If

\[
B = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}
\] (11)

then \( \kappa(B) = 2 \) and \( \kappa(C) = 1 \). Adding the two matrices together gives

\[
B + C = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}
\] (12)

and hence \( \kappa(B + C) = 5 \). Therefore \( \kappa(B + C) > \kappa(B) + \kappa(C) \).

**Part (c)**

Let \( A \) be an invertible \( 2 \times 2 \) symmetric matrix. First, note that

\[
||2A|| = \max_{v \neq 0} \frac{||2Av||}{||v||} = \max_{v \neq 0} \frac{2||Av||}{||v||} = 2 \max_{v \neq 0} \frac{||Av||}{||v||} = 2 ||A||.
\] (13)
Similarly, note that \(|(2A)^{-1}| = |\frac{1}{2}A^{-1}| = \frac{1}{2}|A^{-1}|.|. Hence

\[
\kappa(2A) = |2A|||(2A)^{-1}| = 2|A|||A^{-1}| = |A|||A^{-1}| = \kappa(A). \quad (14)
\]

Now suppose that \(A\) is a symmetric invertible matrix. Then there exists an orthogonal matrix \(Q\) and a diagonal matrix \(D\) such that

\[
A = Q^TDQ.
\quad (15)
\]

Since \(Q^TQ = QQ^T = I\), it follows that

\[
A^2 = Q^TDQQ^TDQ = Q^TD^2Q. \quad (16)
\]

The matrix norm of \(|A|\) is

\[
|A| = \max_{v \neq 0} \frac{|Q^TDQv|}{||v||}. \quad (17)
\]

Since \(Q\) corresponds to a rotation or reflection, it preserves distances under the Euclidean norm and hence \(|Qw| = ||w|| = ||Q^Tw||\) for an arbitrary vector \(w\). Therefore

\[
|A| = \max_{v \neq 0} \frac{||DQv||}{||Qv||} = \max_{u \neq 0} \frac{||Du||}{||u||} = ||D|| \quad (18)
\]

where \(u = Qv\). Similarly \(|A^{-1}| = ||Q^TD^{-1}Q||\), and since \(D^{-1}\) is also diagonal it follows that \(|A^{-1}| = ||D^{-1}||\), so \(\kappa(A) = \kappa(D)\). With reference to the condition number notes, \(\kappa(A) = |\alpha \beta^{-1}|\) where \(\alpha\) is the diagonal entry with largest magnitude and \(|\beta|\) is the diagonal entry with the smallest magnitude.

Since \(D^2\) is also diagonal, it follows that \(|A^2| = ||D^2||\). The diagonal entry of \(D^2\) with the largest amplitude will be \(\alpha^2\), and the diagonal entry of \(D^2\) with the smallest amplitude will be \(\beta^{-2}\). Hence

\[
\kappa(A^2) = |\alpha^2 \beta^{-2}| = (\kappa(A))^2. \quad (19)
\]

**Part (d)**

The result for that \(\kappa(2A) = \kappa(A)\) is true for arbitrary 2 \(\times\) 2 invertible matrices. The derivation that was considered in part (c) did not rely on the matrix being symmetric.

The result about \(\kappa(A^2)\) does not generalize to arbitrary matrices. If

\[
A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad (20)
\]

then

\[
A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}. \quad (21)
\]

One can numerically verify that \(\kappa(A^2) = 5.828\) but \((\kappa(A))^2 = 6.854\), so the two do not agree.
We aim to construct a cubic spline \( s \) where \( \alpha, \beta, \gamma, \delta \) which are chosen because of their favorable properties at \( t = 0 \) and \( t = 1 \). The first and second derivatives must also be continuous at the end points. The pairwise occurrences of each constant are chosen to ensure that the first derivative is continuous.

To set the free parameters, the second derivatives must be made continuous at each end point. The pairwise occurrences of each constant are chosen to ensure that the first derivative is continuous.

Table 1: Properties of the four cubics used as a basis to compute the cubic spline.

