

ME 201/MTH 281ME400/CHE400 Examples of Bessel Expansions *Mathematica 7*

In this notebook, we construct and plot the partial sums of a Fourier-Bessel series for two specific examples. For a much more general approach to computing and graphing such problems, see the notebook entitled Convergence of Bessel Expansions. In this notebook we use only the functions J_0 . These functions arise as the eigenfunctions of the Sturm-Liouville problem given below.

$$\frac{d}{dr} \left(r \frac{d\psi}{dr} \right) + \lambda r \psi = 0, \quad 0 < r < a, \quad \text{with } \psi \text{ well behaved at } r = 0, \text{ and } \psi(a) = 0.$$

We have $\psi_n(r) = J_0(\alpha_n r/a)$, where α_n is the n th root of J_0 . These eigenfunctions are orthogonal on $[0,a]$ with respect to the weight function r . Any well-behaved function $f(r)$ on $[0,a]$ can be expanded in these eigenfunctions. The coefficients in the expansion are calculated using orthogonality. The result is

$$f(r) = \sum_{n=1}^{\infty} C_n \psi_n(r), \quad \text{where } C_n = \frac{\int_0^a f(r) \psi_n(r) r dr}{\int_0^a \{\psi_n(r)\}^2 r dr}$$

We now illustrate this theory by expanding the function

$$f[r_] := a^2 - r^2$$

We choose the value 3 for a :

$$a = 3;$$

We find the first 21 zeros of J_0 and assign them to the list named α .

```

$$\alpha = \mathbf{N}[\mathbf{Table}[\mathbf{BesselJZero}[0, i], \{i, 1, 21\}]]$$

{2.40483, 5.52008, 8.65373, 11.7915, 14.9309, 18.0711, 21.2116,
24.3525, 27.4935, 30.6346, 33.7758, 36.9171, 40.0584, 43.1998,
46.3412, 49.4826, 52.6241, 55.7655, 58.907, 62.0485, 65.19}
```

We define the eigenfunctions for *Mathematica*.

$$\psi[r_ , n_] := \mathbf{BesselJ}[0, r * \alpha[[n]] / a]$$

We call the n th expansion coefficient $\text{coeff}[[n]]$. The theoretical expression is given above in equation (1). We define expressions for both the numerator and the denominator numerically in terms of the *Mathematica* function $\mathbf{NIntegrate}$. We also obtain the coefficients below from an analytical expression derived in class. With more complicated functions $f[r]$, numerical integration may be the only choice.

```
num[n_] := NIntegrate[f[r] *  $\psi$ [r, n] * r, {r, 0, a}]
```

```
den[n_] := NIntegrate[( $\psi$ [r, n])^2 * r, {r, 0, a}]
```

We calculate the first 21 coefficients, and assign them to a list named coeff.

```
coeff = Module[{ans, c, n}, ans = {};
  Do[{c = num[n] / den[n]; ans = Append[ans, c]}, {n, 1, 21}]; ans]
{9.9722, -1.258, 0.409288, -0.188918, 0.104726, -0.0649906,
 0.0435408, -0.0308311, 0.0227658, -0.0173713, 0.0136099,
 -0.0108969, 0.00888466, -0.00735654, 0.00617251, -0.00523902,
 0.00449182, -0.0038857, 0.00338819, -0.00297548, 0.00262987}
```

In class, we also derived an analytical expression for these coefficients. Let's verify that we get the same result that way.

```
analccoeff[n_] := (8 a^2) / (( $\alpha$ [n])^3 BesselJ[1,  $\alpha$ [n]])
```

```
analycoefflist = Module[{ans, c, n}, ans = {};
  Do[{c = analcoeff[n]; ans = Append[ans, c]}, {n, 1, 21}]; ans]
{9.9722, -1.258, 0.409288, -0.188918, 0.104726, -0.0649906,
 0.0435408, -0.0308311, 0.0227658, -0.0173713, 0.0136099,
 -0.0108969, 0.00888466, -0.00735654, 0.00617251, -0.00523902,
 0.00449182, -0.0038857, 0.00338819, -0.00297548, 0.00262987}
```

We get complete agreement, so the two calculations reinforce one another. Note that the fourth coefficient is only about 2% of the first coefficient, and the rest are even smaller. This suggests rapid convergence, which we are about to verify with some graphs. This is not so surprising given that the function being expanded satisfies the same zero boundary condition at the edge as the eigenfunctions.

Now we define the kth partial sum of our Fourier-Bessel series.