<table>
<thead>
<tr>
<th>Function</th>
<th>( c_i(1) )</th>
<th>( c_i'(1) )</th>
<th>( c_i'(0) )</th>
<th>( c_i(0) )</th>
<th>( c_i''(1) )</th>
<th>( c_i''(0) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c_0(t) = t^2(3 - 2t) )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-6</td>
<td>6</td>
</tr>
<tr>
<td>( c_1(t) = -t^2(1 - t) )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>-2</td>
</tr>
<tr>
<td>( c_2(t) = (t - 1)^2t )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>-4</td>
</tr>
<tr>
<td>( c_3(t) = 2t^3 - 3t^2 + 1 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>-6</td>
</tr>
</tbody>
</table>

Problem 4 – Periodic cubic splines

Parts (a), (b), and (c)

We aim to construct a cubic spline \( s_x(t) \) of \( \sin \frac{\pi t}{2} \) over the periodic interval \( t \in [0, 4) \), using four cubic representations over the ranges \( [0, 1), [1, 2), [2, 3), \) and \( [3, 4) \). This gives sixteen degrees of freedom. Each cubic must match the function value at either end of its range (giving eight constraints). The first and second derivatives must also be continuous at the interface between each cubic, giving another eight constraints. Hence, the number of free parameters and the number of constraints agree, so there should be a unique solution.

A possible pitfall is to try and find the solution using a library routine, such as SciPy’s interp1d. However, the boundary conditions considered here are distinctly different from the defaults usually used by these routines. A standard (non-periodic) spline uses the conditions \( s''(0) = 0 \) and \( s''(4) = 0 \). On the other hand, to make the spline periodic, this problem uses the conditions \( s''(0) = s''(4) \) and \( s'(0) = s'(4) \). The different conditions fundamentally alter the solution.

To construct the spline, we make use of basis of cubics

\[
\begin{align*}
c_0(t) &= t^2(3 - 2t), & c_1(t) &= -t^2(1 - t), \\
c_2(t) &= (t - 1)^2t, & c_3(t) &= 2t^3 - 3t^2 + 1,
\end{align*}
\]

which are chosen because of their favorable properties at \( t = 0 \) and \( t = 1 \) that are summarized in Table 1. Using these functions, the spline can be written as

\[
s_x(t) = \begin{cases}
c_0(t) + \alpha c_1(t) + \delta c_2(t) & \text{for } t \in [0, 1), \\
c_3(t - 1) + \beta c_1(t - 1) + \alpha c_2(t - 1) & \text{for } t \in [1, 2), \\
-c_0(t - 2) + \gamma c_1(t - 2) + \beta c_2(t - 2) & \text{for } t \in [2, 3), \\
-c_3(t - 3) + \delta c_1(t - 3) + \gamma c_2(t - 3) & \text{for } t \in [3, 4),
\end{cases}
\]

where \( \alpha, \beta, \gamma, \) and \( \delta \) are unknown constants. The contributions of \( c_0 \) and \( c_3 \) are chosen to ensure each cubic matches \( \sin \frac{\pi t}{2} \) at the end points. The pairwise occurrences of each constant are chosen to ensure that the first derivative is continuous.

To set the free parameters, the second derivatives must be made continuous at each
interface. At \( t = 1, 2, 3, 4 \), by reference to Tab. 1, this gives

\[
\begin{align*}
-6 + 4\alpha + 2\delta &= -6 - 2\beta - 4\alpha, \\
6 + 4\beta + 2\alpha &= -6 - 2\gamma - 4\beta, \\
6 + 4\gamma + 2\beta &= 6 - 2\delta - 4\gamma, \\
-6 + 4\delta + 2\gamma &= 6 - 2\alpha - 4\delta,
\end{align*}
\]

(24) - (27) respectively. In matrix form, this is

\[
\begin{pmatrix}
8 & 2 & 0 & 2 \\
2 & 8 & 2 & 0 \\
0 & 2 & 8 & 2 \\
2 & 0 & 2 & 8
\end{pmatrix}
\begin{pmatrix}
\alpha \\
\beta \\
\gamma \\
\delta
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
-12 \\
0 \\
12
\end{pmatrix},
\]