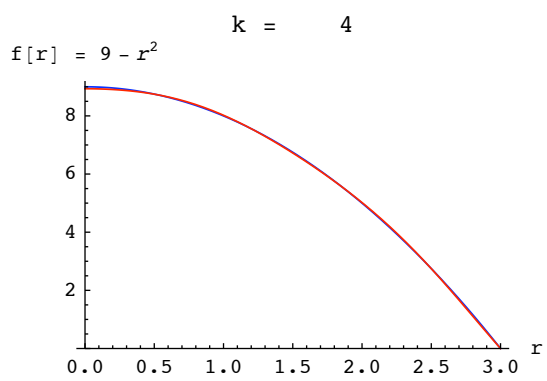
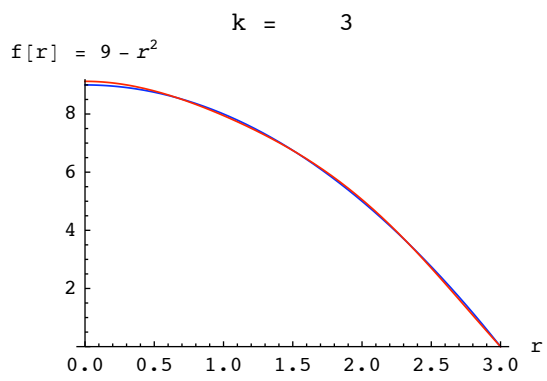
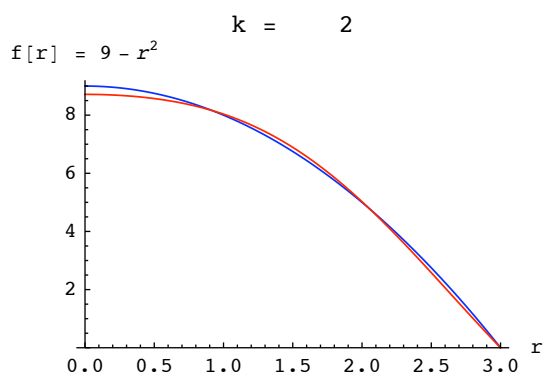
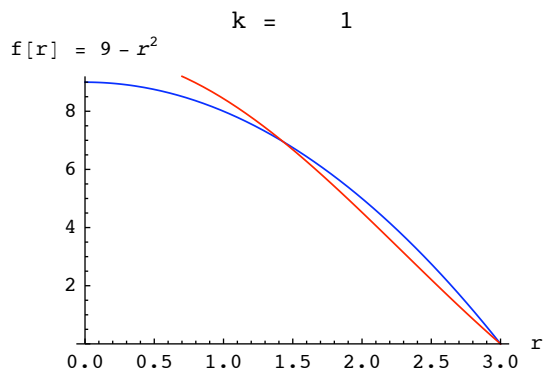
```
fourbess[r_, k_] := Sum[coeff[[n]] *  $\psi$ [r, n], {n, 1, k}]
```

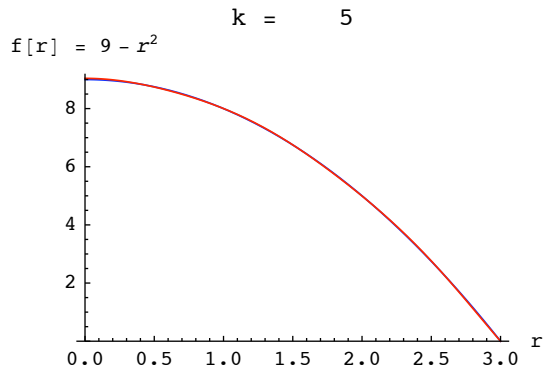
Finally, we define graph[k], which produces a graph of f[r] in blue and the kth partial sum of the Fourier-Bessel series in red. We use pltrange as a variable containing the plot range. We use SetOptions to set the value of the plotting option ImageSize. We set it to 200, a good value for printing. For computer visualization, a value of 350 is better.

```
pltrange = {0, 9.2};
SetOptions[Plot, ImageSize -> 200];
graph[k_] :=
Plot[{f[r], fourbess[r, k]}, {r, 0, a}, PlotRange -> pltrange,
  AxesLabel -> {"r", SequenceForm["f[r] = ", f[r]]},
  PlotLabel -> Row[{"k = ", PaddedForm[k, 3]}],
  PlotStyle -> {{RGBColor[0, 0, 1], Thickness[0.004]},
    {RGBColor[1, 0, 0], Thickness[0.004]}}
```

Now we look at the first 5 partial sums.

```
Do[Print[graph[k]], {k, 1, 5}];
```





The graphs verify our prediction of good convergence.