(28) which has the unique solution \((\alpha, \beta, \gamma, \delta) = (0, -\frac{3}{2}, 0, \frac{3}{2})\). Hence

\[
s_x(t) = \begin{cases} 
  c_0(t) + \frac{3}{2}c_2(t) & \text{for } t \in [0, 1), \\
  c_3(t - 1) - \frac{3}{2}c_1(t - 1) & \text{for } t \in [1, 2), \\
  -c_0(t - 2) - \frac{3}{2}c_2(t - 2) & \text{for } t \in [2, 3), \\
  -c_3(t - 3) + \frac{3}{2}c_1(t - 3) & \text{for } t \in [3, 4). 
\end{cases}
\]

(29) Figure 3(a) shows a plot of \( s_x(t) \) in comparison to \( \sin \frac{n\pi}{2} \). There is a high level of agreement between the two functions.

Since \( \cos \frac{n\pi}{2} \) is the same as \( \sin \frac{n\pi}{2} \), but shifted by \(-1\), it follows that it can be approximated by the spline

\[
s_y(t) = \begin{cases} 
  c_3(t) - \frac{3}{2}c_1(t) & \text{for } t \in [0, 1), \\
  -c_0(t - 1) - \frac{3}{2}c_2(t - 1) & \text{for } t \in [1, 2), \\
  -c_3(t - 2) + \frac{3}{2}c_1(t - 2) & \text{for } t \in [2, 3), \\
  c_0(t - 3) + \frac{3}{2}c_2(t - 3) & \text{for } t \in [3, 4). 
\end{cases}
\]

Figure 3(b) shows a plot of \( s_y(t) \) in comparison to \( \cos \frac{n\pi}{2} \).

**Part (d)**

Figure 4(a) shows a plot of the parametric curve \( \mathbf{s}(t) = (s_x(t), s_y(t)) \) compared to a true circle. The am205_so11_ex3.py code integrates the area enclosed by this curve by dividing it into a large number of triangles. Recall that if \( \mathbf{a} \) and \( \mathbf{b} \) are vectors along two sides of a triangle, then the area is given by \( \frac{1}{2} |\mathbf{a} \times \mathbf{b}| \). For the given problem, consider the triangle with vertices at the origin, \( \mathbf{s}(t) \), and \( \mathbf{s}(t + \Delta t) \). Then for \( \Delta t \) small, vectors along two
sides are given by \( s(t) \) and \( s'(t) \Delta t \). Hence the total enclosed area is the sum of triangles such as this, and is hence given by the integral

\[
A = \frac{1}{2} \int_0^4 |s(t) \times s'(t)| \, dt. \tag{31}
\]

The program shows that \( A = 3.05000 \) to five decimal places using Python’s integration routine. The integrand is a fifth order polynomial, and hence it can also be evaluated exactly using alternative means to determine that \( A = \frac{61}{20} \). This differs from \( \pi \) by 3%.

**Part (e)**

Consider calculating a spline \( s_n^*(t) \) of \( \sin \frac{\pi t}{2} \) on the periodic interval \([0, 4)\) using \( n \) control points at \( t = t_0, t_1, \ldots, t_{n-1} \) where \( t_k = \frac{4k}{n} \). By extending the argument from part (a), one can determine that in the interval \([t_k, t_{k+1})\), the spline is given by

\[
s_n^*(t) = \left( \sin \frac{2k\pi}{n} \right) c_3(t_*) + \left( \sin \frac{2(k+1)\pi}{n} \right) c_0(t_*) + \alpha_{k+1} c_1(t_*) + \alpha_k c_2(t_*) \tag{32}
\]

where \( t_* = \frac{n}{4}(t - t_k) \) and the \( \alpha_k \) are constants for \( k = 0, \ldots, n - 1 \). Due to the periodicity, \( \alpha_{n+k} \) is treated as equivalent to \( \alpha_k \). As in part (a), this form will satisfy that \( s_n^*(t) \) match the
Figure 4: The parametric curve $s(t) = (s_x(t), s_y(t))$ compared to a circle, for (a) four control points and (b) seven control points.

function values at the end of each interval, and will have continuous first derivative. To set the $\alpha_k$, the second derivatives must be considered. Ensuring continuity of the second derivative at $t_k$ yields

$$-6 \sin \frac{2k\pi}{n} + 6 \sin \frac{2(k+1)\pi}{n} - 2\alpha_{k+1} - 4\alpha_k = 6 \sin \frac{2(k-1)\pi}{n} - 6 \sin \frac{2k\pi}{n} + 4\alpha_k + 2\alpha_{k-1}. \tag{33}$$

and hence

$$2\alpha_{k-1} + 8\alpha_k + 2\alpha_{k+1} = -6 \sin \frac{(k-1)\pi}{2n} + 6 \sin \frac{(k+1)\pi}{2n}. \tag{34}$$

By considering Eq. 34 for $k = 0, 1, \ldots, n-1$, a linear system is obtained for the $\alpha_k$, which is the generalization of Eq. 28. The matrix is a circulant matrix, where the only entries are $(2, 8, 2)$ on diagonal lines that wrap around. Linear systems involving such matrices can be solved very rapidly in principle, by employing the fast Fourier transform. Once the $\alpha_k$ are determined, Eq. 32 gives the explicit form of the spline.

In a similar manner, a spline $s^n_y(t)$ of $\cos \frac{\pi t}{2}$ can be constructed. In part (c), this spline was constructed by noting that this function is the same as sine, but shifted by $-1$, which permitted the form of $s^n_y(t)$ to be written down immediately. However, here the same
Figure 5: Relative error $E_{\text{rel}}$ in the approximation of $\pi$ using cubic splines as a function of the number of control points.

The approach will not always work, since for a general $n$ the positions of the control points may not precisely match when shifted by $-1$. It is therefore necessary to explicitly construct this spline, by using Eqs. 32 and 34 again, and replacing all instances of sine by cosine.

The program `am205_so11_ex3e.py` constructs the two splines for an arbitrary number of control points $n$, when $n \geq 3$. Figure 4(b) shows a plot of the resultant circle approximation using seven points, which becomes near-indistinguishable from a true circle. The program also considers a range of different $n$ and computes the approximation to $\pi$. The area integration is performed using three-point Gaussian quadrature\(^\dagger\) on each subinterval, which results in an exact answer up to truncation error. Figure 5 shows a plot of the relative error $E_{\text{rel}}$ in the approximation of $\pi$ as a function of $n$. The relative error is well-fit by the line $E_{\text{rel}} = 4.547n^{-4.007}$ over the range $10 < x < 2000$. It is reasonable for the error to scale like fourth power of $n$. If sine and cosine are being well-fit by cubic polynomials, then the leading term in the approximation error will be quartic in size. For $x > 2000$, truncation error becomes visible, as expected.

\(^\dagger\)This will be explained in detail in Unit 3.
Problem 5 – Fitting a planet’s orbit

The equation of an ellipse is given by \( b_0 + b_1 x + b_2 y + b_3 xy + b_4 y^2 = x^2 \). We aim to solve the overdetermined system \( Ab = x \), which can be written out as

\[
\begin{pmatrix}
1 & x_1 & y_1 & x_1y_1 & y_1^2 \\
1 & x_2 & y_2 & x_2y_2 & y_2^2 \\
. & . & . & . & . \\
1 & x_n & y_n & x_ny_n & y_n^2 \\
\end{pmatrix}
\begin{pmatrix}
b_0 \\
b_1 \\
b_2 \\
b_3 \\
b_4 \\
\end{pmatrix}
= \begin{pmatrix}
x_1^2 \\
x_2^2 \\
. \\
x_n^2 \\
\end{pmatrix}
\]

(35)

where the parameters \( \{b_0, b_1, b_2, b_3, b_4\} \) are the unknowns. In the code `ps1problem5a.m` that is provided, we use the MATLAB built-in function “\" to solve this problem for the original set of data that is given. The best-fit parameters are

\[
b_0 = -0.4329, \quad b_1 = 0.5514, \quad b_2 = 3.2229, \quad b_3 = 0.1436, \quad b_4 = -2.6356.
\]

Figure 6 shows the resulting orbit and data points. The 2-norm of the residual \( r = Ab - x \) is 0.0122.