Let's do one more example with a different function. We must re-define f , and then re-execute the calculation of the coefficients. We choose f to be a constant.

```
f[r_] := 1

coeff = Module[{ans, c}, ans = {};
  Do[{c = num[n] / den[n]; ans = Append[ans, c]}, {n, 1, 21}]; ans]

{1.60197, -1.0648, 0.851399, -0.729645, 0.648524,
-0.589543, 0.54418, -0.507894, 0.478012, -0.452851,
0.431284, -0.412531, 0.396028, -0.38136, 0.368208, -0.35633,
0.345532, -0.335659, 0.326587, -0.318213, 0.310451}
```

Again we had an analytic expression for these coefficients in class, so again we use those in a consistency check.

```
analcoeff[n_] := 2 / (α[[n]] BesselJ[1, α[[n]])]

analycoefflist = Module[{ans, c}, ans = {};
  Do[{c = analcoeff[n]; ans = Append[ans, c]}, {n, 1, 21}]; ans]

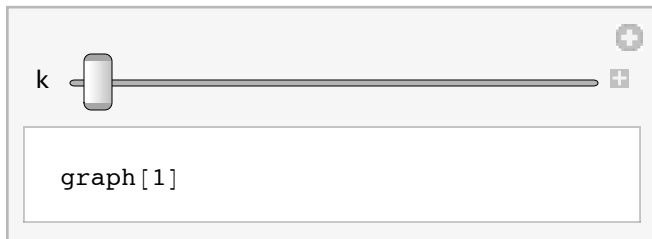
{1.60197, -1.0648, 0.851399, -0.729645, 0.648524,
-0.589543, 0.54418, -0.507894, 0.478012, -0.452851,
0.431284, -0.412531, 0.396028, -0.38136, 0.368208, -0.35633,
0.345532, -0.335659, 0.326587, -0.318213, 0.310451}
```

Same results.

We don't expect the convergence to be as good here, because the function doesn't vanish at the endpoints. Let's do 21 partial sums for this one. We use the command `Manipulate` to produce this graph sequence.

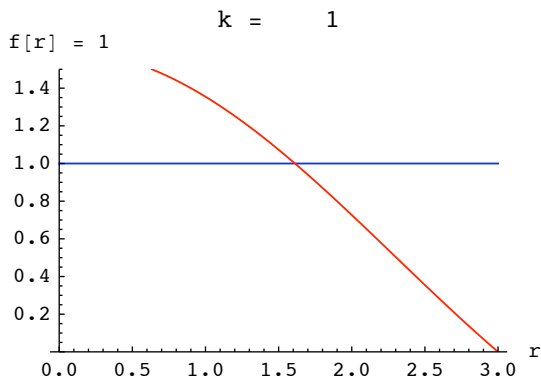
```
pltrange = {0, 1.5};
```

```
Manipulate[graph[k], {k, 1, 21, 1}]
```