The code `ps1problem5b.m` calculates the best-fit parameters for the perturbed data, and obtains

\[
b_0 = -0.4598, \quad b_1 = 0.6532, \quad b_2 = 3.1293, \quad b_3 = -0.5191, \quad b_4 = -1.1516.
\]
Figure 7: Best fit to original and perturbed orbit. The green curve shows the perturbed orbit, and the red curve shows the original orbit. While the orbits are very different, the two sets of data points are very similar.

The perturbed and unperturbed orbits are shown in Fig. 7. Even with the tiny perturbation that is provided, we obtain a very different elliptical orbit. This shows that the matrix $A^T A$ is nearly singular.

**Problem 6 – Approximate solution of a partial differential equation using least-squares fitting**

**Part (a)**

We want to interpolate $y(x, 0) = 10^{-4}x^8(\pi - x)^4$ with a non-polynomial basis in the form of $y(x, 0) \approx \sum_{k=1}^{6} a_k \sin kx$. This is an over-constrained problem because there are more interpolation points than there are fitting coefficients. The $A$ matrix is

$$ A = \begin{pmatrix}
\sin \frac{\pi}{50} & \sin \frac{2\pi}{50} & \ldots & \sin \frac{6\pi}{50} \\
\sin \frac{2\pi}{50} & \ldots & \ldots & \\
\ldots & \ldots & \ldots & \\
\sin \frac{49\pi}{50} & \ldots & \ldots & \sin \frac{294\pi}{50}
\end{pmatrix}. $$

(36)
The normal equation has the form of $A^T A \vec{\alpha} = A^T y$, but we do not directly solve this system because of numerical stability. Instead, by using a built-in least-squares algorithm to solve for $b$, we obtain the coefficients

$$\vec{\alpha} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \end{pmatrix} = \begin{pmatrix} 0.0240 \\ -0.0169 \\ -0.0020 \\ 0.0073 \\ -0.0036 \\ 0.0010 \end{pmatrix}. \quad (37)$$

Part (b)

We can write down the approximate solution as $y_{ap}(x, t) = \sum_{k=1}^{6} \alpha_k \sin kx \cos kt$. Substituting in $c = 1$ and expanding we obtain

$$y_{ap}(x, t) = 0.0240 \sin x \cos t - 0.0169 \sin 2x \cos 2t - 0.0020 \sin 3x \cos 3t + 0.0073 \sin 4x \cos 4t - 0.0036 \sin 5x \cos 5t + 0.0010 \sin 6x \cos 6t. \quad (38)$$

Figure 8 shows $y_{ap}(x, t)$ at $t = 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi$. Note that the solution looks symmetric with respect to the $y$-axis, as this simple wave equation does not have any dissipation.
Part (c)

We can also expand $y(x, 0)$ into an infinite sum of Fourier components as

$$y(x, 0) = \sum_{k=1}^{\infty} b_k \sin kx.$$  \hfill (39)

By making use of the orthogonality relations of the basis functions, the $b_k$ are given by

$$b_k = \frac{2}{\pi} \int_0^{\pi} y(x, 0) \sin kx \, dx.$$  \hfill (40)

These coefficients can be computed using either a symbolic or numerical integrator. They are

$$\vec{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix} = \begin{pmatrix} 0.0240 \\ -0.0169 \\ -0.0020 \\ 0.0073 \\ -0.0036 \\ 0.0010 \end{pmatrix}.$$  \hfill (41)

Note that $a_i = b_i$ to within four decimal places of accuracy for all $i$. The Fourier series expansion and interpolation using Fourier basis are therefore very similar.