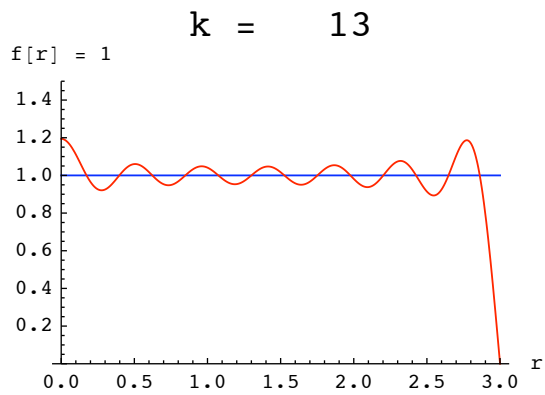
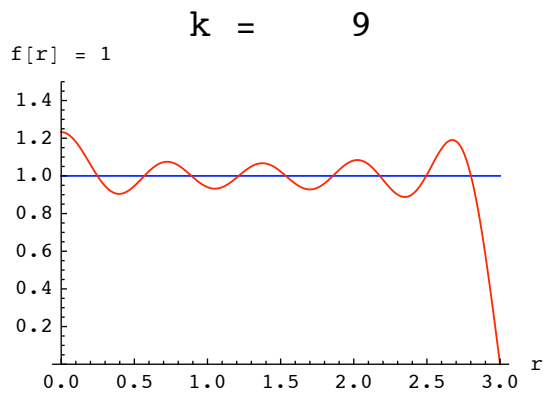
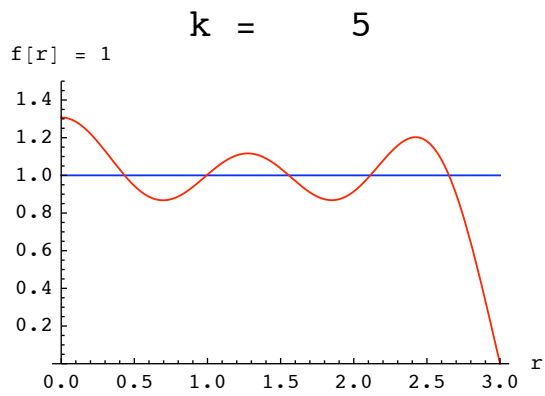
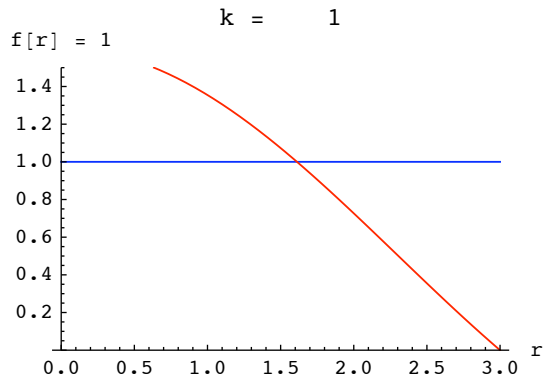
The command `Manipulate` gives us a very easy way to produce a graph sequence. You can use the slider to look quickly at the graphs in the sequence, and this gives a good idea of how the convergence proceeds. If you click on the little plus sign at the end of the slider, you will get controls for displaying the graphs as a movie. This gives a nice dynamic view of the convergence process, but it is also limited. You notice that the movie is rather jerky. The problem is that there is a lot of computation behind the later graphs in the series, and so the time interval between the later graphs is longer than that between the earlier graphs. In a later notebook we will see a better computational scheme which will give us better results. We can also get better results here by using a `Do` loop to print out all of the graphs, and then animating those graphs. We start with the `Do` loop. After the graphs are produced we will collect them in a single cell, so you will see only the first graph. You can animate the sequence by first selecting it, and then going to the menu item **Graphics->Rendering->Animate Selected Graphics**. The movie produced in that way is very smooth.

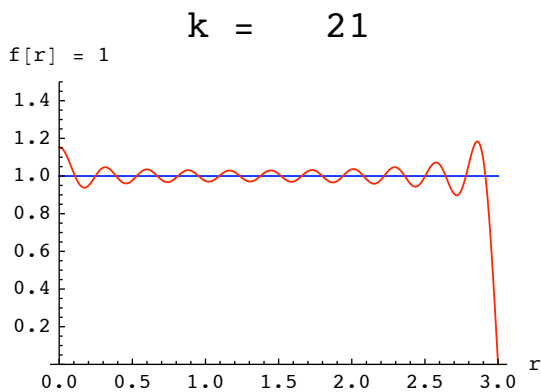
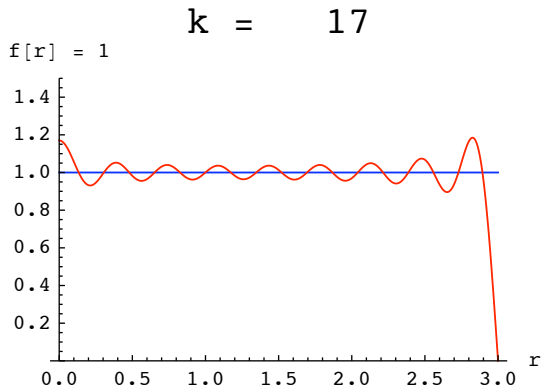
```
Do[Print[graph[k]], {k, 1, 21}]
```



For visualization in the printed version of the notebook, we construct and display every fourth graph in this sequence.

```
Do[Print[graph[k]], {k, 1, 21, 4}];
```





The series is clearly converging, but the struggle is very reminiscent of the Fourier series for a square wave, complete with the Gibbs phenomenon. The resemblance is more than accidental. Remember that the Bessel functions for large argument (i.e., large n in the series) behave like damped trig functions, so the convergence issues are mathematically related to those of the Fourier series.

The above calculation has been very inefficient. In the sequence of 21 partial sums, we re-computed at each step all of the terms that appeared in the preceding graph, plus one new term. It would be far more efficient to save the n th partial sum, and then create the $n+1$ st partial sum by computing only the new term and adding it to the n th partial sum. This is exactly what is done in the notebook *Convergence of Bessel Expansions*. It is far more efficient, but the coding is not as easy to follow.